UNIVERSIDAD NACIONAL DE LA PLATA
FACULTAD DE CIENCIAS EXACTAS

TRABAJO DE GRADO

MICROCOSMO

SISTEMA DE HIPERMEDIA ABIERTO

DIRECCION: PROF. GUSTAVO ROSSI

CASTELLANO, Mariana Carla – 26029/7
FABIANO, Sandra Beatriz – 26230/6
SUCARELLI, Adriana Marcela – 26330/9

AÑO 1997/1998

SISTEMA DE HIPERMEDIA ABIERTO

INDICE

1. CONCEPTOS GENERALES

1.1. MULTIMEDIA

Son aquellas aplicaciones que combinan distintos medios tales como: texto, imágenes, video, animación y sonido para proveer los modos de visualizar e interactuar con información en forma entendible.

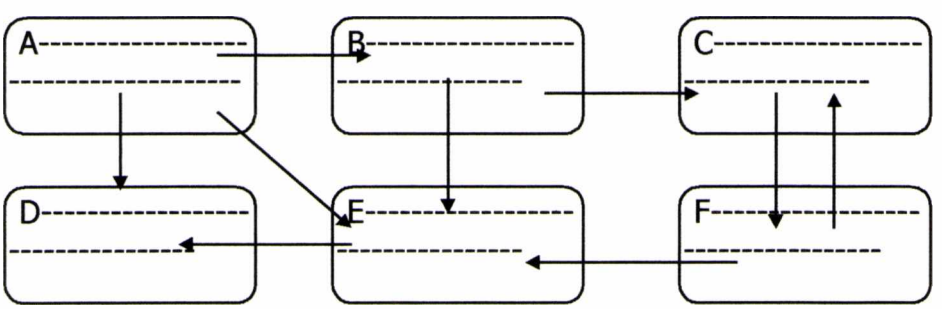
Con el uso de la multimedia se incrementa la posibilidad de obtener información desde la computadora, se mejora la calidad de la información y representación de la misma.

Se logra una interacción con el usuario más atractiva permitiendo un fácil uso para aquellos no experimentados.

1.2. HIPERTEXTO

Todo texto tradicional es secuencial, lo que significa que es un secuencia lineal, simple que define el orden en el cual el texto es leído.

El hipertexto es no secuencial. No hay un orden particular que determine la secuencia en que el texto sea leído. La siguiente figura da un ejemplo :



Se comienza leyendo el texto A, la estructura muestra que a partir de aquí hay tres opciones donde ir: B, D ó E. Si se decide ir a B luego se puede decidir ir a C ó a E y de E se puede ir a D. También es posible ir directamente de A a D. Este ejemplo muestra que hay diferentes caminos que conectan dos elementos en una estructura de hipertexto.

Como se ve en la figura el hipertexto consiste de piezas de textos interenlazadas. Cada una de estas piezas de información es llamada *nodo* que pueden ser pantallas, ventanas, archivos o bits de información más pequeñas.

Cualquiera sea el tamaño de estos nodos, cada uno de ellos pueden tener punteros a otras unidades. Estos punteros son llamados *enlaces* o *links*.

La figura también muestra que la estructura de hipertexto entera forma una red de nodos y enlaces. El movimiento a través de esta red es una actividad

referenciada como recorrido o navegación más que "lectura". El usuario determina activamente el orden en el que leerá los nodos.

Los enlaces de hipertexto están asociados con partes específicas de los nodos a los que están conectados, más que con los nodos en su totalidad.

Si el usuario requiere retornar a nodos ya visitados, lo hace a través de una facilidad de backtrack (volver atrás). Cada ejecución del backtrack lo retorna al nodo anterior inmediato.

Los hipertextos permiten organizar gran cantidad de información en fragmentos, los que se relacionan entre sí. El usuario solo necesita una parte de la información por vez.

1.2.1. Usabilidad del hipertexto

1- **Fácil de aprender:** El hipertexto en sí mismo es fácil de aprender. Los usuarios son capaces de aprender rápidamente los comandos básicos y las opciones de navegación y los usan para localizar la información buscada.

2- **De uso eficiente:** Una vez que el usuario ha aprendido el sistema se obtiene un alto nivel de productividad.

3- **Fácil de recordar:** Después de un período que no se ha usado el hipertexto el usuario no tiene problemas en recordar cómo usar y navegar en el hipertexto.

4- **Pocos errores:** Los usuarios no cometen muchos errores durante el uso del sistema, si los cometen pueden recuperarlos fácilmente.

5- **De uso amigable:** Los usuarios quedan satisfechos con el uso del hipertexto ya que sienten que tienen el control sobre él y que pueden moverse libremente por el sistema.

Entre las aplicaciones en las que se usa hipertexto podemos mencionar:

- Procesamiento de texto.
- Ayudas para el usuario.
- Enciclopedias dinámicas.
- Guías turísticas y museos.
- Conferencias interactivas.
- Aplicaciones educativas.

1.3. HIPERMEDIA

Los documentos de hipermedia contienen enlaces no sólo a otras partes de texto, como en el caso de hipertexto sino también a otras formas de media: sonido, dibujos, movimiento, animaciones, etc.

El destino de un enlace es un nodo. Los dibujos pueden también ser usados para enlazar a otros nodos. Hipermedia simplemente combina hipertexto y multimedia. Esto permite a los autores conectar información de diferentes maneras, permitiendo a los usuarios explorar a través de una compleja red de información interenlazada de acuerdo a sus necesidades e intereses.

1.4. SISTEMAS DE HIPERMEDIA

Los sistemas de hipermedia se caracterizan por tener una arquitectura modular. Esta modularidad está basada en que la performance de las funciones son independientes una de la otra. Al ser modular permite al usuario elegir entre distintas opciones para recorrerlo.

2. SISTEMAS DE HIPERMEDIA ABIERTO

2.1. EVOLUCION:

Los usuarios de PC realizan actividades que producen gran cantidad de datos. Para navegar a través de estos datos los usuarios usan como método principal la estructura de directorio de archivos.

Alrededor del año 1992, las técnicas de hipermedia para enlazar y navegar a través de la información demandaba un alto costo, ya que para hacer uso de un documento en un sistema de hipermedia, éste primero debía ser importado al sistema. Cuando los enlaces eran agregados al sistema el dato que representaba a ese enlace era almacenado en el archivo como alguna forma de mark-up. El dato estaba, a partir de ese momento, en un sistema cerrado. No era posible procesar el dato con el paquete que lo había creado ya que éste estaba almacenado en un formato privado para el sistema de hipermedia.

Mientras los sistemas de hipermedia permanecían cerrados estaban restringidos a aplicaciones donde los datos permanecían relativamente estáticos. La nueva generación de hipermedia debía aparecer al usuario como una facilidad del sistema operativo que está permanentemente disponible para agregar facilidades de enlace y navegación de información con la mínima cantidad de intervención del usuario y sin perder alguna funcionalidad que estaba previamente disponible.

Resumiendo, los problemas que surgían al ser los sistemas de hipermedia cerrados eran:

- Almacenar la información de los enlaces en el dato.
- Requerir que los datos sean importados en un formato propio del sistema.
- No permitir crear enlaces a datos almacenados en otras aplicaciones.
- Crear aplicaciones que son estáticas.
- Tener alto costo de creación, actualización y mantenimiento de datos y enlaces.
- Estar limitados a importar formatos de datos particulares.

Los requisitos generales de los usuarios para un sistema de hipermedia abierto son:

- Un ambiente adaptable para la integración de datos, herramientas y servicios.
  Los sistemas cerrados que mantienen el dato en un formato privado también deben proveer herramientas para acceder y procesar ese dato. Por ejemplo INTERMEDIA provee aplicaciones como INTERTEXT e INTERDRAW. Esta es una solución cerrada. Los usuarios no quieren estar limitados a utilizar un determinado paquete de software. Los sistemas de hipermedia deben permitir a

los usuarios crear datos en cualquier paquete que ellos elijan y hacer enlaces desde un dato a otro, o crear anchors en esas aplicaciones.

- Un sistema que es independiente de la plataforma y es distribuido a través de plataformas.
  Los usuarios desearán enlazar datos y aplicaciones en una máquina a datos y aplicaciones en otras. Ellos querrán que esas operaciones sean transparentes. Esta es una propiedad en el dominio de los sistemas operativos distribuidos. Sin embargo la funcionalidad de hipermedia debe ser distribuida a través del sistema operativo y la información del enlace debe ser portable a través de plataformas de hardware.

- Un sistema que permita a los usuarios encontrar, actualizar, anotar e intercambiar información fácilmente.
  Si los usuarios tienden a usar sistemas de hipermedia, entonces el sistema debe agregar ayuda navegacional para que su uso sea más fácil. Debe ser posible cambiar el dato usando las herramientas disponibles en el sistema y agregar nueva información y anotaciones para uso público y privado.
  Cuando los cambios son hechos a información pública se debe notificar a los usuarios de esos cambios. Debe haber una noción de espacio de trabajo público y privado y debe ser posible mover información (incluyendo la información de los enlaces) entre esos espacios de trabajo.

- Un sistema en el que todos los formatos de datos y media son tratados en una forma conceptualmente similar.
  Si el proceso de creación de enlaces en un paquete tiene una interface distinta de otro, entonces no será fácil para los usuarios aprender el uso de todos los beneficios y medias.

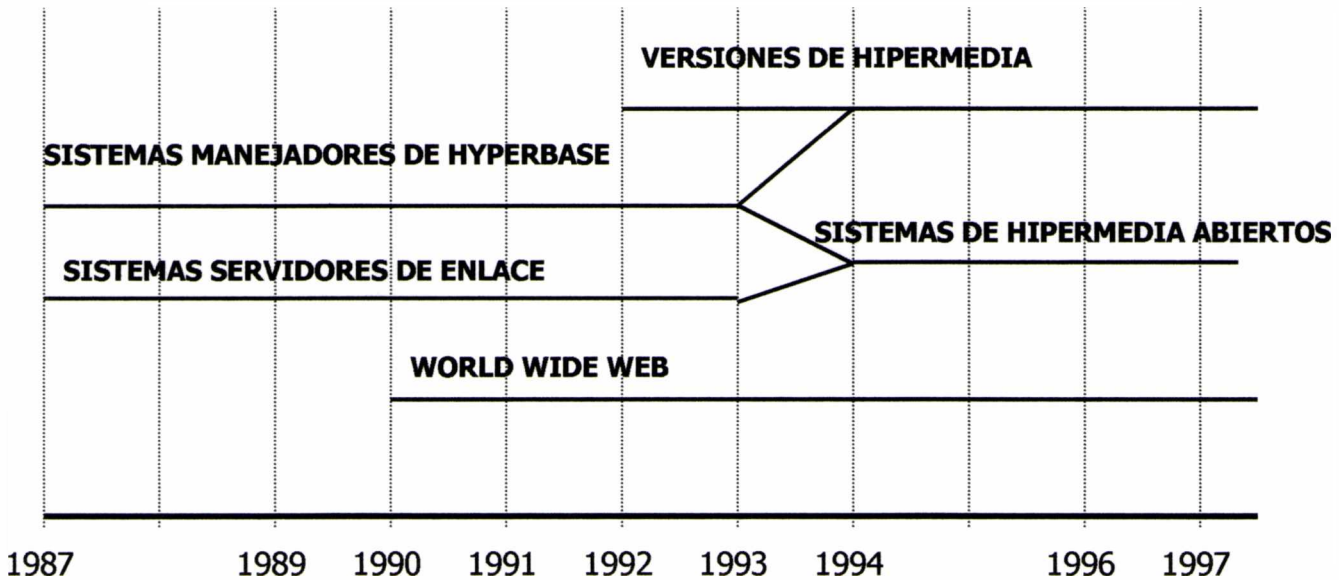
Los conceptos de hipermedia actualmente se desarrollan en distintos sistema de información tales como la WWW, ambientes de desarrollo de software, empresas de ingeniería, sistemas de creación colaborativa y sistemas de librería digital.

La investigación actual sobre los Sistemas de Hipermedia Abiertos (OHS) tiene sus raíces en dos grupos de investigación totalmente independientes: Sistemas Servidores de Enlaces (LSS) y Sistemas Manejadores de Hiperbase (HBMS).

Los LSS desarrollan componentes que proveen funcionalidad de enlace de hipermedia. Proporcionan enlaces de hipermedia **a** y **desde** la información manejada por las herramientas existentes en el ambiente de computación sin alterar la información misma.

Los HBMS, además de proveer funcionalidad de enlaces de hipermedia, provee soporte para almacenamiento.

A partir de 1994 la gente de las áreas HBMS y LSS se unió y formó la comunidad OHS (Sistemas de Hipermedia Abiertos).



2.2. REQUISITOS PARA LOS SISTEMAS DE HIPERMEDIA ABIERTOS:

Hay distintos enfoques en cuanto a la definición de sistemas abiertos. Algunos consideran a un sistema abierto si puede ser distribuido a través de una red. Otros dicen que es abierto si hay detalles publicados de cómo agregar y extender el sistema. En el campo de hipermedia los sistemas han sido calificados como abiertos por el hecho de que el lector tiene los mismos derechos de acceso que el autor. Para que un sistema de hipermedia sea considerado abierto debe cumplir como mínimo con los siguientes factores:

1. Un sistema que no imponga ninguna restricción sobre los datos para que estos sean accedidos por otros procesos que no pertenecen al sistema.

2. Un sistema que pueda integrarse con cualquier herramienta que corra bajo el sistema operativo del host.

3. Los datos producidos por herramientas que no son parte del sistema pueden ser usados en el sistema sin agregar ningún valor especial a este dato y sin comprometer el uso del dato fuera del sistema.

4. Un sistema en el que los datos y procesos puedan ser distribuidos a través de una red y a través de plataformas de hardware.

5. Un sistema en el que no hay distinción entre lectores y autores.

6. Un sistema en el que es posible agregar nueva funcionalidad, por ejemplo insertar fácilmente nuevos módulos de programa.

Los sistemas deben ser capaces de hacer enlaces a y desde datos producidos por otras herramientas de procesamiento de información y deben proveer herramientas para generar y mantener enlaces automáticamente.

Los sistemas de hipermedia abiertos son capaces de actuar como un servicio de enlace a un conjunto de aplicaciones existentes.

Microcosmo es un ejemplo de sistema de hipermedia que cumple con las mencionadas características.

Cualquier sistema que intenta integrar aplicaciones debe ser un sistema de hipertexto abierto, es decir extensible, adaptable y poseer alguna interface pública por medio de la cual las viewers puedan comunicarse con el servicio de enlace. Por ejemplo en Multicard esto es alcanzado usando Scripts y en Microcosmo usando pasaje de mensaje.

Se han identificado varios niveles para integrar aplicaciones como viewers de datos para sistemas de hipermedia. Ellos son:

1. Tailor made viewers (viewers hechas a medida): estas aplicaciones son escritas específicamente para la integración con el sistema de hipermedia.

2. Source code adaptation (adaptación del código fuente): si el código fuente de una aplicación está disponible, entonces es posible agregar el código para lograr la comunicación con el servicio de enlace del sistema de hipermedia.

3. Object oriented re-use (reuso orientado a objeto): se crea una clase de viewers de hipertexto básica y las viewers para tipos de datos específicos heredan de esta clase.

4. Application interface level adaptation (adaptación del nivel de interface de aplicación): muchas aplicaciones proveen interface flexible y lenguaje de programación de macros, con lo cual es posible agregar funcionalidad de hipermedia.

5. Shim or proxy programs: son programas que están situados entre el servicio de enlace de hipermedia y las viewers, traduciendo acciones en un sistema a acciones que el otro pueda entender. Un ejemplo es la Viewer Universal de Microcosmo.

6. Launch only viewers: es el caso en el que el sistema de hipermedia puede abrir un programa dado con un conjunto de datos dado, pero desde el programa no habrá funcionalidad de hipertexto. Este es el peor caso.

2.3. EJEMPLOS DE SISTEMAS DE HIPERMEDIA ABIERTOS:

Algunos ejemplos de Sistemas de Hipermedia Abiertos son:

- Microcosmo
- DeVise Hipermedia
- HyperDisco
- Hyperform
- Chimera
- SPN/HBN
- Hoss
- Multicard
- Sun´s Link Service

Desarrollaremos brevemente los conceptos y arquitectura del Sistema Chimera y profundizaremos los conceptos, arquitectura y funcionamiento del Sistema Microcosmo.

2.4. CONCEPTOS DE HIPERMEDIA

2.4.1. *Anchors* :

Es el punto de comienzo y punto de fin de un enlace y puede ser una o más palabras, una región de un gráfico o el comienzo de un documento.

Un anchor identifica parte de contenido de un nodo, tal como una declaración de función en un módulo de programa o una definición en un glosario.

2.4.2.*Enlaces* :

Un enlace es un conjunto de anchors. Existen diferentes tipos de enlaces:

- ✓ *Enlaces específicos*: el anchor origen (invisible) es una selección particular en un documento y el anchor destino es una selección particular en otro o en el mismo documento.

- ✓ *Enlaces locales:* el anchor origen (invisible) es algún punto en el documento actual. El destino es alguna selección en algún lugar en otro o en el mismo documento. Puede haber distintos enlaces locales establecidos desde un anchor origen dado. Por ejemplo supongamos que la palabra CASA es seleccionada como anchor origen y el enlace se hace a la figura de una casa. Entonces todas las apariciones de la palabra CASA en el documento origen (solamente) son enlazadas a la figura.

- ✓ *Enlaces genéricos:* el anchor origen (invisible) es algún punto particular en algún documento. El anchor destino es una selección particular en el documento destino. Los enlaces genéricos ofrecen grandes ventajas de productividad ya que por el costo de un enlace particular muchos enlaces están disponibles. Además cuando nuevos documentos son introducidos inmediatamente tienen acceso a todos los enlaces genéricos que han sido previamente definidos. Si para el ejemplo anterior el enlace se define como genérico, todas las

apariciones de la palabra CASA en cualquier documento son enlazadas a la figura.

- ✓ *Enlaces dinámicos:* (por ejemplo computed links) no todos los enlaces necesitan ser creados explícitamente por el autor. Por ejemplo las relaciones entre piezas de información pueden ser computadas basadas en el análisis estadístico del contenido como en el caso del computed linker, o la información relacionada puede ser recuperada basada en códigos de atributos de documentos.

- ✓ *Botón:* es un enlace específico pero con la diferencia de que el anchor origen es visible.

2.4.3. *Viewers*:

Son programas que le permiten al usuario ver un documento en su formato natural. La tarea de las viewers es permitir al usuario examinar el documento, hacer selecciones y elegir acciones. Las acciones típicas son: follow link, start link y end link (donde los enlaces pueden ser a procesos o a documentos). Las acciones mismas no son efectuadas por la viewers.

2.4.4. *Views*:

Es un par **(v,o)** en donde **v** es una viewer para un objeto **o** . Un objeto puede ser mostrado por más de una viewer y por lo tanto participar en múltiples views. Las views contienen anchors.

2.5. CHIMERA: SISTEMA DE HIPERMEDIA ABIERTO

Los ambientes de desarrollo de software se caracterizan por la heterogeneidad: ellos están compuestos por diversos almacenamientos de objetos, interfaces de usuarios y herramientas. El sistema Chimera provee servicios de hipertexto en este ambiente heterogéneo, lo cual incluye anchors establecidos con respecto a las views interactivas de los objetos, más que a los objetos mismos. Pueden ser establecidos links n-arios entre anchors de diferentes vistas interactivas de objetos almacenados en distintas bases de objetos. Las vistas pueden ser implementadas en distintos lenguajes de programación provistos por una arquitectura cliente-servidor.

2.5.1. Introducción::

Los ambientes de desarrollo de software (SDE) son usados para desarrollar y mantener diversas colecciones de objetos de softwares muy interrelacionados. Los sistemas de software pueden consistir de múltiples versiones de especificaciones de requerimientos, diseños, prototipos, código, testeo de información, scripts y documentación. Las conexiones entre estos componentes son muchas y muy complejas.

Los sistemas de hipertexto permiten capturar y visualizar estas relaciones. Aportan una ingeniería que permite asociar objetos libremente sin considerar el tipo de esos objetos o donde están almacenados. Luego se puede acceder a estas relaciones a través de una interface del usuario permitiendo una fácil navegación a través de la información.

Editor de objetos heterogéneos y soporte de viewer: Los SDE contienen una amplia variedad de herramientas para desarrollar y manipular objetos. Se usan distintos tipos de editores para distintos tipos de objetos. Los SDE incluyen múltiples viewers de objetos, donde cada viewer presenta distintos aspectos del objeto.

Anchors especializados para vistas particulares: Dado que distintas viewers de un mismo objeto pueden presentar distintas descripciones o una viewer puede presentar una descripción de información sintetizada de distintos objetos, los anchors se ven más naturalmente asociados con vistas que con objetos.

Display activos, concurrentes y múltiples vistas: Es un sistema que provee la capacidad de presentar muchas vistas simultáneamente, donde distintas vistas pueden ser del mismo objeto y las acciones en las vistas pueden ser autónomas y concurrentes.

Enlaces a través de manejadores de objetos heterogéneos: Los SDE manejan amplia variedad de objetos, de distintas herencias y tipos. Estos están comenzando

a soportar manejadores de objetos heterogéneos. Esto es esencial para establecer enlaces entre objetos manejados por distintas fuentes.

Especificaciones de acción en anchors y enlaces: Dado que muchos usuarios colaboran en un proyecto usando un SDE, es de gran utilidad proveer acciones programables sobre anchors y enlaces.

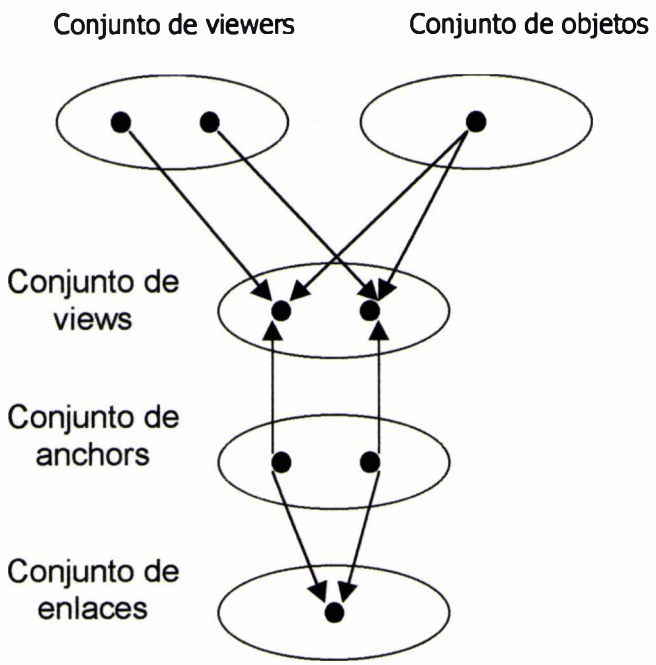
Enlaces escalables: Dos herramientas claves que los ingenieros emplean en problemas de gran escala son la jerarquía y abstracción. El soporte de hipertexto para los SDEs debe proveer dichas herramientas para tratar con un volumen de información grande y complejo.

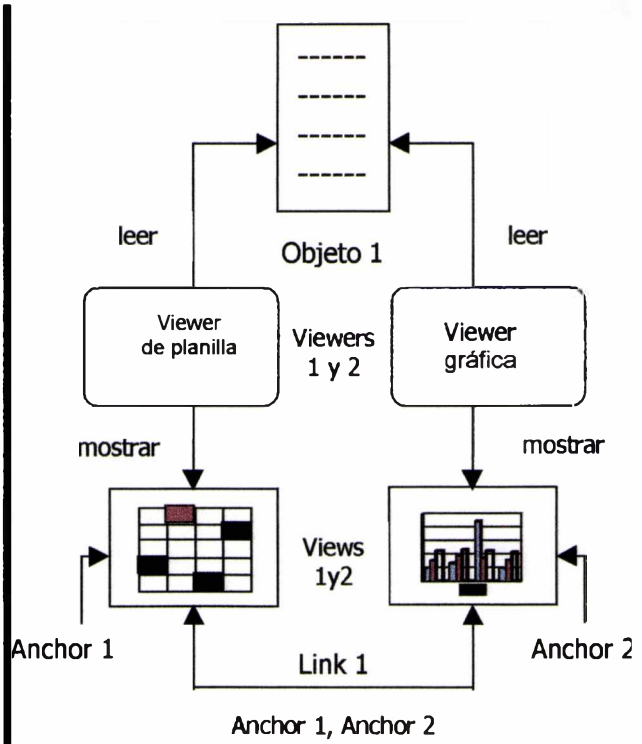
***Enlaces n-arios*:** El desarrollo de software involucra situaciones donde varias piezas de la información representan un concepto particular o son agrupadas en el mismo sentido. El soporte de hipertexto para aplicaciones SDE proveerá dichas capacidades en la forma de enlaces n-arios.

Contextos múltiples: Los procesos de desarrollo de software involucran muchos estados donde algunos tipos de información son más importantes que otros. El soporte de hipertexto para los SDEs será capaz de tener en cuenta el estado actual del proceso de desarrollo de software y permitir el acceso rápido a la información crítica.

El concepto de viewers es de fundamental importancia en estos sistemas. Las viewers en el SDE deben ser programadas para utilizar la interface del programa de aplicación del sistema de hipertexto (API). Las viewers son responsables del mantenimiento de asociaciones que se crean entre los anchors, ellas anuncian al sistema de hipertexto y a los objetos o procesos (por ejemplo un botón en una ventana) qué es lo que mostrarán en sus views.

Los ambientes heterogéneos son frecuentemente multilenguaje y distribuidos, la arquitectura genérica está basada en cliente-servidor y se utiliza un mecanismo de llamada a procedimientos remotos multilenguaje (RPC). La implementación del prototipo de Chimera utiliza sistema manejadores de objetos





Ejemplo del concepto de Chimera: los conceptos de hipertexto de Chimera son mostrados en la izquierda. Se combinan dos viewers con un objeto para producir dos vistas distintas. Un anchor es agregado para cada view y se combinan en un enlace. En la derecha, una hyperweb presenta un archivo de datos (almacenado como un archivo en el sistema operativo) siendo mostrado por dos viewers diferentes. Una viewer muestra el dato como una planilla, creando una view de la planilla del archivo de datos. La otra viewer muestra el dato como un gráfico, creando una view de gráfico del mismo dato. El anchor es indicado por una celda negra en la planilla y por una línea negra en el gráfico. Los anchors son almacenados en la base de datos de Chimera, no en el archivo de datos. Los dos anchors son miembros del enlace

2.5.2. Conceptos de Hipertexto:

Chimera tiene un conjunto flexible de conceptos de hipertexto que mapea en el dominio de ambientes de desarrollo de software.

2.5.2.1 Objetos: son nominados, entidades persistentes cuya estructura interna es desconocida e irrelevante para Chimera.

2.5.2.2. Viewers: son entidades activas nominadas que muestran objetos. Las operaciones provistas por una viewer son específicas de la viewer y el tipo de objeto que ella muestra. Típicamente proveen funcionalidad de browsing, creación y edición sobre objetos en su dominio.

2.5.2.3. Views: denota un par (v,o) donde **v** es una viewer para un objeto **o**. Un objeto puede ser mostrado por más de una viewer, y por lo tanto participar en múltiples views. Las views contienen anchors.

2.5.2.4. Anchors: son definidos y manejados por viewers en el contexto de una view. Los anchors están hechos por una viewers de la view particular del objeto que está siendo mostrado. A diferencia de los sistemas de hipertexto que requieren anchors directos a objetos mapeados con anchors incorporados en los objetos mismos, los anchors de Chimera pueden representar algún componente visual (botón u otro elemento de interface).

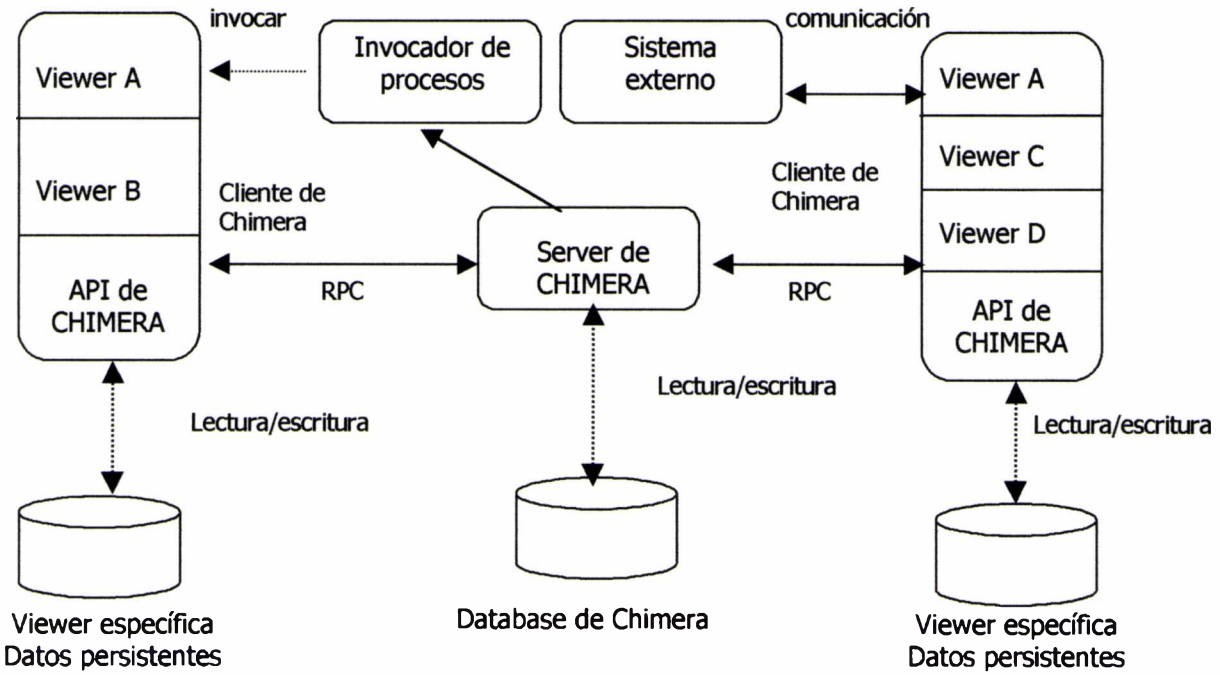
2.5.2.5. Links: es un conjunto de anchors. Los enlaces relacionan porciones de views. En Chimera los enlaces son objetos primera clase y una viewer puede ser construida para mostrarlos. Los anchors pueden ser creados sobre sus vistas de enlaces e incluidos en otros enlaces. Chimera soporta "enlaces a enlaces".

2.5.2.6. Pares atributo – valor: cada instancia de un concepto de hipertexto de Chimera puede tener un número arbitrario de pares atributo – valor asociados con ella. Un par atributo – valor consiste de dos strings asociados, donde un string contiene el nombre del atributo y el otro su valor.

2.5.2.7. Hyperwebs: En Chimera es una colección de objetos, viewers, views, anchors y enlaces, con sus atributos.

2.5.3. La Arquitectura conceptual de Chimera:

Esta arquitectura adopta una aproximación cliente – servidor para proveer servicios de hipertexto y colaborar en los SDE heterogéneos en donde intervienen muchos usuarios.



Arquitectura conceptual de Chimera: Los clientes Chimera consisten de una o más viewers que acceden al server de Chimera para proveer servicios de hipertextos a sus usuarios. No hay restricciones en el número de clientes, el número de viewers por cliente y la misma viewer puede aparecer en múltiples clientes. El invocador de procesos puede llamar a clientes de Chimera o ser dirigido por el server de Chimera.
Los clientes de Chimera pueden además acceder a sistemas externos al ambiente, tal como server de gráficos o server de sonidos. Todas las entidades son procesos Unix separados y pueden leer y escribir información a un sistema de archivo o manejador de objetos.

