Maestría en Informática
PEDECIBA

Mantenimiento de bases de datos alimentadas con páginas web

Autor: Miriam Steiner
Tutor: Dr. Alejandro Gutiérrez

Facultad de Ingeniería, Universidad de la República
Montevideo, Uruguay

20 de febrero de 2003

Resumen

Se presenta un mecanismo semi-automático para mantener actualizada una base de datos relacional cargada con información contenida en tablas de páginas web. Se parte de un mecanismo de extracción en donde el usuario establece una correspondencia entre el esquema relacional, que representa el dominio de interés, y las columnas de las tablas presentes en una página web. En este trabajo nos concentramos en el problema de mantener los datos de la base actualizados frente a cambios en las páginas que se usaron para cargarla. Se presenta un relevamiento de trabajos relacionados con esta problemática y el mecanismo utilizado para identificar y repercutir los cambios de las páginas en la base de datos utilizando la correspondencia definida por el usuario.

Palabras claves: base de datos, páginas web, extracción, mantenimiento.

Indice

1. Introducción

La web es hoy en día una de las principales fuentes de información para muchas personas. En ella se puede encontrar información comercial, como por ejemplo pueden ser productos, precios, tiempos de entrega y disponibilidad. Se puede encontrar información de actualidad, como publicaciones de libros, diarios y revistas. Se puede encontrar información de nivel académico sobre investigaciones y reportes técnicos. Se puede encontrar información turística, como pueden ser lugares para visitar, agencias de viajes, hoteles, museos y horarios de transportes. Es tan amplio el contenido de la web que es difícil detallarlo o agruparlo completamente sobre la base de ejemplos. Prácticamente cada empresa, institución comercial o educativa coloca su información en la web, como consecuencia una persona encuentra muchos sitios con datos pudiendo contener información de su interés.

En el ambiente web un problema es el tiempo que una persona necesita para obtener la información que desea y la cantidad de sitios que debe visitar. Además, cada sitio es diferente, con su propia interfase y características.

Un enfoque para encontrar una solución al problema mencionado anteriormente es disponer de una base de datos local al usuario, lo cual tiene varias ventajas. Se evita que el usuario tenga que recorrer varios sitios para encontrar la información que busca, donde cada sitio tiene su topología y presentación propia, ofreciéndole por lo tanto al usuario una unificación de la interfase de acceso. Se pueden realizar consultas complejas sobre la información, como por ejemplo obtener el mejor proveedor de un determinado producto. Se mejora el tiempo de respuesta por hacer la consulta en forma local. Se mantiene la información disponible si alguno de los sitios momentáneamente no está accesible. Por último, la base de datos local puede ser tomada como base operacional permitiendo construir un data warehouse conteniendo información extraída de la web.

Por la naturaleza dinámica de la web un problema que presenta este enfoque es mantener actualizada la base de datos. Si la información que obtiene el usuario ya no se corresponde con la existente en la web puede que no le sirva en el momento de realizar una compra, y de nada le sirve para ponerse al día con la información. Por lo tanto es necesario un proceso de actualización de la base de datos. En este proceso de actualización hay que considerar varios aspectos. Un aspecto a considerar puede ser la estrategia de consulta a los sitios web utilizada, llamada "pull" si el usuario solicita las páginas al servidor y "push" si el servidor le envía las páginas al usuario sin un pedido previo. Otro aspecto es el momento en que debe realizarse la actualización.

Por el gran volumen de información que manejan los sitios, intuitivamente podemos deducir que internamente los datos son manejados con aplicaciones que luego se encargan de hacer la publicación en la web. Lo cual implica que en muchos casos la información al usuario final se presentará en forma semi-estructurada con la utilización de tablas y listas en lugar de texto simple. Con esta idea presente, en este trabajo se plantea extraer la información que al usuario le interesa a partir de la información publicada y contenida en tablas de páginas web. Para ello se parte de un mecanismo de extracción en donde el usuario establece una correspondencia entre el esquema relacional representando su dominio de interés y las columnas de las tablas presentes en una página. Esta correspondencia se define para cada una de las páginas de las cuáles se extrae la información. Nos concentramos en el problema de mantener (en forma semi-automática) actualizados los datos de la base frente a cambios en las páginas que se usaron para cargarla.

Este informe se organiza de la siguiente manera. En la sección ***2*** se presenta la descripción del problema general. La sección ***3*** contiene un relevamiento de trabajos relacionados con esta problemática para tener una visión de las soluciones existentes. En la sección ***4*** se detalla el mecanismo de mantenimiento propuesto en esta tesis y en la sección ***6*** se presenta una evaluación del mismo. Finalmente, en la sección *7* se presentan las conclusiones y trabajos futuros.

2. Descripción general del problema

Se pretende cargar una base de datos relacional con información extraída de páginas web y mantenerla actualizada. El escenario de trabajo se ilustra en la Figura 1.

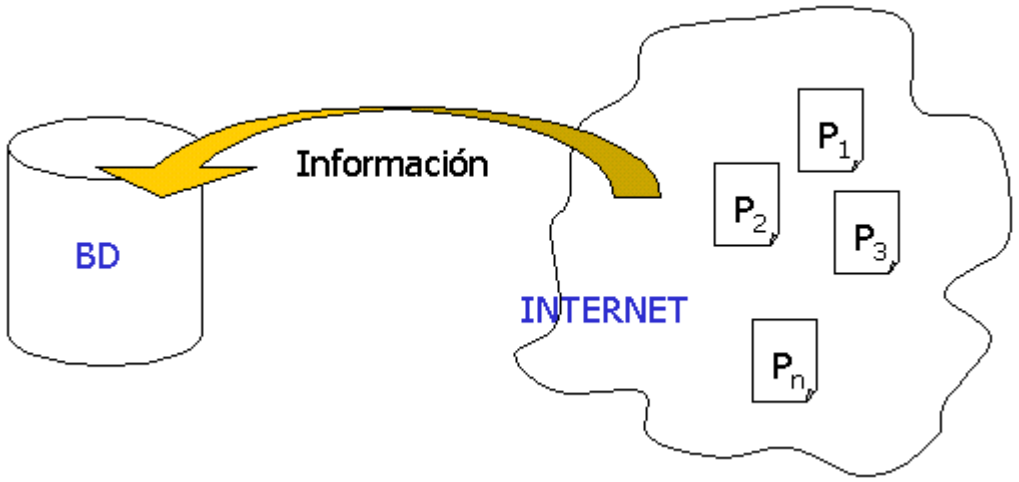


Figura 1 - Escenario del Trabajo

En Internet consideramos un conjunto de páginas { p_i } que contienen la información de interés para el usuario. Obtener este conjunto de páginas es el primer problema que se debe resolver. Una vez identificado el conjunto de páginas se pretende guardar en la base de datos la información de interés para el usuario. Es decir que no se guardan páginas web, por lo tanto es necesario identificar y extraer la información de interés existente en las mismas. Luego hay que cargar en la base de datos la información extraída. Y por último, la base de datos debe mantenerse actualizada para reflejar los cambios ocurridos en Internet en las páginas web del conjunto considerado. En este proceso descripto hay una serie de problemas o puntos que se deben resolver, los cuales se tratan en la próxima sección que es un relevamiento de lo existente que permite entender mejor cada punto. A continuación simplemente se enumeran y se indican en la Figura 2 dentro del escenario del trabajo:

1. Encontrar las páginas con información de interés.
2. Extraer la información de las páginas.
3. Cargar la base de datos.
4. Detectar cuando cambia una página.
5. Identificar los cambios de la página.
6. Propagar los cambios a la base de datos.

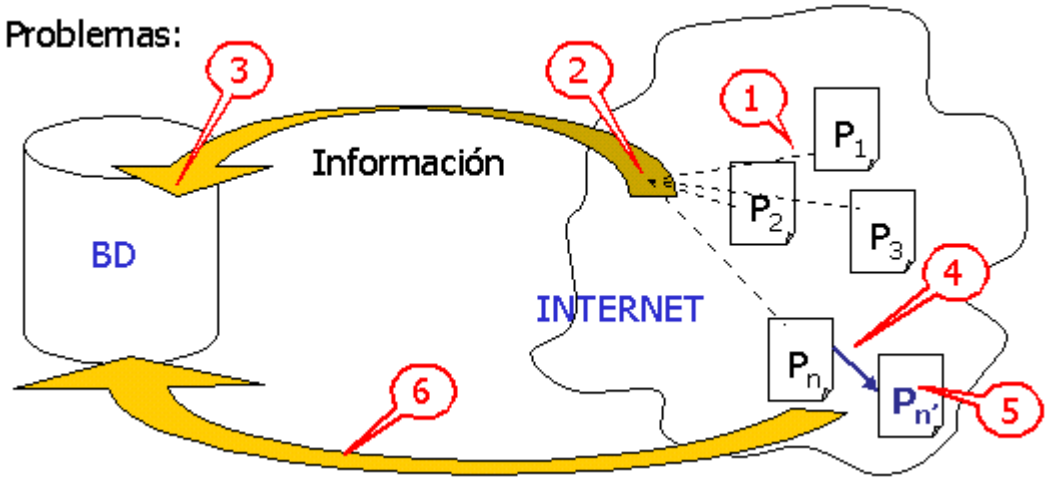


Figura 2 - Problemas a Resolver

3. Relevamiento

Esta sección presenta un relevamiento de los trabajos existentes relacionados con el tema Mantenimiento de bases de datos alimentadas con páginas web. Se consideraron herramientas que tratan o se relacionan con alguno de los problemas mencionados previamente y que se describen más adelante.

En la sección ***3.1*** se presenta el resultado del relevamiento consistente de una recopilación, breve descripción y análisis de las soluciones identificadas a cada uno de los problemas enumerados en la sección ***2***. La sección ***3.2*** contiene una lista de las herramientas encontradas. En la sección ***3.3*** se presentan cuadros comparativos de algunas herramientas basándose en los problemas mencionados en la sección ***3.1***. Finalmente en la sección ***3.4*** se presentan las conclusiones.

3.1. Soluciones encontradas

En las siguientes subsecciones se presentan posibles soluciones para cada uno de los problemas enumerados previamente en la sección 2 junto con una descripción del mismo.

3.1.1. Encontrar las páginas con información de interés

Este problema consiste en determinar el conjunto de páginas web que contienen información de interés para el usuario, teniendo en cuenta dos aspectos: el método utilizado y el momento en que se realiza la búsqueda.

Los métodos utilizados para especificar las páginas con información de interés son presentados a continuación:

- Recibir una lista de páginas web como parámetro al ejecutarse.
- Registrar una lista de páginas web.

- Registrar una lista de sitios y definir una estrategia para determinar las páginas de cada sitio que se consideran. Por ejemplo: todas las páginas, las páginas accesibles a través de links desde la principal hasta cierto nivel, las páginas que cumplen un criterio.
- Utilizar un buscador existente o uno propio, en cuyo caso hay que definir las consultas que se deben realizar.

La diferencia entre el primer caso que recibe la lista de páginas web como parámetro y el segundo donde se registra un lista de páginas web, es que en el primero no se guarda la lista.

Tener una lista fija de páginas permite detectar si una página no existe más, pero no permite detectar nuevas páginas. Tener una lista fija de sitios permite detectar si un sitio, o una página del mismo, ya no existe y permite encontrar páginas nuevas si éstas pertenecen a un sitio registrado. Sin embargo, no permite detectar nuevos sitios. La ventaja de registrar una lista radica en la facilidad de implementación. La desventaja se encuentra en la tarea previa de búsqueda dejada al usuario. Dicha tarea puede llegar a ser complicada según el volumen de sitios a manejar.

Un aspecto a considerar con respecto a tener una lista registrada, ya sea de páginas o sitios, es su mantenimiento. Las modificaciones a dicha lista pueden ser realizadas únicamente por el usuario o también por la herramienta en forma automática con algoritmos a tales efectos. Esta segunda opción de permitir que la herramienta modifique la lista tiene como ventaja que se pueden detectar páginas o sitios nuevos en forma automática, la eficiencia depende del mecanismo implementado. También causa que no se consulten, en forma reiterada, páginas o sitios que ya no existen. Pero tiene como dificultad la complejidad de la herramienta, que depende del algoritmo para detectar nuevas páginas, por ejemplo si al considerar una página se recorren los links de la misma implicaría implementar algoritmos de detección de loops, estrategias para finalizar la recorrida de las páginas y además ignorar los links típicos como la página inicial (home). Hay que considerar también que se puede llegar a trabajar con páginas que ya no son de interés. En definitiva, existe una gran dificultad de automatizar completamente la tarea de mantenimiento de la lista sin intervención del usuario.

Obtener las páginas utilizando buscadores permite encontrar nuevas páginas de interés, pero se incrementa la complejidad de la herramienta. Por ejemplo si la cantidad de páginas encontradas es muy grande, se deben procesar todas o sólo algunas, también hay que determinar si la página de recuperada es de interés o no.

Con respecto al momento de la búsqueda, las posibles soluciones son:
- Indicación del usuario.
- Planificación fijando la fecha y/u hora.
- Planificación con frecuencia fija.
- Planificación con frecuencia variable. Esta frecuencia puede estar determinada por un criterio como por ejemplo la carga de trabajo.

3.1.2. Extraer la información de las páginas

Este problema consiste en determinar el mecanismo para extraer la información de las páginas. Teniendo en cuenta los siguientes aspectos: la visión o modelo que se tiene de la web, que se toma en cuenta de las páginas y el método para realizar la extracción.

Con respecto a la visión o modelo que se tiene de la web las posibilidades son:

- Considerar que la web es una única base de datos de donde extraer información.
- Considerar que cada sitio es una base de datos.
- Considerar que cada página es independiente.

En cuanto a que se toma en cuenta de la página las posibilidades son:

- Todo lo existente en la página. Como por ejemplo texto, imágenes, links.
- Sólo el texto existente en la página.
- El texto que se encuentra en determinada posición o dentro de determinadas estructuras de la página. Como por ejemplo títulos, listas, tablas.
- El texto relacionado con algún tema. Determinado por ejemplo con heurísticas y/o palabras claves.

Algunos de los métodos utilizados para la extracción de la información son presentados a continuación:

- Utilizar los tags HTML (HyperText Markup Language), es decir las marcas del lenguaje en que están escritas las páginas web.
- Buscar palabras claves.
- Utilizar gramáticas.

En cuanto a la visión o modelo de la web considerar que es una única base de datos enorme es un concepto sencillo y da una visión homogénea de algo que en la realidad es complejo. La dificultad es que se pueden modelar muy pocas características generales. Esto es debido la gran heterogeneidad de los sitios y su autonomía.

Considerar que cada sitio es una base de datos permite modelar más características con un mayor grado de información en comparación con la opción anterior. Esto es bajo la hipótesis razonable de que dentro de un sitio, por razones de legibilidad y mantenimiento de las páginas, se mantiene un "estilo" homogéneo. Por ejemplo dentro de un sitio puede que todas las páginas tengan un título u otra característica que se desea modelar. La dificultad es que existen páginas que pueden no ajustarse al modelo y que no todas las páginas de un sitio son de interés.

Considerar que cada página es una fuente de información implica tratar a cada página como un documento. Esta visión permite utilizar técnicas más específicas, como pueden ser: la estructura de la página, heurísticas o técnicas de tratamiento del lenguaje natural.

En cuanto a lo que se toma en cuenta de la página surge que considerar todos los elementos de la página asegura que no se omite información que puede ser de interés. Pero tiene como inconveniente la complejidad necesaria para poder manipular todo tipo de formatos, como por ejemplo textos, imágenes, animaciones, archivos.

Considerar todo el texto presente en la página implica considerar datos que no son de interés. Esto es debido a que la mayoría de las páginas tienen información general del sitio, por ejemplo menús, propaganda de la empresa o institución, datos sobre la actualización o autor de la página. Esta opción tiene la complejidad de que el texto se encuentra en distintas estructuras, como ser títulos, párrafos, links, tablas, listas.

Considerar el texto que se encuentra en determinada posición o estructura permite identificar lo que se quiere extraer fácilmente. Además se puede observar que los sitios que publican cierta cantidad de información importante tienden a hacerlo en forma estructurada, para así facilitar el mantenimiento o porque las páginas se generan en forma automática. Tiene como desventaja que puede ignorar información existente que sea relevante, si la dicha información no se encuentra en la estructura considerada. Es necesario un conocimiento previo de la estructura de las páginas web.

Considerar el texto relacionado con algún tema implica tener un conocimiento del mismo. Por ejemplo para poder implementar las heurísticas necesarias o crear el conjunto de palabras claves a utilizar. La implementación del mecanismo de extracción puede llegar a ser complejo. Con este método no es necesario conocer la estructura de las páginas web.

3.1.3. Cargar la base de datos

Este problema consiste en cargar la base de datos con la información extraída previamente de las páginas web. Dicha información debe adecuarse a un modelo y determinar la parte que se guardará en la base de datos y el método con el cual se realizará dicha tarea.

Las posibilidades para modelar la información extraída de las páginas web son:

- No utilizar modelo, es decir trabajar simplemente con un texto.
- Utilizar el modelo relacional o alguna extensión del mismo.
- Utilizar el modelo orientado a objetos o alguna extensión del mismo.
- Utilizar un modelo propio.

En cuanto a la parte de la información extraída que se guardará en la base de datos se tienen las siguientes opciones:

- Todo lo extraído de la página.
- Sólo lo relevante para el usuario.

El posible método para guardar la información en la base de datos es:

- Definir una correspondencia entre el modelo utilizado para la información extraída y la base de datos.

De los posibles modelos de la información se puede observar que utilizar extensiones del modelo relacional o el modelo orientado a objetos, permite utilizar los conocimientos adquiridos en el área de Base de Datos y adaptarlos al ambiente de la web. Con lo cual el punto de partida es conocido y no hay que comenzar de cero. Sin embargo, en algunos casos puede ser una limitación tratar de adaptar algo que no se creo con el fin que se le pretende dar.

Utilizar modelos nuevos, por ejemplo de tipo gráfico, permite fácilmente modelar la topología de links que es uno de los conceptos principales del ambiente web. Aquí cabe destacar que no todos los links se deben modelar. Esto es porque muchos links en realidad sólo existen para facilitar la navegación o legibilidad de las páginas, como por ejemplo, los links que llevan al principio de la página o a una sección de la misma, los link cuyo destino es la página inicial (home). Si se modelaran todos los links existentes en las páginas existirían ciclos en el modelo. Los ciclos no aportan información pero aumentan la complejidad de las herramientas, ya que es necesario la introducción de algoritmos para la detección de loops y de terminación de las recorridas.

