UNIVERSITAT JAUME I

Escola Superior de Tecnologia i Ciències Experimentals

Departament de Llenguatges i Sistemes Informàtics

TESIS DOCTORAL

Extracción y Recuperación de Información Temporal

Presentada por:

Dolores María Llidó Escrivá

Dirigida por:

Rafael Berlanga Llavorí y María José Aramburu Cabo

Castellón, 2002

UNIVERSITAT JAUME I

Escola Superior de Tecnologia i Ciències Experimentals

Departament de Llenguatges i Sistemes Informàtics

TESIS DOCTORAL

Extracción y Recuperación de Información Temporal

Memoria presentada por Dolores María Llidó Escrivá para optar al grado de Doctora en Informática por la Universitat Jaume I

Dirigida por: Rafael Berlanga Llavorí y María José Aramburu Cabo

Castellón, 2002

A mi marido y a mis padres

"La ciencia tiene una característica maravillosa, y es que aprende de sus errores, que utiliza sus equivocaciones para reexaminar los problemas y volver a intentar resolverlos, cada vez por nuevos caminos. "

Ruy Perez Tamaño

Agradecimientos

En todo proceso de realización de una tesis doctoral hay muchas personas que nos apoyan y empujan para que la llevemos a término. En primer lugar agradecer a mis directores de Tesis, la doctora Mª José Aramburu y el doctor Rafael Berlanga, que me hicieron indagar en el campo de la Extracción y Recuperación de Información, y apoyaron para que esta tesis llegara finalmente a buen puerto.

Quiero agradecer el apoyo especial de mis padres, Manolo y Dolores, y sobretodo el apoyo incondicional de mi marido Manolo Gil. Quería hacer una mención especial a todos aquellos familiares he tenido que dejar a un lado, y a muchos kilómetros, con el fin de finalizar esta tesis.

También quería agradecer el empuje que me han dado los compañeros del grupo de investigación de Bases de Conocimiento Temporal, Juan Manuel Perez, e Ismael Sanz, y a los compañeros de los departamantos de Lenguajes y Sistemas Informáticos, y de Ingenieria y Ciencias de los Computadores de la Universitat Jaume I, en especial al Dr. Michael Gould con el que he compartido muchos Proyectos de investigación dentro del campo de los Sistemas de Información Geográfica.

Por último me gustaría agradecer a mis antiguos compañeros del Instituto de Robótica de la Universidad de Valencia, que me introdujeron en el interesante mundo de la investigación, y me apoyaron en mis primeras investigaciones, en concreto a Enrique Bonet, Juan Domingo Esteve, Pablo Barrachina, Ramón Cirilo, Ariadna Fuertes, Paco Soriano y muchos otros con los que compartí muchas pizzas esas noches que finalizaban las entregas de los Proyectos Europeos. Al Dr. Gregorio Martín y al Dr. Juan Pelechano, que me concedieron mis primeras becas de investigación, permitiéndome entrar en el mundo universitario.

Resumen

En este trabajo, se presentan y evalúan diferentes aproximaciones novedosas para la búsqueda y detección de sucesos en grandes colecciones de documentos, los cuales se basan en el aprovechamiento de las propiedades temporales intrínsecas de los documentos.

Para obtener las propiedades temporales presentes en los documentos hemos de extraer y analizar las referencias temporales presentes en los documentos. A la aplicación diseñada con este fin la hemos denominado TimeExtractor. Esta aplicación permite extraer y representar en el tiempo absoluto todas las referencias temporales presentes en los documentos, o sea los patrones de fecha y las expresiones absolutas o relativas. Para facilitar esta tarea hemos definido un modelo de tiempo basado en el Calendario Gregoriano. Este modelo permite representar puntos, intervalos o duraciones de tiempo, basándonos en el uso de multiples granularidades y además establece un álgebra para relacionar estas entidades y realizar desplazamientos en el tiempo.

La dificultad del manejo de múltiples fechas e intervalos como metadato, nos sugiere la necesidad de asignar de modo automático un dato simple que nos permita representar el intervalo en que transcurre la acción principal narrada en cada documento, al cual llamaremos *periodo de suceso*. Hemos diseñado un algoritmo que a partir del análisis estadístico de las referencias temporales presentes en los documentos y su proximidad con la fecha de publicación calcula automáticamente el *periodo de suceso*.

Con el propósito de ver si las propiedades temporales extraídas de los documentos, ya sean la lista de fechas o el *periodo de suceso* mejoran los sistemas de Recuperación de Información y Detección de Sucesos, hemos diseñado diversas herramientas.

Por un lado, se comprueba que la consideración del *periodo de suceso*, en un sistema de Recuperación de Información ayuda a los usuarios a detectar sucesos

eficazmente, siempre que el usuario conozca las características básicas del suceso que desea buscar, es decir, las palabras clave representativas del suceso, y el intervalo de tiempo en el que éste se ha producido.

En realidad la mayoría de las veces generalmente el usuario no sabe cuándo se ha producido un suceso, o está interesado en conocer distintos sucesos sobre cierto tema particular. Por ello hemos decidido definir una nueva herramienta ,TimExplR, que organiza los documentos recuperados por un sistema de Recuperación de Información no solo por la relevancia respecto a la consulta, sino agrupando los documentos por sus *periodos de suceso*. Para permitir la exploración de estos grupos se ha diseñado un interfaz gráfico que los visualiza de diversos modos. Por un lado, muestra un histograma con las particiones de los documentos relevantes organizados por las fechas en que se producen los sucesos. Y por otro, muestra una secuencia de *crónicas*,o segmentos temporales que agrupan los documentos de un mismo suceso.

Por último, se ha diseñado una herramienta que permite a los usuarios conocer todos los tópicos que existen en una colección de documentos. Donde un tópico estará formado por todos los documentos que relatan un mismo suceso, o estén directamente relacionados con él. Los sistemas de detección de sucesos son una de las tareas de los Sistemas de Detección y Seguimiento de Tópicos, que se resuelven utilizando algoritmos de clasificación supervisada. En nuestro sistema, hemos decidido utilizar un algoritmo de clasificación supervisada muy simple, pero que en los Sistemas de Detección de sucesos da buenos resultados, el algoritmo de *Single Pass*. La diferencia con los sistemas propuestos en la literatura es la utilización de las propiedades temporales intrínsecas de los documentos. La mayoría de sistemas toman la aproximación de que la fecha de publicación coincide con la fecha en que se produce el suceso. Hemos comparado los resultados de la utilización de las fechas presentes en los documentos y del *periodo de suceso*. Como resultado hemos comprobado que con estas dos propiedades temporales se obtienen resultados similares, y que además mejoran los resultados de los sistemas de detección que utilizan sólo la fecha de publicación. De aquí, se concluye el periodo de suceso es una buena propiedad temporal para representar el tiempo en que transcurren los sucesos, y que produce mejoras en los sistemas de Recuperación de Información y Detección de Sucesos.

Abstract

Many digital documents that currently populate the Web have a relevant temporal component. Newspapers, articles, medical reports and legal text are some examples of documents whose contents can be clearly located along time.

This thesis concerns with the automatic generation of document temporal metadata, and how to exploit it in current Information Retrieval (IR) and Topic Detection and Tracking (TDT) systems. Specifically we first propose a Time Model to represent temporal expressions and its relationships. Secondly we propose a method to automatically extract temporal expressions in Natural Language. With TimeExtractor we extract from the text all temporal references and for each temporal reference we try to assign a point or interval in the Gregorian Calendar. To obtain the *event-time* period we make a statistical analysis of all the dates and intervals that appear on it, taking into account the proximity with the publication date.

In this work, we have demonstrated how the *event-time* period improves the results of the Topic Detection and Tracking systems in the Topic Detection Task, with similar results as the use of all the dates present in the documents. As the use of a list of dates and intervals is usually not implemented in IR and TDT systems, and because its processing is very time consuming, we think that the use of the *event-time* period as metadata is more appropriate to help the user to retrieve document related with events and topics.

As current Information Retrieval Systems can include document metadata, we also try to improve these systems by adding automatically the event-time period to each document. We have created a graphical interface, TimExplR, to allow users to explore the documents retrieved by the system according to their temporal properties. This tool helps users to retrieve the desired event when he/she does not know exactly when it has happened, showing to the user a histogram that contains the documents grouped by the *event-time* period, and a sequence of *chronicles* of the different events related with the query.

Acrónimos

- EI : Extracción de Información
- IA : Inteligencia Artificial
- LN: Lenguaje Natural
- ME : Multilingual Entity Task
- MUC :Message Understanding Conference
- NE : Name Entity Recognition Task
- RI : Recuperación de la Información
- Q & A : Question and Answering
- TDRL:Temporal Document Retrieval Language
- TDT : Topic Detection and Tracking
- TE : Template Element Task
- TOOOR :Temporal Object-Oriented Document Organisation and Retrieval
- TR : Template Relation Task
- TREC :Text Retrieval Conferences
- SUMMAC :Summarization Conference

Índice general

Índice de figuras

List of Algorithms

Índice de cuadros

Capítulo 1

Introducción

En la actualidad existen Sistemas de *Recuperación de Información* (RI) que permiten búsquedas con fechas, para ayudar a encontrar documentos sobre cierta información o suceso. Por ejemplo, en los repositorios sobre datos geográficos, la fecha en la que se ha actualizado un determinado mapa resulta crucial para los usuarios. Pero en el caso de búsquedas de sucesos en repositorios de documentos, la fecha de publicación no siempre ayuda al usuario, debido principalmente a tres razones. Primero, el usuario no siempre sabe cuándo se publicó el suceso, sino que generalmente lo que conoce es en qué momento transcurre el suceso. Segundo, cuando se produce un suceso no tiene por qué coincidir con la fecha de su publicación. Por ejemplo, existen trabajos de investigación que tratan sobre la Edad Media publicados en la actualidad ¿Cómo podrían localizarse estos trabajos?. Por último, existen sucesos de cierta duración, que se publican durante varios días en los documentos de mucha actualidad, como pueden ser los periódicos, pero que no se publican todos los días consecutivos. Otras veces los documentos añaden más datos a investigaciones sobre sucesos ocurridos anteriormente a la fecha de publicación. ¿Cómo puede un sistema automáticamente reconocer que todos estos documentos hablan sobre el mismo suceso?.

En este capítulo referenciaremos algunos trabajos que tratan de detectar los sucesos relatados en documentos de actualidad a partir de su fecha de publicación, y que en sus conclusiones destacan que la fecha de publicación no es suficiente. Por todo esto, nos surge la idea de extraer de los documentos la información temporal que contienen, o sea los periodos de tiempo o fechas que se citan tanto implícita como explícitamente, y estudiar cómo éstas pueden ayudar al usuario a localizar en los repositorios los documentos de su interés.

En las próximas secciones vamos a denotar como *suceso* algo que ocurre en un lugar y tiempo determinado. El 'algo' es lo que denominaremos *tema*, y el tiempo

de ocurrencia del suceso es lo he llamaremos *periodo de suceso*.

Con el propósito principal de permitir al usuario recuperar documentos que hablan sobre un suceso vamos a intentar resolver las siguientes cuestiones:

- ¿Cómo obtener automáticamente de un documento, el intervalo temporal sobre el cual se narran sus hechos, o sea el *periodo de suceso* de un documento?.
- ¿Qué mejoras puede producir la inserción del *periodo de suceso* en un repositorio de documentos? ¿Puede el *periodo de suceso* ayudar a agrupar los documentos que relatan el mismo suceso?

1.1. Motivación

La irrupción de la World Wide Web como medio de comunicación, junto con el aumento de las capacidades de los ordenadores, ha dado lugar a que existan grandes repositorios de documentos que se modifican e incrementan a lo largo del tiempo. El acceso a este tipo de información generalmente se realiza mediante sistemas de *Recuperación de Información* (RI) o los que más coloquialmente se denominan *buscadores*. Pero debido a la cantidad de información presente en los documentos es difícil localizar la información que se desea, sobretodo cuando queremos información sobre algún suceso particular.

Una buena parte de los documentos electrónicos que manejamos a diario nos informan acerca de sucesos que transcurren en un espacio de tiempo determinado. Los periódicos, noticias, informes médicos o textos legales son un claro ejemplo de documentos cuyo contenido está organizado en el tiempo cronológico. Dado que estos tipos de documentos relatan sucesos y las referencias temporales presentes en ellos juegan un papel muy importante para entender su contenido. ¿Cómo podemos recuperar información sobre los sucesos que se relatan en una colección de este tipo de documentos?. ¿Como podemos ayudar al usuario a recuperar los documentos que hablan de un suceso determinado?. Los sistemas de RI permiten obtener todos los documentos que hablan sobre un tema, si se especifica bien la consulta, ya que están diseñados principalmente para buscar palabras o conceptos. El estudio de los sucesos, plantea nuevos desafíos al área de la Recuperación de la Información debido principalmente a dos razones:

- Estos documentos requieren nuevos modelos de datos y consulta para expresar su temporalidad [Ara99].

- Dado que los sucesos evolucionan en el tiempo, es interesante estudiar técnicas automáticas para detectar nuevos sucesos y monitorizar los sucesos en los repositorios.

Sobre el modelado de datos temporales podemos encontrar mucha información en la literatura [Bet00, Dar97, Fer96, Jac97, Ning01, Spc98, Ste97]. En la mayoría de aplicaciones de Recuperación de Información, al igual que en los recientes trabajos en el area del detección de sucesos, *Topic Detection and Tracking* (TDT), se utiliza la fecha de publicación de un documento como un atributo sobre el cual realizar consultas. Dentro del área de TDT se está tratando la problemática de la detección y seguimiento de sucesos [All98b, Yang98] utilizando la fecha de publicación, donde se supone que ésta representa cuándo se produce el suceso. En algunos trabajos dentro del área de TDT se demuestra que la fecha de publicación produce mejoras, pero estas no son suficientes para recuperar sucesos [All98, Swa99, Yan98b]. En el caso de noticias generalmente la fecha de publicación representa cuándo se produce un suceso con un error de un día, aunque hay excepciones debido a dos razones:

- Generalmente cuándo se publica una noticia depende de varios factores, principalmente de la novedad e importancia del suceso que se relata.
- Existen sucesos que no son puntuales, tienen cierta duración, pero sólo se publica información sobre ellos si son muy importantes o se producen imprevistos.

1.2. Objetivos

Esta tesis intenta demostrar cómo los sistemas de RI y los sistemas de TDT mejoran si se añade una componente temporal extraída del texto automáticamente, que denominaremos *periodo de suceso*, y que representa el espacio de tiempo en el que transcurre el suceso principal relatado en cada documento. Con este propósito nos hemos marcado los siguientes objetivos:

1. Definición de un *modelo de tiempo* para representar y manipular las referencias temporales que aparecen en un texto.
2. Desarrollo de una aplicación para la extracción de expresiones temporales lingüísticas y el reconocimiento del intervalo absoluto que referencian según el calendario Gregoriano.

3. Implementación de un sistema para la extracción automática del *periodo de suceso*.

4. Mejorar los actuales sistemas de RI, y concretamente los sistemas de TDT.

El objetivo de la definición de un *modelo de tiempo*, proviene de la necesidad de formalizar las expresiones temporales que utilizamos en Lenguaje Natural. En este modelo se definen unas entidades temporales (puntos, intervalos y duraciones)y un álgebra que permite comparar y operar con las las distintas entidades temporales para poder realizar las operaciones implícitas en las expresiones temporales. Así pues, si somos capaces de reconocer cada expresión temporal y representarla en este modelo de tiempo, operando con el álgebra definida en este modelo obtendremos *periodo de suceso* implícito en el texto.

Para estudiar la información temporal presente en los documentos nos hemos planteado el diseño y desarrollo de una aplicación para la extracción de expresiones temporales. Esta aplicación analizará los documentos y cuando detecte una expresión temporal, intentará analizar si se trata de una duración o una referencia a un instante de tiempo. En este último caso la aplicación intentaría obtener el tiempo cronológico absoluto a partir de la fecha de publicación del documento o de una fecha calculada con anterioridad, según sea el punto de referencia del hablante.

Una vez tenemos la información temporal presente en cada documento, y todos los instantes temporales que se referencian en él, la siguiente cuestión es ¿cómo podemos utilizar esta información para la recuperación de documentos?. La falta de sistemas que permitan el manejo de multiples fechas, y el alto coste de ejecución que supondría una herramienta de este tipo, debido a los problemas de cálculo que supone el manejo de multiples fechas con distintos niveles de detalle, nos plantea el problema de buscar un único metadato que nos permita definir las propiedades temporales de los documentos. Como hemos visto en la introducción de este capítulo, la fecha de publicación de un documento no ayuda al usuario a recuperar sucesos ya que no refleja cuándo se produce un suceso, por ello proponemos buscar un intervalo temporal que refleje el tiempo en que transcurre el suceso principal narrado en el documento y que denominaremos *periodo de suceso*.

¿Hay algún mecanismo de obtener el periodo de suceso de modo automático?. Según nuestra propuesta, si un documento narra un suceso, la mayoría de sus referencias temporales deben describir el intervalo de tiempo en el que transcurre el suceso. Por ello proponemos la obtención del *periodo de suceso* a partir del estudio estadístico de las referencias temporales.

Con respecto al cuarto objetivo, proponemos una serie de herramientas que permiten al usuario recuperar documentos de un repositorio de documentos, para:

- Localizar sucesos. El usuario puede realizar un seguimiento completo de un suceso, asumiendo que conoce el tema principal de un suceso y cuando se ha producido, y desea ampliar información sobre él, o sea obtener todos los documentos que hablan sobre ese suceso. Como solución en el modelo TOODOR [Ara99] se permiten consultas por conceptos, para especificar los temas principales del suceso, y por periodos de sucesos para especificar cuando transcurre el suceso. Nosotros vamos a proponer mejorar el sistema TOODOR proporcionando herramientas para añadir automáticamente el tiempo de suceso a los documentos.

- Algunas veces el usuario está interesado en conocer todos los sucesos que se han producido sobre un determinado tema, ('guerras', 'terremotos', etc.). ¿Existe alguna herramienta que nos permita obtener este tipo de información?. Los sistemas de RI permiten obtener esta información, pero cuando la cantidad de documentos es muy grande el usuario no tiene tiempo de explorar toda la información. Como solución en TOODOR [Ara99] se definen consultas que permiten especificar el tema de interés y devuelven al usuario los documentos relevantes agrupados por proximidad en sus periodos de suceso. Este tipo de consultas generan lo que en TOODOR se denomina una *secuencia de crónicas*, o sea agrupaciones de documentos que hablan sobre el mismo tema y cuyos periodos de suceso cumplen ciertas propiedades temporales. Hemos encontrado ciertas limitaciones en las consultas de tipo crónica definidas en TOODOR, por lo cual vamos a proponer una nueva herramienta para el cálculo de crónicas que hemos denominado *TimExpIR*.

- Localizar todos los sucesos relatados en un repositorio. Puede ser que el usuario esté interesado en conocer los sucesos que se han producido durante un periodo de tiempo determinado, detectar todos los documentos que relatan un mismo suceso. Esta tarea se investiga en el área del TDT, dentro de los sistemas de detección de tópicos. Los sistemas de detección de tópicos agrupan los documentos que hablan sobre temas similares con fechas de publicación cercanas. Debido a ciertas limitaciones de la fecha de publicación para la ubicación temporal de los sucesos vamos a proponer mejorar los sistemas de detección utilizando como propiedad temporal el periodo de suceso.

1.3. Contribución

En el transcurso de la investigación realizada en esta Tesis, hemos realizado varias contribuciones en distintas áreas de la computación:

1. Dentro del área del *Razonamiento Temporal* se ha desarrollando un Modelo de Tiempo que permite representar las expresiones temporales de los textos, relacionarlas con el calendario Gregoriano, y operar con ellas.

2. Dentro del área de la *Extracción de la Información* se ha creado la aplicación TimeExtractor que permite etiquetar las expresiones temporales, y reconocer el tiempo absoluto que referencian estas expresiones.

3. En el área del *Topic Detection and Tracking* se ha propuesto un algoritmo simple de detección de sucesos basado en el uso del *periodo de suceso* que mejora los resultados de sistemas similares de la literatura.

4. Y en el área de la *Recuperación de la Información*, destacan dos contribuciones:

   - Se permite incluir automáticamente como elemento para la consulta y recuperación de documentos el *periodo de suceso* de un documento
   - La herramienta TimExplR, es una interfaz gráfica que permite explorar los documentos por sus propiedades temporales. Para ello hemos requerido la utilización de un sistema de RI, que soporte consultas en campos de tipo intervalo. Para el desarrollo de esta interfaz hemos optado por utilizar el sistema TOODOR [Ara99], el cual en su modelo de datos incluye un atributo que denota el *periodo de suceso* de los documentos.

1.4. Organización de la Tesis

A continuación describiremos cómo se organiza el resto de la tesis.

En el Capítulo 2, analizamos la literatura relacionada con el tema de la tesis en el área del Procesamiento del Lenguaje Natural (PLN), los sistemas de Recuperación de Información (RI), los sistemas de Clasificación, los sistemas de Extracción de Información (EI) y los sistemas de Detección y Seguimiento de Sucesos (TDT).

En el Capítulo 3 presentamos un modelo de tiempo, que permite representar las expresiones temporales relacionadas con el tiempo cronológico según el calendario Gregoriano. Este modelo está basado en el modelo de tiempo más general, también basado en granularidades temporales, definido por C. Bettini, S. Jajodia y X.S. Wang [Bet00].

En el Capítulo 4 se propone un método de extracción automático de las fechas tanto implícitas como explícitas en textos en lengua española, y se analiza la

posibilidad de trasladar este método a otros idiomas. La extracción de formatos de fecha requieren un sistema de preprocesado por búsqueda de patrones. Pero la detección de las fechas implícitas, como 'dentro de 2 semanas', requiere un análisis más profundo del texto, y por tanto la aplicación de un sistema de extracción de información, el cual mediante técnicas de procesamiento de Lenguaje Natural, permita analizar las expresiones temporales, y asignarles una fecha determinada.

En el Capítulo 5, vamos ha describir varias herramientas para la consulta de la información temporal extraída de los textos, además de comprobar cómo ésta mejora los sistemas de detección de sucesos. El análisis estadístico y de proximidad con la fecha de publicación de las expresiones temporales que aparecen en el documento nos permitirá calcular automáticamente el *periodo de suceso* del documento. En este capítulo también demostraremos que en la obtención de *tópicos* (documentos que relatan el mismo suceso y su evolución), obtenemos resultados similares utilizando la semejanza en función de la proximidad entre los *periodo de suceso* de los documentos calculados automáticamente, y utilizando la semejanza entre todas las fechas presentes en los documentos.

En el Capítulo 6 se presentan los principales logros obtenidos en este trabajo, y las líneas de investigación que han quedado abiertas.

Capítulo 2

Estado del Arte

2.1. Introducción

A partir de los años 40, gracias a la evolución de los ordenadores, comenzó a almacenarse cada vez más información en formato digital, haciéndose cada vez más notable la necesidad de sistemas para mejorar el acceso a la información almacenada, con sistemas más rápidos y eficientes. Aunque los sistemas operativos poseen comandos para la búsqueda de ficheros de texto a partir de una consulta que posee una cadena de caracteres,(ej. 'grep de Unix'), estas aplicaciones son muy lentas a la hora de procesar grandes cantidades de texto de gran longitud, y además estos comandos son programas que simplemente buscan los ficheros que contienen la secuencia de caracteres de la consulta, no reconocen la información presente en los documentos.

Uno de los primeros sistemas de procesamiento de documentos fueron los *sistemas de catálogos digitales para bibliotecas*. En estos sistemas sólo se almacenaban los datos de las fichas existentes en las bibliotecas para facilitar la búsqueda de libros, como: el título, el autor, el año de publicación, un resumen, etc. Los sistemas de catálogo digital procesaban las fichas almacenadas en ficheros de texto permitiendo la búsqueda de libros en el catálogo especificando el texto que debe poseer cierto atributo del libro. La diferencia básica de los sistemas de catálogos y los sistemas de procesamiento de documentos actuales radica principalmente en la posibilidad actual de almacenar los documentos completos y no sólo ciertos atributos. Generalmente en los documentos de un sistema de procesamiento de documentos se añade una cabecera estructurada con datos acerca de la información que contiene el documento, a los cuales se denominan *metadatos*. Generalmente si los documentos son libros, los metadatos, suelen ser los atributos de las fichas

en los catálogos. Los documentos generalmente poseen una estructura, que no siempre se halla explícita en el formato digital, pero que permite que los sistemas de procesamiento de los documentos puedan centrar sus consultas en partes del documento [Veg99].

Mientras en las *Bases de Datos Relacionales* (BDR) se busca un campo que contenga exactamente las palabras de la consulta, en los *sistemas de procesamiento de documentos* se busca que las palabras de la consulta existan en cualquier parte del texto, se realiza lo que llamaremos un *emparejamiento matching* entre cada una de las palabras de la consulta y del texto. Además, los campos de las BDR tienen un tipo de datos fijo, y en el caso de que sea texto, éste posee una longitud determinada, mientras que en los *sistemas de procesamiento de documentos* los campos de búsqueda (metadatos del documento o partes de un documento) son textos sin una longitud predeterminada.

Entre las técnicas de procesamiento de documentos caben destacar las de recuperación, *routing*, filtrado, interpretación, clasificación, creación de resúmenes o de etiquetado automático de los documentos. Cada una de estas técnicas nos permite obtener un tipo distinto de información de la colección de documentos, aunque generalmente las aplicaciones están compuestas por varios de ellos. Estas aplicaciones se pueden clasificar básicamente en dos familias: los sistemas de *Recuperación de Información (RI)* y los sistemas de *Extracción de Información (EI)*. La tarea principal de un sistema de RI consiste en buscar los documentos relevantes dentro de la colección que más se asemejen a la consulta del usuario, y devolver éstos al usuario. Si lo permite el sistema, los documentos recuperados se ordenarán según un valor de relevancia establecido por el sistema. Un sistema de EI [Leh96] procesa los documentos de una colección para extraer información estructurada específica. No intenta entender todo el documento, sino que analiza aquellas porciones de cada documento que contienen información relevante según unas pautas predefinidas.

Los *sistemas de procesamiento de documentos* suelen aplicar técnicas tanto de un sistema de RI como de un sistema de EI. Por ello, en muchos casos, es difícil clasificar una aplicación en uno de estos sistemas. Un método para su clasificación es el estudio de la funcionalidad del sistema de procesamiento de documentos [App99b]: en un sistema de RI cada fichero o documento se ve como una secuencia de posibles palabras o términos significativos, y en un sistema de EI, cada fichero contiene frases o cláusulas significativas relevantes a un tema particular.

Al hallarse los documentos escritos en *Lenguaje Natural* se intentó inicialmente, aunque con poco éxito, aplicar en los *sistemas de procesamiento de documentos* las técnicas de *Procesamiento de Lenguaje Natural (PLN)*. Actualmente estas técnicas son básicas en los sistemas de EI [Paz99], y su uso en los sistemas de RI cada vez

se está extendiendo más, ya que se pueden aplicar bien en el lenguaje de consulta del usuario, cómo en la representación y clasificación interna de los documentos.

En 1997 surgió una nueva iniciativa en el campo de los sistemas de RI, los sistemas de seguimiento y la detección de sucesos en noticias de actualidad. A esta iniciativa se le denominó *Topic Detection and Tracking* (TDT) y está muy ligada a nuestro propósito de obtener los distintas *tópicos* sobre los sucesos que se relatan en los periódicos.

En la próxima sección vamos a ver como las técnicas de PLN permiten procesar los documentos o las consultas de los documentos para mejorar los sistemas de procesamiento de documentos. En la sección 2.3 presentaremos los sistemas de RI actuales para el procesamiento de grandes colecciones de documentos, las principales técnicas de búsqueda que se utilizan, y los sistemas de evaluación de estos sistemas. En la sección 2.4 veremos los distintos algoritmos de agrupamiento de los documentos. En la sección 2.5 estudiaremos los sistemas que nos permiten obtener cierta información predeterminada de los documentos mediante un procesando automático de éstos, o sea los sistemas de EI. Y en la sección 2.6 veremos con detalle unos sistemas de RI centrados en la detección de sucesos, que se denominan sistemas de TDT. Finalizaremos el capítulo con unas conclusiones sobre las limitaciones de estos sistemas para alcanzar nuestros objetivos.

2.2. Sistemas de *Procesamiento de Lenguaje Natural*

El *Procesamiento de Lenguaje Natural* (PLN) es una técnica esencial de la *Inteligencia Artificial* que tiene como propósito el modelado y procesamiento computacional del lenguaje humano, normalmente para su comprensión. Todo sistema de PLN intenta simular el comportamiento lingüístico humano. Para ello debe tomar conciencia tanto de las estructuras propias del lenguaje, como de su universo de discurso. O sea un sistema de PLN conoce los significados de las palabras y como éstas afectan al significado global de la oración, ya que posee un conocimiento del universo de la colección de textos, los cuales contienen información de un contexto restringido.

El PLN incluye el estudio de todos los niveles del lenguaje humano, a saber:

- *Nivel morfológico-léxico.* Transforma cada secuencia de caracteres en una secuencia de unidades significativas haciendo uso de diccionarios, reglas morfológicas o bien tesauros. Busca los sufijos, prefijos, sinónimos, generalizaciones, especializaciones, etc.

- *Nivel sintáctico.* Analiza la estructura gramatical de la secuencia de unidades léxicas, y produce una representación de su estructura bien sea en forma de árbol o red. Permite extraer los distintos sintagmas de la oración.
- *Nivel semántico.* A partir de la estructura generada por el análisis sintáctico se genera otra estructura o forma lógica asociada, que representa el significado o sentido de la sentencia, independientemente del contexto.
- *Nivel del discurso o contextual* (función pragmática). Utiliza la estructura semántica del nivel anterior para desarrollar la interpretación final de la oración en función de las circunstancias del contexto.

En todos los niveles de PLN nos vamos a encontrar con el problema de la ambigüedad. En el análisis morfológico-léxico es necesario discernir entre palabras que tienen la misma raíz, a nivel sintáctico hay que unir distintos sintagmas simples en complejos, a nivel semántico se debe discernir entre los distintos significados de cada palabra, y a nivel contextual, hay que relacionar las distintas oraciones que hablan del mismo suceso [Jur00]. Aunque cabe destacar que el problema de la desambiguación del significado de las palabras, (*Word Sense Disambiguation*)es uno de los campos de investigación más importantes de los sistemas de PLN [Esc99]

El estudio del PLN comenzó en los años 40, tomando como base los autómatas finitos, pasando más tarde a la utilización de las gramáticas de contexto libre. Antes de los años 60 ya se conocía el potencial de la estructuración automática dando lugar a numerosos proyectos de investigación. La aplicación de las gramáticas de contexto libre (generadas para dominios restringidos) al procesado de documentos completos de ámbito general, para crear una forma lógica del texto a partir de la cual obtener información o clasificar documentos, se encontró con serios problemas debido a que:

- Casi ninguna gramática es adecuada para cubrir todos los textos del mundo real.
- Los textos suelen tener frases muy largas, lo que genera en su análisis problemas combinatorios casi irresolubles, incluso empleando heurísticas o guías estadísticas.
- Los sistemas resultantes son lentos debido a la explosión combinatoria inherente al problema.

A raíz de estos problemas, a partir de los 80 se vuelve a la utilización de gramáticas de estados finitos en los sistemas de PLN, de modo que se utilizan éstas junto con

las gramáticas de contexto libre [Jur00]. Church [Chu80] en su tesis en 1980 propuso la vuelta a utilización de gramáticas de estados finitos como modelo de PLN. También Ejerhed [Eje88] propuso un modelo de análisis basado en estados finitos en 1988. Abney, destacó en 1991 la importancia de la utilización de técnicas de análisis parcial (*Shallow parsing*) en los sistemas de PLN [Bew91]. Como sistema de extracción de información de textos en Lenguaje Natural podemos destacar el sistema FASTUS [App93], el cual funciona con un conjunto de autómatas de estados finitos no deterministas que operan en cascada, siguiendo las especificaciones de las primeras conferencias del MUC (Message Understanding Conference).

Los sistemas de PLN cuentan con una base de conocimiento que hace uso de distintos recursos lingüísticos, entre los que podemos destacar:

- *Bases de datos léxicas.* Listas de términos junto su significado léxico dentro de un contexto específico.
- *Diccionarios electrónicos.* Diccionarios tradicionales en formato electrónico.
- *Tesauros.* Agrupaciones de términos en clases para un determinado dominio.
- *Redes semánticas.* Representación de los términos con una estructura de red para mostrar las jerarquías conceptuales existentes entre los conceptos.
- *Matrices léxicas*: Representación en forma matricial de la relación entre cada palabra y su significado. Mediante esta representación se pueden obtener las relaciones de polisemia y sinonimia.

Estas fuentes de información permiten identificar tanto términos lexicográficamente diferentes como semánticamente equivalentes, de este modo se puede conseguir disminuir la ambigüedad que plantea el Lenguaje Natural . La utilización de Bases de Datos léxicas, diccionarios, tesauros o redes semánticas mono o plurilingües es muy útil en las técnicas de procesamiento de documentos, tanto para expandir la consulta del usuario, como para la extracción de términos de indexación o bien para la clasificación de documentos.

2.3. Sistemas de *Recuperación de Información*

Un sistema de *Recuperación de Información* (RI) es un sistema de procesamiento de documentos que trata de recuperar de una colección de documentos aquellos que se asemejan a la consulta del usuario. Un sistema de RI se encarga

tanto de la recuperación de documentos como de su almacenamiento y organización, al igual como ocurre en los *Sistema de Gestión de Bases de Datos*. Pero hay que tener en cuenta que mientras en estos sistemas, los datos están totalmente estructurados, organizados expresamente para ser recuperados por el gestor de la base de datos, en los sistemas de RI los datos pueden ser de cualquier tipo, y de cualquier longitud. En un SRI, un dato de consulta puede ser el documento completo, una parte del documento (puede ser una estructura del documento si este está estructurado o simplemente un párrafo del documento), o un metadato de la cabecera del documento, si este posee una cabecera estructurada. Así pues, en los SRI se permite consultar en un campo que representa todo el documento. Si poseen una estructura, permiten consultar en ciertos campos que representan las distintas secciones del documentos y/o si poseen una cabecera estructurada, podremos restringir la consulta a unos campos de consulta, que son metadatos del documento. En cualquiera de estos campos se pueden tener cualquier tipo de datos, no previamente especificados, y en le caso datos de tipo texto, pueden ser de cualquier longitud.

Un *sistema de recuperación de datos* [Rij79], simplemente comprueba si un dato existe en un fichero, o sea, busca documentos que casan con una palabra, mientras en los sistemas de RI se seleccionan aquellos documentos que coinciden parcialmente (búsqueda aproximada) o totalmente (búsqueda exacta) con los términos de la consulta del usuario, creándose una lista de documentos que en el caso de búsqueda aproximada se puede ordenar según un índice de relevancia entre la consulta y el documento.

El *índice de relevancia* de un documento con respecto a una consulta, se calcula con una medida de semejanza que devuelve un valor en función de los términos de la consulta que se encuentran en el documento. En algunos sistemas, incluso se tiene en cuenta las veces en que ese término aparece en el documento. Generalmente en los sistemas de búsqueda aproximada se fija un umbral de semejanza, que depende de la la colección [Swa99]. Cuando la relevancia de un documento respecto a una consulta está por debajo de este valor se supone que el documento no es relevante para el usuario, por lo que no se incluye en la lista de documentos relevantes.

Las aplicaciones directas de estos sistemas son los servicios de información: catálogos, bibliotecas digitales, buscadores Web, enciclopedias, ofimática, documentación (patentes, leyes, bibliografía), sistemas multiligües o bien sistemas de integración y distribución de noticias, etc. Indirectamente estos sistemas ayudan a la construcción de léxicos (corpus, ontologías, bases de conocimiento, diccionarios, tesauros) y a la clasificación de documentos.

La mayoría de sistemas de RI permiten que las consultas se expresen como una lista de palabras que el usuario espera encontrar en el documento y que para

él lo caracterizan. Algunos sistemas incluso permiten la utilización de operadores, principalmente booleanos. Pero a los usuarios que no son expertos, les resulta difícil expresar sus requerimientos con estos lenguajes de consulta.

La mayoría de sistemas de RI buscan por palabras o cadenas con comodines, o incluso por conceptos, en el texto o en ciertos metadatos del documento, pero pocos de ellos utilizan la estructura del documento. La inclusión de consultas por estructura requiere que el usuario conozca de antemano la estructura del documento, y además, como cada colección de documentos posee su propia estructura, las tarea de reconocimiento de las distintas estructuras posibles añade más complejidad a los sistemas de indexación y recuperación.

Aunque no existen estudios sobre la influencia del uso de la estructura en la efectividad de los sistemas de RI, parece claro que ésta debe aumentarla. En [Veg99] se demuestra cómo la inclusión de la estructura en las consultas hace que varíe la relevancia de cada documento, y por lo tanto su posición en el conjunto de documentos recuperados, de modo que aumenta la precisión.

Las conferencias TREC (Text Retrieval Conferences) están centradas en el desarrollo de los sistemas de RI. En ellas se ha comprobado que su efectividad está muy ligada a la consulta que hace el usuario, y a la elección de los términos de indexación de los documentos. Por ello en las investigaciones actuales se está trabajando con el propósito de llegar a la utilización del Lenguaje Natural como lenguaje de consulta, de modo que que se está investigando en los siguientes aspectos: la ayuda al usuario para la realización de las consultas por medio de tesauros, la expansión de consultas, y realimentación automática de consultas. La gran cantidad de documentos almacenados actualmente en los ordenadores y la longitud de éstos, son los dos grandes retos de los sistemas de RI actuales, que dan lugar a que las aplicaciones desarrolladas teóricamente y probadas en pequeñas colecciones de documentos cortos, al aplicarse a los grandes repositorios de documentos completos existentes en la actualidad, no den los resultados esperados. Esta problemática se está estudiando en las últimas conferencias TREC [Hum99].

2.3.1. Arquitectura de un sistema de RI

Los sistemas de RI tradicionales utilizan programas que gestionan ficheros invertidos, ya que éstos permiten campos textuales de cualquier longitud, e incluso documentos multimedia. Cabe destacar que las bases de datos actuales en la actualidad se pueden utilizar para gestionar documentos. Un sistema de RI generalmente se compone de [Rod99, Tur99, Bae99]:

- Una colección de documentos. Los documentos pueden ser estructurados y/o tener una cabecera estructurada con metadatos sobre el documento.
- Un lenguaje de consulta. Una consulta suele ser una lista de términos (palabras, combinación de palabras, raíces de palabras, cadenas con comodines) que pueden estar relacionados con operadores y/o ponderados según un nivel de relevancia.
- Una aplicación que se encargue de la organización, selección y presentación de documentos. Esta aplicación está a su vez compuesta de:
  - Un gestor del repositorio con un sistema de indexación de los documentos.
  - Un proceso de emparejamiento de la consulta con cada uno de los documentos, que devuelve la semejanza entre cada documento y la consulta.
  - Un proceso que organice los documentos relevantes, y los represente en una lista para que el usuario pueda extraer de esa lista los documentos que desea.

2.3.1.1. Sistema de Indexación

Generalmente los sistemas de RI indexan los documentos para su posterior búsqueda. El método de indexación para textos completos más utilizado se basa en el uso de *ficheros invertidos*. Un fichero invertido contiene como entradas todos los términos significativos de la colección de documentos, y para cada término se indica en qué documentos aparece, pudiendo incluir ciertos atributos adicionales como la frecuencia de aparición o la posición de esa palabra en el documento. Un término puede bien ser una palabra, un lema o un sintagma nominal.

Los sistemas de búsqueda avanzada, permiten realizar consultas indicando la posición de los términos en las distintas partes del documento, o sea el campo, subcampo, párrafo o línea donde aparece la palabra en el documento. Todas estás propiedades se deben añadir al *fichero invertido* [Mof98, Tom94]. Si el sistema permite además especificar la proximidad de los términos en el texto , se debe indicar en el fichero la posición relativa de cada palabra en el documento. Si el sistema permite realizar búsquedas por contenido, el índice no solo tiene por cada entrada las apariciones de esa palabra sino también ofrece las apariciones de los términos similares. Para permitir la consulta de los términos que aparecen en ciertos metadatos, en las entradas de cada término se debe indicar las distintas apariciones del término en cada metadato.

Otro sistema de indexación muy utilizado son los *ficheros de signaturas* [Lee94]. Estos son ficheros que contienen patrones de bits, denominados *signaturas*, que representan los documentos. Un estudio comparativo de los dos métodos utilizando varias colecciones de documentos [Mof98], ha demostrado que en realidad los ficheros de signaturas no mejoran los sistemas de RI, ya que los ficheros de signaturas producen más errores de emparejamiento de los esperados, y el índice puede resultar mayor y más costoso de construir y actualizar.

2.3.1.2. La estructura en los sistemas de RI

En las colecciones de documentos completos, cuando se calcula el nivel de semejanza, éste a veces no refleja lo que busca el usuario. Esto es debido a que cuando un documento es muy largo, generalmente se habla de más de un sucesos o temática. Generalmente hay un tema o suceso principal e información relacionada. Por ello se están estudiando técnicas que permitan centrar la búsqueda en pasajes, utilizando la estructura del texto. De este modo se calcula la semejanza para cada pasaje y se combinan para calcular la relevancia global.

La obtención de los pasajes depende de la estructura que tenga el documento. La cual puede ser: ser plana, solapada, jerárquica o anidada según se superpongan los distintos pasajes del texto.

La estructura de un documento será explícita si el texto posee etiquetas que limiten secciones del texto, Y en otro caso será implícita. Por ejemplo un documento HTML posee una estructura implícita ya que el estudio del tipo de letra nos permite identificar las partes lógicas del documento. En cambio, un documento XML es un documento con estructura explícita, donde las etiquetas identifican cada una de las partes del documento.

En estos sistemas se puede ampliar el lenguaje de consulta permitiendo incluir operaciones con estructuras (inclusión, posición, distancia), establecer condiciones acerca de sus antecesores, discriminar los nodos que no pertenecen al mismo pasaje, o simplemente restringir las consultas a unos campos determinados, como en una base de datos relacional.

Los sistemas de RI que utilizan la estructura utilizan distintos modelos de documentos, entre los que cabe destacar el Modelo Híbrido, los Árboles Patricia, las Listas Solapadas, las Listas de Referencia, el Árbol de Coincidencia y los Nodos Próximos [Bae99, Veg99].