Esta aproximación cliente – servidor permite que múltiples usuarios en distintas máquinas accedan a una hyperweb desde un conjunto de viewers que cambian dinámicamente; los eventos de hipertexto son originados en una viewers y propagados a otros, vía el server. El uso de un server soporta una aproximación multilenguaje donde los clientes pueden estar implementados en distintos lenguajes de programación, cada uno de ellos accediendo al server a través de su API específico del lenguaje. El uso de un server también mantiene procesos y archivos de objetos de tamaños pequeños, ya que el código para manejar una hyperweb está centralizado en el server.

Los componentes de la arquitectura de Chimera son:

2.5.3.1. Server de Chimera: Las responsabilidades principales del server de Chimera son:

- Proveer servicios que permitan a los clientes acceder a conceptos de hipertexto de Chimera.

- Manejar la persistencia de una hyperweb. A través del uso del manejador de objetos, el server de Chimera debe mapear instancias de conceptos de hipertexto de Chimera en un almacenamiento persistente. El server de Chimera no es responsable del almacenamiento persistente de datos objetos del cliente.

- Recibe, rutea y genera eventos de hipertexto, el server también debe proveer una forma para que los clientes registren interés en un subconjunto de esos eventos.

- Maneja cada uno de los clientes conectados. Por ejemplo, el server Chimera conocerá que viewers se están ejecutando, que view de cada viewer está activa y si la viewer está lista para recibir eventos de hipertexto.

2.5.3.2. Invocador de procesos: Es responsable de invocar a los Clientes de Chimera. Esto ocurre cuando el server de Chimera determina la necesidad de enviar un evento de hipertexto a una viewer que no está ejecutándose. El invocador de procesos mantiene un mapeo entre nombres de viewers y programas ejecutables. Cuando el server de Chimera envía al invocador de procesos un nombre de viewers, éste llama al programa ejecutable a través de los servicios del sistema operativo.

2.5.3.3. Cliente de Chimera: incluye una o más viewers y las librerías necesarias para comunicarse con un server de Chimera. Cada viewer de un cliente es responsable de lo siguiente:

- Definición de los conceptos "objeto" y "views". Cada viewer debe determinar el significado preciso de estos conceptos en su propio contexto. Esto puede generar ambigüedad con respecto a las versiones del objeto.

- Definición del concepto "anchor". Esto incluye identificar que elementos de una view pueden tener anchors vinculados a ellos, cómo estos anchors son creados y borrados, cómo es indicada la presencia de un anchor y qué atributos pueden ser asignados a un anchor. Como cada viewer puede elegir implementar esta funcionalidad en distintas formas no puede ser garantizado un estilo de interacción uniforme.

- Una función de mapeo desde un identificador de anchor (recibido desde el server de Chimera en el momento en que el anchor es creado) en una región u objeto específico de su pantalla.

- Un manejador de eventos que responderá a eventos de hipertexto desde el server de Chimera.

- Comunicarse con el server de Chimera. Esto incluye registrar eventos de hipertexto, indicar su view actual (que puede cambiar si la viewer es pedida por su usuario para mostrar un objeto distinto), acceder a los servicios relacionados con los conceptos de hipertexto de Chimera, y hacer que el server de Chimera conozca que está listo para recibir eventos de hipertexto (esto le da tiempo para que un cliente invocado nuevamente se prepare antes que el server envíe algún evento de hipertexto)

- Proveer un mecanismo para almacenamiento persistente de datos de objetos.

2.5.3.4. Sistema Externo: las viewers en un cliente de Chimera pueden interactuar directamente con el usuario, pueden requerir el uso de herramientas externas o pueden usar un sistema manejador de interface de usuario para presentar su interface. Chimera no pone ninguna restricción sobre los sistemas externos a los cuales pueden acceder sus clientes o cómo estos clientes se comunican con sus usuarios.

2.5.4. Implementación de la Arquitectura:

Se ha construido un prototipo de Chimera bajo el Sistema Operativo Unix, usando los lenguajes de programación ADA y C. A continuación se hace una descripción de dicha arquitectura.

2.5.4.1. Server de Chimera: Este cumple con las responsabilidades que se enunciaron anteriormente. En el server de Chimera hay un conjunto de paquetes Ada que implementan conceptos de hipertexto de Chimera como ADTs. El server de Chimera coordina los accesos a este conjunto de ADTs, respondiendo a llamadas de procedimientos remotos hechos por clientes de Chimera. Estos clientes son invocados por usuarios finales (o el invocador de procesos) y contienen una viewer. El server de Chimera puede manejar múltiples clientes ejecutados por múltiples usuarios al mismo tiempo.

El nivel de acceso a la información es mantenido para cada viewer conectada al server de Chimera. Los anchors y enlaces pueden ser filtrados tal que los diferentes conjuntos de estos conceptos pueden ser provistos a diferentes usuarios por la misma view. Los usuarios pueden seleccionar el nivel de filtración recibido si la viewer provee una interface para hacerlo (como Chimera es un sistema de hipertexto abierto, esta funcionalidad no puede ser garantizada a través de todas las viewers. Sólo existirá si el desarrollador de la viewer decide incluirlo en una viewer particular). Por defecto no hay un nivel de filtración, por ejemplo todos los anchors y enlaces para una view particular están accesibles. El otro nivel de filtración es sólo de usuario, tal que sólo los anchors y enlaces creados por un usuario en una view particular son accesibles. Esta funcionalidad permite a Chimera proveer soporte para múltiples contextos en una hyperweb particular. Eventualmente se planea implementar un sistema con propiedades y permisos modelados como los permisos de archivo de Unix. Así, sólo los anchors y enlaces que son legibles para un usuario serán accesibles por una view particular. Luego se extenderá el soporte

de Chimera para múltiples contextos permitiendo que un usuario tenga diferentes roles y así tener acceso a diferentes conjuntos de anchors y enlaces.

Un enlace activo es mantenido para cada usuario conectado al server de Chimera. Un enlace activo es un mecanismo provisto por el server de Chimera para permitir la creación de enlaces sin modelos.

En los sistemas de hipertexto típicos, la creación de enlaces se hace de un modo particular. El usuario indica que desea un nuevo enlace, agrega anchors a este enlace y luego continúa trabajando. En Chimera, un usuario puede crear varios enlaces vacíos, seleccionar uno de esos enlaces para ser activado, y luego agregar anchors a este enlace activo en algún momento más tarde. El usuario también puede, en cualquier momento, seleccionar otro enlace para ser el enlace activo. Las viewers no son requeridas para usar el mecanismo de enlace activo. Esto es provisto sólo como una función de conveniencia en un intento de favorecer estilos de interacción comunes entre las viewers de Chimera. Una viewer puede renunciar al mecanismo de enlace activo, registrando sus propios enlaces y los modifica independientemente de las otras viewers.

Las hyperwebs son mapeadas en directorios de Unix, donde el server de Chimera, almacena archivos temporarios en tiempo de ejecución junto con el estado persistente de sus ADTs. Los usuarios seleccionan a que hyperweb quieren acceder a través de un archivo de recursos (.chimeraarc), localizado en sus directorios que el server de Chimera lee en el arranque.

El server de Chimera, actualmente soporta cuarenta eventos de hipertexto.

NOMBRE DE LOS EVENTOS	***INFORMACION ASOCIADA***
Desconectar server	Ninguna
Viewer nueva	Identificador de viewer
Borrar viewer	Identificador de viewer
Nuevo objeto	Identificador de objeto
Borrar objeto	Identificador de objeto
Nueva view	Identificador de view
Borrar view	Identificador de view
Nuevo anchor	Identificador de anchor
Borrar anchor	Identificador de anchor
Nuevo enlace	Identificador de enlace
Borrar enlace	Identificador de enlace
Modificar enlace	Identificador de enlace
Seguir enlace	Identificador de anchor
Activar enlace	Identificador de enlace

Uno de los eventos consiste en reportar cambios a la hyperweb, tal como borrar o agregar conceptos de hipertexto. Otro evento es el de seguir enlaces o activar enlaces.

Los clientes proveen una variedad de información al server acerca de ellos mismos. Esta información incluye si cada viewer en cada cliente está lista para recibir eventos de hipertexto, en que eventos de hipertexto cada viewers está interesada y en qué view está cada viewer.

El server también conoce como cada viewer será invocada vía un atributo de política de invocación asociada a cada viewer. Esta política es usada cuando el server debe enviar un evento de seguir un enlace a un view que actualmente no está mostrada por alguna de las viewers conectadas. Una viewer con una política de invocación de "once-only" (sólo una vez) es invocada sólo una vez y todos los eventos posteriores de seguir enlaces son enviados a ella. Esto es para las viewers que pueden mostrar múltiples views, quizás abriendo una nueva ventana para cada una o cerrando las previas antes de abrir una nueva.

Una política de invocación alternativa es "Every time" (en todo momento) lo que causa que el server invoque (vía el invocador de procesos) una nueva instancia de una viewer cada vez que ocurre un seguimiento de enlace en una view no mostrada.

2.5.4.2. El invocador de procesos: es un server RPC (llamado a proceso remoto).

El server de Chimera envía mensajes al invocador de procesos cuando quiere una viewer particular. El server de Chimera envía un nombre de viewer que el invocador de procesos mapea en un programa ejecutable o un Shell Script que luego invoca. Esta llamada es manejada para que el invocador de procesos haga uso del comando **fork** de Unix para comenzar un proceso hijo. Este proceso hijo llama al comando **execvp** de Unix que reemplaza al proceso hijo con el programa ejecutable especificado que luego empieza a ejecutarse.

El mapeo que el invocador de procesos usa para determinar qué programa ejecutable comienza, es leído una vez que arranca y es almacenado en la hyperweb a la que el usuario está accediendo.

La información en el archivo de mapeo debe ser entrada manualmente vía un editor de texto.

2.5.4.3. Cliente de Chimera: el mayor beneficio de la arquitectura cliente – servidor de Chimera es que sus clientes pueden ser escritos en más de un lenguaje. Se requiere una API (interface de programa de aplicación) al server de Chimera para un lenguaje particular, antes que el lenguaje pueda ser usado para construir los clientes de Chimera. Una ventaja del API es que los detalles de pasaje de mensajes del RPC de bajo nivel al server de Chimera están totalmente ocultos a los clientes que usan el API. En su lugar los clientes invocan subprogramas como Register, Anchor, o Traverse-link y el API convierte esta llamada a subprogramas (y sus parámetros) en mensajes RPC apropiados y los envía al server de Chimera. El API también pasa algunos valores de retorno del server al cliente que llama. Esta conversión es abierta e incluye la creación de un nuevo buffer RPC, ordenar los parámetros en el buffer, hacer la llamada RPC actual, recuperar algunos valores de retornos del buffer y desalocar el buffer del RPC.

Chimera soporta clientes escritos en Ada y en C con API para ambos lenguajes. Varios clientes han sido escritos usando cada una de estas APIs.

El API Ada crea dos tareas por viewer. Una tarea maneja el envío de mensajes al server de Chimera, la segunda tarea maneja la recepción de eventos de hipertexto desde el server. Estas tareas operan independientemente y mantienen sockets Unix separados. Esto permite múltiples conexiones al server Chimera con un proceso Unix único.

El API Ada fue probado para trabajar sucesivamente con otros sistemas cliente-servidor, lo más notable fue una conexión simultánea para una viewer a un server Chimera, un server Chirón y un server de sonido.

El API C permite que programas C usen servicios de Chimera con un proceso Unix particular. Dos sockets son mantenidos por el API C, los autores de las aplicaciones tendrán la responsabilidad del registro de transmisión de mensajes y la recepción de eventos. Como muchos programas usan el API de C también usarán X-Windows para producir sus interfaces de usuarios. Los eventos de Chimera son manejados usando un mecanismo callback (volver a llamar). Cuatro programas C han sido escritos, los cuales son simultáneamente clientes de Chimera.

2.5.4.4. Sistemas externos: siete viewers han sido integradas con el sistema Chimera y ellas son:

- Frame Maker: este programa fue integrado en el sistema Chimera sin modificar su imagen ejecutable. Un programa empaquetado traduce entre conceptos de hipertexto de Chimera y conceptos de hipertexto incorporados en Frame Maker.

- xvi: El clone vi de dominio público fue integrado con Chimera por un grupo de estudiantes. Todas las ediciones vi existentes trabajan normalmente, con la funcionalidad de hipertexto accedido a través de un panel de control de gráficos escrito usando el conjunto de herramientas MOTIF. Los servicios de Chimera son accedidos a través de la API C. Los seguimientos de enlaces al xvi hacen que un buffer diferente sea abierto por cada nuevo archivo referenciado en el enlace.

- Simulador de vuelo: es una simulación de un avión. Escrito en Ada usando Chirón, el simulador de vuelo representa cada medida en un panel de control separado.

- MPEG: la viewer mpeg permite que los anchors sean definidos en películas mpeg. Los anchors pueden ser definidos en secciones de cada frame, y pueden tener una duración desde un frame hasta la película entera. El soporte de creación permite que los anchors sean definidos en un frame, luego copiados de frame a frame. Los autores pueden seguir paso a paso hasta el final el film y ajustar los anchors en frames individuales de la forma más conveniente. Los anchors pueden ser agregados al enlace activo desde cualquier frame. La viewer mpeg fue creada para extender un intérprete mpeg de dominio público usando el API de C.

- XGIF: la viewer xgif permite que los anchors sean definidos en secciones de una imagen gif (formato de intercambio gráfico). Una viewer GIF de dominio público fue

mejorada usando la API de C para mostrar un panel de control de hipertexto escrito usando el Toolkit Motif. El panel de control varía dependiendo de si la viewer GIF está en modo autor o en modo view. Los seguimientos de enlaces al XGIF hacen que un nuevo proceso XGIF sea invocado para ver el archivo GIF enlazado.

- Intérprete de sonido (sound player): presenta una lista de sonidos a la cual los anchors están ligados. Los seguimientos de enlaces a un anchor particular causan que el intérprete de sonido reproduzca el sonido asociado. El intérprete de sonidos usa Chiron para su interface de usuario, el API de ADA para su funcionalidad de hipertexto, un server de sonido de SUN para la capacidad de sonido, haciéndolo simultáneamente un cliente de tres servers separados.

- Botón: es una viewer simple que muestra una ventana que contiene solamente un botón. Luego un anchor es asociado con este botón. Este anchor sólo puede ser miembro de un enlace (una restricción forzada para esta viewer particular).

El API de Chimera consiste de aproximadamente 90 puntos de entrada. La librería API de Ada es de 296 Kb. La librería API de C es de 62 Kb. El server de Chimera es de 2,3 Mb y usa aproximadamente 5 Mb de memoria cuando se ejecuta.

2.5.5. Versiones

Los mecanismos de control de versión son muy importantes para los sistemas de hipertexto. En Chimera el mecanismo para versiones es una prioridad de investigación.

Como Chimera no ofrece soporte para almacenar objetos de aplicaciones en su base de datos de hipertexto, la responsabilidad de control de versión debe ser compartida entre Chimera y sus aplicaciones clientes. Por ejemplo, Chimera es usado para crear relaciones entre archivos de código fuente. La responsabilidad de control de versión para los archivos sólo dependerá de sistemas de control de revisión existente tal como RCS.

La responsabilidad para versionar las relaciones entre los archivos residirá con Chimera.

Chimera será capaz de múltiples versiones. Esto aumenta la importancia de mantener la consistencia cuando una instancia es cambiada. Por ejemplo, cuando un anchor es borrado, también debe ser removido de aquellos enlaces a los que pertenece, requiriendo una nueva versión de estos enlaces.

Otro problema es conservar la consistencia entre las versiones mantenidas por un sistema de versiones externos y las versiones mantenidas por Chimera. Resolver este problema y proveer un mecanismo para la denominación de versiones multi-concepto flexible requiere un nuevo concepto de primera clase en Chimera, **la configuración**.

Una configuración contendrá el reflejo instantáneo de las versiones actuales de un subconjunto de hyperwebs. Las versiones de objetos externos pueden luego ser mapeadas a una configuración.

2.5.6. Integrando Chimera con la WWW

El Sistema de Hipermedia Abierto emplea una variedad de técnicas para proveer servicios de hipermedia a diversas aplicaciones. La WWW es un sistema de hipermedia distribuido en uso y fue desarrollado independientemente de los OHS. La popularidad de la WWW junto con los problemas propios de su diseño ha motivado a los investigadores de los OHS a integrar sus sistemas con ella.

El primer enfoque fue mejorar la funcionalidad de la Web usando los servicios de los OHS. Se hicieron tres experimentos de integración de Chimera con la Web.

Los OHS proveen servicios de hipermedia a aplicaciones clientes vía una arquitectura abierta distribuida. El OHS es responsable del manejo de enlaces de hipermedia para sus aplicaciones clientes, asegurando consistencia de los enlaces en las fases de cambio. Esa consistencia se mantiene construyendo un modelo de estructura de hipermedia creada por sus aplicaciones clientes. Si una aplicación cliente borra un anchor, el OHS responde removiendo el anchor de todos los enlaces que contienen a éste. Este modelo de hyperweb permite recorrer y buscar rápidamente dentro de la estructura de hipermedia.

A diferencia de los sistemas de hipermedia monolíticos (tal como el KMS), un OHS puede integrar una amplia variedad de clientes a través de una serie de técnicas de integración. Además la arquitectura de un OHS es distribuida en términos de ejecución, datos y tiempo. La primera categoría se refiere a la ejecución concurrente de clientes y servidores a través de un conjunto de máquinas host. La segunda categoría implica que el cliente y el servidor puedan almacenar y recuperar su información persistente localmente o en un sistema de archivo remoto, o un objeto almacenado. La distribución a través del tiempo se refiere a los soportes de los sistemas de hipermedia para colaborar con sus usuarios finales.

Los protocolos empleados no son standard y varían de acuerdo a los sistemas, por lo tanto el usuario final no puede trabajar fácilmente con múltiples sistemas de hipermedia abiertos y el estudio de uno no implica el conocimiento acerca del funcionamiento de otro.

La WWW provee soporte para cada una de las tres áreas anteriormente descriptas. Hace uso de formatos de datos tal como HTML y protocolos de acceso tal como HTTP, que son abiertos, extensibles y standard. La existencia de esa standaridad ha permitido que la WWW esté disponible en todas sus plataformas, contando con una barrera de entrada baja por lo cual su uso se ha extendido.

Pero a pesar de esto, la WWW tiene varias dificultades en lo relacionado a la hipermedia. El modelo de hipermedia de HTML tiene los enlaces embebidos, no hay separación entre los datos enlazados y los enlaces. Esto lleva a

una diversidad de problemas incluyendo el mantenimiento de enlaces en la fase de cambios y existencia de enlaces colgantes. Además sólo soporta enlaces punto a punto, no pudiendo manejar enlaces n-arios.

Por otra parte, las formas standard de hipermedia como ser guías turísticas, no son provistos automáticamente y el usuario debe generarlas manualmente o depender de un generador HTML para incluirlos. Finalmente, las anotaciones están prohibidas por los protocolos standard. Una forma limitada de anotar puede ser provista a través del uso de CGI scripts los cuales son dificultosos para desarrollar y el mecanismo resultante no es standard.

Se apunta a integrar los OHSs con la Web. El objetivo de esto es lograr un sistema híbrido donde las ventajas provistas por uno, ayude a contrarrestar las desventajas del otro. Los OHSs pueden usarse para proveer sofisticadas formas de hipermedia no presentes en la Web y permitir enlaces consistentes y manejo funcional sobre los datos almacenados en la Web. La Web puede proveer una mayor distribución de información de los OHSs. A continuación se detallará los beneficios obtenidos al realizar esta integración.

2.5.6.1. Trabajos de integración:

Se lograron teniendo en cuenta los 4 elementos arquitecturales fundamentales comunes a ambos sistemas: clientes, servidores, protocolos y formatos de datos.

Dato OHS a dato WWW

Una simple integración es producir herramientas que traducen enlaces y contenidos en OHS a HTML.

Es probable que haya restricciones en el tipo de información y funcionalidad que puedan ser traducidas, pero la integración tiene una propiedad importante y es la de no modificar los elementos de ambos sistemas, lo cual alivia el trabajo. Se requerirá un gran esfuerzo en el desarrollo de los traductores, pero aún así es significantemente menor al esfuerzo de tener que modificar el server de un OHS o extender un server de la WWW.

Este tipo de integración permite al usuario final tener los beneficios de creación aportados por un OHS y al mismo tiempo contar con la habilidad para publicar información en la Web.

En Microcosmo se ha hecho un trabajo de este tipo produciendo una herramienta que convierte aplicaciones de Microcosmo dentro de un conjunto de documentos HTML enlazados. Esa conversión tiene algunas restricciones tales como que sólo documentos RTF e imágenes BMP pueden ser traducidos por la herramienta dentro de HTML y GIF respectivamente. En cuanto a la funcionalidad de hipermedia, la herramienta puede fácilmente convertir enlaces específicos y botones de Microcosmo en enlaces HTML, los enlaces locales y genéricos también pueden ser convertidos lo cual involucra investigar el texto de todos los documentos incluidos en la aplicación para hallar todos los destinos posibles, además de tener la

dificultad de traducir enlaces con múltiples destinos dentro de HTML.

Cliente WWW como Cliente OHS

Otra alternativa es integrar un cliente WWW (como ser una browser de la Web) con un OHS. Dicha integración fue hecha con Microcosmo y con Chimera. La primera integración se hizo usando la Viewer Universal de Microcosmo y la segunda usando el Wrapper de Chimera.

Este tipo de integración obtiene un nivel de funcionalidad tal que las páginas HTML pueden ser enlazadas fuera de la Web y dentro de otros clientes soportados por el OHS. Esto tiene la ventaja de permitir a los autores escribir su documentación en HTML y enlazarlos directamente a las aplicaciones relevantes.

Server WWW como cliente OHS

Se puede realizar una integración extendiendo el server WWW para ser un cliente de un OHS. Con esto los usuarios pueden acceder a la funcionalidad del OHS desde una consulta de Web standard inmodificable.

El server Web es modificado para producir llamados en el OHS en respuesta a los requerimientos de ciertos URLs. Basado en la respuesta de esos requerimientos, el server genera HTML para ser enviado nuevamente al cliente que presumiblemente contiene marcas que manifiestan la presencia del OHS.

Este avance tiene el principal beneficio de separar enlaces del documento de la Web, los enlaces son generados e insertados dinámicamente. Además los autores pueden continuar para crear hyperwebs en sus OHS favoritos y tienen el resultado disponible vía la Web.

Los mecanismos de implementación para la integración consisten en extender el server WWW directamente vía cambios a su código orígen, usando un server plug-in o implementando script CGI.

La primera de las dos opciones provee beneficios de funcionamiento, mientras el otro es más portable entre servers Web. Un mecanismo de implementación para estos avances es usar un proxi HTTP del lado del cliente. Este mecanismo tiene la ventaja de permitir al usuario especificar cuando éste quiere acceder a los servicios del OHS y el beneficio adicional es que se evita el trabajo requerido para modificar el server Web.

Microcosmo ha mejorado el trabajo en esta área en un proyecto llamado Servicio de enlace distribuido. En este sistema el server Web ha sido extendido para acceder a linkbases y filtros distribuidos. En respuesta a requerimientos de clientes, la información de las linkbases es compilada dentro del documento HTML como URLs .

Integración Híbrida

Combina dos o más técnicas descriptas y aplica esa simultaneidad en una aplicación de hipermedia. Una integración híbrida se logró combinando las 2 secciones previas. El resultado fue un poderoso ambiente de hipermedia para el usuario final.

Además de usar clientes OHS normales, estos pueden ahora enlazarse dentro y fuera de documentos Web. El server integrado, los anchors y enlaces creados en los documentos Web son almacenados externamente en las bases de datos de hipermedia de los OHS, sin embargo aparecen como enlaces standard de la Web, mientras es recorrido por una browser de la Web. Un usuario también puede configurar y acceder a otros servicios del OHS desde la browser de la Web debido a la integración cliente. El DLS logra esta integración híbrida modificando la Viewer Universal de Microcosmo que permite a los usuarios acceder a las características standard de Microcosmo a través de la integración del server de la Web, descripto anteriormente.

Server OHS como server WWW

Una forma interesante de integración es modificar el server OHS para ser enmascarado como un server WWW. Mientras el resultado de esta integración, desde el punto de vista del usuario final, es similar a las secciones previas, ésta ofrece mayor flexibilidad para el OHS. La función es mejorada ya que el OHS recibe pedidos HTTP directamente y tiene inmediato acceso a los documentos requeridos y a su base de datos de hipermedia. Además los servicios pueden estar disponibles ya que una API del OHS puede ser presentada a clientes WWW. Sin embargo esta integración implica un esfuerzo de desarrollo significativo ya que implementar la funcionalidad de un server de la Web no es sencillo. Esta complejidad puede ser mejorada limitando el alcance del manejador del URL para el server OHS.

Un importante trabajo en este área fue mejorado por la Universidad de tecnología de Graz, a través de su proyecto Hyper-G. Este sistema es un ambiente de hipermedia avanzado que integra con la Web usando estos servers enmascarados como un server WWW. Los servers Hyper-G tienen la habilidad para recibir requerimientos desde los Clientes Web standard y transmitir esa información (con una pérdida asociada de funcionalidad) dentro del HTML. Los usuarios pueden así conseguir acceso a la amplia cantidad de información estructural almacenada en Hyper-G desde algún cliente WWW standard.

2.5.6.2. Protocolos WWW dentro de Chimera

La estructura jerárquica empleada por los servers WWW para organizar información dentro de las Websites pueden ser usadas para organizar el manejador de Hyperwebs dentro de un OHS. Además el uso de las URLs para encontrar los componentes de los OHS permite que esos componentes sean globalmente distribuidos, accesibles en forma familiar para todos los usuarios de la Web.

Los protocolos Web no son utilizados para proporcionar integradores de herramientas para acceder a espacios de trabajo remotos. Se usa un servicio de nombres que acomoda una pequeña cantidad de trabajos del usuario, en términos de mantenimiento (los usuarios deben entrar manualmente la información de conexión para espacios de trabajo remotos), pero ofrece más transparencia y flexibilidad que el protocolo URL (el usuario puede asignar un nombre a los espacios de trabajo y permite al servicio de nombres mapear esos nombres al propio espacio de trabajo).

2.5.7. Eperimentos de Integración:

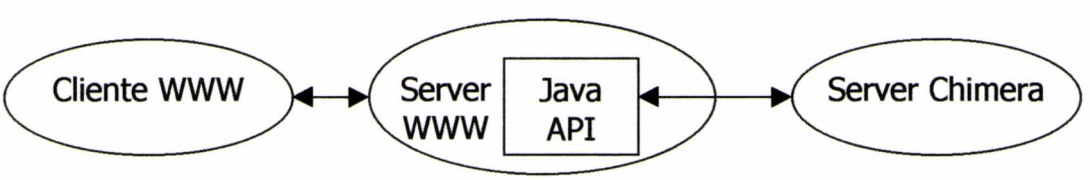
Chimera es un sistema de hipermedia abierto con un énfasis en la modelización flexible de ambientes de ingeniería de software.

Este consiste de un server que provee servicios de hipermedia para aplicaciones clientes con total consistencia de enlace y capacidad de manejo.

El primer experimento de integración utiliza la versión inicial de Chimera para demostrar un cambio sobre la integración de un server WWW con un OHS. El segundo experimento describe el desarrollo de una nueva versión de Chimera que utiliza un protocolo standard de la Web. Este le permite localizar y conectar varios componentes y simplifica el manejo de las hyperwebs de Chimera. El tercer experimento describe el uso de una integración híbrida OHS-WWW para proveer la presencia de Chimera en toda la Web.

1º- Extendiendo un server WWW

En este experimento un server WWW escrito en Java fue extendido para proporcionar el uso del Java API de Chimera. Esta extensión permite al server acceder a los anchors y enlaces almacenados en la hyperweb de Chimera.



Arquitectura: un server WWW standard hace uso del Java API de Chimera 1.0 para comunicarse con el server de Chimera. Este interpreta los requerimientos HTTP como operaciones en el server de Chimera y genera HTML para mostrar los resultados.

El server de Java conecta a Chimera en el arranque e interpreta los URLs como requerimientos en el server de Chimera. Un requerimiento URL hace que el server recupere todos los anchors desde la actual hyperweb, mostrándolos como una lista de identificadores de anchors. Clickiando en uno de esos anchors listados hace que el server recupere la información adicional tal como sus views asociadas, incluyendo a las viewers y objetos. En particular si el objeto de una view es un texto o un archivo GIF, este archivo es recuperado y mostrado.

Luego todos los enlaces asociados con el anchor actual son mostrados. Esta lista muestra el conjunto de anchors contenidos en cada enlace. Clickiando en alguno de los destinos se sigue el enlace y recupera la información asociada. Los usuarios pueden además, acceder directamente a enlaces o anchors especificando su identificación con un URL de la forma <http//WWW.some.domain/[anchors/link] ###>.

En esta forma los usuarios son capaces de seguir una hyperweb de Chimera sin tener que invocar a los clientes asociados que normalmente manejan el display, el manejador de anchors e información de view. Este mecanismo de consulta habilita una hyperweb para ser accedida rápidamente por el usuario.

En el futuro, este server puede ser extendido para mostrar anchors para los formatos de datos que él conoce. Además una interface simple permite que un usuario pueda consultar directamente al server de Chimera en un camino standard en toda la Web.

Este trabajo es similar al trabajo realizado por DLS extendiendo un server WWW para acceder a los filtros DLS, cuando recupera documentos HTML. La diferencia es que este trabajo utiliza mecanismos de la Web existentes para mostrar la estructura de hipermedia contenida en una hyperweb de Chimera y no intenta modificar dinámicamente el contenido de un documento HTML basado en la hyperweb actual.

2º- Utilizando protocolos Web dentro de un OHS

La versión original de Chimera tenía algunas restricciones de implementación que obstaculizaba el aspecto distribuido del sistema. Este experimento reduce la curva de aprendizaje para nuevos usuarios utilizando protocolos de la Web y estilos de interacción standard de la Web.

Estos problemas encontrados en la versión inicial de Chimera fueron mejorados durante el desarrollo de este experimento.

Para que el usuario pudiera acceder a una hyperweb de Chimera, la cuenta del usuario, los clientes de Chimera, el server Chimera y toda la hyperweb deseada debían residir en sistemas de archivos de la red. Esta restricción era la que limitaba al soporte de Chimera para distribución a una LAN.

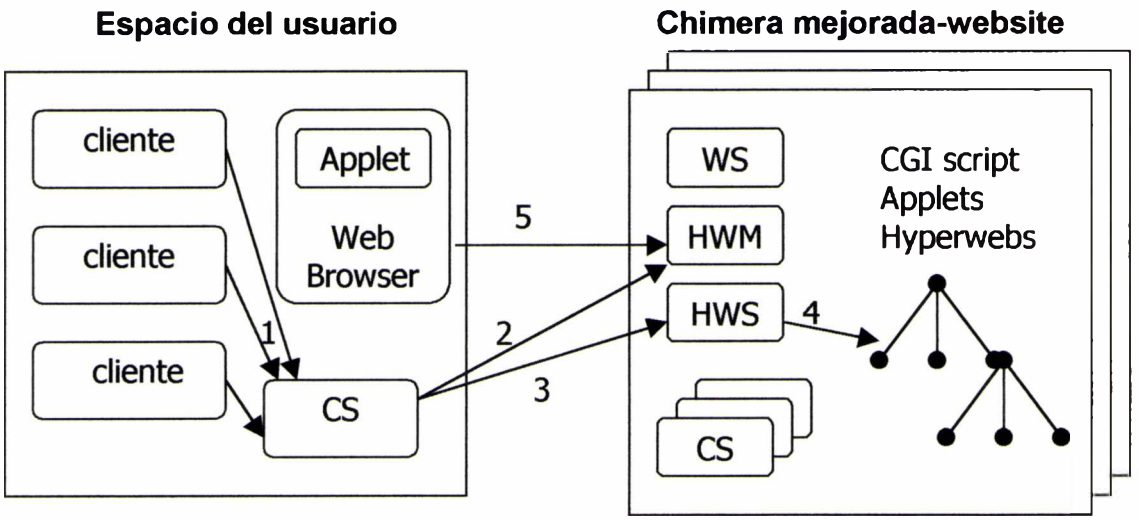
El motivo de esta restricción involucraba la posibilidad para acceder a varios archivos de texto que contenían la localización de la hyperweb y la información de conexión para el server de Chimera.