Del modelo que se aplique depende el lenguaje de consulta utilizado para manejar los datos extraídos de las páginas. Si el modelo es una extensión de los modelos relacionales u orientado a objetos el lenguaje puede ser una extensión de los lenguajes existentes, como por ejemplo una extensión a SQL (Search Query Language), o puede ser un lenguaje particular desarrollado por la herramienta. Los lenguajes pueden ser declarativos, procedurales, lógicos, con técnicas de análisis del lenguaje natural. Lo que en cada caso tienen que definir un punto de equilibrio entre la sencillez y el poder de consulta que logran.

3.1.4. Detectar cuando cambia una página

Este problema consiste en determinar el momento en que una página web ha cambiado. Teniendo en cuenta los siguientes aspectos: el método utilizado y el momento en que se observan las páginas. Pero previamente hay que definir lo que se considera un cambio.

Las posibilidades en cuanto a lo que se considera un cambio en la página web son:

- Cualquier modificación de la página, ya sea en la información que contiene o en la presentación.
- Una modificación en la información de interés.
- Una modificación en las estructuras de interés.

Los métodos utilizados para detectar un cambio en una página web son presentados a continuación:

- Chequear si cambió la fecha de la página.
- Generar un valor basándose en el contenido de la página y compararlo con uno anterior.
- Aplicar directamente el algoritmo para identificar los cambios.
- Suscribirse al sitio para recibir los cambios, si el sitio considerado ofrece dicho servicio.

Sobre el momento en que se deben observar las páginas web las posibles soluciones son:

- Indicación del usuario.
- Planificación fijando la fecha y/u hora.
- Planificación con frecuencia fija.
- Planificación con frecuencia variable. Esta frecuencia puede estar determinada por un criterio como por ejemplo la carga de trabajo.

No es sencillo determinar lo que se considera un cambio ya que son variadas las modificaciones que pueden ocurrir en una página, incluyendo su alta y baja. Pero no todas las modificaciones afectan la información de interés para el usuario. Por lo tanto, sería deseable que si las siguientes modificaciones ocurrieran fueran ignoradas:

1. Modificaciones de estilo, como por ejemplo en colores o fuentes.
2. Modificaciones en las propagandas. Las propagandas en general existen en una página y cambian frecuentemente para mantener el interés del usuario en dicha página.
3. Modificaciones en las estructuras utilizadas con fines de presentación. Es común el uso de tablas con el propósito de presentación, por ejemplo para encolumnar valores.

Algunas herramientas distinguen entre determinar que ha ocurrido un cambio y la identificación de los cambios y otras no. Estas últimas, es decir las que no hacen la distinción, encuentran que la página web ha cambiado en el proceso de identificar los cambios al encontrar uno. Sin embargo se puede tratar primero de determinar de alguna manera si la página ha cambiado, por ejemplo para evitar tráfico de páginas o demoras analizando páginas que no han sido modificadas. Si se determina que la página ha sido modificada luego se procede a identificar los cambios.

El uso de suscripciones a las páginas de interés si estas lo permiten, para saber si han cambiado o recibir los cambios de dichas páginas depende del sitio y no de la herramienta a implementar. En la actualidad, pocos sitios ofrecen el servicio mencionado.

3.1.5. Identificar los cambios de la página

Este problema consiste en identificar los cambios de una página que fue o se presume modificada.

Las posibilidades en cuanto al método a utilizar para la tarea de identificar los cambios son:

- Comparar dos versiones de la página. La actual y una guardada previamente.
- Comparar la información extraída de la página actual con la información extraída de una versión anterior.
- Comparar la información extraída de la página con la información de interés almacenada en una base de datos.

La diferencia entre la segunda y la tercer alternativa es que la información de interés para el usuario almacenada en la base de datos puede ser sólo parte de la información extraída.

Del método seleccionado depende la complejidad de la implementación y lo que se necesita almacenar.

3.1.6. Propagar los cambios a la base de datos

Este problema consiste en determinar si hay que propagar un cambio a la base de datos o hay que ignorarlo porque no corresponde a la información de interés para el usuario. En caso de que hubiera que propagarlo hay que considerar la forma de realizarlo.

En cuanto a la forma para propagar un cambio, donde una posibilidad es ignorarlo, se tienen los siguientes:

- Propagación manual. Es el usuario que se debe encargar en cada caso de tomar las acciones necesarias.
- Propagación semi-automática. Donde en forma automática se propaga todo lo posible y para el resto se le permite al usuario tomar las decisiones correspondientes.
- Propagación automática. Donde sólo se propaga lo que es posible sin intervención del usuario.

En cuanto a la propagación de los cambios se puede argumentar que si la propagación se realiza en forma manual se deben notificar los cambios al usuario. Esto puede realizarse por ejemplo por medio de un e-mail o almacenándolos en alguna estructura que el usuario pueda consultar. Luego será responsabilidad del usuario tomar las medidas pertinentes para actualizar la base de datos.

Un mecanismo semi-automático tiene que implementar una interfase para el usuario que le permita en el proceso de actualización tomar las decisiones necesarias para propagar todos los cambios de interés.

3.2. Lista de herramientas

A continuación se presentan tres herramientas que observan páginas web registradas y notifican cuando ocurre un cambio. Al proceso de observar las páginas también se le puede llamar monitorear. La notificación se realiza por ejemplo vía e-mail. En algún caso se detalla la información que se le

envía al usuario. En otro caso se puede consultar un resumen de los cambios aunque no explican el método utilizado para la identificación de los cambios.

Mind-It

La tarea de monitorear las páginas la realiza una componente llamada **URL-minder**, luego otra componente llamada **Highlights** identifica los cambios de la página que se envían por e-mail. **URL-minder** está disponible on-line para monitorear páginas de Internet [1].

BuzzCity

El usuario registra una lista de páginas e indica en que horas deben ser consultadas para determinar si hay cambios e identificarlos, luego manda un e-mail con los resultados [2].

URLy Warning

Esta herramienta disponible on-line se ejecuta desde su sitio [3]. Un persona al ingresar al sitio debe registrar un usuario y las páginas que desea monitorear. Al entrar nuevamente el usuario visualiza la lista de las páginas donde se indica cuales cambiaron. Guarda una versión de las páginas modificadas con los cambios resaltados para que el usuario los pueda visualizar. Se puede bajar un cliente de esta herramienta para acceder al sitio con el fin de facilitar el trabajo.

A continuación se presentan dos herramientas que comparan dos versiones de una página web para identificar los cambios que han ocurrido.

HtmlDiff

Compara dos páginas html y el resultado es una nueva página con los cambios resaltados. Permite realizar alguna indicación como el de ignorar blancos e indicar si considera las mayúsculas y minúsculas iguales o no, lo que se conoce como "case sensitive". Está disponible on-line para comparar páginas de Internet [4].
TopBlend es una nueva implementación en Java [38] de HtmlDiff que destaca un nuevo algoritmo de comparación y la posibilidad de ver los cambios uno por uno [5].

HTML Compare

Compara dos páginas html y crea una nueva resaltando las diferencias. También puede comparar dos directorios. En este último caso crea una página en la que se indican los archivos nuevos, los borrados y los modificados. Para los archivos modificados coloca un link a la versión vieja, otro a la nueva y si se trata de un archivo html un link a la comparación de las dos versiones. Los directorios que se van a comparar deben estar en el PC o en una red local [6].

A continuación se presentan herramientas que almacenan información en una base de datos.

WebGUIDE (Web Graphical User Interface to a Difference Engine)

Esta es una herramienta para detectar cambios en páginas web y en la estructura de los sitios, es decir en los links existentes entre las páginas de un sitio. Soporta comparación recursiva, al comparar dos páginas también compara las páginas destino de los links,

aunque no especifica el mecanismo utilizado para parar la ejecución recursiva. Los usuarios también pueden explorar las diferencias de una página respecto a dos fechas dadas. Las diferencias de las páginas son resumidas automáticamente en una nueva página HTML y las diferencias en la estructura de links son mostradas mediante una representación gráfica [7].

Está compuesta por las siguientes herramientas o componentes:

AT&T Internet Difference Engine (AIDE)

Esta es una herramienta que detecta los cambios de las páginas web que monitorea, almacena las distintas versiones de las páginas, y luego compara dos versiones para presentar las diferencias entre las mismas. En forma automática compara la última versión de la página vista en el sistema por el usuario con la más reciente, pero se pueden comparar otras. Tiene tres módulos o componentes que son: **w3newer** para detectar los cambios, **snapshot** para almacenar las páginas y **HtmlDiff** para comparar y mostrar los cambios [8].

Ciao

Esta herramienta es un navegador gráfico que permite a los usuarios consultar y navegar los links existentes en el repositorio de documentos. Tiene un componente llamado **abstractor** que convierte los documentos fuentes para almacenarlos en una base de datos. Dicha conversión la realiza con un modelo propio de los datos que describe la estructura interna del documento, el cual incluye páginas, anclas, encabezados, imágenes, relaciones entre ellos. Tiene otro componente llamado **repositorio** que guarda versiones de los documentos y su correspondencia en la base de datos. Y por último, tiene una **internase gráfica** que permite consultar y visualizar la estructura de la información.

ACADEMIA

Este proyecto consiste de un agente que junta información desde la web con el fin de mantener una base de datos local y asegurar su correctitud. Como ejemplo trata de mantener una base de datos con información sobre contactos académicos, sus proyectos y publicaciones, aunque esto es configurable. La ejecución del agente es manejada por un perfil de extracción que le indica lo que debe extraer y la forma de hacerlo. Cuando el nivel de confianza es suficiente automáticamente modifica la base de datos para actualizarla según la información extraída [9].

ARANEUS

El objetivo de este proyecto es manejar la información de bases de datos y de la web en el mismo estilo de base de datos. El enfoque es generalizar la idea de vistas a la web, como herramienta de reestructuración e integración. El sistema es diseñado para soportar varias clases de aplicaciones como ser: (i) el acceso de alto nivel a los datos en la web, (ii) el diseño, implementación y mantenimiento de sitios web, (iii) aplicaciones cooperativas sobre la web. Las vistas de sitios web externos son materializadas localmente [10] al [21].

Generación Automática de una Base de Datos desde Documentos de la Web

El objetivo central de este trabajo es la extracción de información de documentos HTML y la consolidación de esta información en una base de datos. Se propone un mecanismo basado en una ontología del dominio, en patrones sintácticos típicos para la inferencia de

algunos tipos de datos y en heurísticas para la interpretación de títulos y tablas. Mediante este mecanismo se construye automáticamente una correspondencia entre elementos de un documento HTML y las entidades del dominio del usuario. Utilizando esta correspondencia, se transfiere la información extraída de la página web a la base de datos local [22].

WebCQ (CONTINUAL QUERY)

WebCQ está diseñado para detectar cambios a páginas web eficientemente y proveer una notificación personalizada de los cambios a las páginas de interés, indicando lo que cambió y en que forma. Los requerimientos de monitoreo de cambios indicados por los usuarios son modelados como consultas continuas sobre la web. Puede monitorear varios tipos de cambios en páginas estáticas o dinámicas. Ofrece la posibilidad de personalizar la notificación de los cambios y resumir los cambios que se quieren monitorear [23] [24].
Por más información sobre consultas continuas (Continual Query) ver [25] [26].

WebBeholder

El objetivo de este proyecto es brindar un servicio para encontrar y visualizar cambios sobre la web, basándose en una estructura de una comunidad de agentes cooperativos. Varios agentes y componentes en la comunidad interactúan con otros para lograr la meta de los usuarios del sistema. El sistema consiste de un agente proveedor de servicio que observa y detecta los cambios sobre la web, un número de agentes que representan cada usuario y un número de mediadores para negociar con el agente proveedor de servicio por los requerimientos de los agentes personales [27].

Clock

Es un prototipo para propagar las modificaciones, en forma incremental, de los datos de documentos XML (eXtensible Markup Language) almacenados. Es decir, reflejar correctamente las modificaciones de documentos XML externos en los datos XML cargados en una base de datos relacional [28].

Los siguientes son ejemplos de herramientas que monitorean las páginas de resultados de algún buscador.

TracerLock

Esta herramienta monitorea las páginas de resultados de AltaVista [34] a una consulta dada y notifica cuando aparecen nuevos resultados. Además proporciona los links más relevantes a la información consultada [29].

The Informant

Esta herramientas tiene dos posibilidades:

1. Monitorear las páginas de resultados de AltaVista [34], Lycos [35], Excite [36] e Infoseek [37] permitiendo especificar hasta tres consultas y el intervalo de tiempo para realizar el chequeo. Monitorea las 10 páginas de resultados más relevantes y notifica vía e-mail si alguna nueva página se encuentra en dichas páginas de resultados o si las páginas existentes han sido modificadas.
2. Monitorea páginas web, se pueden indicar hasta 4 páginas y el intervalo de tiempo para realizar el chequeo, notifica vía e-mail cuando alguna de ellas cambia [30].

A continuación se presentan ejemplos de herramientas que permiten extraer información de una página mediante la construcción de componentes de software para dicha tarea llamados wrappers.

JEDI

El objetivo de este proyecto es la creación de wrappers, componentes de software para extraer información, y mediadores, componentes de software para combinar información, desde diferentes fuentes de información independientes. Para los wrappers utiliza gramáticas con atributos que son evaluadas con una estrategia de parser tolerante a fallas, para tratar con gramáticas ambiguas y fuentes irregulares. Para los mediadores utiliza un modelo orientado a objetos genérico [31].

W4F (Word Wide Web Wrapper Factory)

Es una herramienta para construir componentes de software llamados wrappers que integran datos publicados en la web dentro de aplicaciones. Divide el proceso de desarrollo de los wrappers en dos fases una de extracción y otra de mapeo [32] [33].

3.3. Cuadros comparativos

En esta sección se comparan algunas de las herramientas con el fin de obtener una mejor comprensión de los problemas y las posibles soluciones planteados en la sección ***3.1***.

3.3.1. Clasificación general

Las herramientas mencionadas en la sección ***3.2*** se pueden clasificar en los siguientes grupos:

1. Las que trabajan con páginas web y notifican cuando cambian.
2. Las que dadas dos versiones de una páginas web identifican los cambios.
3. Las que cargan una base de datos según los requerimientos definidos.
4. Las que se encargan de la actualización de una base de datos dependiendo de que se les pase el cambio.

Según las descripciones previas hay herramientas que pertenecen a más de un grupo.

Cuadro comparativo indicando a que grupo de los mencionados corresponde cada herramienta:

Mind-It	Grupo 1 y 2, URL-minder grupo 1.
HtmlDiff	Grupo 2.
WebGUIDE	Grupo 1 y 2.
ACADEMIA	Grupo 3.
WebCQ	Grupo 1 y 2.
WebBeholder	Grupo 1 y 2.
Clock	Grupo 4.

3.3.2. Encontrar las páginas con información de interés

Cuadro comparativo sobre el método que utilizan las herramientas para encontrar las páginas de interés:

Mind-It	Por medio de una lista fija de páginas que cada usuario debe registrar.
HtmlDiff	Recibe como parámetros las dos páginas que tiene que comparar.
WebGUIDE	Por medio de una lista fija de páginas que cada usuario debe registrar y permite indicar que se considere todo un sitio.
ACADEMIA	Mantiene una base de datos de contactos académicos. La clave de cada contacto en la base de datos es el nombre y la URL (Uniform Resource Locator), es decir la dirección de su página. Por lo tanto cada vez que se ejecuta el agente lee de la base cada contacto, formado por el nombre y la URL, y busca la información en la web. Si no existe una URL asociada al contacto, o al buscarla no la encuentra, utiliza el buscador Alta Vista [34] para tratar de encontrar una página con el nombre y apellido del contacto. Esto no asegura que se recuperaran documentos relevantes. Busca la información en las primeras 10 páginas recuperadas, si encuentra información consulta al usuario sobre la confiabilidad de la misma. En resumen, utiliza direcciones fijas existentes en la base de datos, si no tiene la dirección o no existe utiliza el buscador Alta Vista.
WebCQ	Por medio de una lista fija de páginas que cada usuario debe registrar y permite indicar que se considere sólo parte de una página.
WebBeholder	Por medio de una lista fija de páginas que se registran por el agente del usuario.
Clock	No se encarga de este tema.

3.3.3. Extraer la información de las páginas

Cuadro comparativo sobre el modelo que la herramienta tiene con respecto a la web:

Mind-It	No tiene una visión global de la web cada página es independiente.
HtmlDiff	No tiene una visión global de la web cada página es independiente.
WebGUIDE	Sin información.
ACADEMIA	No tiene una visión global de la web cada página es independiente, aunque considera que los links llevan a otras páginas con posible información de interés, por lo tanto también las revisa.
WebCQ	No tiene una visión global de la web cada página es independiente.
WebBeholder	No tiene una visión global de la web cada página es independiente.
Clock	La web es un conjunto de documentos XML.

Cuadro comparativo en cuanto a lo que se toma en cuenta de las páginas para extraer información:

Mind-It	No extrae información.
HtmlDiff	No extrae información.
WebGUIDE	No extrae información.
ACADEMIA	Información sobre contactos académicos, es decir busca un tema específico.
WebCQ	Lo que indica el usuario.

WebBeholder	No extrae información.
Clock	No se encarga de este tema.

Cuadro comparativo sobre el método con el cual realiza la extracción de información de las páginas web:

Mind-It	No extrae información.
HtmlDiff	No extrae información.
WebGUIDE	No extrae información.
ACADEMIA	El proceso de extracción es controlado por un ***perfil de extracción*** (extraction profile) que le especifica la forma como la información va a ser extraída de los documentos. Basándose en una combinación de búsqueda de palabras claves ("matching" de términos) y medidas de proximidad. Medidas de confiabilidad son asociadas con los varios patrones de extracción, permitiéndole así al agente calcular puntajes de confiabilidad para los elementos extraídos. Dichos puntajes se calculan luego de extraer la información de todas las páginas encontradas para un mismo contacto (Ya que si un dato se repite en todas tiene un puntaje alto pero si es diferente debe tener un puntaje bajo).
WebCQ	La información es extraída basándose en los tags html, basándose en expresiones regulares, o considera la página completa. Todo depende de lo que debe monitorear. Si debe monitorear cualquier cambio recupera la página completa. Si debe monitorear tablas, listas, links o imágenes extrae la información contenida en los tags correspondientes a dichos elementos. Si debe monitorear una parte arbitraria del texto se basa en la expresión regular definida al indicarle lo que debe monitorear y extrae el texto que la verifica.
WebBeholder	No extrae información.
Clock	No se encarga de este tema. Recibe el cambio que debe propagar.