2.3.2. Modelos de Recuperación de Información

Uno de los problemas más importantes en los sistemas de RI es cómo discernir entre los documentos relevantes o no. Los sistemas de RI clásicos utilizan técnicas de búsquedas booleanas y de reconocimiento de patrones. Estas técnicas llamadas de búsqueda exacta son técnicas muy restrictivas, ya que no tienen en cuenta las ambigüedades que aparecen en el Lenguaje Natural y generalmente los usuarios tienen problemas en la construcción de las consultas. Esto provoca que los usuarios no obtengan los resultados deseados, o bien no recuperan suficientes documentos, o recuperan tantos que es imposible comprobar cuales son los relevantes [Gro98].

Debido a los problemas que plantean los sistemas de búsqueda exacta se han desarrollando técnicas de búsqueda basadas en la información estadística, las denominadas técnicas de búsqueda aproximada. El sistema de búsqueda aproximada no hace un emparejamiento exacto y permite que el usuario especifique la importancia de cada uno de los términos. El sistema calcula la relevancia en función de los términos de la consulta que aparecen en el documento. Si la relevancia supera un cierto umbral, el documento se recupera, aunque alguno de los términos de la consulta no exista en el documento. La realización de las consultas resulta más compleja en estos sistemas.

2.3.2.1. Sistemas de RI de Búsqueda Exacta

En los sistemas de RI de búsqueda exacta se utiliza una correspondencia exacta entre los términos de las consultas y de los documentos. Cuando se realiza la correspondencia entre una consulta y un documento, el sistema le asigna un valor de relevancia de '1' al documento, mientras que si no se realiza emparejamiento tiene relevancia '0'. Así pues el sistema devuelve al usuario una lista con todos los documentos con relevancia '1'. En este sistema el primer documento de la lista no tiene por que ser más interesante para el usuario, simplemente es el primero que ha encontrado relacionado con la consulta. Es decir, todos los documentos de la lista tienen la misma relevancia, no se tiene en cuenta la frecuencia de aparición, ni el orden o importancia de los términos de la consulta.

Destacaremos dos modelos de sistemas de RI por búsqueda exacta:

Por Búsqueda de Patrones. Los sistemas de RI por búsqueda de patrones, utilizan las técnicas de reconocimiento de patrones. En estos sistemas la consulta puede ser una colección de palabras, cadenas con comodines o expresiones regulares. El sistema intenta buscar los documentos que contengan el patrón de la

consulta. Este sistema de búsqueda no requiere índices, puede utilizar los documentos de la colección directamente. No suele ser muy útil en colecciones grandes por ser muy lento, pero es muy útil en colecciones de documentos que se modifican frecuentemente.

Por Indexación Booleana. Los sistemas de RI basados en la indexación booleana utilizan un índice con los términos del texto, y la consulta es una expresión booleana. Las consultas booleanas tienen una semántica precisa, por lo que son fáciles de implementar debido a su claro formalismo, pero por otro lado, para el usuario suele ser complicado concretar su consulta con expresiones de este tipo. El sistema va buscando en el índice los términos de la consulta de modo que todos aquellos documentos que validen la consulta booleana serán relevantes y se mostrarán al usuario. Generalmente los índices de estos sistemas se almacenan en ficheros invertidos.

	t_1	t_2	t_3	...	t_j	...	t_m
d_1	0	0	1	...	1	...	1
d	0	0	1	...	0	...	1
d_i	0	1	1	...	1	...	0
d	0	0	1	...	0	...	1
d_n	0	1	1	...	0	...	1

Figura 2.1: Matriz de términos-documentos.

Para mejorar la rapidez de estos sistemas se suelen representar tanto los documentos como las consultas, mediante un vector booleano que se denomina *vector de términos* $d_k = (w_{1,k}, \ldots, w_{m,k}, w_{m,k})$, donde m es el número de términos de la colección y $w_{j,k}$ representa la existencia del termino t_j en el documento d_k. Si $w_{j,k}$ tiene el valor '1' significa que el término t_j está presente, y si tiene el valor '0', ese término no existe. Así pues la colección de documentos se puede representar mediante una *matriz de términos*, según se muestra en la figura 2.1. Esta matriz posee una dimensión de m términos y n documentos, donde cada fila contiene el vector de términos de un documento. Con esta representación de la consulta y la colección de documentos, la búsqueda de los documentos relevantes a una consulta, consiste en buscar en la matriz de términos, un vector compatible con el vector de términos de la consulta.

2.3.2.2. Búsquedas Aproximadas

Los estudios sobre el modelo booleano dieron lugar a que estos modelos se ampliaran para no desestimar un documento porque en él no aparecieran todos

los términos de la consulta. A partir del modelo booleano aparecieron distintos modelos de búsqueda aproximada, en función de cómo se asocia la relevancia $w_{i,j}$ a cada término t_j del documento d_i, pudiendo este tomar cualquier valor real mayor o igual a cero. Sólo toma el valor cero en el caso de que no exista ninguno de los términos de la consulta. Para acotar la cantidad de documentos que se recuperan, se debe establecer experimentalmente un valor umbral a partir del cual se reconozca el documento como relevante para la consulta.

En la siguiente sección veremos con detalle el *modelo vectorial*, que es el más utilizado en los sistemas de RI, a continuación describimos otros modelos de búsqueda aproximada.

Búsquedas Probabilísticas. Este modelo utiliza la teoría de la probabilidad para construir la función de búsqueda. Este modelo es similar al vectorial, pero a cada término t_j del documento d_i, en vez de asociarle la frecuencia de aparición de cada término, se le asocia como peso $w_{i,j}$ una medida en función de la probabilidad de que ese término aparezca en un documento relevante $P(R|t_j, d_i)$ o no $P(\bar{R}|t_j, d_i)$. Así, la semejanza entre una consulta y un documento se calcula como una combinación de dos probabilidades, la probabilidad de que el término aparezca en la consulta y la probabilidad de que el término no aparezca en la consulta. Dichas probabilidades se obtienen de los valores de la distribución de los términos indexados a lo largo de la colección, o de un subconjunto de ella.

Redes de Inferencia Bayesiana. Modelo que utiliza una red bayesiana para representar los documentos. La red bayesiana es un grafo dirigido acíclico en el cual los nodos representan variables aleatorias y los arcos representan las relaciones de causalidad entre estas variables y donde la fuerza de inferencia entre las relaciones se representa mediante una probabilidad condicional. Los nodos de la red en un sistema de RI representan a los documentos, a los términos de los documentos y a los términos de las consultas a distintos niveles. En esta red, los nodos raíz son los documentos y el nodo terminal es la consulta, de modo que los nodos hijos de los documentos son los términos que aparecen en un documento, los cuales se unen con los términos de la consulta que son los nodos padres de la consulta. Los arcos unen cada documento con sus términos representan la probabilidad condicional de que el término aparezca en el documento. La relevancia de un documento se mide como la cantidad de apoyo evidencial que la observación del documento d_j da a la consulta q.

Redes Neuronales. En este modelo se utiliza una red neuronal para representar los documentos, de forma que cada nodo de la red es un documento o un

término. La consulta activa una secuencia de neuronas que disparan enlaces a los documentos. La fuerza de cada enlace en la red se transmite al documento de forma que se establece un coeficiente de semejanza entre la consulta y el documento. La red es entrenada ajustando los pesos de los enlaces, en base a las respuestas de un conjunto de documentos de entrenamiento preestablecidos como relevantes o irrelevantes.

Algoritmos Genéticos. Se utilizan algoritmos genéticos para obtener una consulta óptima por evolución de una consulta inicial. La consulta inicial se utiliza junto con un peso estimado o establecido aleatoriamente, de forma que se generan nuevas consultas modificando esos pesos. Una nueva consulta sobrevive si está próxima a un documento reconocido como relevante, mientras que las consultas que obtienen menos documentos relevantes se borran en las siguientes generaciones de consultas.

Recuperación por Conjuntos Borrosos. Los términos que aparecen en los documentos o las consultas se agrupan en conjuntos borrosos, donde cada conjunto borroso contiene un término del documento junto con aquellas palabras semánticamente relacionadas. Un conjunto borroso se caracteriza por una función de pertenencia que asocia a cada elemento un valor entre 0 y 1 . El nivel de semejanza entre documentos se calcula mediante operaciones de unión, intersección y complemento entre los conjuntos borrosos de los términos de la consulta y del documento.

2.3.2.3. El modelo vectorial

El modelo vectorial es muy utilizado en los sistemas de RI de búsqueda aproximada y está basado en el modelo booleano, pero mejorado, de modo que se asigna a cada término de la consulta un peso w_j que puede ser cualquier valor positivo (binario, entero o real). De este modo ahora la representación lógica de los documentos no es un vector booleano, sino un vector de pesos, donde $w_{i,j}$ indica el grado de relevancia de que el término t_j esté presente en el documento d_i. Este peso suele estar relacionado con la frecuencia de aparición del término.

Estos sistemas permiten añadir a los términos de las consultas distintos pesos en función de lo relevante que sea cada término de la consulta para el usuario. Así los documentos se representarán mediante una matriz de frecuencias de términos, según se muestra en la figura 2.2, y una consulta se representará de la misma forma $q_k = (w_{k,1}, \ldots, w_{k,j}, w_{k,m})$.

El modelo vectorial hace la suposición básica de que la proximidad relativa entre dos vectores es proporcional a la distancia semántica de los documentos. En

	t_1	t_2	t_3	...	t_j	...	t_m
d_1	w_{11}	w_{12}	w_{13}	...	w_{1j}	...	w_{1m}
d_2	w_{21}	w_{22}	w_{23}	...	w_{2i}	...	w_{2m}
..	..	..	..	..	..	..	..
d_i	w_{i1}	w_{i2}	$w_{i3}1$	...	w_{ij}	...	w_{im}
..	..	..	..	..	..	..	..
d_n	w_{n1}	w_{n2}	w_{n3}	...	w_{nj}	...	w_{nm}

Figura 2.2: Matriz de frecuencias de términos.

la figura 2.3 [Sal98] se muestran las distancias más utilizadas como medidas de semejanza en los sistemas de RI vectoriales.
En este modelo se proponen las siguientes propiedades para los términos:

- tf_{ij}: es la frecuencia de aparición del término t_j en el documento d_i.
- df_j: indica el número de documentos en los que aparece el término t_j.
- $dv_j = Q - Q_j$: es el poder discriminador de un término j. Indica la capacidad de disminuir la semejanza entre dos documentos de un término t_j. Q es la densidad de semejanza sin seleccionar el término t_j y Q_j es la densidad de semejanza al considerar el término t_j.

A partir de éstas, el peso $w_{i,j}$ se calcula generalmente según una de las siguientes funciones:

- $w_{i,j} = tf_{i,j} \cdot idf_j$, donde idf es la función inversa de df.
- $w_{i,j} = tf_{i,j} \cdot dv_j$, tomando $Q = \frac{1}{N \cdot (N-1)} \sum_i \sum_{j(j \neq i)} sim(d_i, d_j)$.
- $w_{i,j} = tf_{i,j} \cdot dv_j$ tomando $Q = \frac{1}{N} \sum_i sim(C, d_i)$ y donde C es el centroide de la colección de los documentos.

Así pues, si se utiliza la medida del coseno, se define la semejanza entre un documento d_j y la consulta q_k, siendo m el número de términos como:

$$sim(d_j, d_k) = \frac{\sum_{i=1}^{m} w_{j,i} \cdot w_{k,i}}{\sqrt{\sum_{i=1}^{m} w_{j,i}^2 \cdot \sum_{i=1}^{m} w_{k,i}^2}} \tag{2.1}$$

Medida de Similitud	*Modelo Booleano*	*Modelo Vectorial*
Producto escalar	$\|X \cap Y\|$	$\sum_{j=1}^{m} X_j \cdot Y_j$
Coeficiente de Dice	$\frac{2 \cdot \|X \cap Y\|}{\|X\| + \|Y\|}$	$\frac{2 \cdot \sum_{j=1}^{m} X_j \cdot Y_j}{\sum_{j=1}^{m} X_j^2 + \sum_{j=1}^{m} Y_j^2}$
Coseno	$\frac{\|X \cap Y\|}{\sqrt{\|X\|} \cdot \sqrt{\|Y\|}}$	$\frac{\sum_{j=1}^{m} X_j \cdot Y_j}{\sqrt{\sum_{j=1}^{m} X_j^2 \cdot \sum_{j=1}^{m} Y_j^2}}$
Coeficiente de Jaccard	$\frac{\|X \cap Y\|}{\|X\| + \|Y\| - \|X \cap Y\|}$	$\frac{\sum_{j=1}^{m} X_j \cdot Y_j}{\sum_{j=1}^{m} X_j^2 + \sum_{j=1}^{m} Y_j^2 - \sum_{j=1}^{m} X_j \cdot Y_j}$

Figura 2.3: Distancias entre dos vectores de términos.

2.3.3. Evaluación de los sistemas de RI

Los sistemas de RI se pueden evaluar en función de distintos parámetros [Sal98, Bae99, Jur00] siendo los más utilizados la *Precisión* (*Precision*) y la *Cobertura* (*Recall*).

La Precisión. Refleja la exactitud del resultado obtenido para una consulta, la proporción de los documentos recuperados relevantes, en relación a todos los obtenidos. Un sistema que recupera más información incorrecta que correcta trabaja con una menor precisión que aquel que no genera información incorrecta.

La Cobertura. Refleja la proporción de documentos relevantes obtenidos. Esta medida muestra la cantidad de documentos correctos y relevantes devueltos por un sistema respecto a la cantidad total de documentos relevantes y correctos presentes en la colección.

El F-measure. Es una medida que combina R y P para evaluar los sistemas de RI del siguiente modo:

Sean:

N_R= el número documentos recuperados relevantes
N_B= el número documentos relevantes de la colección y,
N_E= el número documentos recuperados
Se define:

$$\mathbf{P} = \frac{N_R}{N_E}$$

$$\mathbf{R} = \frac{N_R}{N_B}$$

$$\mathbf{F - measure} = \frac{(\beta^2 + 1) \cdot P \cdot R}{(\beta^2 \cdot P + R)}$$

Un F-measure alto indica una buena eficacia global de recuperación de documentos relevantes.

Existen otras medidas para evaluar otros aspectos de un sistema de RI, a saber:

Fallout. Refleja la tendencia de un sistema a asignar datos incorrectos.

Overgeneration. Representa la cantidad de información irrelevante que es generada por el sistema, o sea los documentos recuperados que no son relevantes para la consulta realizada.

Tiempo de Respuesta. Tiempo promedio que requiere el sistema de RI para responder a una consulta.

Eficiencia de almacenamiento. Calcula la sobrecarga de espacio como la relación entre el tamaño de los índices y documentos con respecto al tamaño de la colección.

2.3.4. Evolución de los sistemas de RI

A finales de los años 50, **Luhn** [Rij79] fue el primero en aplicar la indexación automática por contenido en los documentos, destacando en sus experimentos que la frecuencia de aparición de las palabras en el cuerpo del documento podría ser usada para indicar el grado de relevancia, definiendo así, un método sencillo de asignación de pesos para las palabras clave.

El uso de información estadística acerca de la distribución de palabras en documentos fue más ampliamente explotado por **Maron, Kuhns y Stiles** en la década de los 60, demostrando que se mejora el número de documentos recuperados realizando asociaciones estadísticas entre palabras clave. **Good y Fairtone**, en 1960, sugieren que la clasificación automática de documentos es muy útil en la recuperación de información. Aunque no fue hasta varios años más tarde, en 1965, cuando **Doyle y Rocchio** realizaron los primeros experimentos. **Salton y Yang** en los años 60, llegaron a la conclusión de que un término que está con mucha frecuencia en los documentos no es útil para la recuperación de textos, de modo que

los términos de indexación con una frecuencia media son los más representativos de los documentos, proponiendo la utilización de una *lista de palabras de parada* para mejorar los sistemas de RI.

Doyle en 1965 argumentó que los principios subyacentes a la IR basada en la asociación de términos podrían aplicarse tanto si las asociaciones se determinaban por hombres como por máquinas. Pero fueron **Grouch y Yang** los primeros que generaron automáticamente un tesauro a partir de palabras clave en textos, las cuales podrían usarse para indexar documentos y realizar consultas. El enfoque de **Grouch** se basa en el modelo vectorial de **Salton** y la teoría de discriminación de términos.

Bely, Borillo, Virbel y Siot-Decauville, a principios de los años 70, realizaron el primer tesauro automático, a partir de resúmenes de documentos. Identificaron referencias a conceptos de tesauros y relaciones entre los conceptos.

En 1971 **Sparc Jones** llevó a término un trabajo de recuperación de información utilizando medidas de asociación entre palabras clave (conceptos que extraía de un tesauro) basándose en su frecuencia de co-ocurrencia(frecuencia de aparición de las dos palabras juntas en el mismo documento [App99]). Demostró que si existen N documentos y un término de indexación aparece n veces en ellos entonces a ese término de indexación se le debería dar un peso proporcional a $log(N/n) + 1$ para conseguir una mayor efectividad de recuperación.

El concepto de *realimentación* en recuperación de información fue debido a **Rochio** en 1971. En 1984 **Spark Jones y Tait** analizaron las consultas y para mejorar los sistemas de RI mediante la *Expansión de Consultas*, de modo que buscaban todas las variaciones de la consulta original que expresaban la misma necesidad. En 1987 **Fagan** comparó los sistemas de RI utilizando frases generadas por un sistema de PLN y combinaciones de palabras, concluyendo que el coste del sistema de PLN era demasiado caro para la poca mejora obtenida en los sistemas.

A partir de los 90, los sistemas de RI dejaron de trabajar con colecciones pequeñas de textos cortos y resúmenes, para plantearse el problema de la grandes colecciones de documentos completos que se disponía debido principalmente al auge de la WWW. La WWW añadió varios problemas a los sistemas de RI como los hiperenlaces, la falta de una estructura bien definida de los documentos, y la longitud que los documentos pueden alcanzar. Esto dio lugar a un nuevo planteamiento, ya que en los textos cortos o resúmenes, la característica más importante son los textos en sí, por lo que los sistemas de PLN no habían sido muy efectivos. Pero en las colecciones de texto completos no podemos descartar la utilidad de estas técnicas.

En los últimos años se ha comprobado que por mucho que han mejorado las técnicas de indexación, y se han propuesto numerosas medidas de semejanza, no se han logrado con estos sistemas la *Cobertura* y *Precisión* esperados. La razón de esto se debe a que se ha pasado de aplicar unas técnicas que en principio iban destinadas a bibliotecarios y expertos en la información, a tener cualquier usuario, sin conocimiento previo de los datos de la colección que va a consultar. Por esta razón las consultas que éste realiza no permiten obtener una respuesta con la efectividad esperada.

En la actualidad se están buscando nuevas técnicas que mejoren la efectividad de los sistemas de RI. Para mejorar los sistemas de RI en las colecciones grandes de documentos largos, se están aplicando técnicas de agrupamiento de documentos y expansión de consultas basándose en técnicas de PLN [Per00]. Sin embargo en ambos casos, no se obtienen los resultados teóricos esperados, debido principalmente a que se han generalizado los conocimientos de las técnicas Lenguaje Natural, que son ámbito restringido, a colecciones generales que tratan diversos temas no conocidos totalmente a priori.

En el esfuerzo de incentivar y canalizar las investigaciones en el campo de procesamiento de documentos se creó el proyecto TIPSTER [Yan01] financiado por el DARPA (Defense Advanced Research Projects Agency) en 1989. El objetivo central de este proyecto, que finalizó en 1998, fue establecer el estado del arte en los campos de recuperación de textos, construcción de resúmenes (capacidad de condensar el tamaño de los documentos o la colección de documentos manteniendo solo las ideas clave) y extracción de información (capacidad de localizar información específica dentro de un texto). Estas tres áreas de TIPSTER dieron pie a tres conferencias en 1998, las cuales todavía se celebran en la actualidad :

- TREC (Text Retrieval Conferences), centradas en aplicaciones de RI.
- MUC (Message Understanding Conference), centradas en aplicaciones de EI .
- SUMMAC (Summarization Conference), centradas en aplicaciones de creación automática de resúmenes.

Las conferencias TREC [Har00, Hum99, Buc00, Spr00], comenzaron en 1991, como una serie de talleres diseñados para alentar la investigación en recuperación de textos mediante aplicaciones reales, proveyéndolas de una gran colección de textos para testear las aplicaciones, así como de procedimientos de evaluación uniformes, además de ser un foro para que las organizaciones puedan comparar sus resultados.

En un principio en estas conferencias se estudiaban y evaluaban distintos métodos de recuperar documentos basándose en métodos estadísticos. Pero en 1996 (TREC-5) se percataron de que el principal problema de los sistemas de RI era la incapacidad del sistema para reconocer en los documentos recuperados, la presencia o ausencia de conceptos consultados en la pregunta, debido principalmente a la poca información de las consultas. Por ello decidieron insertar una nueva tarea de estudio: la construcción automática de consultas para poder posteriormente encontrar un método para la expansión de la consulta. En el TREC-7 se añadió la tarea de conseguir un proceso de búsqueda que intentara responder las consultas de los usuarios frente a representaciones basadas en términos de los documentos, determinando el nivel de relevancia entre los dos a partir del número y tipo de términos coincidentes. Fue a partir de 1998 cuando se percataron de que la indexación de palabras, la utilización de técnicas de agrupamiento y la asignación de pesos, no daban lugar a los resultados esperados. Parece ser que esto se debe a la aleatoriedad a la hora de seleccionar los términos que caracterizan los documentos y al sistema de asignación de pesos. La solución a estos problemas parece que se podría resolver de diversas formas:

- aplicando herramientas que permitan al usuario formular correctamente la pregunta, o bien métodos que permitan expandir la consulta del usuario.
- aplicando técnicas de EI que permitan mejorar los resultados, indexando no solo palabras sino también los conceptos relacionados, grupos de palabras que tratan un mismo concepto, y reconociendo nombres propios de personas, lugares, entidades, etc.

Como resultados del TREC-7(1998) se observó que la aproximación de utilizar palabras, frases o bien pasajes para expandir la especificación de búsqueda inicial era muy efectiva, mientras que la aplicación de un análisis sintáctico para la obtención de términos compuestos relevantes del texto no mejora los sistemas de RI. Esto último podría deberse bien a que no es suficiente el análisis sintáctico, o bien a que las predicciones semánticas hechas a partir de estructuras sintácticas (núcleo+modificador) no son buenas.

En la conferencia TREC-8 (1999) [Hum99] se introdujo una nueva tarea: *los sistemas de construcción de respuestas a preguntas* **(Q & A systems)**. La esencia de esta tarea es que el sistema encuentre una cadena con la respuesta literal a una pregunta simple dentro de la colección de documentos. Esta tarea ya se planteó en el área de la IA en 1986 con este mismo nombre, aunque TREC-8 se distingue de estos trabajos en que la pregunta a responder es potencialmente una pregunta no restringida en Lenguaje Natural, y además la información se encuentra en una colección de textos no estructurados.

Cabe destacar que a partir de las conclusiones de las primeras conferencias TREC, se han realizado varios sistemas de RI muy potentes como son: SMART [Buc00] y INQUERY [Bro94].

2.3.5. El *procesamiento del Lenguaje Natural* en RI

En 1999 comenzó una discusión sobre las mejoras de la utilización de las técnicas de PLN en los sistemas de RI [Spr00, Per00, Sme98, Har00, Hum99]. En la actualidad la mayoría de investigadores están a favor de que la contribución de las técnicas avanzadas de PLN es en realidad pequeña y además la efectividad de los sistemas de RI está estrechamente relacionada con la formulación de las consultas. Si una consulta no está muy bien formulada, los errores que producen los sistemas de PLN a menudo pueden empeorar la eficacia de los sistemas de RI. Pero ello no significa que los sistemas de PLN no sean útiles en RI, muy al contrario, las técnicas básicas de PLN, como la extracción de raíces, o técnicas más avanzadas como detección de multi-términos y nombres propios, detección de relaciones de sinonimia y expansión de consultas juegan un papel muy importante en los sistemas de RI.

Distintos estudios dentro de las conferencias TREC [Hum99, Spr00, Per00] han probado que los sistemas son más eficaces si se realiza un preprocesado de cada término de indexación, de modo que se extraigan de él prefijos y sufijos, o bien se obtengan sus sinónimos. Este preprocesado además permite minimizar el tamaño del índice. La obtención de raíces o sinónimos, mejoran los sistemas ya que permiten recuperar documentos con variaciones de los términos de la consulta. Por ejemplo si buscamos el término 'taxonomía', recuperaremos documentos que contengan las siguientes palabras: 'taxonomías', 'taxonomía', 'clasificación', 'catalogación', 'encasillamiento (*routing*)', etc. También se aumenta la precisión al añadir como términos de indexación combinaciones muy usuales de palabras, como por ejemplo 'Nueva York' o 'Fuerzas Armadas'.

Algunos experimentos que han utilizado un buen tesauro para reemplazar las palabras por sus significados han demostrado que la efectividad del sistema de RI disminuye, en vez de aumentar como cabría de esperar.

De ahí que surja la pregunta ¿Es realmente la ambigüedad un problema real?. Parece ser que en el caso de sistemas de RI que utilizan el texto completo, si se utilizan consultas de una longitud moderada no hay problemas de ambigüedad, normalmente la ambigüedad del significado de una palabra se elimina considerando otra palabra. Lo que se ha probado es que el PLN es bueno para el problema de la desambiguación, pero las consultas son difíciles de procesar si se tienen en

cuenta todos los sinónimos, y los emparejamientos de palabras, lo que produce una explosión lingüística.

Las líneas de investigación del PLN aplicado a los sistemas de RI son básicamente las siguientes:

- la interacción basada en el significado (búsqueda conceptual),
- respuesta a preguntas concretas (no búsqueda de documentos),
- creación automática de resúmenes como respuesta a las consultas,
- integración de información,
- creación de consultas altamente descriptivas, precisas y elaboradas,
- multilingüísmo .

Las implementaciones lingüísticas básicas provenientes de los sistemas de PLN para mejorar los sistemas de RI son:

- Segmentación del texto en vocablos (Tokenizing). En algunos idiomas esta tarea es muy sencilla ya que existen separadores entre las palabras. Pero en idiomas como el japonés o el chino, para extraer las palabras del texto se requiere la ayuda de un diccionario y la utilización de patrones.

- Extracción de raíces. Se suele utilizar para unificar los términos con variantes morfológicas. La falta de un dominio específico o del conocimiento del contexto da lugar a que estos sistemas provoquen fallos en la recuperación (varias palabras con significados distintos a veces tienen la misma raíz). También hay que tener en cuenta que si para todas las variantes morfológicas de un verbo tomamos la misma raíz, no podremos posteriormente en un análisis sintáctico diferenciar entre verbos nominales y verbos con funciones verbales. Es por tanto, recomendable que a los verbos nominales se les añada alguna terminación estándar a la raíz (ej.: *almacenamiento* convertirlo en *almacén*).

- Utilización de listas de palabras de parada. El estudio de los sistemas de RI ha demostrado que la presencia de las palabras con poco valor semántico o demasiado frecuentes influyen poco en su efectividad. Por ello los indexadores de los sistemas de RI, poseen colecciones de palabras de este tipo, de modo que cuando una palabra pertenece a esta lista no se indexa. Con ello se reduce el espacio de búsqueda, y el tamaño del índice. Esto puede provocar problemas en la resolución de algunas consultas como 'Vitamina A', pero en resultados experimentales se ha probado que los casos en que ocurre esto son muy poco frecuentes.

Algunos sistemas utilizan algunas implementaciones de técnicas de PLN [Jur00, Sme98] un poco más complejas como son:

- Identificación de frases. El objetivo es utilizar frases como unidades de indexación. Estas aplicaciones derivan del trabajo en el área de cohesión textual (*lexical chain*). La creación de frases como unidades, se basa en presuponer que algo ha pasado con anterioridad en las frases precedentes, de modo que cuando hay elementos cohesivos entre varias sentencias se forma una unidad. La captura de frases (secuencias de palabras próximas relacionadas en el texto) pueden proporcionar un contexto para la resolución de términos ambiguos, y así permitir identificar el concepto que el término representa. La identificación de frases hacen que las palabras generales puedan tener unos significados más específicos. La reiteración es una forma de cohesión léxica que involucra la repetición de un término léxico, mediante repetición de las palabras o uso de sinónimos. La identificación de frases se puede realizar con los siguientes métodos [Sme98]:

  - Por aproximación estadística, tomando todos los pares frecuentes de términos adyacentes. El problema es que se añaden muchas sentencias sin sentido.
  - Mediante técnicas de PLN como:
    - *Runs of words*: tomando aquellas frases que se encuentran entre dos palabras de la lista de palabras de parada, y que aparecen en el documento más de n veces.
    - Análisis profundo o superficial del documento con el propósito de extraer frases nominales, verbales o bien preposicionales
    - Uso de modelos estadísticos para la obtención de frases, bien por métodos heurísticos, modelos de Markov o el algoritmo de Viterbi.

- Identificación de nombres de entidades. Estos puede ayudar a identificar nombres propios, nombres de lugar y organizaciones. Para ello se aplican técnicas de análisis de patrones, a partir de la aplicación de reglas (manuales o creadas mediante un sistema de aprendizaje) o bien mediante modelos de Markov (que requieren un conjunto de entrenamiento de documentos etiquetados).

- Extracción de conceptos. Es una versión más general de la extracción de nombres de entidades. Se intentan identificar nombres de ciudades, países, provincias, títulos, fechas, monedas, porcentajes, nombres químicos, etc. La obtención de esta información es similar a la de nombres de entidades, el problema a resolver es determinar cuándo se deben utilizar los conceptos y cuándo un sistema de RI los requiere. Otra cuestión por resolver, es la normalización de conceptos y los problemas en su detección. Por ejemplo, si se desean extraer porcentajes, hay que tener en cuenta sus posibles representaciones '95 %', '0.95', '95 por ciento'. Esta tarea es básica en los sistemas de EI, y se intenta aplicar a los sistemas de RI para mejorar su eficacia. La extracción de conceptos requiere técnicas de:

- Desambiguación del significado de las palabras.
- Adquisiciones léxicas.
- *Part of speech.*
- Análisis de sentencias.
- Expansión de sinónimos.
- Resolución de anáforas y co-referencias.

De los experimentos realizados se concluye que la aplicación de técnicas de PLN básicas, así como las de identificación de frases y entidades, mejoran la efectividad de los sistemas de RI. Estas técnicas permiten que tanto en la indexación como en las consultas, se tengan en cuenta las variaciones morfológicas, la polisemia, las relaciones semánticas de sinonimia e hiper/hiponimia, además de reconocer términos multi-palabras, dependencias terminológicas, colocaciones, agrupamientos, etc.

2.4. Sistemas de Clasificación de Documentos

En la mayoría de los sistemas de RI se realiza una clasificación de documentos para limitar el espacio de búsqueda aumentando la rapidez del sistema. Su éxito radica en que evita los documentos no relevantes antes de comenzar la búsqueda, y así se reduce el número de búsquedas ha realizar. Si hay n documentos y x clases, el número medio de búsquedas a realizar es $x + (n/x)$. En estos sistemas la búsqueda de documentos relevantes se realiza en dos etapas, primero se busca la consulta en la clase, y luego se buscan los documentos relevantes en esa clase. El inconveniente de estos sistemas es el requerimiento de un mayor tiempo para el procesado, pero este coste se amortiza si la colección es grande, y no se requieren muchas actualizaciones.

Los sistemas de clasificación se dividen en dos grandes grupos, los *sistemas de clasificación supervisados* y los *sistemas de clasificación no supervisados* [Bae99, Gre00, Man99, Pons01, Rij79]. En los sistemas de clasificación supervisados, también denominados *sistemas de clasificación*, existe un universo de objetos a clasificar, un conjunto de clases posibles, y un conjunto de objetos de entrenamiento ya etiquetados con la clase a la cual pertenecen.

En los sistemas de clasificación no supervisada, denominados también *sistemas de agrupamiento* (*Clustering Systems*), no se conoce a priori las clases en las que se ha de dividir el conjunto de objetos. Los documentos se agrupan si su medida de semejanza supera cierto umbral. Estos sistemas también se utilizan para analizar estructuras y relaciones en los datos. Los sistemas de agrupamiento se pueden

clasificar a su vez en *restringidos* (*no restringidos*) si se especifica (o no) el número de clases que debe obtener el clasificador.

Los algoritmos de clasificación en los sistemas de RI se pueden utilizar para:

- Mejorar el ranking de los documentos dada una consulta.
- Mejorar la representación de los documentos.
- Ayudar al usuario en la realización de la consulta, por ejemplo ofreciéndole consultas similares que comparten el mismo URL [Bee00].
- Ayudar a descubrir conceptos y relaciones desconocidas. Bien para reconocer tópicos como se hace en TDT [TDT, TDT98], bien para la creación semi-automática de taxonomías, aplicación que se utiliza por ejemplo en el buscador Yahoo, o para explorar los documentos por grupos (aplicación para la que se desarrolló el algoritmo de clustering Scatter/Gatther [Cut92]).
- Mejorar la precisión y eficiencia de los sistemas de categorización.

Los problemas relacionados con las implementaciones actuales de agrupamiento de documentos son:

- Qué representación de documentos utilizar.
- Cómo incorporar el dominio de conocimiento.
- Qué estrategias de comparación de resultados son útiles en los agrupamientos.
- Cuántos parámetros requiere el sistema.
- Cómo son de estables los parámetros obtenidos automáticamente.
- Cómo medir la efectividad.

En las siguientes secciones veremos algunas soluciones propuestas en la literatura.

2.4.1. Sistemas de Clasificación Supervisada

El problema a resolver en los de sistemas de clasificación supervisada y los sistemas de RI es semejante, buscar aquellos documentos que validan una consulta. La diferencia básica estriba en que la consulta a realizar en los sistemas de clasificación se conoce a priori, mientras en los sistemas de RI, lo que se conoce a priori son los documentos a procesar [Gre00]. Por tanto en los sistemas de RI se realiza un preprocesamiento de los documentos, y en los sistemas de clasificación de las consultas.

Entre los sistemas de clasificación supervisada destacaremos:

Sistema de Categorización. La categorización, es uno de los primeros problemas estudiados, donde se pretende clasificar los documentos con respecto a un conjunto de clases pre-existentes, donde la colección de documentos se va incrementando a lo largo del tiempo. Cada vez que se añade un documento a la colección, éste se asocia a una de las clases de interés, y si no pertenece a ninguna se desecha.

Sistema de Encasillamiento. Es un sistema de categorización donde sólo existe una clase de interés. Generalmente estos sistemas no parten de una gran colección de documentos sino de un conjunto de entrenamiento.

Sistemas de Filtrado. Se diferencia del encasillamiento por la metodología específica de evaluación, ya que solo se encarga de ver si un documento es o no relevante para realizar una acción con él, aunque el problema de clasificación es el mismo. El filtrado puede ser tanto positivo como negativo, o sea que se puede seleccionar bien los documentos que pertenece a una clase o bien los que no pertenecen a esa clase. Como ejemplos de aplicación tenemos el filtrado de los mensajes electrónicos o bien el envío de noticias a los usuarios interesados.

Sistemas de Seguimiento. Simula un entorno de clasificación on-line. Esta tarea es como el encasillamiento, donde se incluye un paso de retroalimentación del sistema de modo que se etiqueta ese documento como bueno para el tópico de búsqueda, y se utiliza para reformular preguntas.

La dificultad principal en los sistemas de clasificación es la asignación de la medida de semejanza, de modo que esta medida indique el grado con que el documento satisface la consulta. Para mejorar la efectividad de los sistemas de clasificación se han utilizado entre otras técnicas de aprendizaje, optimización heurística, y técnicas de expansión de consultas.

2.4.2. Sistemas Agrupamiento *(clustering)*

Los sistemas agrupamiento de documentos tienen la tarea de clasificar los documentos según sus propiedades intrínsecas en varios grupos (*clusters*). Mientras que en los sistemas de clasificación los documentos se clasifican según la semejanza o relevancia con ciertas clases previamente especificadas, en el agrupamiento se trata de buscar características que permitan separar los documentos en grupos basándonos en las propiedades internas de la colección. Idealmente los grupos deben estar completamente separados, pero algunas veces el solapamiento entre grupos es inevitable. El correcto funcionamiento de estos sistemas depende de las propiedades

estadísticas de la colección. Generalmente estos sistemas se aplican en colecciones estáticas, aunque también se puede aplicar a colecciones que se incrementan en el tiempo.

En los algoritmos de agrupamiento se debe especificar: la medida de semejanza utilizada, el criterio de agrupamiento, y el algoritmo de agrupamiento. Los algoritmos de agrupamiento se clasifican en dos grandes subgrupos según se muestra en la figura 2.4: los algoritmos **jerárquicos** y los **no jerárquicos** [Yan01, Rij79, Man99].

- Los *algoritmos no jerárquicos* generan una partición del conjunto de documentos en un conjunto de grupos sin relaciones jerárquicas entre los grupos, utilizando distintas heurísticas. Entre estos algoritmos destacaremos los siguientes algoritmos:

  Single-pass. Cada vez que llega un documento se compara con todos los grupos generados hasta el momento. Si ningún grupo se asemeja al documento, entonces el documento forma un nuevo grupo. Produce grupos dependientes del orden de procesado [Yan00].

  Relocalización (*reallocation*). Se constituye un conjunto de N grupos aleatoriamente, e interactivamente se van re-asignando los componentes en esos N grupos hasta encontrar la mejor agrupación. En estos métodos generalmente no es posible determinar cuándo se ha encontrado la solución óptima.

  Vecino más próximo (*k-means*). Este método asigna a un mismo grupo aquellos vecinos más semejantes. Ya que la semejanza se mide en función de la distancia, si son más semejantes la distancia es menor, por lo cual se hallan más próximos. El usuario define el número de vecinos a considerar k y el nivel de semejanza entre los vecinos de la lista. El algoritmo selecciona aleatoriamente un conjunto de k centroides, y se asigna a cada documento el centroide más cercano de modo que se obtienen k-grupos. A partir de estos k grupos se recalculan iterativamente los centroides de esos nuevos grupos, y se asignan de nuevo los documentos a los centroides más próximos hasta que se estabiliza el cálculo de los centroides de los grupos [Yan00].

- Los *algoritmos jerárquicos* calculan la semejanza entre todos los pares de grupos y producen una secuencia anidada de particiones. La ventaja de estos algoritmos viene dada por la posibilidad de utilizar las técnicas de búsquedas en árboles para la resolución de consultas. Uno de los problemas de los algoritmos jerárquicos es la elección del número de grupos a obtener, ya que cuando la colección es grande (100 objetos), la representación utilizando jerarquías no suele ser muy apropiada. Estos algoritmos a su vez se clasifican

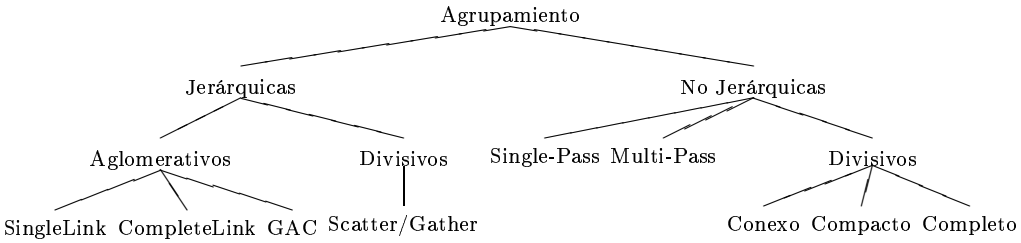


Figura 2.4: Clasificación de los algoritmos de agrupamiento.

en aglomerativos y divisivos. Los algoritmos divisivos suelen tener un coste computacional más bajo que los aglomerativos, pero dan peores resultados.

- Los *algoritmos aglomerativos* utilizan una clasificación de abajo-arriba. Comienzan suponiendo que cada documento es un grupo individual, y se unen iterativamente de dos en dos los grupos cuya función de semejanza supere un valor. A partir de las tres funciones de semejanza se obtienen tres algoritmos: el *Single-Link*, el *Complete-Link* y el *Group Average*.
- Los *algoritmos divisivos* utilizan una clasificación arriba-abajo, generan al principio un grupo formado por todos los documentos, y progresivamente se van sub-dividiendo hasta conseguir tantos grupos como documentos [Man99, Rij79]. Para dividir los grupos se crea un árbol MST (Minimal Spanning Tree), uniendo los documentos más similares por ejes, un grupo se divide en dos sucesivamente al eliminar el eje más largo. En función de la función de similitud utilizada se definen tres algoritmos: el *Single-Link Coherence*, el *Complete-Link Coherence* y el *Group Average Coherence*.

En estos algoritmos se suelen utilizar tres funciones de semejanza entre los grupos a partir de la semejanza entre los miembros de cada grupo [Man99, Rij79]:

Single-Link, toma como semejanza, el valor entre los miembros más semejantes de cada grupo.

Complete-Link, toma como semejanza el valor de la semejanza entre los miembros de cada grupo menos semejantes.

Group Average Cluster (GAC), toma como semejanza la semejanza media entre los miembros de los grupos.

Entre los algoritmos jerárquicos divisivos, cabe destacar el algoritmo de *Scatter/Gather* [Cut92], que se caracteriza por tener un coste lineal en vez de cuadrático.

2.5. Sistemas de *Extracción de Información*

A diferencia de los sistemas de RI, en un sistema de *Extracción de Información* (EI) se intenta no solo recuperar documentos que contienen información relevante, sino extraer aquellos hechos relevantes previamente establecidos, explícitamente presentes en los documentos, y representarlos en una forma útil. El formato de representación de la salida suele ser limitado y fijo en la mayoría de este tipo de sistemas. Los programas de extracción de información están situados entre los sistemas de RI y los de PLN [App99b, App99, Paz97]. La ventaja de estos sistemas es que permiten ignorar porciones de texto que no son relevantes para el dominio, de forma que no hace falta analizar todo el texto.

Estos sistemas son útiles en colecciones de documentos grandes que incluyen diversos temas, cuyos documentos poseen una longitud considerable, por lo cual no pueden ser abordados por los sistemas de PLN actuales. En la actualidad los sistemas de PLN funcionan correctamente en documentos de un área de conocimiento restringida, y no son lo suficientemente flexibles para exportarse de un dominio de aplicación a otro, por un lado debido a que no existen especificaciones predeterminadas o límites a los aspectos semánticos, por ello requieren mucho conocimiento, y por otro a que deben ser suficientemente flexibles para capturar el significado del texto.