Las hyperwebs eran dificultosas para manipularlas y ser usadas efectivamente. El único camino posible para referenciarlas era con su nombre de camino absoluto.

No era posible hacer enlaces entre hyperwebs, por lo tanto el usuario debía acomodar toda la información dentro de una hyperweb antes de distribuir

información por medio de hyperwebs relacionadas. Finalmente Chimera no proveía mecanismos para descubrir la existencia de hyperwebs creadas por otros usuarios.

La nueva versión de Chimera tiene una arquitectura dividida entre un ambiente de usuario y una Website externa.



CS ---------- Server cliente
HWM ---------- Manejador de hyperweb
HWS ---------- Server de hyperweb
WS ---------- Server WWW

Los URLs son usados para referenciar las Websites y sus hyperwebs de Chimera. El HTTP es usado para pedir información acerca del sitio desde el manejador de hyperweb.

Interacciones dentro de la arquitectura de Chimera 2.0: este diagrama muestra alguna de las interacciones posibles entre los componentes Chimera:

(1) Un usuario puede invocar clientes Chimera que interactúan con el server cliente local
(2) El server cliente local presenta una interface de usuario que permite especificar un URL de una Website Chimera. El server cliente se conecta al manejador de hyperweb del sitio para determinar la información de conexión del server hyperweb del sitio.
(3) El server cliente se conecta con el server hyperweb y permite al usuario seleccionar entre las hyperwebs disponibles.
(4) Una vez seleccionada, el server hyperweb carga la hyperweb en memoria, permitiendo que sea manipulada por los clientes activos.
(5) El manejador de hyperweb, que tiene un número de port predefinido, puede manejar directamente requerimientos HTTP para información acerca del sitio.

Un espacio del usuario contiene el cliente Chimera, una consulta de la Web y un server cliente. Una website de Chimera contiene un server WWW, un manejador de hyperweb, un server de hyperweb y un conjunto de servers clientes.

La función del server de Chimera original ha sido dividida entre el server Cliente y el server de la hyperweb. El server cliente provee al usuario de una interface para especificar la website de Chimera de interés y seleccionar entre las hyperwebs almacenadas allí (ambos vía URLs). Este además maneja la conexión al manejador de la hyperweb del sitio y al server de la hyperweb y rutea al server de la

hyperweb los requerimientos de los clientes del usuario para manipular la hyperweb actual. El server cliente no utiliza un servicio de nombres para localizar los servers remotos. Los URLs son usados para inicializar los requerimientos HTTP para establecer la conexión entre el server cliente y el sitio remoto. Una vez conectado, éste desvía al protocolo natural de Chimera para una comunicación más eficiente.

El server de hyperweb es responsable del almacenamiento persistente y manejo de las hyperwebs de los websites de Chimera. Cada hyperweb contiene instancias de conceptos de hipermedia de Chimera y es localizada bajo el espacio de nombres del server de la WWW. Esto les permite fácilmente ser referenciadas a través de los URLs.

Chimera mantiene una estricta separación entre contenido de las aplicaciones específicas e información de hipermedia.

El browser de la Web en un ambiente de usuario puede bajar clientes de Chimera en la forma de Java Applets ya que la comunicación applet está restringida a sus hosts servers, esos applets conectan a los servers clientes ejecutados en una website de Chimera. Estos reciben la información de la conexión necesaria desde el manejador de la hyperweb del sitio. Una vez conectado, éstos pueden usar el server cliente para conectar a cualquier website de Chimera, no justamente desde donde fueron bajados.

El manejador de la hyperweb permite la creación y borrado de hyperwebs, éste sigue la pista del estado de esas websites incluyendo la información de conexión para los otros servers de Chimera relacionados. Este manejador de hyperweb está disponible para servir a los requerimientos HTTP y permitir que los clientes remotos se conecten a estos servers. Por ejemplo <HTTP//WWW.SOME.DOMAIN:HWM/SERVERS> consulta al manejador de hyperweb para la información de conexión de los servers clientes en su dominio.

Un URL de la forma:
<HTTP//WWW.SOME.DOMAIN:HWM/CHIMERA/HYPERWEBS>
es usado para determinar los nombres de las hyperwebs disponibles para un sitio.
El URL <HTTP//WWW.SOME.DOMAIN:HWM/CHIMERA/HYPERWEBSERVER> recupera la información de conexión para los servers de hyperwebs del sitio.

Los URLs conectan el manejador de la hyperweb para recuperar la información de conexión del server que es usada para ser especificada en archivos de texto sobre un sistema de archivos compartidos. El descubrimiento de nuevas hyperwebs se realiza a través de URLs de hyperwebs que retornan información sobre todas las hyperwebs disponibles en un sitio particular.

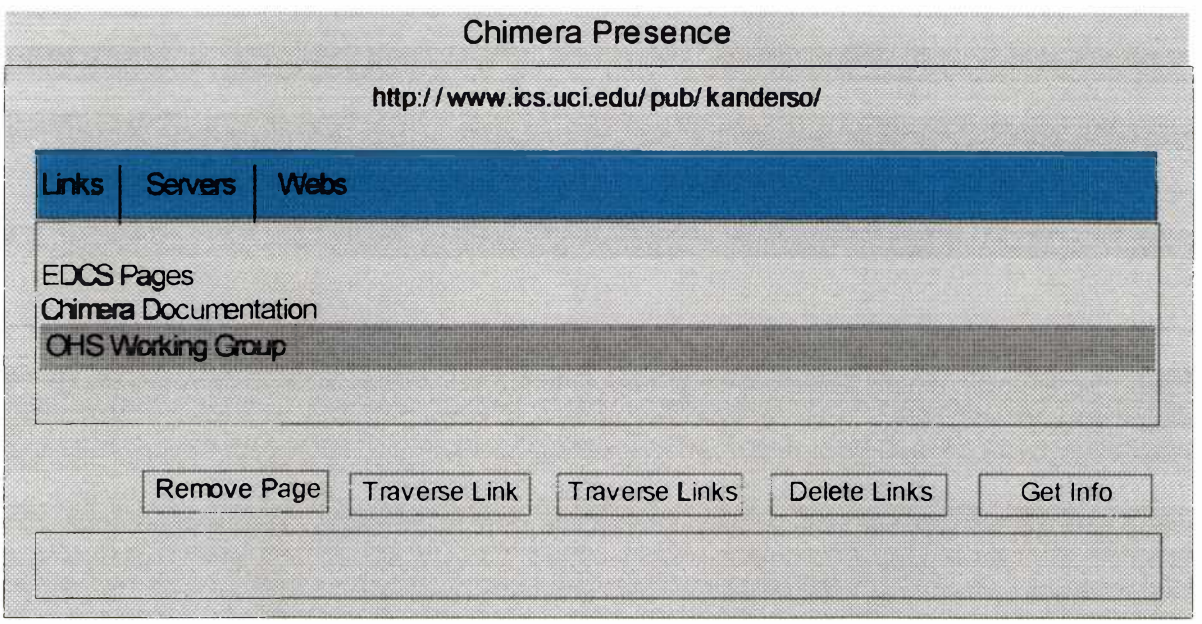
Las hyperwebs son fáciles de manejar y entender debido a su nueva estructura jerárquica explícita y a la interface provista por el manejador de hyperweb que provee operaciones, tales como: CREATE, DELETE, RENAME y GET INFO. También se soportan enlaces entre hyperwebs.

En la versión original de Chimera, el server de Chimera podía sólo manejar una hyperweb a la vez, lo cual imposibilitaba hacer enlaces a otras hyperwebs. Con la nueva separación entre el cliente y el server de la hyperweb, un

cliente puede tener a un server cliente conectado a múltiples servers de hyperwebs y comenzar la construcción de un nuevo enlace. Cada anchor en el enlace está asignado al URL de su hyperweb. Se guarda una copia de esos enlaces en cada hyperweb. Luego el server cliente puede realizar un seguimiento de enlace desde cualquier hyperweb y usar los URLs asociados con cada anchor para recorrer la hyperweb correcta.

3º- Proveyendo presencia OHS en la Web

La idea es juntar un applet Java, que provee acceso a servicios de Chimera, a todas las páginas de la Web que un usuario visita. El usuario puede crear y manipular enlaces, conmutar entre hyperwebs e inicializar seguimiento de enlaces, todo desde la interface de usuario de applet.



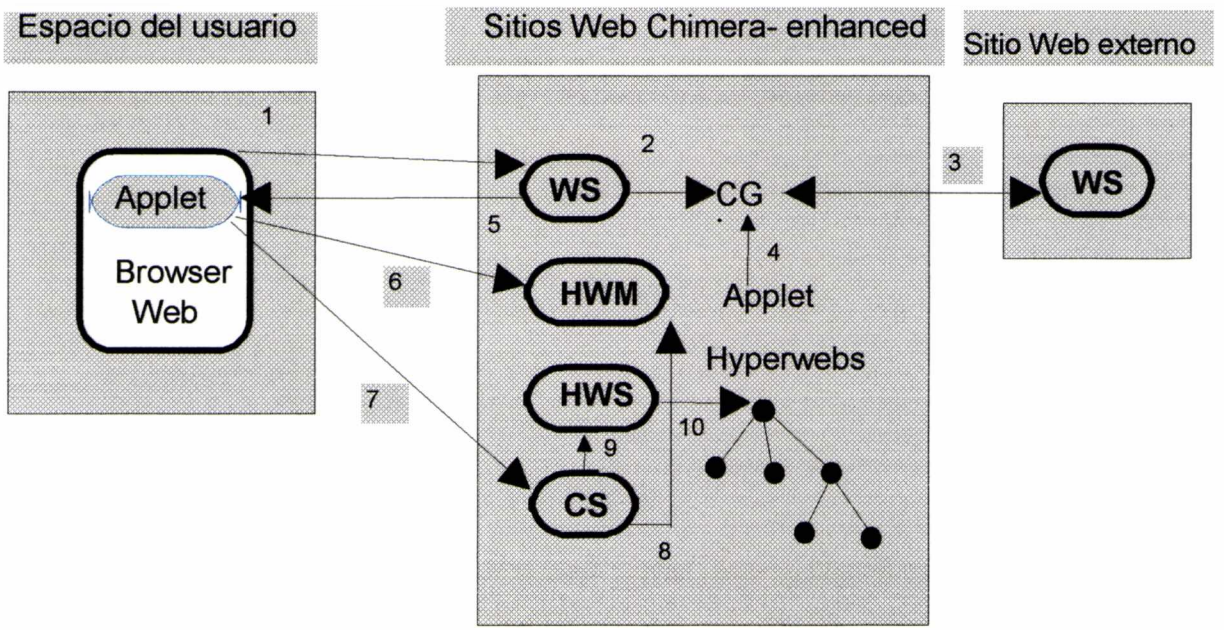
El applet presenta una interface de usuario dividida en paneles. El panel del server permite al usuario especificar la website Chimera deseada mientras el panel webs les permite seleccionar entre hyperwebs del sitio. El panel links muestra todos los enlaces contenidos en la hyperweb y provee operaciones para manipular enlaces, iniciar seguimientos, examinar en detalle un enlace y agregar o remover la página Web a o desde el enlace seleccionado.

Los mecanismos applets de Java no son capaces de obtener el objetivo de presencia, debido a las restricciones de seguridad en applets. Un applet no puede leer el contenido de un URL. Un applet sólo puede establecer una conexión socket a la máquina desde la que fue bajado. Un applet no es capaz de comunicar estados entre instancias de sí mismo localizadas en páginas web separadas.

La solución para este experimento involucra tres partes:

1. El applet implementa los requerimientos HTTP directamente en el manejador de la hyperweb.
2. Se usa un script CGI para juntar el applet llamado Chimera Presence a una página Web.

3. Se usa una forma HTML para inicializar el script CGI con el nombre del usuario. El script CGI establece un HTTP para salvar el nombre del usuario para cada invocación del applet.



Arquitectura del 3er. Experimento:

(1) El usuario carga una forma HTML especificando su nombre de usuario y destino.
(2) El server Web recibe esta forma e invoca su script CGI asociado.
(3) El script CGI recupera el documento destino, modifica sus anchors e
(4) Inserta la etiqueta applet al final del documento.
(5) Como resultado el applet es cargado en la browser de la Web.
(6) Después de la inicialización, el applet se conecta (vía un requerimiento HTTP) al manejador hyperweb ubicado en su sitio server.
(7) El manejador de hyperweb lo provee con la información de conexión para un server cliente que luego éste conecta.
(8) Después de determinar el sitio destino (construido en el applet o provisto por el usuario) el server cliente se conecta al manejador hyperweb del destino.
(9) Esto permite al usuario seleccionar una hyperweb, que es accedida vía el server hyperweb del sitio.
(10) Si la hyperweb no está actualmente activa, el server hyperweb la carga en la memoria. Cuando se pasa a una nueva página el script CGI se conecta nuevamente y el proceso se repite. Esta vez el applet por defecto usa la website y la hyperweb seleccionadas en la iteración anterior.

El problema de que un applet no esté capacitado para acceder al contenido de los URLs es resuelto implementando los requerimientos HTTP directamente en el manejador de la hyperweb. El manejador de la hyperweb puede ser conectado desde su creación a la website de Chimera mejorada, desde el cual el applet fue cargado. El manejador de hyperweb hace uso de un número de port predefinido evitando la necesidad de un servicio de nombres para que los clientes lo localicen.

El script CGI está dado por un parámetro: el URL de una página Web. Este recupera el contenido del URL especificado, modifica los enlaces y agrega el HTML necesario para incluir el applet Chimera Presence al final de la página.

El script CGI resuelve estos problemas de la siguiente forma:

1. Modifica la página de la Web para llamar siempre al script CGI para los siguientes recorridos de enlaces. Esto permite al script CGI juntar los applet en cada página que el usuario visita siguiendo los enlaces. El applet no aparecerá cuando un usuario entre directamente un URL en la browser de la Web.
2. Ejecuta una website de Chimera mejorada lo cual significa que será capaz de comunicarse con el manejador de la hyperweb y el server cliente de ese sitio.
3. Establece parámetros applet para especificar la página actual y los nombres de usuarios. Este recupera el nombre del usuario cuando es invocado por primera vez por la forma HTML.

Como el sitio actual y la hyperweb son cambiados por el usuario desde el applet, éste almacena esa información con el manejador de hyperweb indexado por nombre de usuario. Cuando es invocado, el applet recupera los valores actuales para el sitio e hyperweb desde el manejador de hyperweb usando el nombre de usuario proporcionado por el script CGI.

Chimera 2.0 es una integración híbrida y el script CGI permite al server proveer funcionalidad OHS.

2.6. MICROCOSMO : SISTEMA DE HIPERMEDIA ABIERTO

2.6.1. La Historia:

Un grupo de investigación de multimedia comenzó a trabajar en la Universidad de Southamptom en Inglaterra en 1985. Por supuesto no fue llamado multimedia en aquellos días. La única tecnología disponible en aquel momento para integrar texto, gráficos, videos y sonidos fue el video disco interactivo.

El grupo desarrolló un número de aplicaciones educacionales usando algunos recursos basados en video disco que estaban disponibles en esa época.

El objetivo en esos días, fue crear ambientes de manejo y creación de información que permitiera a los constructores de la aplicación, acostumbrarse a los recursos disponibles (multimedia) en ambientes de aprendizaje individual.

Una consecuencia de esto fue aplicar tecnología de hipertexto para permitirle a los usuarios consultar la base de recursos, usando el software de la aplicación. La idea sobre el concepto de hipertexto surgió en los años 60, pero a mediados de los años 80 aún había muy pocos sistemas de hipertexto disponibles.

Al mismo tiempo, la Hartley Library de la Universidad de Southamptom le derivó a este grupo de investigadores el archivo de Lord Mountbatten, el Conde de Burma, quien estuvo relacionado con la familia real y con la vida militar. Su archivo reflejó todos estos aspectos de su vida, el cual consiste de alrededor de 250.000 documentos de texto, 50.000 fotografías, una colección de voces grabadas en registros de 78 rpm. y un gran número de filmes y videos.

Una tarea del Departamento de Manuscritos y Archivos de la Universidad fue catalogar el archivo y hacerlo accesible para que lo comprendan los investigadores. Tratar con gran cantidad de fotografías, sonidos, filmes (en distintos formatos) y videos llevó a un problema de gran magnitud y a la necesidad de utilizar sistemas de información multimedia, para lograr una versión electrónica de dicho archivo.

Los investigadores se aproximaron al problema construyendo aplicaciones basadas en recursos usando sistemas de hipertexto. Los principales temas para el autor/diseñador de hipertexto son el tamaño completo del archivo, su naturaleza de multimedia y el hecho que distintos usuarios accedan al archivo desde distintas perspectivas.

Se comenzó un proyecto piloto tomando una pequeña parte del archivo. Se publicó una guía de diseño inicial para Microcosmo junto con una descripción de la implementación del primer prototipo.

Uno de los objetivos fue reducir el esfuerzo de creación. Con grandes colecciones de documentos la creación manual consume mucho tiempo. Además si los enlaces de hipertexto, una vez creados son incorporados en los documentos mismos es muy difícil que estos enlaces sean reusados en otros documentos. Es

casi imposible para los autores seguir la pista de los enlaces que ellos crearon, mucho menos pasar esa información a otros.

Como un resultado de este análisis se resolvió construir un sistema de hipertexto que no imponga marcas sobre los datos que van a ser enlazados.

Se combinó con la idea de usar la selección - acción de interface de usuario gráfica como el método básico de interacción con el sistema, dando a los autores y lectores la capacidad para seleccionar un área de interés en el documento y elegir una acción para ser ejecutada sobre esta selección, tal como seguir o crear un enlace. Esto los llevó a diseñar un sistema de hipermedia que cumpla con los siguientes criterios:

- Un sistema que no impone ninguna marca sobre los datos lo cual previene que el dato sea accesible por otros procesos que no pertenecen al sistema.
- Un sistema que puede integrar cualquier herramienta que corre bajo el sistema operativo del host.
- Un sistema en el que los datos y procesos pueden ser distribuidos a través de una red.
- Un sistema en el cual no hay distinción entre lectores y autores.
- Un sistema en el que es fácil agregar nueva funcionalidad.

Esta filosofía se utilizó para diseñar el Sistema Microcosmo.

El manejo de información está separada en tres capas : la vista de la información del usuario, el enlace de hipermedia y los servicios de filtros y el sistema manejador de documentos.

Las consecuencias de estos principios de diseños fue la necesidad de desarrollar un sistema manejador de enlaces y un método para permitir que los enlaces se puedan hacer a aplicaciones que no están bajo el control del sistema de hipermedia. Se trató de mantener el sistema tan abierto como sea posible. La separación de estructura de dato y enlace permite el reuso de datos existentes sin afectarlos en alguna forma.

La esencia de Microcosmo es realmente un conjunto de protocolos de comunicación que permite la integración de todos los tipos de herramientas de procesos de información, incluyendo un servicio de enlace de hipermedia.

2.6.2. ¿Qué es Microcosmo?

Este sistema inicialmente fue diseñado con la intención de proveer un sistema de prueba sobre el cual un grupo de investigación de la Universidad de Sounthampton podría experimentar con varias ideas en el campo de multimedia. Pero ahora es usado en varios sitios para integrar aplicaciones de multimedia. La característica abierta del sistema es la atracción principal del producto. Sin embargo, se cree que usar un sistema de hipermedia como una plataforma de presentación para información de multimedia es simplemente una solución parcial a un problema mayor, que es la provisión de un ambiente de hipermedia completo en el nivel del sistema operativo.

Microcosmo es un sistema de hipermedia abierto que permite al usuario recorrer y consultar grandes colecciones de multimedia. Sin embargo, es diferente a muchos sistemas por su habilidad para integrar información producida usando una variedad de aplicaciones de terceras partes. Microcosmo puede ser visto como un ambiente que permite al usuario crear enlaces desde documentos en una aplicación a documentos en otra.

Si bien el browser HTML / NEWS permite recorrer grandes volúmenes de información difiere de Microcosmo en que HTML requiere poner directamente enlaces en los documentos mientras que Microcosmo almacena los enlaces en bases de datos.

Microcosmo es un software que le permite al usuario usar documentos en su formato original. Otros productos de hipermedia requieren que todos los datos se conviertan a un formato que su software pueda manejar.

Cada usuario puede crear un conjunto de enlaces que considere apropiado para su navegación. Microcosmo permite crear un conjunto de enlaces básicos y los usuarios pueden agregar sus propios enlaces individuales. Cada usuario sólo ve los enlaces que han sido creados por el autor y por ellos mismos.

Microcosmo ha sido usado en el campo educacional para grandes archivos, para un Sistema de Información Urbano, para computadoras de ayuda en ingeniería y para entrega de materiales de multimedia en el campo de otras aplicaciones.

2.6.3. ¿Qué hace Microcosmo?

Microcosmo permite combinar distintos medios: animación, gráficos, videos, textos y planillas de cálculo. Esto difiere de otros sistemas porque permite mantener los archivos en su formato y localización original. Otros softwares obligan a importar, mover o convertir la información original en su propio formato.

Microcosmo provee la facilidad de crear enlaces a documentos antes que estos estén completos, los cambios hechos en esos documentos son reflejados instantáneamente cuando se ven a través de Microcosmo.

Se puede hacer una variedad de enlaces en los diferentes medios: específicos, locales y genéricos. Una de las ventajas de Microcosmo es la habilidad de tener diferentes linkbases (base de datos que contiene detalles de los enlaces creados). Como autor de una aplicación Microcosmo, cada usuario puede crear un conjunto de enlaces que considera que pueden ser necesarios para otros usuarios.

Como un autor no puede anticipar todos los requerimientos de todos los usuarios, Microcosmo permite a cada usuario agregar su propio conjunto de enlaces para complementar aquellos del autor. Estos enlaces son específicos de cada persona, sólo los enlaces originales del autor son universales.

2.6.4. ¿Cómo usarlo?

Con Microcosmo se pueden crear enlaces en forma muy simple.

1. Importar al DCS todos los documentos que se desean enlazar.

2. Seleccionar el lugar en que se quiere comenzar un enlace, éste puede ser una palabra de texto, un fragmento de un video, una parte de una figura o un objeto en un dibujo Cad.

3. Seleccionar el botón **START LINK** del menú.

4. Seleccionar el lugar en donde se quiere terminar el enlace. Este puede ser el mismo o diferente documento, animación, video, sonido, texto, etc.

5. Seleccionar en botón **END LINK** del menú.

6. Asignar un nombre descriptivo al enlace y seleccionar el tipo de enlace que se desea hacer: botón, específico, local o genérico.

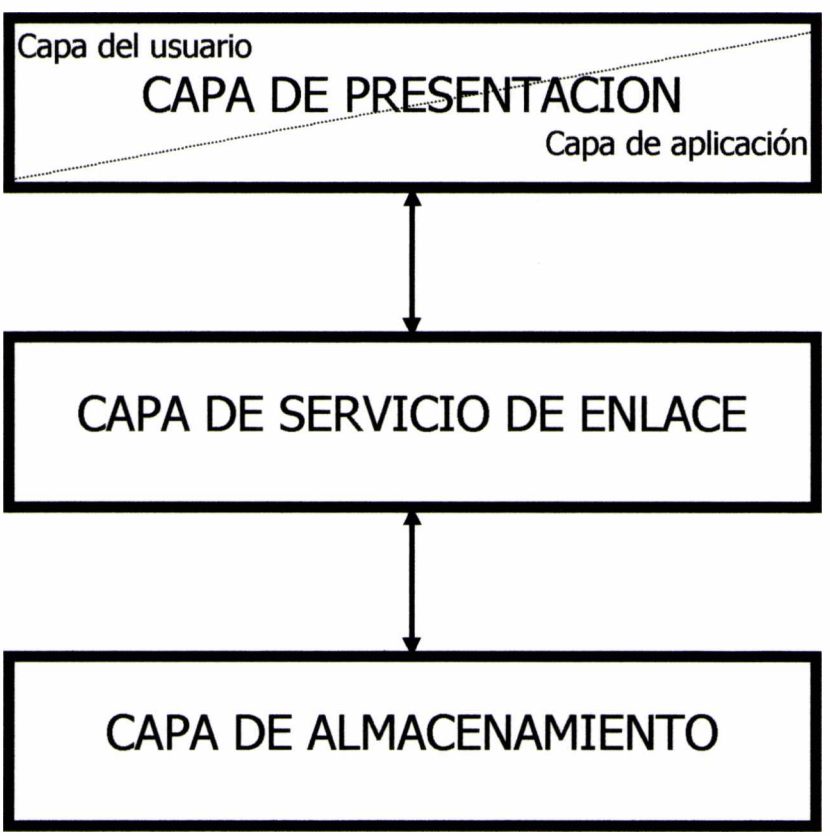
7. Clickear el botón OK con lo cual queda creado el enlace.

Se puede usar la opción **COMPUTE LINK** para encontrar todos los ítems relevantes que igualen su criterio y hacer enlaces a ellos.

2.6.5. Arquitectura de Microcosmo

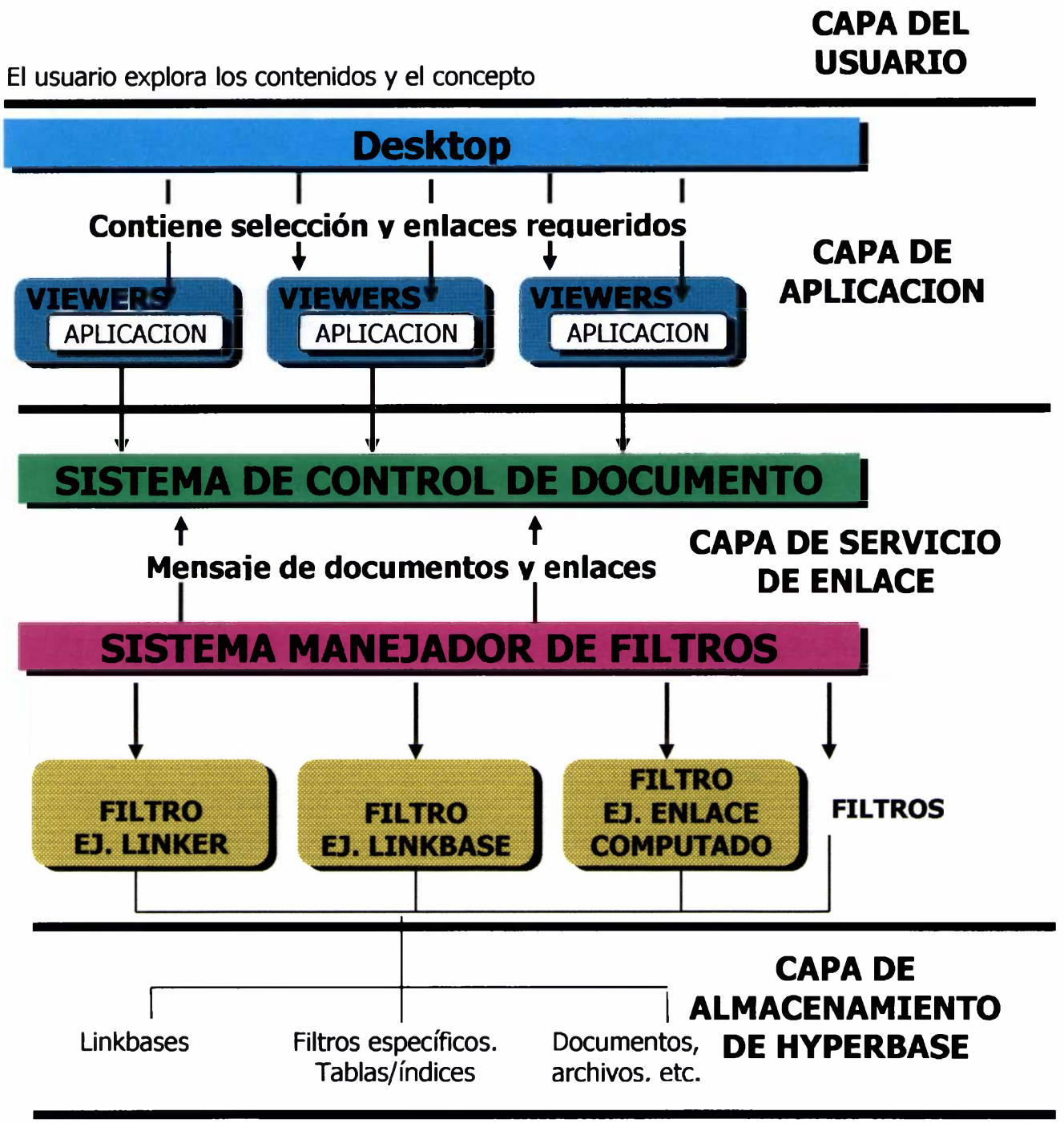
El Sistema Microcosmo comprende cuatro capas:

- *Capa de Usuario*: ofrece al usuario acceder a la funcionalidad de Microcosmo y a los documentos y enlaces en el sistema.
- *Capa de Aplicación y sus Viewers*: integra las aplicaciones del usuario al ambiente de hipermedia de Microcosmo.
- *Capa de Servicio de Enlace*: implementa las acciones de hipertexto.
- *Capa de Almacenamiento de hiperbase*: es el almacenamiento permanente para los recursos de hipermedia, documentos y enlaces.



Microcosmo consiste de un número de procesos autónomos que se comunican entre sí por un sistema de pasaje de mensajes.

Los documentos de todos los tipos son referenciados en sus formatos de aplicación original. Esto difiere de las aplicaciones de hipertexto tradicionales que incorporan la información de enlace en versiones de documentos especializados y marcados.



Microcosmo tiene su información de enlaces en objetos llamados LINKBASE (colecciones de enlaces individuales). Cada enlace consiste de ítems tales como un anchor origen (punto de partida del enlace), un anchor destino (destino del enlace) y atributos tales como la descripción del enlace. Esto ofrece tres características importantes:

- Permite que más de una linkbase sea aplicada a una colección de documentos dados.
- Permite hacer enlaces a medios de sólo lectura.
- Permite enlazar tanto procesos como documentos.

Muchos sistemas de hipertexto soportan una acción única, presionar un botón causa que un enlace sea "seguido". Microcosmo ofrece diferentes acciones que describiremos más adelante. El usuario selecciona el ítem de interés (por ejemplo, una porción de texto) y usa un menú para elegir una acción a seguir. Para Microcosmo un botón es un enlace de una selección específica con la acción particular FOLLOW LINK.

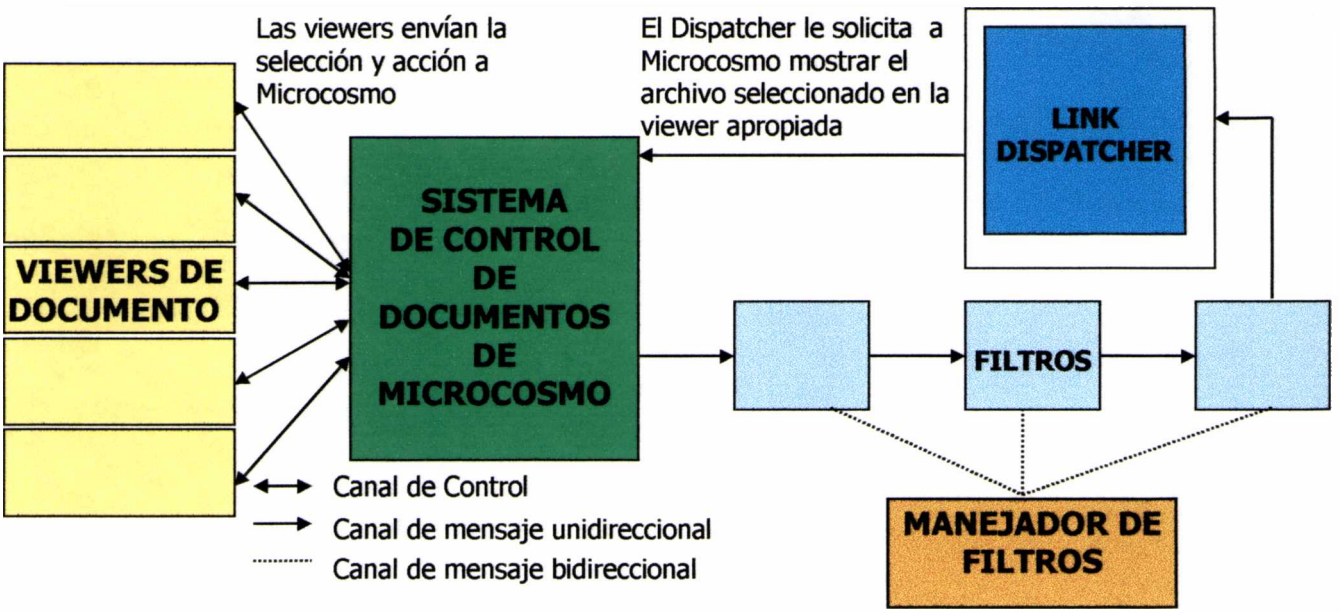
Microcosmo ha generalizado el concepto de anchor origen para ofrecer varias posibilidades de enlace desde anchors definidos explícitamente a anchors generados dinámicamente.