3.3.4. Cargar la base de datos

Cuadro comparativo sobre el modelo de la información extraída de las páginas web:

Mind-It	Sin información.
HtmlDiff	Sin información.
WebGUIDE	Sin información.
ACADEMIA	Utiliza una base de datos con el modelo orientado a objetos con su lenguaje para manipular los datos.
WebCQ	Habla de objetos sin mayor información.
WebBeholder	Sin información.
Clock	Se basa en el DOM (Document Object Model).

Cuadro comparativo sobre la parte de la información extraída que se guardará en la base de datos:

Mind-It	No corresponde.
HtmlDiff	No corresponde.
WebGUIDE	No corresponde.

ACADEMIA	Toda la información extraída.
WebCQ	No corresponde.
WebBeholder	No corresponde.
Clock	No corresponde.

Cuadro comparativo sobre el método para guardar la información en la base de datos, este punto se refiere a la carga inicial de la base de datos luego se trata la actualización:

Mind-It	No corresponde.
HtmlDiff	No corresponde.
WebGUIDE	No corresponde.
ACADEMIA	Sin información.
WebCQ	No corresponde.
WebBeholder	No corresponde.
Clock	No corresponde.

3.3.5. Detectar cuando cambia una página

Cuadro comparativo sobre lo que se considera como un cambio:

Mind-It	Detecta cualquier modificación en la página o su eliminación, como las páginas se registran no detecta nuevas páginas.
HtmlDiff	En la página resultado de la comparación resalta lo nuevo y lo eliminado, una modificación lo indica como algo borrado y algo nuevo.
WebGUIDE	Detecta cualquier modificación en la página o su eliminación, al monitorear un sitio puede detectar páginas nuevas en el mismo.
ACADEMIA	Detecta modificaciones en la información de interés, ya que no monitorea páginas sino que busca la información de los contactos. Puede encontrar nuevos contactos o determinar que alguno ya no existe si no lo encuentra.
WebCQ	Puede detectar cualquier modificación en una página o sólo monitorear parte de la misma como tablas, listas, párrafos, links y/o imágenes. También se puede indicar una expresión regular. Como las páginas se registran no detecta nuevas páginas pero si detecta su eliminación.
WebBeholder	Detecta cualquier modificación en la página. Como las páginas se registran no detecta nuevas páginas pero si detecta su eliminación.
Clock	Puede actualizar cualquier modificación de un documento XML que se pueda especificar a nivel del DOM.

Cuadro comparativo sobre el método utilizado para detectar que ocurrió un cambio:

Mind-It	Utiliza un checksum de la página, sin mayor información.
HtmlDiff	Directamente trata de identificar los cambios.
WebGUIDE	Utiliza la fecha de modificación de la página.
ACADEMIA	Busca la información y directamente trata de identificar los cambios.
WebCQ	Utiliza la fecha de modificación de la página.
WebBeholder	Sin información.

Clock	No se encarga de este tema.

Cuadro comparativo sobre el momento en que se observan las páginas:

Mind-It	Chequea las páginas con frecuencia fija determinada por el usuario.
HtmlDiff	Se ejecuta manualmente.
WebGUIDE	Chequea las páginas con frecuencia fija determinada por el usuario.
ACADEMIA	Se ejecuta con una frecuencia fija determinada por el usuario.
WebCQ	Chequea las páginas según la condición especificada en la consulta que realiza el monitoreo.
WebBeholder	Chequea las páginas con frecuencia fija, no menciona si se puede especificar dicho período de tiempo.
Clock	Se ejecuta manualmente.

3.3.6. Identificar los cambios de la página

Cuadro comparativo sobre el método que utilizan para la tarea de identificar los cambios:

Mind-It	Sin información.
HtmlDiff	Compara dos versiones de la página, sin información del algoritmo.
WebGUIDE	Compara dos versiones de la página, menciona la solución de Hirshberg al problema LCS (Longest Common Subsequence). Se aplica a todas las páginas que cambian en un sitio. Se guardan las páginas y se pueden ver los cambios entre dos versiones cualquiera.
ACADEMIA	Compara la información extraída con la existente en la base de datos.
WebCQ	Utiliza HtmlDiff para comparar dos versiones de la página o sólo partes de la misma.
WebBeholder	Compara dos versiones de la página, sin información del algoritmo.
Clock	No se encarga, asume que le dan el cambio.

3.3.7. Propagar los cambios a la base de datos

Cuadro comparativo el método utilizado para propagar un cambio:

Mind-It	Notifica vía e-mail cuando ocurrió un cambio.
HtmlDiff	Los cambios son presentados en una nueva página html donde resalta lo nuevo y lo eliminado. Una modificación lo indica como algo borrado y algo nuevo.
WebGUIDE	Los cambios en un sitio son presentados en un formato gráfico. Donde se indican los nuevos links, los links borrados y se marcan las páginas modificadas. Al seleccionar una página modificada se presenta la misma con los cambios resaltados.
ACADEMIA	Cuando termina el proceso de búsqueda comienza la interacción con la base de datos, cada elemento encontrado con un puntaje alto de confiabilidad, que supera el límite especificado por el usuario, es grabado en la base de datos y también se graba un log de dicha operación. En caso contrario, consulta al usuario si debe ser almacenado o no. El agente también almacena las decisiones del usuario para futuras referencias y así evitar realizar la misma pregunta.

WebCQ	Tiene un servicio de notificación que le indica al usuario cuando ocurrió un cambio. Dicha notificación es a un intervalo de tiempo definido por el usuario. Para ver los cambios hay que ingresar al sitio y se presentan en una de tres posibles formas dependiendo de lo indicado. Una puede ser una nueva página con los elementos nuevos resaltados, los elementos borrados tachados y los elementos presentes en ambas páginas con texto normal. Otra puede ser un resumen de todos los cambios fuera del contexto de la página. O puede ser que se presente cambio por cambio visualizando ambas páginas a la vez. También se puede acceder a un resumen de cambios que contiene una lista de las páginas monitoreadas con los últimos datos de evaluación de la consulta, como ser fecha, hora, si la página cambió o no.
WebBeholder	Presenta una nueva página html con los cambios resaltados.
Clock	Actualiza la base de datos donde fueron almacenados los documentos XML.

3.4. Conclusiones

Al considerar la descripción del problema realizada en la sección ***2*** y el análisis de la sección ***3.1,*** se aprecia en el relevamiento que algunos de los problemas ya están solucionados. En cambio, las soluciones o herramientas que existen para otros problemas son particulares de un tema o realidad diferentes a la de este trabajo. A continuación se comenta cada uno de los problemas.

Para encontrar las páginas con información de interés existen herramientas disponibles, como por ejemplo los buscadores. Lo cual le permite al usuario u otra aplicación elaborar una lista de páginas para ser procesadas. Por ello en este trabajo se asume que la página se recibe como un dato.

Para extraer la información de las páginas existen herramientas disponibles. En este trabajo se considera cada página independiente sin un modelo global de la web y se utilizará la herramienta W4F para extraer la información contenida en tablas de la página web.

Para la tarea de cargar la base de datos se trabajará con un modelo relacional. La carga se realizará estableciendo una correspondencia entre el modelo relacional de la base de datos y la información extraída. Por lo tanto se cargará únicamente lo que se incluya en la correspondencia definida.

Para detectar cuando cambia una página hay herramientas disponibles, algunas de ellas on-line. En este trabajo se asume que cuando una página cambia se recibe la misma para procesarla.

En este trabajo nos enfocaremos en los temas de identificar y propagar los cambios de la página a la base de datos. Hay trabajos sobre estos aspectos en temas particulares, por ejemplo para contactos académicos o documentos XML. Pero lo existente no se aplica a la realidad planteada para este trabajo porque nos basaremos en la estructura de la página, en las tablas, pero en el caso de los contactos académicos se basa en la información y ciertas palabras claves. En el caso de los documentos XML se basa en el modelo DOM.

4. Propuesta

La propuesta consiste de un mecanismo para realizar la carga de una base de datos a partir de la información contenida en tablas de páginas web y su posterior actualización. Se incluye la implementación de dicho mecanismo.

La propuesta se organiza en las siguientes secciones. En la sección ***4.1*** se presenta una descripción general. En la sección ***4.2*** se explica la estructura de las tablas en las páginas HTML, su utilización en la práctica y el método para compararlas. El mecanismo de carga y actualización se explica en las secciones ***4.3*** y ***4.4*** respectivamente. Por último, en la sección ***4.5*** se presenta una segunda solución para evitar problemas que surgen al juntar los datos de diversas páginas.

4.1. Descripción general

Según lo presentado en la sección ***2*** al dar la descripción general del problema, el objetivo es desarrollar un mecanismo de carga y actualización de una base de datos relacional a partir de la información contenida en páginas web, en particular de la información contenida en tablas. Según el relevamiento presentado en la sección ***3*** y en particular de las conclusiones presentadas en la sección ***3.4***, de los seis problemas analizados previamente nos concentraremos en la carga de la base de datos, en la identificación de los cambios y su propagación, porque para el resto de los problemas existen herramientas disponibles.

El mecanismo de carga a desarrollar consiste en establecer una correspondencia entre el esquema relacional de la base de datos, que representa el dominio de interés del usuario, y las tablas de una página web. Se asume que el esquema relacional y la página web son elementos conocidos que se toman como punto de partida.

Como el mecanismo de carga se basa en dos elementos pueden ocurrir cambios en cualquiera de ellos. Se asume que los cambios en el esquema relacional de la base de datos reflejan cambios en la realidad del usuario. En este trabajo no se trata este tema, pero al producirse uno de estos cambios se puede comenzar desde el principio y cargar una nueva base de datos vacía. Por un trabajo que aborda este tema ver [39]. El otro elemento utilizado en el mecanismo de carga es la página web, de los posibles cambios que puede tener sólo se consideran los cambios a las tablas existentes en la página, que es la información relevante en este trabajo. Por lo tanto nos enfocaremos en estos cambios y su propagación, pero no nos ocuparemos de detectar el momento en que ocurre un cambio en las páginas web.

El objetivo de cargar y actualizar una base de datos es muy genérico en cuanto a las páginas web que se pretende reflejar en la base de datos. A modo de ejemplo algunas posibilidades son reflejar la información de todas las páginas, de las páginas que cambiaron recientemente sin importar tanto las previas, algunas páginas en particular. De esto también depende como resolver los problemas que se presentan al juntar los datos, como ser valores distintos para algún atributo de un objeto o si un objeto se borra de la página web deberá ser borrado de la base de datos.

La primera meta que se trata de alcanzar es cargar y actualizar una base de datos relacional cuyo esquema es exactamente el brindado por el usuario. Esto permitiría mantener actualizada una base de

datos ya existente utilizada por otras aplicaciones. Como se muestra más adelante esta opción presenta situaciones que no se pueden resolver. Por lo tanto luego se propone cargar y actualizar una base de datos relacional con el esquema modificado agregando más información en cuanto al origen de los datos para poder así resolver los problemas presentados al juntar los datos.

En ambos casos para lograr el objetivo de desarrollar un mecanismo de carga y actualización de una base de datos relacional a partir de la información contenida en tablas de páginas web se puede dividir el problema en dos partes. Considerar primero la carga inicial de la información contenida en las tablas de una página web a la base de datos la primera vez que se considera dicha página. Y segundo considerar la propagación de los cambios ocurridos en las tablas de una página web que ya fue cargada.

Luego la información cargada en la base de datos puede ser utilizada como punto de partida para otras aplicaciones, como por ejemplo para un data warehouse u otros sistemas.

En la Figura 3 se visualiza la estructura de la propuesta dentro del ambiente general mencionado.

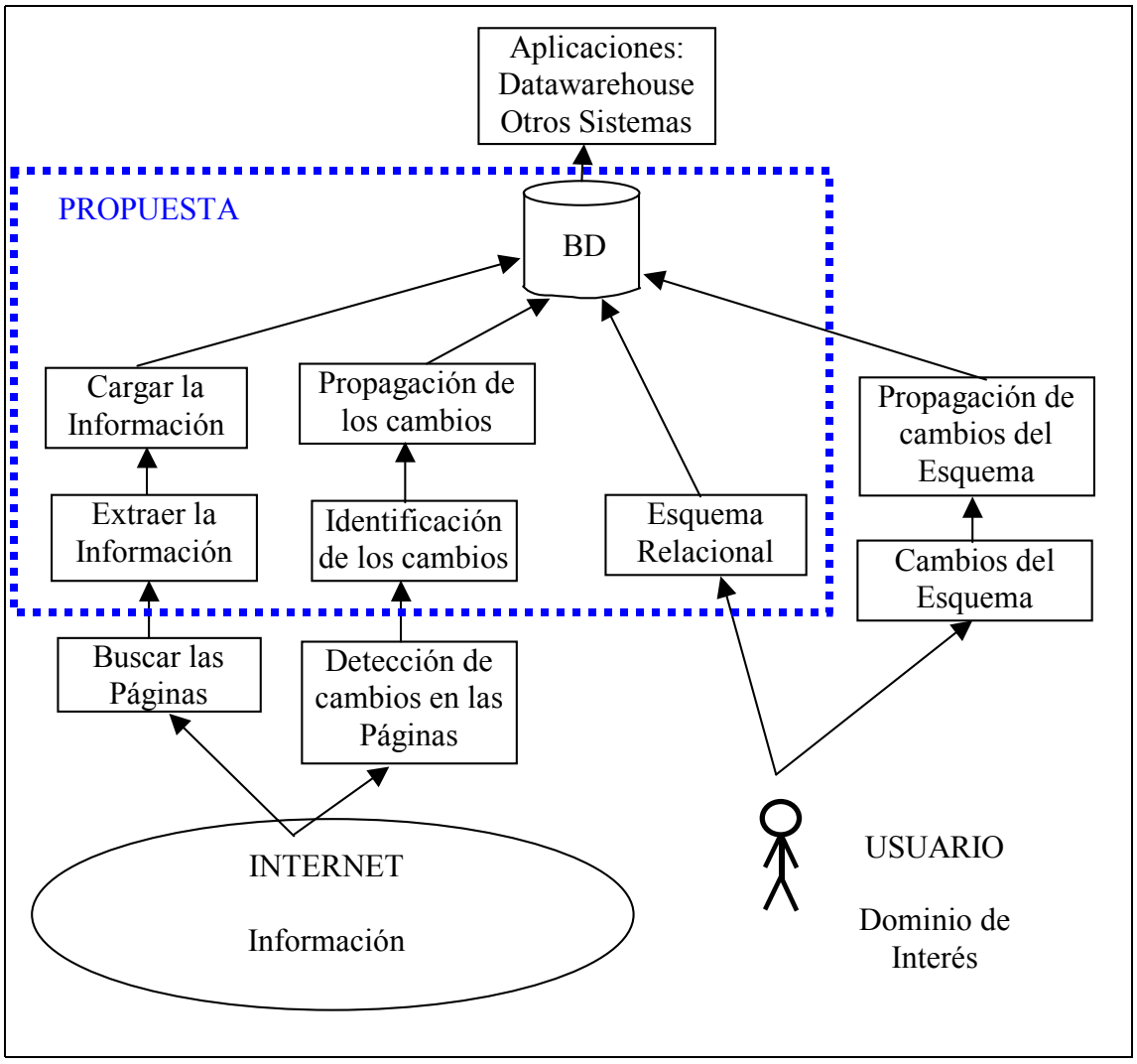


Figura 3 - Estructura de la Propuesta

4.2. Tablas de páginas web

Esta sección se organiza de la siguiente forma. En la sección ***4.2.1*** se describe brevemente la forma en que se definen las tablas en una página web. Se observaron páginas web reales para observar el uso de las tablas y el resultado se presenta en la sección ***4.2.2***. Por último en la sección ***4.2.3***, se presenta la comparación de tablas que definimos para identificar los cambios en esta propuesta.

4.2.1. Descripción de la estructura de una tabla en una página web

Las páginas web son documentos o archivos HTML (HyperText Markup Language). HTML es un lenguaje para publicar hipertexto en la web. Un documento HTML está formado por texto con marcadores que indican su estructura. Los marcadores comienzan con el carácter "<" y terminan con ">". Algunas estructuras tienen marcadores de comienzo y de fin, en dicho caso el marcador de fin es similar al de comienzo con el carácter "/" al comienzo luego del "<". Por ejemplo, "" indica que comienza negrita y "" indica que termina la negrita. Hay marcadores que no tienen un marcador de fin, por ejemplo <hr> indica una línea en el documento.

Un documento HTML simple tiene la siguiente estructura:

```
<html>
    <head>
        Encabezado
    </head>
    <body>
        Contenido
    </body>
</html>
```

En el encabezado por ejemplo hay estructuras que indican el título de la página web, información para los buscadores o información sobre la herramienta con la cual se creó la página web. En el contenido se encuentran por ejemplo marcadores para títulos, párrafos, listas, tablas, imágenes, líneas. Además marcadores para variar el estilo de la fuente, color, tamaño, entre otros.

Los marcadores que describen las tablas HTML son: <table>, <caption>, <tr>, <th> y <td> y sus correspondientes marcadores de cierre.

El marcador **<table>** define la tabla y especifica las características generales como el fondo, el borde y el espacio entre las celdas.

El marcador **<caption>** dentro del marcador <table> es opcional y especifica un título o nombre de la tabla.

El marcador **<tr>** indica cada fila de la tabla y contiene los marcadores para cada celda dentro de una fila como <th> y <td>. El marcador **<th>** indica una celda que corresponde al título de una

columna. El marcador **<td>** indica una celda común. En ellos además se pueden indicar sus propiedades como por ejemplo tamaño, alineación y fondo.