Los problemas de la comprensión del Lenguaje Natural son los propios de una representación adecuada del conocimiento. Es posible comprender un texto si somos capaces de representar en una máquina el mismo conocimiento representado en el texto. Los sistemas de EI realizan una integración sintáctico-semántica para el análisis de colecciones de documentos. Éstos son menos complejos que los sistemas PLN, además de ser más portables. Con ellos se pueden analizar entradas que en los sistemas de PLN darían lugar a errores por no tener suficiente cobertura gramatical o por ser una entrada no gramatical.

En la actualidad la búsqueda y manipulación de datos en una colección de documentos para la obtención de información se realiza manualmente, siendo esta tarea la motivación principal de los sistemas de EI. Éstos se definen para estructurar la información, y obtener ciertos hechos relevantes citados en los documentos. El rango de hechos interesantes podría restringirse a un pequeño número de sucesos y relaciones aplicados a entidades de un conjunto limitado de tipos, especificados con anterioridad. Esta tarea es más simple que la de los sistemas de PLN, y permite procesar grandes cantidades de documentos en un periodo de tiempo razonable.

Aunque los diferentes lenguajes naturales son libres de contexto [Jur00], se pueden reconocer constituyentes no recursivos con gramáticas finitas, lo cual permite

abordar estos sistemas con técnicas de análisis superficial utilizando autómatas de estado finito que operan en cascada. Aunque esto parece una limitación, produce implementaciones robustas y rápidas. En la mayoría de aplicaciones prácticas de los sistemas de EI se suelen utilizar transductores (autómatas de estados finitos) [Jur00, App93]. Un transductor acepta símbolos constituyentes de frases que poseen asociadas sus características léxicas produciendo como salida, una anotación de las frases con las características o con propiedades relevantes del dominio. Los transductores pueden operar en cascada, siendo la salida de uno la entrada de otro.

En un esfuerzo de evaluar el estado del arte de los sistemas de EI, se crearon las MUC (Message Understanding Conferences) conducidos por el NOSC (Naval Ocean Systems Center) [Leh96] y patrocinadas por DARPA (Defense Research Projects Agency). Los organizadores han provisto un dominio de aplicación para la extracción de información y han definido las reglas de las tareas de extracción, para las cuales han proporcionado un corpus documentos etiquetados con la información que se debe extraer, donde la única restricción a los sistemas era la intervención humana. Los organizadores han creado un dominio de aplicación con un corpus de textos etiquetados con la información a extraer y un conjunto de textos para evaluar las aplicaciones de las organizaciones que quieren participar. En el MUC se ha optado por la utilización de plantillas atributo-valor para la evaluación de los sistemas de EI, de modo que ésta se realize comparando las salidas del sistema de EI con unas plantillas rellenadas manualmente.

Aunque los sistemas de EI todavía no son muy conocidos, como es el caso de los sistemas de RI (con los cuales ya estamos muy familiarizados gracias a los catálogos digitales en bibliotecas y principalmente al auge de la Web), existen en la actualidad varios campos en los cuales se están aplicando satisfactoriamente estos sistemas [App99, App99b]:

- Envío de asistencia médica. El sistema de EI está diseñado para resumir los campos del historial médico de cada paciente, extrayendo los diagnósticos, síntomas, análisis realizados y tratamientos. Estos sistemas se puede utilizar para asistir a los médicos, o bien para las compañías de seguros, para el cálculo de los reembolsos que debe dar a cada paciente asegurado.

- Inspección de literatura científica o técnica. Existen sistemas de EI para capturar información relevante de artículos técnicos o científicos, un ejemplo es la captura de las propiedades de un producto farmacéutico.

- Recabar información publicada en periódicos digitales o en noticias radiofónicas.

2.5.1. Evaluación de los sistemas de EI

Para la evaluación de los sistemas de EI se ha optado por utilizar las mismas medidas que se utilizan en los sistemas de RI, debido a que los resultados que se quieren obtener con ambos sistemas son semejantes. Así pues evaluaremos los sistemas de EI en función de la *Precisión*, la *Cobertura*, el *F-measure*, y el *Fallout*, los cuales definiremos a continuación para adaptarlos a estos sistemas [Jur00].

Precisión. Indica la proporción de información correcta extraída con respecto a las respuestas devueltas.

$$P = \frac{respuestas\ correctas\ devueltas}{respuestas\ devueltas}$$

Cobertura. Indica cuánta información relevante ha sido capturada del texto con respecto a todas las respuestas correctas posibles.

$$R = \frac{respuestas\ correctas\ devueltas}{respuestas\ correctas\ posibles}$$

F-measure. Medida que combina la cobertura y la precisión.

$$F - measure = \frac{(\beta^2 + 1) \cdot P \cdot R}{(\beta^2 \cdot P + R)}$$

Fallout. Indica la habilidad del sistema para ignorar la información errónea.

$$F = \frac{respuestas\ incorrectas\ devueltas}{nmero\ textos\ con\ informaci\acute{o}n\ falsa}$$

La experiencia de las conferencias MUC ha demostrado que la EI es una tarea difícil. Es interesante destacar lo difíciles que son de evaluar estos sistemas por los usuarios, incluso por analistas entrenados. La forma más natural de medir estos sistemas es midiendo las coincidencias entre las anotaciones humanas y las del sistema. En los experimentos realizados hasta la fecha se obtiene una media de coincidencia del 60-80 %, fijándose una barrera máxima en el F-measure de 0.6 [App99b] .

2.5.2. Técnicas utilizadas en EI

En los sistemas de Extracción de Información se pueden utilizar las aproximaciones de *Ingeniería del Conocimiento* y de *Aprendizaje Automático* [Bae99, Jur00]:

- En *Ingeniería del Conocimiento*, el diseñador del sistema debe estar familiarizado con los recursos lingüísticos existentes y los requerimientos del dominio para fijar las reglas que debe aplicar y las gramáticas de extracción que requiera el sistema. Para el desarrollo de las reglas puede aplicar conocimiento general, intuición o heurísticas. Las gramáticas se construyen a mano y los patrones del dominio son descubiertos por el experto humano a través del estudio del corpus. La tarea de refinamiento y perfeccionamiento de estos sistemas suele ser muy costosa.

  Existen dos formas de extraer patrones bajo esta aproximación:

  - *Aproximación molecular.* El experto lee distintos textos y trata de identificar los patrones más comunes con los cuales se expresa la información. Después se construyen reglas para generalizar esos patrones, y para finalizar se buscan los patrones menos comunes que no cumplen las reglas anteriores. Esta aproximación tiene como objetivo obtener una alta precisión.
  - *Aproximación atómica.* La idea básica es asumir que todas las frases nominales y verbos de un tipo o clase determinada, contienen la información que se desea extraer e indican un suceso o relación de interés, independientemente de las relaciones entre ellos. Esto da lugar a una proliferación de todas las descripciones posibles de sucesos/relaciones, de modo que posteriormente se tienen que combinar para producir las instancias completas de suceso/relación, y entonces filtrar los resultados según ciertos criterios. Esta aproximación se puede realizar si las entidades en el dominio tienen tipos fácilmente identificables y las plantillas están estructuradas de modo que sólo admiten un tipo de entidades y éstas sólo se utilizan para rellenar un pequeño número de atributos.

- La aproximación utilizando el *Aprendizaje Automático* se basa en un aprendizaje supervisado, de modo que se proporciona al sistema un conjunto de documentos de entrenamiento, los cuales tienen etiquetada la información que deben extraer. Esta técnica utiliza métodos estadísticos cuando es posible. El sistema aprende reglas a partir de textos etiquetados y de la interacción con el usuario. La parte costosa de este sistema es la construcción de un gran corpus etiquetado para el correcto funcionamiento del sistema.

Desde un punto de vista científico la aproximación utilizando el aprendizaje es más atractiva, aunque en la actualidad, los mejores sistemas de EI se han construido estableciendo las reglas manualmente.

La aproximación mediante sistemas de aprendizaje se ve limitada por la necesidad de un experto humano para obtener el corpus o guiar el proceso de aprendizaje.

El proceso de aprendizaje está limitado a obtener patrones de extracción de sucesos, ya que los patrones lingüísticos para entidades o relaciones son independientes del dominio y se pueden obtener manualmente, sin necesidad de aprendizaje automático para nuevos dominios. Pero a pesar de ello, éstos dependen del estilo del autor del corpus y del contexto, por lo cual se construyen manualmente los patrones lingüísticos para cada corpus.

Cuando se extraen patrones, hay que tener en cuenta que es posible encontrarse ambigüedades. Un ejemplo claro puede ser un dígito de 8 cifras que puede tratarse de un código o de una fecha, por lo cual se requiere el estudio del contexto para saber de que entidad se trata ese patrón.

2.5.3. Componentes de un sistema de EI

Un sistema de EI típico posee cuatro módulos principales: un segmentador, un procesador morfológico-léxico, un analizador sintáctico, y un analizador del dominio. Cabe destacar que se pueden obviar algunos de estos módulos según la lengua del dominio, ya que la parte del problema que tratan de resolver puede resultar trivial.

2.5.3.1. El segmentador

Este módulo permite la extracción de la porción del documento a analizar. Para ello debe ser capaz de extraer los vocablos, las oraciones, e incluso en algunas aplicaciones la estructura del documento.

La extracción de oraciones y vocablos en algunas lenguas como el español, pueden ser bastante triviales, pero no ocurre así en lenguajes como el chino que requieren un módulo para dividir las cadenas en palabras.

2.5.3.2. Procesamiento morfológico y léxico

Este módulo realiza un análisis morfológico-semántico para asignar a la parte del discurso a analizar (palabras u oraciones) ciertas características morfológicas o léxicas. Lenguajes con muchas inflexiones morfológicas, como el español, requieren un análisis morfológico, mientras otros lenguajes como el inglés no precisan esta etapa. En el análisis léxico se asignan características léxicas a las palabras. Para ello se puede hacer uso de un *lexicon*, un diccionario de dominio específico,

o una base de datos terminológica. Existe la tendencia a intentar completar lo más posible los lexicones, y aunque sea una paradoja, no por ser más extensos se mejoran los resultados del sistema EI, ya que también implica la aparición de más ambigüedades.
Para desambiguar los distintos significados de cada palabra, una vez se ha reducido a sus lexemas, se pueden utilizar dos métodos :

Part of Speech Tagging. Este método etiqueta cada una de las palabras del texto con la información que se dispone en un lexicon. Para tratar palabras con significados poco comunes y eliminar las ambigüedades se intenta tener un lexicon muy completo. Hay que tener en cuenta que este método suele producir un buen etiquetado en el 95 % de los casos, pero generalmente donde más importante suele ser la información que se requiere es precisamente en los casos en que es más propenso el sistema a producir errores. Además hay que tener en cuenta que este proceso suele requerir mucho tiempo de procesado, y bastante esfuerzo para ejercitar el sistema en textos largos.

Word Sense Tagging. Es un método alternativo para asignar significados a las palabras, con ayuda de un diccionario con el cual se etiquetan las palabras, y en el caso de palabras ambiguas se asigna a cada palabra una lista con los significados posibles y la frecuencia de cada uno de esos sentidos. Esta alternativa no es perfecta pero llega a un buen compromiso entre la precisión y la eficiencia.

El proceso más complicado de este módulo es la detección y análisis de entidades (nombres propios, fechas, porcentajes, etc.). En el caso por ejemplo de nombres propios, aunque se utilizara un léxico que poseyera una larga lista de nombres propios, hay que tener en cuenta que la aplicación puede producir errores bien cuando hay nuevos nombres que no existen en la lista, o bien cuando se producen ambigüedades ya que un nombre propio a veces tiene varios significados. Por ejemplo, Julio puede ser un nombre propio de persona o bien un mes. La mayoría de sistemas del reconocimiento de nombres propios utilizan técnicas de aprendizaje basados en los modelos de Markov, aunque también existen modelos basados en gramáticas de estados finitos con comportamientos muy buenos como es el caso de FASTUS [Hob96, App93] y TextPro [Dou95].

2.5.3.3. Análisis sintáctico

El módulo de análisis sintáctico permite detectar entidades relevantes al dominio. El análisis sintáctico se realiza con técnicas de PLN, aplicando ciertas restricciones para poder procesar grandes cantidades de texto en un tiempo razonable.

Esto es posible gracias a que los sistemas de EI están debidamente dirigidos hacia la extracción de relaciones relativamente simples entre entidades singulares. Al estar los sistemas de EI típicamente diseñados para fragmentos estructurados simples, estos se pueden analizar con gramáticas de estado finito, las cuales proporcionan un procesado sencillo, robusto y rápido.

Uno de los factores más complicados es el análisis de los modificadores y de los distintos constituyentes, como son las frases preposicionales. Éstos se suelen ignorar excepto en un pequeño conjunto de palabras relevantes para el dominio. En este conjunto de casos simples es posible aplicar ciertas heurísticas para unir correctamente los distintos elementos. En los casos en que las sentencias no sean relevantes para el dominio, no se debe extraer información, con lo cual no importa que el análisis sea correcto.

Destacaremos algunas estructuras sintácticas complejas que aplicadas a un cierto dominio, pueden resolverse mediante un análisis parcial, como son el caso de:

- Cláusulas relativas unidas al sujeto de otra cláusula. Este tipo de sentencias se pueden analizar mediante una regla de *dominio no determinista*, asumiendo que ambas cláusulas contienen información relevante para el dominio. En ese caso el núcleo del sujeto es el sujeto de las dos cláusulas y requiere solamente una regla que omitir el pronombre relativo que une a las dos.

- El problema de la coordinación de sentencias es uno de los problemas sintácticos más difíciles de resolver en el PLN. La aproximación que se realize en los sistemas de EI dependerá de lo que se intenta coordinar y de las propiedades de las sentencias relevantes para el dominio.

- Las frases preposicionales son otro de los problemas en los sistemas de análisis parciales, que se pueden solucionar mediante un segundo análisis de la frase para buscar ciertas preposiciones. Este segundo análisis puede también juntar locativos a grupos nominales relevantes al dominio. Si las preposiciones están subcategorizadas a verbos relevantes, y los objetos son relevantes, entonces los argumentos de la frase pre-posicional pueden analizarse por reglas que se activan al encontrar esos verbos relevantes. En otros casos estas frases se trataran como adjuntos adverbiales. Los adjunto-locativos y temporales son típicamente interpretados asumiendo que el suceso que modifican es relevante para el dominio.

Se ha intentado aplicar un análisis sintáctico completo en algunos sistemas de EI, como en la aplicación de TACITUS (MUC-3) [App99] y Proteus(MUC-6) [Gri95]. La experiencia demuestra que el uso de análisis completo sin restringirlo a un dominio específico no es apropiado debido a que produce una explosión

combinatoria en las frases largas. Por ello se ha llegado a la conclusión de que los análisis de estado finito parciales son la mejor alternativa en los sistemas de EI.

2.5.3.4. Análisis del dominio

En este módulo se pueden destacar tres operaciones:

- Resolución de co-referencias, anáforas, deíxis y ambigüedades. En la extracción de información las entidades involucradas en un suceso relevante y sus relaciones se pueden referenciar de distintas formas a lo largo del texto, y normalmente suelen estar bastante separadas. Como caso especial, la resolución de las co-referencias en los nombres de personas, compañías o entidades se resuelve mejor en el módulo de reconocimiento de nombres.

  Otro caso particular de co-referencias es el problema de las referencias temporales relativas, como 'hoy', 'hace 2 días', etc. Para su resolución se requiere determinar la fecha en el que el texto fue escrito o publicado, o bien reconocer fechas absolutas citadas con anterioridad y realizar ciertas operaciones que requieren un calendario. El caso especial de las co-referencias temporales ilustran hasta que punto las referencias, y por lo tanto las co-referencias , son distintos modos de representar una misma entidad.

- Combinación de resultados parciales. Consiste en combinar la información extraída de diferentes frases acerca del mismo suceso. Si un sistema puede fácilmente identificar el mismo suceso a través de múltiples documentos, éste puede explotar esta redundancia para obtener más información de un suceso que solo utilizando un único artículo.

- Construcción de la salida. Ésta es la tarea más crítica para los sistemas de EI. Algunos simplemente extraen las frases que contienen la información relevante, mientras que otros más ambiciosos tratan de expresar la información extraída de un modo independiente al del texto original.

2.5.4. Las tareas de investigación en los sistemas de EI

Dentro de las conferencias del MUC, donde se están desarrollando gran parte de la tecnología de los sistemas de EI, se está investigando en las siguientes tareas genéricas:

***Name Entity Recognition Task* (NE).** Tarea de reconocimiento de entidades (nombres de personas, organizaciones, lugares, fechas, tiempo, monedas, porcentajes). Generalmente se añaden etiquetas SGML en el texto para etiquetar las cadenas que representan las entidades.

***Multilingual Entity Task* (ME).** Tarea de reconocimiento de entidades en idiomas que no son el inglés. Se ha aplicado tanto al español, el chino y al japonés.

***Template Element Task* (TE).** Tarea de añadir información descriptiva a los resultados de NE. Requiere reconocer la individualización de entidades. Por ejemplo, el reconocer como la misma entidad a todas las formas de escribir el mismo nombre de una persona: nombre y apellido, sólo uno de los dos, o bien abreviaciones.

***Template Relation Task* (TR).** Tarea para descubrir relaciones entre entidades. Requiere la identificación de un pequeño número de posibles relaciones entre los distintos elementos. La extracción de relaciones entre entidades es un punto clave en la mayoría de tareas de extracción de información.

Coreference Task. Tarea de identificar las expresiones del texto que hacen referencia al mismo objeto. Capturan la información de las expresiones de coreferencia (referencias a entidades), incluyendo toda esta información en las etiquetas de NE y TE. Requiere la identificación de expresiones de referencia y la separación de las expresiones de referencia en clases de equivalencia.

Scenario Template Task. Tarea para la obtención información sobre sucesos o relaciones de interés. Generalmente para esta tarea se tiene una plantilla que involucra ciertas relaciones y sucesos de interés que se tienen que extraer de un texto. Un ejemplo es la identificación de las víctimas de un atentado.

2.6. Seguimiento, detección y clasificación de sucesos

Los sistemas para el seguimiento y la detección de sucesos en las noticias (TDT-*Topic Detection and Tracking*) comenzaron como una iniciativa en 1997 soportada por el DARPA dentro del programa TIDES (*Translingual Information Detection, Extraction and Summarization*). El propósito de este proyecto es investigar nuevas técnicas computacionales para analizar las colecciones de noticias (habladas o escritas) y detectar los sucesos en ellos narrados.

En TDT [TDT, Yang98, Pap99] se han definido cinco tareas de investigación:

Segmentación de noticias (*Story Segmentation*). Consiste en extraer cada noticia de la colección. Esta etapa es trivial en el caso de noticias escritas, como los periódicos, ya que el formato del texto permite detectar cuando empieza y termina cada noticia. Pero en las noticias habladas, como las radio-noticias y los telediarios, esta tarea es más compleja, siendo todavía una línea de investigación abierta.

Seguimiento de sucesos (*Topic Tracking*). Consiste en clasificar las noticias en un conjunto de sucesos predeterminados. El problema se resuelve con técnicas de clasificación supervisada donde el sistema conoce a priori los sucesos de interés. Para ello se posee una colección de noticias ya clasificadas en cada uno de los sucesos de interés para el entrenamiento del sistema.

Creación de tópicos (*Topic Detection*). Consiste en buscar los distintos sucesos que aparecen en las noticias y agrupar las noticias que hablan sobre el mismo tópico. Este es un problema de clasificación no supervisada, donde se trata de organizar o agrupar automáticamente las noticias sobre el mismo suceso. En este caso, a diferencia del anterior, no hay documentos para entrenar el sistema, y además no se conocen a priori los sucesos de interés. Existen dos modos de analizar la colección de noticias en esta tarea:

- Inmediata (*on-line*). El sistema decide si el documento de la colección habla de una nueva historia antes de mirar en el siguiente relato de la colección.
- Retardada (*retrospective*). El sistema debe decidir si un documento representa un nuevo tópico considerando todos los relatos del corpus, no solo el primer documento del corpus que habla sobre un nuevo suceso.

Ambas formas de detección no tienen conocimiento previo de los sucesos a detectar, aunque pueden tener acceso a las noticias anteriores, de modo que se pueden utilizar para contrastar y determinar cuándo se produce un nuevo suceso [Yan00].

Detección de la primera noticia sobre un suceso (*First Story Detection*). Trata de identificar en la colección de noticias, aquellas que narran por primera vez un suceso. Esta tarea está muy relacionada con la anterior, ya que el sistema de detección crea un nuevo tópico cada vez que localiza una noticia que no se asemeja a las que posee el sistema, o sea, esa noticia se clasifica como la primera que relata un nuevo suceso. La correcta detección de la primera noticia sobre un suceso dará lugar a que el sistema de detección funcione mejor. Por ello se ha separado como una tarea independiente.

Enlazar noticias (*Story Link Detection*). Trata de detectar cuándo dos noticias hablan sobre el mismo suceso. El sistema debe comprender qué es un tema,

independientemente de los temas específicos, y calcular la semejanza entre pares de documentos. Esta tarea no tratan de dividir los documentos en conjuntos ortogonales, se permite que un documento hable de distintos temas, por lo que un documento puede pertenecer a varios grupos.

En los trabajos de TDT se ha comprobado que la utilización de técnicas que detecten en el relato las expresiones que hablan acerca de quién, qué, cuándo y dónde ocurre el suceso, permiten aumentar la efectividad de estos sistemas, ya que estas expresiones son básicas en la definición de un suceso [All98b, Pap99]. Sin embargo, hay que tener en cuenta que estas palabras pueden diferir a lo largo del tiempo en los relatos que tratan de la misma historia, debido principalmente a la evolución del suceso, lo cual produce la inclusión en las noticias de nuevos sucesos muy relacionados, con más información y datos sobre el suceso.

En el trabajo de tesis de Ron Papka [Pap99], se ha realizado un buen estudio sobre el estado del arte de los sistemas de TDT. Allí se destaca que uno de los errores más comunes en la detección de sucesos se produce cuando se trata de sucesos específicos. Concretamente, se plantea el problema de cómo distinguir sucesos que hablan acerca del mismo tema, pero a distintas granularidades ('Accidente de un avión en USA vuelo 427', 'accidentes de aviones en USA'). Todavía existe cierta ambigüedad en la definición del concepto de suceso en TDT.

Según se destaca en [All98b], quedan todavía preguntas por resolver en este campo:

- ¿Cómo se relacionan dos sucesos entre sí, o un suceso con un sub-suceso?
- ¿Es necesario utilizar modelos para capturar nociones preferentes de granularidad de sucesos, o es suficiente una definición general?
- ¿Existe algún modo de seleccionar sólo los sucesos interesantes de las noticias y excluir las noticias que no son de interés?
- ¿El PLN puede ayudar a identificar características relacionadas con quién, qué, dónde y cuándo?

2.6.1. Evaluación de los sistemas TDT

Los sistemas de TDT son sistemas de recuperación de información utilizados para la detección de sucesos. Por ello para su evaluación se utilizan dos medida:

F1-measure. Éste parámetro nos va a permitir evaluar el sistema como un sistema de RI (ver la sección 2.3.3).

Coste de Detección (C_{Det}). El coste de detección [TDT00, Pap99], evalúa en una sola medida los dos tipos de errores que se producen en los sistemas de detección; los errores por *falsas alarmas* (P_{Fa}) y los *errores por omisiones* (P_{miss}). Dado un suceso, las *falsas alarmas* son todos aquellos documentos que el sistema asigna como pertenecientes al suceso y no lo son, y los *errores por omisión* son todos aquellos documentos que el sistema no a asignado a ese suceso.

$$C_{Det} = C_{FA} \cdot P_{FA} \cdot P_{nrel} + C_{Miss} \cdot P_{Miss} \cdot P_{rel} \tag{2.2}$$

donde :

- $P_{FA}(P_{Miss})$ indica la probabilidad de producirse una *falsa alarma* (*error por omisión*),
- C_{Miss} y C_{FA} son los costes del sistema por producir esos errores,
- P_{rel} es la probabilidad de que un documento sea relevante con el tópico que se le asigna, de modo que $P_{nrel} = 1 - P_{rel}$.

En la segunda fase del TDT (TDT-2) se utilizan los valores de $P_{rel} = 0{,}02$ y $C_{Miss} = C_{FA} = 1$ [TDT98]. En la evaluación del año 2000 (TDT-2000) se redefine la función de coste de modo que $C_{Miss} = 1$ y $C_{FA} = 0{,}3$ en la tarea de segmentación, y de $C_{FA} = 0{,}1$ en las otras [TDT00]. Así pues se considera un buen sistema de detección si el valor de C_{Det} no supera el valor de 0.02.

En TDT se evalúan los sistemas utilizando la siguiente tabla de contingencia [Yan98b]:

	Relevante	No Relevante
Recuperado	a	b
No recuperado	c	d

Así se define para cada tópico :

- $Cobertura = R = \dfrac{a}{a+c}$
- $Precisión = P = \dfrac{a}{a+b}$
- $F1 - Measure = 2 * \dfrac{P * R}{P + R}$
- $MissRate = P_{Miss} = \dfrac{c}{a+c}$

- $FalseAlarmRate = P_{FA} = \frac{b}{b+d}$
- $C_{Det} = C_{FA} * \frac{b}{b+d}(1 - P_{rel}) + C_{Miss}\frac{c}{a+c} * P_{rel}$

Para evaluar el comportamiento del sistema global para todos los sucesos se suelen utilizar dos métodos:

Promedio de los documentos (*Story-weighted*). El cual asigna el mismo peso a cada suceso, de modo que se acumulan los valores de a, b, c y d que se obtienen para cada suceso, y luego se aplica la medida de evaluación con los valores acumulados.

Promedio por sucesos (*Topic-weighted*). Se evalúa el sistema para cada suceso, y luego se promedian los valores obtenidos para cada tópico.

Debido a la alta variabilidad del número de noticias o documentos en cada suceso, el método *promedio por sucesos* estima mejor el comportamiento del sistema de TDT [TDT98].

En los sistemas de detección también se suele representar el comportamiento del sistema mediante una curva denominada C-DET (*Detection Error Trade of Curves*), la cual se obtiene representando en una gráfica la evolución de los *errores por omisión* frente a las *falsas alarmas*, según se varían los parámetros del sistema. Estas curvas permiten refinar el sistema para mejorar su funcionamiento. En un principio se representaba el comportamiento de los sistemas de detección mediante curvas ROC (*Receiver Operating Characteristic*) [Mar97] donde se representaba el porcentaje de datos correctamente detectados frente a las falsas alarmas. En las curvas C-DET se representan los dos tipos de errores. Estos métodos permiten que la curva de un sistema que tiene un buen comportamiento se asemeje a una recta dentro del cuadrante inferior izquierdo $(50, 50)$ al escalar los ejes con una escala de desviación normal. La curva $y = -x$ representa el comportamiento aleatorio del sistema. Y el comportamiento de un sistema será mejor cuanto más se aproxime su curva al extremo inferior izquierdo.

2.6.1.1. El tiempo en el TDT

Ron Papka en [Pap99] analiza cómo evoluciona la publicación de las noticias sobre un suceso a medida que pasa el tiempo. De su análisis extrae las siguientes conclusiones:

- Un nuevo suceso normalmente produce una cadena de noticias sobre ese suceso en un espacio de tiempo próximo a él.
- Una ventana temporal entre dos secuencias de noticias que hablan del mismo tema generalmente indican sucesos diferentes.
- Un cambio significativo del vocabulario y de la frecuencia de distribución de los términos suelen indicar un nuevo suceso.
- Generalmente se habla de un suceso en un espacio de tiempo relativamente pequeño.
- Se ha observado que las noticias relacionadas con el mismo suceso ocurren en grupos, los sucesos inesperados producen mucho interés en la audiencia, y por ello dan lugar a que se hable de otros sucesos del pasado de características similares al actual.
- Dentro de la cobertura de las noticias, los nombres de personas, lugares y otros datos de interés en la historia no se mencionan generalmente mucho en el pasado, por lo cual son importantes para caracterizar la noticia.
- Un problema a tener en cuenta en este campo es la variación de las noticias que hablan sobre un suceso a lo largo de los días en que se publica. Por ejemplo, en el caso de la 'Bomba de Oklahoma'. Las primeras noticias hablan sobre las consecuencias de la bomba. A los 61 días después descubren que el culpable es McVigh y lo publican, ¿cómo se relacionan estas noticias?. La solución adaptada por [All98b] se basa en *adaptive tracking*, que consiste en ir añadiendo a la pregunta inicial las nuevas palabras clave que identifican el suceso con uno anterior.

Ron Papka propone la utilización de un algoritmo de clustering (*Single-Pass*), extendido con el tiempo como estrategia tanto para la detección de la primera noticia, como para la detección de sucesos. Se supone que las noticias cercanas a la fecha de publicación son mas propensas a hablar sobre el mismo tema. De este modo, un documento pertenecerá a una clase si supera un umbral, y la distancia en días entre los documentos no supera un determinado umbral. La introducción de la fecha de publicación mejora el sistema de TDT, debido a que ayuda al sistema a capturar la periodicidad y el solapamiento temporal entre las noticias.

En el trabajo realizado por Russell Swan y James Allan [Swa99], se intenta agrupar las noticias para la obtención de la historia completa sobre un suceso ayudados por la fecha de publicación y algunas características que se extraen de los documentos como son nombres de entidades o frases nominales relevantes. En este trabajo se demuestra como la aplicación de técnicas estadísticas sencillas, y el

uso de la fecha de publicación como metadato permiten una descripción sobre las historias de los sucesos relatados en el corpus de documentos. Cada historia de un suceso en su sistema viene representado mediante un conjunto de características (nombres de entidades o frases nominales relevantes de los documentos que se han agrupado para formar ese suceso) y un periodo de tiempo que representa el ámbito de días en los que se ha publicado ese suceso. Para la generación de las historias de un suceso, se extraen las características relevantes de cada noticia cuya frecuencia de aparición supera cierto umbral, y que se publican el mismo día o en días consecutivos. La agrupación de estas características cuando se solapan en el tiempo permiten la construcción la historia de un suceso. A partir de este método se pueden aplicar dos estrategias:

- Sistema de multi-día. Sólo se seleccionará una característica si es relevante más de un día.
- Sistema de un día. Todas las características se toman como posibles características de una historia.

La utilización del sistema de un día, da lugar a muchas historias falsas, mientras que la utilización del sistema multi-día, da un conjunto de sucesos muy significativos pero alguna de las historias se pierden (si el suceso dura sólo un día, como ocurre con los sucesos puntuales).

Se observa que existe cierto error con respecto a las agrupaciones manuales. Esto es debido a dos razones, bien a que manualmente se asigna una fecha al suceso que suele diferir de la fecha de publicación, siendo en la mayoría de los casos el día anterior, o bien a que existen noticias cuyo suceso dura varios dias y no todos los días se publica información sobre él.

En este trabajo se han generado las historias con dos mecanismos, utilizando los términos que aparecen en los documentos o bien extrayendo de los documentos los nombres de entidades. El segundo método da mayor calidad y más información sobre la noticia, de modo que el conjunto de características suele ser menor y además los nombres de entidades permiten describir la historia de un suceso de un modo bastante comprensible.

En la evaluación final sobre el TDT-2 [TDT98], se destacan las mejoras detectadas en los sistemas de detección y seguimiento de noticias utilizando la fecha de publicación, haciendo hincapié en los sistemas desarrollados por UMass [Swa99, All00], y CMU [Yang98, Yan00, Yan98b], y las comparativas de estos sistemas con y sin utilización de la fecha de publicación. En estos trabajos se concluye que estos resultados se deberían mejorar, destacando que la fecha de publicación produce

ciertos errores en la detección de los sucesos. Queda como tema abierto de investigación cómo puede mejorar la información temporal en la detección y seguimiento de sucesos.

2.7. Conclusiones

Con el propósito de analizar la problemática de localizar automáticamente documentos que hablan sobre sucesos, en este capítulo hemos presentado distintos sistemas relacionados con el procesamiento de los documentos para su recuperación.

Hemos visto cómo los sistemas de RI permiten recuperar documentos, y como debido al auge de la Web, es un área todavía en investigación para conseguir sistemas que ayuden mejor a usuarios no expertos. En la actualidad para mejorar estos sistemas se están aplicando las técnicas de EI. Los sistemas de EI permiten obtener con análisis no muy complejos cierta información sobre los documentos, como personas involucradas, lugares, relaciones entre entidades, etc. Estos sistemas también son capaces de reconocer las deíxis, anáforas y co-referencias que aparecen en los textos. Los estudios demuestran que la aplicación de estas técnicas mejoran los resultados de los sistemas de RI.

Por otro lado hemos encontrado que existe un área de reciente creación dentro de los sistemas de RI, para el seguimiento y detección de los sucesos, los sistemas de TDT. En estos sistemas se utilizan distintos métodos de clasificación para agrupar los documentos que hablan sobre el mismo suceso. En las investigaciones realizadas se ha observado como mejoran los resultados al utilizar un sistema de EI para obtener información acerca de las personas involucradas, además de información de cuándo y dónde se produce el suceso. Para reconocer cuándo se produce un suceso suelen utilizar la aproximación de que un suceso se produce el día en que se publica. Pero como hemos descrito anteriormente, los resultados obtenidos con la utilización de la fecha de publicación podrían mejorarse si se consideran otras propiedades temporales.

En la literatura hemos observado que en los sistemas de RI se permite la utilización de atributos de tipo fecha en las consultas. En los sistemas de EI, dentro de la tarea de reconocimiento de entidades, algunos sistemas tratan de detectar expresiones temporales. Pero no hemos encontrado dentro de los sistemas de procesamiento de los documentos herramientas automáticas que obtengan la información temporal implícita en los documentos, y las utilicen para ayudar a localizar

documentos semejantes. Como veremos en el resto de este trabajo, nosotros proponemos una herramienta para detectar y reconocer distintas expresiones temporales tanto absolutas como relativas, utilizando un modelo de tiempo y técnicas de PLN.

Como se destaca en otros trabajos de TDT, determinar cuándo ocurre un suceso es relevante para localizarlo. A lo largo de los siguientes capítulos, vamos a proponer distintas técnicas para analizar la información temporal implícita en los documentos. También trataremos de ver como éstas pueden ayudar a los usuarios en la recuperación de los documentos sobre un suceso de interés, así como en la detección automática de los sucesos narrados en una colección de documentos. Esta última tarea se relaciona con la detección de sucesos propia de TDT, para lo cual utilizaremos algoritmos de agrupamiento de documentos que tengan en cuenta tanto los conceptos que aparecen en los documentos como la información temporal presente en ellos.

Capítulo 3

Modelo del Tiempo

3.1. Introducción

Para medir el tiempo nos basamos en el movimiento periódico de los astros y los planetas, los cuales nos permiten definir distintas unidades de medida para el tiempo, así se define:

- un año corresponde al tiempo empleado por la Tierra en completar su órbita alrededor del Sol (365 días, 5 horas, 48 minutos y 46 segundos),
- un mes es el tiempo que tarda la Luna en regresar a la misma posición con respecto al Sol y la Tierra (30 días aproximadamente),
- una semana coincide con el tiempo empleado por la Luna en dar una vuelta a la tierra (7 días),
- el día equivale al tiempo empleado por la Tierra para efectuar una rotación sobre su propio eje.

Estas unidades de medida se pueden utilizar bien para describir acciones periódicas, duraciones o bien el tiempo cronológico o absoluto. Para poder representar el tiempo absoluto, se construyeron los calendarios. Estos permiten definir un instante de tiempo con una precisión que depende de las unidades de medida que se empleen. En el caso del calendario Gregoriano podemos tener una precisión de días, meses, años, décadas, siglos o milenios. A partir de las distintas unidades de medida del tiempo se construyen expresiones lingüísticas en Lenguaje Natural [Fer96] para representar un instante ('en mayo de 1999'), un intervalo temporal ('del miércoles

al viernes'), así como duraciones ('durante dos días'). A estas expresiones las vamos a denominar **expresiones temporales**. Los instantes e intervalos temporales se pueden expresar explícitamente utilizando expresiones absolutas ('en mayo de 1999') o bien implícitamante utilizando expresiones relativas ('dos días después'). Sin embargo, una duración puede representar implícitamente un instante o un intervalo temporal ('durante los dos meses siguientes a las elecciones', 'durante ese mes', 'hace una década') aunque en otros casos representa acciones periódicas ('todos los lunes', 'cada dos días'), o simplemente duraciones de sucesos ('estuvo en la prisión durante tres meses', 'el arroz requiere media hora para cocerse'), donde lo importante es la duración, no cuándo ocurre el suceso.

Las actividades humanas están muy relacionadas con las unidades básicas del calendario o del reloj. En este trabajo pretendemos reconocer las expresiones temporales que representan instantes o intervalos de forma explícita, y también aquellas duraciones que implícitamente denotan un instante o intervalo de tiempo ('hace dos días'). En el el próximo capítulo describiremos el proceso de detección y reconocimiento de las distintas expresiones temporales utilizando el modelo de tiempo que vamos a describir en este capítulo.

El modelo de tiempo formal propuesto permite representar entidades temporales (instantes,intervalos y duraciones temporales) y las operaciones entre éstas, con el objetivo de poder extraer los tiempos cronológicos presentes en los documentos, operando con las distintas entidades temporales que encontremos en las expresiones temporales.

En el próximo apartado vamos a definir qué se entiende por un modelo de tiempo. Definiremos en la sección 3.2.1 el concepto de granularidad y las relaciones que se pueden establecer entre ellas utilizando el glosario propuesto por Bettini en [Bet98], pero ampliándolo con los conceptos de Jajodia y Wang [Ning01]. En el apartado 3.2.2 mostraremos las limitaciones de los modelos formales de tiempo propuestos en la literatura. En el apartado 3.3 propondremos un modelo de tiempo basado en el calendario Gregoriano, que permite operar entre entidades temporales a distintos niveles de abstracción.

3.2. El modelado del tiempo

El espacio temporal consta de una única dimensión, que se representa como una línea que puede ser discreta o continua. La dirección en la línea temporal separa los instantes pasados de los futuros con respecto del punto de referencia. Aunque el tiempo se percibe como continuo generalmente se utiliza un modelo discreto para

su representación y medida, debido principalmente a que las medidas del tiempo tienen una precisión inherente a los sistemas de medida.

El modelo de tiempo puede ser:

- *Lineal*, el tiempo avanza del pasado al futuro, en una sola línea de forma que sólo existe un pasado y un futuro
- *Ramificado*, el tiempo es lineal hasta el instante presente, a partir del cual parten los diversos caminos que puede tomar el futuro.
- *Cíclico*, permite representar procesos recurrentes en el tiempo, entre los cuales no se puede establecer un orden, y por tanto no se puede distinguir entre pasado y futuro

El tiempo cronológico se suele representar mediante un modelo de tiempo lineal discreto. Cuando se representa el tiempo lineal en un espacio discreto se utiliza un isomorfismo entre éste y los números enteros realizando divisiones del tiempo regulares. Debido a que existen diversas unidades de medidas del tiempo, podemos representar un mismo punto temporal con distintos niveles de abstracción, los cuales determinarán la precisión de un punto o intervalo temporal.

3.2.1. Glosario de conceptos para modelar el tiempo

En los trabajos de [Bet98] y [Ning01] se proporcionan las definiciones básicas para la creación de un modelo del tiempo general. Éstas son:

Dominio de tiempo. Un dominio de tiempo es un par $(T; \leq)$, donde T es un conjunto no vacío de instantes de tiempo y, $\leq$ es la relación de orden total en T. Utilizaremos $<$ para denotar orden estricto entre primitivas.

Granularidad. Una granularidad G es un mapeado desde el conjunto de números enteros al dominio del tiempo, así pues los números enteros permiten indexar los elementos de G, de modo que

1. si $i < j$, donde $i, j \in \mathbb{Z}$, y $G(i)$ y $G(j)$ no son vacíos, entonces cada elemento $G(i)$ es menor que los elementos de $G(j)$, y
2. si $i < k < j$, donde i, j y $k \in \mathbb{Z}$ y $G(i)$ y $G(j)$ no son vacíos, entonces $G(k)$ no es vacío.

Granularidad etiquetada. Una granularidad etiquetada G es un par $(\mathcal{L}, g)$, donde g es un mapeado desde el subconjunto de los números enteros $\mathcal{L}$, al dominio del tiempo, de modo que:

1. si $i < j$, donde $i, j \in \mathcal{L}$, y tanto $G(i)$ cómo $G(j)$ no son vacíos, entonces cada elemento $G(i)$ es menor que los elementos de $G(j)$,
2. y si $i < k < j$, donde i, j y $k \in \mathcal{L}$, y $G(i)$ y $G(j)$ no son vacíos, entonces $G(k)$ no es vacío.

Gránulo. Cada subconjunto $G(i)$ no vacío, donde i es uno de los índices y G es una granularidad, se denomina gránulo de G.

Relaciones entre granularidades. Sean G y H dos granularidades que pueden ser etiquetadas o no, se definen las siguientes relaciones:

Agrupa. Diremos que una granularidad G se agrupa en una granularidad H, y lo denotaremos como $G \trianglelefteq H$, si para cada índice j de H existe un conjunto $S \subset \mathbb{Z}$ tal que $H(j) = \bigcup_{i \in S} G(i)$.

Fina. Una granularidad G es más fina que la granularidad H, y lo denotaremos como $G \preceq H$, si para cada índice i de G, existe un j de H, tal que $G(i) \subseteq H(j)$. Si $G \preceq H$ entonces se dice que H es más **gruesa** que G.

Particiona. Una granularidad G particiona una granularidad H, y lo denotaremos cómo $G \barwedge H$, si $G \trianglelefteq H$ y $G \preceq H$.

Agrupa periódicamente. Diremos que una granularidad G se agrupa periódicamente en una granularidad H, lo cual denotaremos como $G \; \pi \; H$, si $G \trianglelefteq H$ y existe $n, m \in \mathbb{Z}^+$ donde para todo $i \in \mathbb{Z}$, si $H(i) = \bigcup_{r=0}^{k} G(j_r)$ y $H(i+n) \neq \varnothing$, entonces $H(i+n) = \bigcup_{r=0}^{k} G(j_r + m)$.