En Microcosmo el usuario ve una interface que consiste de sus aplicaciones standard, la coordinación de éstas, está dada por el SISTEMA DE CONTROL DE DOCUMENTOS (DCS). El DCS actúa como un mediador del SISTEMA DE MANEJO DE FILTROS (FMS) que provee la funcionalidad de hipertexto. Como las interacciones de usuario con las aplicaciones generan acciones de hipertexto, los mensajes son enviados al DCS y luego al FMS. Esto a su vez puede activar enlaces, lo que lleva al FMS a enviar mensajes de vuelta al DCS instruyéndolo para que abra nuevos documentos.

El DCS simplemente, implementa un protocolo abierto para la coordinación de vistas de documentos, el FMS no implementa funciones de hipertexto, pero ofrece un protocolo similar para el control de procesos pequeños o filtros que implementan funciones particulares. Esto permite que la funcionalidad de Microcosmo crezca sin restricciones.

En Microcosmo el usuario interactúa con una VIEWER. Una viewer es cualquier aplicación en la que el dato puede ser mostrado. Los mensajes para ejecutar acciones son enviados desde la viewer de Microcosmo a través de una CADENA DE FILTROS. Cada uno de estos filtros tiene la oportunidad de responder al mensaje bloqueándolo, pasándolo o cambiándolo antes de pasarlo. Basado en los contenidos del mensaje, algunos filtros pueden agregar nuevos mensajes a la cadena.

Luego los mensajes pasan de la cadena de filtros al LINK DISPATCHER. Este examinará los mensajes para ver si ellos contienen algunas de las acciones disponibles (tal como enlaces a seguir) y si es así ofrecerá esas acciones al usuario.



2.6.6. Viewers

Una viewer en Microcosmo es cualquier programa que sea capaz de mostrar algún formato de datos y que puede comunicarse con el servicio de enlace de Microcosmo. Este requerimiento para comunicarse con el servicio de enlace ha sido un problema para los sistemas que intentan usar las aplicaciones de terceras personas como viewer, por ejemplo el servicio de enlace de Sun, requiere que todas las aplicaciones que quieran usar el servicio de enlace sean modificadas para llegar a tener "conocimiento del servicio de enlace".

Una viewer en Microcosmo debe:

1. Proveer un menú pull down, ofreciendo opciones como: Follow link, Compute link, Start link y End link.

2. Empaquetar un objeto seleccionado por el usuario (como una porción de texto) y una acción elegida del menú y comunicar este mensaje a Microcosmo.

3. Ser capaz de resaltar ciertos objetos los que pueden ser tratados como botones.

4. Dar la posibilidad de iniciar la aplicación con un conjunto de datos dados tal que la aplicación es mostrada en un estado específico, tal como un ítem de datos particular en pantalla.

Para alcanzar todas estas funciones debemos tener acceso al código fuente de la aplicación, pero esto no siempre es posible. Por esta razón en Microcosmo las viewers son clasificadas de acuerdo al nivel de conocimiento que Microcosmo tiene sobre ellas:

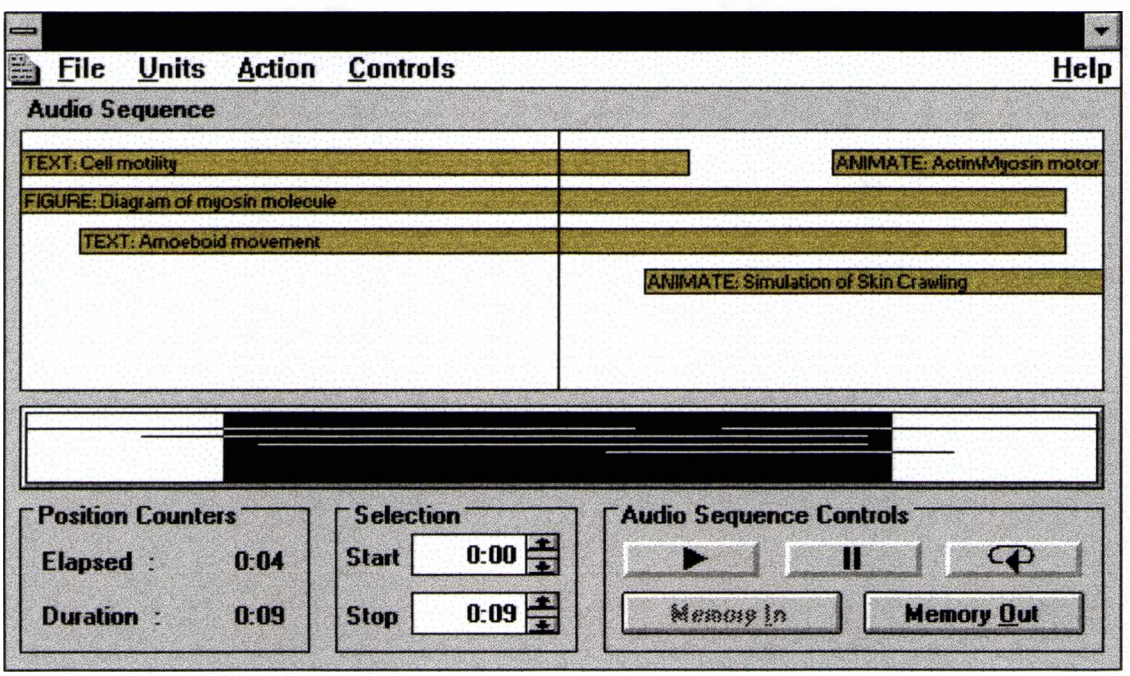
- ***Viewers de Microcosmo totalmente conocidas***: hay viewers que están totalmente integradas con Microcosmo. Este tipo de viewers han sido escritas explícitamente para comunicarse con Microcosmo. Existe un canal de mensajes

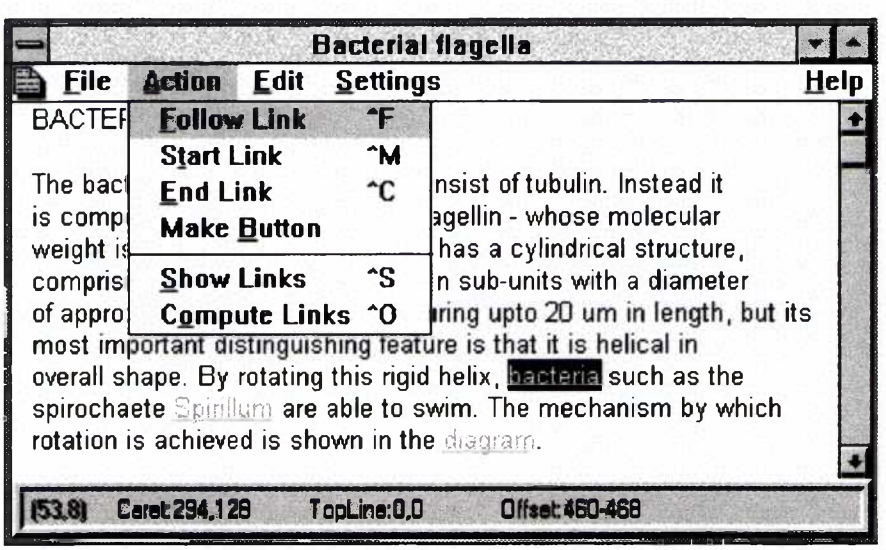
bidireccional entre la viewer y Microcosmo, permitiendo a la viewer emitir pedidos y recibir respuestas.

El código fuente de la viewer está disponible y ha sido modificado para incluir una llamada al API (Interface de Programa de Aplicaciones) de Microcosmo. Esta interface abierta permite la creación de mensajes de acuerdo a un protocolo de comunicación flexible y permite que la viewer se comunique directamente con Microcosmo. Se escribieron viewers para tratar con formatos de datos comunes al ambiente como texto, mapas de bits, video, audio, etc.

Una viewer de Microcosmo puede mostrar áreas activas como botones. Estas son combinaciones de selección y acción de objetos que están de alguna forma resaltados, y la información acerca de la selección está almacenada en las linkbase. La viewer pide esta información cuando carga los datos. Las viewers de Microcosmo también pueden ser solicitadas para empezar con el foco en algún punto particular en los datos.

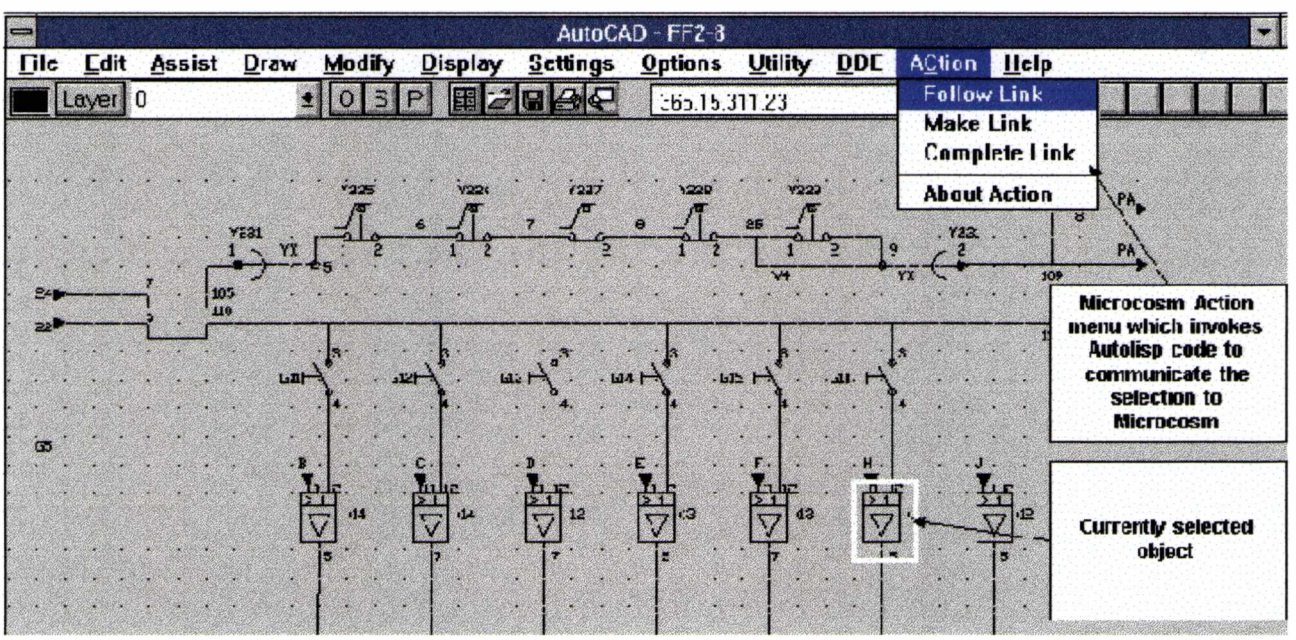
Las siguientes figuras son ejemplos de este tipo de viewers:



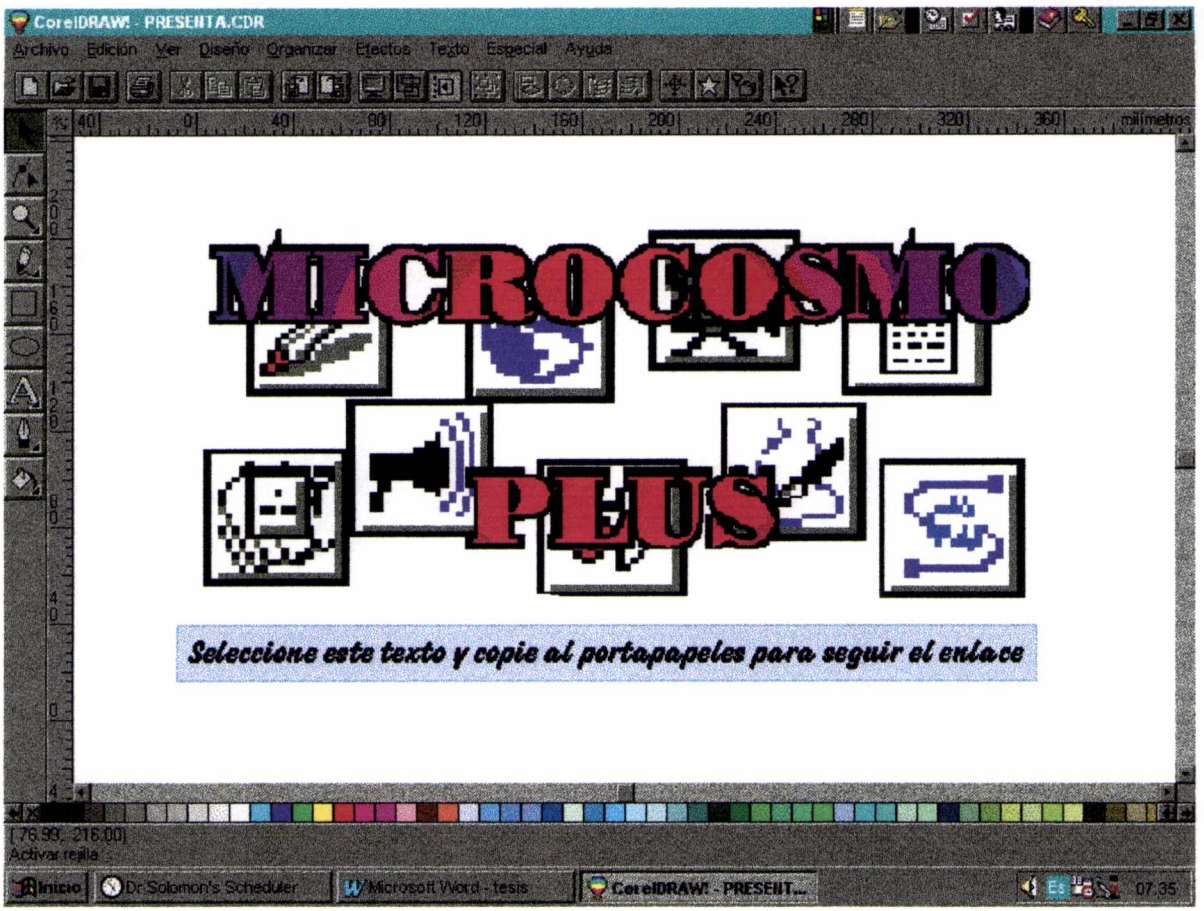


- ***Viewers de Microcosmo parcialmente conocidas***: estas son aplicaciones de terceras personas que han sido adaptadas para ser reconocidas por Microcosmo. Muchos paquetes como Word para Windows, Toolbook y Access tienen algún nivel de programabilidad. En estas aplicaciones es muy sencillo escribir el código necesario para producir un menú de acción y empaquetar una selección y acción en un mensaje para despacharlo a Microcosmo. El proceso para adaptar esas aplicaciones es diferente de reescribir la aplicación para que el enlace sea recorrido.
  Microcosmo es capaz de controlar qué documento puede abrir la aplicación. Además el usuario es capaz de seguir o crear enlaces genéricos desde el interior de ellos, pero la viewer no es capaz de una comunicación total con Microcosmo ya que no puede recibir mensajes.

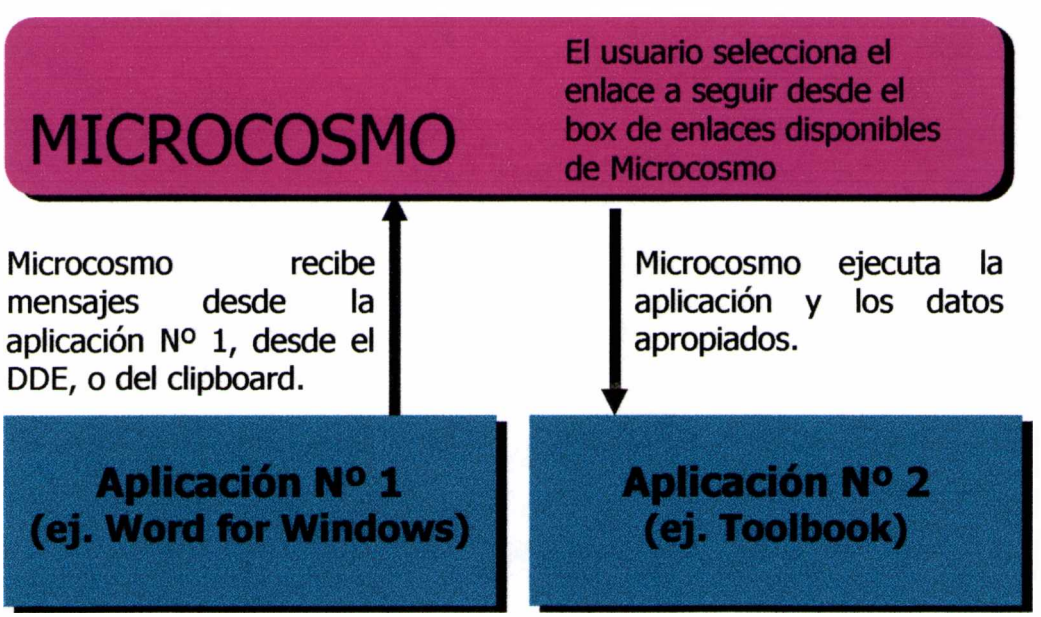
Estos tipos de aplicaciones utilizan el DDE (Dynamic Data Exchange). Los scripts o macros deben ser creados por la aplicación para manejar el DDE, pues ésta no es una característica de Microcosmo.



- ***Viewers de Microcosmo desconocidas***: son aplicaciones que no tienen comunicación directa con Microcosmo. Es el peor de los casos, donde no es posible construir en la viewer alguna forma de menú de acción, se introdujo la idea de "clipboard links" (enlace del portapapeles). El usuario hace una selección desde una aplicación en la forma normal y luego copia la selección en el portapapeles. Puede ser construido un menú de acción desde el ícono de Microcosmo y éste es responsable de tomar el contenido del portapapeles y la acción elegida y empaquetarlos en un mensaje. Un refinamiento de esta aproximación permite al usuario pedir a Microcosmo "monitorear el portapapeles": siempre que el portapapeles contenga cambios, Microcosmo automáticamente empaquetará el nuevo contenido con una acción preseleccionada tal como follow link.



Hay algunos problemas con esta aproximación. La interface al menú de acción no es uniforme para todas las aplicaciones, y las viewers parcialmente conocidas y desconocidas no serán capaces de proveer información acerca de la posición exacta en un documento en el que fue seleccionado el dato, puede no ser posible crear enlaces específicos desde dichos documentos. Sin embargo son posibles enlaces locales y genéricos y proveer al usuario con funcionalidad de hipermedia desde cualquier aplicación.



Las viewers actualmente incluidas en Microcosmo son:

- ✓ TEXT - RTF
- ✓ AUDIO
- ✓ ANIMATIONS
- ✓ BITMAPS
- ✓ VIDEO
- ✓ MIMICS (TOURS GUIADOS)

Algunos programas que presentan formatos de datos al usuario pueden ser conectados a Microcosmo como una viewer. La versión 3 de Microcosmo es capaz de usar Word para Windows y Toolbook , permitiendo conectar otros programas como viewers (por ejemplo una Web browser para archivos HTML). Ejemplos de productos que han sido conectados de esta forma son:

- ✓ PROCESADOR WORD.
- ✓ WEB BROWSERS.
- ✓ PAQUETES CAD/CAM.
- ✓ BASES DE DATOS.
- ✓ SPREADSHEETS.
- ✓ PAQUETES DE INDUSTRIA.

La habilidad para integrar aplicaciones de terceras partes " conocidas de Microcosmo" o usar el portapapeles para la comunicación significa que Microcosmo es más que otro sistema de hipermedia. Pueden ser visto como una forma de integrar las aplicaciones que constituyen el ambiente de trabajo existente, más que agregar desorden al escritorio.

2.6.7. Protocolo de Comunicación de las Viewers

Una viewer totalmente conocida es la que puede participar en todos los protocolos de comunicación de Microcosmo. Sin embargo es posible que participe sólo en un subconjunto del conjunto de protocolos.

- *Documento Launch*: debe ser posible ejecutar un programa viewer con un conjunto de datos dados cargados desde el código externo. Este protocolo es pre-requisito para proveer destinos a enlaces de hipertexto.

- *Mostrar botones*: una viewer totalmente conocida, en cuanto comienza, envía mensajes a Microcosmo pidiendo detalles de algunos botones (los enlaces específicos están resaltados en alguna forma). Microcosmo responde enviando la información del botón que la viewer usará para repintar la pantalla. Sin embargo si la viewer no hace pedido de botones, la información no será enviada y los botones no serán pintados. Esto no significa que los enlaces no estén disponibles sino que el usuario es responsable de seleccionar el anchor origen y elegir el FOLLOW LINK desde el menú de acción. Esto significa que podríamos tener nodos de hipertexto sin botones.

- *Opciones start-up*: cuando una viewer arranca es conveniente que indique el anchor destino de alguna forma. En el caso de un documento de texto podría significar recorrer el documento ya que el texto anchor está en la línea superior, en un video significa moverse al cuadro correcto, en un dibujo significa resaltar un objeto importante.

- *Chequeo de integridad del enlace*: cuando una viewer muestra un documento es conveniente chequear que algunos desplazamientos usados en el documento para describir posiciones de anchors de enlaces se correspondan con los objetos requeridos. Esto puede ser logrado pidiendo que las Linkbase provean una lista de todos los enlaces que contiene, especificando anchor origen y destino en el documento actual. Si estos enlaces son fechados luego de la última vez que el documento fue editado entonces serán seguros, pero si el documento ha sido editado después de haber creado algún enlace entonces la viewer deberá realocar el anchor, si puede, y actualizar la linkbase.
  Si la viewer no es capaz de comunicarse con la linkbase es posible que los anchors sean alocados incorrectamente. Las posibles soluciones a este problema son:
  1. Hacer al documento de sólo lectura, así éste no puede ser editado. Si debe ser editado se produce otra versión.
  2. No usar anchor de enlace específico. Todos los enlaces serán genéricos o locales y tendrán sus destinos al principio del archivo más que dentro de él.
  3. Usar una máquina de búsqueda para encontrar anchor de enlace más que información de desplazamiento/posición. Si el anchor es único en el archivo, y el anchor mismo no es cambiado, será posible editar el archivo.

- *Servicios de acciones de usuario*: Una vez que el usuario empieza a interactuar con el sistema hará acciones de elección y selección del menú (o click en los botones que están disponibles). La viewer es responsable de:
  1. Proveer un menú de acción.

2. Identificar la selección hecha.
3. Identificar (en alguna forma apropiada para la aplicación) la posición de la selección.
4. Identificar el archivo de datos actual.
5. Empaquetar todo lo anterior en un mensaje y enviarlo a Microcosmo.

Si la aplicación no es capaz de identificar la posición de la selección, entonces Microcosmo sólo permitirá la creación de enlaces locales y genéricos; los enlaces específicos serán imposibles.

2.6.8. Sistema de Control de Documentos (DCS)

Una aplicación de Microcosmo consiste de un gran número de documentos (que pueden ser texto, gráficos, animación, video, sonido u otros tipos) almacenados como archivos. Esos documentos han sido importados y registrados en Microcosmo tal que sus características son almacenadas en bases de datos conocidas como Sistema de Control de Documentos. Los datos de cada documento (atributos) incluyen:

- Su localización dentro de la estructura lógica. Un documento puede estar en más de una parte de la estructura lógica de archivos.

- Un nombre describiendo su contenido. Esos nombres son mostrados cuando el documento es visto.

- Su localización y nombre de archivo real (DOS), es decir el nombre del camino completo, incluyendo la letra del drive.

- El tipo de documento, el cual es usado para identificar la viewer que se usará para mostrar a ese documento.

- El nombre de su/s autor/es y la fecha en que fue importado.

- Palabras claves que pueden ser usadas para describir el contenido del documento.

Las dos últimas son opcionales.

Cuando un documento es importado a Microcosmo, éste no es cambiado en ninguna forma. La información en el DCS actúa como un enlace entre Microcosmo y el documento original. Si un documento es removido del DCS, el archivo original no sufre cambios.

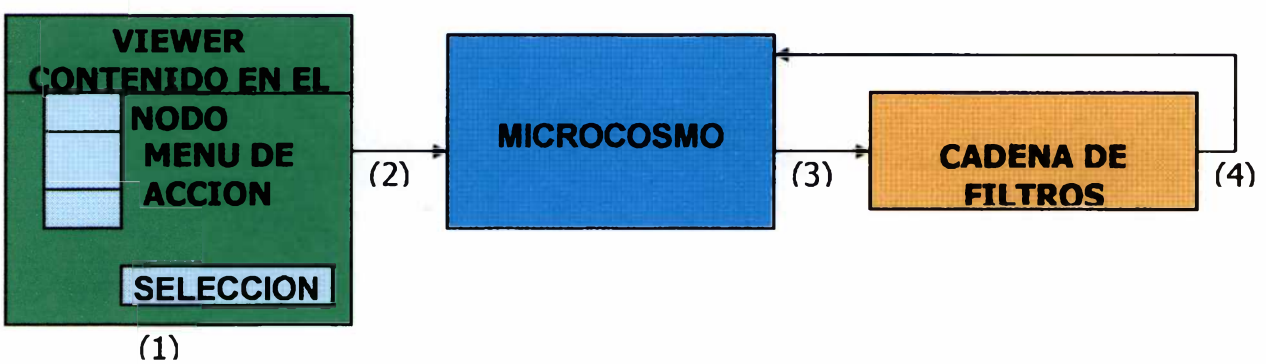
Sólo hay un DCS para todos los documentos importados a Microcosmo, aunque haya varias aplicaciones. Se pueden ver todos los documentos del DCS usando la ventana SELECT A DOCUMENT.

El DCS también es conocido como Docuverse: Universo de Documentos importados a Microcosmo.

2.6.9. Servicio de Enlace

El modelo Microcosmo provee un servicio de enlace que se usa para seguir enlaces dentro y fuera de aplicaciones que no son parte del modelo. Muchos paquetes de aplicaciones Windows tienen facilidades que permiten comunicación programable con el Intercambio de Datos Dinámicos (D.D.E.) y en este caso es posible tratar a dicha aplicación como una viewer de Microcosmo ya que son aplicaciones parcialmente conocidas.

El sistema Microcosmo asume que el nodo que contiene la viewer es capaz de mostrar un menú de acción que contiene ítems como FOLLOW LINK, COMPUTE LINK, START LINK y END LINK y es capaz de comunicar la acción elegida junto con los detalles del objeto seleccionado, la posición de éste y el archivo en el que fue encontrado a Microcosmo. Este proceso luego se ocupa de pasar este mensaje directamente a un conjunto de programas conocidos como FILTROS, los que responden al mensaje como muestra la siguiente figura:



1) El usuario hace la selección y elige una acción del menú por ejemplo FOLLOW LINK.

2) La viewer pasa el mensaje a Microcosmo, el que contiene la selección, el archivo fuente, la acción elegida y si es posible la posición de la selección.

3) Microcosmo pasa el mensaje a los programas filtros, los que deciden que acción tomar, por ejemplo que enlaces están disponibles.

4) Los destinos del enlace seleccionado por el usuario son enviados a Microcosmo que carga los datos y viewers apropiados.

Los usuarios interactúan con la información presentada en las viewers, los mensajes que se generan son enviados al SISTEMA DE CONTROL DE DOCUMENTOS (DCS) quien maneja los procesos de visualización de los documentos. El DCS pasa los mensajes al SISTEMA MANEJADOR DE FILTROS (FMS), el cual provee funcionalidad de hipertexto, a través del uso de módulos conocidos como FILTROS.

El FMS se encarga de enviar los mensajes a todos los filtros activos. La implementación del FMS organiza los filtros disponibles en una estructura de cadena. Cuando el FMS recibe un mensaje, este pasa al primer filtro en la cadena y a medida que los mensajes son retornados al FMS por los filtros, son pasados al siguiente filtro en la cadena.

Cuando un filtro recibe un mensaje lo chequea y decide si lo debe procesar, tomar una acción o retornar el mensaje al FMS. Cuando los mensajes llegan al final de la cadena, el FMS los retorna al DCS, que los pasará dentro de una viewers, si es necesario.

El Modelo Microcosmo permite varios tipos de enlaces:

- ***Botones :*** creados manualmente desde un punto origen a un punto destino. Son enlaces desde un anchor visible en un documento fuente a un lugar específico en un documento destino. Los documentos de animación o bitmap sólo soportan este tipo de enlace. Estos pueden ser usados en situaciones donde el enlace sería obvio, por ejemplo un enlace desde un documento de texto a una foto o ilustración asociada.

- ***Específicos :*** Son enlaces desde un anchor invisible en un documento origen a un lugar específico en un documento destino. Estos pueden ser usados en situaciones donde los enlaces proveen información detallada acerca de un tema particular.

- ***Locales :*** son enlaces desde un anchor/s invisible/s en un documento origen a un lugar específico en un documento destino. Supongamos que la palabra *casa* es seleccionada como el anchor origen y se hace un enlace a una figura de una casa. Si se selecciona el tipo de enlace local, cuando el enlace es creado, todas las ocurrencias de la palabra *casa* en el documento origen (solamente) son enlazadas a la figura. Estos pueden ser usados en lugar de un enlace genérico, cuando el enlace desde una palabra o frase necesita ser restringido por alguna razón. Por ejemplo puede ser usado para que los usuarios sólo puedan obtener ciertos documentos por una búsqueda cuidadosa o para evitar presentarlos con información inapropiada.

- ***Genéricos :*** son enlaces desde un anchor/s invisible/s en ALGUNA porción de texto a un lugar específico en un documento destino. Por ejemplo, si el enlace en *casa* descripto antes fue creado como tipo de enlace genérico, entonces todas las ocurrencias de la palabra *casa* en cualquier documento son enlazadas a la figura. Estos serían usados en situaciones en donde el enlace estaría disponible en toda la aplicación, por ejemplo un enlace a una explicación de un término técnico en un glosario.

- ***Computados :*** es una poderosa herramienta que permite a los usuarios seleccionar algún texto y encontrar rápidamente otros documentos que contienen una alta aparición de palabras significantes en la selección. Esto permite que los documentos relevantes sean encontrados aún cuando no existan enlaces apropiados.
  Hay dos formas de realizar este tipo de enlace. La primera es usar un filtro grep, que intenta unir el texto seleccionado con el mismo texto en cualquier otro documento en algún conjunto pre-definido de documentos de texto y retorna todos los nombres de documentos posibles en el Link Dispatcher. Este es un método lento. Un segundo método requiere la construcción de un índice invertido de todos los documentos deseados antes de usar el sistema y usa recuperación de información standard para comparar el vocabulario del origen seleccionado con el vocabulario de los documentos y ofrecer al usuario los enlaces más parecidos. Este método produce alta calidad de comparaciones en un menor tiempo.