Un ejemplo de una tabla con todos los marcadores mencionados es:

```
<table border="1">
	<caption>Tabla ejemplo</caption>
	<tr>
		<th>Columna 1</th>
		<th>Columna 2</th>
		<th>Columna 3</th>
	</tr>
	<tr>
		<td>A</td>
		<td>B</td>
		<td>C</td>
	</tr>
	<tr>
		<td>D</td>
		<td>E</td>
		<td>F</td>
	</tr>
</table>
```

Este ejemplo se visualiza de la siguiente forma:

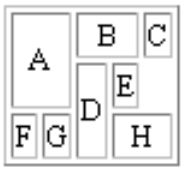
Tabla ejemplo

Columna 1	Columna 2	Columna 3
A	B	C
D	E	F

En el ejemplo previo cada fila tiene exactamente la misma cantidad de columnas, pero puede que no sea así. Hay celdas que pueden ocupar más de una fila o más de una columna. A continuación se presenta un ejemplo de una tabla con celdas de distintos tamaños, incluso espacios que no son ocupados por ninguna celda.

```
<table border="1">
	<tr>
		<td rowspan="2" colspan="2" align="center" valign="center">A</td>
		<td colspan="2" align="center" valign="center">B</td>
		<td align="center" valign="center">C</td>
	</tr>
	<tr>
		<td rowspan="2" align="center" valign="center">D</td>
		<td align="center" valign="center">E</td>
	</tr>
	<tr>
		<td align="center" valign="center">F</td>
		<td align="center" valign="center">G</td>
		<td colspan="2" align="center" valign="center">H</td>
	</tr>
</table>
```

Este ejemplo se visualiza de la siguiente forma:



El contenido de cada celda puede ser texto o cualquier otra estructura o elemento. Por eso, puede que una celda contenga a su vez otra tabla.

4.2.2. Análisis de páginas web existentes con tablas

Se observaron páginas web existentes con el fin de realizar un breve estudio sobre su utilización en Internet.

Se observó que muchas páginas web utilizan tablas para distribuir o alinear los elementos que contienen. Por este motivo no todas las tablas contienen información. Hay páginas que contienen tablas dentro de tablas, en dichos casos decimos que hay tablas anidadas. Cuando existen tablas anidadas hay que determinar cuáles tablas son utilizadas con el fin de alineación y cuáles contienen información que puede ser de interés para el usuario. En general se observó que aquellas tablas que no contienen dentro otra tabla contienen información en forma de texto o imágenes.

En el caso de las páginas con tablas anidadas sólo consideraremos aquellas que no tienen otra tabla dentro, es decir las más interiores en el anidamiento. Esto es con un fin práctico para poder extraer fácilmente la información de las tablas de las páginas que se consideran. La propuesta es independiente de la cantidad de tablas, por lo tanto para considerar tablas anidadas habría que definir el criterio con que se extrae la información. Las posibilidades de esto último pueden ser a manera de ejemplo: (1)

considerar cada tabla como una nueva como si estuvieran una a continuación de la otra, (2) combinar de alguna forma la información de las tablas anidadas para que sea un elemento de la tabla original.

Se observaron los marcadores utilizados en las tablas de las páginas web. De los marcadores descriptos en la sección ***4.2.1*** en general no se utilizan <caption> y <th>. Por lo tanto no nos podemos basar exclusivamente en estos marcadores.

4.2.3. Comparación de tablas de páginas web

Para identificar los cambios y así mantener actualizada la información que cargamos comparamos las tablas correspondientes a dos versiones de la misma página web. El problema es determinar si dos tablas son iguales. Para ello se define un mecanismo de comparación basado en la información existente. Como se mencionó anteriormente, dicho mecanismo no se puede basar exclusivamente en los marcadores <caption> o <th>, ya que puede que no existan.

Para determinar si dos tablas se corresponden nos basaremos en sus columnas. Si las columnas tienen título (marcador <th>), consideramos que las columnas se corresponden si el título es igual. Si no tienen título la comparación se realiza por su contenido. En este último caso, se asume una cierta cantidad mínima de elementos que deben ser iguales. En otras palabras, si la cantidad de elementos iguales de una columna supera el mínimo definido las columnas se corresponden. La "*cantidad mínima de elementos iguales"* es un parámetro que define el usuario. Como no es lo mismo considerar 10 elementos iguales en una columna cuyo total de elementos es 1000, que en un total de 15, dicha cantidad mínima se expresa en forma de porcentaje. También se define como parámetro la "*cantidad mínima de columnas iguales"*, si al comparar dos tablas la cantidad de columnas que se corresponden supera la cantidad mínima de columnas iguales se considera que las tablas se corresponden.

4.3. Mecanismo de carga

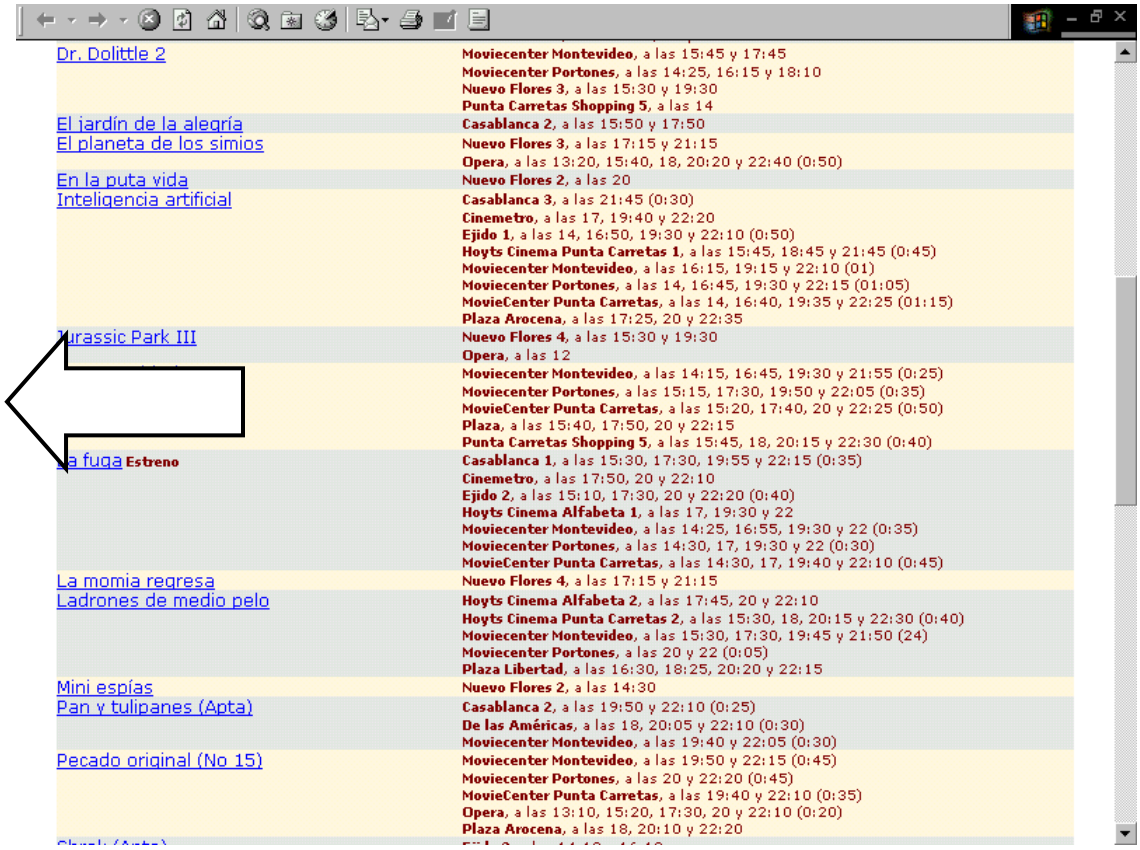
Dado el esquema de una base de datos relacional y una página web. Se pretende cargar la información de interés para el usuario contenida en las tablas de dicha página a una base de datos con el esquema dado. Para ello la primera vez que se procesa cada página el usuario define la correspondencia entre ambos elementos.

Como ejemplo se consideró cargar una cartelera de cines, donde se encuentran los cines, las películas y sus proyecciones, pero la propuesta es independiente del dominio elegido. La idea es cargar las tablas que representan la cartelera con la información existente en la página web, lo cual se ilustra en la Figura 4.

```
CREATE TABLE
CINES (
  NOMBRE
VARCHAR(50),
  DIRECCION
VARCHAR(50),
  TELEFONOS
VARCHAR(50),
CONSTRAINT
PK_CINES PRIMARY
KEY (NOMBRE) );

CREATE TABLE
PELICULAS (
  NOMBRE
VARCHAR(50),
  GENERO
VARCHAR(20),
  ACTORES
VARCHAR(100),
  COMENTARIO
VARCHAR(250),
CONSTRAINT
PK_PELICULAS
PRIMARY KEY
(NOMBRE) );

CREATE TABLE
PROYECCION (
  CINE
VARCHAR(50),
  PELICULA
VARCHAR(50),
  HORARIOS
VARCHAR(50)
CONSTRAINT
PK_PROYECCION
PRIMARY KEY(CINE,
PELICULA) );
```



Dr. Dolittle 2	**Moviecenter Montevideo**, a las 15:45 y 17:45 **Moviecenter Portones**, a las 14:25, 16:15 y 18:10 **Nuevo Flores 3**, a las 15:30 y 19:30 **Punta Carretas Shopping 5**, a las 14
El jardín de la alegría	**Casablanca 2**, a las 15:50 y 17:50
El planeta de los simios	**Nuevo Flores 3**, a las 17:15 y 21:15 **Opera**, a las 13:20, 15:40, 18, 20:20 y 22:40 (0:50)
En la puta vida	**Nuevo Flores 2**, a las 20
Inteligencia artificial	**Casablanca 3**, a las 21:45 (0:30) **Cinemetro**, a las 17, 19:40 y 22:20 **Ejido 1**, a las 14, 16:50, 19:30 y 22:10 (0:50) **Hoyts Cinema Punta Carretas 1**, a las 15:45, 18:45 y 21:45 (0:45) **Moviecenter Montevideo**, a las 16:15, 19:15 y 22:10 (01) **Moviecenter Portones**, a las 14, 16:45, 19:30 y 22:15 (01:05) **MovieCenter Punta Carretas**, a las 14, 16:40, 19:35 y 22:25 (01:15) **Plaza Arocena**, a las 17:25, 20 y 22:35
Jurassic Park III	**Nuevo Flores 4**, a las 15:30 y 19:30 **Opera**, a las 12
	Moviecenter Montevideo, a las 14:15, 16:45, 19:30 y 21:55 (0:25) **Moviecenter Portones**, a las 15:15, 17:30, 19:50 y 22:05 (0:35) **MovieCenter Punta Carretas**, a las 15:20, 17:40, 20 y 22:25 (0:50) **Plaza**, a las 15:40, 17:50, 20 y 22:15 **Punta Carretas Shopping 5**, a las 15:45, 18, 20:15 y 22:30 (0:40)
La fuga **Estreno**	**Casablanca 1**, a las 15:30, 17:30, 19:55 y 22:15 (0:35) **Cinemetro**, a las 17:50, 20 y 22:10 **Ejido 2**, a las 15:10, 17:30, 20 y 22:20 (0:40) **Hoyts Cinema Alfabeta 1**, a las 17, 19:30 y 22 **Moviecenter Montevideo**, a las 14:25, 16:55, 19:30 y 22 (0:35) **Moviecenter Portones**, a las 14:30, 17, 19:30 y 22 (0:30) **MovieCenter Punta Carretas**, a las 14:30, 17, 19:40 y 22:10 (0:45)
La momia regresa	**Nuevo Flores 4**, a las 17:15 y 21:15
Ladrones de medio pelo	**Hoyts Cinema Alfabeta 2**, a las 17:45, 20 y 22:10 **Hoyts Cinema Punta Carretas 2**, a las 15:30, 18, 20:15 y 22:30 (0:40) **Moviecenter Montevideo**, a las 15:30, 17:30, 19:45 y 21:50 (24) **Moviecenter Portones**, a las 20 y 22 (0:05) **Plaza Libertad**, a las 16:30, 18:25, 20:20 y 22:15
Mini espías	**Nuevo Flores 2**, a las 14:30
Pan y tulipanes (Apta)	**Casablanca 2**, a las 19:50 y 22:10 (0:25) **De las Américas**, a las 18, 20:05 y 22:10 (0:30) **Moviecenter Montevideo**, a las 19:40 y 22:05 (0:30)
Pecado original (No 15)	**Moviecenter Montevideo**, a las 19:50 y 22:15 (0:45) **Moviecenter Portones**, a las 20 y 22:20 (0:45) **MovieCenter Punta Carretas**, a las 19:40 y 22:10 (0:35) **Opera**, a las 13:10, 15:20, 17:30, 20 y 22:10 (0:20) **Plaza Arocena**, a las 18, 20:10 y 22:20

Figura 4 - Una estructura relacional de una cartelera de cine

A continuación se explican las tareas que debe realizar el proceso de carga.

1. Recibir y leer los elementos de los cuales parte que son: el esquema relacional de la base de datos y la página web. El esquema relacional está dado por las sentencias SQL de creación de las tablas. Y la página web está identificada por la URL de la misma o el nombre del archivo.
2. Extraer las tablas de la página web.
3. Presentar al usuario una interfase que le permita establecer la correspondencia entre el esquema relacional y las tablas de la página web. Teniendo en cuenta que puede que no existan en dicha página los datos necesarios para cargar toda la información representada por el esquema relacional. Además no es necesario que toda la información extraída de la página se cargue, tan sólo se cargará la que el usuario indique.
   En el ejemplo anterior se observa que una columna del esquema relacional, en particular los horarios de la proyección de las películas, se corresponde con una parte de una columna de la tabla en la página web, la segunda columna. Este es otro aspecto a tener en cuenta.
4. Luego de que se definió la correspondencia hay que cargar la base de datos y guardar la correspondencia para ser utilizada en el proceso de propagación de los cambios.

La base de datos puede ser cargada con información extraída de más de una página web. Por lo tanto cuando se carga la información de una página web puede que la base de datos ya contenga datos. Hay que tener esto presente para no duplicar la información y si un mismo objeto tiene valores diferentes para algún atributo en la base de datos quedará grabado el del último que se procesó.

4.3.1. Descripción del proceso de carga propuesto

El proceso de carga recibe como entrada el esquema relacional de la base de datos y la página web que hay que cargar.

El esquema relacional está dado por las sentencias de creación de las tablas en lenguaje SQL. Las sentencias de creación de tablas que consideramos tienen la siguiente forma:

```
CREATE TABLE <NOMBRE TABLA>
(<NOMBRE COLUMNA> TIPO(TAMAÑO),
<NOMBRE COLUMNA> TIPO(TAMAÑO),
 ...
CONSTRAINT <NOMBRE DE LA CLAVE>
PRIMARY KEY (NOMBRE DE COLUMNAS));
```

Consideramos una versión simplificada de la sentencia de creación de una tabla en el lenguaje SQL. Donde tipo puede ser "varchar" o "number" y el tamaño es un número. Además consideramos únicamente la definición de la clave primaria ("primary key") la cual indica las columnas que identifican un registro de la tabla relacional. Si no se define una clave se considera que la identificación del registro es por todas sus columnas.

La página web se indica por su URL o por el nombre del archivo si se encuentra en forma local.

La correspondencia que el usuario define se basa en las columnas. Para cada columna del esquema relacional se puede indicar de que columna de las tablas de la página web se extrae la información. Pueden existir columnas o tablas del esquema relacional para las cuales no se indique una correspondencia. Además, pueden existir columnas o tablas de la página web que no se indican como parte de la correspondencia, por lo tanto su información no se carga en la base de datos.

La correspondencia para cada columna del esquema relacional se identifica de la siguiente manera:

- La tabla dentro de la página web, es un número secuencial que indica primera, segunda y así sucesivamente.
- La columna dentro de la tabla indicada anteriormente, es un número secuencial con el mismo criterio.
- Un string que le indica dentro de la columna desde donde considerar, esto es opcional.
- Un string que le indica dentro de la columna hasta donde considerar, esto es opcional.
- Si la información a cargar en una tabla relacional debe ser extraída de más de una tabla en la página web hay que indicar como juntar la información de las tablas de la página web para formar una tabla del esquema relacional. Para ello se puede indicar un número de tabla, un número de columna y en forma opcional los strings desde donde y hasta donde considerar. Las columnas indicadas deben ser de diferentes tablas y así se unen las filas de ambas tablas tomando aquellas cuyo valor para las columnas indicadas es igual.
  Si no se indica esta segunda columna cada fila de una tabla se juntará con cada una de las filas de la otra tabla.
- Se puede indicar un valor por defecto en caso de que la celda en la tabla se encuentre vacía, esto es opcional.

Tanto el string que indica dentro de la columna desde donde considerar como el que indica hasta donde considerar no se tratan como parte de la información porque se asume que son un separador. Si se repite se tomó el criterio de considerar el primero.

Para que resulte más claro se hace una analogía con los conceptos del modelo relacionales. La forma de juntar las filas de las tablas de la página web si se indican dos columnas es con un join de las tablas por dichas columnas, sino se realiza un producto cartesiano de ambas tablas. En la implementación de la propuesta se realiza un join pero se podría ampliar y aplicar el concepto de "outer join" tomando un criterio de la forma en la cual tratar las filas que existan en una tabla y no tengan una fila en la otra tabla con el valor en la columna indicada igual.

En el ejemplo de la Figura 4 de la cartelera de cines, en la segunda columna de la tabla en la página web se encuentra la información del cine y el horario de proyección, en este caso están separados por una coma. Por lo tanto se puede indicar que el cine se extrae de la segunda columna hasta la coma y el horario se extrae de la segunda columna desde la coma. El string indicado en las opciones *desde* y *hasta* no se considera como parte de la información para la carga porque se asume que es un separador.

En cuanto al valor por defecto se puede utilizar la palabra "ANTERIOR" para indicar que si un valor se omite en la página web, el mismo es igual al de la fila precedente.

Se implementó una interfase gráfica para que el usuario pueda definir la correspondencia la cual se muestra en la Figura 5.