Subgranularidad. Diremos que una granularidad G es subgranularidad de H, lo cual denotaremos cómo $G \sqsubseteq H$, si para cada índice i existe un índice j tal que $G(i) = H(j)$.

Granularidad equivalente. Diremos que una granularidad G es equivalente a H, lo cual denotaremos cómo $G \leftrightarrow H$, si $G \sqsubseteq H$ y $H \sqsubseteq G$.

símbolo	relación
$\leq$	orden
$\trianglelefteq$	agrupa
$\prec\!\!=$	fina
$\succ\!\!=$	particiona
π	agrupar periódicamente
$\sqsubseteq$	sub-granularidad
$\leftrightarrow$	equivalencia
g_rel	relación genérica

Cuadro 3.1: Relaciones entre dos granularidades.

Granularidad Elemental. Dado un conjunto de granularidades que poseen el mismo dominio temporal, y una relación de orden entre ellas g_rel, diremos que G es una granularidad elemental del conjunto con respecto a g_rel si se cumple que $G\ g\_rel\ H$, para todo H del conjunto de granularidades.

Calendario. Un calendario $\mathcal{C}$ es un conjunto de granularidades (etiquetadas o no) sobre un dominio de tiempo que incluye una granularidad elemental B con respecto a $\trianglelefteq$, junto con un conjunto de reglas que permiten derivar las granularidades de $\mathcal{C}$ a partir de B. Un calendario $\mathcal{C}$ lo podemos representar como un par $(\mathcal{G}, \mathcal{E})$ donde $\mathcal{G}$ son las granularidades y $\mathcal{E}$ las relaciones entre las granularidades.

Un calendario $\mathcal{C}$ se genera recursivamente a partir de una granularidad elemental B, de modo que :

- (B,{}) es un calendario,
- sea $(\mathcal{G}, \mathcal{E})$ un calendario y e una regla en función de las granularidades de $\mathcal{G}$ que nos define una granularidad G, tal que $G \notin \mathcal{G}$, entonces $(\mathcal{G} \bigcup \{G\}, E \bigcup e)$ es un calendario.

Según esta definición, todos los calendarios se originan a partir de una granularidad elemental, de modo que para nueva granularidad del calendario se añade la granularidad junto con una regla que permite obtener esta granularidad a partir de las ya existentes.

3.2.2. Limitaciones de los modelos de tiempo existentes

Los modelos propuestos en [Bet96, Bet98, Bet98b, Bet00], son muy generales ya que permiten describir modelos de tiempo lineal, ramificado o cíclico, donde

los índices que se asignan a cada granularidad pueden no ser continuos. Además permiten huecos entre granularidades, por ejemplo una granularidad podría ser los lunes, con lo cual hay un hueco de 6 días entre cada granularidad de lunes.

Por otro lado, el Lenguaje Natural utiliza el calendario Gregoriano, en el que se suelen utilizar granularidades cuyas etiquetas se repiten a lo largo del tiempo, y cuyos valores dependen de una granularidad superior. Por ejemplo, los meses se indexan del 1 al 12, de modo que después del 12 se ha completado un año y volvemos a comenzar con el mes 1.

En el modelo general propuesto en [Bet00] los índices pueden tomar cualquier valor perteneciente al conjunto de los números enteros. Esta limitación se encuentra modificada en el modelo temporal de [Ning01]. En este modelo se define una *granularidad etiquetada* como aquella granularidad cuyos valores están acotados en un subconjunto de los números enteros. En este modelo un instante se representa mediante un *vector de etiquetas*, o sea una secuencia de índices donde cada índice representa el valor de una granularidad. Este modelo es válido cuando hay una forma canónica de representar las expresiones temporales. Si se sabe que todas las expresiones temporales van a utilizar el patrón (año, mes, domingo), 'el segundo domingo de julio de 1998' se representa con la tupla '(1998,7,2)' . Sin embargo, en Lenguaje Natural, no se tiene una forma canónica de representación del tiempo, sólo sabemos que se utilizan un conjunto de granularidades, con unas relaciones de orden y de equivalencia entre ellas definidas por el calendario Gregoriano. Por ello el modelo propuesto en [Ning01] no es aplicable a nuestro escenario.

3.3. Modelado del tiempo para Lenguaje Natural

El modelo de tiempo que vamos a describir se basa en el calendario Gregoriano y por tanto sólo será válido para aquellas lenguas que se apoyan en él, principalmente las lenguas occidentales. Vamos a intentar definir nuestro modelo según el modelo general propuesto en [Bet00].

El calendario Gregoriano nos permite modelar el tiempo absoluto, utilizando un modelo de tiempo lineal discreto con gránulos contiguos, de modo que entre cada gránulo podemos establecer una relación de orden total en la dirección temporal del pasado al futuro. Esto nos permite poder identificar cada punto mediante un índice, de modo que podemos realizar un isomorfismo entre el espacio de los números enteros y cada uno de estos instantes.

Las unidades de medida del calendario Gregoriano, las podemos clasificar en las que dependen del periodo de rotación de los astros como son los *días, meses(30 días), años(365 días), semanas(7 días)*, y los que se forman como grupos

regulares de años cómo son *décadas(10 años), siglos(100 años) y milenios(1000 años)*. Hay que tener en cuenta que existe una relación de particionamiento entre las unidades de medida que dependen del movimiento periódico de los astros, la cual es transitiva y periódica.

Nuestro propósito es realizar un modelo de tiempo para poder expresar computacionalmente las expresiones temporales que se utilizan en Lenguaje Natural. Es decir, para expresar instantes de tiempo cronológico basados en el calendario Gregoriano y donde además no existen huecos entre los gránulos de una granularidad. Esta continuidad entre los gránulos da lugar a que si tenemos una granularidad etiquetada con $\mathcal{L} \equiv \mathbb{Z}$, la imagen de esa granularidad en el dominio temporal es igual a su extensión, o sea, a todo el dominio del tiempo. Un gránulo que pertenece a una granularidad con $\mathcal{L} \equiv \mathbb{Z}$ referencia un único punto en el dominio temporal, es una referencia absoluta a un instante de tiempo cronológico determinado. Por ello denominaremos a este tipo de granularidad **granularidad absoluta**.

Vamos a introducir algunas definiciones:

- Definiremos **granularidad absoluta** a una granularidad etiquetada $G = (g, \mathcal{L})$ con $\mathcal{L} \equiv \mathbb{Z}$.
- Sea $\mathcal{C} = (\mathcal{G}, \mathcal{E})$ un calendario, y G y $H \in \mathcal{G}$. Diremos que H es una granularidad **superior** a G, si $G \barwedge H$ $y \nexists K \in \mathcal{G}$ que cumpla que $G \trianglelefteq K \trianglelefteq H$.
- Sea $\mathcal{C} = (\mathcal{G}, \mathcal{E})$ un calendario, y H una granularidad superior a G, con $G = (g, \mathcal{L})$. Denotaremos $\mathcal{L}_g$ al conjunto de etiquetas que puede tomar g respecto a la granularidad superior H.

Con estas definiciones las granularidades del calendario Gregoriano se generan a partir de la agrupación de días. En el calendario Gregoriano los años, siglos y los milenios son granularidades absolutas, ya que pueden indexarse con cualquier valor entero. Sin embargo los meses son granularidades relativas a años, un año tiene 12 meses, por lo cual $\mathcal{L}_{año}$=1..12. Es decir la granularidad de año es superior a la granularidad de mes. Por otro lado aunque un año es absoluto, como particiona a otras granularidades, a veces actúa como relativo, un año con respecto a décadas es relativo, ya que una década está formada por 10 años.

Las granularidades etiquetadas que vamos a utilizar en nuestro modelo son: *día(i), díames(d), díasemana(x), semana(w), mes(m), trimestre(t), cuatrimestre(q), semestre(e), año(y), década(z), siglo(s) y milenio(l)*, a las cuales denotaremos por la letra minúscula entre paréntesis.

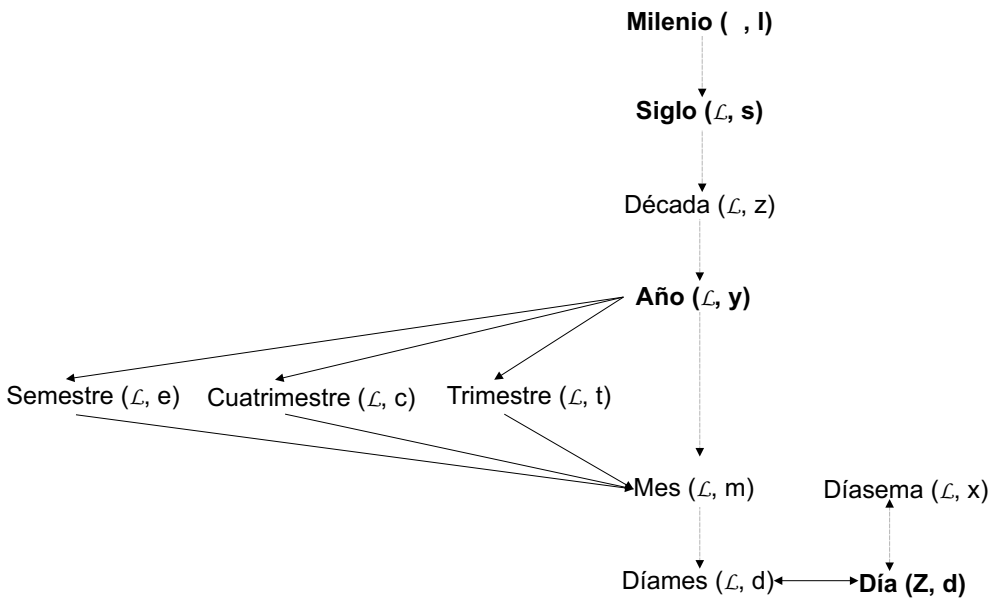


Figura 3.1: Grafo dirigido por la relación $\overline{\wedge}$ entre las granularidades del calendario Gregoriano.

Nuestro **dominio del tiempo** estará formado por un par $(\mathbb{Z}, \leq)$ donde $\leq$ es una relación de orden total entre los números enteros. En este dominio definiremos el calendario Gregoriano cómo:

$$\mathcal{C} = (\mathcal{G}, \mathcal{E}) \tag{3.1}$$

donde:

- $\mathcal{G} = \{(i, \mathbb{Z}), (x, 1.,7), (y, \mathbb{Z}), (z, 1.,10), (s, \mathbb{Z}), (l, \mathbb{Z}), (m, 1.,12), (e, 1.,2), (c, 1.,3), (t, 1.,4), (d, 1..L_d), (w, 1..L_w)\}$, o sea G está formado por un conjunto de granularidades etiquetadas $(g, \mathcal{L}_g)$,
- la granularidad terminal de $\mathcal{C}$, es la granularidad i según la relación de partición $\overline{\wedge}$ y,
- $\mathcal{E}$ son las reglas que relacionan las granularidades entre sí para formar el calendario, las cuales se describen en la tabla 3.2. En las granularidades no absolutas se han indicado los rangos que pueden tomar las etiquetas de la granularidad g mediante $\mathcal{L}_g$.

Cabe destacar que las granularidades i,d,y x, correspondientes a día, día de la semana y día del mes, son granularidades equivalentes. Todas ellas tienen el mismo gránulo, el gránulo de días, pero para diferenciar entre días de la semana, días del mes y días absolutos hemos creído conveniente diferenciar entre estos tres

Granularidad	**Definición**
$(x,\ 1..7)$	$x \leftrightarrow i,\ \exists\, j\ \in\ \mathbb{Z},\ k \in 1.,7\ tal\ que\ k = mod(j/7) + 1$
$(y,\ \mathbb{Z})$	$\pi(y, i) = 400$
$(z,\ 1..10)$	$\pi(\mathbb{Z}, y) = 1, 10$
$(s,\ \mathbb{Z})$	$\pi(s, y) = 1, 100$
$(l,\ \mathbb{Z})$	$\pi(l, y) = 1, 1000$
$(m,\ 1..12)$	$\pi(m, y) = 12$
$(s,\ 1..2)$	$\pi(m, s) = 6$
$(c,\ 1..3)$	$\pi(m, c) = 4$
$(t,\ 1..4)$	$\pi(m, t) = 3$
$(d,\ 1..L_d)$	$d \leftrightarrow i,\ y\ \mathcal{L}_d \in [28.,31]$ dependiendo del instante a la granularidad de años
$(w,\ 1..L_w)$	$w \leftrightarrow i,\ \mathcal{L}_w \in [1.,6]$ dependiendo del instante a la granularidad a nivel de meses

Cuadro 3.2: Reglas de generación del calendario

tipos de granularidades al igual como se hace en Lenguaje Natural para definir un mismo instante con la misma precisión pero con nombres distintos, ('miércoles de la primera semana de mayo', 'mayo día 5', 'día 5 de mayo miércoles','día 731854'). Obsérvese que aunque, 'día 731854', es una expresión válida, en Lenguaje Natural no se utiliza, sin embargo es la granularidad básica para la definición de las otras granularidades del calendario.

Según observamos en las reglas de la tabla 3.2, todas las granularidades superiores a mes se generan por agrupaciones periódicas de un mismo número de granularidades, de modo que el rango de valores de las etiquetas no varía con el tiempo. Sin embargo, aunque todos los años tienen 12 meses, el número de días de un año no es siempre de 365. Ello produce a que existan meses con distintos días, 28, 29, 30 o 31, dependiendo del mes y del año. Por ello los valores que puede tomar $\mathcal{L}$ en las granularidades inferiores a meses dependen del punto temporal, de modo que para conocer la cota de una granularidad g_i , requerimos conocer el valor del punto temporal a nivel de una granularidad superior a g_i. La cota de d depende del punto temporal a nivel de días, por ello 'febrero de 1999' está acotado entre [1..28]. Aunque estos valores dependen del mes y del año, existe una periodicidad cada 400 años.

En la figura 3.1 se ha representado el calendario mediante un grafo, donde se muestra la relación $\overline{\wedge}$ entre las distintas granularidades, y donde la doble flecha representa la equivalencia entre granularidades. En los arcos se ha anotado la relación de periodicidad entre dos granularidades consecutivas en el grafo. Además

se han remarcado en negrita las granularidades absolutas[1].

Debido a la transitividad de la relación $\overline{\wedge}$, una granularidad se puede representar como relativa a todas las granularidades más gruesas ('el tercer año del milenio', 'el tercer día del mes', etc.).

3.4. Entidades temporales

A partir del calendario $\mathcal{C} = (\mathcal{G}, \mathcal{E})$ ya podemos definir las entidades temporales que vamos a utilizar en nuestro modelo, éstas son: puntos, intervalos y duraciones temporales.

3.4.1. Punto de tiempo

En primer lugar expresaremos un *punto de tiempo* como una secuencia alterna de granularidades y números enteros,

$T = g_1 n_1 g_2 n_2 \ \ldots \ g_k n_k$
donde $g_i \in \mathcal{G}$, $n_i \in \mathcal{L}_{g_i}$ y si $i < j$ entonces $g_j \overline{\wedge} g_i$.

El tamaño de los puntos temporales en el espacio temporal discreto, o sea la precisión del instante de tiempo, nos la va a dar granularidad más fina del punto, o sea la última granularidad g_k a la cual denominaremos **grano** del punto temporal, y lo denotaremos cómo $gran(T)$, de modo que $gran(T) = g_k$.

Si g_1 es una granularidad absoluta, diremos que el punto temporal es un instante de tiempo en el dominio del tiempo.

Vamos a definir una relación de orden $\leq$ entre dos puntos temporales con el mismo grano.

Sea $T = g_1 n_1 g_2 n_2 \ldots g_k n_k$ y $T' = g_1 n'_1 g_2 n'_2 \ldots g_k n'_k$ dos puntos,
diremos que $T \leq T'$ si se cumple que $n_i \leq n'_i$ *con* $1 \leq i \leq k$

[1] Nótese que la década siendo una agrupación también de años no es absoluta, ya que depende de su granularidad precedente, el siglo, cuando decimos la *década de los 80* nos podemos referir a *1981-1990 o 1881-1890*, por ello forma parte de las granularidades relativas.

3.4.2. Intervalo de tiempo

Un *intervalo de tiempo* I es el espacio temporal entre dos puntos temporales $T1$ y $T2$ tal que $T1 \leq T2$ y $gran(T1) = gran(T2)$, y se denota como:

$$I = [T1, T2]$$

Esta definición nos permite referenciar al punto inicial $T1$ del intervalo I con **start(I)**, y al punto final $T2$ con **end(I)**. Además por la definición de la relación $\leq$ entre dos puntos, se ha de cumplir que la $gran(T1) = gran(T2)$, así pues definiremos el *grano* de un intervalo como **gran(I)**, donde se cumple que $gran(I) = gran(start(I)) = gran(end(I))$.

3.4.3. Duración de tiempo

Finalmente, definimos una duración de tiempo, como un espacio temporal no anclado en el dominio de tiempo, el cual se mide en función de una granularidad y que se refiere a un periodo futuro (+) o pasado (-), formalmente:

$$S = \pm ng, \text{ con } g \in \mathcal{G} \text{ y } n \in \mathcal{L}_g.$$

Las duraciones de tiempo se pueden utilizar, bien para definir sucesos en función de otras fechas ('dentro de tres días'), o bien en función de otros sucesos ('tres días después de las elecciones').

3.5. Operaciones sobre las entidades temporales

En este apartado describiremos las operaciones necesarias entre las entidades temporales, que nos permitirán comparar entidades con distintas granularidades y distinto grano.

Primero describiremos las operaciones de inicio, final y formateo de un punto de tiempo, las cuales se resuelven mediante la consulta al calendario, y que normalmente están incluidas en la mayoría de sistemas gestores de bases de datos o en las librerías para el manejo de fechas de los lenguajes de programación.

Inicio de un punto respecto a una granularidad inferior .

La operación $first(T, g)$ nos permite obtener el primer valor que puede tomar la granularidad inferior al grano del punto temporal T.

Sean $T = g_1 n_1 g_2 n_2 \ldots g_k n_k$ y $g \in \mathcal{C}$ tal que $g \overline{\wedge} g_k$,
entonces $first(T, g) = T'$
donde $T' = g_1 n_1 g_2 n_2 \ldots g_k n_k g n$ siendo $n = min(\mathcal{L}_{g_k})$

Ejemplos:

```
first(x,y2002m5w1)=x3
first(d,y2002m5)=d1
```

Final de un punto respecto a una granularidad inferior .

La operación $last(T, g)$ nos permite obtener el último valor que puede tomar la granularidad inferior al grano del punto temporal T.

Sea $T = g_1 n_1 g_2 n_2 \ldots g_k n_k$ y $g \in \mathcal{C}$ tal que $g \overline{\wedge} g_k$
entonces $last(T, g) = T'$
donde $T' = g_1 n_1 g_2 n_2 \ldots g_k n_k g n$ siendo $n = max(\mathcal{L}_{g_k})$

Ejemplos:

```
last(x,y2002m5w5)=x5
last(d,y2002m5)=d31
```

Formateo de puntos temporales absolutos .

La relación $\overline{\wedge}$ definida en $\mathcal{C}$ nos permite referenciar un mismo punto temporal absoluto con el mismo grano, utilizando distintas granularidades. Por ejemplo: 'y1999'='l1y999', o 'y1999m1d1'='y1999m1w1x5'= 'l1y999m1w1d5'. La única restricción para la construcción de estos puntos temporales equivalentes es que tengan el mismo grano, o granos equivalentes, y la primera granularidad en los dos puntos sea absoluta.

Por tanto vamos a definir la función de formateo de puntos temporales cómo *format*, de modo que:

Sea $T = g_1 n_1 g_2 n_2 \ldots g_k n_k$ un punto temporal
entonces $format(T, g'_1 g'_2 \ldots g'_k) = T'$
donde $T' = g'_1 n'_1 g'_2 n'_2 \ldots g'_j n'_j$, $g'_j \leftrightarrow g_k$, $\mathcal{L}_{g'_1} = \mathbb{Z}$ y $\mathcal{L}_{g_1} = \mathbb{Z}$.

Ejemplos:

```
format(y1999m1d1,ymwx)=y1999m1w1x5
format(y1999d1,ymd)=y1999m1d1
```

Para permitir obtener fechas a partir de expresiones relativas, definimos tres funciones: el refinamiento, la abstracción y el desplazamiento de entidades temporales.

Refinamiento de entidades temporales .

El refinamiento es una operación que nos permite expresar un punto temporal a una granularidad inferior según $\overline{\wedge}$. Esta operación siempre produce un intervalo temporal. La operación de refinamiento se define de la siguiente manera:

Sea $T = g_1 n_1 g_2 n_2 \ldots g_k n_k$ y g tal que $g \,\overline{\wedge}\, g_k$
entonces $refine(T, g) = [T1, T2]$
donde $T1 = g_1 n_1 \ldots g_k n_k g(first(T, g))$ y $T2 = g_1 n_1 \ldots g_k n_k g(last(T, g))$.

De modo similar, podemos definir el refinamiento de un intervalo: $refine(I, g) = [start(refine(start(I), g)), end(refine(end(I), g))]$. Veamos varios ejemplos:

```
refine(y1999, m)  =[y1999m1, y1999m12]
refine(y2000m3, w) = [y2000m3w1, y2000m3w5]
refine([y2000, y2001], m) = [y2000m1,2001m12]
```

Abstracción de un punto temporal .

La abstracción es la operación inversa al refinamiento y permite expresar un punto temporal mediante una granularidad más gruesa. Esta operación se define cuando la granularidad a la que se se quiere abstraer existe en el punto temporal.

Es importante mencionar que el proceso de abstracción supone una pérdida de información, ya que se pierden los detalles de las granularidades inferiores al eliminarse éstas de la expresión del punto temporal.

Sea $T = g_1 n_1 g_2 n_2 \ldots g_j n_j \ldots g_k n_k$ y $g_j \leftrightarrow g$
entonces $abstract(T, g) = T'$
donde $T' = g_1 n_1 g_2 n_2 \ldots g_j n_j$ y $1 \leq j \leq k$.

Desplazamientos de puntos temporales .

Muchas veces para definir un punto o intervalo temporal se utiliza una duración de tiempo anclada en el pasado o futuro respecto a un punto temporal. Para obtener el punto o intervalo al que se refiere la expresión deberemos realizar una operación de desplazamiento del punto de referencia hacia el pasado o futuro (según el signo de la duración). A continuación definimos esta operación.

$Sea\ T = g_1 n_1 g_2 n_2 \ldots g_k n_k,$
$\quad S = \pm ng\ y\ g_k = g\ entonces$
$shift(T, S) = T + S.$

Aquí el operador $\pm$ es la suma/resta aritmética sobre los valores de la granularidad de la duración. Esta suma debe tener en cuenta el posible acarreo sobre las granularidades anteriores a la granularidad de la duración en el punto temporal. Por ejemplo:

```
shift(y1999m3, +10m) = y2000m1
shift(y1998m2w2, -3w) = y1998m1w4
```

Si se desea refinar(abstraer) un punto de tiempo a una granularidad g para la cual no está definida la operación, se ha de buscar el camino más corto en el grafo de la relación $\overline{\wedge}$ entre la $gran(T)$ y g, de modo que se apliquen sucesivamente operaciones de abstracción o refinamiento, según se suba o baje de nivel en el grafo. Asimismo, cuando la operación de abstracción no está definida por no existir la granularidad en la expresión, deberemos realizar una operación de formateo.

Ejemplos:

```
refine(y2000,d)=refine([start(refine(y2000,m)),end(refine(y2000,m))],d)
abstract(y2000m5d7,w)=abstract(format(y2000m5d7,ymwd))
shift(y1999m5d1,+3w)=shift(abstract(format((y1999m5d1,w),w),+3w))
```

3.6. Conclusiones

Tanto en el lenguaje hablado como en el lenguaje escrito, se utilizan construcciones que involucran granularidades temporales para denotar tanto instantes temporales absolutos (`mayo de 1999`), o relativos (`ese mes`), así como simples duraciones de tiempo (`Durante dos meses`).

En este capítulo hemos introducido un modelo temporal, que nos va a permitir codificar las expresiones temporales basadas en el calendario Gregoriano, para representar las distintas entidades temporales presentes implícita o explícitamente en los documentos. Además con este modelo se puede operar con entidades temporales con distintos niveles de precisión, lo cual nos permite relacionar las distintas expresiones temporales, además obtener en el caso de instantes temporales relativos la fecha o periodo absoluto referenciado.

En el próximo capítulo, mostraremos cómo traducir las expresiones en Lenguaje Natural a entidades de nuestro modelo del tiempo, además de cómo determinar el instante absoluto utilizando las operaciones definidas en este capítulo entre entidades temporales, utilizando como una fecha de referencia según el punto de referencia del hablante para completar entidades que no poseen una granularidad absoluta, o realizar desplazamientos con las duraciones. Según el punto de referencia utilizaremos la fecha de publicación o un instante absoluto calculado anteriormente.

Capítulo 4

Extracción de Información Temporal

En este capítulo vamos a plantear cómo extraer automáticamente la información temporal presente en documentos de ámbito general . Es decir, vamos a definir una herramienta que nos permita extraer las fechas y periodos presentes en los documentos tanto explícitamente mediante un formato de fecha, o bien implícitamente mediante expresiones temporales.

En la literatura podemos encontrar aplicaciones que requieren conocer la información temporal presente en los documentos. Estos sistemas identifican algunos tipos de expresiones temporales con bastante éxito, pero son aplicaciones diseñadas para un ámbito muy determinado. A modo de ejemplo presentamos tres sistemas:

- En [Kni98] se presenta un sistema para el análisis de documentos legales. Este sistema extrae las expresiones temporales para comprobar las consistencias temporales entre ellas, pero no intenta reconocer el instante de tiempo absoluto asociado al documento, y generalmente trata con expresiones temporales con granularidades de nivel inferior o igual a día.
- El sistema Verbomovil [Ste98], gestiona automáticamente las citas entre dos usuarios. Este sistema extrae las expresiones temporales, y las codifica formalmente para posteriormente obtener la fecha referenciada. El contexto en el que se aplica este sistema (diálogos) da lugar a que no exista el problema de ambigüedad para reconocer el punto de referencia del hablante, y además la fecha de referencia, siempre es la fecha anteriormente citada. En el contexto de este sistema tampoco se producen expresiones temporales con imprecisiones (ej. 'durante varios días').

- También hemos encontrado un sistema de análisis de expresiones temporales en documentos de ámbito general en inglés [Koe00], para el cual todas las expresiones relativas se tratan como duraciones.

Visto que las aplicaciones encontradas en la literatura estaban enfocadas a aplicaciones muy determinadas, y no permiten extraer toda la información del tiempo cronológico presente en los documentos, hemos decidido crear nuestro propio sistema de extracción de entidades temporales presentes en los documentos. Así, hemos creado herramienta TimExtractor, que permite detectar las expresiones temporales presentes en los documentos para codificarlas semánticamente utilizando una notación que hemos denominado CodTemp. Una vez detectada una expresión temporal, el sistema analiza semánticamente la expresión codificada y trata de obtener la fecha o periodo referenciado.

Las especificaciones de la herramienta TimExtractor son dos. Primero, el sistema debe ser capaz de detectar todas las expresiones que denotan un instante temporal y reconocer la localización del instante en el dominio temporal. Segundo, este sistema tiene que ser fácilmente trasladable a otros idiomas que utilicen el calendario Gregoriano. En la actualidad, y principalmente por falta de un repositorio con los sucesos más relevantes, no hemos sido capaces de obtener el tiempo que se referencia en las expresiones relativas a otros sucesos (ej. 'tres días después de la firma').

TimeExtractor es un sistema para la detección y reconocimiento de las expresiones temporales, el cual añade al documento original, alrededor de las expresiones temporales detectadas, la etiqueta XML TIMEX, siguiendo las directivas de etiquetado de las expresiones temporales definidas en la tarea de *Name Entity* del MUC [Chi97] y del HUB [Hir99] (ver apartado 4.3). En nuestra herramienta esta etiqueta tiene dos atributos: Type que indica el tipo de expresión, y Value que representa la expresión temporal codificada semánticamente o bien el punto o intervalo temporal absoluto que se referencia en la expresión.

Para facilitar la portabilidad a otros idiomas y la obtención del tiempo absoluto referenciado en las expresiones, TimeExtractor posee dos módulos que operan en cascada: TagTimex y ModelTimex. El módulo TagTimex se encarga de detectar las expresiones temporales y representarlas semánticamente utilizando la notación CodTemp. El módulo ModelTimex analiza la representación semántica de la expresión para obtener el tipo de expresión temporal y calcular el tiempo absoluto de referencia. Así pues, el módulo ModelTimex es independiente del idioma, con lo cual sólo se requiere adecuar el módulo TagTimex a la lengua en la cual están escritos los documentos.

En la proxima sección vamos a analizar y clasificar las expresiones temporales. La herramienta TimeExtractor la describiremos en el apartado 4.2. En la sección

4.2.1.1 definiremos la notación CodTemp, una notación formal independiente de la lengua para representar semánticamente las expresiones temporales. En el apartado 4.3 describiremos las líneas de trabajo que se han realizado en el área de extracción de información temporal. Y en el apartado 4.3.1, hemos realizamos una revisión de varios sistemas de representación del conocimiento temporal.

4.1. Expresiones Temporales

Al igual que ocurre en los sistemas métricos que están basados en una unidad básica de medida pero que soportan distintas unidades de medida, el tiempo posee como unidades de medida las granularidades. Por ello en las expresiones en Lenguaje Natural (LN) se suelen utilizar distintas granularidades para referenciar el mismo instante de tiempo. Según se cita en [Bet00], ello es debido a dos razones. Por un lado, y por conveniencia, es más sencillo decir 10 años que 3652 días. Y por otro lado, porque las granularidades nos permiten aludir a instantes de tiempo con cierta imprecisión. Cuando decimos 'Pepe lavó el coche este fin de semana', no significa que estuvo dos días lavando el coche, nos indica con una imprecisión de dos días cuándo lavó el coche.

Según el tipo de información que representan las expresiones temporales las podemos clasificar de la siguiente forma:

- Expresiones que no representan entidades temporales:
  - Frases hechas que no denotan ningún suceso como 'el fin de los días' o 'hoy por hoy'.
  - Expresiones periódicas, como 'cada día' o 'todos los lunes'.
- Duraciones no ancladas que expresan una edad o la duración de un suceso, acción, o fenómeno: 'requiere dos horas para su cocción', 'estuvo en la cárcel durante tres días', 'tiene tres meses de edad', etc.
- *Expresiones de tiempo* que representan una localización en el dominio temporal y que podemos clasificar en otros dos grupos.
  - *Expresiones temporales absolutas*, las cuales expresan un punto o intervalo temporal que puede estar fijo o no, en el tiempo. Estas a su vez se clasifican en:
    - `deícticas`: 'hoy', 'mañana', 'este año'.

  - `completas`: 'el día 5 de mayo de 1999', '3-4-1999', 'el día 3 y 4 de los meses de mayo y junio de 1999', 'la década de los 80'.
  - `incompletas:` que requieren conocer el punto temporal del hablante, y la dirección temporal. Ej. 'el próximo mes de mayo', 'el lunes se celebrará', 'el lunes anterior'.

- *Expresiones temporales relativas*, que se representan mediante una entidad temporal de tipo duración que debe tener explícita la dirección temporal. Las expresiones temporales pueden ser relativas:

  - al punto de referencia del hablante: 'dentro de tres días', 'antes de tres días', 'algunos días antes', 'este mes', 'esta mañana'.
  - a una fecha citada anteriormente: 'durante esos días', 'ese mes', 'al día siguiente'.
  - a sucesos: 'desde la firma del contrato', ' tres días antes de la firma del contrato'.
  - a expresiones temporales con granularidad inferior a días, que representan un periodo de tiempo de una duración menor del día, y que permiten enfatizar una fecha anteriormente citada o la fecha de publicación sin mencionarla: 'durante las tres horas primeras de reunión', 'esa mañana', 'a los pocos minutos'.

- Expresiones de tiempo complejas, o sea expresiones temporales que se componen de otras expresiones temporales para aludir a un instante de tiempo:

  - intervalos: 'entre el lunes y el martes', 'de mayo a junio', '1999-2000', 'hasta el próximo jueves'.
  - secuencias: 'el día 1 y 2', 'el miércoles y el viernes'.
  - compuestas: 'en mayo de este año', 'el día 5 de mayo', 'el 2 y 3 de ese mes'.

Las frases hechas y las expresiones periódicas al no representar ninguna de las entidades del modelo de tiempo descrito en el capítulo anterior (una fecha, intervalo o duración), además de no ser muy relevantes para relacionar sucesos, no van a ser detectadas por nuestro sistema. Sin embargo aunque las duraciones de tiempo no referencian directamente una fecha o intervalo, son muy importantes ya que se utilizan en Lenguaje Natural

- bien para definir periodos de tiempo donde lo importante no es el tiempo cronológico sino la duración de un suceso (ej. 'estudió inglés cinco años'),

- o bien para definir sucesos:
  - en función de otras fechas: 'dentro de tres días',
  - o en función de otros sucesos: 'tres días después de las elecciones',
  - o para representar acciones periódicas: 'cada semana'.

Aunque en principio sólo nos interesan las duraciones que referencian una fecha, debido a la complejidad de discernir entre éstas y las duraciones propiamente dichas, el sistema TagTimex etiqueta ambos tipos de expresiones, y el módulo ModelTimex se encarga después de analizar cada una de ellas para detectar de qué tipo de expresión se trata.

4.2. TimeExtractor

TimeExtractor es un sistema de extracción y reconocimiento de entidades temporales multilingüe. Tras el análisis con este sistema, se añade al documento original la etiqueta TIMEX alrededor de las expresiones temporales con dos atributos: Type y Value. Mediante el atributo Type se clasifican las expresiones temporales en tres tipos: *DATE, EVENT* y *DURATION*.

Todas aquellas expresiones temporales a las que que el sistema es capaz de asignar un instante de tiempo cronológico, se clasifican como de tipo *DATE*. Las expresiones relativas a sucesos se clasifican como de tipo *EVENT*, y todas las demás como de tipo *DURATION*. El sistema no es capaz de resolver algunas expresiones relativas debido a que no es capaz de asignarlas una fecha de referencia, o una dirección temporal.

El valor de la etiqueta Value depende del tipo de expresión temporal. Si se trata de una expresión temporal de tipo *DATE*, en Value tendremos el instante de tiempo representado como una entidad temporal de tipo punto o intervalo del modelo de tiempo descrito en el capítulo anterior. En los otros casos, esta etiqueta contendrá la codificación semántica de la expresión temporal utilizando la notación CodTemp que se detalla en la subsección 4.2.1.1.

Para conseguir que TimeExtractor sea fácilmente portable a otros idiomas, hemos creado dos módulos que operan en cascada siguiendo de alguna manera la arquitectura propuesta en Verbomovil [Ste98]. El primer módulo TagTimex, es el encargado de extraer las expresiones temporales, y etiquetarlas con la etiqueta TIMEX, añadiendo en el atributo Value la representación semántica. Este módulo es dependiente de la lengua ya que extrae las expresiones temporales utilizando un análisis sintáctico-semántico superficial.TagTimex no reconoce el tipo de expresión de que se trata. El segundo módulo ModelTimex, opera en cascada y trata

de reconocer el tipo de expresión temporal, analizando la notación semántica para obtener su localización en el tiempo absoluto. Gracias a que este módulo requiere sólo la codificación semántica es independiente del idioma, con el cual está escrito el texto y es válido para cualquier documento que utilize el calendario Gregoriano.

4.2.1. TagTimex: Etiquetador de expresiones temporales

El módulo TagTimex busca expresiones temporales en un textoy las etiqueta añadiendo una representación semántica independiente del idioma. Este módulo no trata de analizar el significado de las expresiones temporales, simplemente las codifica semánticamente para su posterior análisis con el módulo ModelTimex. Éste será el encargado de discernir de qué tipo de expresión se trata, y si es posible identificará el instante al que se alude.

Un análisis de las expresiones temporales nos ha llevado a la siguiente conclusión: toda expresión temporal, o bien tiene un patrón numérico de fecha (por ejemplo 1999/04/05) o bien tiene un conjunto de palabras que las identifican, y que están directamente relacionadas con las granularidades temporales. A las palabras de este conjunto las denominaremos *núcleos temporales* (Ver tabla 4.1).

En la sección 3.4 vimos que una entidad temporal está formada por una o varias granularidades cuantificadas mediante un número entero. En el caso de una entidad de tipo punto o intervalo, el número nos sirve para fijar una granularidad, mientras que en el caso de duraciones nos indica la propia duración. En las expresiones temporales las granularidades vienen a veces cuantificadas por la misma palabra que identifica la expresión temporal, o sea el *núcleo temporal* ('lunes', 'mayo'). En general necesitaremos cuantificadores, que pueden ser bien dígitos ('día 2'), adjetivos indefinidos ('algunos días'), adjetivos numerales cardinales ('dos días') o bien ordinales ('segundo día'). Los *cuantificadores temporales* utilizados en nuestra aplicación para el análisis de las expresiones temporales se muestran en la tabla 4.2.

Categoría	Término	Código	Categoría	Término	Código
grano	día	d	*diasem*	lunes	x1
grano	día_semana	x	*diasem*	martes	x2
grano	semana	w	*diasem*	miércoles	x3
grano	mes	m	*diasem*	jueves	x4
grano	trimestre	t	*diasem*	viernes	x5
grano	cuatrimestre	c	*diasem*	sábado	x6
grano	semestre	e	*diasem*	domingo	x7
grano	año	y	*nmes*	enero	m1
grano	década	z	*nmes*	febrero	m2
grano	siglo	s	*nmes*	marzo	m3
grano	milenio	l	*nmes*	abril	m4
nucleo	fecha	d	*nmes*	mayo	m5
nucleo	hoy	ny#nm#nd	*nmes*	junio	m6
nucleo	ayer	-1ny#nm#nd	*nmes*	julio	m7
nucleo	anoche	-1ny#nm#nd	*nmes*	agosto	m8
nucleo	mañana	+1ny#nm#nd	*nmes*	septiembre	m9
nucleo	anteayer	-2ny#nm#nd	*nmes*	octubre	m10
nucleo	Purificación	m12#d10	*nmes*	noviembre	m11
nucleo	Fallas	m3#d19	*nmes*	diciembre	m12
nucleo	Purísima	m12#d10	*partedia*	madrugada	rd
nucleo	víspera	r-1d	*partedia*	mañana	rd
nucleo	Navidades	I0y#m12#d25,+r1y#m1#d5	*partedia*	mediodía	rd
nucleo	primavera	I0y#m3#d22,y#m6#d21	*partedia*	tarde	rd
nucleo	verano	I0y#m6#d22,y#m9#d21	*partedia*	noche	rd
nucleo	otoño	I0y#m9#d22,y#m12#d21	*partedia*	momento	rd
nucleo	invierno	I0y#m12#d22,+r1y#m3#d21	*partedia*	jornada	rd

Cuadro 4.1: Codificación de núcleos temporales.

Categoría	Término	Código	Categoría	Término	Código
indefinido	demasiados	' '	*cuant*	un	1
indefinido	algunos	' '	*cuant*	dos	2
indefinido	algunas	' '	...	...	...
indefinido	varios	' '	*cuant*	veintiocho	28
indefinido	varias	' '	*cuant*	veintinueve	29
indefinido	unos	' '	*cuant*	treinta	30
indefinido	decenas	' '	*cuant*	cuarenta	40
indefinido	unas	' '	...	...	...
indefinido	los	' '	*cuant*	noventa	90
indefinido	las	' '	*cuant*	cien	100
indefinido	pocos	' '	*cuant*	una	1
indefinido	pocas	' '	*cuant*	uno	1
indefinido	muchas	' '	*cuant*	primer	o1
indefinido	muchos	' '	*cuant*	primero	o1
indefinido	tantos	' '	*cuant*	primera	o1
cuant	par	2	...	...	...
cuant	quince	15	*cuant*	décimo	o10
cuant	docena	12	*cuant*	décima	o10

Cuadro 4.2: Codificación de los cuantificadores temporales.

Excepto en el caso de expresiones absolutas completas, para resolver la expresión temporal se requiere conocer la dirección temporal y un punto de referencia. Generalmente el punto de referencia viene determinado por adjetivos demostrativos (ej. 'este' indica punto de referencia del hablante, mientras 'ese' indica que es relativo a fecha citada anteriormente). Para conocer la dirección temporal, si habla acerca del pasado, presente o futuro, en Lenguaje Natural hacemos uso del tiempo verbal ('hace dos días'), o de modificadores ('el día anterior'). En la tabla 4.3 hemos detallado los *modificadores temporales* que se requieren para poder analizar las expresiones temporales. Además, se han incluido algunos modificadores que nos indican una parte de un gránulo o intervalo temporal ('principios de verano', 'finales de año', 'fin de semana'), o bien modificadores que nos permiten distinguir entre un conjunto de instantes y un intervalo ('entre el lunes y el viernes').

Categoría	Término	Código	Categoría	Término	Código
modificador	saliente	n	*mods*	cuestión de	I+
modificador	entrante	1+n	*mods*	han sido	I-
modificador	antes	r-	*mods*	va de	nO
modificador	atrás	-	*mods*	a lo largo de todo el	r
modificador	después	r+	*mods*	a lo largo de	r
modificador	actual	n	*mods*	que viene	+
modificador	corriente	n	*mods*	poco más de	' '
modificador	presente	n	*mods*	poco más tarte	+
modificador	ya	P-	*mods*	más tarde	+r
modificador	principio	A	*mods*	dentro de	+
modificador	fin	F	*mods*	más de	' '
modificador	final	F	*mods*	menos de	' '
modificador	finales	PF	*mods*	al menos	' '
modificador	principios	PA	*mods*	un par de	2
modificador	inicios	PA	*mods*	ha sido	P-
modificador	inicio	A	*mods*	antes de	' '
modificador	primeros	PA	*mods*	después de	' '
modificador	últimas	P	*mods*	de antelación	r-
modificador	último	P-	*verbos*	hace	O-
modificador	última	P-	*verbos*	hacía	IO-
modificador	recientes	P-	*verbos*	lleva	IO-
modificador	siguiente	+	*verbos*	llevaba	IO-
modificador	nueva	+	*verbos*	llevaban	IO-
modificador	anterior	-	*verbos*	llevan	IO-
modificador	posterior	+	*verbos*	llevo	O-
modificador	anteriores	-	*verbos*	faltan	O+
modificador	posteriores	+	*verbos*	dejará	O+
modificador	ese	Or	*verbos*	fue	-
modificador	esta	n	*verbos*	dura	I+
modificador	aquellas	Orp	*verbos*	durará	I+
modificador	misma	Or	*verbos*	duraron	Ir
modificador	pasadas	O-	*verbos*	tardaron	Ir
modificador	apenas	' '	*verbos*	han	O-
modificador	sólo	' '	*verbos*	harán	O+
modificador	casi	' '	*verbos*	tendrán	O+
modificador	durante	' '	*verbos*	tienen	O+
modificador	para	' '	*verbos*	fueron	-r

Cuadro 4.3: Codificación modificadores temporales.