- ***Enlaces a ejecutables***: Microcosmo permite hacer enlaces a archivos ejecutables, usando la opción MAKE BOTON en el menú de acción de las viewers de texto y gráficos (viewers totalmente conocidas). Como el enlace no tiene un documento destino, entonces no aparecerá en la ventana SELECT A DOCUMENT.

Cada enlace en una linkbase está representado por un conjunto de campos marcados. La siguiente figura representa un conjunto de marcas:

\ SourceFile 100. 02. 24. 93.11. 39. 54
\ SourceSelection SPC
\ SourceOffset 214
\ SourceDocType TEXT
\ DestFile 100. 02. 24. 93. 12. 40. 49
\ DestSelection manual
\ DestOffset 312
\ DestDocType TEXT
\ Description SPC Definition

Este ejemplo describe un enlace desde el texto "SPC" en un archivo de texto (que el sistema conoce por un identificador de documento único) y aparece en el carácter 214 del archivo.

Si un mensaje fue enviado a la linkbase pidiendo el Follow link desde este documento origen, conteniendo la selección "SPC" elegida en el desplazamiento de 214 caracteres a través del archivo, luego el documento destino será enviado con el cursor en la línea que contiene el carácter 312 (el comienzo de la palabra "manual"). Llamamos a éste *enlace específico*. En Microcosmo un botón es un enlace específico que es coloreado por la viewer tal que se puede seguir el enlace simplemente clickeando en él.

La ausencia de un SourceOffset en la linkbase indicará que este enlace podría ser seguido en cualquier lugar que el texto "SPC" sea elegido en el archivo origen. Llamamos a éste, *enlace local*.

La ausencia de un SourceOffset y un SourceFile en la linkbase indicará que el enlace podría ser seguido en cualquier lugar que el texto "SPC" haya sido seleccionado. Llamamos a éste *enlace genérico*.

Los desplazamientos en este ejemplo están descriptos en términos de bytes, pero este no es un requisito. La semántica del desplazamiento es decidida por la viewer y puede ser resuelta por la linkbase, con tal que el método para expresar el desplazamiento sea usado consistentemente por la viewer.

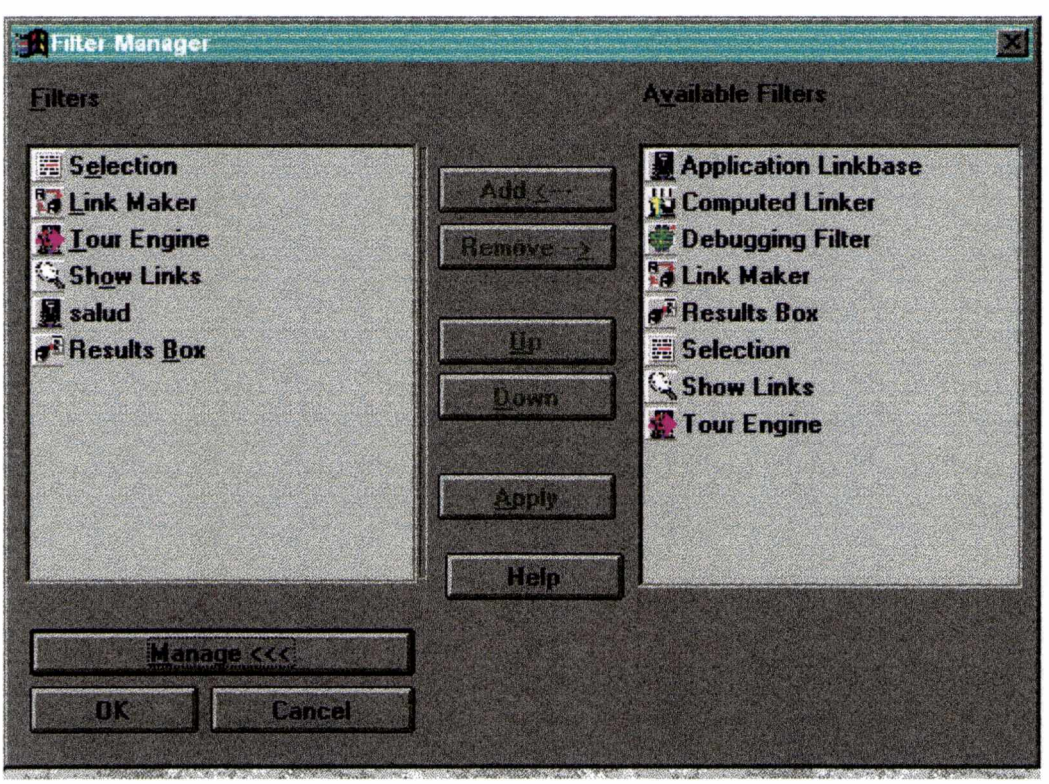
Excepto en el caso en que hay un botón indicando la presencia de un anchor de enlace, es normal que el usuario marque selecciones y luego pida una acción como Follow link. También es posible seleccionar amplias áreas de interés como un párrafo de texto, y el sistema luego encontrará todos los anchors y enlaces disponibles en esta selección.

2.6.10. Sistema Manejador de Filtros (FMS)

El FMS recibe mensajes del DCS y los pasa al conjunto de filtros activos. Cuando un filtro recibe un mensaje examina el tipo y si corresponde toma una acción futura. Por ejemplo un filtro que implementa una base de datos de enlace responderá a un mensaje de FOLLOW LINK con una serie de mensajes que describen algunos enlaces aplicables. Un filtro almacena una historia de acciones de usuario, simplemente registrando detalles de algunos mensajes recibidos.

El FMS ordena los filtros activos en una cadena lineal. Los mensajes son enviados al primer filtro de la cadena y luego los mensajes que son recibidos desde los filtros son pasados al próximo filtro en la cadena. Cuando el mensaje llega al final de la cadena el FMS lo devolverá al DCS.

El FILTER MANAGER permite acceder a los filtros que están actualmente en la cadena. Puede ser usado para agregar o sacar filtros de la cadena, o cambiar el orden en que aparecen.



2.6.11. Filtros

Los filtros son procesos que proveen funcionalidad al usuario y a las viewers. Reciben los mensajes, toman acciones particulares y pueden generar otros mensajes para las viewers u otros filtros. Microcosmo tiene una cadena de filtros que definen los tipos de acción actualmente disponibles.

En el modelo de cadena de filtros el orden en el cual los filtros aparecen, están bajo el control del usuario. También hay un sistema manejador de filtros inteligente. Los filtros pueden ser instalados y removidos para configurar el sistema de acuerdo a la preferencia del usuario.

Hay distintos tipos de filtros:

2.6.11.1. Linkbases :

En Microcosmo, los documentos no están marcados internamente: el dato de enlace está contenido en las Linkbases y las viewers se comunican con éstas para establecer qué botones y enlaces existen a un documento particular. Las Linkbases son bases de datos que tienen toda la información referente a los enlaces entre documentos y consisten de un archivo de texto indexado donde cada enlace está representado por un conjunto de campos. Esta información es agregada automáticamente a la linkbase cuando el enlace es creado. Los detalles almacenados son:

- El fingerprint del documento que contiene el anchor origen. El fingerprint es un número de identificador único asignado automáticamente a todos los documentos cuando son importados al DCS.
- La localización de este anchor en el documento.
- El texto definido por el anchor origen.
- El fingerprint y nombre del documento que contiene el anchor destino.
- La localización de este anchor en el documento.
- El tipo de enlace (botón, específico, local o genérico).

Una característica importante de Microcosmo es la facilidad de instalar filtros dinámicamente, particularmente Linkbases. Se puede instalar más de una linkbases a la vez, haciendo posible proveer diferentes vistas de la información creando dos conjuntos de enlaces en forma separada.

Una linkbases *pública* que contiene todos los enlaces hechos por el autor original y una linkbases *privada* que contiene todos los enlaces hechos por el usuario individual. Esto asegura que los cambios y los enlaces privados no sean visibles en el sistema público.

De esta manera es posible tener un conjunto de documentos multimedia, con diferentes linkbases que podrían contener vistas completamente separadas en el mismo conjunto de información.

2.6.11.2. Show Links

Un documento de texto puede no tener enlaces visibles (botones), esto quiere decir que contiene un número de enlaces invisibles, esto se da especialmente cuando el usuario no usa para leer ese documento la viewer de texto totalmente conocida.

Bajo estas circunstancias el usuario puede tener interés de conocer cuales son los enlaces que están disponibles.

Si el usuario selecciona una parte del texto y usa la acción Show Links, el sistema enviará todas las palabras en el texto y todos los pares de palabras consecutivas del texto seleccionado a través de la cadena de filtros para determinar todos los enlaces disponibles. Por este motivo, en el orden de la cadena debe estar antes de las linkbases.

2.6.11.3. Linker

Éste maneja mensajes para crear un enlace (comenzando en alguna selección origen) y para terminarlo (en algún punto o selección destino). Al tener seleccionado el anchor origen y destino, cuando se clickea en el botón COMPLETE, aparecerá la ventana LINKER que permite seleccionar el tipo de enlace.

2.6.11.4. Filtros Navegacionales

Hay otros dos filtros que son de particular interés en la ayuda para navegar a través de la información.

El mecanismo ***HISTORY***, es un filtro que mantiene una lista de todos los documentos que han sido consultados por el usuario y permite a éste retornar a un documento particular. Las historias pueden ser salvadas al finalizar una sesión.

El mecanismo ***MIMIC***, permite al usuario seguir un camino predefinido a través de los documentos. Tienen la característica que todas las acciones de Microcosmo normales están disponibles mientras se sigue un Mimic. Esto significa que el usuario puede desviarse de este mecanismo predefinido en cualquier momento, mientras sea capaz de retornar al recorrido cuando lo requiera. Puede estar en cualquier lugar dentro de la cadena de filtros.

2.6.11.5. Computed Linker

Responde a la acción COMPUTE LINK. Usa un índice ya creado de algún conjunto de archivos de texto para aplicar técnicas de recuperación de información para tratar de proveer enlaces donde no han sido creados manualmente. Muestra los documentos identificados en la ventana RESULT. En la cadena de filtros debe estar antes del RESULT BOX.

2.6.11.6. Link Maker

Ejecuta todos los procesos para la creación de enlaces. Este filtro puede ser usado directamente para especificar el final de un enlace. Esto es útil si el documento destino tiene una viewer desconocida de Microcosmo (la cual no tiene el menú de acción) o es un documento de sólo lectura (por ejemplo archivos de sonido o animación).

La acción MAKE LINK envía un mensaje a este filtro para definir un anchor de enlace basado en el objeto seleccionado y la acción COMPLETE LINK le envía un mensaje para definir un destino de enlace basado en el objeto seleccionado.

Estará situado cerca del principio de la cadena de filtros, después del filtro SELECTIÓN, pero antes de cualquier linkbase.

2.6.11.7. Selection

Este filtro puede ser usado para las operaciones de seguimiento de enlace sin seleccionar texto en los documentos. Es una buena forma de definir el anchor para un enlace genérico. Debe estar delante de cualquier filtro que use selección de texto (por ejemplo COMPUTED LINKER, SHOW LINKS y LINK MAKER) y de las linkbases.

Si se presiona el botón START LINK, permite hacer enlaces genéricos a un documento destino sin tener que seleccionar el anchor origen en un documento.

2.6.11.8. Result Box

Cumple varias funciones:

- Si una operación de seguimiento de enlace identifica más de un documento destino, el filtro mostrará todos estos documentos en una **Sesión de Enlaces Disponibles** y permite al usuario elegir qué documento desea ver. Esto es equivalente al filtro Enlaces Disponibles de las versiones anteriores de Microcosmo.

- Mantiene una **Sesión Record** de todas las operaciones de seguimiento de enlace, a las que el usuario puede retornar luego. Cada sesión puede volver a verse y ser editada.

- Mantiene un registro, la **Sesión History**, de todos los documentos que han sido mostrados por Microcosmo. El History puede ser salvado y cargado luego en sesiones posteriores de Microcosmo. Esto es equivalente al filtro History de las versiones anteriores de Microcosmo.

Result Box debe estar ubicado, dentro de la cadena de filtros, después de las linkbases y el Computed Linker.

2.6.12. Viewer Universal de Microcosmo

Muchas aplicaciones tienen un lenguaje de programación de macro y algún grado de adaptabilidad para que se puedan agregar nuevas opciones de menú. Por ejemplo las aplicaciones Office de Microsoft cuentan con Visual Basic para aplicaciones, las aplicaciones de la serie Office de Lotus tienen lenguajes de macro y Autocad cuenta con Autolisp. Usando estas facilidades es muy simple agregar el menú de Microcosmo a una aplicación y luego agregar las macros apropiadas para cada acción elegida.

Uno de los principales objetivos de Microcosmo era poder alcanzar la funcionalidad de hipermedia en aquellas aplicaciones donde no resulta fácil la comunicación con otras aplicaciones que son standard en el ambiente de operación del host.

Las versiones anteriores de Microcosmo permitían al usuario establecer una opción que le decía a Microcosmo qué acción ejecutar sobre un dato nuevo que entra al clipboard (usado en el caso de una viewer desconocida). El usuario podía ejecutar algún programa que soporte el clipboard y los datos eran copiados continuamente desde la aplicación, Microcosmo tomaba la acción apropiada tal como FOLLOW LINK o COMPUTE LINK, que sería ejecutada.

Esta aproximación parecía muy poderosa, pero la interface a la funcionalidad era dificultosa y raramente era usada.

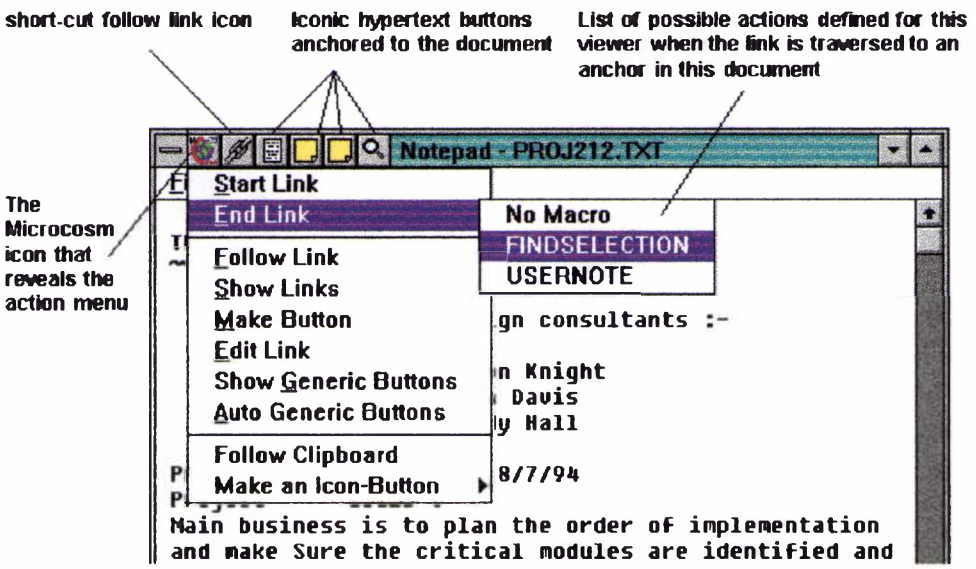
Para mejorar esta interface fue diseñado un programa llamado VIEWER UNIVERSAL (UV). Se mantiene una lista de aplicaciones con las que la UV coopera y la UV monitorea eventos Microcosmo para identificar cuando un programa en esa lista es ejecutado: cuando éste identifica tal evento se prende un pequeño ícono de Microcosmo en la barra de títulos de la aplicación. Cliqueando sobre este ícono se habilita un menú idéntico al menú de acción de cualquier otra viewer. La UV es responsable de todas las comunicaciones con Microcosmo y maneja algunas comunicaciones que son posibles con la aplicación.

Para cooperar con una aplicación, la UV necesitará ser preconfigurada para conocer algunos detalles:

- *Cuando se selecciona FOLLOW LINK u otra acción desde el menú UV ¿Cómo comunicará UV a la aplicación que la aplicación tomará la selección actual?*
  Hay varias opciones. La más común es usar un UVMACRO que es una lista de selecciones de menú que causará que la aplicación tome la selección actual. Otra opción es usar DDE (dynamic date exchange) cuando es soportado por la aplicación o NONE cuando no es posible o la aplicación no tiene el concepto de selecciones.

- *Cuando la selección fue hecha, ¿cómo comunicará la aplicación esa selección a UV?*
  Las opciones son vía el clipboard, vía el DDE, vía archivos de cambios nominados o NONE cuando no es posible.

- *¿Qué opciones estarán disponibles para localizar un anchor destino cuando un enlace es seguido a un documento en esta aplicación?*
  El problema es cambiar la aplicación desconocida a un estado donde muestre el anchor destino, por ejemplo recorriendo un documento de texto hasta que el anchor esté a la vista y luego resaltar el texto entero. Este enlace puede ser creado usando una Macro cuando el archivo destino es cargado. Estas macros pueden ser definidas para cada aplicación. Una macro típica es FINDSELECTION que tomará el texto del anchor destino (desde la información de enlace) y ejecuta el método de búsqueda de la aplicación para encontrar la primera ocurrencia de ese texto.

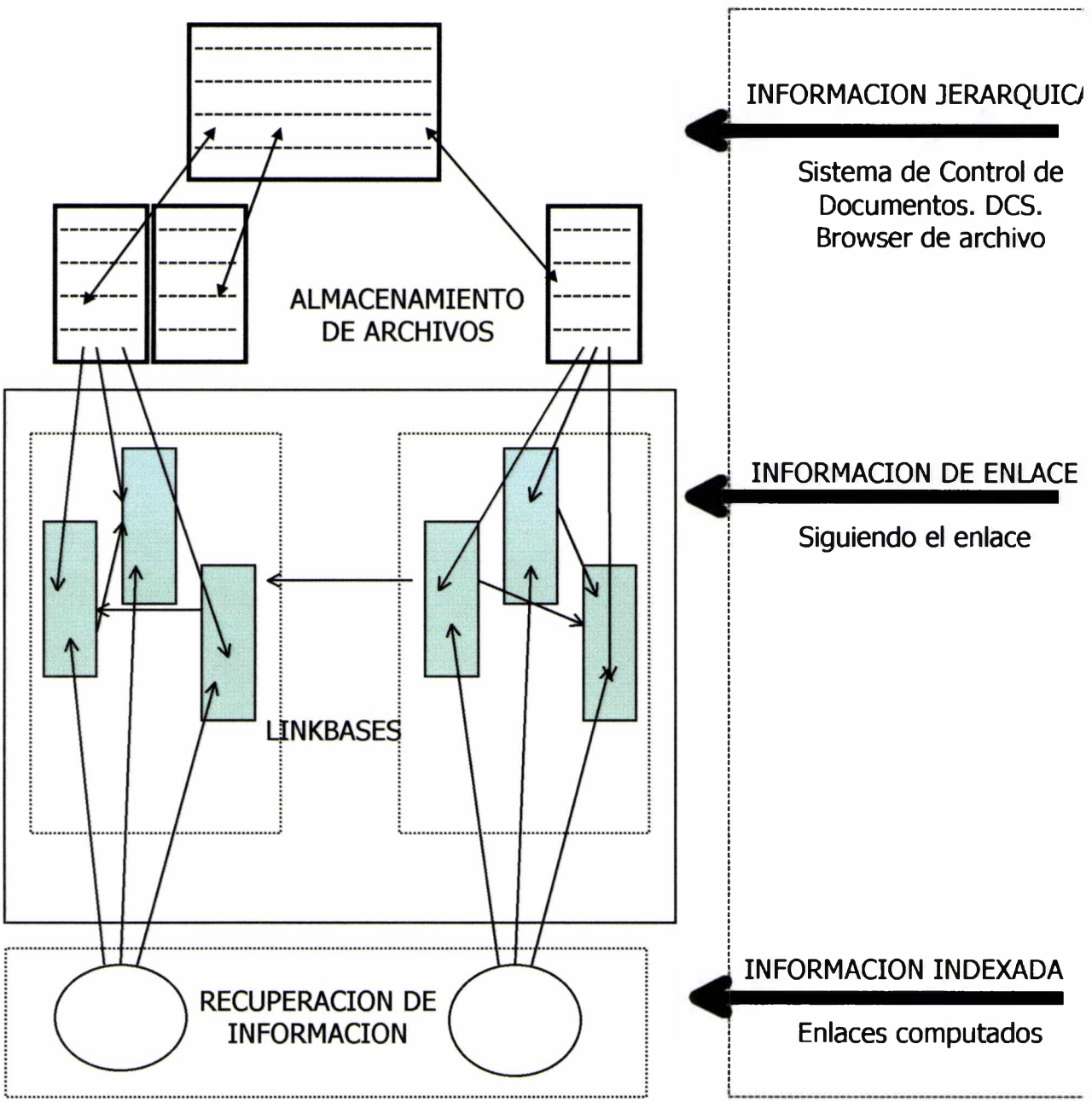


  En el caso donde no es práctico o posible manejar la aplicación en un estado apropiado para mostrar el anchor destino, la única opción será ejecutar la aplicación en su estado de arranque normal. Una opción para mejorar esto es ejecutar la aplicación y al mismo tiempo mostrar un USERNOTE que será un mensaje al usuario instruyéndolo de cómo proceder para que el destino del enlace pretendido sea claro.

En la versión de Microcosmo Plus la metodología para enlazar aplicaciones desconocidas fue modificada. En estos casos la aplicación usada para ver el documento deber ser registrada como una vierwer de Microcosmo. Para definir el anchor origen o cualquier otra acción de seguimiento de enlace, se copia la información seleccionada al clipboard. Luego se accede a la interface de usuario clickiando con el botón derecho del mouse en el ícono de Microcosmo de la barra de tareas. Para definir el anchor destino se hace usando el Link Maker.

2.6.13. Niveles de acceso a la información:

El modelo Microcosmo provee al usuario con distintos modos para navegar a través de la información.



- ***Browser de documentos***: Los usuarios tienden a manejar la estructura de directorios como un método para acceder a la información. Por esta razón se ha introducido un browser de archivo que permite al usuario almacenar y ver atributos de archivo además de ayudar a la navegación y selección de documentos. Estos atributos son palabras claves definidas por el usuario y son accedidas por éste como un método de evaluación de utilidad de un documento particular.

- ***Enlaces creados manualmente***: Son los enlaces específicos, genéricos y locales.

- ***Enlaces computados***: Hay enlaces que no requieren esfuerzo manual y se extienden desde técnicas de búsqueda de strings simples hasta técnicas de recuperación de información completa. Una vez que un enlace ha sido computado es posible seguir inmediatamente el enlace o incorporarlo permanentemente en la linkbase, en cuyo caso éste es indistinguible de un enlace creado manualmente.

Además Microcosmo permite acceder a la información haciendo uso de los mecanismos ***History*** y ***Mimics***.

2.7. EL MODELO MICROCOSMO DISTRIBUIDO

La arquitectura de filtro modular que provee la funcionalidad de hipertexto de Microcosmo permite que el sistema sea fácil y dinámicamente configurable para soportar los requerimientos de un usuario particular. El objetivo de la versión distribuida es proveer esta configuración en un ambiente de red y permitir un nivel complementario de flexibilidad en el modelo distribuido creado.

La definición de un sistema de hipertexto abierto incluye un requerimiento para que los sistemas ofrezcan funcionalidad distribuida en un ambiente heterogéneo. Sin embargo hay varias interpretaciones de funcionalidad e información de hipertexto distribuido. Esto sugiere que se requiera una aproximación flexible y obviamente que sean integradas plataformas heterogéneas. También es un requerimiento básico que esté disponible un protocolo abierto basado en un sistema de comunicación de red.

Otro requerimiento para esta definición es que los sistemas sean capaces de utilizar herramientas existentes en el ambiente del host, como por ejemplo procesadores de textos, planillas de cálculos, esto también se aplica a la incorporación de otros sistemas distribuidos.

Un usuario de un sistema que quiere utilizar las facilidades de los servicios de información distribuida (por ejemplo WAIS y GOPHER) serán capaces de usar sus sistemas de hipertexto particular junto con estas facilidades, por ejemplo crear enlaces **a** y **desde** la información accedida con estas herramientas.

El modelo flexible y abierto que provee Microcosmo es muy adecuado para satisfacer estos requerimientos:

- Los servicios de información pueden ser fácilmente incorporados como filtros o viewers, dependiendo del tipo de funcionalidad que proveen. Por ejemplo las facilidades orientadas a browsing de Gopher podrían ser incorporadas como una viewer. Un sistema basado en consultas como WAIS (Wide-Area Information Server) es más adecuado para incorporarlo como un filtro, esto permitirá una poderosa facilidad de investigación para complementar la creación de enlaces.

- La distribución de la funcionalidad propia de Microcosmo requiere extensiones del sistema existente, sin embargo la arquitectura de sistema modular ofrece una base ideal para proveer dicha extensión. Se puede incorporar fácilmente, el acceso a documentos remotos, ya que todos los accesos son hechos vía el sistema de base de datos de documentos de Microcosmo (DCS).

La versión distribuida de Microcosmo apunta a proveer un rango flexible de configuraciones basadas en el proceso de filtros como una unidad de distribución. Esto permite la flexibilidad de configuración que provee el modelo de filtros para ser heredado por la versión distribuida del sistema. Por ejemplo permitir un subconjunto particular de la configuración de filtros local que esté disponible a sistemas remotos, o a la inversa, asimilar los filtros remotos en el conjunto de filtros local.

Como el FMS maneja el ruteo de mensajes a los filtros es el lugar lógico para localizar las extensiones que se requieren para manejar un sistema donde los filtros pueden ser accedidos desde sistemas remotos.

Para proveer tal funcionalidad, el FMS es extendido tal que estos filtros locales puedan ser "públicos", permitiendo a otros sistemas utilizarlos y así estos filtros públicos remotos pueden ser conectados al sistema local. El FMS entonces puede usar un protocolo bien definido para transferir mensajes a sistemas remotos cuando sea necesario. Cada FMS debe ser capaz de recibir mensajes desde sistemas remotos para responder a mensajes enviados a un filtro dentro de este sistema, o responder a un mensaje que será entregado a un filtro público en la configuración del filtro local.

Un sistema como éste permite que los enlaces y otras formas de funcionalidad de hipertexto sean accedidos desde un sistema remoto. Esta actividad probablemente lleve referencias a documentos almacenados en otros sistemas. Por lo tanto se requerirá una extensión complementaria al sistema para permitir que dichos documentos sean recuperados.

Con múltiples usuarios compartiendo múltiples filtros, surgen nuevos problemas en seguridad (por ejemplo autorización de usuario), configuración para usuarios y bloqueo de bases de datos de enlaces.

El DCS de Microcosmo ofrece una forma abstracta de acceso al sistema de archivos, por lo tanto éste es el lugar lógico para que sea incorporado el acceso a documentos distribuidos. Además para determinar si un identificador de documentos se refiere a un archivo local o remoto, este identificador es extendido para incorporar detalles del sistema host. Esto permite que el intento de acceso a archivos remotos sea detectado fácilmente y favorece el uso de un protocolo standard.

El filtro es el elemento básico de distribución para Microcosmo. En el caso más simple, dos usuarios pueden colaborar a través del uso de linkbases compartidas, cada uno proveyendo al otro de enlaces a anotaciones y documentos personales.

A través de este sistema es posible un amplio rango de modelos distribuidos, como se describe en la siguiente tabla:

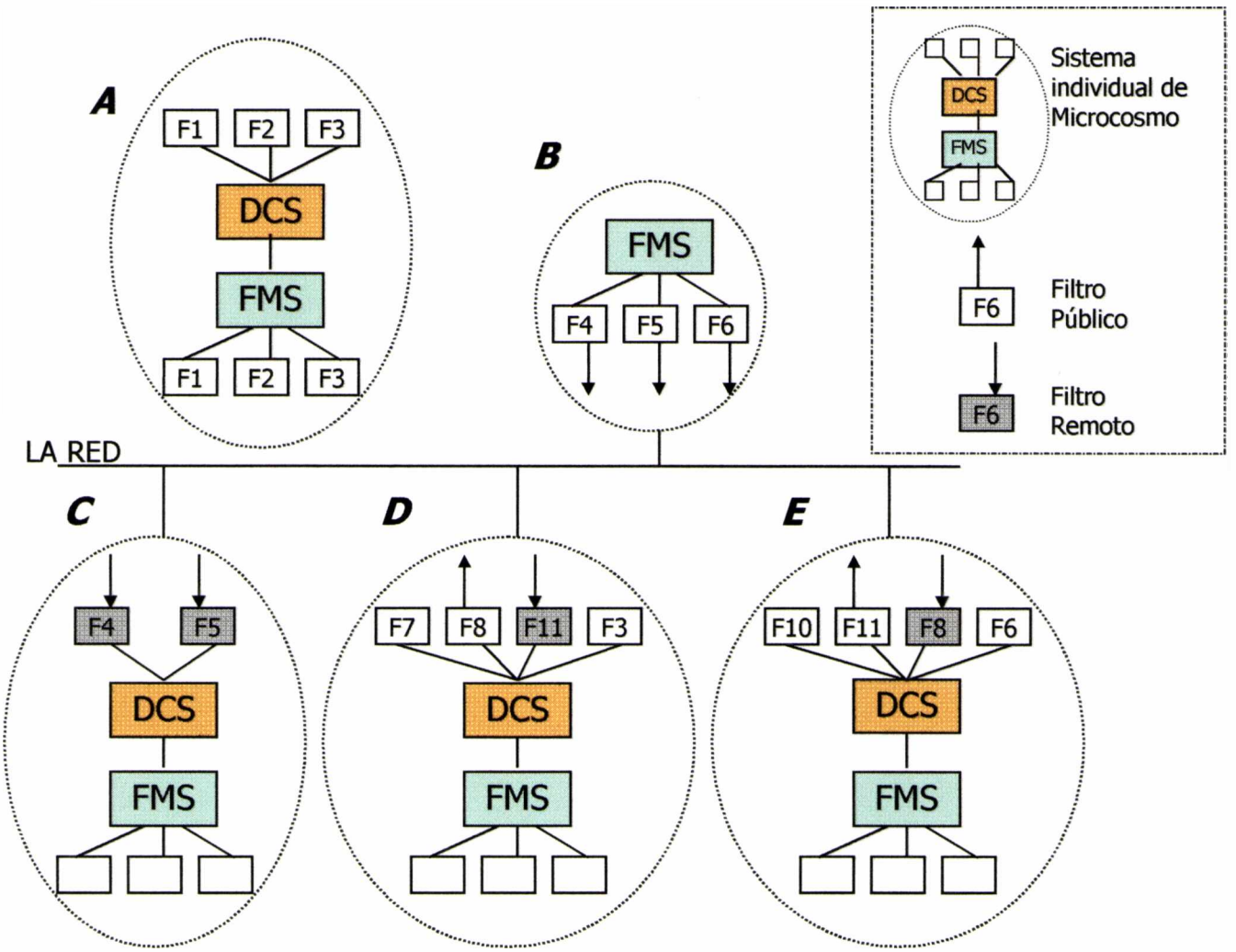
Modelo distribuido para el sistema local	Cómo opera el sistema
Sin distribución	El sistema opera como un sistema de usuario único, sin interacción con sistemas remotos.
Sólo clientes	Opera puramente como un cliente con toda la funcionalidad de hipermedia provista por un server (s) remoto (s).
Sólo server	Alternativamente el sistema daría toda su funcionalidad disponible a sistemas remotos y la interacción local con el sistema estaría deshabilitada, operando puramente como un server.
Interacción de igual a igual	El sistema puede combinar la funcionalidad del cliente y del server con la misma configuaración. Esto crea una red que consta de sistemas iguales.

Los sistemas basados en componentes de cliente y servidor distintos, como WWW o el Servicio de Enlace de Sun pueden ser fácilmente simulados configurando los sistemas de Microcosmo particulares como un sólo-server o sólo-cliente. Similarmente es posible para Microcosmo actuar como un sistema Stand-alone.

La posibilidad de contar con distintas configuraciones permite más flexibilidad que las configuraciones cliente-servidor estrictas provistas por otros sistemas de hipertexto distribuido, ofreciendo muchas formas de colaboración entre usuarios y grupos de trabajo.