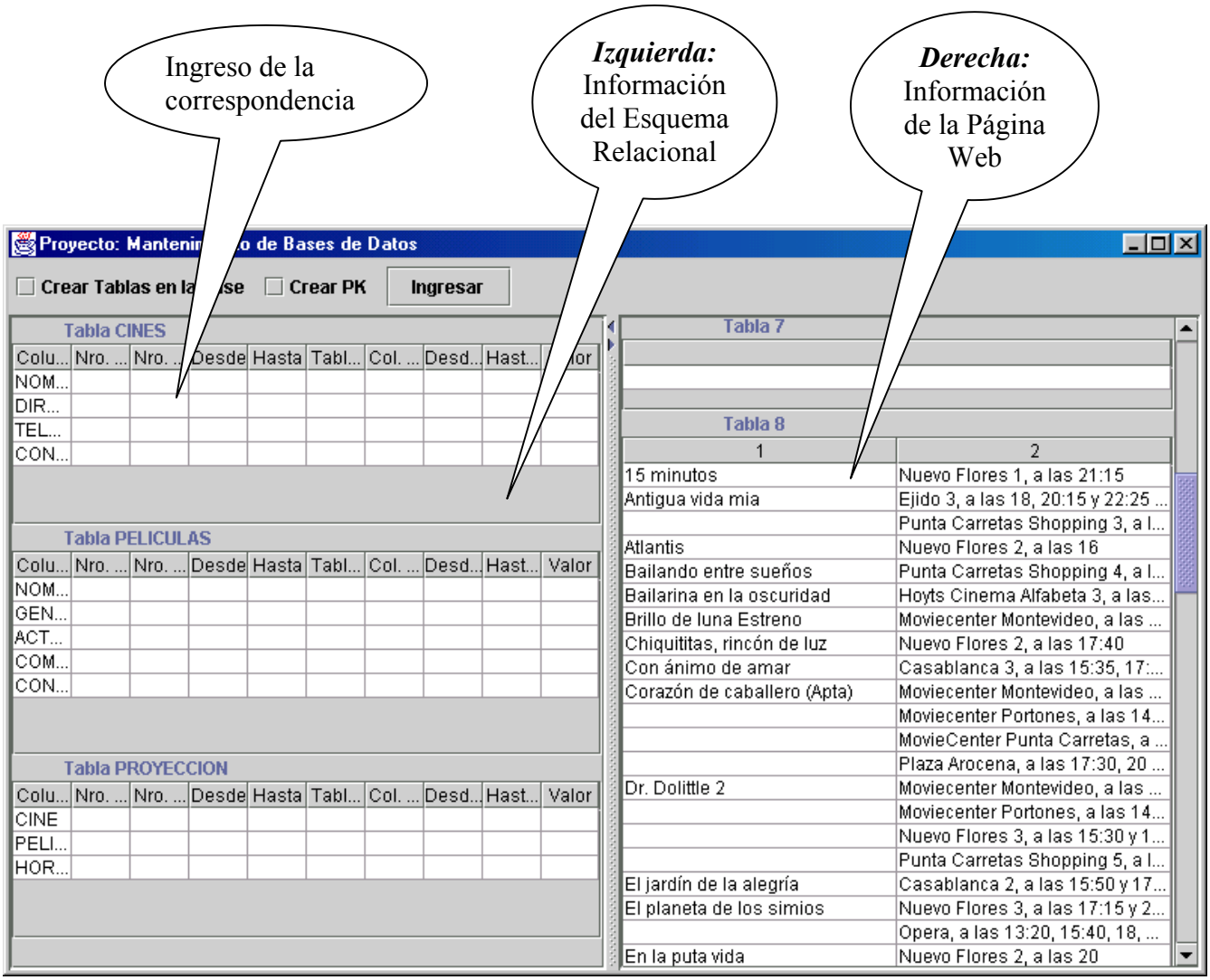


Figura 5 - Interfase del proceso de carga.

En la parte izquierda de la pantalla de la interfase se coloca la información del esquema relacional que representa el dominio de interés del usuario. Presenta una cuadrícula por cada tabla del esquema relacional. Cada fila de la cuadrícula en la interfase representa una columna de la tabla del esquema relacional permitiendo ingresar la correspondencia explicada previamente. En la parte derecha se colocan las tablas extraídas de la página web, donde se numera cada una y dentro de cada tabla se numeran las columnas, facilitando así la tarea del usuario.

En el prototipo implementado el proceso para cargar la información trabaja por medio de una conexión ODBC (Open DataBase Connectivity) en la base de datos. Además, permite indicar si es necesario crear las tablas, lo cual se realiza marcando la opción “Crear Tablas en la Base”. Si la base de datos lo permite se puede indicar crear las claves (Primary Key) de las tablas marcando la opción “crear PK”.

En la Figura 6 se muestra en forma ampliada la cuadrícula de la interfase donde se debe ingresar la correspondencia para la tabla del esquema relacional del ejemplo correspondiente a CINES para apreciar las columnas.

Tabla CINES

Columna	NroTabla	NroColu...	Desde	Hasta	Tabla Join	Col. Join	Desde Join	Hasta Join	Valor
NOMBRE									
DIRECCI...									
TELEFO...									

Figura 6 - Ampliación de la interfase.

Una vez que el usuario ingresa la correspondencia debe presionar el botón "Ingresar" y se realizan las siguientes tareas:

- Se carga la información en la base de datos usando una conexión Odbc.
- Se genera un archivo con las sentencias SQL ejecutadas, un archivo de log o registro de lo realizado. Incluye las sentencias INSERT, UPDATE y DELETE pero no las sentencias SELECT. El objetivo de este archivo de log es que pueda ser utilizado en otro proceso, por ejemplo para actualizar un data warehouse, por ello se registran las operaciones que modificaron la base de datos pero no las consultas realizadas.
- Se guarda la representación de la página web para su utilización en el proceso de actualización. No se guarda toda la página sólo la información extraída contenida en las tablas.
- Se guarda la correspondencia definida por el usuario para su utilización en el proceso de actualización.
- Se registra la página y su fecha de procesamiento. Información necesaria si se opta por la alternativa recarga en una etapa posterior.

El proceso de la carga se muestra en forma esquemática en la Figura 7.

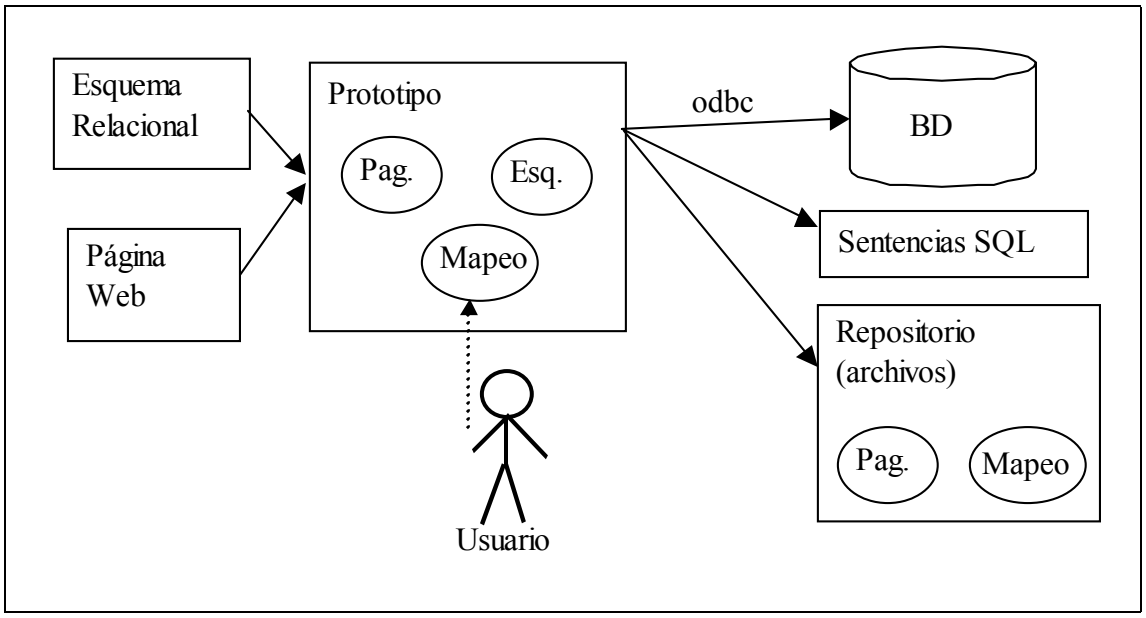


Figura 7 - Proceso de carga.

Se observa que el prototipo recibe el esquema relacional de la base de datos y la página web. En círculos se representan los objetos con los que trabaja el prototipo que corresponden a cada elemento.

Luego el usuario ingresa la correspondencia (llamada mapeo). El prototipo carga la base de datos utilizando una conexión odbc. Luego de cargar la base de datos genera un archivo con todas las sentencias SQL ejecutadas. Dicho archivo puede ser considerado como un archivo de registro histórico (o log) que podrá ser utilizado por alguna otra aplicación, por ejemplo actualizar un data warehouse. También se guardan las representaciones de la página y la correspondencia definida para ser utilizadas luego en el proceso de actualización. En este prototipo la representación de la página y la correspondencia se guardan como archivos pero podrían guardarse por ejemplo en la base de datos en estructuras con dicho fin.

4.3.2. Algoritmo de carga

Para realizar la carga inicial de la base de datos por cada tabla del esquema relacional se crea en memoria la tabla con los datos según la correspondencia definida.

Por cada fila de dicha tabla
Se ejecuta un SELECT para verificar si existe la fila en la base de datos
Si existe ⇒ se verifica si es igual o diferente
Si es igual ⇒ no se hace nada
Si es diferente ⇒ se ejecuta una sentencia UPDATE
se graba el log de la sentencia UPDATE
Si no existe ⇒ se ejecuta una sentencia INSERT.
se graba el log de la sentencia INSERT

4.4. Mecanismo de actualización

El mecanismo de comparación entre tablas de páginas web, presentado en la sección ***4.2.3***, permite identificar cambios en la estructura de la página al nivel de tablas, como ser una tabla nueva, la eliminación de alguna o un cambio de orden de las tablas. Permite identificar cambios al nivel de columnas, como ser una columna nueva dentro de una tabla, la eliminación de una existente o el cambio de orden de las mismas. La propagación de estos cambios se realiza en forma semi-automática debido a que pueden afectar la correspondencia que definió el usuario.

La idea es que si se borró una tabla o una columna que existía en la correspondencia se quita de la correspondencia y se indica al usuario la identificación de la tabla o columna que fue borrada. Si se cambió de lugar una tabla o columna que existía en la correspondencia se realiza el ajuste y se indica al usuario la identificación anterior y la nueva. Si existen elementos nuevos, ya sea una tabla o una columna, se le indica al usuario su identificación para que rápidamente pueda observar lo que el mecanismo propuesto encontró. Por ello el mecanismo es semi-automático. En cualquiera de los casos que se deben indicar cambios al usuario se presenta una interfase para que pueda modificar la correspondencia con las indicaciones mencionadas. Luego se propagan los cambios.

Otro tipo de cambio que puede ocurrir es en el contenido de las tablas, como ser si se agregó, modificó o eliminó una fila de la tabla. Estos cambios no afectan la correspondencia definida entre el esquema relacional de la base de datos y las tablas de las páginas web. La tarea consiste entonces en detectar y propagar estos cambios a la base de datos en forma automática.

El mecanismo de actualización debe propagar la información de las tablas de la página web considerando la correspondencia definida. Esta actualización se realiza en dos pasos: (1) se borra de la base de datos la información que fue borrada de la página web y (2) la información que existe en la página web se trata de forma similar que en el proceso de carga inicial.

La base de datos puede ser cargada con información extraída de más de una página web. Por lo tanto cuando se actualiza la base de datos hay que tener presente que aunque cierta información fue borrada de una página puede ser que aun exista en otra. Este es un problema que no se puede resolver a menos que se guarde información en cuanto al origen de los datos cargados en la base de datos y por ello se presenta luego una segunda propuesta. Otro ejemplo del tipo de problemas que se enfrenta es que si un mismo objeto tiene valores diferentes para algún atributo en la base de datos quedará grabado el del último que se procesó.

Se consideran dos alternativas para la propagación de los cambios. Una alternativa incremental donde se realizan las operaciones necesarias para propagar únicamente los cambios sin afectar el resto de la base de datos. Y por otro lado una alternativa de recarga, es decir, vaciar la base de datos y cargarla nuevamente. Si la base de datos fue cargada a partir de varias páginas implica cargar nuevamente todas las páginas involucradas utilizando la correspondencia definida durante la carga de cada página.

4.4.1. Descripción del proceso de actualización de la base de datos

El proceso de actualización recibe como entrada el esquema relacional de la base de datos en el formato mencionado en la sección ***4.3*** y la página web que cambió. Consideramos dos posibles alternativas. Una posibilidad es vaciar la base de datos y cargar toda la información nuevamente, a esta alternativa le llamamos recarga de la base de datos. Si la base se cargó con información de más de una página implica volver a cargar cada una de las páginas utilizadas. Una segunda alternativa es la propagación en forma incremental de los cambios. Identificar exactamente que se borró, que se modificó, que se agregó en la página y realizar las operaciones correspondientes en la base de datos.

Recarga de la base de datos

Se lleva un registro de las páginas que se han cargado. Por lo tanto esta alternativa consiste en primero vaciar la base de datos. Luego cargar el resto de las páginas con la opción recarga en el orden que fueron cambiando desde la más antigua hasta la que cambió más recientemente. Y por último procesar la página que cambió como si fuera la primera vez que se carga. En definitiva esto es así porque comenzamos con la base de datos vacía.

La opción recarga implica que se utiliza para la carga la correspondencia definida y guardada previamente en lugar de consultar al usuario. Lo cual es válido porque las páginas no han cambiado, ya que si una de las páginas hubiera cambiado es la que se procesa en primer lugar. Si varias páginas cambian se repite el mecanismo una por una.

Propagación incremental de los cambios en la base de datos

Dada una página web que cambió, debemos identificar los cambios en la estructura de las tablas y en la información que contienen. Para identificar los cambios en la estructura de las tablas tenemos que comparar dos versiones de la página web porque en la base de datos no guardamos datos a tales efectos.

Para identificar los cambios en la información estudiamos dos posibilidades: (1) comparar la información de las tablas de la página con la existente en la versión anterior, (2) contra la información cargada en la base de datos. Si en la base de datos cargamos información contenida en una única página el resultado en ambas alternativas es similar, pero si se pretende que la base de datos refleje el contenido de varias páginas existen diferencias si en las mismas hay información en común.

Identificar los cambios comparando la información contenida en las tablas de la página solamente con la existente en la versión anterior puede causar que el contenido de la base de datos no refleje la información contenida en un conjunto de páginas web, lo cual se ilustra en el ejemplo de la Figura 8.

Supongamos que tenemos dos páginas (p_1) y (p_2) ambas con un cierto valor (v), esto es al comienzo digamos en un tiempo (t_1), por lo tanto al procesar las páginas se carga en la base de datos el valor (v).
Luego, digamos en un tiempo (t_2), cambia una de las páginas (p_1) donde se borra el valor (v) entonces dicho valor se borra de la base de datos.
Aquí la base de datos ya no refleja el contenido de todas las páginas web cargadas y por ello luego se presenta una segunda propuesta. Pero si el objetivo es cargar la información reflejando el contenido de la última página que cambio permitiendo en cierto grado esta falta de información alcanzaría con cargar el contenido de la página comparando con la existente en la base de datos. Después de todo si la información se quitó de una página posiblemente ya no sea de interés.
Por último, supongamos que luego, en un tiempo (t_3), se modifica la página (p_2) que aun contiene el valor (v) borrado de la otra página, la modificación puede ser por ejemplo agregando el valor (x). La idea es que el valor (v) sea reflejado en la base de datos. Si comparamos la página con la versión anterior de la página (p_2) el valor (v) no se considera como un cambio y por lo tanto no se carga, pero si comparamos con la base de datos se cargaría porque en la misma no existe.

La Figura 8 del ejemplo es el contenido de la base de datos si comparamos la página con la información de la versión anterior. Observamos que en el tiempo (t_3) el contenido de la base de datos no se corresponde con la información existente en las páginas (p_1) y (p_2) a partir de las cuales se cargó.

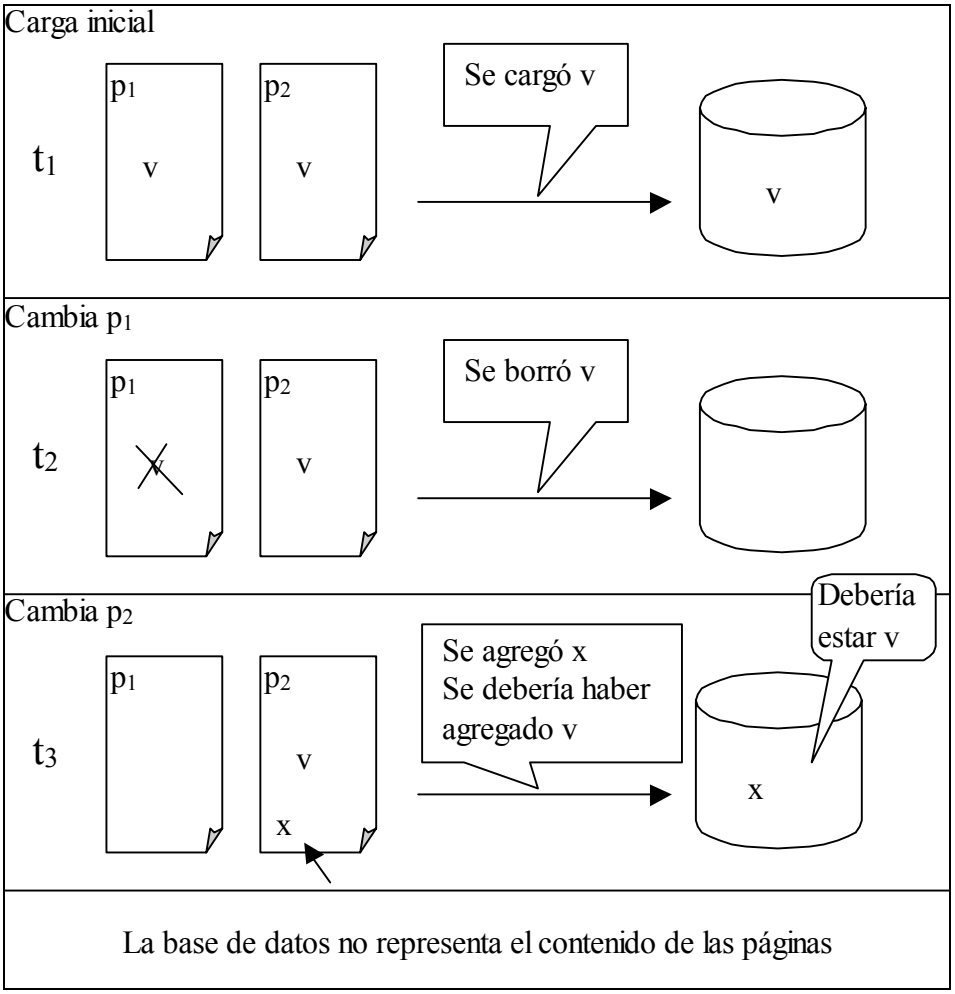


Figura 8 - Ejemplo: detección de cambios comparando con la versión anterior de la página.