Así pues, las palabras que forman parte de una expresión temporal se clasifican en:

- *Núcleos temporales.* Denotarán aquellas palabras relacionadas con las granularidades temporales. Éstos son las granularidades, los días de la semana, las partes del día ('mañana', 'tarde'), los meses del año, los periodos festivos, las estaciones del año y las deíxis como ('hoy', 'ayer', 'mañana', 'anteayer').

- *Cuantificadores temporales.* Llamaremos así a todos los adjetivos cardinales, ordinales e indefinidos, así como a los números romanos.

- *Modificadores temporales.* Son las formas gramaticales involucradas en una expresión temporal. Éstos los podemos clasificar en:

  - palabras que indican que la expresión es un intervalo. (Ej. 'en', 'durante'),
  - palabras que indican la dirección temporal, como son los adverbios de tiempo (ej. 'después', 'antes'), y algunas formas verbales muy usuales (ej. 'hace', 'duró'),
  - palabras que indican una parte de un intervalo temporal (ej. 'principio', 'final', 'mediados', 'mitad').

Para la detección de las expresiones temporales, hemos generado una gramática que contiene aquellas producciones que nos permiten extraer todos los tipos de expresiones temporales. Esta gramática debe tratar de solucionar las distintas ambigüedades que se producen en los niveles del Lenguaje Natural:

- A nivel semántico debe ser capaz de diferenciar:
  - entre algunos nombres propios y núcleos temporales como 'julio' o 'domingo', desechando como posible expresión temporal aquellas expresiones con dos palabras continuas en mayúsculas.
  - Diferenciar entre siglas y siglo. ej.: 'en el XX Congreso' o 'el siglo XX'.
  - Diferenciar entre años y códigos. Ej. 'En 1999', 'expediente 2000'.
  - Diferenciar entre algunos conceptos con distintos significados. Ej. 'mañana se celebra', 'por la mañana'.
- A nivel sintáctico existen problemas para reconocer expresiones complejas. Por ejemplo, a veces resulta difícil diferenciar entre un intervalo, una secuencia de fechas o una fecha. Ej. 'del martes al jueves', 'un 2 % el día 4 y un 5 %'.

- A nivel de contexto destacaremos dos tipos de ambigüedades. Por un lado el discernir entre expresiones temporales que representan entidades de nuestro modelo de tiempo, o bien expresiones comunes y frases hechas. Y por otro lado, reconocer cual de las fechas citadas anteriormente es la fecha de referencia.

Las producciones de esta gramática han sido generadas mediante una aproximación molecular. El análisis de las expresiones temporales nos ha permitido

obtener los patrones de todas las expresiones temporales simples, y además, hemos construido los patrones de las expresiones temporales más complejas. Entre los patrones temporales de las expresiones temporales simples tenemos las reglas que reconocen expresiones comunes y nombre propios que se confundirían con entidades temporales, además de las expresiones absolutas y relativas. Todas estas reglas se han ordenado de modo que se trate primero de rechazar las expresiones comunes y los posibles nombres propios, y posteriormente buscar las expresiones más largas, las expresiones complejas, dando prioridad a las expresiones temporales absolutas frente a las relativas.

En la subsección 4.2.2 se describen con detalle las producciones y reglas utilizadas en la gramática para la detección de expresiones temporales. Esta gramática se ha implementado con Sictus Prolog [Sic97], pudiendo encontrar el código completo en el Apéndice B. Para facilitar la multilingualidad del sistema, este modulo se ha implementado en dos ficheros Prolog: Tagtime.pl y Base_lexico.pl. Tagtime.pl es el fichero principal, donde tenemos las producciones independientes del idioma y que se encarga de analizar la estructura del documento y etiquetar las expresiones. En Base_lexico.pl se encuentran las producciones dependientes del idioma, los reglas para reconocer las expresiones temporales absolutas y relativas simples, y la base de hechos, o sea las palabras presentes en las expresiones temporales: núcleos, modificadores y cuantificadores, junto con su codificación semántica.

Cuando el módulo TagTimex detecta una expresión que valida una regla correspondiente a una expresión temporal, añade la etiqueta TIMEX al documento, y asigna al atributo *Value* la notación semántica correspondiente. La regla gramatical que valida una expresión se encarga de crear la notación semántica concatenando en un determinado orden las codificaciones semánticas de las palabras que forman parte de la expresión. Al no reconocer el módulo TagTimex el tipo de entidad temporal codificada, no se asigna ningún valor a la etiqueta *Type*.

4.2.1.1. Notación para la Codificación Semántica de Expresiones Temporales

Para la codificación semántica de las expresiones temporales se ha utilizado como base la notación del modelo de tiempo del apartado 3.3. En la tabla 4.4 se encuentran los códigos directamente relacionados con el modelo del tiempo y su significado semántico.

La complejidad de las expresiones temporales ha requerido la definición de códigos que no hemos utilizado en el modelo del tiempo, pero necesarios para representar la complejidad de las expresiones en Lenguaje Natural. Estos códigos nos permiten diferenciar entre los siguientes elementos:

Código	Significado
l	milenio
s	siglo
z	década
y	año
c	cuatrimestre
t	trimestre
m	mes
w	semana
d	día
x	día semana
O	indica presente
-	indica pasado
+	indica futuro
I	indica intervalo
,	separador entidades

Cuadro 4.4: Codificación basada en el modelo de tiempo.

- Expresiones relativas a la fecha anterior, al punto de referencia del hablante, o a un suceso. Por ejemplo, 'ese día', 'el día de la fiesta', o bien 'este día'. Una **r** indicará relativo a fecha anterior, una **n** relativo a punto de referencia del hablante, y una **R** indicará que se trata de una expresión relativa a un suceso.
- Granularidades en singular y plural. Por ejemplo, 'hace días', 'el día de antes'. Si no existen cuantificadores, la granularidad en plural indica una indeterminación cuantitativa, mientras que si se halla en singular significa una única granularidad. Por ello añadiremos una **p** para denotar una granularidad en plural.
- Numerales y ordinales. Por ejemplo, 'el segundo día', frente a 'dos días'. Una **o** delante del número indica que éste es un ordinal.
- Intervalo o una secuencia de puntos temporales. Por ejemplo, 'del lunes al viernes' frente a 'el lunes y el viernes'. Una **I**[1] indica intervalo.
- Expresiones simples de expresiones compuestas, como por ejemplo 'en mayo y este año' frente 'en mayo de este año'. Utilizaremos un # para separar las expresiones simples de una expresión compuesta, y facilitar su posterior análisis con ModelTimex.
- Referencias a una fecha o a un intervalo abierto. Por ejemplo, 'dentro de unos días' , representa una fecha en el futuro, mientras que 'durante los próximos

[1]En el modelo temporal lo representábamos con '[ ]', pero para facilitar el procesado de las expresiones lo hemos convertido a un único símbolo

días' representa un intervalo que comienza en el presente y termina en el futuro. El código para representar un intervalo abierto es una **P**

Código	Significado	Ejemplo
p	granularidad en plural	'días' ⇒ *pd*
o	número indica ordinal	'primer día' ⇒*o1d*
P	periodo	'durante los próximos días' ⇒*+Ppd*
A	principio de un intervalo	'comienzos de mayo' ⇒ *Apm5*
F	final de un intervalo	'fin de mayo' ⇒ *Fm5*
n	relativo a la fecha de publicación	'este año' ⇒ *nOy*
r	relativo a una fecha	'ese día' ⇒ *rd*
R	relativo a un suceso	'dos días después del 20J '⇒ *R+2pd*
#	separador expresiones simples	'tercer día de este mes'⇒ *o3d#n0m*
I	intervalo temporal	'entre el lunes y el jueves'⇒*Ix3,x5*

Cuadro 4.5: Codificación semántica.

En las tablas 4.1, 4.2, y4.3 se encuentra el léxico de palabras en español, que se requieren para extraer las expresiones temporales. En la columna 'código', hemos codificado el significado semántico de los vocablos según la notación propuesta en las tablas 4.4 y 4.5. Además para la correcta aplicación de las gramáticas a las expresiones temporales, hemos agrupado las palabras de cada clase en distintas categorías según los patrones sintácticos en los que se utilizan estas palabras.

A continuación, a modo de ejemplo, mostramos cómo se codifican semánticamente algunas expresiones temporales, agrupadas según el tipo de entidad temporal que representan.

- Puntos temporales:
  - formatos de fecha estándar
    '3 de mayo del 1970'⇒ $y1970\#m5\#d3$,
  - festividades señaladas como 'Navidad'⇒ $m12d25$,
  - deíxis como 'hoy'⇒ $nOy\#nOm\#nOd$ o 'ese mes'⇒ rm,
  - puntos temporales sin granularidad absoluta que dependen de la fecha de publicación como 'el próximo día 15'⇒ $+d15$,
  - puntos temporales relativos a un intervalo como 'el primer día del mes'⇒ $m\#o1d$.
- Intervalos temporales:
  - periodos temporales con nombre, como 'otoño'⇒ $Im9d22, m11d21$,

  - intervalos formados por dos puntos temporales como 'del lunes al miércoles' $\Rightarrow Iw0\#d1, w0\#d3$,
  - intervalos dentro de otros intervalos como 'a principios de este mes'$\Rightarrow An0m$.

- Duraciones:

  - expresiones relativas a la fecha de publicación que dependen del tiempo verbal, como 'hace dos años'$\Rightarrow -2y$,
  - expresiones relativas a otros sucesos, como 'tres días antes de las elecciones'$\Rightarrow R-3dp$,
  - periodos de tiempo no anclados como 'durante cinco años'$\Rightarrow P5y$, 'dentro de quince días'$\Rightarrow 15ds$, 'durante los próximos quince días'$\Rightarrow P+15dp$.

4.2.2. Gramática para el reconocimiento de expresiones temporales

A continuación vamos a describir las principales reglas de producción de la gramática que reconoce las expresiones temporales relacionadas con las entidades del modelo temporal. La gramática se ha implementado en Sictus Prolog, y en el Apendice B se detalla la especificación completa en este lenguaje.

- *DateSearch.*: Regla inicial que detecta cuando una secuencia de palabras valida una regla que reconoce una expresión temporal, y devuelve la secuencia inicial etiquetada con TIMEX. En el atributo VALUE tenemos la expresión temporal completa codificada semánticamente según los códigos propuestos en la sección anterior.
- *NoTempExp.* Regla que reconoce expresiones que podrían confundirse con una expresión de una entidad temporal.
- *AbsExp.* Regla que reconoce expresiones temporales absolutas y devuelve la expresión codificada semánticamente. En esta regla tenemos que tratar de evitar algunas ambigüedades, como la que se produce con la palabra 'mañana' que puede referirse a una parte del día o al día siguiente.
- *RelExp.* Regla que reconoce expresiones temporales relativas y devuelve la expresión codificada semánticamente.
- *Nucleus.* Regla que reconoce un núcleo temporal de la categoría núcleo (ver tabla 4.1), y devuelve su codificación semántica.
- *DayPart.* Regla que reconoce expresiones que contienen núcleos de la subcategoría *partedia* definidas en la tabla 4.1.

- *SimpleExp*. Regla general que reconoce expresiones temporales completas simples relacionadas con una entidad temporal. Esta regla busca modificadores o palabras de enlace alrededor de expresiones que validan las reglas *AbsExp*, *RelExp*, *Nucleus* o *DayPart*. La expresión *DayPart* puede formar una expresión simple ella sola, o bien en compañía de otra expresión temporal como 'la mañana de hoy' o 'esta mañana'. Además, el orden en que se aplican las reglas que se tienen que validar tratan de buscar la expresión más compleja, por ello, primero buscamos expresiones absolutas, luego relativas y por último los núcleos.
- *GroupExp*. Esta regla trata de encontrar expresiones temporales compuestas por varias expresiones simples, o que validan la regla *SimpleExp*, buscando la relación entre estas expresiones para ver si se trata de:
  - un intervalo. Si la expresión comienza por una sentencia que valida la regla *Interval*, contiene los terminales 'a' o 'hasta'. En este caso la codificación semántica será una **I** seguida de las codificaciones de las expresiones simples unidas con el separador **','**.
  - una secuencia si están enlazadas por 'y'. En este caso el código semántico sera la unión de las codificaciones semánticas de cada expresión mediante el separador **','**.
  - una única entidad temporal donde cada sentencia simple es una parte de la entidad temporal, ej. 'el día 5' de 'este mes'. En este caso las expresiones simples están enlazadas por el terminal 'de', y la producción devolverá las codificaciones de las expresiones temporales unidas por el separador **'#'**.
- *Mod*. Es una regla que reconoce uno o varios modificadores con algunas palabras de enlace, ej.: 'antes de', 'fue firmado','después del final'.
- *Granularity*. Regla para reconocer una granularidad, o sea un núcleo temporal de tipo grano en la tabla 4.1.
- *NumList*. Regla que reconoce listas de números (ej.: '3, 4 y 5', '1999-2000', 'dos,tres y cuatro'), o números simples (ej.: 'dos' o 'segundo').
- *DayList*. Regla que reconoce listas de días (ej.: 'lunes y martes', 'día 3,4 y 5', 'lunes al viernes').
- *MonthList*. Regla que reconoce tanto un mes, como una lista de meses (ej.: 'marzo y abril', 'meses de enero y febrero', 'primer y segundo mes').
- *YearList*. Regla que reconoce un año o una lista de años (ej.: 'año 99', '1999-2002', 'los años 85 y 87').
- *Separator*. Reconoce cualquier carácter que no sea una letra o un número.
- *Roman*. Regla para detectar números romanos, los cuales suelen aparecer en las expresiones absolutas con siglos.
- *Quantifier*. Regla para reconocer un número entero bien sea un dígito o una palabra que representa un número.

- *Interval.* Regla que detecta palabras presentes en los intervalos temporales (ej.: 'a partir de', 'desde', 'entre').

El orden en que se aplican las reglas de esta gramática nos permite obtener las expresiones temporales completas. El orden en que se evalúan los terminales y se genera la expresión codificada, nos permite representar el significado semántico de la expresión pudiendo diferenciar expresiones muy semejantes como: 'marzo del año pasado' ⇒ -y##Oy@m3, y ' marzo pasado'⇒ -Oy#m3.

```
<TEXTSTART:SCODE= 'PAIS(0).sect(1).Sect(0).artl(6).News(0).sbts(0).TXT(0)',
FECHA = '19990601'>
Fidel Castro destituyó  <TIMEX Value=-1ny#nm#nd> ayer </TIMEX>  de forma
fulminante a Roberto Robaina como ministro de Relaciones Exteriores de
Cuba, en  una  decisión que, según fuentes diplomáticas, 'se veía venir'
<TIMEX Value=IO-wp> desde hace semanas </TIMEX>  aunque sorprendió por su
brusquedad. Veinticuatro horas antes de darse a conocer el cese, un portavoz
de la Cancillería informó a la prensa extranjera de que Robaina realizaría
una gira por  Venezuela,  Panama y Haití a partir  <TIMEX Value=#Oy#m6#d1>
del 1  de  junio </TIMEX>. <TIMEX Value=Ow> Ya </TIMEX>  no será así. El
nuevo canciller cubano es Felipe  Pérez Roque, de  <TIMEX Value=34yp> 34
años </TIMEX> , un  cercano colaborador de Castro  <TIMEX Value=I#y1992>
desde 1992 </TIMEX> .
<TEXTEND>
```

Figura 4.1: Salida de TagTimex.

En la figura 4.1 se muestra un texto etiquetado resultado de aplicar esta gramática a una noticia del periódico de 'El País' del día 1/6/1999.

4.2.3. ModelTimex

ModelTimex es un módulo implementado en Python [Lut99], que trata de analizar las expresiones temporales etiquetadas con el módulo TagTimex. El módulo toma como entrada los documentos procesados por el módulo TagTimex, busca las etiquetas TIMEX, y analiza la codificación semántica de la expresión que se encuentra en el atributo Value. Una vez analizada la notación semántica de las expresiones, trata de reconocer la referencia temporal y clasificar las expresiones en uno de los siguientes tipos.

- *DATE* . La aplicación ha podido obtener un instante de tiempo cronológico absoluto según el calendario Gregoriano. Por tanto el atributo Type toma el

valor de *Fecha* y se modifica el atributo Value asignandole el valor obtenido según la notación de las entidades del modelo de tiempo descrito en el Capítulo 3.

- *EVENT*. Si a partir de la codificación semántica no ha podido obtener un instante, y en la expresión codificada existe una 'R', la expresión temporal es relativa a un suceso. En este caso añadiremos a la etiqueta Type el valor de *Event* que denota que tenemos una referencia temporal relativa a un suceso ('dos días antes de la firma' $\Rightarrow$ R-2d).
- *Duration*. El resto de expresiones temporales detectadas por el sistema y que no se han rechazado por ser frases hechas o expresiones comunes se etiquetarán como este tipo. La mayoría de estas expresiones son duraciones no ancladas en el tiempo.

En el diseño y la implementación de este módulo nos encontramos principalmente con tres problemas:

- Para la resolución de expresiones relativas a una fecha citada anteriormente (*anáforas*), se suele suponer que el punto de referencia es justo la fecha anteriormente citada, lo cual puede provocar ciertos errores. Por ejemplo, en el texto 'el 1 de septiembre de 1999 ... En agosto de ese año ... Al día siguiente ...', la última expresión toma la fecha de 1/9/1999 como referencia para el 'día siguiente', pero si la expresión fuera 'ese mes' se debe tomar como referencia 8/1999. Para minimizar este tipo de errores, vamos a almacenar en la variable DateP de la aplicación, la última fecha citada con granularidad de día, y en PeriodP el último intervalo o punto temporal con una granularidad superior a día. Así pues, la fecha de referencia, será DateP, con la excepción de que la granularidad de la expresión que se desee fijar, sea mayor a días y se encuentre en PeriodP. Este tipo de expresiones vienen caracterizadas por el código semántico **r**.

- En la resolución de las expresiones temporales en tiempo presente se requiere conocer el punto de referencia del hablante. El punto de referencia del hablante en los documentos de actualidad, como es el caso de los periódicos, puede suponerse que es la fecha de publicación. Con esta simplificación podemos producir ciertos errores, debido a que algunas expresiones son parte de frases escritas en estilo directo, transcripciones literales, citadas por una tercera persona. Cuando se utiliza estilo directo, el punto de referencia del hablante también puede ser distinto a la fecha de publicación. Por ejemplo, la frase 'El presidente dijo: Mañana...' escrita en estilo directo, producirá errores en nuestro sistema. Estas sentencias son poco comunes, y dado que generalmente la fecha sólo difiere entre el hablante y el escritor en un día, no es necesario analizar en detalle estas expresiones. De todos modos hemos detectado que existen verbos que suelen utilizarse en este tipo de expresiones,

```
<TEXTSTART:SCODE = 'PAIS(0).sect(1).Sect(0).artl(6).News(0).sbts(0).TXT(0)',
FECHA = '19990601'>
Fidel Castro destituyó  <TIMEX type = DATE Value=y1999m05d31> ayer </TIMEX>
de forma fulminante a Roberto Robaina como ministro de Relaciones Exteriores
de Cuba, en una decisión  que, según  fuentes  diplomáticas, 'se  veía venir'
<TIMEX type=DATE  Value=[y1999m5w3,19990531]> desde  hace  semanas </TIMEX>
aunque  sorprendió  por su brusquedad. Veinticuatro  horas  antes  de darse
a conocer el cese, un portavoz  de  la  Cancillería  informó  a  la  prensa
extranjera  de  que  Robaina  realizaría  una  gira  por Venezuela, Panama y
Haití a partir  <TIMEX type=DATE Value=y1999m6d1> del 1 de junio </TIMEX > .
<TIMEX type=DATE Value=y1999m6w1> Ya </TIMEX Ow>  no  será  así. El nuevo
canciller cubano es  Felipe Pérez Roque, de  <TIMEX type=DURATION Value=34yp>
34  años  </TIMEX> , un cercano  colaborador de Castro <TIMEX type=DATE
Value=[y1992,1999]> desde 1992 </TIMEX> .
<TEXTEND>
```

Figura 4.2: Texto etiquetado con ModelTimex.

y además la mayoría de veces se suele referenciar la fecha del hablante, por ello hemos incluido en la notación el código **O**, que denota presente respecto al hablante, el cual puede variar respecto al escritor. Así en estos casos, si existe una fecha de referencia, la tomamos, y en caso contrario tomamos como referencia la fecha de publicación.

- Este módulo se encarga también de resolver las indeterminaciones cuantitativas. En una expresión que no posee un cuantificador definido, y se tiene en la codificación una **p** que indica granularidad plural, supondremos que la imprecisión es de tres unidades en la granularidad utilizada. Por ejemplo, 'los próximos días', 'hace unos días', lo resolveremos desplazando la fecha de referencia 3 días.

El Algoritmo 4.1 presenta de modo simplificado el procedimiento utilizado en el módulo sf ModelTimex para la conversión del modelo semántico al modelo de tiempo. A continuación describimos algunas funciones que utiliza este algoritmo.

GranularityOf(*component***).** Función que devuelve la granularidad presente en *component*.

FindRefererenceDate(*component***, *pd*, *DateP*, *PeriodP*).** Función que devuelve la fecha de referencia que hay que tomar para la resolución de la expresión.

- Si existe una **n** en *component* devuelve la fecha de publicación *pd*.
- Si existe una **r** en *component*. Hay que tener en cuenta que en *DateP* tenemos la última fecha citada, y en *PeriodP* el último periodo citado. Por tanto:

Algoritmo 4.1 ModelTimex Algorithm.

```
Entradas: expression{Coded temporal expression with CodTemp},
    sentence{Sentence that contains the expression }, pd{Publication Date}
    DateP{Previous date}PeriodP. {Previous interval}
Salidas: InstantList {List of the instants that appears on the text}
 1: for all chain ∈ expression.split(',')[2] do
 2:   FF=''
 3:   for all component ∈ chain.split('#') do
 4:     g=GranularityOf(component)
 5:     referenceDate=FindReferenceDate(g,pd,DateP,PeriodP,component)
 6:     if referenceDate=='-1' then
 7:       return []
 8:     end if{The time dirección could be: presente O ,past '-' or future '+'}
 9:     if (T=TimeDirection(component))==-1 then
10:       if (T =TimeDirection(chain))==-1 then
11:         if (T=TimeDirection(expression))==-1 then
12:           T=VerbalTime(expression)
13:         end if
14:       end if
15:     end if
16:     d=find('\d',component)
17:     A=find(.^A|F",expression)
18:     if d<>-1 and find(g+d,component) then
19:       FF+=g+'\d'
20:     else
21:       if A==-1 then
22:         if find('n|0'+g,component) then
23:           FF+=ActualGranularity(referenceDate,g)
24:         else
25:           if d==-1 then
26:             if find("p",component) then
27:               d=3
28:             else
29:               d=1
30:             end if
31:           end if
32:           FF=shift(FF,referenceDate,g,T,d)[3]
33:         end if
34:       end if
35:     end if
36:   end for
37:   if A<>-1 then
38:     if FF==" then
39:       FF=referenceDate
40:     end if
41:     FF=SubIntervalOf(FF,expression)
42:   end if
43:   if FF<>" then
44:     if IsDate(FF) then
45:       DateP=FF
46:     else
47:       PeriodP=FF
48:     end if
49:     InstantList.append(FF)
50:   end if
51: end for
52: if 3>len(InstantList)>0 and find("I",expression) then
53:   if len(InstantList)==2 then
54:     InstantList=["[InstantList[0],InstantList[1]]"]
55:   else
56:     if InstantList[0]<pd then
57:       InstantList=["[InstantList[0],pd]"]
58:     else
59:       InstantList=["[pd,InstantList[0]]"]
60:     end if
61:   end if
62: end if
63: return InstantList
```

- en el caso de que existan los dos valores, generalmente tomaremos $DateP$, a excepción de que la granularidad g sea distinta de día y mayor o igual que la granularidad mínima de $PeriodP$,
- si sólo existe uno de los dos valores, éste es el que tomaremos como fecha de referencia, y
- si no existe ninguno devuelve -1.

▪ Si existe una **O** en *component*:

- si existe $DateP$ y la distancia a la pd es de menos de siete dias devuelve $DateP$,
- en otro caso devuelve pd.

▪ Si existe una **R** en *component*: entonces es una fecha relativa a un suceso, la cual no podemos calcular. Devuelve -1.
▪ En cualquier otro caso devuelve pd.

TimeDirection(*cadena***).** Función que trata de discernir si el hablante se refiere al presente, pasado o futuro. Así pues analiza *cadena* y:

▪ si existe + devuelve +,
▪ si existe - devuelve -,
▪ si existe **n** devuelve **n**,
▪ si existe **O** devuelve **O**, y
▪ en cualquier otro caso devuelve -1.

VerbalTime(*cadena***).** Es una función que trata de obtener la dirección temporal a partir del tiempo verbal. Para ello hace uso de MACO+ [Tur99, Cas98], un lematizador y analizador morfológico en español que posee un diccionario con 770.000 entradas de verbos y 225.000 entradas correspondientes a otras partes del habla.

find(*patron***, *cadena*).** Función que busca en *cadena* la expresión regular *patron* devolviendo la subcadena de *cadena*, que valida el *patron*, o si no se cumple la expresión regular devuelve -1 .

ActualGranularity(P, g). Función que trata de extraer el valor de la granularidad g en el punto temporal P. La función devuelve una cadena formada a partir de la concatenación de la granularidad g con el valor de esa granularidad en P. Si ésta no existe devuelve un -1.

shift(FF, *date*, g, T, d) : Es la función que realiza la operación de desplazamiento definida como *shift* en el capítulo del modelo de tiempo, o sea es equivalente a: $shift(punto, T + d + g)$, donde *punto* tomará el valor de FF si contiene a la granularidad g, y si no tomará fecha.

SubIntervalOf(FF**,** *expr***).** Es una función que trata de extraer un subintervalo de FF. Si en *expr* tenemos una **A** trata de obtener el subintervalo inicial, y si tenemos una **F** el subintervalo final a nivel de la granularidad de la expresión o de días. Esta operación se requiere en expresiones del tipo 'principios de mayo' (primeros días del mes de mayo), 'A final del mes' (últimos días del mes), o 'últimos meses' (toma los últimos meses del año).

A modo de ejemplo de ejecución de este algoritmo vamos a ver dos pequeñas trazas:

```
expression="PFny"
sentence="<TIMEX TYPE=DATE VALUE=PFny>A finales de este año</TIMEX> se celebra el congreso"
pd="19990202"
DateP=""
PeriodP=""
Como tenemos una expresión simple, entonces el sistema:
-linea 2: FF=''
-linea 4: g='y'
-linea 5: referencedate="19990202"
-linea 9: T='n'
-linea 16: d=-1
-linea 17: A='F'
-linea 39: FF=y1999m2d2
-linea 41: FF=[y1999m12d28,y1999m12d31]
-linea 47: PeriodP=FF
-linea 49: InstantList=[FF]


expression="IOw#d1,Ow#d5"
sentence="<TIMEX TYPE=DATE VALUE=IOx1,Ox5> Del lunes al viernes</TIMEX> se celebra
el congreso" pd="y1999m2d13"
DateP="y1999d2d10"
PeriodP=""

-Linea 1: chain =IOx1
-linea 2: FF=''
-Linea 3: component =IOw
-linea 4: g='x'
-linea 5: referencedate="y1999m2d10" ya que la distancia de DateP  a pd es
menor de 7 días
-linea 9: T='O'
-linea 16: d=1
-linea 19:
FF=y1999m2w2x1
-linea 45: DateP=FF
-linea 49: InstantList=[y1999m2w2x1]

-repito a linea 1:
-Linea 1: chain =Ox5
-linea 2: FF=''
-Linea 3: component =Ox5
-linea 4: g='x'
-linea 5: referencedate="y1999m2d10" ya que la distancia de DateP  a pd es menor de 7 días
-linea 9: T='O'
-linea 16: d=1
-linea 19: FF=y1999m2w2x5
-linea 45: DateP=FF
-linea 49: InstantList=[y1999m2w2x1,y1999m2w2x1]
-linea 54: InstantList=["[y1999m2w2x1,y1999m2w2x1]"]
```

4.3. Trabajos relacionados con la Extracción de Información Temporal

Ya en el año 1997, dentro de las conferencias del MUC-7 [Chi97] se destaca la importancia de la detección de expresiones temporales proponiéndose su etiquetado. En la versión 3.5 de la definición de la tarea de *Name Entity* del MUC [Chi97], podemos encontrar la primera propuesta de etiquetado de expresiones temporales. Según esta propuesta, las expresiones temporales absolutas o relativas se deben etiquetar con una etiqueta denominada `TIMEX`. Esta etiqueta posee un atributo Type que clasifica las expresiones temporales en expresiones en dos tipos, TIME para expresiones con granularidades inferiores a días y DATE para el resto.

En 1999, en el NIST (*National Institute of Standard and Technology*), se redefine la tarea *Name Entity* del MUC dentro del proyecto 'HUB4' [Hir99] también subvencionado por ARPA. El HUB4 es un proyecto que intenta analizar las transmisiones de noticias sean en papel, radio o televisión mediante un sistema de indexado a nivel de sucesos utilizando *frames*. En este sistema se posee una rejilla para las fechas, que se etiquetan también con la etiqueta TIME con dos tipos DATE y DURATION. Se etiquetan como de tipo DATE sólo las expresiones absolutas. Aquí, una expresión absoluta es aquella que especifica una fecha o intervalo de tiempo determinados, como 'lunes', o bien sean un día característico como 'el día de todos los santos'. Para las expresiones temporales que indican periodos de tiempo anclados o no, es decir, las expresiones que poseen una granularidad temporal y un adjetivo numeral, se añade la etiqueta DURATION. En este sistema las expresiones relativas a sucesos ('desde el principio de las negociaciones') no se etiquetan. Los adjetivos indefinidos no forman parte de estas expresiones temporales excepto los numerales como 'pocos', 'par', o 'varios'. Por esta razón en este sistema la expresión 'dos años antes' se etiquetará como `'<TIMEX TYPE= DURATION>dos años </TIMEX>antes'`.

En la actualidad existen muchos analizadores sintácticos que incluyen la extracción de algunos tipos de expresiones temporales. Destacaremos el sistema TACAT [Tur99, Cas98], que es un sistema de análisis sintáctico para textos en español, y que es capaz de detectar expresiones simples como '5 de mayo de 1999' o '5/mayo/99', 'día 3','lunes', aunque no es capaz de detectar una expresión temporal única como ' mayo de este año', 'próximo lunes', o 'dos días después de la firma'.

4.3.1. Sistemas de representación del conocimiento temporal

En la literatura podemos encontrar sistemas de extracción de expresiones temporales presentes en documentos y sistemas de representación del conocimiento temporal. Estos sistemas además de extraer las expresiones temporales, y clasificarlas en distintos tipos, son capaces de inferir ciertas relaciones entre las expresiones temporales, comprobar la consistencia de éstas o incluso asignar una fecha a la expresión temporal. Entre ellos podemos destacar los siguientes:

- En [Kni98] se propone un sistema de análisis de documentos legales donde se intenta analizar los documentos y ver si la explicación de los hechos de los distintos testigos concuerda. En el sistema se define un formalismo para representar tanto de expresiones temporales relativas como absolutas, y posee un mecanismo formal para la comprobación de las consistencias temporales. Las expresiones se representan mediante una tripla (P, $Meets$, Dur) donde: P es el conjunto de primitivas de tiempo(puntos o intervalos), $Meets$ es una relación binaria sobre los elementos de P que se encuentran ordenados respecto a la relación $Meets$, y Dur es una función entre P y los números reales positivos que representa una duración temporal. Este sistema utiliza un modelo de tiempo lineal, sin ramificaciones, donde no se permiten bucles temporales. Con este modelo un elemento temporal t es un punto si $dur(t) = 0$, en otro caso se trata de un intervalo. En el modelo hay relaciones definidas entre intervalos, puntos y entre puntos e intervalos.

  A diferencia de nuestro caso, en este modelo lo importante no es conocer la fecha en que se producen las acciones o sucesos, no se intenta comprobar qué fechas se referencian en las expresiones relativas, sólo se trata de comprobar la consistencia entre las expresiones temporales.

- El sistema Vermobil [Ste98] es un sistema automático multilingüe (para inglés, alemán y japonés) de negociación de citas entre dos socios. Los sistemas de cada socio dialogan entre sí hasta encontrar una fecha que se ajuste al calendario de los dos. Debido a la complejidad del Lenguaje Natural, que permite expresar una misma fecha en distintas formas, y a que muchas veces se tiene información incompleta o errónea, se han decidido analizar estas expresiones en dos pasos. En el primer paso se extraen del diálogo las expresiones temporales y se asocian a una representación formal que denominan ZeitGram. Posteriormente estas representaciones se interpretan semánticamente realizando un análisis en profundidad, de modo que al final se obtiene un intervalo. Según este modelo, un intervalo es una representación canónica computacional con una fecha de inicio y una fecha final, donde la fecha tiene

el formato (año, mes, día, hora, minuto). Zeitgram es una gramática de contexto libre que permite representar las expresiones temporales utilizando un lenguaje próximo al natural pero con ciertas abstracciones. De este modo el análisis posterior permite obtener la fecha y hora exacta a la que se refiere cada expresión, utilizando el conocimiento de un calendario para verificar las consistencias.

Su gramática reconoce:

- *Puntos* que son representados en el formato (fecha,hora, minuto),
- *Pointlike* que denotan las partes del día, los días de la semana, las partes del año (estaciones), las semanas del año, la semana del mes, el día del mes, el mes, el año, los festivos, y el día de referencia ('ese').
- *Intervalos* que representan duraciones ancladas donde se pueden tener bien los dos extremos ('desde P1 hasta P2', 'entre P1 y P2') o uno sólo ('después/antes + P ').
- *Expresiones con puntos de referencia.* Estas son las referencias implícitas ('ahora'), bien referencias a un día de la semana ('el viernes'), o bien duraciones ancladas, antes/después de un punto de referencia ('dos días antes').

Para resolver las anáforas, en el dominio del discurso a este sistema le basta con tomar la fecha u hora de la expresión anterior.

- En el artículo [Koe00] se propone un sistema para extraer las fechas de las noticias digitales, con el objetivo de proveer automáticamente información suplementaria a cada artículo para ayudar al lector. El sistema trata de extraer expresiones temporales relativas a fechas, horas, intervalos, duraciones y edades. Para ello posee un conjunto de patrones temporales clasificados según el tipo de expresión temporal que se trate, y dentro de cada grupo los patrones se hayan organizados de mayor a menor nivel de detalle para evitar ambigüedades. En este sistema se reconocen como fechas sólo aquellas expresiones temporales absolutas. Además, tiene en cuenta que en el caso de expresiones absolutas donde se requiere la dirección temporal como ('en mayo', 'el lunes', 'el pasado jueves') se debe buscar en la frase el tiempo verbal. Para simplificar el proceso, y ya que en inglés los verbos en pasado terminan con 'ed', se asigna siempre la dirección en el futuro, a no ser que se encuentre en la misma sentencia una palabra terminada con 'ed'. En este sistema las expresiones relativas como ' hace dos días' o 'dos días antes' se clasifican como duraciones. Una vez extraídas las referencias temporales se convierten a un formato normalizado, para poder compararlas y secuencializarlas.

4.4. Evaluación del sistema

```
After narrowly defeating rightist candidate Joaquín Lavín in  <TIME
Value=#y16#0y#m1#d1> a January 16 </TIME>  run-off election, Ricardo
Lagos took office on  <TIME Value=0y#m3#d11> March 11 </TIME>,  becoming
the first socialist president since Salvador Allende was toppled in a
<TIME Value=#y1973> 1973 </TIME>  military coup led by General Pinochet.
<TIME Value=-1w> A week before </TIME>  Lagos' inauguration, Pinochet
returned to Chile from the United Kingdom after the House of Lords ruled
him medically unfit to be extradited to Spain, where he faced charges
of crimes against humanity. On  <TIME Value=0y#m3#d3> March 3 </TIME>,
<TIME Value=rd> the day </TIME> Pinochet returned to Chile, army officials
ordered the media off the Santiago military airstrip where his plane was
going to land. The order was later rescinded.<br>
```

Figura 4.3: Salida deTagTimex para un texto en inglés.

Aún con todos los problemas propios del Lenguaje Natural, nuestro sistema de extracción de fechas e intervalos TimeExtractor produce tan sólo un 3.3 % de errores. Si desechamos los formatos de fecha normalmente reconocidos por otros sistemas ('mayo de 1999', 'día 1 de mayo' o '1/5/99'), y expresiones como 'hoy' y 'ayer', el error aumenta al 5.6 %. Esto significa que se produce un error del 5.6 % en la detección de fechas que otros sistemas no detectan. Para esta evaluación hemos utilizado los contenidos completos de tres periódicos de 'El Pais Digital', donde el sistema ha obtenido un total de 613 expresiones temporales, siendo 246 expresiones temporales absolutas. Del total de expresiones temporales, el sistema ha detectado que 518 representan un instante de tiempo.

Hemos desestimado como errores aquellos producidos por fallos del escritor, debidos generalmente a la no concordancia entre la expresión temporal y el verbo, o a errores tipográficos. Además estos errores pueden dar lugar a que expresiones relativas a la fecha anterior no puedan ser bien interpretadas por nuestro sistema debido a su falta o incorrecta asignación.

Destacaremos que la mayor parte de los errores son debidos a fallos en el módulo TagTimex. Hay que tener en cuenta que las expresiones que se emplean en el Lenguaje Natural, no se adecúan totalmente al modelo temporal. Por ejemplo, en los textos relatados al final del año 1999 nos encontramos con muchas noticias que señalan que el año 2000 comienza el nuevo siglo, lo cual fue un error muy popular. Otro problema surge con el significado de 'hace dos semanas'. Esta expresión no referencia a dos semanas anteriores en el calendario, sino a 2*(duración una

semana), o sea a 14 días antes. Como norma, en nuestro modelo los desplazamientos a nivel de semanas los calcularemos como desplazamientos de 7 días por cada semana, que parece ser el significado más utilizado.

4.5. Conclusiones

El sistema TimeExtractor descrito en este capítulo es un sistema capaz de detectar todo tipo de expresiones temporales tanto fechas absolutas, periodos relativos o bien duraciones. Una de sus limitaciones es el no detectar la fecha o periodo absoluto que se referencia en el caso de expresiones temporales relativas a sucesos, debido a que esto requeriría tener un repositorio con los periodos de suceso de los acontecimientos más importantes. Esta aplicación se ha desarrollado para el idioma español, pero con una arquitectura portable a otros idiomas, gracias a la definición de una notación semántica para las componentes de las expresiones temporales según el modelo de tiempo propuesto en el capítulo anterior.

En el próximo capítulo vamos a ver cómo utilizar referencias temporales presentes en los textos en la búsqueda de documentos sobre sucesos específicos, y también en cómo estas nos pueden ayudar a agrupar los documentos para obtener información sobre los sucesos narrados en la colección de documentos.

Capítulo 5

Recuperación de información temporal

Los sistemas de recuperación de documentos tradicionales, como los buscadores existentes en la red tipo 'Yahoo' o 'Goggle', son útiles para la búsqueda de documentos que hablan sobre ciertos conceptos específicos que se pueden expresar mediante una consulta con palabras clave, pero no permiten responder a consultas genéricas como ¿Qué ha ocurrido?, ¿Qué cosas nuevas se han producido?, ¿Qué sucesos de tal tipo se han producido?. Estos sistemas no proveen información muy útil cuando se busca por ejemplo qué acontecimientos han ocurrido en cierto lugar durante un periodo dado, o cuando se quiere recabar información sobre sucesos similares a uno dado, o bien conocer los hechos relevantes que le han sucedido a cierta persona. Los sistemas de RI, para este tipo de consultas devuelven los documentos relacionados con los conceptos de la consultas pero no son capaces de descubrir sucesos. Además si la colección de documentos es muy grande, el usuario no puede analizar todos los documentos que devuelve el sistema aun cuando estén organizados por relevancia con la consulta, ya que no pueden relacionarse en el tiempo en el que se producen las acciones narradas en los documentos. Los sistemas de RI tradicionales son útiles cuando se conoce de forma precisa la naturaleza de los hechos que se buscan, pero no permite realizar un seguimiento de los sucesos y hechos que evolucionan en el tiempo. Ello es debido principalmente a que estos sistemas no soportan operadores para manejar metadatos temporales, y por otro lado a la dificultad de obtener manualmente dichos metadatos, que en el caso de grandes flujos de documentos resulta impracticable.

En este capítulo vamos a desarrollar algunas herramientas que tratan de resolver consultas de RI, donde la información temporal presente en los documentos es muy importante para organizar las respuestas que se devuelven al usuario.

Además propondremos la asignación automática de un metadato que contenga la información temporal sobre cuándo se produce el suceso principal relatado en los documentos, y al cual denominaremos *periodo de suceso*. Como veremos más adelante éste facilitará la búsqueda de sucesos y el agrupamiento cronológico por los hechos narrados de los documentos.

A lo largo de este capítulo utilizaremos las siguientes definiciones para referirnos a los distintos tipos de información que vamos a tratar:

- Un *suceso* es una acción o actividad que transcurre en un lugar y tiempo específico.
- Un *tópico* es un conjunto de sucesos directamente relacionados como consecuencia de un suceso origen muy relevante.
- Un *tema* es la acción o hecho principal de la que trata un suceso.
- Una *crónica* es una colección de documentos que hablan sobre un mismo suceso, el cual tiene asociado un intervalo de tiempo que representa el *periodo del suceso* principal común a todos los documentos de la crónica.