Además como todas estas configuraciones son posibles dentro del mismo sistema y cada una esta basada en un protocolo abierto bien definido, todas las variaciones de configuraciones pueden co-existir. Así cualquier sistema particular puede ser configurado en tal forma para satisfacer su propósito, mientras no se excluya su uso para otros sistemas que son utilizados en configuraciones alternativas.

La siguiente figura muestra las posibles configuraciones y como son capaces de interactuar:



El sistema A muestra a Microcosmo operando en una forma Stand-alone como fue posible en versiones previas a este sistema. El sistema B muestra a Microcosmo actuando como un simple server de hipertexto sin interacción local. El sistema C muestra a Microcosmo actuando puramente como un cliente, obteniendo toda la funcionalidad de hipertexto desde filtros remotos (en este caso provistos por el sistema B). Finalmente, los sistemas D y E muestran dos sistemas operando en forma de igual a igual, comparten filtros para permitir la cooperación entre los sistemas, además de incorporar filtros locales y filtros remotos desde otros sistemas de Microcosmo.

2.7.1. Servicio de Enlace Distribuido

Al evolucionar el modelo, se logró que el servicio de enlace esté totalmente separado del servicio de documento: esto es lo que se llama Servicio de Enlace Distribuido (DLS).

El DLS permite al cliente conectarse a un *server de enlace* para solicitar un conjunto de enlaces para un dato en el documento.

Un server de enlace entrega una linkbase que es cargada al cliente y luego éste maneja la resolución de enlaces localmente. El server de enlace es un server de linkbase y entregar linkbases es análogo a entregar documentos. La evolución final del modelo es permitir server de enlaces múltiples para satisfacer un pedido de enlace.

Hay dos aspectos importantes en la implementación del DLS : la herramienta de interface del lado del cliente y las utilidades de bases de datos de enlace del lado del server. Toda la comunicación entre el cliente y el server es provista usando protocolos WWW standard.

El cliente DLS usa un cliente WWW standard para enviar pedidos de enlace codificados como URLs. Estas permiten al cliente acceder a un server de enlace que opera vía un server WWW.

2.7.2. Server de enlace:

El server del DLS, primero fue implementado como script CGI (programas invocados en el server en respuesta a consultas) y accedidos usando un server WWW standard. Los scripts principales son aquellos que permiten la creación, seguimiento y edición de enlaces que son almacenados en bases de datos de enlaces. Luego fue codificado un server DLS dedicado, el que provee una solución más eficiente, corre sobre un número de port TCP/IP diferente a un server HTTP, lo cual le permite cohabitar con un server de documentos en un mismo host.

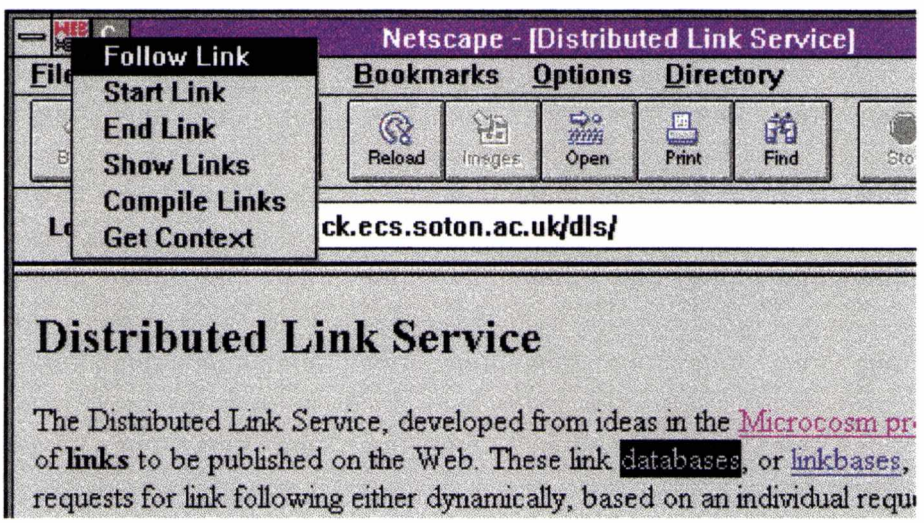
Cuando un software de server de enlace recibe un pedido de FOLLOW LINK para una selección en un documento, el script FOLLOW LINK junta varias bases de datos de enlace para satisfacer el pedido, éstas son:

- Una *linkbase específica del documento* que contiene enlaces que pertenecen sólo a ese documento.
- Una *linkbase específica del recurso* que contiene enlaces que son pertinentes a un grupo de archivos (probablemente una jerarquía de directorios completa).
- Una *linkbase específica del server* que contiene enlaces que pueden ser relevantes para cualquier archivo provisto por el server.
- Una *linkbase específica del usuario* que contiene enlaces creados por el usuario que emitió el pedido.
- Una *linkbase específica del contexto* que contiene enlaces relacionados a una tarea específica, la cual mediante la interface de usuario le permite al lector elegirla en cualquier momento durante una sesión.

El script que maneja pedidos de enlaces FOLLOW LINK simplemente chequea todas las bases de datos apropiadas para cualquier enlace aplicable y los retorna al usuario como una lista de enlaces disponibles en un documento HTML generado automáticamente.

El usuario puede ir directamente a los destinos de los enlaces.

El pedido SHOW LINKS es un caso especial del proceso FOLLOW LINK, tomando una amplia selección y realizando distintos pedidos de FOLLOW LINK sobre las palabra y frases que componen la selección.



El script CREATE LINK acepta detalles de los puntos inicial y final de un enlace e inserta un enlace en la base de datos para el contexto especificado o en la base de datos personal del usuario si no está en tal contexto.

Finalmente, un script está disponible para compilar los enlaces contenidos en estas bases de datos en un documento HTML, así se provee acceso directo a todos los enlaces aplicables sin la necesidad de hacer pedidos individuales al server.

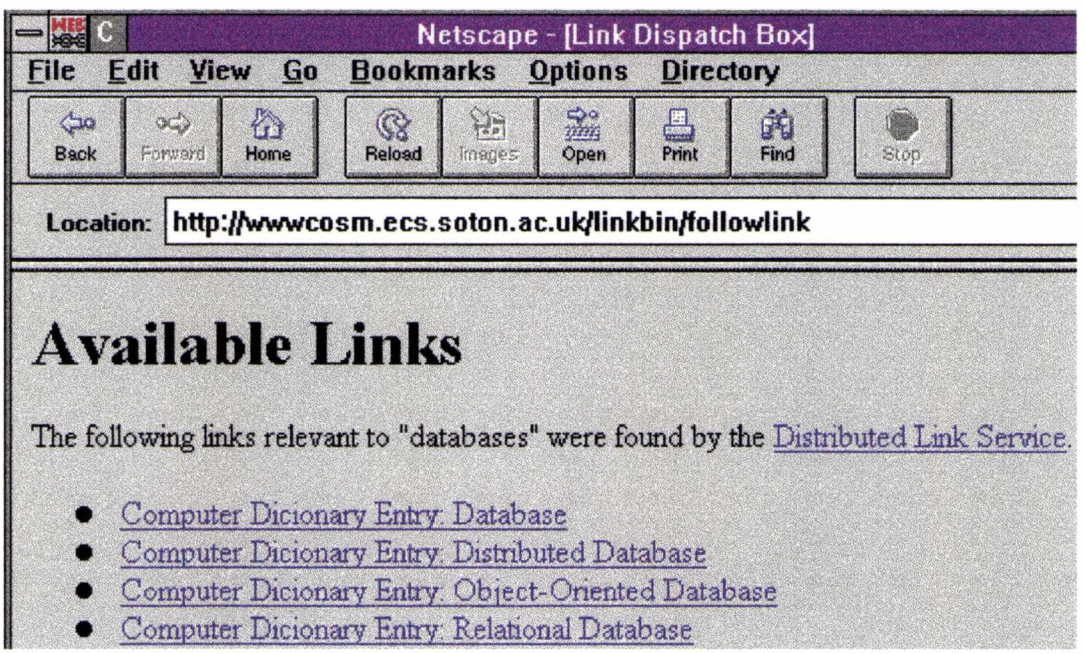
2.7.3. Interface cliente:

Una interface cliente está disponible para PC, Macintosh y plataformas Unix y cuya tarea es realizar pedidos para enlazar funciones y emitir esos pedidos al server de enlace DLS para procesarlos. La interface es una ayuda para cualquier aplicación existente. Algunas aplicaciones pueden ser programadas para comunicarse directamente con el DLS (por ejemplo el editor Emacs).

La tarea principal de la utilidad cliente es responder a un pedido de un usuario para llevar a cabo una función de enlace particular (por ejemplo START LINK, FOLLOW LINK) extrayendo los detalles de la selección que el usuario ha hecho y si es posible los detalles del documento que contiene la selección. Esta información es codificada en un pedido HTTP, reflejando la función elegida y los detalles de la selección.

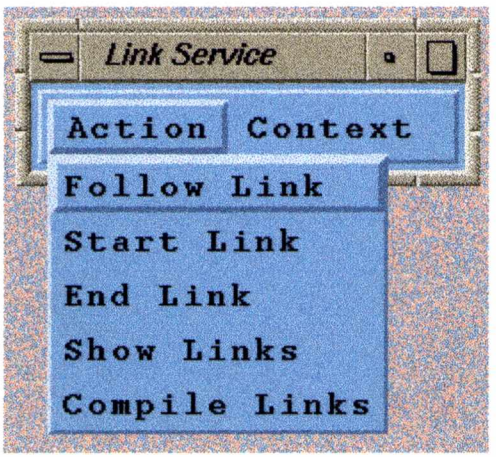
Este pedido debe ser entregado al server de enlace especificado actualmente. Esto es realizado por comunicación con el browser WWW del usuario y pidiéndole cargar la nueva URL. Los resultados del pedido de enlace son retornados como HTML y mostrados por el browser WWW. Por ejemplo al hacer un pedido para seguir un enlace desde una selección particular, la respuesta podría indicar que no fueron encontrados enlaces o listar los enlaces que fueron identificados.

La versión MS-WINDOWS del cliente usa software desarrollado en el proyecto Microcosmo, para adicionar menúes a la barra de título de la aplicación. Un menú provee acceso para enlazar funciones y otro le permite al usuario elegir desde los contextos disponibles actualmente para actividades de enlaces. Las listas de contextos son almacenadas en un archivo de configuración que puede ser actualizado vía el *script context* en el server DLS.



El cliente Unix es una pequeña aplicación Motif que ejecuta las mismas funciones que la versión MS-WINDOWS : un menú para seleccionar la acción de enlace a ser realizada y un menú para seleccionar un contexto para la acción.

La versión X-WINDOWS sin embargo es capaz de acceder al server de enlace DLS directamente, para obtener la lista de contextos disponibles y por lo tanto es capaz de asegurar que el menú contexto está siempre actualizado.



Como con las otras versiones, el cliente MACINTOSH ofrece un menú de acción de enlace y un menú contexto. Estos son provistos como submenúes del sistema "Apple Menú" e implementado como "Applescripts" que a su vez se comunica con Netscape usando "AppleEvents".

Un servicio de enlace de hipermedia provee una importante funcionalidad a los sistemas de hipermedia. Junto con la Web provee una herramienta poderosa con la cual tratar mucha de las restricciones experimentadas con los servicios tradicionales de la Web, incluyendo facilidad de mantenimiento de información y capacidades de creación mejoradas. Un servicio de enlace simple puede ser implementado usando browsers y servers standard de la Web.

2.7.4. Acceso a Documentos Remotos:

El acceso a documentos remotos puede ser provisto extendiendo el DCS de Microcosmo. La principal extensión requerida es la habilidad para distinguir identificadores de documentos de sistemas remotos de los disponibles en el sistema local.

El identificador de documento único es construido usando el día y hora en que un documento es importado a la base de datos. Además de identificar la localización de red de un documento, es necesario identificar el sistema host en el que está almacenado y el identificador de documento local.

El formato elegido fue el LOCALIZADOR DE RECURSOS UNIVERSAL (URL) usado por el sistema World Wide Web. Esta es una estructura de propósito general que permite que sean especificados varios protocolos de acceso. El identificador de documento en el sistema Microcosmo distribuido está formado de la siguiente manera:

< access method >:\\ < network > [: < port >] \ < unique id >

Para los documentos manejados por Microcosmo, el método de acceso es establecido como "mcm".

Los enlaces pueden hacerse a documentos almacenados en otros sistemas, por ejemplo en la WWW (modo de acceso "http") o FTP (modo de acceso "ftp"), El identificador único local para un documento será el identificador DCS, pero éste también puede ser un nombre de archivo único. Esta es una diferencia clave de la definición standard de la URL de la WWW, aunque esta parte del URL puede ser construida de diversas formas, dependiendo del server, esto es usado para especificar el nombre del archivo. La ventaja del uso de un identificador de documento es que el archivo actual puede ser movido o renombrado en el sistema local, y puede ser accedido desde sistemas remotos, ya que el DCS del host grabará cualquier cambio de ubicación.

Para usar este identificador de documentos extendido, la función que crea un nuevo identificador debe ser actualizada, y todas las funciones que usan tal identificador deben extraer el método de acceso y detalles de host para verificar si el documento que está siendo referenciado está presente en el sistema local. Sino debe usarse un protocolo standard en los distintos sistemas para transferir el archivo apropiado, o su registro de la base de datos.

Para acomodar el uso de una variedad de protocolos de acceso dentro del sistema distribuido, el DCS ha sido extendido a una forma modular. Se provee de módulos separados para implementar un tipo particular de acceso, éstos son cargados dinámicamente de acuerdo a los requerimientos. Esto reduce el esfuerzo para incorporar nuevos protocolos de acceso en el sistema.

2.7.5. Sistema Manejador de Filtros:

El uso de una cadena de filtros lineal requiere que todos los mensajes sean ruteados a través de todos los filtros activos, aunque no todos los filtros conozcan todos los tipos de mensajes. Esto causa una reducción significante en la performance de un sistema distribuido.

Para mejorar esto se creó una versión más avanzada del FMS. Los filtros activos registran el tipo de mensajes que ellos aceptarán y así permiten al FMS construir una tabla de acciones disponibles y los filtros que las procesarán. Por lo tanto el FMS mantendrá distintas cadenas pequeñas (una para cada acción) en lugar de una cadena compuesta. La tabla de ruteo de mensajes resultantes puede ser usada por el FMS para enviar mensajes de un tipo particular en forma inteligente, enviándolos sólo a aquellos filtros que son capaces de procesarlos. Un filtro que provee una base de datos de enlaces responderá a mensajes tales como CREATE LINK o FOLLOW LINK, mientras que un filtro que mantiene una historia de las acciones del usuario responderá a mensajes de OPENED DOCUMENT.

Al recibir un mensaje el filtro puede elegir procesarlo o ignorarlo. En el último caso simplemente retorna el mensaje que será ruteado a otro filtro y no toma ninguna acción. Si el mensaje es de un tipo que el filtro puede procesar hará una de las siguientes cosas:

1. Puede realizar alguna acción basada en el contenido del mensaje. Por ejemplo al recibir un mensaje FOLLOW LINK, un filtro de base de datos de enlace buscará algún enlace comparando los detalles en el mensaje y retorna un mensaje que describe cualquier enlace encontrado.

2. Puede modificar el contenido del mensaje.

3. Puede registrar el mensaje anotando o monitoreando propósitos. Por ejemplo el filtro HISTORY registra detalles de documentos que un usuario ha visto, manteniendo copias de mensajes que describen documentos que han sido abiertos.

4. Puede bloquear el mensaje e impedir que sea enviado a otros filtros, por ejemplo al recibir un mensaje de CREATE LINK un filtro de base de datos de enlaces agregará el enlace a su base de datos pero no lo pasará para impedir que sean creados enlaces duplicados en otras bases de datos de enlaces.

Para soportar el modelo distribuido el FMS ha sido extendido para permitir la comunicación con otros hosts de red.

El primer paso es ubicar los sistemas Microcosmo remotos, y establecer donde esos sistemas tienen filtros que pueden ser incorporados a la configuración local. Esto requiere las siguientes preguntas para el FMS distribuido:

- Un host particular ofrece un server Microcosmo?

- Un server particular tiene algún filtro público?

- Cuáles son los detalles de un filtro particular, de modo que pueda ser creada una conexión virtual? Por ejemplo qué tipo de mensajes aceptará y qué port de red será usado.

Estas preguntas son entregadas a un port de red y así se garantiza que si un sistema particular ofrece servicios de Microcosmo, entonces otro sistema Microcosmo puede localizarlo.

Una vez que los filtros remotos han sido integrados en una configuración de un sistema particular, los mensajes generados en el sistema necesitarán ser transferidos a esos sistemas remotos. Estos mensajes son entregados a un port especificado cuando el filtro remoto es "conectado".

Para permitir a cada FMS establecer si está tratando con un mensaje originado en un sistema local o remoto, los campos adicionales que identifican al mensaje origen son agregados a éste antes de ser transferido. Esto permite al FMS remoto rutear el mensaje sólo al filtro apropiado.

Si un filtro genera mensajes adicionales, todos los campos deben ser copiados en ellos. Esto permite que los mensajes sean retornados al sistema origen una vez procesados.

2.7.6. Sistema de Control de Documentos Distribuido

Se han realizado construcciones similares para el DCS de Microcosmo que permiten la recuperación de documentos desde sistemas remotos. Como los documentos pueden ser bastante largos, y demanda mucho tiempo recuperarlos, cada sistema mantiene una copia de documentos para incrementar la performance.

Para los documentos manejados por Microcosmo, el método de acceso es establecido como "mcm". El DCS trata con docuverse (universo de los documentos importados al DCS).

Las funciones en el docuverse extraen la información acerca del método de acceso y si es necesario usan un protocolo standard para acceder a la información de sistemas docuverse remotos.

Para adaptar el uso de una variedad de protocolos de acceso con el sistema distribuido, el DCS ha sido extendido en forma modular. Los módulos separados pueden ser provistos para implementar un tipo de acceso particular y cargados dinámicamente como sean requeridos.

La ventaja del DCS es que provee una lista de todos los documentos en los que se está interesado. Esto hace más fácil crear herramientas que puedan iterar sobre los documentos de esa lista: dichas herramientas luego pueden chequear regularmente que cada server y documento conocido por el DCS estén aún accesibles y puede chequear usando el comando Head del protocolo HTTP que el documento no fue modificado. El DCS puede chequear si hubo problemas al acceder a los documentos; por ejemplo coloreando el ícono del documento.

Otra posibilidad usando el DCS es almacenar una copia local de cualquier documento que es referenciado. Si el documento Web se vuelve inaccesible se podría ofrecer al usuario la posibilidad de ver la copia almacenada localmente. Esta versión puede no estar actualizada como la copia remota, pero podría dar al usuario la información que fue requerida.

Alternativamente, si el documento referenciado es actualizado, entonces el DCS es capaz de mostrar un resumen de los cambios, si el usuario está interesado.

En un sistema multiusuario es necesario el control de versión como así también definir los principios de creación.

2.8. COMPARANDO MICROCOSMO CON LA WWW

La WWW es el sistema de hipermedia más usado en el mundo. Fue diseñada para usarla en un ambiente distribuido con acceso a documentos remotos provistos por un esquema de direccionamiento simple (URL).

Las principales diferencias entre Microcosmo y la WWW son:

- La creación de documentos en la WWW implica el uso de HYPERTEXT MARK-UP LANGUAGE (HTML), hay distintos editores HTML (por ejemplo Hotmetal y WebMagic) y programas de conversión (por ejemplo WebMaker y LaTeX2HTML) para ayudar al mecanismo de creación. En Microcosmo los documentos son mostrados en sus formatos originales y solamente son agregados enlaces.

- A diferencia de Microcosmo, la WWW es un sistema "cerrado" ya que solamente los documentos que pueden contener enlaces son los HTML, si el destino del enlace no es un documento de este tipo, entonces sería un enlace perdido. Los enlaces en la WWW están incorporados en los documentos y no en las linkbases.

- Actualizar un hipertexto de Microcosmo agregando nuevos nodos implica que:
  1) Si los nodos son recursos nuevos generales (materiales primarios) entonces un grupo de nuevos enlaces genéricos deben ser agregados y serán aplicados a los componentes de hipertexto existentes.
  2) Si son materiales secundarios nuevos (ensayos o comentarios) estarán afectados por los enlaces existentes.

  Por lo tanto el modelo de hipertexto de Microcosmo es poderosamente escalable.

  En muchos ambientes de hipermedia incluyendo la Web, cambiar los enlaces significa reescribir el texto, ya que los enlaces están incorporados en el mismo, lo cual dificulta su escalabilidad y mantenimiento.

- Si un documento en la WWW es movido o borrado entonces todos los documentos que contienen un enlace a él deben ser actualizados. Esto lleva a encontrar enlaces perdidos, este problema está siendo solucionado con programas como el WebLinker. En Microcosmo sólo se necesita actualizar la Linkbase.

- Los enlaces realizados en la WWW son únicamente punto a punto, en cambio Microcosmo ofrece la posibilidad de crear distintos tipos de enlaces: locales, genéricos, específicos, computados.
  Un enlace genérico, el tipo de enlace más común en Microcosmo, permite al autor asociar un documento con cualquier ocurrencia de un string de texto particular en cualquier documento. Esto puede parecer una operación de recuperación de texto, pero desde el punto de vista práctico, un enlace genérico no requiere la

indexación del posible documento destino ni una operación de búsqueda en todos los documentos del hipertexto.
La diferencia entre enlaces genéricos y recuperación de texto es que en un enlace el autor conoce la relación entre el significado de dos entidades en el hipertexto mientras que una recuperación de texto expresa una similitud estadística en las características textuales de dos entidades de hipertexto. Un enlace genérico define una colección de orígenes aplicables mientras que una operación de recuperación de texto describe una colección de destinos aplicables.

- Un documento HTML contiene un conjunto de enlaces fijo, el cual será visto por todos sus lectores. Para proveer distintos conjuntos de enlaces desde el mismo documento para distintas aplicaciones se deben mantener múltiples versiones del documento. Esto se resuelve con Microcosmo teniendo múltiples linkbases que se refieren al mismo conjunto de documentos pero ofrecen diferentes perspectivas de la información que contienen.

- Ambos sistemas tienen una forma de llevar la historia de los documentos visitados, las browsers WWW permiten back-tracking a través de las funciones siguiente y previo, pero esto no es suficiente para salvar la historia, sólo agrega direcciones específicas a un "hotlist". Microcosmo provee un filtro "history" que permite al usuario ver una lista de todos los documentos visitados e ir a cualquiera de ellos.

- La WWW se caracteriza por:
  1) Un formato de datos natural único y bien definido para usar con una viewer de documento.
  2) Un esquema de direccionamiento universal con un protocolo de transferencia asociado.
  3) Un esquema de creación de hipertexto que establece que las direcciones destino de los enlaces sean especificadas como parte de los documentos origen.

  Microcosmo está caracterizado por:
  1) Una estructura cooperativa para diversas viewers de documentos.
  2) Una estrategia de creación de hipertexto basada en una relación genérica entre documento origen y destino.

2.8.1. Integración de Microcosmo con la WWW

A los fines de lograr la integración de Microcosmo con la WWW se definieron las siguientes propuestas:

1. Una forma de incorporar material WWW en Microcosmo es usar clientes WWW como viewers de Microcosmo, y tratar al material WWW como información a ser enlazada. Usando un cliente WWW como "conocido de Microcosmo" es posible agregar un enlace desde algún documento WWW a otro material. Normalmente no se podían hacer modificaciones a estos documentos WWW, ya que todos los enlaces estaban incorporados en el documento origen.

2. Usar la WWW como una forma de entrega de material desarrollado en Microcosmo. Esta propuesta tiene varias ventajas:

   - Microcosmo ofrece una forma mucho más eficiente para desarrollar material de hipertexto que editar documentos HTML. Las facilidades tales como enlaces genéricos y enlaces computados pueden incrementar la velocidad de creación de enlaces.

   - Como el material creado en Microcosmo almacena los enlaces en linkbases separadas, el material puede ser reconfigurado fácilmente con estructura de enlaces alternativas y estas estructuras pueden ser editadas fácilmente.

   - Usar la WWW provee un método de entrega de material de Microcosmo de bajo costo para los usuarios que no tienen acceso al ambiente Microcosmo.

   Se ha desarrollado una utilidad que puede ser usada para "compilar" material de Microcosmo en HTML, para entregarlo vía un server WWW. Esta herramienta acepta una lista de documentos y una base de datos de enlace, y para cada documento crea un HTML equivalente con todos los enlaces relevantes del texto, obtenidos desde la base de datos de enlaces. Los enlaces se pueden hacer a cualquier documento, aunque los clientes WWW pueden necesitar aplicaciones auxiliares adicionales para tratar con formatos no standard. Los enlaces también se pueden hacer a cualquier URL válida de la WWW, permitiendo hacer enlaces a recursos de la Web existentes, como así también a material de Microcosmo.
   Si el documento HTML fue generado desde Microcosmo, al realizarse un cambio, los documentos originales podrían ser actualizados al mismo tiempo, y las versiones HTML regeneradas automáticamente.
   Se intentó desarrollar más esta propuesta, para proveer un sistema que pueda crear y entregar documentos HTML dinámicamente desde materiales creados en Microcosmo. Tal sistema reduciría el mantenimiento ya que los cambios hechos al material Microcosmo inmediatamente serán reflejados en los documentos HTML que son generados. También permitiría a los usuarios WWW tener acceso al material que cambia dinámicamente.

3. Otra propuesta ha sido proveer servicios de enlaces flexibles de Microcosmos a los usuarios WWW que no tienen un ambiente Microcosmo local. Esto se logra imitando la arquitectura de Microcosmo y la base de datos de enlaces externa (*) en la Web. Como se describió antes, la arquitectura de Microcosmo consiste de pedidos individuales ordenados a través de una cadena de procesos filtros: sobre la Web estos pedidos son implementados como mensajes HTTP, recibidos por el script CGI y luego ruteados a través de un conjunto de procesos sobre el server Unix. Cada uno de estos procesos puede intentar satisfacer un pedido de enlace accediendo a una base de datos de enlace particular, o comparando algún dato en un recurso externo como un diccionario o un conjunto de páginas manuales. Algún resultado obtenido retornará al usuario como respuesta al pedido HTTP.
   La interface al usuario de esta tercer propuesta está en la forma de un auxiliar al browser WWW standard, un ícono que permite al usuario acceder al menú de opciones de enlace (FOLLOW/CREATE/SHOW LINKS). Este ícono puede ser localizado en la barra de títulos del browser (como en la versión MS-Windows) o puede ser una parte del escritorio. El usuario puede seleccionar una pieza de texto en la ventana cliente WWW (o cualquier otra) y seguir el "FOLLOW LINK" desde el menú del auxiliar. El auxiliar hace que el browser envíe un pedido HTTP al server (usando el CGI standard si está disponible) y luego de una pequeña demora el cliente recibe un documento HTML con una lista de los destinos posibles que fueron determinados por el server. Este documento (titulado "enlaces disponibles") contiene un conjunto de botones HTML standard, que enlaza a los destinos dados en la base de datos de enlace y permite al usuario elegir del conjunto de destinos posibles.

Con estas propuestas se demuestra que el Modelo Microcosmo se extiende naturalmente a un ambiente de información totalmente distribuido, donde el manejo de información de multimedia es un tema muy importante.

(*) Las linkbases pueden ser clasificadas como internas, que son aquellas que se refieren a una estructura particular sobre el recurso y definen una navegación del usuario a través del recurso; o como externas que son aquellas que llevan al lector a materiales relacionados en otros recursos, o traen lectores desde otros recursos dentro de este mismo.
Las colecciones de estos recursos son consideradas aplicaciones de hipermedia.

2.9. MICROCOSMO TNG

Es un sistema de hipermedia distribuido abierto. Es un modelo reflexivo para facilitar la construcción dinámica de jerarquías de aplicaciones de hipermedia distribuida.

En los sistemas de hipermedia distribuido se han distinguido tres requerimientos claves:

* Compartir información distribuida
* Organización de información distribuida: es la capacidad de presentar colecciones lógicas de documentos distribuidos físicamente en forma unificada.
* Soporte para servicios configurables distribuidos: esto permite a los usuarios acostumbrarse a ambientes de hipermedia para incorporar servicios distribuidos.

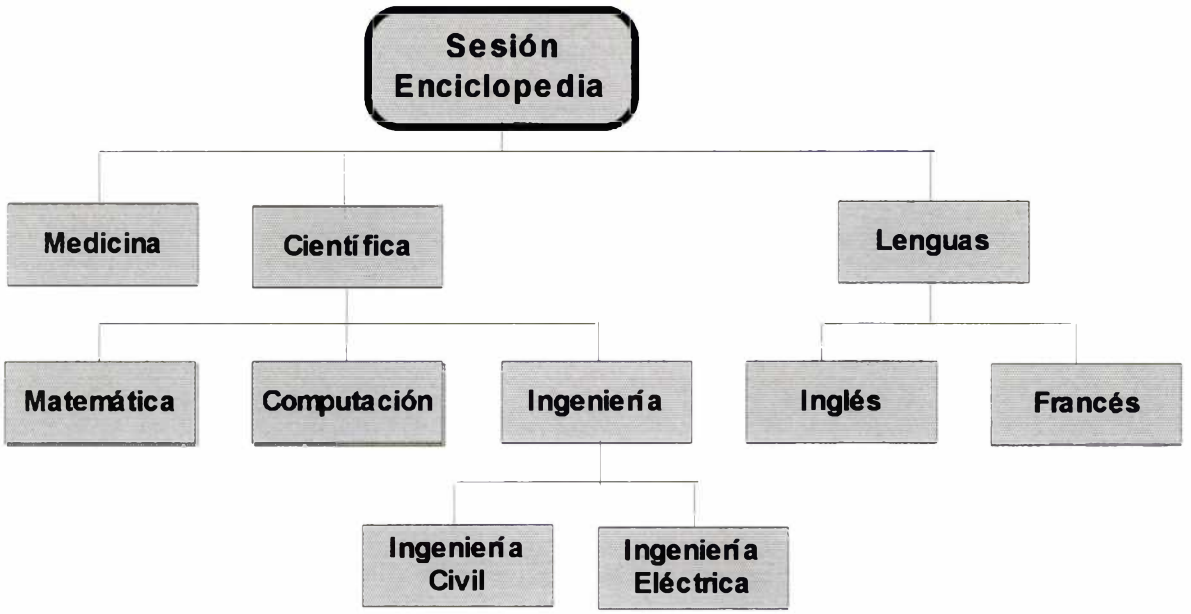
Se define como aplicación de hipermedia a una encapsulación de un grupo lógico de documentos, los servicios que operan en la colección junto con algunos detalles de configuración asociados. Esta definición permite que cada aplicación sea una entidad compartible, publicable e independiente.

Se incluye el concepto de usuarios y sesiones. Un usuario puede tener múltiples sesiones, cada una conteniendo múltiples aplicaciones que pueden, a la vez, ser compartidas entre múltiples usuarios. Esto les permite incluir y compartir aplicaciones públicas con sus respectivas sesiones para construir mejores recursos de información.

Se ideó un mecanismo que permite a los usuarios crear colecciones de aplicaciones lógicas. Esto fue reflejado en este modelo abstracto, extendiendo la definición de una aplicación para incluir una o más aplicaciones opcionales. Esta definición recursiva provee una propiedad reflexiva. De aquí el nombre de **Modelo Reflexivo**.

Con este modelo los usuarios pueden hacer referencia a otras aplicaciones y asociarlas dinámicamente con sus propias estructuras para construir recursos de información que pueden constar de otras aplicaciones distribuidas.