Identificar los cambios comparando la información contenida en las tablas de la página contra la existente en la base de datos no permite identificar la información que se borró. Nosotros asumimos que la base de datos se carga con información a partir de varias páginas web. Por lo tanto la información que existe en la base de datos y no existe en la página web puede ser que se haya borrado de la misma o fue cargada a partir de otra página web.

Proponemos comparar la representación de la página web con la versión anterior de la misma. Con esto se logra identificar los cambios en la estructura de las tablas e identificar la información que se borró de dicha página web. Además, proponemos comparar la información actual existente en la página web con la existente en la base de datos. Por lo tanto para identificar los cambios en la información utilizaremos una combinación de las dos posibilidades mencionadas previamente. Con esto se logra que la base de datos refleje el contenido de las páginas web consideradas con cierto grado inexactitud.

En el ejemplo previo en el tiempo (t_1) y (t_3) la base reflejaría correctamente el contenido de las páginas web. Pero en el tiempo (t_2) no estaría reflejando el contenido de la página (p_2) pero si el de la página (p_1) que fue la que cambió recientemente. Si la frecuencia de cambio de las páginas consideradas es alta puede que esta propuesta alcance para los fines deseados. De lo contrario más adelante se incluye otra propuesta.

El proceso de actualización recibe la misma entrada que el proceso de carga, es decir el esquema relacional de la base de datos y la página web para la cual se deben propagar los cambios. Lee la entrada y luego recupera lo almacenado en el proceso de carga. Esto es la representación anterior de la página web y la correspondencia establecida por el usuario. Compara las dos versiones de la representación de la página web para identificar la lista de los cambios en la estructura, ya sea al nivel de las tablas o de las columnas de las mismas. Si se identifican cambios de este tipo se presenta una interfase gráfica al usuario para que éste tome las decisiones correspondientes para la correcta actualización de la base de datos, es decir que ajuste la correspondencia definida previamente.

A modo de ejemplo si se había establecido la correspondencia con la tabla número 7 y se agrega una tabla antes en el lugar número 2 se presenta la pantalla de la Figura 9.

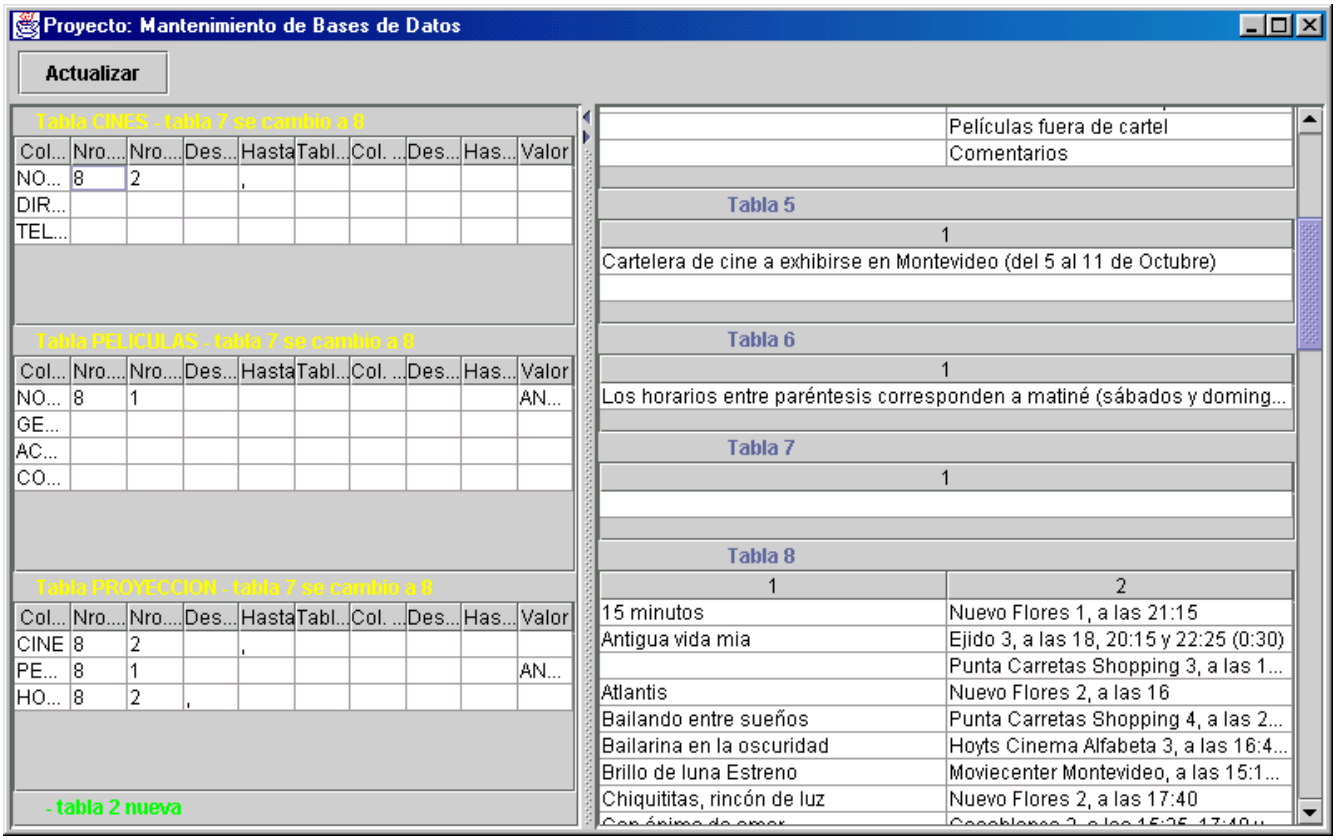


Figura 9 - Interfase del proceso de actualización.

En la interfase que se presenta al usuario se ajustó la correspondencia pero se le indica el cambio "tabla 7 se cambio a 8" en la Figura 9 se observa en amarillo. Además, se indica que hay una tabla nueva "tabla 2 nueva" en la Figura 9 se observa en verde, por si la información de la misma es de su interés. En esta pantalla puede realizar los ajustes que considere necesarios y luego debe presionar el botón "Actualizar".

Luego de que se actualice la correspondencia o si esto no es necesario, se realizan las siguientes tareas:

- Se compara la información existente en las tablas de la página web con la existente en la versión anterior, identifica aquella información cargada que actualmente no existe y la borra en la base de datos.

- Se carga la información existente en las tablas de la página web en la base de datos, agregándola o modificando la existente según corresponda.
- Se genera un archivo con las sentencias SQL ejecutadas, un archivo de log o registro de lo realizado. Incluye las sentencias INSERT, UPDATE y DELETE pero no las sentencias SELECT.
- Se guarda la representación de la página web para su utilización en un proceso de actualización posterior.
- Se guarda la correspondencia ajustada por el usuario para su utilización en un proceso de actualización posterior.
- Se registra la página y su fecha de procesamiento. Información necesaria si se opta por la alternativa recarga en una etapa posterior.

El proceso de actualización se muestra en forma esquemática en la Figura 10.

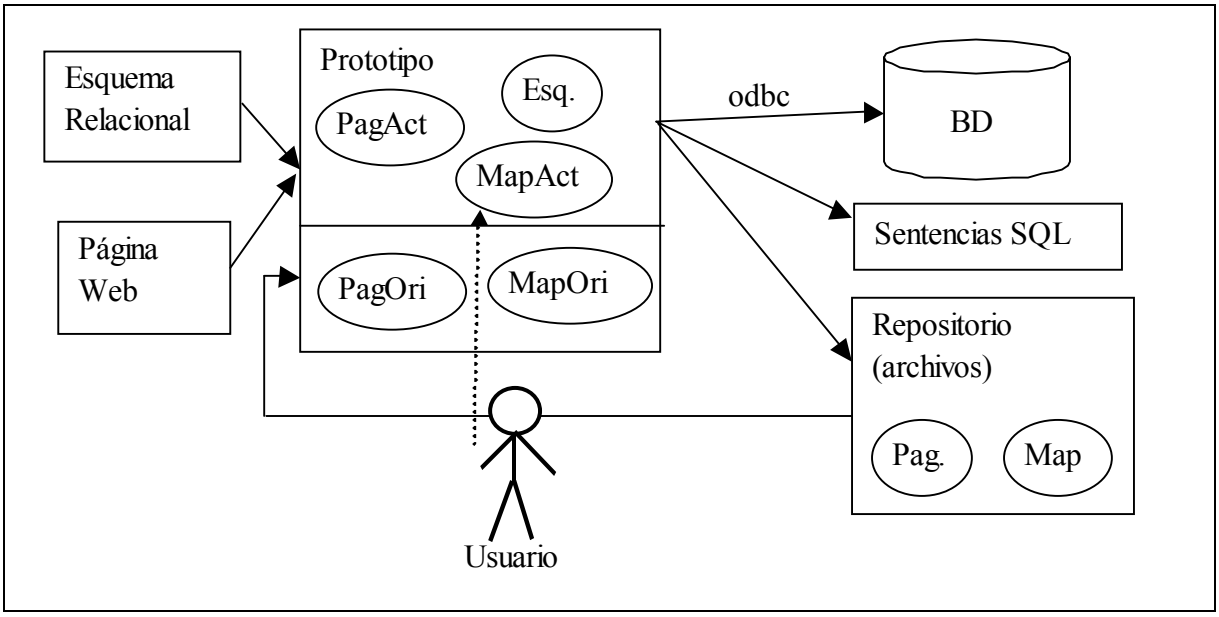


Figura 10 - Proceso de actualización.

Se observa que el prototipo recibe el esquema relacional de la base de datos y la página web. En círculos se representan los objetos con los que trabaja el prototipo que corresponden a cada elemento. Lee la versión anterior de la página web y la correspondencia o mapeo guardados previamente (se indican como "ori" por original y "act" por actual). Analiza si hay diferencias entre la página web actual y la anterior que necesiten la intervención del usuario para ajustar la correspondencia. El prototipo actualiza la base de datos utilizando una conexión odbc. Luego de actualizar la base de datos genera un archivo con todas las sentencias SQL ejecutadas. También se guardan las representaciones de la página y la nueva correspondencia para ser utilizadas luego en otro proceso de actualización.

4.4.2. Algoritmos de actualización

Para realizar la actualización de la base de datos por cada tabla del esquema relacional se crean en memoria dos tablas con los datos correspondientes. Una para la versión anterior de la página web con la correspondencia original (tabla1) y otra para la nueva versión de la página web con la correspondencia ajustada (tabla2).

Se comparan ambas tablas y se obtiene el conjunto de filas que se encuentran en la tabla1 pero no se encuentran en la tabla2, es decir el conjunto {tabla1 - tabla2}.

Por cada fila en el conjunto {tabla1 - tabla2}
Se ejecuta un SELECT para verificar si existe la fila en la base de datos
Si existe ⇒ se ejecuta una sentencia DELETE
se graba el log de la sentencia DELETE
Si no existe ⇒ no se hace nada

Luego se procesa la tabla con la información actual (tabla2).

Por cada fila de la tabla2
Se ejecuta un SELECT para verificar si existe la fila en la base de datos
Si existe ⇒ se verifica si es igual o diferente
Si es igual ⇒ no se hace nada
Si es diferente ⇒ se ejecuta una sentencia UPDATE.
se graba el log de la sentencia UPDATE
Si no existe ⇒ se ejecuta una sentencia INSERT.
se graba el log de la sentencia INSERT

La segunda parte del proceso de actualización de una página es igual al proceso de carga.

4.5. Segunda Solución

Ya no se aplica la primera meta mencionada de tratar de cargar y actualizar una base de datos relacional cuyo esquema es exactamente el brindado por el usuario. La idea es cargar una base de datos cuyo esquema es similar al brindado por el usuario con las modificaciones que se detallan a continuación:

A cada tabla se le agregan las columnas:

1. ***OrigenDatos***: donde se almacena la identificación de la página desde donde se cargó el registro ya sea la URL o el nombre del archivo si es una página web local.
2. ***FechaProcesamiento***: que es la fecha en que se cargó la página o la fecha en que se modificó por última vez.

Se modifican las claves (Primary Key) de las tablas. Si en el esquema brindado por el usuario la tabla tenía definida una clave a la misma se le agrega la columna “OrigenDatos”. En caso contrario la clave son todas las columnas excepto “FechaProcesamiento”.

Siguiendo con el ejemplo de la cartelera de cines se presenta una de las tablas modificada

```
CREATE TABLE CINES (
  NOMBRE VARCHAR(50),
  DIRECCION VARCHAR(50),
  TELEFONOS VARCHAR(50),
  ORIGENDATOS VARCHAR(100),
  FECHAPROCESAMIENTO VARCHAR(40),
CONSTRAINT PK_CINES PRIMARY KEY (ORIGENDATOS,NOMBRE) );
```

Con esta solución se simplifica el mecanismo de carga y el de actualización.

4.5.1. Mecanismo de carga

El mecanismo de carga inicial es similar al presentado en la sección ***4.3*** excepto en la parte de grabar la información en la base de datos.

Algoritmo de carga

Para realizar la carga inicial de la base de datos por cada tabla del esquema relacional se crea en memoria la tabla con los datos según la correspondencia definida.

Por cada fila de dicha tabla
    Se ejecuta una sentencia INSERT
        (incluyendo OrigenDatos que es la identificación de la página y FechaProcesamiento)
    Se graba el log de la sentencia INSERT

4.5.2. Mecanismo de actualización

El mecanismo de actualización es similar al presentado en la sección ***4.4*** en cuanto a la detección de los cambios en la estructura de la página y los ajustes en la correspondencia que debe realizar el usuario. La diferencia se encuentra en la propagación de la información a la base de datos. En esta segunda propuesta no tienen sentido pensar en dos alternativas una incremental y otra de recarga, debido a que no se realiza una unión de los datos provenientes de distintas páginas. Al estar bien identificado el origen de la información se puede agregar, modificar y borrar al procesar una página.

Tiene como ventaja que no tengo los problemas al juntar los datos y/o actualización mencionados anteriormente, puedo reflejar en la base de datos todo el contenido de las páginas web consideradas. Por otro lado no junta los datos lo cual se podría presentar al usuario definiendo vistas sobre la base de datos. En cuanto a la unión de la información se puede consultar [41] por más información.

Algoritmo de actualización

Para realizar la actualización de la base de datos por cada tabla del esquema relacional se crea en memoria la tabla con los datos según la correspondencia ajustada si fue necesario.

Por cada fila de dicha tabla

Se ejecuta un SELECT para verificar si existe la fila en la base de datos
(recordar que es para esa página porque OrigenDatos es parte de la clave)
Si existe ⇒ se verifica si es igual o diferente
Si es igual ⇒ se realiza una sentencia UPDATE modificando únicamente
la FechaProcesamiento (no se graba un log)
Si es diferente ⇒ se ejecuta una sentencia UPDATE
(incluyendo la FechaProcesamiento)
se graba el log de la sentencia UPDATE
Si no existe ⇒ se ejecuta una sentencia INSERT.
(incluyendo OrigenDatos y FechaProcesamiento)
se graba el log de la sentencia INSERT

Por cada registro existente en la tabla de la base de datos con OrigenDatos la página actual y FechaProcesamiento menor a la actual se realiza una sentencia DELETE y se graba el log correspondiente.

5. Diseño e implementación de un prototipo

Se realizó un prototipo en lenguaje Java que implementa el mecanismo descripto, del cual ya se presentaron algunas pantallas. En realidad son dos prototipos uno para cada propuesta. Se describe el prototipo de la primer propuesta porque el segundo es muy similar y aunque agrega columnas a las tablas que define en la base de datos sus algoritmos son más simples.

En la sección ***5.1*** se presentan los elementos con los cuales trabaja el prototipo utilizando el modelo conceptual. En la sección ***5.2*** se describe el diagrama de secuencia del sistema que muestra los eventos que ocurren. En la sección ***5.3*** se describen los mensajes que controlan la comunicación entre los distintos objetos con el diagrama de colaboración. Y por último en la sección ***5.4*** se muestra el diagrama de clases correspondiente. Se utiliza el lenguaje UML (Unified Modeling Language) que es una notación estándar para el modelado de objetos en la orientación a objetos, por más detalle ver [40].

5.1. Modelo Conceptual

Un modelo conceptual es una representación de conceptos en el dominio del problema. Se modelan las páginas web desde donde se extrae información, pero sólo se consideran las tablas que es la parte con la cual se trabaja. Se modela el esquema relacional de la base de datos en la cual se carga la información. Se modela la correspondencia (llamada mapeo) establecida por el usuario entre las columnas de las tablas del esquema relacional y las columnas de las tablas presentes en las páginas web.

En la Figura 11 se presenta el modelo conceptual correspondiente.