En la actualidad existen distintos sistemas que permiten recuperar de algún modo documentos que relatan determinados sucesos, a saber:

- Los sistemas de Recuperación de Información permiten recuperar del repositorio el conjunto de documentos que hablan sobre un suceso especificando en la consulta los conceptos que caracterizan al suceso, o sea el tema del suceso. Los documentos recuperados se muestran al usuario según la semejanza de cada documento con la consulta. Cuando existen muchos sucesos que tratan el mismo tema en el repositorio, el usuario tiene grandes dificultades para discernir qué documentos hablan sobre el suceso que le interesa, ya que aunque hablan sobre el tema de interés, el usuario no tiene la posibilidad de especificar cuándo se produjo el suceso que busca. De este modo, el usuario debe leer los documentos recuperados para detectar si hablan o no sobre el suceso que le interesa.
- Los sistemas de RI que también indexan la fecha de publicación permiten recuperar documentos que hablan de un cierto tema y que se han publicado un cierto día, ó en una ventana temporal. Si el usuario conoce el día en que se ha publicado el suceso, y el tema principal del suceso, será capaz de recuperar el suceso deseado. Esta aproximación tiene varios problemas: la fecha en la que se produce un suceso no siempre coincide con la fecha de publicación, y el usuario no siempre conoce la fecha en la que se ha producido el suceso.

- El modelo TOODOR [Ara98c, Ara99] permite realizar una aproximación al análisis de sucesos, utilizando un atributo temporal denominado **periodo de suceso**, el cual representa el intervalo de tiempo en el que transcurre la acción principal del documento. Una breve introducción a este sistema se presenta en la sección 5.4. Sin embargo la limitación principal de este sistema es que no proporciona ningún mecanismo para la asignación del *periodo de suceso*, con lo que se asume que ésta se realiza de forma manual. Entre las tareas que permite realizar este sistema destacaremos dos:
  - recuperar los documentos que relatan un suceso, especificando en la consulta tanto el tema principal del suceso, como predicados sobre su periodo de suceso.
  - recuperar todos los documentos que tratan el tema que se especifica en la consulta, agrupados por su periodo de suceso. Representar los documentos agrupados por sus propiedades temporales ayuda al usuario a localizar los documentos que hablan sobre el suceso de interés cuando éste no conoce con exactitud la fecha en la que se ha producido o cuando se desea conocer todos los sucesos relacionados con un determinado tema.
- En los sistemas de detección y seguimiento de tópicos (TDT) se intenta detectar los sucesos existentes en una colección de documentos, aplicando distintas técnicas de agrupamiento automático de documentos (*clustering*). En estos sistemas, la definición de Tópico ha ido evolucionando, y en la segunda fase del proyecto TDT, TDT-2 [TDT00], se define un Tópico [1] como un conjunto de sucesos, de modo que uno de ellos es el suceso principal y los otros están directamente relacionados con él o son consecuencia del mismo. En estos trabajos, la fecha de publicación se ha utilizado para agrupar los documentos que se encuentran dentro de una ventana temporal y/o bien disminuir su semejanza en función de la proximidad temporal [All98]. Como se demuestra en [Swa99], la utilización de la fecha de publicación, aunque mejora la efectividad de los sistemas, no produce todas las mejoras que cabría esperar de la utilización de la temporalidad de los documentos.

Los sucesos muy relevantes generalmente tienen una evolución a lo largo del tiempo, de modo que éstos se suelen relacionar entre sí utilizando referencias temporales. Las referencias temporales, junto con algunas palabras clave, nos permiten reconocer y diferenciar unos sucesos de otros (por ejemplo 'elecciones de 1999', 'elecciones de 2002'). Ya que las referencias temporales se utilizan para ubicar y relacionar los sucesos, si somos capaces de extraer las referencias temporales presentes en los documentos, seremos capaces de mejorar los sistemas de RI y los

[1] En TDT-2 se utiliza el término **event** para **suceso** y **topic** para **tópico**.

sistemas de TDT. La evolución de los sucesos en el tiempo dificulta la tarea de su recuperación mediante un sistema RI. Esto es debido a que los términos, personas y lugares del suceso pueden ir cambiando a lo largo del tiempo. Por esta razón, en los sistemas de TDT se ha optado por aplicar técnicas de agrupamiento de documentos, los cuales permiten relacionar documentos que hablan sobre temas similares. Sin embargo, el agrupamiento de documentos por conceptos permite obtener todos los documentos que relatan los mismos temas, pero no el mismo tópico o suceso. Para localizar los tópicos o sucesos se requiere conocer cuándo se han producido. De ahí que el conocimiento de las propiedades temporales de los documentos juegue un papel muy importante en la detección de tópicos y sucesos.

Con el objetivo de ayudar al usuario a obtener documentos relacionados con un determinado suceso, vamos proponer varias mejoras del sistema TOODOR y los sistemas de TDT, que aprovechen las propiedades temporales de los documentos. Para ello nos vamos a centrar en un tipo determinado de documentos con un marcado carácter temporal: los periódicos.

Con respecto a TOODOR, en este capítulo vamos a definir un sistema automático para la obtención del *periodo de suceso* a partir de las fechas e intervalos que aparecen en los documentos, y que se obtienen tras el procesado de los documentos con la herramienta TimeExtractor, definida en el capítulo anterior. Además, proveeremos al usuario con una herramienta de análisis de los documentos recuperados mediante gráficas de evolución temporal, que presentan los documentos agrupados por las características temporales. A este sistema lo denominamos TimExplR (Time Exploration for Information Retrieval). El análisis de estas gráficas nos permite generar un nuevo método de obtención de crónicas, que mejora las limitaciones encontradas en las crónicas definidas en el modelo TOODOR (ver sección 5.4.4).

En relación a los sistemas de TDT, proponemos la utilización de las fechas que aparecen en los documentos y del *periodo de suceso* para el agrupamiento automático de los documentos que hablan sobre el mismo suceso.

El resto del capítulo se organiza como sigue. En la próxima sección, realizamos un estudio de las distintas propiedades temporales que podemos asignar a un documento. En la sección 5.3 definiremos el algoritmo para la obtención del *periodo de suceso* de un documento. En la sección 5.2 describimos la colección sobre la que evaluaremos las herramientas propuestas, así como la representación adoptada para los documentos. En la sección 5.6.1 presentamos las medidas de evaluación de las crónicas, basada en las medidas propuestas para los tópicos en los sistemas de TDT [TDT00]. En la sección 5.4 describimos la arquitectura de TOODOR, evaluaremos el sistema de creación de crónicas utilizando el periodo de suceso, y detectaremos sus principales limitaciones. Además, proponemos una representación gráfica de los documentos mediante gráficas temporales. En la sección 5.5 vamos a

definir un sistema de extracción automática de crónicas a partir de una colección de documentos que se va incrementando en el tiempo, esta tarea en TDT se denomina *On-line Topic Detection*. En esta sección compararemos el impacto que se produce en la efectividad del sistema al utilizar las entidades temporales de los documentos y sus periodos de suceso. Finalizaremos este capítulo con las conclusiones generales sobre las herramientas desarrolladas en este capítulo.

5.1. Propiedades temporales de los documentos

Los modelos de datos tradicionales pueden ser extendidos con diversas dimensiones de tiempo para reflejar algún aspecto de éstos que necesita ser representado y consultado [Sno95]. Al igual que estos modelos, los documentos también pueden tener asociados varias propiedades temporales, a saber:

- *Tiempo de inserción*, que representa la fecha en la cual el documento se inserta en el repositorio.
- *Fecha de publicación*, que es la fecha en la cual aparece publicado el documento en los medios de comunicación.
- *Referencias temporales*, lista de puntos o intervalos temporales que aparecen en el documento bien explícitas mediante fechas, o bien implícitas en forma de expresiones temporales.
- *Periodo de suceso*, intervalo temporal en el cual transcurre el suceso principal del documento.

El *tiempo de inserción* puede ser muy relevante para la gestión de los documentos, pero no da ninguna información sobre los sucesos que se relatan en un documento. Éste suele asignarse automáticamente a los documentos por el sistema de almacenamiento.

La *fecha de publicación* suele asignarse como metadato en los documentos y se utiliza en los sistemas de Recuperación de Información y en los sistemas TDT para mejorar su efectividad. La idea de aplicar la fecha de publicación para la localización de sucesos en grandes colecciones surgió a partir de unos estudios realizados sobre la distribución temporal en las que aparecen publicadas las noticias [All98, Yang98, Yan00], en los cuales se detecta que:

- un suceso se publica uno o varios dias después de su inicio dependiendo de su importancia y actualidad,
- cada nuevo suceso produce una secuencia de noticias relacionadas con él durante un periodo de tiempo próximo a la fecha de publicación de la primera noticia sobre ese suceso,

- un hueco temporal entre las fechas de publicación de dos secuencias de noticias que hablan del mismo tema indica sucesos diferentes,
- el periodo de tiempo en el que se habla de un suceso suele ser relativamente corto.

En los sistemas TDT se suele definir una ventana temporal sobre la fecha de publicación de los documentos entrantes con el fin de tener en cuenta las observaciones anteriores. Sin embargo, la inclusión de esta propiedad temporal en los algoritmos de detección de sucesos no producen tantas mejoras como cabría esperar, debido principalmente a los siguientes factores [Swa99]:

- No siempre se conoce cuando van a ocurrir los sucesos, y las noticias se publican a primera hora del día, con lo cual sólo coincide la fecha de publicación con la fecha del suceso cuándo son sucesos conocidos y programados.
- Por regla general, los nuevos sucesos generalmente se publican el día posterior al suceso. Sin embargo, como los periódicos tienen un tamaño determinado, y no todas los noticias tienen el mismo impacto, si el suceso no es muy importante no se publica de inmediato.
- Los sucesos no siempre son puntuales, pueden tener varios días o semanas de duración (ej. una inundación, una guerra, etc.)
- Debido a la aparición de otras noticias de más interés, o bien a que no todos los días se tienen nuevos datos sobre el suceso, un suceso de cierta duración no aparece todos los días publicado.

Las *referencias temporales* que aparecen en un documento permiten ubicar el suceso principal en el tiempo, y además relacionar el suceso con otros sucesos muy similares. Pero su uso en un sistema de RI o en una base de datos tradicional resultaría muy costosa, además de que no existen sistemas comerciales que soporten múltiples intervalos o fechas en un mismo campo o atributo para operar con ellos. Como alternativa, el *periodo de suceso* da información sobre cuándo se produce el suceso principal. El sistema TOODOR [Ara99] es un ejemplo de sistema que utiliza el *periodo de suceso*. Pero ¿existe algún método que nos permita extraer *periodo de suceso* automáticamente?, ¿puede el *periodo de suceso* ayudar a recuperar documentos con la misma eficiencia que las referencias temporales? Estas dos preguntas las intentaremos resolver a lo largo de este capítulo.

En la gráficas de la figura 5.1 se han representado varios artículos periodísticos funto con las fechas que se referencian en los textos (las cuales han sido identificadas con TimeExtractor). El estudio de las fechas que aparecen en los documentos nos permite identificar una serie de propiedades sobre dichas referencias:

- La mayoría de las fechas o periodos referenciados se hayan próximos a la fecha de publicación, lo que confirma la hipótesis de los sistemas TDT.

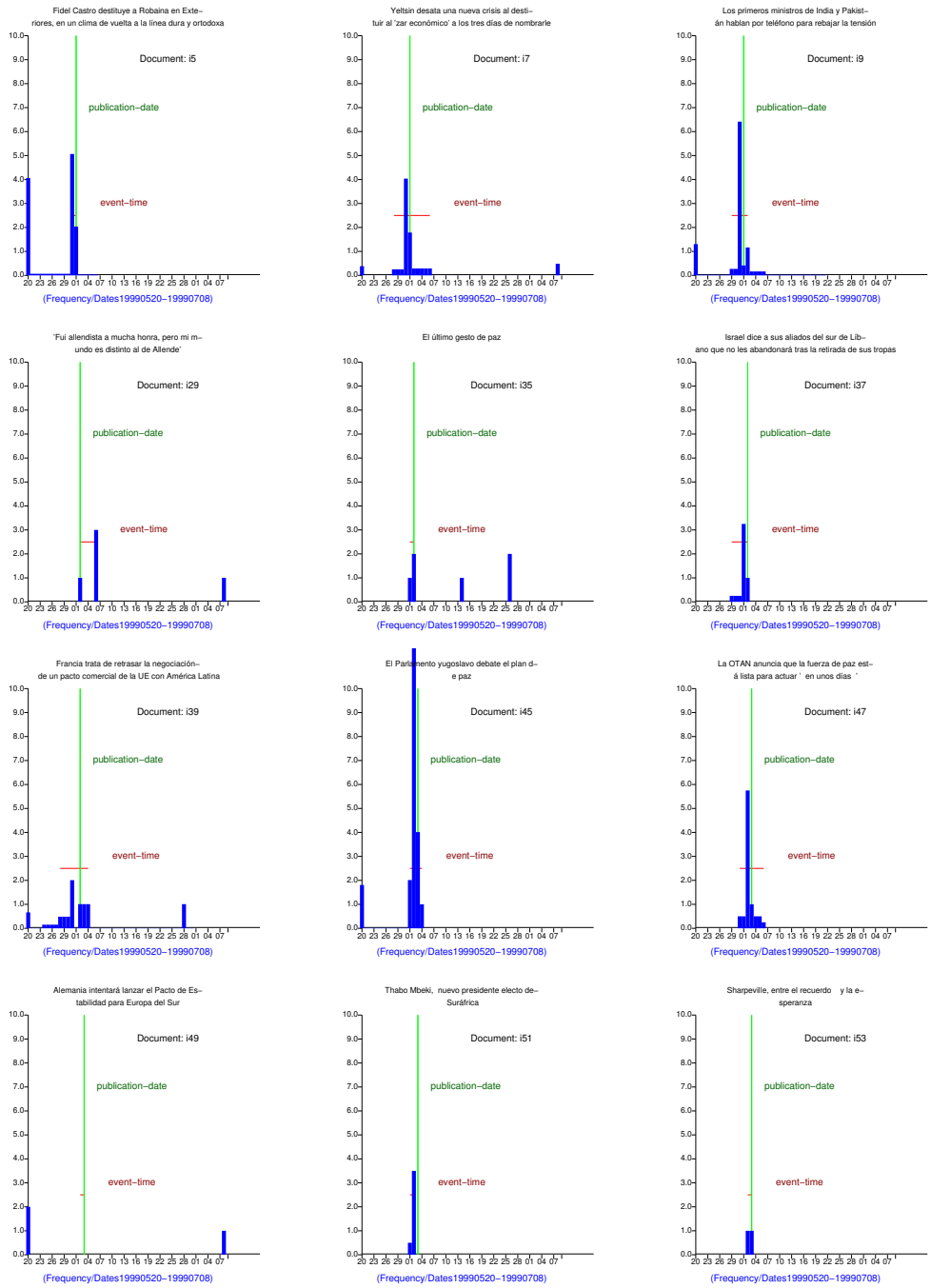


Figura 5.1: Histogramas de las fechas que aparecen en las noticias.

- Si se trata de un suceso no esperado, las fechas más relevantes se producen con anterioridad a la fecha de publicación (ej.: i37).
- Sólo los sucesos esperados suelen reflejarse en las noticias apareciendo la mayoría de fechas relevantes en fechas posteriores a la fecha de publicación (ej.: i29).
- En algunos documentos existen un número de referencias temporales muy lejanas a la fecha de publicación (a una distancia de más de 1 año) que suelen relacionar el suceso con sucesos acontecidos a las personas involucradas (su nacimiento, nombramientos, etc.).
- Existen un número de referencias temporales relativamente próximas a la fecha de publicación, que relacionan el suceso con otros relacionados y que pueden formar parte del mismo *tópico*.
- Existe una gran variedad de distribuciones temporales, lo que hace imposible definir una ventana temporal fija sobre la fecha de publicación.

5.2. Representación de documentos

Para realizar nuestros experimentos vamos a trabajar con una colección de periódicos digitales en formato XML. Estos documentos se han obtenido automáticamente del periódico "El País Digital", los cuales originariamente se encontraban en formato HTML. El formato XML a diferencia del formato HTML, permite tanto poder representar el formato del documento como la estructura del documento. La estructura del documento facilita el acceso a cada parte del documento, lo cuál es útil para permitir consultas que combinan estructura y contenido [Veg99, Bae99, Ber99, Lli99]. Las etiquetas XML, permiten además incluir ciertos metadatos interesantes, como son la fecha de publicación, el autor, el titular, etc.

En [Lli99] se propone un DTD, para representar la estructura de los periódicos digitales españoles en XML, en la figura 5.2 se muestra un fragmento de dicho DTD. En [San98] se realizó un análisis de las etiquetas HTML que junto con el conocimiento de las características estructurales del periódico (artículos, secciones, imágenes, titulares, despieces...) permite pasar de forma semiautomática del formato original HTML a un formato semi-estructrurado (XML) como el que se muestra en la figura 5.3.

```
<!ELEMENT  Noticia (seccion?,cintillo?,Imagen?,antetitulo*,
     titular+,Imagen?,Data,subtitulo*,Contenido+,Despiece*)>
<!ELEMENT Despiece (antetitulo*,titular+,Data?,Contenido+)>
<!ELEMENT  Data (autor,lugar?)>
<!ELEMENT  Imagen (foto,pie?)>
<!ELEMENT Contenido (parrafo|Imagen)+)>
```

Figura 5.2: DTD de una noticia.

```
 <Noticia>
  <seccion>INTERNACIONAL</seccion>
  <cintillo> Sección especial....</cintillo>
  <titular> Alta Tensión</titular>
  <Data>
           <autor>W. Pérez</autor><lugar>Bruselas </lugar>
  </Data>
  <Contenido>
    <Imagen>
     <foto><img src="i.jpg"></foto> <pie> Imagen de ...</pie>
    </Imagen>
    <parrafo>
    La puesta en marcha ...
    </parrafo>
    ...
</Noticia>
```

Figura 5.3: Fichero XML de una noticia.

Para trabajar con estos documentos dentro de los sistemas propuestos en este capítulo se propone la siguiente representación. Cada documento d_i, viene caracterizado por:

- Un código de estructura (*scode* [Ber99, Lli99]) que es un identificador único del documento que representa el lugar que ocupa el documento en la jerarquía de agregación de los documentos que forman parte de cada periódico (ver figura 5.5).
- Un vector de términos $T^i = (TF_1^1, \ldots TF_n^i)$, siendo TF_k^i la frecuencia relativa de cada término t_k dentro del documento d^i.
- Un vector $F^i = (TF_{f_1^i} \ldots TF_{f_k^i})$ con los instantes e intervalos de fechas referenciados en el documento, donde $TF_{f_m^i}$ representa la frecuencia absoluta de f_m en el documento d^i.
- La fecha de publicación del documento pd^i (*publication date*) .
- El *periodo de suceso* del documento et^i (*event time*).

Las características anteriores se obtienen tras un preprocesado de los artículos de los periódicos, el cual se describe a continuación:

- El vector de términos T^i se obtiene a partir de las frecuencias de aparición de las palabras en cada documento. Al conjunto de palabras presentes en cada artículo se aplica una lista de palabras de parada (Anexo C) y se reduce al mismo término todas las palabras que tienen el mismo lema. Para ello utilizamos el lematizador y analizador morfológico para el español MACO [Tur99, Cas98]. Así pues, con los términos de cada documento se construirá el vector de pesos T^i, donde se asignará como peso la frecuencia de los términos normalizada TF (*Term Frequency*).
- El vector de instantes F^i se obtiene tras procesar los documentos con TimeExtractor, tomando todas las entidades temporales codificadas con tipo DATE. Las entidades temporales se refinan al nivel de días, y se crea un vector con las fechas e intervalos que aparecen en cada documento. Finalmente a cada elemento del vector f^i_j, se le asigna un peso según su frecuencia de aparición en el documento ($TF_{f^i_j}$).
- La fecha de publicación de un documento pd^i es un metadato del documento.
- El *periodo de suceso* et^i es un metadato que se asigna al documento automáticamente tras el procesado de éste con TimeExtractor. El algoritmo para obtener el periodo de suceso de modo automático se describe en la sección 5.3

Para simplificar todos los cálculos hemos decidido expresar todas las entidades temporales en días. Si se dispone de una entidad temporal a una granularidad superior, con un preproceso se refina a días. De este modo, facilitaremos las operaciones entre entidades temporales sin perder información.

5.3. Cálculo automático del *periodo de suceso*

Según la definición propuesta en el modelo TOODOR [Ara98c], el *periodo de suceso* de un documento es aquel *intervalo temporal en el cual se desarrolla la acción principal del artículo*. Según esta definición, el cálculo del *periodo de suceso* puede variar según la interpretación del lector que lo analice, bien porque cada lector puede destacar un hecho distinto como principal, bien porque según los conocimientos del lector, éste podrá o no resolver las referencias temporales dependientes de otros sucesos (ej. 'dos días después de la firma del convenio'). Por lo tanto la asignación del *periodo de suceso* posee una incertidumbre intrínseca y debe ser aproximada.

A partir del estudio de las fechas que aparecen en los documentos, podemos establecer varias hipótesis que nos permitirá asignar a cada documento un *periodo de suceso*. Éstas son:

I- Las noticias se publican en fechas próximas a la fecha del suceso.
II- Las referencias temporales lejanas a la fecha de publicación son referencias a sucesos indirectamente relacionados con la acción principal del artículo.
III- La relevancia de las fechas que abarca un intervalo temporal es inversamente proporcional a la duración del intervalo. A modo de ejemplo, las fechas que aparecen dentro del intervalo 'de lunes a jueves' son más relevantes a las fechas que aparecen en el intervalo 'este verano'.
IV- Cuando los sucesos no son puntuales, generalmente se utilizan las fechas más destacadas de los sucesos, entre las cuales puede haber huecos de varios días, de modo que el *periodo de suceso* debe ser el intervalo que recubra las fechas destacadas.
V- Generalmente, si no se cita cuando ocurre el suceso es debido a que ocurrió el día anterior. En nuestros estudios cuando se habla de un suceso que se produce el día de la publicación o los dias inmediatamente posteriores, se suele citar alguna expresión temporal que referencia la fecha del suceso.

Cabe destacar que de los 554 documentos periodísticos analizados, solo 30 ellos no poseen ninguna referencia temporal, y existen 13 documentos que, aunque poseen referencias temporales, ninguna de ellas se encuentra a menos de 7 días de la fecha de publicación. La mayoría de estos 13 documentos corresponden con despieces de una noticia donde se habla de sucesos históricos relacionados con la noticia.

A continuación describimos un algoritmo para calcular el *periodo de suceso* a partir de las entidades temporales presentes en el documento. Cabe destacar que no todas las entidades temporales contribuyen por igual al *periodo de suceso*, ya que algunas referencias temporales que aparecen en el texto estarán relacionadas con los sucesos o tópicos principales, mientras que otras los relacionarán con otros sucesos similares o que involucran a los agentes del suceso descritos en el documento.

Vamos a definir el *periodo de suceso* como *aquel intervalo de fechas con mayor frecuencia de aparición próximo a la fecha de publicación.* Siguiendo esta definición, se propone el algoritmo 5.1 para el cálculo automático del *periodo de suceso* a partir de las referencias temporales de cada documento, siempre definidas al nivel de días. El algoritmo para el cálculo del periodo de suceso requiere cinco parámetros de entrada: la fecha de publicación, la lista de entidades temporales extraídas con TimExtractor, un umbral de relevancia para las entidades temporales (*threshold*), el hueco de días máximo permitido entre dos fechas relevantes (*gap*),

Algoritmo 5.1 Event Time Algorithm.

Entradas: pd^i, F^i, $threshold$, gap, $maxdist$
{pd^i: publication date ; F^i: list of extracted time entities ; $threshold$: lowest frequency for a relevant time entity ; gap: maximum distance between relevant time entities ; $maxdist$: maximum days-distance between two proximal dates}
Salidas: $EventTime$

$[D\_Dates, D\_Intervals] = ExtractDatesIntervals(F^i)$
{Split F^i in two hash tables: date frequencies (D_Dates), and interval frequencies ($D\_Intervals$).}
for all $[f_1, f_2] \in D\_Intervals$ **do**
  if $dayDistance(f_1, f_2) \leq maxdist$ **then**
    add to D_Dates all dates between f_1 and f_2
    delete $[f_1, f_2]$ from $D\_Intervals$
  else
    add to D_Dates the dates f_1 and f_2
  end if
end for
$F0 =''$; $cont = 0$; $F1 =''$; $FI =''$; $FF =''$
for $F1 \in D\_Dates$ **do**
  if $F0 ==''$ **then**
    continue
  else
    if $dayDistance(F1, F0) < gap$ **then**
      $cont+ = D\_Dates[F1]$; $FF = F1$
    else
      $D\_Intervals[FI, FF] = cont$
      $F0 = F1; FI = F1; FF = F1; cont = 0$
    end if
  end if
end for
$EventTime = MostRelevantInterval(D\_Intervals, maxdist, pd^i, threshold)$
{Obtain most relevant interval near pd^i}
if not $EventTime$ **then**
  $EventTime = [pd^i - 1, pd^i]$
end if

y la duración máxima de intervalo para considerar cada una de sus fechas constituyentes como relevante ($maxdist$). Los valores de gap, $threshold$, y $maxdist$ se ajustan experimentalmente.

En la línea 1, la función $ExtractDatesIntervals()$ divide el vector de frecuencias de fechas F^i en dos, uno con las entidades de tipo fecha y otro con el de los intervalos. A continuación, las fechas de los extremos de cada intervalo son añadidas al vector de fechas (líneas 3-10). Si la duración del intervalo es menor que $maxdist$, también añadiremos las fechas intermedias, aplicando la hipótesis III anterior.

Las instrucciones entre las líneas 11 y 23, generan intervalos a partir de fechas consecutivas, permitiendo huecos de tamaño máximo de gap días. A los intervalos así generados se le asigna como relevancia la suma de las relevancias de las fechas que lo han constituido, y se añaden al vector de intervalos. Aquí se aplica la hipótesis IV.

Para finalizar, en la línea 24, la función $MostRelevantInterval()$, busca el intervalo más relevante del vector de intervalos, cuya relevancia supera el umbral $threshold$, y que cumpla que alguno de sus extremos diste de la fecha de publicación menos de $maxdist$ días (aplicamos las hipótesis I y II). Si no se ha podido obtener este intervalo, la función devuelve falso. En este caso, el algoritmo, en la línea 27, le asigna como *periodo de suceso* el intervalo formado por el día anterior a la fecha de publicación y la propia fecha de publicación, según la hipótesis V.

Para cada tipo de colección documentos se deben ajustar los parámetros gap, $threshold$ y $maxdist$. Para ello se toma un conjunto de documentos representativo a los que se les asigna manualmente su periodo de suceso. Posteriormente estos documentos se procesan con el algoritmo y se ajustan los parámetros hasta conseguir que se ajuste lo más posible el valor calculado automáticamente al asignado manualmente. En nuestra colección hemos determinado los siguientes valores para estos parámetros: $gap = 7$, $threshold = 2/gap$ y $maxdist = 15$.

En la figura 5.4 se muestra un conjunto de documentos donde se representa mediante un tramo horizontal la duración del *periodo de suceso* calculado según el algoritmo anterior. Mediante un punto se representa la fecha de publicación de ese documento. Se observa que, para esta colección, pocos documentos tienen un periodo de suceso de más de 10 días, y además casi todos suelen comenzar antes de la fecha de publicación.

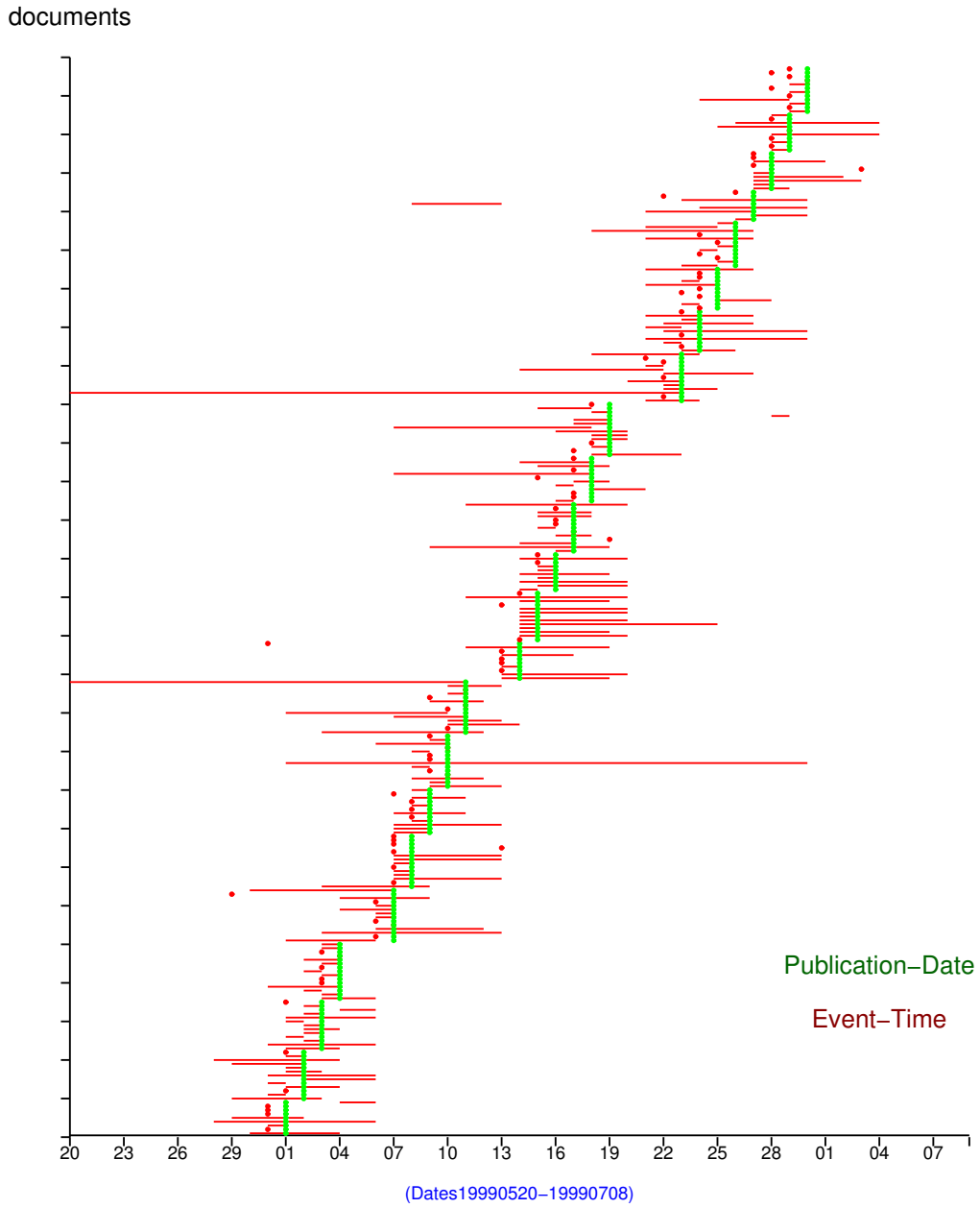


Figura 5.4: Periodo de suceso de los documentos de la colección 'El País Digital'.

5.4. Obtención de Tópicos con un sistema de RI con *periodo de suceso*

TOOOR (Temporal Object-Oriented Document Organisation and Retrieval) es un modelo de base de datos orientado al desarrollo de aplicaciones de bibliotecas digitales con documentos estructurados [Ara98, Ara99]. Su lenguaje y sus constructores están diseñados para permitir la descripción de las estructuras de los documentos, facilitando su manipulación por aplicaciones y usuarios. Una característica que distingue a TOODOR de otros modelos de documentos es el modelo temporal, el cual permite, por un lado, asegurar la integridad de los objetos con respecto a una organización dinámica de sus clases, y por otro permite el desarrollo de operadores de consulta temporales que ayudan al usuario a obtener información histórica de las bibliotecas digitales. Esta segunda característica es la que nos interesa, y la que describiremos a lo largo de esta sección.

Las principales características del modelo TOODOR son:

- Las clases describen las estructuras de los documentos del repositorio, con un juego de tipos que permiten componentes repetitivas y alternativas, de manera similar a los constructores utilizados en SGML.
- Las clases de las estructuras de un documento se pueden modificar a lo largo del tiempo, de modo que las clases en el esquema de la base de documentos no varía, sólo el tipo de datos que contienen esas clases.
- Posee una dimensión temporal asociada al contenido del documento denominada *periodo de suceso*, que representa el periodo de tiempo en el que se produce el suceso más relevante del documento.

El lenguaje de consulta de TOODOR está basado en *OQL* (Object Query Language) y permite recuperar documentos especificando condiciones en la estructura, contenido y propiedades temporales. Este lenguaje de consulta se ha diseñado para permitir relacionar documentos en el tiempo, así como analizar la evolución de los tópicos.

En TOODOR una **clase** es una 5-tupla que contiene:

- *class_id*: identificador de la clase,
- *lifespan*: periodo temporal en el cual la clase está definida,
- *h_type*: secuencia de pares (p_i, T_i) donde T_i es el identificador del tipo de objeto de esa clase durante el periodo p_i,

- *h_population*: secuencia de pares (p_i, I_i) donde I_i es el conjunto de identificadores de los objetos que se han creado en el periodo p_i.

Un **objeto** en TOODOR es una 5 tupla formada por:

- *class*: identificador de la clase,
- *p_time*: fecha en la cual se ha insertado el objeto en la base de datos,
- *event_time*: *periodo de suceso* del objeto,
- *val*: valor del objeto.

5.4.1. Modelo de documentos

Según el modelo TOODOR, los documentos de la biblioteca digital se representan como objetos XML con identidad única y con dos atributos temporales: el tiempo de inserción y el periodo de suceso.

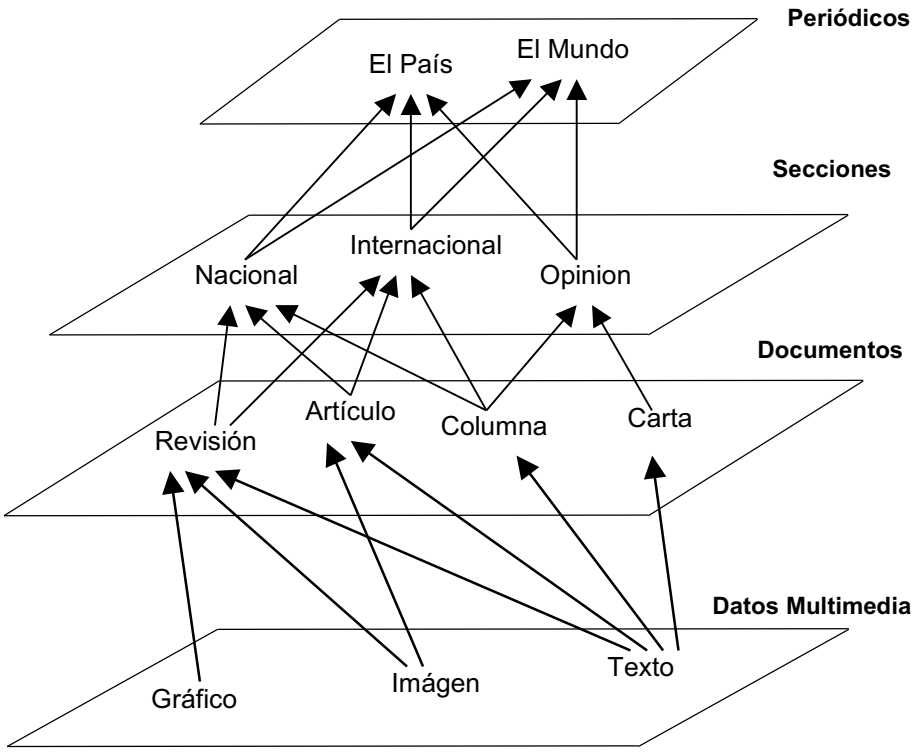


Figura 5.5: Estructura de un periódico.

El modelo TOODOR además permite componer documentos complejos a partir de otros más simples mediante el mecanismo de agregación del modelo orientado

a objetos como se muestra en la figura 5.5. De esta forma los documentos estructurados se representan como árboles cuyas hojas contienen información textual y multimedia.

5.4.2. Modelo de Tiempo

TOODOR utiliza un modelo de tiempo discreto basado en el calendario Gregoriano. En este modelo se definen los puntos, intervalos y duraciones de un modo similar al del Capítulo 3, con la limitación de que solo se permite almacenar estos datos con formato de fecha estándar dd/mm/yyyy.

De este modelo destacan dos operaciones temporales que permiten abstraer una entidad temporal a cualquier granularidad, devolviendo siempre un intervalo de fechas, éstas son:

- **trans_date**$(\mathbf{T}, \mathbf{g}) = [T1, T2]$.
  Esta operación abstrae un punto a cualquier granularidad. Por ejemplo, trans_date(11/4/1997,month)=[1/4/1997,31/4/1997]
- **abs**$([\mathbf{T1}, \mathbf{T2}], \mathbf{g}) = [begin(trans\_date(T1, g)), end(trans\_date(T2, g))]$.
  Esta operación abstrae un intervalo a cualquier granularidad g. Por ejemplo, abs([1/4/1997,4/5/1998],month)=[1/4/1997,31/5/1998]

5.4.3. El lenguaje de consulta TDRL

El lenguaje de consulta del modelo TOODOR se denomina TDRL (Temporal Document Retrieval Language). TDRL tiene como propósito recuperar información de los documentos almacenados en una biblioteca digital por contenido, estructura y/o por propiedades temporales, de modo que los documentos recuperados se ordenan según su orden de relevancia. En una consulta por estructura se pueden especificar restricciones que deben cumplir las estructuras de los documentos o bien relaciones de composición. Las consultas por tiempo permiten seleccionar documentos cuya fecha de inserción y/o periodo de suceso se encuentre en un determinado rango de fechas.

El formato en TDRL de una consulta tiene la forma:

select $V_{goal}[.proj]$
from $Path_1$ as $V_1, \ldots, Path_n$ as V_n
[where $Condition_1$ and ... and $Condition_n$]
[at $TimePeriod$]

Predicado	**Semántica**
contains(*obj*.[*proj*], *IRE*, *Rel*)	El argumento IRE es una consulta de RI sobre un conjunto de palabras clave [28]. Esta expresión se evalúa sobre el texto incluido en un objeto (*obj*) o en una proyección del objeto(*obj.proj*). Este predicado es cierto si el objeto satisface la expresión con una relevancia superior a *Rel*.
$o_1.proj\ comp\ o_2.proj$	El valor proyectado del objeto o_1 es comparado con el de o_2. El operador *comp* compara tipos atómicos de datos(e.g. $=$, $<>$, $>$, $<$, ...).
in(o_1, o_2, [*Position*])	Si el objeto o_2 está incluido en la jerarquía de composición de o_1, el predicado devuelve verdadero. El tercer argumento puede usarse para indicar la posición del objeto o_2 en objetos con componentes multivaluados.

Cuadro 5.1: Predicados de TOODOR para las condiciones estructurales y de contenido.

El resultado de esta consulta es un conjunto de documentos que satisfacen las restricciones de la cláusula *from*, los predicados de la cláusula *where* y las proyecciones temporales de la cláusula *at* sobre el tiempo de inserción.

En la cláusula *from*, además de seleccionar la parte del esquema donde se desea realizar la consulta, se pueden especificar las restricciones sobre la estructura de los documentos a recuperar mediante trayectorias, por ejemplo: P.article(3).title as d3...

En la cláusula *where* se establecen las distintas condiciones sobre :

- las relaciones composicionales entre documentos (ver tabla 5.1),
- el contenido de los documentos (ver tabla 5.1),
- los atributos temporales del documentos (ver tabla 5.2). Todos los predicados temporales se evalúan a la granularidad de días.

Un ejemplo de consulta TDRL es:

Predicado	**Definición**
intersects(x, y)	$x \cap y \neq \emptyset$
intersects(x, y, g)	intersects(abs(x,g),abs(y,g))
starts-before(x, y)	$begin(x) < begin(y)$
starts-after(x, y)	$begin(x) > begin(y)$
before-span(c, x, y)	$begin(x) < (begin(y) - c)$
after-span(c, x, y)	$begin(x) > (begin(y) + c)$
intersects-within(c, x, y)	$end(x) \geq (begin(y) - c) \wedge end(y) \geq (begin(x) - c)$
before-within(c, x, y)	$begin(x) < begin(y) \wedge begin(x) > (begin(y) - c)$
after-within(c, x, y)	$begin(x) > begin(y) \wedge begin(x) < (begin(y) + c)$

Cuadro 5.2: Semántica de los predicados temporales de TDRL: x e y son dos periodos temporales, y c denota una duración de tiempo.

```
select a from Article as a, Column as b
where contains(a, 'political review', 0.8)
and contains(b, 'european meeting', 0.8)
and after-within(10 day, a.et, b.et)
at [05/06/1999,30/06/1999]
```

5.4.4. Crónicas en el lenguaje TDRL

Para el estudio de los sucesos o tópicos que aparecen en los documentos de una biblioteca digital, en el modelo TOODOR se introduce el concepto de crónica. Una *crónica* se define como una colección de documentos con contenido similar y tiempos de suceso cercanos (solapados o contiguos) en función de una determinada precisión que viene dada por una granularidad. A las crónicas se les asocia también un *periodo de suceso*, que se calcula como el intervalo máximo de los periodos de suceso de los documentos que los componen.

En TOODOR se han establecido cláusulas para la extracción crónicas basándose en la posible cadena de sucesos producidos a partir de un suceso inicial relevante. Para generar crónicas, una vez se obtienen todos los documentos de la consulta que describe al suceso o tópico, se realiza su agrupamiento a partir de sus periodos de suceso.

El resultado de agrupar los documentos que cumplen la consulta R y cuyos periodos de suceso se encuentran cercanos utilizando la granularidad g, consiste en una *secuencia de crónicas* que denotaremos como:

$$CH_R^g = ((p1, O1), \ldots, (p_n O_n)) \tag{5.1}$$

Operador	**Patron Temporal**	**Definición del predicado**
event(p, g)	$intersects(x,\ y,\ g)$	$abs(x,\ g) \cap abs(y,\ g) \neq \emptyset$
cascade(p, g)	$overlaps(x,\ y,\ g)$	$begin(abs(x,\ g)) < begin(abs(y,\ g))$ $\wedge\ end(abs(x,\ g)) < end(abs(y,\ g))$ $\wedge\ end(abs(x\ ,g)) > begin(abs(y,\ g))$
sequence(p, g)	$meets(x,\ y,\ g)$	$end(abs(x,\ g)) = begin(abs(y,\ g))$
hierarchy(p, g)	$during(x,\ y,\ g)$	$begin(abs(x,\ g)) < begin(abs(y,\ g))$ $\wedge\ end(abs(x,\ g)) > end(abs(y,\ g))$

Cuadro 5.3: Predicados para la generación de secuencias de crónicas, x y y son tiempos de suceso de los objetos TOODOR pertenecientes a la misma crónica, y g es una granularidad temporal.