Un ejemplo de este modelo está dado en la siguiente figura:

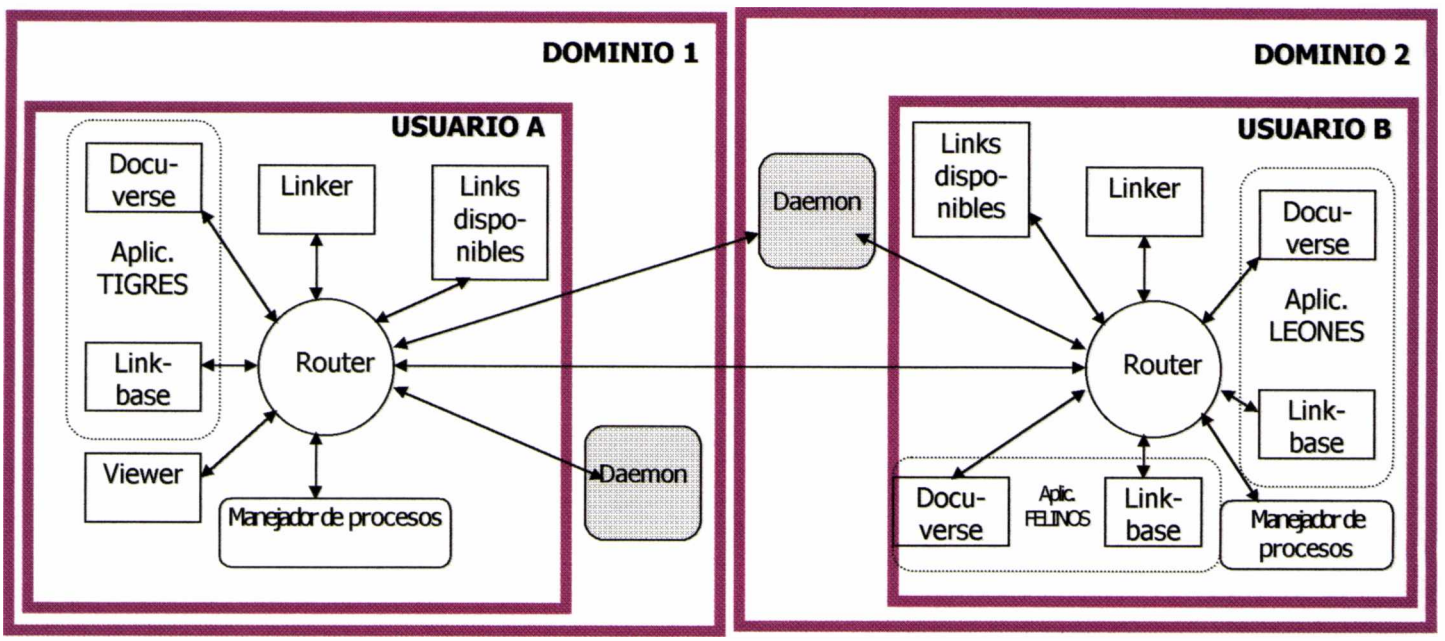


ENCICLOPEDIA es una sesión que está compuesta de muchas aplicaciones que describen su funcionamiento, pero en términos de contenido de información y funcionalidad. Dicha sesión es una entidad extensible dinámicamente con la cual se pueden agregar o sacar nuevas aplicaciones en cualquier momento. Como la sesión no está definida estáticamente las vistas alternativas pueden ser generadas incluyendo o excluyendo varias subjerarquías de aplicación. Por ejemplo, una ENCICLOPEDIA JUNIOR puede usar alguna de las aplicaciones de ENCICLOPEDIA pero no incorporar otras consideradas sin interés por los jóvenes. El modelo reflexivo provee una aproximación flexible y extensible que permite a un usuario crear múltiples y diferentes vistas en el mismo conjunto de aplicación, dependiendo del contexto de la estructura en la que están situados. Además estas aplicaciones pueden ser compartidas entre varios usuarios provocando dispersión de información y aumentando el nivel en el cual se pueden reusar los recursos de hipermedia.

Se construyó un sistema prototipo llamado MICROCOSMO TNG, que conserva la filosofía de Microcosmo.

El prototipo tiene tres capas distintas: **la capa de presentación** : es responsable de producir la información de multimedia en la pantalla**. La capa de servicio de enlace** : consta de un número de procesos conocidos como filtros, que responden a algún pedido de usuario. **La capa de comunicación** : ofrece facilidades para pasar mensajes y manejar procesos.

En la siguiente figura se puede observar como Microcosmo TNG es modelado como procesos discretos en la estructura.



El usuario A tiene varios procesos ejecutando en su sesión como lo hace el usuario B en el dominio 2. Las líneas punteadas denotan una colección de procesos que han sido publicados como aplicaciones. El usuario A ha publicado una aplicación llamada TIGRES y el usuario B puede usar esta aplicación junto con su aplicación LEONES para enriquecer su aplicación general FELINOS.

El modelo reflexivo permite al usuario componer colecciones de procesos en jerarquías relacionadas coherente y lógicamente. Estas a su vez pueden ser publicadas por otros para su uso. La propiedad reflexiva de las aplicaciones es modelada en la estructura a través del pasaje de mensaje. Cuando un usuario emite un pedido de servicio a una aplicación particular, el pedido es propagado en forma transparente hacia abajo a través de la jerarquía de aplicaciones.

Por lo tanto, Microcosmo TNG es un sistema multiusuario, heterogéneo, ampliamente distribuido que provee una estructura en la cual residen las aplicaciones reflexivas.

2.10. CONTROL DE VERSION

El control de versión es esencial en un ambiente de trabajo cooperativo ya que diferentes autores comparten la misma información y pueden actualizarla en distintos momentos, requiriendo que más tarde sea integrada.

El control de versión en un sistema de hipermedia va más allá del control normal de versión para documentos comunes, éste agrega el concepto de control de versión de enlaces entre documentos. Un control de versión de hipermedia mantiene la historia de los cambios para una red de hipermedia segura y además permite al usuario experimentar con diferentes configuraciones de enlaces.

Microcosmo es un sistema de hipermedia abierto desarrollado para PC. Actualmente hay una versión Unix que tiene herramientas para soportar múltiples usuarios.

En la implementación de control de versión de Microcosmo, los enlaces y nodos son tratados como objetos que pueden existir en más de una versión.

Exodus, la base de datos orientada a objetos elegida para la linkbase en la versión cooperativa de Microcosmo, tiene algunos mecanismos primitivos para controlar versiones de objetos almacenados en ella. Básicamente para crear una nueva versión de un objeto es necesario congelarlo y crear una nueva versión de ese objeto congelado. Un objeto congelado no puede ser modificado en un futuro y un nuevo identificador del objeto (OID) es mostrado por la nueva versión del objeto congelado. Como Microcosmo enlaza y almacena dentro de una base de datos, este mecanismo primitivo para controlar versiones ha sido mejorado y usado para control de versión de enlaces. Las versiones de nodos son creadas usando RCS (Sistema de control de revisión).

Una aplicación en Microcosmo es una colección de nodos y linkbases. Esto permite al usuario utilizar versiones para mantener distintos estados de toda la aplicación. La interface del usuario permite la creación de versiones para un conjunto de aplicaciones en un modo temporal y los usuarios son libres para moverse de una versión a otra si así lo desean. Por lo tanto se ofrece una forma fácil para que el usuario cree versiones y al mismo tiempo permitirle navegar a través de las versiones ya que no hay restricciones impuestas durante el recorrido.

Todos los objetos individuales de una aplicación tienen versiones individuales, pero una aplicación es un conjunto bien definido de objetos versionados.

PIE es un sistema desarrollado para asistir al control de versión en aplicaciones de ingeniería de software. Este implementa distintas capas para cada versión diferente. Básicamente tiene un nivel inicial y se crean nuevas capas para acomodar conjuntos de cambios para una o más entidades en la red. Así una capa puede ser referenciada como el conjunto de cambios.

Una combinación fija de capas puede ser almacenada como un contexto. El concepto de contexto en PIE no existe en esta implementación, se limitó

al modelo de red en capas para representar sólo relaciones temporales, así cada nueva capa creada constituye una nueva versión de una aplicación entera y una evolución de la capa anterior. Los usuarios pueden cambiar de una capa a otra durante la consulta. Esta facilidad de consulta es más flexible que los contextos en PIE ya que no se imponen estructuras rígidas y no hay un mecanismo hasta el momento para salvar una combinación de objetos pertenecientes a distintas versiones en un contexto.

Las funcionalidades que el usuario puede alcanzar con el nuevo sistema son las siguientes:

- El usuario puede crear una nueva versión cuando lo crea conveniente.

- El usuario puede listar todas las versiones disponibles y leer una descripción resumida de ellas.

- El usuario puede cambiar de una versión a otra en un simple paso.

- El concepto de versión puede ser resumido aún más para permitirle al usuario elegir qué versión de un enlace quiere seguir.
  El usuario puede pasar de una capa a otra sin darse cuenta que está navegando a través de versiones completamente diferentes. Por ejemplo si una aplicación de hipermedia muy grande fue creada con distintas versiones de la red creada para acomodar diferentes niveles de detalle y complejidad, los lectores podrían saltar de una capa (versión) a otra de acuerdo a la cantidad de detalles que ellos desean ver como resultado de seguir un enlace de hipertexto. Desde el punto de vista de trabajo cooperativo, suponiendo que co-autores hacen modificaciones en diferentes versiones, es posible seguir enlaces a diferentes versiones del mismo nodo y comparar distintos puntos de vista de distintos autores. Usando la información disponible en Microcosmo, los autores pueden acordar sobre una versión final.

- Como varias linkbases pueden estar activas al mismo tiempo, más de una versión de una aplicación dada puede estar activa al mismo tiempo. Los co-autores pueden usar esta funcionalidad junto con mecanismos conocidos para mezclar versiones.

Por lo tanto, el control de versión provee una interface fácil de usar para que el usuario cree y visualice diferentes versiones de una aplicación de hipermedia. No sólo se creó un mecanismo de control de versión eficiente, sino también una interface de uso fácil al usuario. También se desarrollaron herramientas eficientes para permitir a los usuarios mezclar sus trabajos, ya que esta actividad normalmente necesita de mucha intervención de los usuarios.

2.11. EXPERIENCIAS DE INTEGRACION DE APLICACIONES CON MICROCOSMO Y CHIMERA

Los sistemas de hipermedia abiertos tienen la capacidad de integrar muchas aplicaciones externas, en la actualidad hay una gran cantidad de experiencias dentro del campo de la hipermedia, que colaboran en realizar trabajos de integración fácilmente.

Existen métodos para crear modelos arquitecturales de integración de aplicaciones con sistemas de hipermedia abiertos. Estos modelos pueden ser usados para:

1. Modelar aplicaciones antes de ser integradas.

2. Describir las características de una integración completa.

3. Categorizar integraciones existentes.

4. Proveer una estimación de grado de esfuerzo requerido para realizar una integración.

5. Proveer una guía en la selección de la arquitectura de una integración terminada.

EJEMPLOS

A continuación se describe la integración del editor de texto XEMACS con el Sistema Chimera y una integración del Calendario de Microsoft con Microcosmo usando la viewer universal.

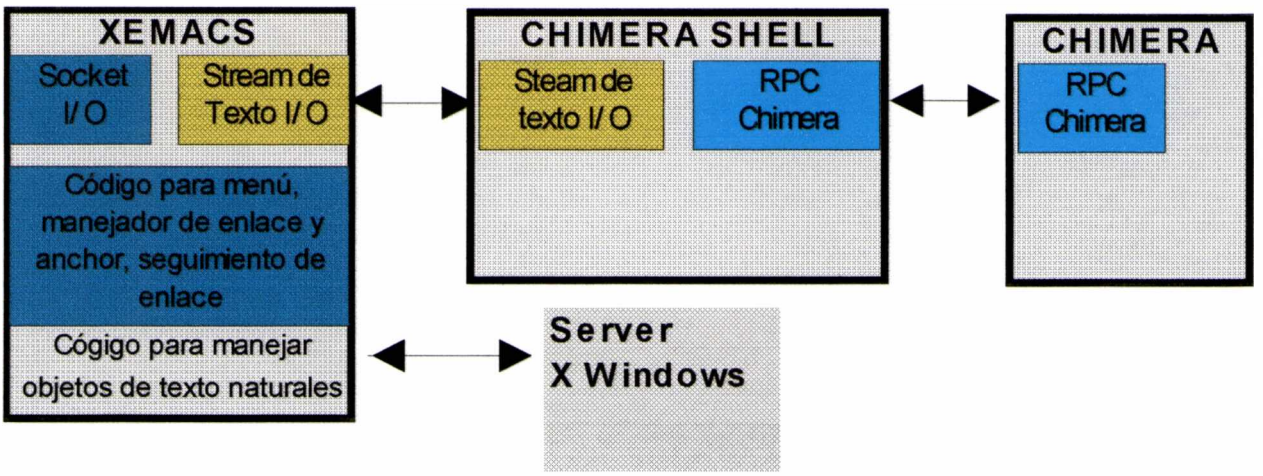
2.11.1. Integración XEMACS/Chimera:

La integración XEMACS/CHIMERA permite agregar anchors y enlaces a un documento de texto sin modificarlo. Usando el menú Chimera se puede: crear y borrar anchors sobre una porción de texto, crear nuevos enlaces, hacer a un enlace "activo" desde el cual se pueden agregar o remover anchors y hacer un seguimiento de enlace. En Chimera cada usuario tiene un enlace activo, por lo cual los anchors pueden ser agregados o borrados sin que todas las aplicaciones conozcan su identificador de enlace.

Con un enlace activo, un usuario no tiene que ingresar un identificador de enlace para modificarlo y cuenta con una opción CREATE LINK en su interface haciendo más fácil la creación del enlace.

En la integración XEMACS/CHIMERA, al realizar un seguimiento de enlace se puede seleccionar entre abrir una nueva ventana o usar una existente para mostrar el anchor destino.

La siguiente figura muestra la arquitectura de integración XEMACS/CHIMERA.



XEMACS no soporta llamadas a procedimientos remotos (RPC), formato de comunicación natural de Chimera, la integración usa el stream de texto I/O de XEMACS para comunicarse con el Chimera Shell, que convierte comandos de texto en mensajes RPC de Chimera. El menú de Chimera y secuencias de claves de control son creados por el código EMACS LISP (lenguaje con el que se definen muchas de las funciones del editor de texto). La creación, borrado, muestra y selección de anchors son manejados por las funciones EMACS LISP, mientras que para los enlaces, su creación, iniciación y muestra de un seguimiento de enlace son manejados por otras funciones. Estas funciones interactúan con EMACS a través de librerías EMACS LISP y con Chimera enviando comandos de textos al Chimera Shell.

Cuando se requiere una operación de hipermedia, un comando de texto se envía al Chimera Shell, que lo analiza, lo transforma a RPC e invoca al server Chimera. Este actúa sobre el comando y entonces el Chimera Shell transforma los valores de retorno RPC en texto antes de imprimirlos. El texto retorna valores que son leídos por código Emacs Lisp que los convierte en tipos naturales.

2.11.2. Integración Calendario/Microcosmo:

Como el calendario no tiene una interface de programa de aplicación externo, la integración con Microcosmo se realiza a través de la Viewer Universal. Ésta agrega un ícono Microcosmo a la barra de título de la aplicación.

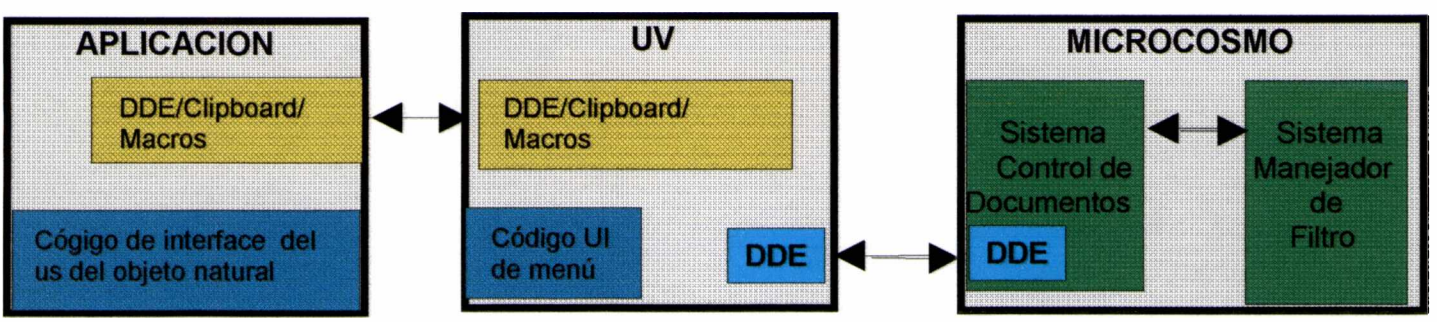
La integración CALENDARIO/MICROCOSMO, semejante a todas las integraciones que utilizan la Viewer Universal, provee la capacidad de crear, editar y seguir enlaces.

Las integraciones con la Viewer Universal emplean las facilidades provistas por las aplicaciones para resaltar regiones de texto indicando anchor.

Cuando una región de texto es resaltada y se ha seleccionado FOLLOW LINK o alguna otra acción desde el menú de la Viewer Universal, la aplicación es introducida vía macros, Microsoft Windows DDE o un Windows Recorder File para tomar la selección y devolverla a la Viewer Universal a través del clipboard, DDE o un archivo de cambio nominado.

En el caso de la integración CALENDARIO/MICROCOSMO se usa una macro para copiar la selección resaltada al clipboard.

En la siguiente figura se muestra la arquitectura de la integración de una aplicación Windows con Microcosmo usando la Viewer Universal.



Mientras la aplicación contiene códigos para manejar sus tipos de objetos naturales y el resaltado de anchors, la Viewer Universal contiene código para manejar enlaces y recorrerlos. Como la aplicación no cuenta con un lenguaje, no puede ser modificada para enviar mensajes DDE naturales de Microcosmo, la Viewer Universal actúa como un mediador entre la aplicación y Microcosmo, usando uno de los distintos caminos de comunicación posibles para recuperar información del anchor desde la aplicación, que combina con la solicitud del usuario y transmite usando el protocolo de comunicación DDE natural de Microcosmo.

2.11.3. Similitudes:

Las integraciones XEMACS/CHIMERA y CALENDARIO /MICROCOSMO tienen algunas similitudes. En ambos casos la aplicación no podía comunicarse usando el protocolo natural de los sistemas de hipermedia, por lo que fue necesario usar un intermediario de comunicaciones.

En el caso de la integración XEMACS/CHIMERA este intermediario fue el Chimera Shell y en la integración CALENDARIO/MICROCOSMO fue la Viewer Universal. XEMACS y CALENDARIO no cuentan con las posibilidades de manipular anchors y enlaces, estas facilidades deben ser provistas por la integración a través de una interface de usuario.

2.11.4. Modelo Arquitectural de las Integraciones:

La arquitectura de una integración de aplicaciones se caracteriza por su soporte de interface de usuario para manipular anchors y enlaces y el tipo de comunicación entre la aplicación y el sistema de hipermedia.

La arquitectura consiste de tres elementos principales: **ARTISTAS**, modelan los aspectos de la interface del usuario, **COMUNICADORES**, modelan las comunicaciones y **CONTAINERS**, donde se agrupan los otros elementos.

ARTISTAS: Es un código responsable de manejar una interface de usuario que representa e intermedia la manipulación de un objeto particular. Para modelar integraciones hay tres tipos de artistas:

1. *Artista natural:* Representa e intermedia operaciones en un tipo de dato natural de la aplicación (por ejemplo, un procesador de texto es considerado por su formato de archivo natural).
2. *Artista anchor:* Representa y media operaciones sobre anchors.
3. *Artista enlace*: Representa y media operaciones sobre enalces.

Esta subdivisión se debe a que la interface de usuario para aplicaciones de hipermedia conocidas contienen recursos para manejar anchors, enlaces y los tipos de objetos naturales de la aplicación. En el caso de las aplicaciones de hipermedia no conocidas, los artistas anchor y enlace están separados del resto de la interface de usuario, ya que estas aplicaciones no tienen la capacidad para manejarlos, lo cual debe ser provisto por la integración.

COMUNICADORES: Es el código responsable de establecer y manejar las comunicaciones entre las aplicaciones usando un protocolo específico, por ejemplo en Chimera, el código que maneja esta comunicación usando el protocolo Chimera RPC podría ser modelado como un comunicador Chimera.

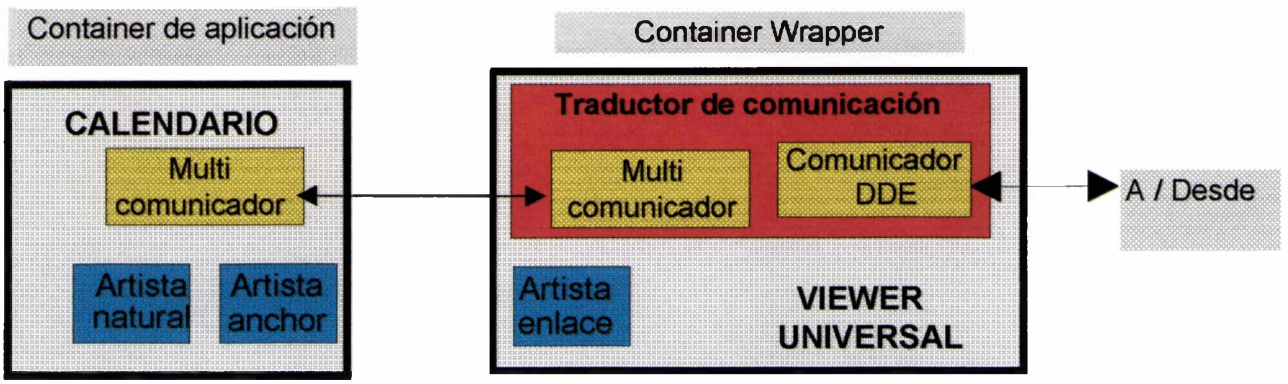
Un ***TRADUCTOR DE COMUNICACIONES*** consiste de dos comunicadores dentro de un *Container traductor* junto con el código que traduce entre los dos protocolos de comunicación.

CONTAINERS: Es usado para agrupar los otros elementos arquitecturales (artistas, comunicadores y traductores). Hay tres tipos usados en el modelo:

1. Container de aplicación: Representa la aplicación integrada con el sistema de hipermedia.
2. Container Wrapper: Representa cualquier proceso wrapper usado en una integración.
3. Container traductor: Contiene los dos comunicadores y el código que traduce entre los dos protocolos de comunicación.

Hay dos reglas que rigen el uso de los containers: Todos los artistas y comunicadores deben pertenecer a un container. Un container puede contener en su totalidad a otro container.

La siguiente figura muestra el modelo arquitectural de la integración del Calendario con Microcosmo usando la Viewer Universal.



En esta integración el artista natural y anchor residen en el container de la aplicación Calendario. El artista natural representa la funcionalidad del CALENDARIO, pero éste no tiene la funcionalidad de hipertexto, el artista anchor permite resaltar el texto en el calendario que la Viewer Universal copia al clipboard antes de leer y transmitir como texto anchor a Microcosmo. Además el artista anchor modela el código en el Calendario que maneja la interface de usuario del anchor.

El artista enlace localizado en el container Wrapper de la Viewer Universal, representa las operaciones provistas por el menú pull down de la barra de títulos y maneja los botones de enlace sobre la barra de título.

El multi-comunicador modela el flujo de información del texto seleccionado desde el calendario en el clipboard y en la Viewer Universal junto con la selección del anchor destino usando una macro de búsqueda. El traductor de comunicaciones en la Viewer Universal convierte entre comunicaciones DDE naturales de Microcosmo y las opciones de comunicación disponibles del Calendario.

2.11.5. Caracterizando una aplicación previo a su integración

Una aplicación puede ser caracterizada previo a ser integrada, en cuanto a la existencia de artistas anchors y enlaces, y basados en su nivel de comunicación. Esto usa el modelo arquitectural en el sentido de describir el estado inicial de la aplicación. Por definición, una aplicación siempre contiene un artista natural para manipular su tipo de objeto natural, y en el peor de los casos este artista puede también servir como el artista anchor para lograr un efecto de hipermedia. Si la aplicación cuenta con una noción de hipermedia, entonces tendrá posibilidades en su interface de usuario de manipular enlaces y probablemente anchors. Por ejemplo, Frame Maker tiene una noción de hipermedia interna y contiene elementos en la interface de usuario para manipular anchors y enlaces. Como resultado, Frame Maker contiene artistas anchor y enlaces previo a la integración.

Otras aplicaciones que no contienen el concepto de hipermedia, necesitan un artista enlace, y también necesitan una facilidad de interface de usuario que permita representar y manipular anchors.

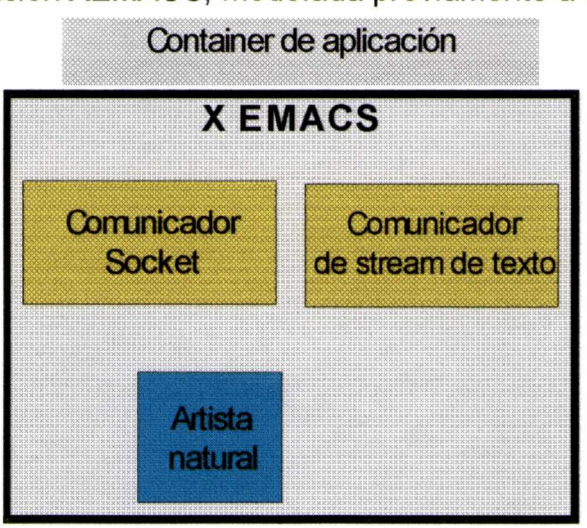
El estado inicial de una comunicación de la aplicación puede ser descripto, desde la perspectiva de un sistema de hipermedia abierto, como "**natural**", "**no natural**" o "**no comunicativa**". Una aplicación que es un comunicador natural, conoce completamente el protocolo de comunicaciones usado por el sistema de hipermedia abierto cuando se comunica con sus clientes.

Una aplicación escrita para usar un protocolo de sistema de hipermedia es considerada un comunicador natural.

Una aplicación no-natural tiene una interface de programa de aplicación externa (API) que puede ser usada para lograr funcionalidad de hipermedia, especialmente una operación de seguimiento de enlace, pero que difiere del protocolo de comunicaciones naturales de un sistema de hipermedia abierto. Frame Maker es un ejemplo de comunicador no natural. Estas aplicaciones son modeladas teniendo un comunicador para el protocolo no natural de la aplicación específica.

Una aplicación no comunicativa es cerrada al mundo exterior excepto vía su propia interface de usuario y no tiene una API externa.

Ejemplo: la aplicación XEMACS, modelada previamente a ser integrada



La aplicación XEMACS se comunica vía una conexión socket de red o vía un stream de texto. Xemacs tiene la habilidad de resaltar regiones de texto subrayándolos, permitiendo esto representar visualmente a un anchor, no tiene la habilidad para crear o borrar directamente anchors desde su interface de usuario, o mantener automáticamente una asociación de un anchor de pantalla con un manejo de anchor del sistema de hipermedia, por lo tanto Xemacs es modelada sin un artista anchor. Xemacs no contiene facilidades de manipulación de enlace, entonces es modelada sin un artista enlace.

2.11.6. Propiedades arquitecturales de una integración completa

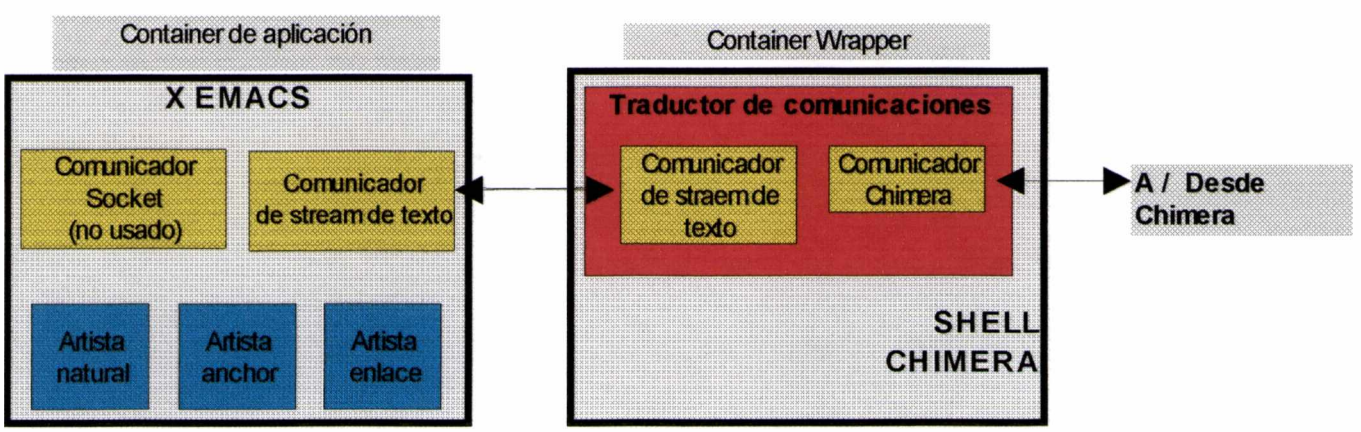
Un modelo arquitectural permite generalizar la experiencia de muchas integraciones de aplicaciones con sistemas de hipermedia abiertos y describe las propiedades de una integración completa:

- Existencia de los tres artistas: natural, anchor y enlace.
- Existencia de un comunicador para el protocolo natural del sistema de hipermedia.
- Si la aplicación tiene un comunicador para un protocolo que es diferente al protocolo del sistema de hipermedia, entonces debe estar presente un traductor de comunicaciones.

Este estado de propiedades para una integración completa consiste de una interface de usuario para manipular anchors y enlaces, y un camino continuo para la comunicación entre la aplicación y el sistema de hipermedia abierto. Al ejecutar una integración de una aplicación externa con un sistema de hipermedia abierto se puede ver como se incorpora la modelización del estado inicial de la aplicación usando el modelo arquitectural, comparando este modelo de estado inicial con las propiedades de una integración completa y entonces proveer los elementos que faltan.

Ejemplo: integración XEmacs/Chimera

Para integrar Xemacs con Chimera se debieron crear artistas anchor y artistas enlaces en Xemacs, y también se creó un traductor de comunicaciones para manejar comuniciaciones entre el Comunicador Socket o el Comunicador de Stream de Texto y el comunicador natural de Chimera. Para la integración final fue usado el comunicador de stream de texto y el Chimera Shell transforma comandos de texto en mensajes RPC de Chimera. La siguiente figura muestra la arquitectura de la integración final XEMACS/CHIMERA.



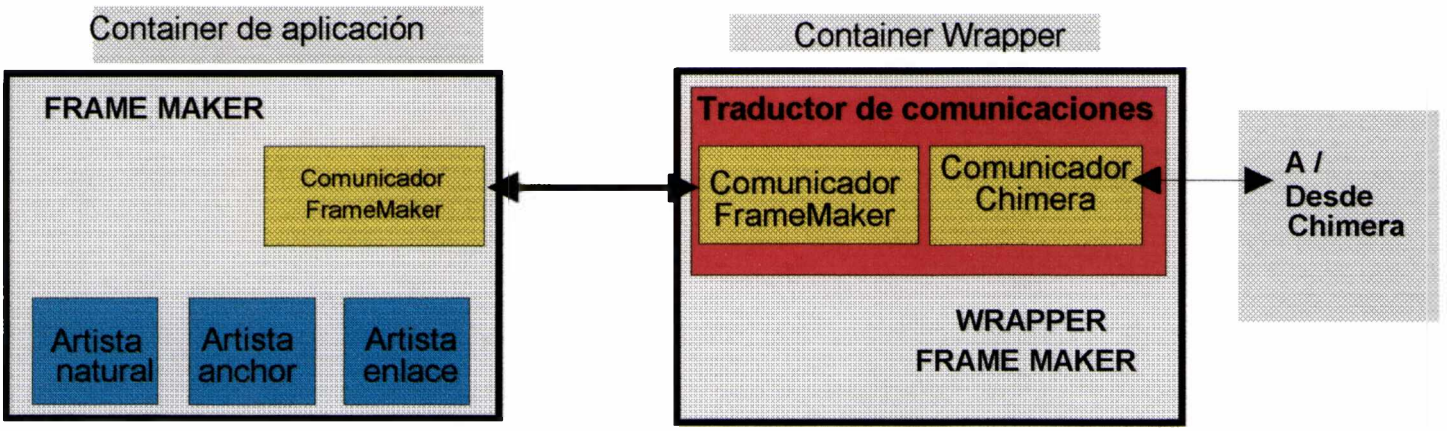
2.11.7. Arquitecturas de integración común

Existen tres categorías de arquitecturas de integración de aplicaciones:

- **Launch only**: en este tipo de arquitectura se construye una aplicación no comunicativa para participar en el sistema de hipermedia abierto. La ventaja principal es la facilidad de integrar rápidamente cualquier aplicación sin modificar su formato natural. Los enlaces sólo terminan en la aplicación y no pueden ser iniciados desde ella. Al recibir un evento de seguimiento de enlace la aplicación es invocada y se le indica que abra un archivo específico. En este caso se define un anchor que es la totalidad del archivo, lo que da como resultado que el artista natural sea idéntico al artista anchor. Las funciones del artista enlace están provistas por opciones de línea de comando o por una herramienta separada. Al ser una aplicación no comunicativa, sus comunicaciones con el sistema de hipermedia deben ser manejadas por algún otro programa.
  Un ejemplo de esta arquitectura es la integración de aplicaciones no comunicativas usando el sistema de hipermdia abierto HyperDisco. Este requiere que la aplicación sea capaz de comenzar en un archivo particular, y ese archivo es el anchor completo. El cliente Xemacs/HyperDisco tiene un menú de opciones especiales que son usadas para crear enlaces a aplicaciones launch only, y actúa como un mediador de comunicaciones para ellos.