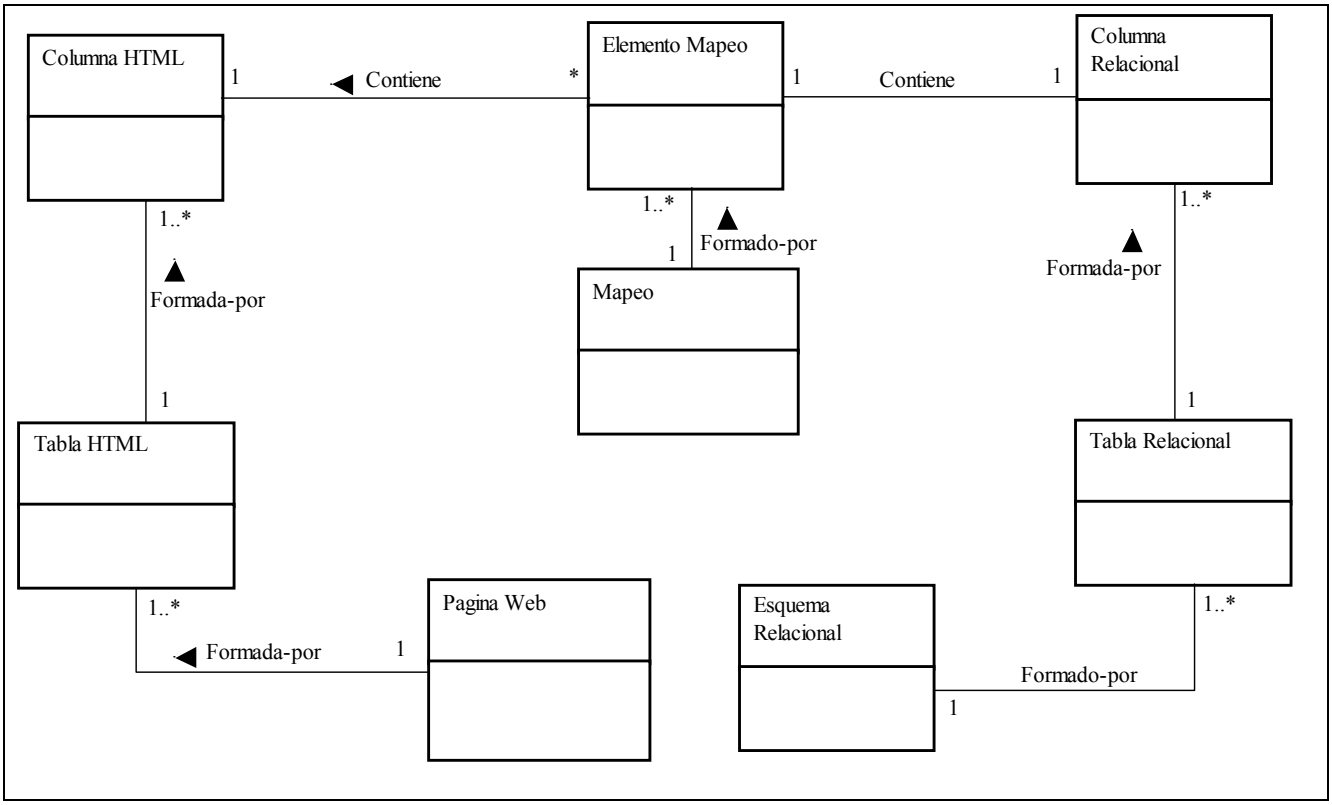


Figura 11 - Modelo Conceptual

El punto de partida del presente trabajo es el esquema relacional y una página web. Esto será el inicio para explicar el modelo anterior. El esquema relacional es un conjunto de sentencias SQL de creación de las tablas de la base de datos, es decir que es un conjunto de tablas relacionales. A su vez cada tabla relacional es considerada como un conjunto de columnas relacionales. No es necesario representar todos los elementos de una página web sólo aquellos que son relevantes. Por dicho motivo se considera que una página web es un conjunto de tablas HTML. Las tablas HTML a su vez se representan como un conjunto de columnas HTML.

La correspondencia o mapeo entre el esquema relacional y la página web se realiza indicando columnas. Entonces se considera como un conjunto de objetos llamados “elemento mapeo”. Cada uno de estos elementos indica la correspondencia entre dos columnas, una relacional y otra HTML.

5.2. Diagrama de secuencia del sistema

El diagrama de secuencia del sistema muestra al sistema como una “caja negra”. En dicho diagrama se indican las operaciones que debe realizar el sistema pero no el detalle de la forma en que las debe realizar. Se presentan tres diagramas que corresponden uno al proceso de la carga inicial de una página web, otro para la actualización de la base de datos con la alternativa de recarga y el último para el proceso de propagación de los cambios en forma incremental.

Diagrama del proceso de carga inicial de una página web:

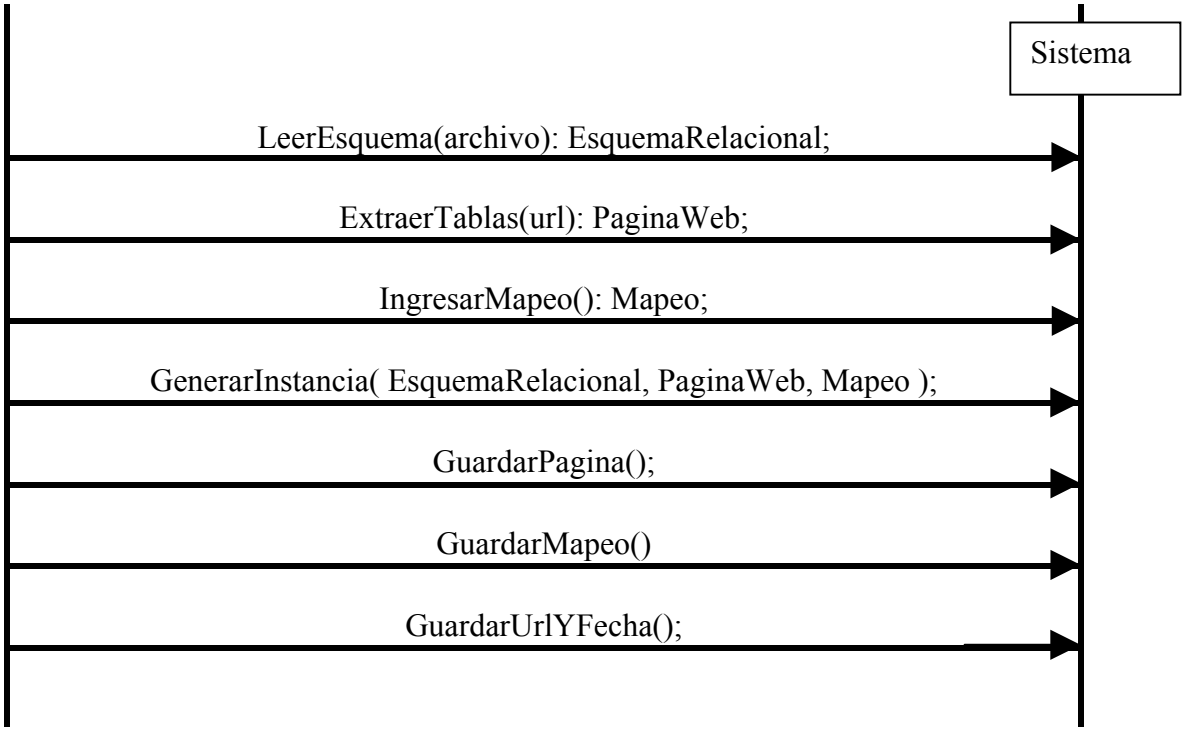


Diagrama del proceso de recarga:

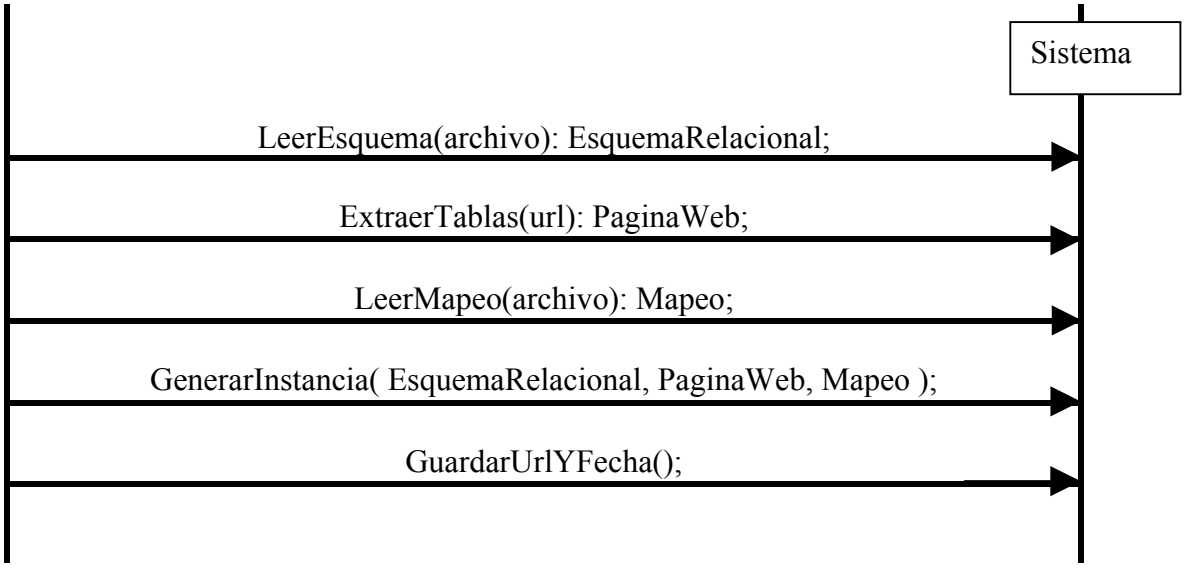
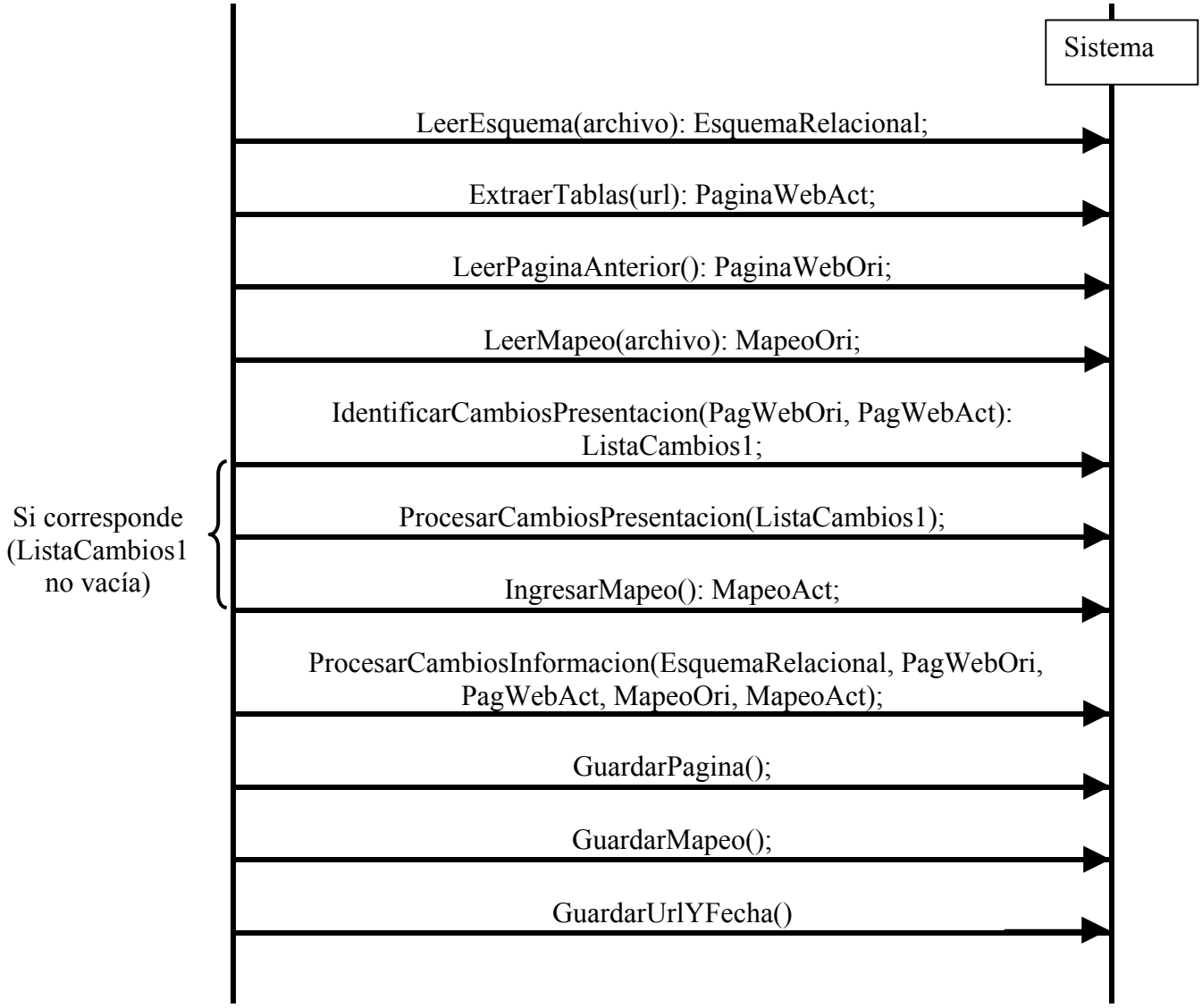
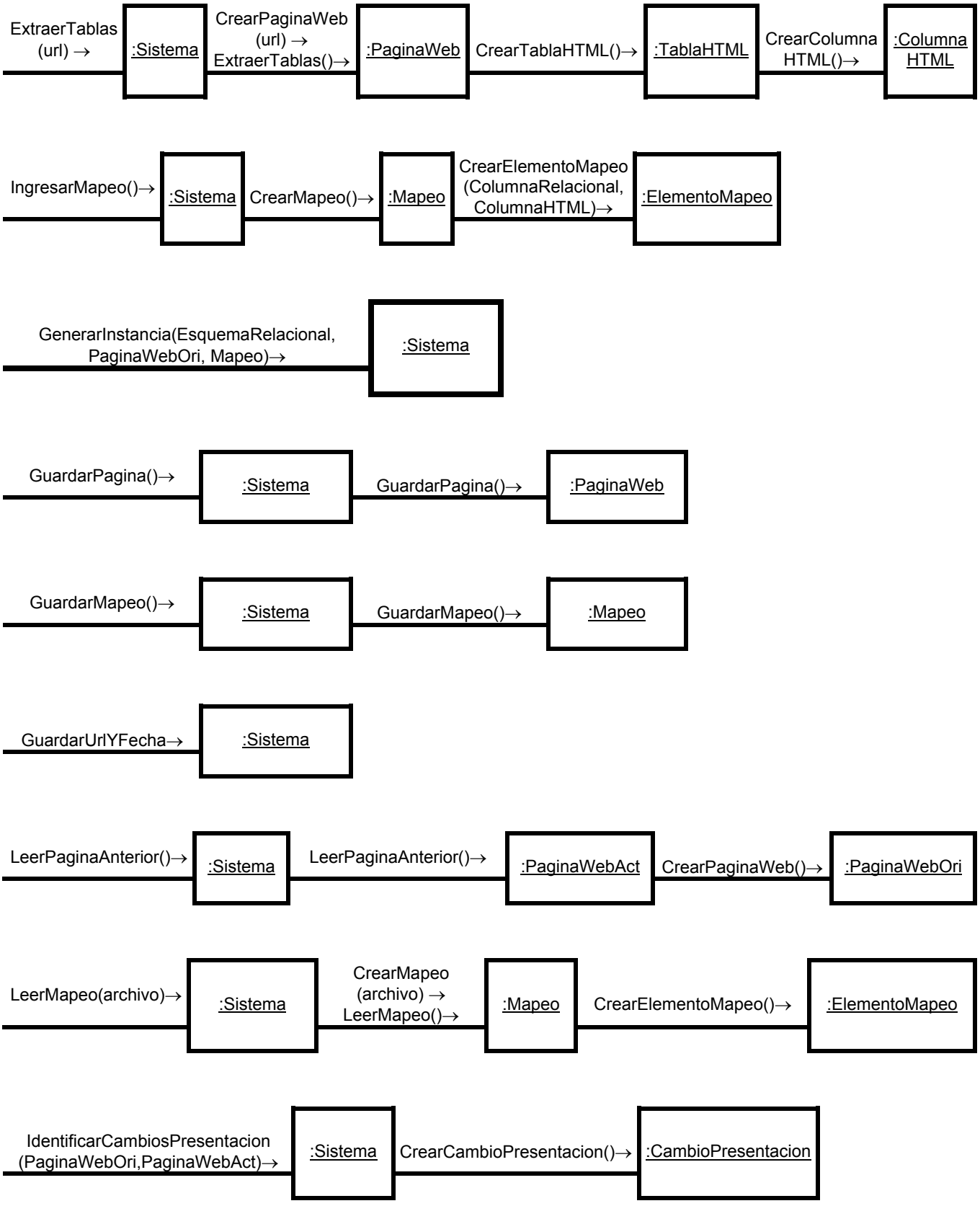


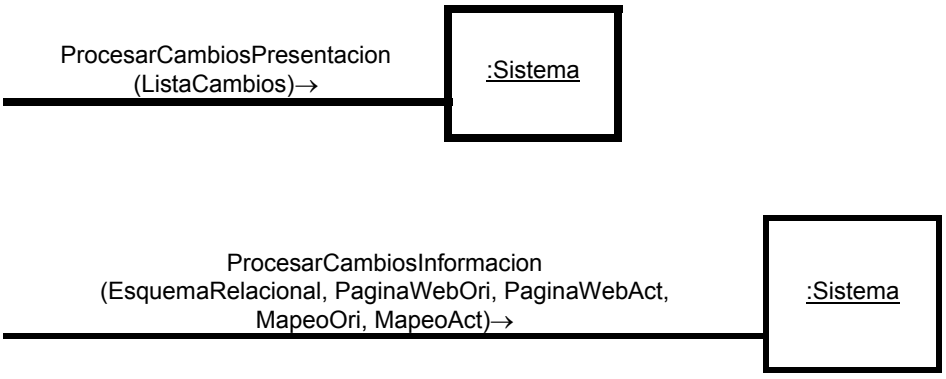
Diagrama del proceso de actualización en forma incremental:



5.3. Diagrama de Colaboración

En el diagrama de colaboración se representa la interacción que hay entre los objetos identificados en el modelo conceptual para cada una de las operaciones identificadas en el diagrama de secuencia del sistema.


ExtraerTablas (url) →
:Sistema
CrearPaginaWeb (url) → ExtraerTablas()→
:PaginaWeb
CrearTablaHTML()→
:TablaHTML
CrearColumna HTML()→
:Columna HTML
IngresarMapeo()→
:Sistema
CrearMapeo()→
:Mapeo
CrearElementoMapeo (ColumnaRelacional, ColumnaHTML)→
:ElementoMapeo
GenerarInstancia(EsquemaRelacional, PaginaWebOri, Mapeo)→
:Sistema
GuardarPagina()→
:Sistema
GuardarPagina()→
:PaginaWeb
GuardarMapeo()→
:Sistema
GuardarMapeo()→
:Mapeo
GuardarUrlYFecha→
:Sistema
LeerPaginaAnterior()→
:Sistema
LeerPaginaAnterior()→
:PaginaWebAct
CrearPaginaWeb()→
:PaginaWebOri
LeerMapeo(archivo)→
:Sistema
CrearMapeo (archivo) → LeerMapeo()→
:Mapeo
CrearElementoMapeo()→
:ElementoMapeo
IdentificarCambiosPresentacion (PaginaWebOri,PaginaWebAct)→
:Sistema
CrearCambioPresentacion()→
:CambioPresentacion



5.4. Diagrama de Clases

El diagrama de clases describe gráficamente las clases implementadas en el prototipo según se muestra en la Figura 12.