Cada crónica consiste en un conjunto de documentos O_i y un periodo de suceso p_i. Para la generación de secuencias de crónicas, en TOODOR se han definido cuatro predicados (ver tabla 5.3) que obtienen secuencias de crónicas con distintas características temporales.

Cada crónica generada por la consulta *event* agrupa documentos similares que hablan del mismo tema y co-ocurren en el tiempo. Sin embargo, éstas no representan la evolución concreta de un suceso, tal y como se pretende con los *tópicos* en TDT; a no ser que el usuario conozca totalmente el tópico y pueda especificar en la consulta todas las características sobre éste. En realidad hay bastantes analogías entre las crónicas y los tópicos, ya que ambos intentan agrupar documentos sobre sucesos, pero existen dos diferencias fundamentales entre ambos conceptos:

- un tópico también agrupa los sucesos que se producen como consecuencia de un suceso origen, aunque éste trate sobre distintos temas, es decir se trata de agrupar documentos semejantes entre sí y no semejantes a una consulta fija que describe el tema del tópico.
- una crónica siempre tiene asociado un *periodo de suceso*, lo que permite su manejo como un documento más del repositorio.

En nuestros estudios experimentales [Lli99b] se observa que con la consulta TDRL *event* no se obtienen realmente tópicos debido principalmente:

- a que las condiciones temporales no son muy restrictivas y dan lugar a que los documentos con periodos de suceso grandes aglutinen distintos sucesos en un solo grupo, y

- a la dificultad por parte del usuario de especificar todos los conceptos que relatan un tópico, o sea un suceso y su evolución. Generalmente porque no los conoce y es lo que trata de averiguar, y por otro lado porque los conceptos evolucionan en el tiempo, y esto no se puede especificar en una sola consulta. Sólo recuperamos los documentos relevantes al tema especificado en la consulta, o sea algunos de los documentos que relatan sucesos relacionados con el tema de la consulta.

5.4.5. TimExpIR: Exploración en el tiempo de los documentos en un sistema de RI

En esta sección vamos a proponer una herramienta, denominada TimExpIR, que permita acceder a los documentos agrupados por sus semejanzas temporales, utilizando el *periodo de suceso* de cada documento. Esta herramienta permitirá explorar los documentos por sus propiedades temporales, así como analizar todos los sucesos relacionados con un tópico. Para ello vamos a representar con diversas representaciones visuales los documentos recuperados y agrupados por el solapamiento temporal de sus *periodos de suceso*. Esta herramienta se ha creado como una extensión al modelo TOODOR, aunque se puede aplicar a cualquier sistema de RI que incluya el periodo de suceso como metadato de los documentos.

Para realizar estas representaciones gráficas dividiremos el espacio temporal en intervalos, y agruparemos los documentos relevantes que se solapan en estos intervalos temporales, de modo que un documento puede pertenecer a más de una partición según la duración de su periodo de suceso. La división del espacio temporal puede realizarse de dos modos diferentes:

- Con particiones donde todos los intervalos tienen la misma duración temporal, o sea mediante una partición regular del espacio temporal, en cuyo caso tendremos un **histograma temporal**. Cada barra del histograma representa una división temporal con una determinada granularidad, y su altura representa la relevancia del tema en función de los documentos relevantes cuyo periodo de suceso se solapa con esa división temporal.
- Mediante agrupaciones con intervalos con distintas duraciones, mediante una división irregular basada en la agrupación de particiones consecutivas que cumplen ciertas restricciones, como veremos en el apartado 5.4.7. Este tipo de agrupación lo denominaremos *crónica*, ya que cada partición representa un tópico al cual le podemos asignar un periodo de suceso, tal y como se define en TOODOR.

Los usuarios suelen estar muy familiarizados con gráficas que muestran la evolución temporal, las cuales son una forma sencilla de analizar la información que varia en el tiempo (ej. evolución de la bolsa, planes de trabajo, etc).

Mediante estas gráficas, el usuario puede analizar mejor y con más facilidad los documentos recuperados por un sistema. Así, si conoce aproximadamente cuándo se ha producido un suceso, se puede desechar fácilmente muchos documentos no relacionados con él y analizar solo los documentos que están en las particiones cercanas al instante en el que se cree que se ha producido el suceso. Por otro lado si no se sabe cuándo ha ocurrido el suceso, analizando sólo los primeros documentos de cada partición se puede detectar más fácilmente el grupo de documentos de interés o bien se puede conocer cuántos sucesos sobre un tópico determinado se han producido.

En los trabajos de TDT, podemos encontrar sistemas que utilizan gráficas temporales para explorar los sucesos clasificados por el sistema [Swa00, Fre01, Swan00b]. En la aplicación TimeMines [Fre01] se intenta representar en el eje temporal cada tópico de un suceso detectado por el sistema descrito en [Swa99] mediante una barra. El grosor de cada barra es relativo a la relevancia de cada suceso, y la longitud de la barra abarca las fechas mínima y máxima en las cuales se han publicado las noticia referentes a dicho suceso. Al pulsar sobre cada barra aparece un menú con las palabras clave que identifican el tópico y los documentos que forman parte de él. Este tipo de herramienta permite obtener los sucesos más relevantes narrados en la colección y relacionar los distintos sucesos detectados.

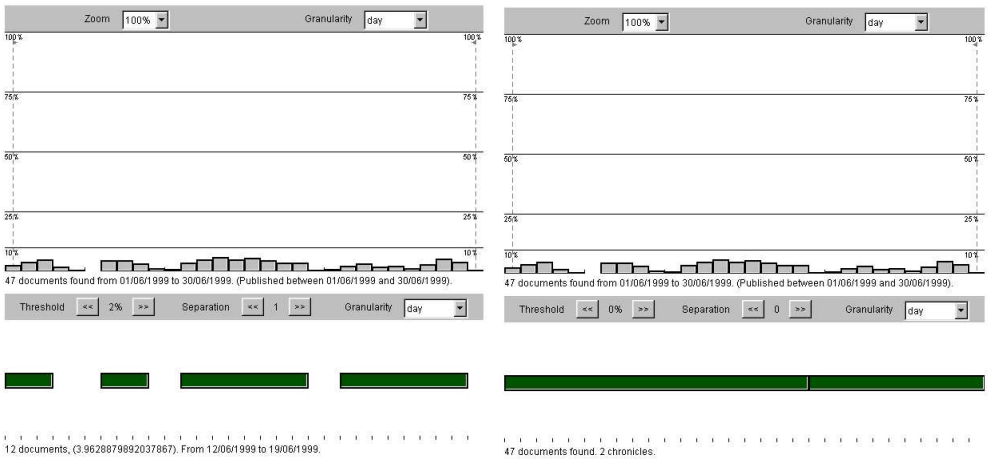


Figura 5.6: Crónicas-TimExpir / Crónicas-event.

En la figura 5.6 se presenta las crónicas generadas con TimExplR y se comparan con las crónicas de TOODOR donde se ha consultado información sobre México

en junio de 1999 utilizando un *threshold*=2 y un $gap = 1$ en TimExpIR. Mientras con las crónicas de TOODOR parece que sólo tengamos un suceso, con TimExpIR descubrimos cuatro sucesos distintos donde los documentos relacionados con cada crónica aparecen al pulsar sobre cada crónica. Las crónicas obtenidas son: 'La oposición pide al presidente mexicano que diga si hubo dinero negro en su campaña', 'Conmoción en México por el asesinato a tiros de un famoso presentador', 'La vuelta de Salinas solivianta a México', 'Europa y Latinoamérica pactarán en Brasil una alianza estratégica para el siglo XXI'.

Aunque describiremos con más detalle en las siguientes secciones la herramienta TimExpIR, avanzamos que esta nos va a permitir las mismas ventajas que la herramienta de exploración gráfica de los tópicos *TimeMines* [Fre01], al permitir explorar los documentos recuperados por el sistema de RI, a partir de sus propiedades temporales.

5.4.6. Histograma Temporal

Para la generación de un histograma temporal dividiremos el espacio temporal que recubre los periodos de suceso de la colección en un conjunto de puntos $\{f_i\}_{0 \leq i \leq k}$ del tamaño de la granularidad g, y agruparemos todos aquellos documentos que cumplen la consulta IRE en cada punto temporal f_i en una partición P_i, formada por aquellos documentos cuyo *periodo de suceso* se solapa en el punto temporal f_i.

Sea $divisiones(D, g) = \{f_i\}_{0 \leq i \leq k}$ el conjunto de puntos temporales a la granularidad de g que recubren el espacio temporal de los periodos de suceso del conjunto de documentos D.

Definiremos el conjunto de particiones para un conjunto de documentos D[2], que cumple la consulta IRE sobre una granularidad g como:

$$Particiones(D, IRE, g) = \{P_i\}_{0 \leq i \leq k} \; con \; P_i = (D^{P_i}, f_i, r^{P_i}) \tag{5.2}$$

donde:

- $f_i \in divisiones(D, g)$
- $D^{P_i} \subset D \; \wedge \; \forall d_j \in D^{P_i} \; r_{j,IRE} > 0 \wedge et_j \cap f_i \neq \emptyset$
- Definiremos la relevancia de una partición como:

$$r^{P_i} = \frac{|D^{P_i}|}{max(|D^{P_j}|)_{j \neq i, 1 \leq j \leq k}} \cdot rel(P_i)$$

[2]Los documentos se representan según hemos visto en la sección 5.2.

$$\text{siendo } rel(P_i) = \frac{\sum_{\forall d_j \in D^{P_i}} (r_{j,IRE}/|D^{P_i}|) + max(r_{j,IRE})_{\forall d_j \in D^{P_i}}}{2}$$

La relevancia de una partición se obtiene a partir del promediado de las relevancias de los documentos de cada partición y el número de documentos de la partición, pero balanceando este promedio con el valor del documento con máxima relevancia. Para dar mayor peso a las particiones con más documentos, el valor de $rel(P_i)$ se multiplica por el número de documentos de la partición y se normaliza dividiendo por el número máximo de documentos de todas las particiones.

El histograma generado con estas particiones permite al usuario realizar un análisis sobre la evolución temporal de un tema, en qué periodos ha sido relevante, y el nivel de relevancia del tema en cada punto temporal.

Algoritmo 5.2 Partition Algorithm

Entradas: IRE, g, D
$\{IRE$: query ; g: granularity ; $D = \{d_i\}/d_i = (scode_i, et_i)$: Documents$\}$
Salidas: $\{Partitions\}$
1: $Q = ConsultaTDRL(IRE, D)$ $\{Q = \{(d_i, r_{i,IRE})\}$ IRE relevant documents$\}$
2: $P = [\,]$
3: **for** $(d, r) \in Q$ **do**
4: $(scode, et) = d$
5: $Points = points\_event\_time(et, g)$ {split the event-time using the granule g}
6: **for** $i \in Points$ **do**
7: $P[i] += [(d, r)]$
8: **end for**
9: **end for**
10: $Max = 0$
11: **for** $i \in P.keys()$ **do**
12: $rel[i] = relevance(P[i], Max)$
13: **end for**
14: **for** $i \in P.keys()$ **do**
15: $Partitions[i] = (P[i], i, \frac{|P[i]|}{Max} * rel[i])$
16: **end for**

Algoritmo 5.3 Relevance Algorithm

Entradas: $List$, Max
$\{List = \{l\} \ / \ l = (d_i, r_{i,IRE})$; Max: maximal relevance$\}$
Salidas: Rel, Max
1: $max_r = 0$; $suma = 0$
2: **if** $|List| > Max$ **then**
3: $Max = |List|$
4: **end if**
5: **for** $(d, r) \in List$ **do**
6: $suma+ = r$
7: **if** $r > max_r$ **then**
8: $max_r = r$
9: **end if**
10: **end for**
11: $media = suma/|List|$
12: $Rel = (media + max_r)/2$

5.4.7. Crónicas de un suceso

Si analizamos los histogramas obtenidos con el algoritmo de la sección anterior, observamos que todos los documentos que hablan sobre el mismo suceso suelen encontrarse en particiones contiguas. Por otro lado cuando tenemos dos particiones contiguas y no existe ningún documento en común, generalmente es debido a que se habla de dos sucesos distintos. También hemos observado que si agrupamos todas las particiones relevantes consecutivas cuya intersección no es vacía, se genera una secuencia de grupos de documentos, donde la mayoría de documentos que pertenecen al mismo grupo relatan el mismo suceso. En el estudio de estas crónicas consecutivas que hablan sobre el mismo suceso, observamos que contienen documentos comunes y que se encuentran relativamente a poca distancia, esto nos permite agrupar las particiones que contengan documentos comunes y que se encuentren a una cierta distancia como pertenecientes al mismo grupo. El resultado final de estos agrupamientos corresponde con el concepto de *crónica* que definimos en la sección 5.4.4.

Formalmente definiremos como $Colecciones$, a una *secuencia de crónicas de un suceso* generadas a partir del agrupamiento de las particiones generadas para un conjunto de documentos D, que superan un valor de relevancia umbral β_{rel} respecto a la consulta IRE, y que distan menos de gap divisiones de granularidad g. Formalmente:

$$Coleccion(D, IRE, g, \beta_{rel}, gap) = \{(D^{C_k}, et^{C_k}, r^{C_k})\} \tag{5.3}$$

donde:

- $D^{C_k} \subseteq D$
  Los documentos de una crónica C_k, estarán formados por aquellos documentos de las $Particiones(D, IRE, g)$ que cumplan:
  - si $r^{P_l} > \beta_{rel} \Rightarrow D^{P_l} \subseteq C_k$
  - si $D^{P_l} \subseteq C_k \Rightarrow D^{P_{l+1}} \subseteq C_k$ sii
    a) $D^{P_l} \cap D^{P_{l+1}} \neq \varnothing \wedge r^{P_{l+1}} > \beta_{rel}$ ó
    b) $\exists P_m \in Particiones(D, IRE, g)),\ l+1 \leq m \leq gap + l$
    $r^{P_m} > \beta_{rel}\ \wedge\ \forall j \in [l, m]\ D^{P_j} \cap D^{P_{j+1}} \neq \varnothing$
- $r^{C_k} = \dfrac{|C_k|}{max(|C_m|)_{m \in Coleccion(D,IRE,g,threshold,gap)}} \cdot rel(C_k)$
  $rel(C_k) = \dfrac{\sum_{\forall d_j \in D^{C_k}} (r_{j,IRE}/|C_k|) + max(r_{i,IRE})_{\forall d_i \in C_k}}{2}$
- $et^{C_k} = [min(first(et_j, g))_{\forall d_j \in D^{C_k}}, max(last(et_j, g))_{\forall d_j \in D^{C_k}}]$[3]

[3] $first$ y $last$ son las operaciones definidas en el modelo del tiempo 3.5 para obtener el principio y final de un intervalo.

Algoritmo 5.4 Temporal Group Algorithm

Entradas: IRE, g, D, gap, β_{rel}
$\{IRE$: query ; g: granularity ; $D = \{d_i\}/d_i = (scode_i, et_i)$: Documents$\}$
gap\{duration expressed with granularity g $\}$
β_{rel} { relevance threshold$\}$
Salidas: C $\{C = \{C_l\}|C_l = (D_l, etc_l, rc_l)$ y $[d_i] \subseteq D\}$

```
1: is_group = false;  distance = 0
2: LP = partition(D, IRE, g) ; Q = []
3: for (P, f, r) ∈ LP{LP=documents list, f the temporal point, r partition relevance } do
4:    if is_group == false and r > β_rel  then
5:       is_group = true; Docs = P
6:       begin = p ; end = p; aux = [ ]
7:    else
8:       if is_group == true then
9:          if P ∩ Q <> ∅ and distance < gap then
10:            aux = aux + P
11:            if r > β_rel then
12:               end = p ; distance = 0 ; Docs+ = aux ;aux = [ ]
13:            else
14:               distance+ = 1
15:            end if
16:         else
17:            {add a new group}
18:            C[begin +' −' + end] = Docs ; Docs = []
19:            is_group = true; distance = 0 ; aux = [ ]
20:            if r >= β_rel then
21:               docs = [ P ] ; begin = p ; end = p ; is_group = true
22:            end if
23:         end if
24:      end if
25:   end if
26:   Q = P
27: end for
28: if is_group then
29:   C[begin +' −' + end] = docs
30: end if
31: Max = 0
32: for i ∈ C.keys() do
33:   rel[i] = relevance(C[i], Max)
34: end for
35: for i ∈ C.keys() do
36:   Cronica.append(C[i], i, rel[i] ∗ (|C[i]/Max))
37: end for
```

En el algoritmo 5.4 se describen los pasos a seguir para la obtención de *la secuencia de crónicas de un suceso*. Primero se generan las particiones del histograma temporal llamando al algoritmo 5.2 en la línea 2, y luego se agrupan para formar las distintas crónicas (líneas 3-31). Una vez obtenidos los documentos de

cada crónica, se calcula la relevancia de cada crónica (líneas 31-34), llamando al algoritmo 5.3, y se finaliza el programa creando una lista de crónicas, donde cada crónica se caracteriza por una lista de documentos, su periodo de suceso y su relevancia.

Cabe destacar que las crónicas definidas en TimeExpIR, en el caso de que se establezca β_{rel}=0 y $gap = 0$, coinciden con las crónicas definidas en TDRL como de tipo event. Los parámetros β_{rel} y gap, dependen de cada suceso. El gap puede ser grande en el caso de sucesos muy largos pero no muy relevantes, como por ejemplo un juicio, pero en sucesos muy relevantes como las guerras, aunque sean largos, el gap puede ser pequeño. En sucesos largos como los términos involucrados evolucionan deberemos poner un β_{rel} pequeño.

Hay que destacar que a diferencia de los sistema de detección de sucesos, las *crónicas de un suceso* generadas por este algoritmo podrían no incluir todos los documentos que hablan sobre el suceso. Esto es debido a que la consulta IRE recupera únicamente los documentos que contienen una determinada combinación de las palabras clave, lo que puede excluir documentos relevantes relacionados que no contengan dicha combinación. Así, este sistema será capaz de recuperar todos los documentos de un tópico según TDT-3, sólo en el caso de sucesos cuyas palabras clave más descriptivas no varíen en el tiempo.

5.4.8. Interfaz gráfica de TimExpIR

En la figura 5.7 se muestra interfaz gráfica de TimExpIR. Esta compuesta de cuatro ventanas. En la primera ventana se especifica la consulta por contenido, estructura y tiempo. En el campo *keywords* se especifican los términos unidos con los operadores *and* o *or*, permitiendo el uso de los comodines % y $*$. En el campo *Section* y *Select* se especifica la sección y subsección en la que se deben buscar los términos. En el campo *Temporal window* se especifica el intervalo temporal en el cual deben estar comprendidas las fechas de publicación de los documentos a seleccionar. Al pulsar el botón *Represent* se genera en la segunda ventana un histograma y una secuencia de crónicas. El histograma representa las particiones de los documentos a la granularidad temporal especificada en el campo *Granularity*. Variando el *Threshold* y el *Gap* modificamos los parámetros para la generación de las crónicas sobre el suceso que se especifica en la consulta.

Al seleccionar bien una barra del histograma o de la secuencia de crónicas, en la tercera ventana aparece una lista con los titulares de los documentos pertenecientes a esa partición. Y al seleccionar uno de los titulares en la cuarta ventana, aparece el documento completo.

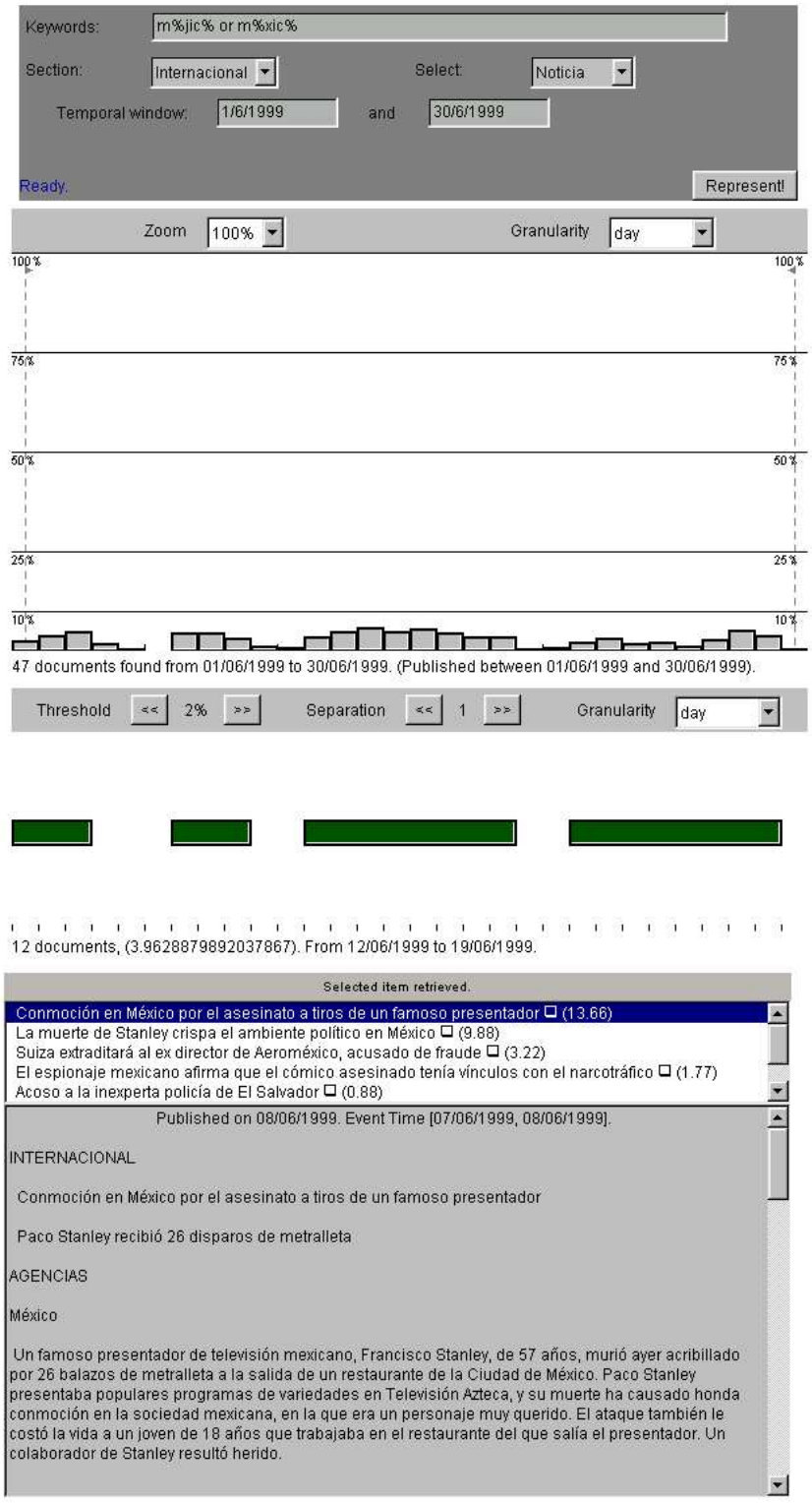


Figura 5.7: Interfaz gráfica.

5.5. Detección de tópicos

En esta sección, vamos a ver cómo podemos generar automáticamente los tópicos presentes en una colección de documentos, utilizando la semejanza conceptual y la distancia temporal entre los documentos. Estas semejanzas se explican a continuación.

- La *semejanza conceptual* trata de expresar cuánto se parecen dos documentos por sus términos. Para obtener dicha semejanza utilizaremos la distancia del coseno que denotaremos por S_t, ampliamente utilizada en los sistemas de RI y también en TDT.

  Sean d^i, d^j dos documentos y TF_k^i la frecuencia de aparición del término k en el documento d^i, definiremos

  $$S_t(d^i, d^j) = \frac{\sum_{k=1}^{n} TF_k^i \cdot TF_k^j}{\sqrt{\sum_{k=1}^{n}(TF_k^i)^2} \cdot \sqrt{\sum_{k=1}^{n}(TF_k^j)^2}} \tag{5.4}$$

  donde n es el número total de términos en la colección de documentos.

- Para calcular la *distancia temporal* entre dos entidades temporales adoptaremos la *distancia de Minkowsky* [Ich94], que denotaremos como d.

  Sean f_1 y f_2 dos intervalos, se define la distancia de Minkowsky como:

  $$d(f_1, f_2) = |f_1 \oplus f_2| - |f_1 \otimes f_2| + \rho \cdot (2 \cdot |f_1 \otimes f_2| - |f_2| - |f_1|) \tag{5.5}$$

  donde:

  - $f_1 \oplus f_2$ es el intervalo unión,
  - $f_1 \otimes f_2$ es el intervalo intersección,
  - y $|f|$ es el número de días entre los extremos del intervalo.

  Para el cálculo de la distancia de Minkowsky si alguno de los argumentos (f_i) es una fecha en vez de un intervalo, lo sustituiremos por el intervalo formado por esa fecha en sus dos extremos, o sea ($[f_i, f_i]$). Nótese que si f_1 y f_2 son fechas, la distancia de Minkowsky devuelve la distancia en días entre f_2 y f_1.

A partir de las definiciones anteriores vamos a proponer tres métodos para la detección de tópicos. Como el problema de detección de tópicos es un problema de clasificación no supervisada utilizaremos en estos sistemas el algoritmo de *Single-Pass*. Este algoritmo es fácil de implementar y como se demuestra en sistemas de TDT [Yan00, Yan98b] se obtienen resultados casi similares a la utilización

Algoritmo 5.5 Algoritmo de detección de tópicos inmediata

Entradas: $Corpus, threshold$
Salidas: $Clusters$

```
1: for all article ∈ Corpus do
2:    max = threshold
3:    for all class ∈ Clusters do
4:       s = similarity(article, class, β)
5:       if s > max then
6:          max = s; c_i = class
7:       end if
8:    end for
9:    if c_i then
10:      re_compute(c_i, article)
11:   else
12:      new_cluster(clusters, article)
13:   end if
14: end for
```

de algoritmos de mayor calidad como puede ser el K-NN. La efectividad de este algoritmo depende del orden de procesado de la colección, lo cual suele ser su desventaja principal. Sin embargo, en nuestro caso, como el orden de procesado es igual al orden de las fechas de publicación esta desventaja incluso puede resultar beneficiosa.

La diferencia entre los métodos propuestos viene dada por la función de similitud que dependerá de la propiedad temporal que utilicemos, así pues definiremos:

- Un sistema TDT que solo contemple la semejanza conceptual, el cual denominaremos *No-Time*.
- Un sistema TDT que utilice todas las referencias temporales que aparecen en el documento, que denotaremos como *Dates*.
- Y un sistema TDT que agrupe los documentos en función de su periodo de suceso, que denotaremos como *Event Time*.

En el algoritmo de 5.5 se describe el algoritmo *Single-Pass*, adaptado a la detección de tópicos. Como entrada se utiliza una colección de documentos representados con las propiedades descritas en la sección 5.2, y se establece un umbral de similitud (*threshold*) entre los documentos y cada clase, donde cada clase trata de representar un tópico de forma similar a un documento. Así, para cada documento se busca la clase c_k que maximize la función de semejanza *similarity*. Si ésta supera el *threshold* se recalcula la clase c_k añadiendo el documento a la

clase con la función *re_compute*. En caso contrario, se crea una nueva clase con la función *new_cluster* a partir de las propiedades del documento.

En la sección 5.6 evaluaremos los tres métodos propuestos. De momento avanzamos que el uso del *periodo de suceso* permite obtener resultados similares al uso de las fechas presentes en los documentos, y ambos mejoran sustancialmente los resultados obtenidos sin tener en cuenta las propiedades temporales.

5.5.1. Tópicos a partir de las referencias temporales

En esta sección vamos a definir un sistema de detección de tópicos que utiliza como propiedades temporales de un documento el conjunto de referencias temporales presente en él. En este sistema un documento se representará por una tupla, que contiene el identificador del documento d_j, el vector de frecuencias de términos TF^j, y el vector de frecuencias de fechas F^j. Como semejanza temporal entre dos documentos utilizaremos la semejanza propuesta en [Pons01, Pon02]. La definición de esta semejanza, es similar a la medida del coseno utilizada en la semejanza conceptual, pero utilizando la proximidad temporal definida por la *distancia de Minkowsky* (ver ecuación 5.5).

Sean d^i, y d^j dos documentos, definiremos

$$S_f(d^i,d^j) = \frac{\Sigma_{k=1}^{m_i} TF_{f_k^i} \cdot TF_{s(f_k^i,d^j)} \cdot g(F_k^i, s(f_k^i,d^j)) + \Sigma_{k=1}^{m_j} TF_{f_k^i} \cdot TF_{s(f_k^j,d^i)} \cdot g(F_k^j, s(f_k^j,d^i))}{(2+\mid m_i - m_j \mid) \cdot \sqrt{\sum_{k=1}^{m_i} TF_{f_k^i}^2} \cdot \sqrt{\sum_{k=1}^{m_j} TF_{f_k^j}^2}} \tag{5.6}$$

donde:

- m_i es la cantidad de entidades temporales en el documento d^i,
- $s(f_k^i, d^j)$ devuelve la fecha de d^j más próxima a f_k^i según la distancia de Minkowsky,
- la función g se define en función de la distancia de Minkowsky como:

$$g(f_1,f_2) = \begin{cases} 1: & if\ d(f_1,f_2) = 0, \\ 0{,}8: & if\ d(f_1,f_2) = 1, \\ \frac{1}{\sqrt{d(f_1,f_2)}}: & otherwise. \end{cases}$$

Ponderando las dos medidas de semejanza, la semejanza conceptual (ecuación 5.4) y la semejanza temporal (ecuación 5.6), obtendremos la semejanza global entre entre dos documentos d^i y d^j, como:

$$S(d^i,d^j) = \omega_t S_t(d^i,d^j) + \omega_f S_f(d^i,d^j)$$

donde ω_t y $\omega_f \in [0,1]$ representa el peso de los distintos componentes.

A partir de esta semejanza definiremos la función *similarity* del algoritmo de detección como:

$$similarity(d^i, d^j) = \begin{cases} S(d^i, d^j): & if\ S_f(d^i, d^j) > \beta_f \wedge S_t(d^i, d^j) > \beta_t \\ 0: & otherwise. \end{cases}$$

donde β_f es el umbral temporal, y β_t es el umbral para los términos.

Cada agrupación de documentos que genera el algoritmo de detección es un tópico. Un tópico c_k en este sistema se representará como los documentos con una tupla que contiene: una lista con los documentos del grupo $D^k \subset D$, un vector con las frecuencias de los términos TF^k, y un vector con las frecuencias de las entidades temporales F^k. Cada vez que se crea una nueva clase, la función $new\_cluster(d_j)$ genera una nuevo tópico $c_k = (d_j, T^j, F^j)$. Cuando se añade un documento a un tópico existente con la función $re\_compute(c_k, d_j)$ se añaden las propiedades del documento d_j a la clase c_k, así pues $c_k = (D^k \cup \{d_j\}, T^{c_k} + T^j, F^{c_k} + F^j)$.

5.5.2. Tópicos a partir del *Periodo de suceso*

Vamos a definir un sistema de detección de sucesos, que utiliza para agrupar los documentos el algoritmo *Single-Pass*, y el *periodo de suceso*. Un documento en este sistema vendrá representado por un trio formado por el identificador del documento d_i, el vector de frecuencias de términos TF^i, y el periodo de suceso del documento et^i. En este sistema la similitud entre dos documentos se calculará en función de la semejanza conceptual (ver ecuación 5.4) y la proximidad temporal según la distancia de Minkowsky (ver ecuación 5.5) entre los *periodos de suceso*:

Sea d^i y d^j, definiremos

$$similarity(d^i, d^j) = \begin{cases} S_t(d^i, d^j): & If\ (d(et^i, et^j) > \beta_f) \\ 0: & otherwise. \end{cases}$$

donde β_f es el la distancia máxima permitida entre los periodos de suceso expresados en días.

En esta aproximación cada tópico c_k se representa al igual que los documentos con una tupla que contiene: una lista con los documentos del grupo $D^k \subset D$, el vector de frecuencias de los términos F^k y el periodo de suceso et^k. Cada vez que se crea una nueva clase la función $new\_cluster(d_j)$ genera un nuevo tópico $c_k = (\{d_j\}, T^j, et^j)$. Cuando se añade un documento a un tópico c_k con la función $re\_compute(c_k, d_j)$, se modifica el tópico añadiendo las propiedades del documento d_j, así pues $c_k = (D^k \cup \{d_j\}, T^{c_k} + T^j, et^{c_k} \oplus et^j)$, siendo $\oplus$ la unión entre los dos intervalos.

5.5.3. Tópicos sin Tiempo

En [Yang98] define un algoritmo de detección de tópicos que obtiene buenos resultados al aplicar en el algoritmo de *Single-Pass*, de modo que dos documentos se agrupan si superan un umbral, y si las fechas de publicación de todos los documentos de ese grupo están dentro de una misma ventana temporal. En los estudios experimentales, se obtienen buenos resultados fijando una ventana temporal 150 días.

Como los documentos que vamos a analizar se han publicado durante un mes, podemos suponer que si en el algoritmo *Single-Pass* solo utilizamos la semejanza conceptual, o sea utilizamos el algoritmo que utiliza el *periodo de suceso* y hacemos $\beta_f = 0$ el algoritmo es similar al propuesto por [Yang98] utilizando una ventana temporal de un mes. La diferencia con su sistema estriba en que nosotros utilizamos la frecuencia de términos para asignar los pesos a los términos de los documentos, mientras en la referencia se utiliza un IDF (*Inverse Document Frequency*) incremental.

5.6. Evaluación de los sistemas propuestos

5.6.1. Metodología de Evaluación

La definición de tópico es un tanto ambigua, y a veces es difícil distinguir cuándo un conjunto de sucesos forman un único tópico o una secuencia de tópicos. Por ejemplo, en el caso de un accidente con el conductor a la fuga, se producen varios sucesos o sub-sucesos: el accidente, la detección del conductor y el juicio. Otro ejemplo podrían ser, los episodios de una guerra: comienzo conflicto, guerra, acuerdos de paz, etc. ¿Forman todos estos sucesos parte de la mismo tópico o son varios sucesos en cascada, es decir una secuencia de tópicos?. En nuestro sistema vamos a suponer que dos sucesos forman parte del mismo tópico si uno es la reacción del otro, y sus acciones se producen próximas en el tiempo.

Para la evaluación de las distintos sistemas propuestos a lo largo de este capítulo, se han clasificado manualmente todos los documentos que aparecen publicados durante un mes en una secuencia de tópicos, denotados c_i. El resultado de cada sistema desarrollado es una secuencia de grupos de documentos, denotados g_j. Gracias a que el objetivo que queremos alcanzar es similar al de los trabajos de TDT, evaluaremos nuestros sistemas en función de las medidas que se proponen en TDT (ver la sección 2.6.1). Estas se muestran a continuación, en función de los parámetros que conocemos en nuestros sistemas:

$$Cobertura = R(g_i, c_j) = \frac{n_{ij}}{n_i}$$
$$Precisión = P(g_i, c_j) = \frac{n_{ij}}{n_j}$$
$$F1(g_i, c_j) = 2 \cdot \frac{n_{ij}}{n_i + n_j}$$
$$Miss\ Rate = P_{Miss}(g_i, c_j) = \frac{n_i - n_{ij}}{n_i}$$
$$False\ Alarm\ Rate = P_{FA}(g_i, c_j) = \frac{n_j - n_{ij}}{N - n_j}$$
$$P_{rel}(g_i, c_j) = \frac{n_i}{N}$$
$$C_{Det}(g_i, c_j) = \frac{C_{FA} \cdot (n_j - n_{ij}) + C_{Miss} \cdot (n_i - n_{ij})}{N}$$

donde:

- N: Número documentos de la colección
- n_i: Cardinalidad del tópico c_i
- n_j: Cardinalidad del grupo g_j generado por el sistema
- n_{ij}: Número de documentos comunes entre grupo g_i y el tópico c_j

La medida F1 [Yan98b] asigna el mismo peso a la precisión y a la cobertura (*recall*). Sin embargo el valor del C_{Det} depende del valor del coste de las falsas alarmas. En nuestra evaluación decidimos tomar $C_{FA} = 0{,}1$ y $C_{Miss} = 1$, según se propone en el documento de evaluación de TDT-2000 [TDT00] para los sistemas de detección de tópicos.

Para evaluar el comportamiento global del sistema se pueden promediar las medidas parciales obtenidas para los mejores emparejamientos de tópicos y grupos. Generalmente para promediar las medidas se utilizan dos métodos la *MicroMedia* y la *MacroMedia*. Si se quiere promediar una medida de evaluación X, primero se busca el grupo g_j que más se parece a cada tópico c_i mediante la función:

$$\sigma(i) = argmax_j\{X(i,j)\},$$

así obtenemos una asociación entre cada tópico c_i y el grupo g_j con $j = \sigma(i)$ que más se parece a la clase según la medida de evaluación X. Para el cálculo de la *MicroMedia* primero se calculan los valores globales para n_i, n_j, y $n_{i,j}$, y sobre estos se calcula la medida X. Para el promedio utilizando la *MacroMedia*, se suman los valores X obtenidos con cada pareja $c_i - g_{\sigma(i)}$ y se asigna como medida global el valor resultante de promediar estos valores. O sea la *MicroMedia* se pondera a nivel de documentos y la *MacroMedia* al de tópicos.

El valor de promediar el C_{det} que se obtiene en ambas métodos es el mismo en nuestro sistema, al depender P_{rel} del número de documentos de cada clase, o sea

Aproximación	β_f	β_t	$WF1$	$MacroF1$	$MicroF1$
no time	0	0,3	0,54545	0,63979	0,61134
no time	0	0,37	0,57316	0,61172	0,63996
no time	0	0,35	0,56979	0,63512	0,6432
dates	0,16	0,28	0,68154	0,65878	0,71103
dates	0,15	0,29	0,68501	0,65842	0,71644
event time	0,36	0,28	0,67663	0,68922	0,71857
event time	0,36	0,27	0,68251	0,67495	0,72003
sentences	0,36	0,28	0,65331	0,65582	0,69259
sentences	0,36	0,26	0,6626	0,64916	0,69338

Cuadro 5.4: Resultados F1 con tópicos con más de un documento

las ecuaciones simplificadas de ambas medidas es el mismo. Adicionalmente, como el número de documentos de una clase varía mucho, hemos decidido considerar otra medida promedio para el F1, que pondera por el tamaño de cada tópico, éste se define en [Lar99] como:

$$F1_{WMacroAverage} = \frac{1}{N_{docs}} \sum_{i=1}^{N_{topics}} n_i F1(i, \sigma(i)). \tag{5.7}$$

5.6.2. Resultados de la evaluación

El corpus de evaluación contiene noticias publicadas en 'El País Digital' en junio de 1999. De la colección de estos periódicos se han seleccionado los 554 artículos correspondientes a la sección de noticias internacionales. El análisis de estas noticias nos ha permitido identificar manualmente 246 tópicos, 80 de los cuales no son unitarios, es decir contienen más de un artículo.

Para la evaluación de los sistemas hemos tenido que fijar los siguientes parámetros. Para el cálculo de la distancia de Minkowsky, hemos tomado el valor del parámetro $\rho = 0{,}2$, según se propone en [Pons01, Pon02]. En el sistema de detección con todas las referencias temporales, *Dates*, se han tomado $\omega_t = 0{,}5$, $\omega_f = 0{,}5$ según se propone en [Pons01, Pon02]. Además, para simplificar la cantidad de parámetros hemos decidido tomar $\beta_t = threshold$.

En la evaluación, hemos considerado importante ver como funcionaría el sistema que utiliza el *periodo de suceso* (*event time*), utilizando sólo los términos presentes en las oraciones que contienen alguna entidad temporal etiquetada, a este sistema lo referenciaremos como *sentences*.

Aproximación	β_f	β_t	$WF1$	$MacroF1$	$MicroF1$
no time	0	0,8	0,52752	0,86301	0,80942
no time	0	0,5	0,62227	0,84462	0,81747
no time	0	0,4	0,64478	0,79959	0,76138
all dates	0,14	0,3	0,72423	0,81112	0,79636
all dates	0,14	0,5	0,58725	0,86767	0,82378
all dates	0,14	0,4	0,67145	0,8511	0,83536
event time	0,36	0,4	0,71405	0,85266	0,84105
event time	0,36	0,37	0,72376	0,85033	0,83563
event time	0,36	0,7	0,52752	0,8633	0,80942
sentences	0,36	0,7	0,52519	0,86301	0,80855
sentences	0,36	0,37	0,69924	0,84946	0,8351
sentences	0,36	0,27	0,71842	0,82871	0,80012

Cuadro 5.5: Resultados F1 con todos los tópicos

En la tablas 5.4 y 5.5 se han presentado los resultados de las tres medidas globales de F1 para las herramientas de detección de tópicos utilizando los cuatro métodos propuestos. Se observa que tanto en la detección de todos los tópicos, como en la detección de tópicos no unitarios, los mejores valores se obtienen con los mismos valores umbrales de β_f. Además se observa que los sistemas que tienen en cuenta las propiedades temporales de los documentos obtienen resultados notablemente mejores, a los sistemas que no tienen en cuenta la temporalidad, *no time*.

Debido a que nuestro corpus contiene un conjunto de documentos grande con historias con un único documento, a partir de ahora mostraremos los resultados de evaluar los distintos sistemas utilizando solamente los tópicos no unitarios, tal y como se hace en las evaluaciones de los sistemas TDT.