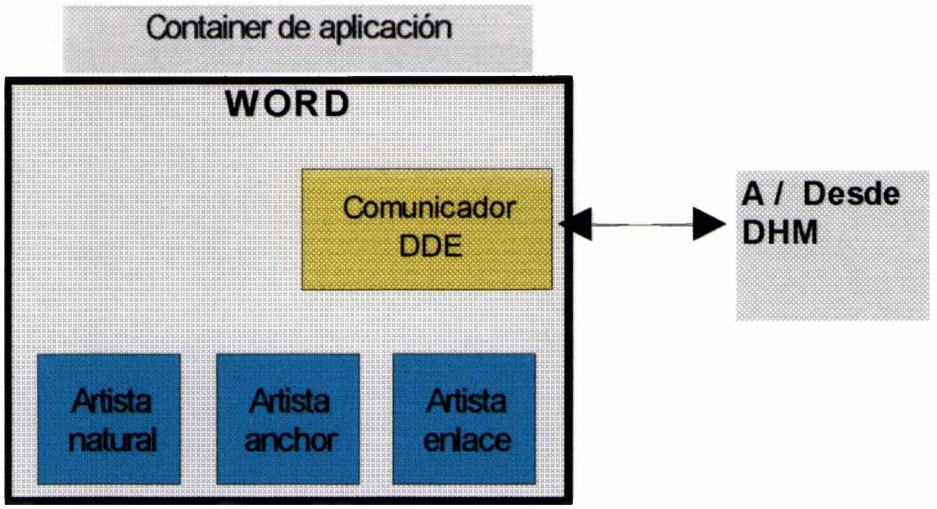
- **Wrapper**: esta arquitectura contiene un elemento computacional separado, frecuentemente un proceso del sistema operativo, que actúa como un intermediario entre la aplicación y el sistema de hipermedia. La principal ventaja de esta arquitectura es que no requiere modificación del formato natural de la aplicación ya que usa la capacidad de hipermedia incluida en la aplicación. Usando la API externa de la aplicación la integración es aislada de los cambios hechos en ella.
  Hay dos casos comunes de arquitecturas wapper:
  1. Un wrapper contiene sólo un traductor de comunicaciones que convierte entre las funciones de hipermedia ofrecidas por la API externa de la aplicación y la funcionalidad natural del sistema de hipermedia.
  2. Un wrapper puede contener un traductor de comunicaciones además de uno o más artistas cuando estos artistas no están presentes en la aplicación. Un ejemplo de esto son las aplicaciones integradas con la UV de Microcosmo.

  Ejemplo: integración Frame Maker/Chimera
  Contiene un traductor de comunicaciones que media entre la funcionalidad de hipermedia natural de Frame Maker, accedido vía su API externa, y la funcionalidad de hipermedia accedida vía la API de Chimera.
  Frame Maker soporta una noción interna de hipermedia y una API que puede manipularla, la aplicación wrapper hace posible integrar Frame Maker sin modificar su formato y sin tener acceso a su código fuente.
  La aproximación wrapper también fue usada para integrar Chimera con Mosaic usando su interface cliente común.



- **Custom**: esta integración consiste en modificar el código fuente de la aplicación o escribir el código en un lenguaje habitual de la aplicación, por ejemplo en Emacs Lisp. Debido al alto grado de control que provee esto sobre la aplicación, una integración custom tiene la ventaja de dar un nivel muy fino de funcionalidad de hipermedia, con anchors que pueden ser más pequeños que un archivo.
  Una integración custom da más control sobre la presentación de un seguimiento de enlace, dando por ejemplo la posibilidad de ir directamente a un anchor o ir a una nueva ventana.
  La desventaja es que la posibilidad de una nueva revisión de la aplicación cortará la integración, requiriendo una actualización del código de integración.
  Cuando la aplicación no cuenta con funcionalidad de hipermedia incorporada en su interface de usuario, la integración implica escribir un artista anchor y un artista enlace. Como la aplicación no tiene un comunicador natural para el sistema de hipermedia, también es necesario crear un nuevo comunicador o adaptar uno de los comunicadores exitentes en la aplicación para convertirlo en un comunicador natural del sistema de hipermedia.

  Ejemplo: integración Word/De Vise Hipermedia
  El lenguaje habitual de Word de Microsoft, Visual Basic para Aplicaciones fue usado para hacer una nueva barra de herramientas que permita crear y manipular nuevos anchors y enlaces, e implementar los artistas anchor y enlace. Las comunicaciones naturales dentro de DHM están basadas en DDE y Word tiene funciones incorporadas en él para enviar mensajes usando DDE, por lo tanto fue fácil adaptar el comunicador DDE como un comunicador DHM. Las integraciones DHM/Excel y Word/Microcosmo tienen la misma arquitectura.



Combinación de arquitecturas:

Algunas integraciones combinan dos categorías de arquitecturas para lograr las ventajas de ambas. Las combinaciones que se han hecho hasta ahora son launch only/wrapper y wrapper/custom. Aún no hay combinaciones launch only/custom, probablemente porque una integración custom implica modificar el código fuente o escribir nuevas capacidades en un lenguaje de costumbre que mejore las ventajas claves de la aproximación launch only.

La integración Xemacs/Chimera es un ejemplo de una combinación de la arquitectura wrapper y custom. El Shell de Chimera, que es un traductor de comunicaciones entre texto y el RPC de Chimera natural, es la porción wrapper de la integración. El código Emacs Lisp que implementa los artistas anchors y los artistas enlaces y que maneja las comunicaciones con el Shell de Chimera constituye la parte custom de la integración.

Esfuerzo de integración

El esfuerzo que se requiere para realizar una integración de aplicaciones está directamente relacionado con su arquitectura. La experiencia ha demostrado que como regla general las integraciones launch only son más fáciles de realizar que las integraciones wrapper, las que a su vez son más fáciles que las integraciones custom. Una integración launch only puede ser realizada en menos de una hora por un usuario experto en sistemas de hipermedia. Una integración wrapper varía ampliamente desde integraciones de la viewer universal que también sólo requiere una hora para integrarla, hasta varias semanas cuando un nuevo wrapper debe ser escrito tal como el Shell de Chimera.

Las integraciones custom puede requerir desde semanas a meses, dependiendo del grado de conocimiento de la aplicación, experiencia con el sistema de hipermedia y el refinamiento de la funcionalidad de hipermedia deseada.

En los casos en que hay varias elecciones, para saber que arquitectura de integración puede ser elegida, teniendo en cuenta las características de las

aplicaciones iniciales, al esfuerzo requerido para hacer la integración se le debe sumar un esfuerzo extra para determinar la arquitectura de integración adecuada. Si una aplicación no comunicativa, sin conocimiento de hipermedia, necesita ser usada inmediatamente es conveniente una arquitectura launch only. Si se desea obtener una capacidad de hipermedia más refinada y están disponibles los recursos del programador y el código fuente, una arquitectura de integración custom dará un mejor resultado.

2.11.8. Diferencias entre Microcosmo y Chimera:

Hay algunas diferencias entre Microcosmo y Chimera:

Microcosmo usa una API basada en mensajes, mientras que Chimera usa una API de programadores multilenguaje. Los mensajes de Microcosmo están en formato ASCII y los filtros actúan sobre las marcas que reconocen en el mensaje e ignoran todas las restantes. Cada filtro puede introducir alguna marca y su dato asociado en cualquier mensaje.

En Chimera los detalles del formato del mensaje están ocultos para el cliente Chimera por la API Chimera y el server Chimera por un mensaje ADT. Esto permite que los desarrolladores del sistema de Chimera cambien libremente el formato de los mensajes sin afectar al resto del sistema. Por lo tanto el formato de los mensajes puede variar desde texto ASCII hasta registros variantes ADA según la semántica de la API Chimera.

Microcosmo no tiene el concepto de view. Cada documento en Microcosmo está asociado con una viewer. Cuando un documento es el destino de un seguimiento de enlace, Microcosmo invoca a la viewer asociada al documento especificado.

El concepto de view de Chimera es independiente de donde está almacenado el dato. Esto permite mayor flexibilidad para manejar múltiples views de un único objeto y también manejar fácilmente el caso en que una view particular consiste de datos accedidos desde múltiples orígenes.

En la versión distribuida de Microcosmo el usuario puede elegir configurar el sistema como stand-alone, sólo-server, sólo-cliente o server-cliente. En el caso de Chimera la configuración es única.

2.12. NUESTRA EXPERIENCIA DE INTEGRACION CON MICROCOSMO PLUS

Trabajamos con material producido en la Dirección de Salud Mental del Ministerio de Salud de la Provincia de Buenos Aires.

En dicha dependencia, en el año 1996, surge el Programa de Salud del Veterano Bonaerense, destinado a incorporar al sistema de salud a los Veteranos de Guerra, detectar sus afecciones y luego ofrecerles su tratamiento.

La Dirección de Salud Mental, se plantea el objetivo de detectar a aquellos que padezcan el PTSD (Síndrome de Estrés Postraumático).

Se convocó a una Consultora de Especialistas en Detección y Tratamiento de PTSD de Estados Unidos, los cuales desarrollaron este programa con los Veteranos de Vietnam.

A partir de la aprobación de este Programa, comenzaron a surgir distintos pedidos de documentación al Departamento Técnico y fue así como se plantearon los pasos a seguir:

1. Relevar, en base a un padrón emitido por el ANSES, a todos los Veteranos de Guerra. Luego de ser cargados en EPI INFO (sistema epidemiológico):
   a) Se debió recabar información para actualizar estos datos (excluir los fallecidos, actualizar domicilios, etc.)
   b) Discriminarlos por Regiones Sanitarias.
   c) Discriminarlos por partidos.

2. Realizar el programa, especificando sus metas, objetivos, funciones, etc.

3. Establecer las pautas de funcionamiento.

4. La Consultora debió realizar capacitación en Detección y Tratamiento en Estrés Postraumático, a los profesionales de cada hospital de cada Región Sanitaria.

5. Se estableció la confección de mapas de cada una de las Regiones Sanitarias.

6. De acuerdo a la cantidad de veteranos de cada partido, se afectaron los correspondientes hospitales, debiendo asignar a cada uno de ellos nuevos becarios profesionales y veterano captador en cada hospital.

7. Se debió asignar a cada región sanitaria un veterano becario captador regional que presta servicios junto con el Coordinador de Salud Mental de cada región.

8. Se convocó a los 4.800 (aproximadamente) veteranos para que asistan a los hospitales más cercanos a su domicilio, a fin de hacer el catastro de salud psicofísica.

9. En este momento el programa está en pleno funcionamiento, sobre todo en las Regiones Sanitarias V, VI, VIIa, VIIb y XI, en donde ya se han formado los Rap

Group (grupos en donde se tratan, con la coordinación de un profesional, a aquellos que como resultado del catastro se les ha detectado PTSD u otro diagnóstico psiquiátrico).

10. En forma permanente los hospitales afectados y los Rap Group ya constituidos deben informar a la Sede Central acerca del funcionamiento del Programa.

Para manejar toda esta información, se debió trabajar en distintos sistemas:

a) Para relevar a los veteranos se utilizó el EPI INFO (para nuestro trabajo de integración, esos archivos los exportamos al Excel a fin de trabajar con una viewer parcialmente conocida de Microcosmo).

b) Para cada región sanitaria hay un mapa de la provincia de Buenos Aires con dicha región sombreada. A su vez por cada una de las regiones hay un mapa ampliado de la misma dividido en partidos. Los mapas son archivos BMP.

c) Las planillas de funcionamiento de los Rap Group, el seguimiento de los Catastro, los recursos afectados a cada Región y el Programa están elaborados en Word.

d) La introducción explicativa acerca del Programa de Salud y sus pautas de funcionamiento están en formato RTF.

Esta información fue integrada con Microcosmo Plus.

⇒ Nuestra aplicación se llama **SALUD**, organizada en una estructura lógica que tiene las siguientes ramas:

INTRODUCCION
DOCUMENTOS
PLANILLAS
REGIONES

⇒ La misma consiste de:

- Un archivo de introducción con formato TXT, definido como documento Start up (documento que es mostrado cuando arranca la aplicación SALUD).

- A partir de este documento se crearon enlaces integrando toda la documentación descripta anteriormente y utilizando los tres tipos de viewers con las cuales trabaja Microcosmo.

  - Viewers totalmente conocidas:
    Archivos con formato TXT, RTF, HTM, ANI (creado con el Generador de Animaciones provisto por Microcosmo), BMP y MMC (creado con el Generador de Tour de Microcosmo).

- Viewers parcialmente conocidas:
  Archivos con formato XLS y DOC (la funcionalidad de hipermedia dada a estos tipos de archivos se logró a través de las macros proporcionadas por Microcosmo).

- Viewers desconocidas:
  Archivos con formato CDR (se utilizó el Corel Draw para realizar una pantalla de presentación y se integró a través del uso del portapapeles).

- Todos los documentos integrados están organizados en una carpeta llamada SALUD, la cual incluye las sub-carpetas ANIMACION (contiene los documentos que forman cada frame de la animación), IMAGEN (contiene los mapas y la imagen de presentación) y DOCUMENTOS (contiene todas las planillas y archivos de texto).

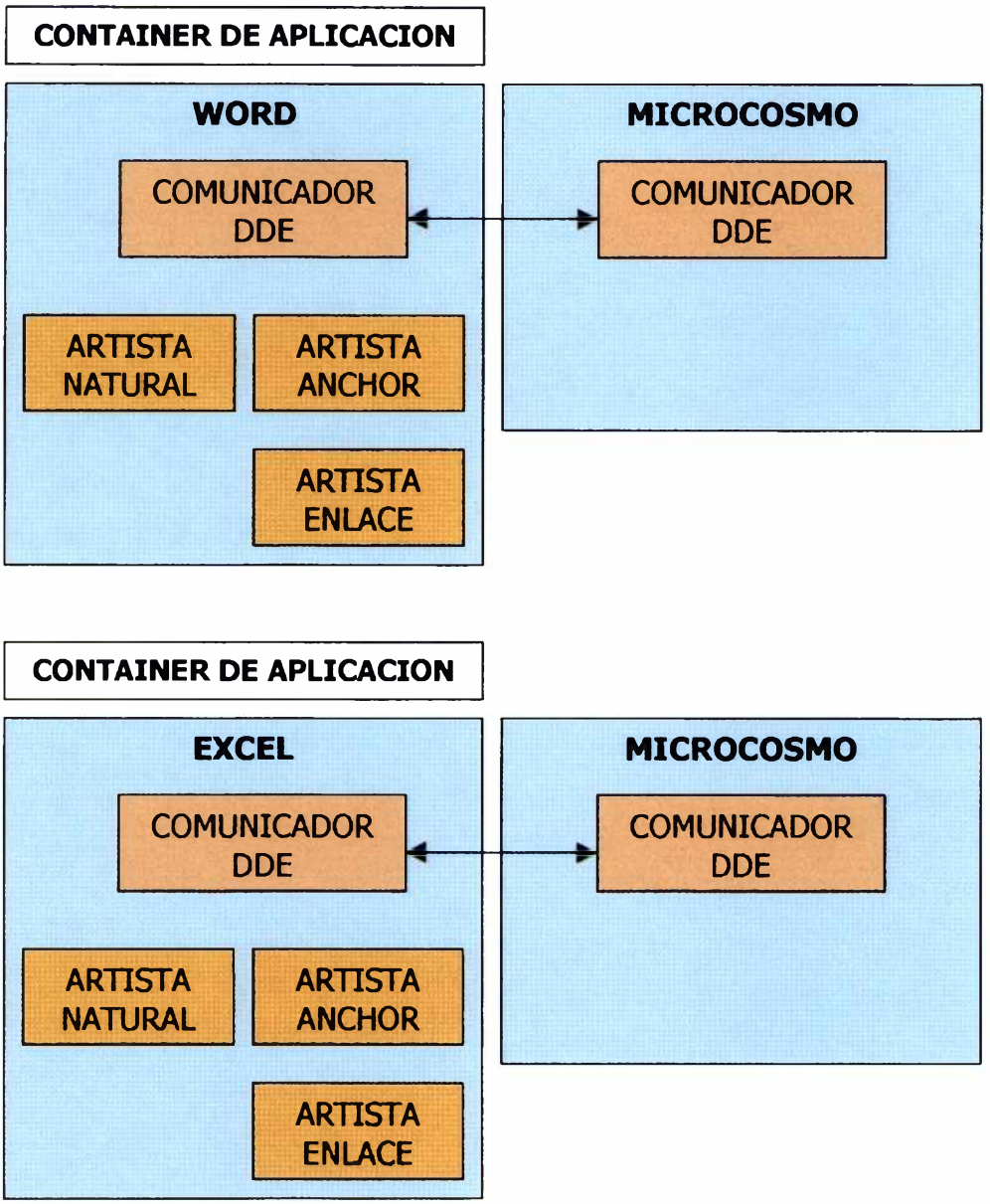
- Dentro de la carpeta SALUD se encuentra almacenada la linkbase que consiste de tres archivos:

  Salud.ddf: archivo de definición de datos que contiene la localización del archivo Salud.nix y Salud.raw.

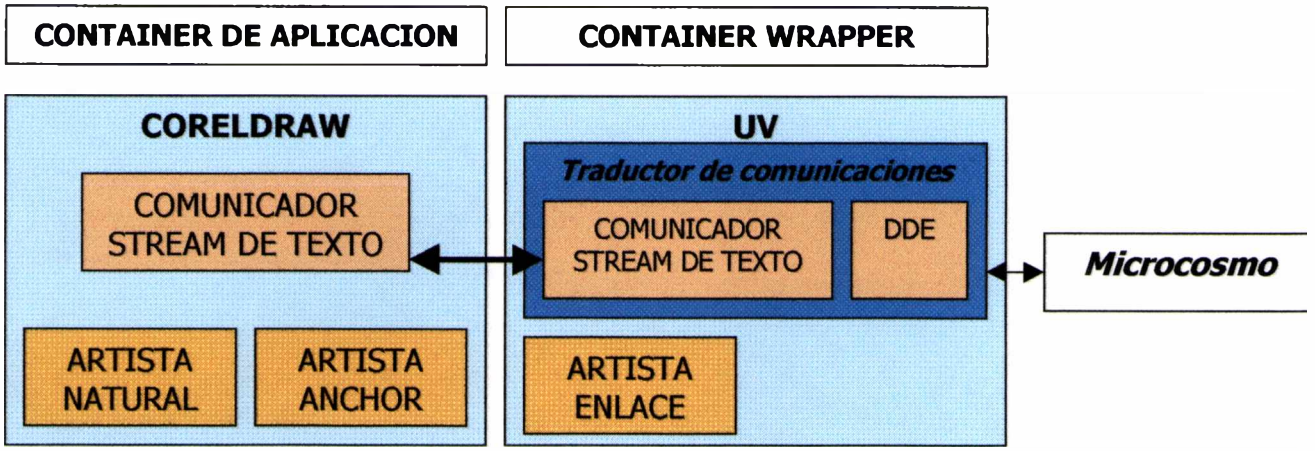
  Salud.raw: contiene todos los datos del enlace. Cuando un enlace es creado los datos del enlace son situados en este archivo.

  Salud.nix: es construido por Microcosmo y si se pierde un nuevo archivo.nix se crea cuando se lee la linkbase.

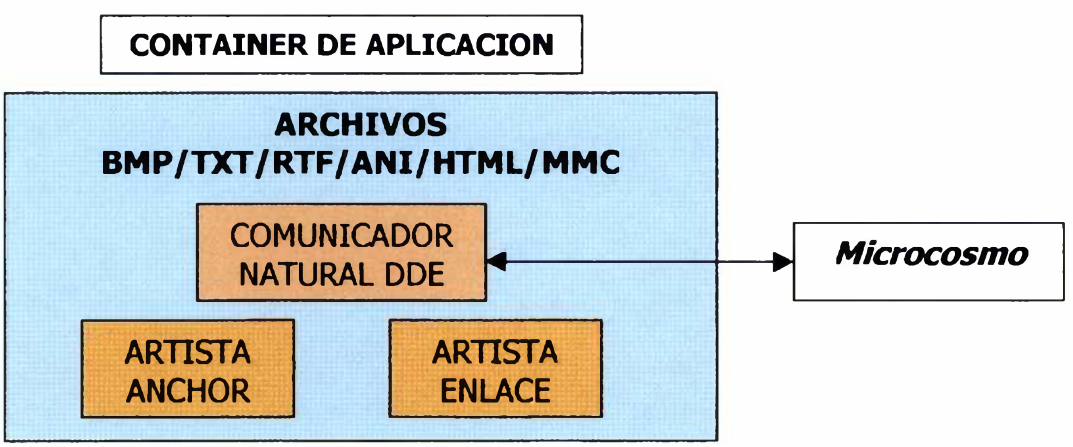
2.13. LA ARQUITECTURA DE NUESTRA INTEGRACION



Son viewers parcialmente conocidas de Microcosmo a las cuales se les incorpora en su barra de herramientas, a través de la instalación de las macros provistas por el paquete de Microcosmo, el menú de acción por medio del cual se accede a la funcionalidad de hipermedia. Así pasan a ser viewers conocidas. Se comunican con el sistema de hipermedia abierto a través del DDE. Estas dos aplicaciones cuentan con el artista natural y el artista anchor. A partir de la instalación de las macros cuentan con el artista enlace.



El CorelDraw es un ejemplo de una viewer desconocida de Microcosmo. Para ser integrada fue agregado un nuevo tipo de viewer dentro de la configuración de Microcosmo. Se comunica con el sistema de hipermedia a través de un stream de texto (el cual se copia al clipboard), siendo intermediaria entre la aplicación y Microcosmo, la Viewer Universal. La aplicación cuenta con el artista natural y el artista anchor, la Viewer Universal aporta el artista enlace.



Estas son viewers totalmente conocidas de Microcosmo que fueron integradas en nuestro trabajo, por lo tanto cuentan con el artista anchor y el artista enlace. Se comunican con Microcosmo a través de su comunicador natural.

3. CONCLUSIONES

3.1. VENTAJAS

Las ventajas que hemos encontrado, en el trabajo de integración con Microcosmo son las siguientes:

- Al manejarnos con grandes volúmenes de información, se hace dificultosa la navegación a través de la misma, haciendo solamente uso de la estructura de directorio. Los sistemas de hipermedia abiertos nos permiten fácil navegación a través de sistemas de información grandes.

- Los enlaces son mantenidos en forma separada del dato, el cual continúa accesible para la aplicación original que lo creó. Al mantener esta información separada el agregado y borrado de enlaces es simple y además reduce la posibilidad de enlaces inactivos.

- Como resultado de separar el servicio de enlace de los archivos mismos, los sistemas de hipermedia abiertos permiten al usuario tener dispositivos de seguimiento de enlace y otras ayudas de navegación.

- Es fácil acceder a las linkbases y editar la información de los enlaces.

- En estos sistemas es posible tener más de una linkbase a la vez. Una configuración común de Microcosmo es tener una linkbase que contiene un conjunto de enlaces sobre un conjunto de documentos definidos por el autor original y cada usuario puede tener su propia linkbase en la cual almacenan sus propios enlaces y anotaciones.

- El uso de enlaces genéricos y enlaces computados reduce el esfuerzo de creación.

- Se pueden hacer enlaces a archivos de sólo lectura, tales como los almacenados en CD-ROM, *.WAV, *.ANI.

- Microcosmo permite utilizar los formatos de datos y localizaciones de almacenamientos existentes. Integrar información sin necesidad de hacer procesos de conversión a formatos específicos, es decir los documentos mantienen sus formatos naturales y su ubicación original.

- Cuando se accede a un documento (mostrado por una viewer parcialmente conocida o desconocida) haciendo un seguimiento de enlace, éste puede ser modificado haciendo uso de todas las herramientas provistas por la aplicación que lo creó.

- Las viewers conocidas de Microcosmo cuentan con un botón que permite retornar al documento anteriormente visitado, mientras que en las viewers parcialmente conocidas o desconocidas esto se logra haciendo uso de la función *salir*, provista por la aplicación.

- Usando el filtro HISTORY, quien mantiene el recorrido de los documentos ya visitados, permite al usuario volver a alguno de ellos sin necesidad de realizar nuevamente todo el recorrido.

- Cualquier aplicación que no se encuentra dentro de las viewers total o parcialmente conocidas de Microcosmo puede ser incorporada como una viewer desconocida. Para ello se requiere un cambio en la configuración de Microcosmo, agregando un nuevo tipo, lo cual es muy sencillo.

- Ofrece la posibilidad de crear enlaces desde un punto específico a otro, por ejemplo desde o hasta una celda de una planilla Excel o una palabra o frase de un documento Word.

- Microcosmo es un sistema que puede ser utilizado tanto en un entorno de PC como en red.

- Un mismo anchor origen puede estar enlazado a múltiples anchors destinos. Al hacer un seguimiento de enlace desde ese anchor, en un filtro llamado RESULT BOX se listan todos los anchors destinos disponibles pudiéndose seleccionar entre uno de ellos.

Desde el punto de vista del modelo distribuido de Microcosmo se deduce:

- Se extendió la arquitectura modular para crear un sistema de hipertexto distribuido que pueda explotar esa modularidad.

- Provee un rango flexible de configuraciones distribuidas, permitiendo al sistema ser utilizado en una amplia variedad de formas. Los usuarios pueden interactuar con otros compartiendo bases de datos de enlaces y documentos o el sistema puede proveer un servicio de almacenamiento de hipertexto central que los usuarios pueden incorporar en sus configuraciones locales. Esta flexibilidad permite a los usuarios una mejor elección de la forma en que sus ambientes de hipertexto están configurados más que proveer un simple modelo distribuido con funcionalidad fija y restricciones para el usuario.

3.2. DESVENTAJAS

Los inconvenientes encontrados en este trabajo de integración fueron:

- En el caso de las viewers conocidas de Microcosmo no ofrece la función de impresión.

- Cuenta con un Help que no ofrece una organización adecuada para su recorrido. No es así el caso del Help de Microcosmo 3.1 ya que este tiene sus ítems numerados.

- En las viewers desconocidas o archivos de sólo lectura, los enlaces sólo pueden hacerse al inicio del documento.

- Microcosmo es un programa de 16 bits y por lo tanto los nombres de los archivos deben tener el formato 8.3. Si se trabaja con Windows 95 ó NT surgen problemas con nombres de archivos largos, por lo cual estos archivos deben ser renombrados antes de ser importados al DCS de Microcosmo.

- Si al crear una aplicación no fue definida para trabajar con enlaces computados, no es posible incorporar luego esta opción al menú de acción.

- Si un documento, que es mostrado por una viewer parcialmente conocida, contiene un enlace en algún punto específico y éste es modificado de tal manera que afecta la posición de este punto, el cambio de posición no se refleja automáticamente en la linkbase, por lo cual se debe editar el enlace y actualizarlo. Lo mismo sucede cuando se quiere borrar un enlace.

- Al ingresar a Microcosmo se debe elegir entre una de las aplicaciones existentes, sólo están disponibles los enlaces de esta aplicación. Para poder acceder a los enlaces de otra aplicación se debe cerrar el sistema Microcosmo y volver a ingresar al mismo eligiendo la aplicación deseada.

BIBLIOGRAFIA

Hypertext and hypermedia. Jacob. Nielsen. Pág. 1.

Extendiendo el Modelo Microcosmo a un ambiente Distribuido. Gary Hill, Wendy Hall. Universidad de Southampton. "ECHT 94 Proceedings".

Hacia un Modelo distribuido para Microcosmo. Gary Hill, Wendy Hall, D. De Roure. Universidad de Southampton. Material de Internet. 1994

El Servicio de enlace Microcosmo. Wendy Hall, Hugh Davis, Adrian Pickering, Gerard Hutchings. Universidad de Southampton. "Hypertext´93 Proceedings". Pág. 231.

Servicio de enlace de Microcosmo. Wendy Hall, Hugh Davis, Gary Hill. Universidad de Southampton. "Hypertext´93 Proceedings". Pág. 256.

Servicio de enlace de hipermedia light. Wendy Hall, Hugh Davis, Simon Knight. Universidad de Southampton. "ECHT 94 Proceedings". Pág. 41.

Control de versión en Microcosmo. M. Melly, W. Hall. Material de Internet. Año 1995/96.

Hacia un ambiente de información integrada con sistemas de hipermedia abiertos. Wendy Hall, Hugh Davis, Gary Hill, I. Heath, R. Wilkins. Universidad de Southampton. "ACM ECHT Conference". 1992

Servicios de información abiertos. Hugh Davis, Gary Hill, Wendy Hall, D. De Roure, L. Carr. Universidad de Southampton. Material de Internet. Año 1996.

Extensiones de Microcosmo a la World Wide Web. Wendy Hall, D. De Roure, L. Carr. Universidad de Southampton. Material de Internet. Año 1994.

Servicio de enlace de hipermedia distribuido. Gary Hill, Wendy Hall, D. De Roure, L. Carr. Universidad de Southampton. Material de Internet. Año 1996.

La historia del proyecto Microcosmo. Wendy Hall. Universidad de Southampton.. Material de Internet. Año 1997.

Microcosmo: un sistema de hipermedia abierto. D. De Roure, G. Hutchings. Universidad de Southampton. Año 1994.

Microcosmo TNG: Una arquitectura distribuida para soportar aplicaciones de hipermedia reflexivas. Wendy Hall, D. De Roure, S. Goose, J. Dale. Universidad de Southampton. ACM Hypertext ´97. Pág. 226.

Chimera: Hipertexto para ambientes de software heterogéneos. Kenneth M. Anderson, Richard N. Taylor, E. James Whitehead. Universidad de California. Echt ´94 Proccedings. Pág. 94.

Un modelo arquitectural para integración de aplicaciones en ambientes de hipermedia abiertos. E. James Whitehead. Universidad de California. Hypertext ´97. Pág. 1.

Integrando sistemas de hipermedia abiertos con la World Wide Web. Kenneth M. Anderson. Universidad de California. Hypertext ´97. Pág. 157.

Primeras Jornadas de Bases de Datos y Nuevas Tecnologías en Informática. Departamento de Informática. UNLP. Año 1993.

MATERIAL DE INTERNET:

Microcosmo.

Microcosmo: una visión técnica.

Microcosmo: Qué es?, Qué hace?, Cómo usarlo?.

Evaluación de la versión 3.0 de Microcosmo para Windows.

Una versión distribuida de Microcosmo.

Filtros y documentos distribuidos.

Integración con la WWW.

Usando la WWW en Microcosmo.

Entregando material de Microcosmo vía la WWW.

Aplicación del modelo Microcosmo a material WWW.