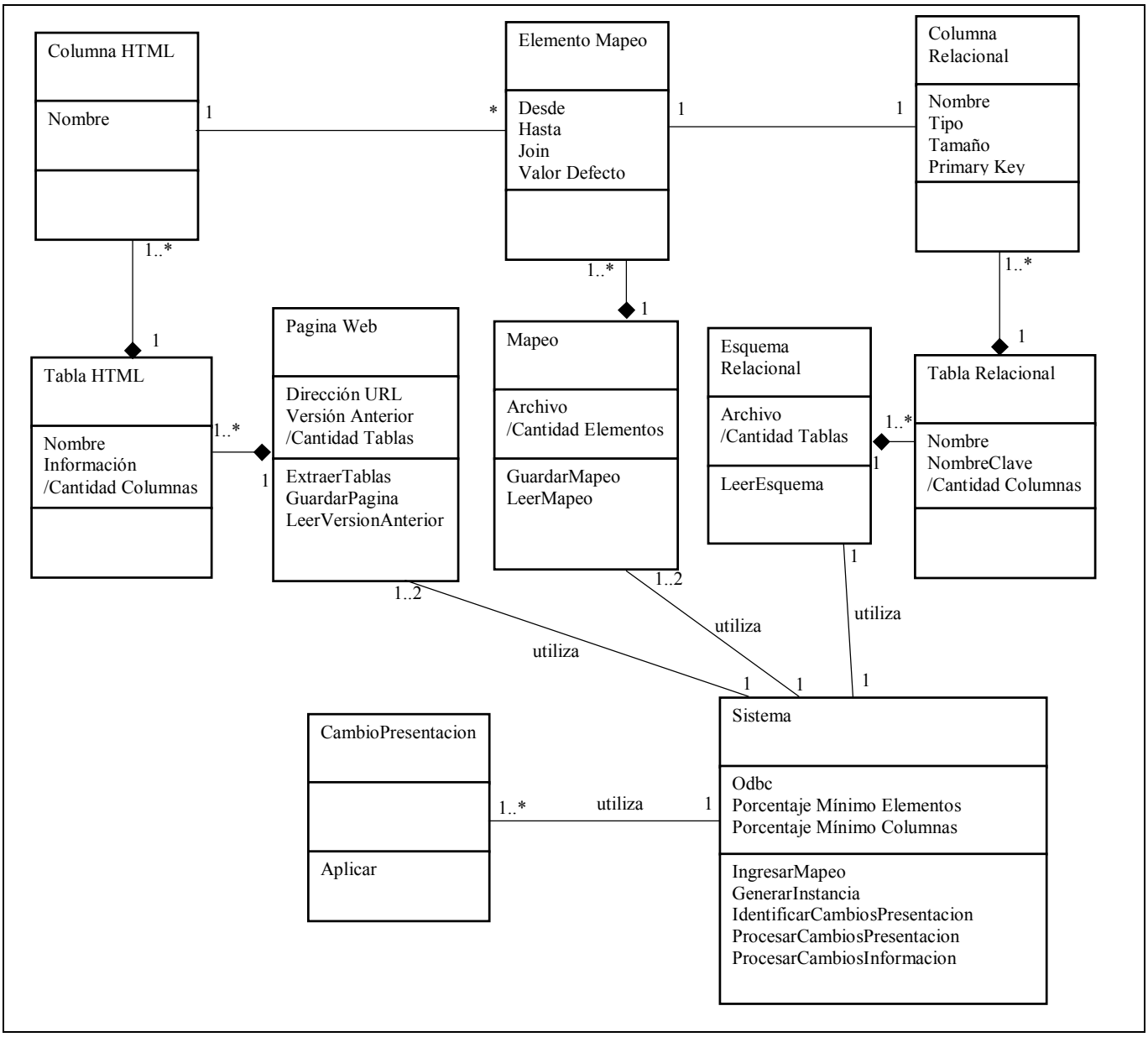


Figura 12 - Diagrama de clases.

5.5. Ejecución

Para ejecutar el prototipo del proyecto se debe crear una conexión odbc llamada “proyecto” el cual es utilizado para realizar la comunicación con la base de datos.

El prototipo se ejecuta por línea de comando con la siguiente sintaxis:

java Proyecto <operacion> <esquema> <pagina> <mapeo> <min-elementos> <min-columnas>

Donde los parámetros son:

<operacion>: Un string que le indica al prototipo la tarea que debe realizar. Dicho string puede ser ***cargainicial***, en este caso le consulta al usuario la correspondencia y carga la base de datos. Puede ser ***recarga***, en este caso lee la correspondencia definida y carga la base de datos. O puede ser ***cambios***, en este caso actualiza la base de datos en forma incremental.

<esquema>: Nombre del archivo de texto con el esquema relacional.

<pagina>: URL o nombre del archivo correspondiente a la página web. Con el mismo nombre pero con extensión "***.ori"*** (por original) guarda la representación de la página para luego leerla en el proceso de actualización. Y con el mismo nombre pero con extensión "***.log"*** guarda las sentencias SQL ejecutadas en la base de datos para dicha página web.

<mapeo>: Nombre del archivo de la correspondencia definida. Con la opción "cargainicial" sólo lo guarda, con "recarga" sólo lo lee y en la actualización de "cambios" lo lee y lo guarda.

<min-elementos>: Porcentaje que indica la cantidad mínima de elementos iguales en el algoritmo de comparación de las tablas existentes en las páginas web para determinar si dos columnas de una tabla se corresponden.

<min-columnas>: Porcentaje que indica la cantidad mínima de columnas iguales en el algoritmo de comparación de las tablas existentes en las páginas web para determinar si dos tablas se corresponden.

6. Resultados

Para evaluar el prototipo se realizaron varias pruebas ejecutándolo en un PC pentium de 433 Mhz con 32 M de memoria.

Una prueba se realizó con el mismo dominio del ejemplo, el de los cines. Es decir llevar actualizada una base de datos con información de los cines, las películas y su proyección. Las razones de utilizar este dominio fueron: por un lado existen en Internet varias páginas con carteleras de cines y por otro su frecuencia de cambios, debido a que cada fin de semana en general se producen estrenos. Por lo tanto se puede testear el prototipo con páginas reales. Pero se aclara que es totalmente genérico aplicable a cualquier otro dominio de interés para el usuario.

Para realizar una prueba del prototipo consideramos las siguientes páginas con información de cines:

- http://multimedios.com.uy/cine/default.asp
- http://www.informes.com.uy/PaginasSecundarias/Cines/cineshorario2.html
- http://www.observa.com.uy/tiempolibre/cines/cartelera.asp

Se obtuvieron los siguientes tiempos:

1. La carga inicial de las páginas ser realizó en aproximadamente un minuto por página.

2. Se modificó una página y se actualizó la Base de Datos por el procedimiento de recarga, es decir borrar el contenido y cargar todas las páginas nuevamente. En total se demoró aproximadamente tres minutos sin importar cuantos cambios se hacían a la página.

3. Se modificó una página y se actualizó la Base de Datos por el procedimiento de propagación incremental de los cambios. Dependiendo de la cantidad de cambios demoró aproximadamente entre uno o dos minutos, lo cual es mejor que la opción anterior.

Para la propagación de los cambios se mostró con este ejemplo sencillo que una alternativa no es siempre mejor que la otra. Considerando el ejemplo, si la base de datos se carga con una única página es mejor la alternativa de actualización por recarga de la base de datos. En cambio, si en la base de datos se cargan tres páginas web es mejor la alternativa de la propagación de los cambios en forma incremental. Esta segunda alternativa además tiene la ventaja de que se manipula una única página web.

Una breve observación de las tareas implicadas nos puede guiar para determinar cual de las alternativas puede ser mejor.

En el proceso de actualización en forma incremental propagando los ***cambios*** tenemos que el costo puede ser considerado como:

costo = costo de ajustar el mapeo +
costo de borrar lo que ya no existe +
costo de cargar la información de la página

En el proceso de actualización por ***recarga*** tenemos que el costo puede ser considerado como:

costo = costo de vaciar la base +
costo de ingresar el mapeo +
costo de cargar la información de la página +
costo de cargar la información de las otras páginas

El costo de vaciar la base varía según la base de datos. Por ejemplo si utilizamos archivos tipo dbf's o Access puede ser el costo de borrar el o los archivos correspondientes. En cambio si utilizamos una base de datos ORACLE o SQL Server, el costo será la suma de todas las sentencias DELETE necesarias o las sentencias de borrar las tablas. Por lo tanto no podemos comparar este costo con el costo de borrar lo que ya no existe de una página.

En cambio podemos considerar que el costo de ajustar el mapeo o ingresar el mapeo es similar. En ambos casos se presenta una interfase al usuario para que ingrese la correspondencia deseada.

Por último, en ambas alternativas tenemos un costo que es igual, el costo de cargar la información de la página que cambió.

Por lo tanto, al comparar las alternativas nos queda lo siguiente:

propagación de los cambios		**recarga**
costo de ajustar el mapeo +	**<**	costo de vaciar la base +
costo de borrar lo que ya no existe +	**>**	costo de ingresar el mapeo +
costo de cargar la información de la página	**?**	costo de cargar la información de la página + costo de cargar la información de las otras páginas

Simplificamos según los comentarios previos y nos queda:

propagación de los cambios		**recarga**
costo de borrar lo que ya no existe	<	costo de vaciar la base +
	>	costo de cargar la información de las otras páginas
	?	

De lo cual, podemos observar que cuanto más páginas se utilizan para cargar la base de datos más costosa es la alternativa de recarga. Para optar por aplicar una de las alternativas hay que determinar que relación cumplen los costos mencionados en el caso específico.

Con el ejemplo de cines se probó el funcionamiento del prototipo con cambios en la información contenida en tablas de páginas web. Como no se encontraron páginas web que cambien frecuentemente la estructura de tablas se probó el prototipo con ejemplos creados con dicho fin.

7. Conclusiones y trabajo futuro

Este trabajo aporta un relevamiento de las herramientas existentes para los problemas encontrados en la tarea de cargar y mantener información en una base de datos a partir de páginas web.

La propuesta se realizó para trabajar con páginas web reales. Para ello se observó la estructura de las mismas y se aplicaron heurísticas para resolver los problemas encontrados. Como ser el hecho de considerar sólo las tablas más internas en el anidamiento y basar la comparación de las tablas en el contenido de las columnas.

Se implementó un prototipo del mecanismo propuesto, permitiendo una mayor capacidad en el proceso de carga de la base de datos en comparación con el trabajo existente y agregando el proceso de actualización de la base de datos frente a los cambios. El principal aporte de tener un prototipo es que sirve como punto de partida para la implementación de otros trabajos.

Si se trabaja con el esquema de la base de datos indicado por el usuario se realiza una unión de los datos básica enfrentándose a problemas que no se pueden resolver. En cuanto al mecanismo de actualización de la base de datos las alternativas propuestas son recargar la base de datos o realizar la propagación de los cambios en forma incremental, se mostró que no siempre una alternativa es mejor que la otra sino que depende de cada caso.

Si se modifica el esquema de la base de datos indicado por el usuario no se realiza la unión de los datos dejando claramente identificado el origen de la información y la fecha en que se procesó la página por última vez. En esta propuesta no tiene sentido hablar de dos alternativas de actualización como en la propuesta previa de recarga o incremental.

Como posibles extensiones se puede considerar ampliar la propuesta para trabajar con otras estructuras de las páginas web, como ser listas o párrafos de texto. Utilizar otros lenguajes como puede ser XML. Además integrar el tema de cambios en el esquema de la base de datos. O basados en la segunda propuesta agregarle la parte de integración de los datos.

8. Glosario

DOM: Document Object Model.
HTML: HyperText Markup Language.
Java: Lenguaje de programación utilizado.
Mapeo: Correspondencia definida entre el esquema relacional y las tablas de la página web.
Monitorear: Proceso de observar las páginas para verificar cuándo ocurre un cambio.
ODBC: Open DataBase Connectivity.
ODMG: Object Data Management Group.
SQL: Search Query Language.
Tag: Marca del lenguaje HTML.
UML: Unified Modeling Language.
URL: Uniform Resource Locator.
XML: eXtensible Markup Language.

9. Referencias

[1] http://www.netmind.com
[2] http://es.buzzcity.com
[3] http://www.urlywarning.com
[4] http://www.htmldiff.com
[5] Yih-Farn Chen, Fred Douglis, Huale Huang, Kiem-Phong Vo: **TopBlend: An Efficient Implementation of HtmlDiff in Java**. AT&T Labs - Research Technical Report 00.5.1. - **2000**.
[6] http://www.htmlcompare.com
[7] Fred Douglis, Thomas Ball, Yih-Farn Chen, Eleftherios Koutsofios: **WebGUIDE: Querying and Navigating Changes in Web Repositories**. WWW5 / Computer Networks 28(7-11): 1335-1344 - **1996**.
[8] Fred Douglis, Thomas Ball: **Tracking and Viewing Changes on the Web**. USENIX Annual Technical Conference 1996: 165-176 - **1996**.
[9] Mario Magnanelli, Antonia Erni and Moira Norrie: **ACADEMIA: An Agent-Maintained Database based on Information Extraction from Web Documents**. Institute for Information Systems, ETH Zürich. April 1998. Proc. of 14th European Meeting on Cybernetics and Systems Research (EMCSR'98), Vienna, Austria. **1998**.
[10] http://www.dia.uniroma3.it/Araneus/
[11] G. Mecca, P. Atzeni, A. Masci, P. Merialdo, G. Sindoni: **From Databases to Web-Bases: The ARANEUS Experience** - Technical Report n. 34-1998 - Dipartimento di Informatica e Automazione, Universita' di Roma Tre, May, **1998**.
[12] Giansalvatore Mecca, Paolo Atzeni, Alessandro Masci, Paolo Merialdo, Giuseppe Sindoni: **The Araneus Web-Base Management System**. SIGMOD Conference 1998: 544-546 - **1998**.
[13] Paolo Atzeni, Giansalvatore Mecca, Paolo Merialdo: **To Weave the Web**. VLDB 1997: 206-215 - **1997**.
[14] Stéphane Grumbach, Giansalvatore Mecca: **In Search of the Lost Schema**. ICDT 1999: 314-331 - **1999**.
[15] Grammars have Exceptions Valter Crescenzi, Giansalvatore Mecca: **Grammars Have Exceptions** - Information Systems, Special Issue on Semistructured Data, **1998**.
[16] Giansalvatore Mecca, Paolo Atzeni: **Cut and Paste**. JCSS 58(3): 453-482 - **1999**.

[17] Giansalvatore Mecca, Alberto O. Mendelzon, Paolo Merialdo: **Efficient Queries over Web Views**. EDBT 1998: 72-86 - **1998**.
[18] Paolo Atzeni, Alessandro Masci, Giansalvatore Mecca, Paolo Merialdo, Elena Tabet: **ULIXES: Building Relational Views over the Web**. ICDE 1997: 576 - **1997**.
[19] Giansalvatore Mecca, Paolo Merialdo, Paolo Atzeni, Valter Crescenzi: **The ARANEUS Guide to Web-Site Development**. SEBD 1999: 167-177 - **1999**.
[20] Paolo Atzeni, Giansalvatore Mecca, Paolo Merialdo: **Design and Maintenance of Data-Intensive Web Sites**. EDBT 1998: 436-450 - **1998**.
[21] Giuseppe Sindoni: **Incremental Management of Hypertextual Views**. WebDB 1998: 98-117 - **1998**.
[22] Fernando Perelló, Jaime Ferreiro, Regina Motz y Dina Wonsever. **Generación automática de una base de datos con información extraída de la web**. En Congreso Argentino de Ciencias de la Computación , CACIC 2000, Ushuaia, Argentina, Octubre del **2000**.
[23] http://www.cc.gatech.edu/projects/disl/WebCQ - http://www.cse.ogi.edu/~lingliu/CQ/
[24] Ling Liu, Calton Pu, Wei Tang: **WebCQ: Detecting and Delivering Information Changes on the Web**. CIKM 2000: 512-519 - **2000**.
[25] Calton Pu and Ling Liu. **Update Monitoring: The CQ project**. (invited paper) in: The 2nd International Conference on Worldwide Computing and Its Applications - WWCA'98, Tsukuba, Japan, Lecture Notes in Computer Science 1368, pp396-411. **1998**.
[26] Ling Liu, Calton Pu, Wei Tang and Wei Han. **Conquer: A Continual Query System for Update Monitoring in the WWW**. To appear in special issue on Web semantics, International journal of Computer Systems, Science and Engineering. Marzo 1999.
[27] Santi Saeyor, Mitsuru Ishizuka: **WebBeholder: A Revolution in Tracking and Viewing Changes on The Web by Agent Community**. Proc. WebNet98, Orlando Florida, Noviembre 1998.
[28] Xin Zhang, Gail Mitchell, Wang-Chien Lee, Elke A. Rundesteiner: **Clock: Synchronizing Internal Relational Storage with External XML Documents**. Procedings of the 11th International Workshop on Research Issues in Data Engineering, Document Management. Alemania 2001.
[29] http://www.tracerlock.com
[30] http://informant.dartmouth.edu/
[31] Gerald Huck, Peter Fankhauser, Karl Aberer, Erich J. Neuhold: **Jedi: Extracting and Synthesizing Information from the Web**. CoopIS 1998: 32-43 - **1998**.
[32] http://www.tropea-inc.com/
[33] Fabien Azavant, Arnaud Sahuguet: **Word Wide Web Wrapper Factory (W4F) User Manual**
[34] www.altavista.com
[35] www.lycos.com
[36] www.excite.com
[37] www.infoseek.com
[38] java.sun.com
[39] Alejandro Gutiérrez, Regina Motz, Daniel Viera. **Building Databases with Information Extracted from Web Documents**. XX International Conference of the Chilean Computer Science Society (SCCC). Santiago, Chile. November **2000**.
[40] **UML y PATRONES Introducción al análisis y diseño orientado a objetos**. Craig Larman. PRENTICE HALL, México **1999**.
[41] **Data Cleaning: Problems and Current Approaches.** Erhard Rahm, Hong Hai Do. IEEE Data Engineering Bulletin 23. **2000**.