Figura 5.8: Representación de los tópicos en el eje temporal

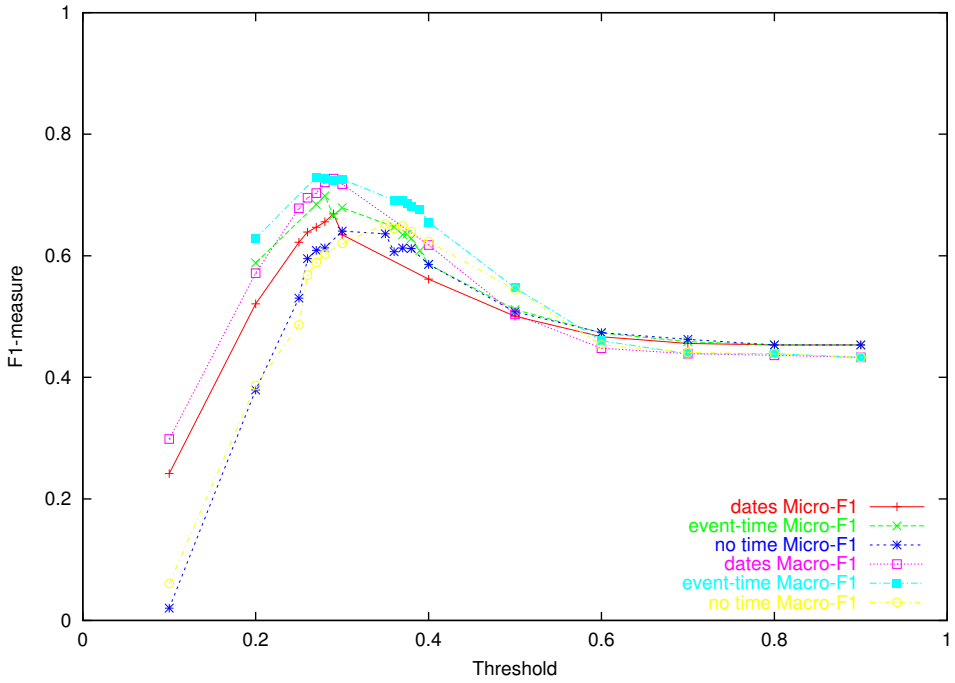


Figura 5.9: Medida F1 tomando los mejores valores β_f para cada sistema.

En la figura 5.9 se presenta la evolución del F1 con respecto umbral *threhold*, manteniendo el valor del umbral temporal β_f constante con su valor óptimo para los cuatro sistemas. En la gráfica se pueden ver que las curvas son similares para los sistemas *event-time* y *all-dates*.

En la figura 5.10 se compara el sistema *event-time* con el sistema *sentences*, que utiliza solo los términos presentes en las sentencias con referencias temporales con el cual se observa cierta mejora.

En la figura 5.12 (5.13) observamos cómo varia la medida del $microF1$ ($macroF1$) para el sistema que utiliza el event-time al variar tanto el *threshold* como el β_f. Además con una linea verde se representan los valores para el mejor valor de $\beta_f = 0{,}37$ y con linea azul para $\beta_f = 0$ que equivale al sistema *no time*, ya que no tienen en cuenta similitud temporal.

En la figura 5.14 mostramos la evolución del C_{Det} (Coste de Detección) al variar los parámetros de los sistemas, donde al igual como con la medida del F1, se observa un comportamiento similar para el *event time* y para *dates*, que tienen un coste ligeramente superior a la utilización de *no time*.

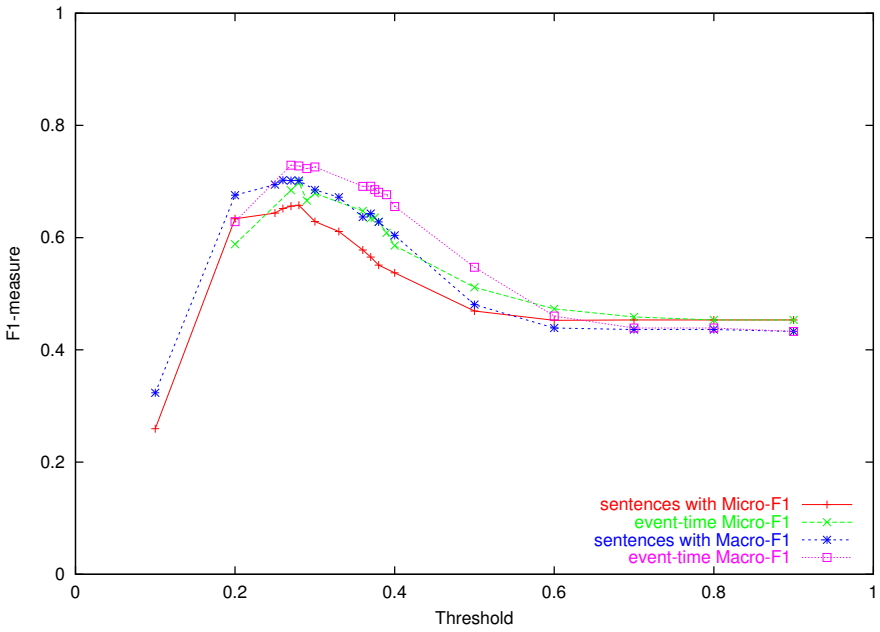


Figura 5.10: Comparación de la Medida F1 comparando el sistema de event-time utilizando todo el texto o solo las frases con sentencias temporales.

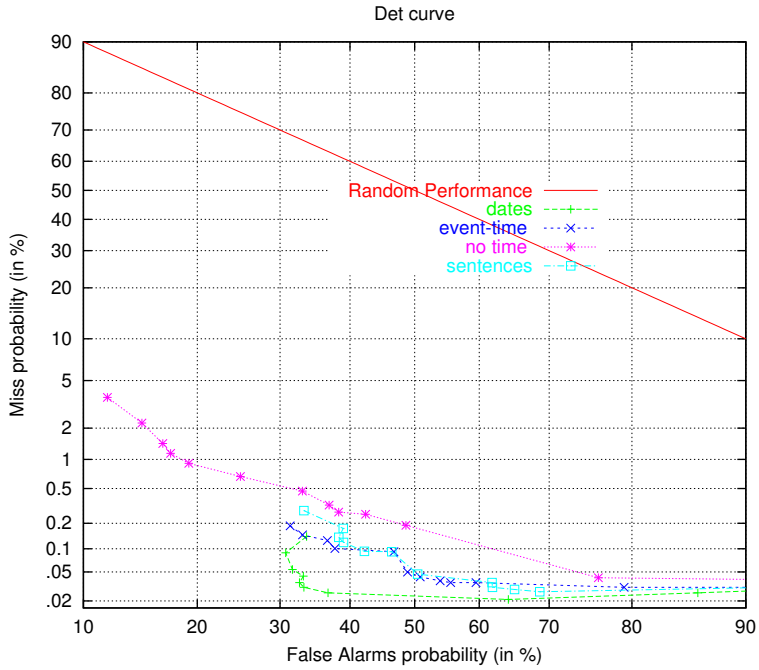


Figura 5.11: Curvas de detección.

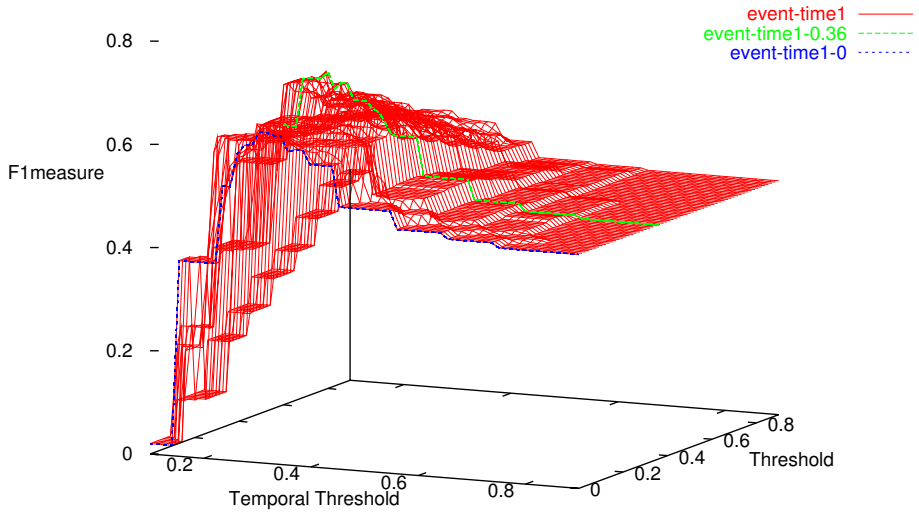


Figura 5.12: Evolución de MicroF1 variando β_f y β.

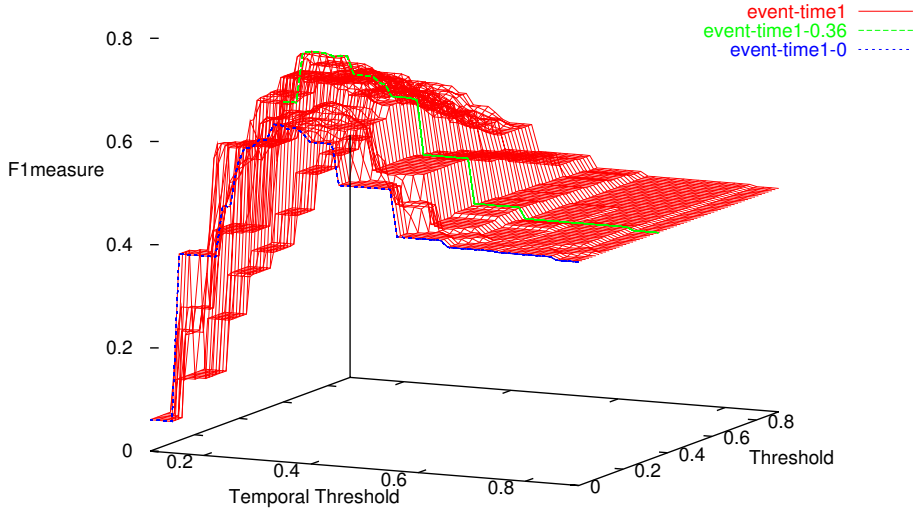


Figura 5.13: Evolución de MacroF1 variando β_f y β.

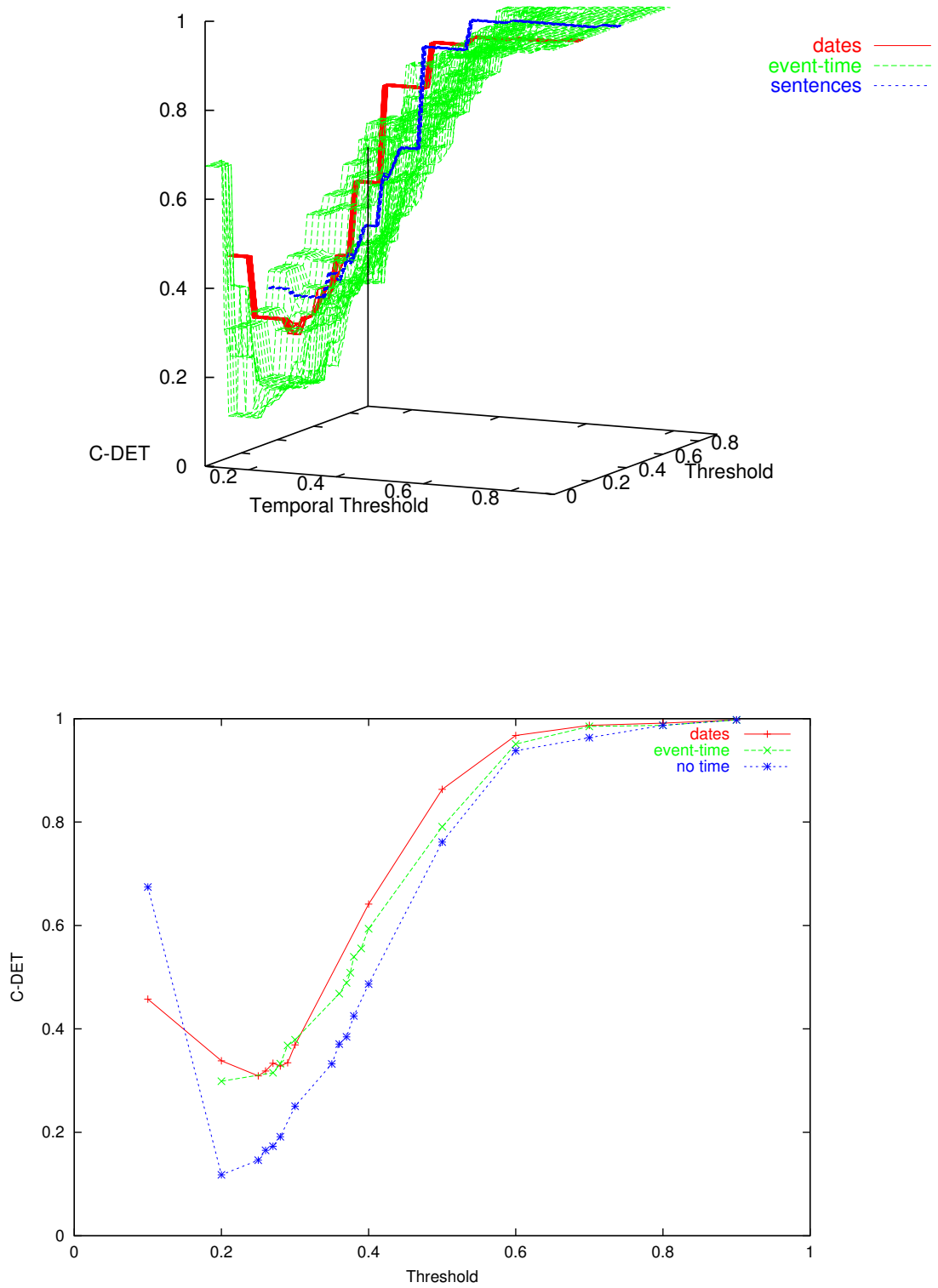


Figura 5.14: Coste de detección de tópicos.

Tema	Palabras clave
EU	(ue or europ %) and elecci %
Colombia	colomb %
IRA	ira or ulster or irlanda
Indonesia	indonesia
Kosovo	kosovo and (tropa % or ejercito %) and espa %
México	m %jic % or m %xic %
narco	narcotr %
Papa	papa
Pinochet	pinochet

Cuadro 5.6: Consultas IRE para creación de crónicas

Estos resultados indican que el uso del *periodo de suceso* es tan efectivo como el uso de todas las entidades temporales presentes en los documentos. Si a esto sumamos que el uso *periodo de suceso* requiere menor tiempo de ejecución, y que es más manipulable por los sistemas de RI actuales que el uso de un conjunto de fechas e intervalos, podemos concluir que *periodo de suceso* calculado automáticamente es una propiedad temporal de los documentos útil como metadato a utilizar en los sistemas de RI y los sistemas de TDT.

A continuación, vamos a comparar como funcionan los distintos sistemas de detección de crónicas propuestos en TOODOR con los sistemas de detección. Con este propósito hemos construido 7 consultas por palabras clave, de modo que cada una de ellas genera más de una crónica. Estas consultas se muestran en la tabla 5.6. En la tabla 5.7 se muestran los resultados de la evaluación de las crónicas obtenidas con una consulta TDRL sin especificar propiedades temporales (*TOODOR sin tiempo)*, con la consulta TDRL *event* (*TOODOR event*), con la herramienta TimExpIR utilizando la fecha de publicación (*TimExpIR pd*), y con el periodo de suceso (*TimExpIR et*), y además, con respecto al sistema de detección de tópicos utilizando el periodo de suceso, tomando para promediar los resultados del sistema de detección sólo los sucesos o tópicos encontrados con los sistemas anteriores (*TDT et*). En esta tabla se observa que el *periodo de suceso* con TimExpIR mejora los resultados de las crónicas TOODOR tipo *event*, pero aún así no se obtiene la efectividad de los resultados de los sistemas de detección. Ello es debido, como hemos señalado en otras secciones a la limitación de los sistemas de RI por palabras clave, que no permiten obtener documentos similares a los que se obtienen con las consultas, con lo que se obtiene menos cobertura con respecto a los sistemas de agrupamiento de documentos. En el Apéndice D se muestran algunas tablas con los resultados de obtener crónicas con estas consultas.

Por tanto, podemos resumir que el *periodo de suceso* calculado de forma automática, nos permite obtener la misma información que el uso de las fechas presentes en los documentos y además mejoran los resultados en los sistemas de TDT y RI.

Aproximación	w-F1	Macro-F1	MicroF1	Recall	Precission
TOODOR Sin Tiempo	0,29722	0,20237	0,24307	0,12433	0,54368
TOODOR event	0,39879	0,35369	0,34944	0,25588	0,57254
TimExpIR pd	0,43681	0,45763	0,45141	0,48875	0,43024
TimExpIR et	0,5102	0,41334	0,47806	0,32855	0,55713
TDT et	0,61032	0,64225	0,67563	0,82194	0,52703

Cuadro 5.7: Comparativa de herramientas

5.7. Conclusiones

A lo largo de este capítulo hemos tratado de explorar la utilidad de las referencias temporales presentes en los documentos con el propósito de proporcionar información añadida sobre la colección de documentos y mejorar los sistemas de RI y TDT.

En la última sección hemos comprobado cómo se cumple nuestra hipótesis inicial de que las entidades temporales presentes en los documentos eran importantes en la detección de sucesos, y hemos comprobado como la asignación automática de un *periodo de suceso* a partir de estas entidades temporales permite resultados similares a la utilización de todas las fechas y mejora la rapidez de ejecución de los sistemas. Además este metadato es más manipulable por los sistemas de procesamiento de documentos que el uso de a una secuencia de fechas e intervalos.

También hemos visto como se ayuda al usuario en la búsqueda de información en los sistemas de RI, si se les provee una interfaz gráfica que agrupe los documentos por sus periodos de suceso.

Aunque no hemos hecho mucho hincapié, hemos dejado abierta la puerta para la investigación de la utilizad de las expresiones temporales en la creación de resúmenes de documentos, ya que hemos comprobado que se obtienen resultados similares en el sistema de detección de tópicos por *periodo de suceso* al utilizar sólo los términos presentes en las sentencias que contenían expresiones temporales.

Como conclusión final de este capítulo, debido a la limitación de los SRI, proponemos una nueva aproximación para la ayuda a los usuarios a localizar sucesos, mediante un sistema de RI que primero preprocese los documentos y los agrupe por similitud semántica y temporal, y posteriormente realize las búsquedas sobre los tópicos devolviendo a los usuarios los documentos de los tópicos agrupados cronológicamente por su *periodo de suceso*.

Capítulo 6

Conclusiones

A lo largo de este trabajo hemos implementado y evaluado varias herramientas para comprobar la utilidad de las expresiones temporales en la búsqueda de información en documentos cuyos contenidos se localizan en el tiempo. Nos hemos centrado en la aplicación de estas herramientas a documentos periodísticos, pero queremos destacar que se pueden aplicar a otros documentos donde la información temporal contenida en expresiones temporales sea muy relevante, como pueden ser los informes médicos sobre los pacientes y su evolución, los documentos de declaraciones en los juicios, boletines oficiales, etc.

El trabajo de esta tesis se ha desarrollado principalmente con documentos escritos en español, pero hemos visto cómo podría trasladarse a otros idiomas. En particular hemos mostrado como el sistema de extracción de información temporal puede aplicarse a documentos escritos en inglés. A este respecto, queremos destacar la falta de recursos lingüísticos para nuestro idioma, sobretodo de libre acceso y queremos agradecer al grupo de Inteligencia Artificial de la Universidad Politécnica de Cataluña el habernos prestado sus herramientas MACO y TACAT.

A modo de resumen, vamos a destacar las aportaciones fundamentales de esta tesis:

1. Hemos desarrollando un modelo de tiempo que permite representar las expresiones temporales que indican el tiempo cronológico basándose en el calendario Gregoriano. Este modelo se compone de tres entidades temporales: puntos, intervalos y duraciones. Sobre estas entidades se ha definido un álgebra que nos permite operar con ellas y compararlas. Los puntos y los intervalos tratan de representar instantes fijos en el espacio temporal según el calendario Gregoriano, mientras que las duraciones son las entidades que

nos permiten desplazar un instante temporal hacia el futuro o pasado. En realidad este modelo es una modificación del modelo de tiempo general de Bettini [Bet00], el cual se ha desarrollado para definir cualquier espacio de tiempo lineal, cíclico o ramificado, a partir de las granularidades como unidad básica de medida. Nosotros hemos adaptado el modelo para un espacio de tiempo lineal discreto donde las granularidades deben cumplir las reglas de formación del calendario Gregoriano.

2. A partir del estudio de las expresiones temporales hemos creado la aplicaciónTimeExtractor, un sistema de extracción de información y detección de entidades temporales a partir de expresiones en Lenguaje Natural que tratan de representar el tiempo cronológico basado en el calendario Gregoriano. . Con el móduloTagtime se etiquetan las expresiones temporales con la notación semántica, *CodTemp*. Esta notación está basada en la notación utilizada en el modelo del tiempo. Posteriormente el modulo ModelTimex analiza cada expresión codificada para obtener el tipo de entidad temporal y detectar el instante temporal haciendo uso del álgebra definida en el modelo del tiempo. El módulo ModelTimex solo analiza las expresiones codificadas, siendo independiente del idioma, lo cual facilita que la aplicación TimeExtractor sea fácilmente trasladable a otros idiomas que utilicen el calendario Gregoriano realizando pequeñas modificaciones al móduloTagtime.

3. La herramienta TimeExtractor, mediante estudios estadísticos sobre de los instantes temporales que se han reconocido con el análisis de las expresiones temporales, y utilizando la proximidad de éstos con la fecha de publicación, es capaz de obtener automáticamente el *periodo de suceso* en el que transcurre la acción principal de cada documento.

4. Hemos estudiado en qué áreas sería útil la información temporal obtenida con TimeExtractor de cada documento (fechas, intervalos y *periodos de suceso*).

   - Hemos analizado cómo la información temporal mejora los sistemas de detección y seguimiento de tópicos (TDT). Estos sistemas ya utilizaban cierta información temporal de los documentos, haciendo la suposición de que la fecha de publicación coincide con la fecha en que se producen los sucesos. Nosotros hemos tratado de comparar un mismo sistema de detección de tópicos utilizando distintas propiedades temporales, las fechas e intervalos detectados en los documentos, y el *periodo de suceso*. En la evaluación de estos sistemas hemos visto que se mejora el sistema frente a la no utilización de propiedades temporales, y con resultados similares para la utilización de todas las fechas o del *periodo de suceso*. Con esto se demuestra que el periodo de suceso que calculamos automáticamente representa con bastante precisión el periodo en el que transcurre la acción principal del documento.

- Hemos mejorado el sistema de Recuperación de Información TOODOR. Por un lado hemos hecho que este sistema sea más operativo al poder asignar el *periodo de suceso* de cada documento de forma automática. Además, hemos mejorado el sistema de detección de *crónicas* que tenia definido este sistema, con la herramienta TimExpIR, que además provee una interfaz gráfica para mostrar los documentos recuperados por el sistema agrupados por sus periodos de suceso. Esta herramienta ayuda a los usuarios tanto a detectar distintos sucesos sobre un mismo tema, como a localizar con mayor facilidad el suceso sobre el cual está interesado, aún sin conocer con exactitud cuándo se ha producido. Por tanto con la herramienta TOODOR hemos comprobado como la información temporal presente en los documentos y representada con el *periodo de suceso* ayuda a los usuarios en la búsqueda de sucesos.

5. Por último proponemos la utilización de la clasificación de los documentos mediante un sistema de detección de sucesos, de modo que las búsquedas en un sistema de RI no se realicen directamente en los documentos, sino sobre los tópicos existentes en la colección, mostrando a los usuarios los documentos sobre el suceso especificado en la consulta organizados por tópicos. Esto permitirá a los usuarios explorar la información tanto por su contenido semántico como por sus relaciones temporales, obtener documentos que hablan sobre el mismo suceso, o bien obtener todas los sucesos que hablan sobre un mismo tópico.

Trabajo Futuro

Ahora que hemos llegado al final de nuestro trabajo de tesis, hemos comprobado que se han quedado muchas líneas de investigación abiertas:

- Gracias a TimeExtractor tenemos una herramienta que nos permite etiquetar las referencias temporales, pero nos falta abordar el problema de las referencias temporales relativas a otro sucesos.

- La herramienta de detección de sucesos a partir del *periodo de suceso*, se podría utilizar como base para la creación de una base de datos de sucesos o tópicos. Hemos hecho pequeños estudios de modo que cogiendo los términos más frecuentes que aparecen en las oraciones con expresiones temporales, junto con los nombres que están en mayúsculas permiten con pocas palabras representar los sucesos. Con este conjunto de palabras podríamos representar los sucesos y éste vendría totalmente localizado gracias al periodo de suceso. Otro mecanismo para poder detectar los conceptos representativos de un

suceso sería el uso de los sintagmas nominales que se utilizan en las referencias temporales a otros sucesos. La creación de una colección de sucesos, nos permitiría:

  - Mejorar la herramienta TimeExtractor para poder obtener las fechas relativas a otros sucesos.
  - Dar un contexto histórico. De modo que el usuario podría consultar qué ha acontecido en una determinada fecha.
  - Añadir enlaces en los documentos de modo que cuando se referencia un suceso con una expresión temporal relativa a otro suceso, se establezca un enlace a ese suceso. Las referencias temporales a fechas fuera del periodo de suceso, generalmente denotan un suceso relacionado. Posiblemente con la información de la frase que contiene esa referencia temporal se puede localizar ese suceso en la colección de sucesos y así podemos añadir un enlace.

- Deberíamos tratar de mejorar el sistema de detección de sucesos, aplicando sinónimos, utilizando un IDF (Inverse Document Frequency) incremental y no solo la frecuencia de aparición de los términos. También sería interesante aplicar otros algoritmos de clasificación no supervisada que no dependan del orden de procesado.
- Otra linea futura consistiría en crear un sistema de RI, que busque en los tópicos y devuelva los documentos agrupados por tópicos, según el orden de relevancia de cada tópico
- Y por último, se podría analizar las semejanzas entre el modelo temporal y un modelo espacial, ya que hemos detectado cierta similitud entre las expresiones de lugar y de tiempo en Lenguaje Natural.

c

Bibliografía

[All98] J. Allan, J. Carbonell, G. Doddington, J. Yamron and Y. Yang. " Topic Detection and Tracking Pilot Study Final Report". Proc. DARPA BroadCast News Transcription and Understanding Workshop, February, 1998.

[All98b] J. Allan, R. Papka, and V. Lavrenko. "On-line New Event Detection and Tracking". Proceedings of the 21st Annual International ACM SIGIR (SIGIR'98), Melbourne, 1998.

[All00] J. Allan, V. Lavrenko, D. Malin and R. Swan. "Detection, Bounds and Timelines: UMass and TDT-3". Proceeding of the TDT-3 Workshop, Virginia, 2000.

[App99] E. Appelt. "Introduction to Information Extraction". AI Communications, Vol. 12, pp. 161-1720, 1999.

[App99b] D. E. Appelt and D. J. Israel. "Introduction to Information Extraction Technology". Tutorial of IJCAI-1999, Stockholm, 1999.

[App93] D. Appelt, J. Hobbs, J. Bear, D. J. Israel and M. Tyson. "FASTUS: a finite-state processor for Information Extraction from real-world text". Proceedings of the 13th International Joint Conference on Artificial Intelligence, pp. 1172-1178, Chambery, 1993.

[Ara98] M.J. Aramburu and R. Berlanga. "A Retrieval Language for Historical Documents". 9th International Conference on Database and Expert System Applications, LNCS 1460, pp. 216-225, Springer Verlag, 1998.

[Ara98b] M.J. Aramburu and R. Berlanga "Temporal Object-Oriented Document Organisation and Retrieval". Integrated Design and Process Technology, Vol. 2: Issues and Applications of Database Technology, pp 368-375, Ed. Society for Design and Process Science, Berlin, July 1998.

[Ara98c] M.J. Aramburu. "TOODOR: A Temporal Object-Oriented Database Model for Historical Documents". PhD Thesis, The University of Birmingham, 1998.

[Ara99] M.J. Aramburu, R. Berlanga, D. Llidó y S. García. "Un modelo para la representación y recuperación de periódicos electrónicos". Novática, Vol. 142, pp. 20-24, Diciembre 1999.

[Bae99] R. Baeza-Yates and B. Ribeiro-Neto. "Modern Information Retrieval". Ed. ACM-Press, 1999.

[Bee00] D. Beeferman, A. Beger. "Agglomerative clustering of a search engine query log". Procedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining (SIGKDD'00). 2000.

[Ber99] R.Berlanga, M.J.Aramburu and S.García "Efficient Retreival of Structured Documents from Object-Relational Databases". Database and Expert System Applications, (DEXA'99), LNCS 1677, pp. 426-435, Springer Verlag, September 1999.

[Ber01] R. Berlanga, J.M. Pérez, M.J. Aramburu and D. M. Llidó. "Techniques and Tools for the Temporal Analysis of Retrieved Information". Database and Expert System Applications (DEXA'2001), LNCS 2113, pp. 72-81 Ed. Springer-Verlag, 2001.

[Bew91] R.C. Berwick, S.P. Abney and C. Tenny. "Parsing by Chuncks. Principle-Based Parsing". Computation and Psycolinguistics, pp. 257-278, Kluwer academics , 1991.

[Bet96] C. Bettini, X. Wang, and S. Jajodia. "A General Framework for Time Granularity and its Application to Temporal Reasoning". Proceedings of TIME-96, IEEE Computer Society Press, KeyWest, 1996.

[Bet98] C. Bettini, C. Dyreson, W. Evans, R. Snodgrass and X. Wang. "A glossary of time granularity concepts". Temporal Databases: Research and Practice, Vol. 1399 in LNCS State-of-the-Art Survey. Ed. O. Etzion, S. Jajodia and S. Sripada, 1998.

[Bet98b] C. Bettini, X. Sean Wang, and S. Jajodia. "Mining temporal relationships with multiple granularities in time sequences". Data Engineering Bulletin, Vol. 21,pp.32-38, 1998.

[Bet00] C. Bettini, S. Jajodia and X.S. Wang. "Time Granularities in Databases". Data Mining, and Temporal Reasoning. Springer-Verlag, July 2000.

[Bro94] J.Broglio, J.P. Callan, W.B.Croft. "INQUERY System Overview". Proceedings of the TIPSTER Text Program (Phase I). San Francisco, CA: Morgan Kaufmann, pp 47-67,1994.

[Buc00] C. Buckley, M. Mitra, J. Walz, and C. Cardie. "Using Clustering and SuperConcepts within SMART: TREC-6". Information Procesing Management Vol. 36, pp. 109-131, 2000.

[Bus00] S. Busemann, et al. "Natural Language Dialogue Service for Appointment Scheduling Agents". In Proceedings of the 15 International Conference of Applied Natural Language, pp. 25-32, Washingon, 1997.

[Car00] J. Carthy and A. F. Smeaton. "The Design of a topic Tracking System". In Proceedings of the 22nd Annual Colloquium on Information Retrieval Research, Cambridge, April 2000.

[Cas98] I. Castellón, M. Civit and J. Atserias. "Syntactic Parsing of Unrestricted Spanish Text". International Conference on Languages Resources and Evaluation, Granada, 1998.

[Che96] H. Chen, B. Schatz, T. Ng, J. Martinez, A. Kirchhoff, and C. Lin. "A Parallel Computing Approach to Creating Engineering Concept Spaces for Semantic Retrieval: The Illinois Digital Library Initiative Project". IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 18, pp. 771–782, 1996.

[Chi97] N. Chinchor. "MUC-7 Named Entity Task Definition". September 1997. `http://www.muc.saic.com/proceedings/ne_task.html`

[Chi99] N. Chinchor, E. Brown, F. Lisa and P. Robinson. "1999 Named Entity Recognition Task: Definition Version 1.4". `http://www.itl.nist.gov/iad/894.01/tests/ie-er/er_99/doc/ne99_taskdef_v1_4.ps`

[Chu80] K.W. Church. "On memory limitations in natural language processing". Masters Thesis, MIT. Distributed by the Indiana University Linguistics Club, 1980.

[Cut92] D. R. Cutting, J. O. Pedersen, D. Karger and J. W. Tukey. "Scatter/Gather: A Cluster-based Aproach to Browsing Large Document Collection". Proceedings of the 15th Annual International ACM SIGIR, Copenhagen, 1992.

[Dar97] H. Darwen. "Time Is On Our Side". Universidad Carlos III de Madrid. Julio, 1997.

[Dou95] D. E. Appelt and D. Martin. "Named Entity Extraction from Speech: Approach and Results Using the TextPro System". Proceedings of the DARPA Broadcast News Workshop, pp. 51- 54, Herndon, Virginia, February - March, 1999.

[Eje88] E.I. Ejerhed."Finding clauses in unrestricted text by finitary and stochastic methods". In Second Conference on Applied Natural Language Processing, Vol.5, pp. 219-227. ACL. 1988

[Esc99] G. Escdero, Ll. Màrquez and German Rigau. ´´An EmpiricalStudy of the Domain Dependence of Supervised Word Sense Disambiguation Systems". In Proceedings of SIGDAT Conference on Empirical Methods in Natural Language Procesing and Very Large Corpora, (CEMNLO/VLC'00), Hong Kong, 2000.

[Fer96] E. Ferrari and G. Guerrini. "A Formal Temporal Object Oriented Data Model". In Proceedings of EDBT, LNCS. 1057, pp. 342-356, SpingerVerlag, 1996.

[Fre01] D. Frey, R. Gupta, V. Khandelwal, V. Lavrenko, A. Leuski and J. Allan. "Monitoring the News: a TDT demonstration system. In the Proceedings of HLT 2001, San Diego, 2001.

[Gai00] R. Gaizauskas and K. Humphreys. "A Combined IR/NLP Approach to Question Answering Against Large Text Collections". In Proceedings of RIAO 2000: Content-Based Multimedia Information Access, pp. 1288-1304, Paris, April 2000.

[Gre00] E. Greengrass. "Information Retrieval: A Survey". Ed. Greengrass. 2000.

[Gri95] R. Grishman. "The NYU System for MUC-6 or Where's the Syntax?". In Proceedings of the MUC-6 Workshop, Washington, November, 1995.

[Gro98] D.A. Grossman and O. Frieder. "Information Retrieval: Algorithms and Heuristics". Kluwer Academic Publishers, 1998.

[Har00] D. Harman."What we have learned, from TREC". In Proceedings of BCS IRSG-2000, 22st Annual Colloquium on Information Retrieval Research, Sidney, 2000.

[Hat00] P. Hatch, N. Stokes, J. Carthy. "Topic Detection, a new application for lexical chaining". In Proceedings of BCS IRSG-2000, 22st Annual Colloquium on Information Retrieval Research, Sidney, 2000.

[Hir99] L. Hirschman, P. Robinson and L. Ferro. "Hub-4 Event guidelines. Version 2.6.1999".
`http://www.muc.saic.com/proceedings/hub4/guidelines.html`.

[Hob96] J.R. Hobbs, D. Appelt, J. Bear, D. Israel, M. Kameyama, M. Stickel, and M. Tyson. "FASTUS: Extracting information from natural-language texts". In Finite State Devices for Natural Language Processing. MIT Press, 1996.

[Hum99] K. Humphreys, R. Gaizauskas, M .Hepple and M. Sanderson. "TREC-8 Question and Answer System". The eighth Text REtrieval Conference (TREC-8), 1999.

[Ich94] M. Ichino and H. Yagushi. "Generalized Minkowski Metrics for Mixed Feature-Type Data Analysis". IEEE Transactions on Systems, Man and Cybernetics, Vol. 24, No. 4, 1994.

[Jac97] M. Jaczynski. "A Framework for the Management of Past Experiences with Time-Extended Situations". In Proceedings of the sixth International Conference on Information and Knowledge Management, (CIKM'97), Las Vegas, 1997.

[Jur00] D. Jurafsky and J.H. Martin. "SPEECH and LANGUAGE PROCESSING: An Introduction to Natural Language Processing". Computational Linguistics, and Speech Recognition. Prentice-Hall, 2000. `http://www-npl.cs.umass.edu/nlgroup/nlpie.html`

[Kni98] B. Knig , J. MA and E. Nissan. "Representing Temporal Knowledge in Legal Discourse". Information & Communications Technology Lay, Vol. 7 Issue 3, Oct 1998.

[Koe00] D.B. Koen and W. Bender. "Time frames: Temporal augmentation of the news". IBM Systems Journal, Vol. 39, 2000.

[Lar99] B. Larsen and C. Aone. "Fast and Efective Text Mining using Linear-Time Document Clustering". Procedings of the fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD'99), San Diego, 1999.

[Lee94] D.L. Lee and L. Ren. "Document Ranking on Weight-partitioned Signature Files". ACM Transactions on Information Systems Vol. 14, No.2, pp. 109-137, 1996.

[Leh96] W. Lehnert. "A performance Evaluation of text analysis technologies". AI Magazine, Vol. 12, No.3, pp. 81-94, 1996.

[Leh96] W. Lehnert. "Information Extraction". Communications of the ACM, Vol. 39, No.1, pp. 80-91, 1996.

[Lli99] D. M. Llidó, M. J. Aramburu, R. Berlanga y I. Sanz "Representación y organización de periódicos digitales con el lenguaje XML". IV Congreso ISKO-España EOCONSID99, 1999.

[Lli99b] D.Llidó, R.Berlanga y M.J.Aramburu. "Extracción y asignación de tiempos de suceso para documentos de actualidad". XVI Congreso de las Sociedad Española para el Procesamiento del Lenguaje Natural, pp. 223-230, Septiembre 2000.

[Lli01] D. M. Llidó, R. Berlanga, and M. J. Aramburu. "Extracting Temporal references to automatically assing document event-time periods". Database and Expert Systems Application(DEXA'2001), LNCS 2113, Springer-Verlag, 2001.

[Lut99] M. Lutz and D. Ascher. "Learning Python". Ed. Oreilly. 1999.

[Man99] C. D. Manning and H. Schütze. "Foundation of Statistical Natural Language Processing". MIT PRESS, 1999.

[Mar97] A. Martin, G. Doddington, T. Kamm, M. Ordowski and M. rzybocki. "The Det Curve in Assessment of Detection Task Performance". In Proceedings Eurospeech 97, pp. 1895-1898, Rhodos, 1997.

[Mof98] A. Moffat and K. Ramamohanarao. "Inverted Files Versus Signature Files for Text Indexing". ACM Transactions on Database Systems, Vol. 23, No. 4, December 1998.

[Ning01] P. Ning, X.S. Wang and S. Jajodia. "An Algebraic Representation of Calendars". Annals of Mathematics and Artificial Intelligence, Special Issue on Spatial and Temporal Granularity, Baltzer, 2001.

[Oha97] J. Wiebe, T. Ohara, K. McKeever and T. Ohrstrom-Sandgren. "An empirical approach to temporal reference resolution". In Proceedings of the Second Conference On Empirical Methods in Natural Language Processing (EMNLP2), August 1997.

[Pap99] R. Papka. "On-line new event detection, clustering and tracking". PhD Dissertation. Department of Computer Science, University of Massachusetts. 1999.

[Paz97] M. T. Pazienza (ed). "Information Extraction". SCIE-97. Springer, 1997.

[Paz99] M. T. Pazienza (ed). "Information Extraction". SCIE-99. Springer, 1997.

[Per00] J. Perez-Carballo and T. Strzalkowki. "Natural Language Information Retrieval: Progress Report". Information Processing and Management Vol. 36, pp. 155-178, 2000.

[Pons01] A.Pons y R.Berlanga. "Métodos y algoritmos para la agrupación automática de documentos". Informe técnico 01/06/2001, Departamento de Lenguajes y Sistemas Informáticos, Universitat Jaume I, 2001.

[Pon02] A. Pons, R. Berlanga y J.Ruiz-Shculcloper. "Temporal Semantic Clustering of News for Event Detection and Tracking ". In Proceedings of the second Workshop on Pattern Recognition for Information Systems (PRIS2002), ICEIS-Press, 2002.

[Rij79] C.J. Rijsbergen. "Information Retrieval". London, Butterworths, 1979. `http://www.dcs.glasgow.ac.uk/keith`

[Ril94] E. M. Riloff. "Information extraction as basis for portable text classification system". PhD Dissertation Report, University of Massachusetts Amherst. September 1994.

[Rod99] H. Rodríquez. "Extracción y Recuperación de Información".Tutorial SEPLEN, 1999.

[San98] I. Sanz, R. Berlanga and M. J. Aramburu. "Gathering Metadata from Web-based Repositories of Historical Publications" 9th International Workshop DEXA98, pp. 473-478, IEEE Computer Soc. Press, 1998.

[Sal98] G. Salton. "Automatic Text Procesing". Addison-Wesley, 1998.

[Sic97] SICStus Prolog User's Manual by Swedish Institute of Computer Science sicstus-request@sics.se, SICStus Prolog 3 #6, November 1997.

[Sme98] A.F. Smeaton. "Information Retrieval and Natural Lenguage Orocessing". In Proceeding of the Conference Prospects for Intelligent Retrieval. Informatic Vol. 10, 21-23, Marzo 1998

[Sno95] R.T. Snodgrass,I. Ahn, G. Ariav, D.S. Batory, H. Clifford, E.E. Dyreson, R. Elmasri, F. Grandi, et al. "The TSQL2 Temporal Query language". Kluwer Academic Publishers, 1995.

[Ste98] M. Stede, S. Haas and U. Küssner. "Understanding and tracking temporal descriptions in dialogue". In B. Schröder, W. Lenders, W. Hess, T. Portele eds. Computers, Linguistics, and Phonetics between Language and Speech. In Proceedings of the 4th Conference on Natural Language Processing (KONVENS'98), Frankfurt, 1998.

[Ste97] A. Steiner and C. Norrie. "Temporal Object Role Modeling". CAISE97, 1997.

[Spc98] S. Spaccapietra, C. Parent, E. Zimanyi. "Modeling Time from Conceptual Perspective". In Proceedings of the seventh International Conference on Information and Knowledge Management (CIKM'98), Betesda, 1998.

[Spr00] K. Sparck Jones. "Further reflecton on TREC". Information Processing and Management, Vol. 36, pp. 37-85, 2000.

[Swa99] R. Swan, and J. Allan. "Extracting Significant Time Varying Features from Text". In Proceedings of the eighth International Conference on Information and Knowledge Management (CIKM'99), pp. 38-45, Kansas City, 1999.

[Swa00] R. Swan and J. Allan. "Automatic Generation of Overview Timeline". Technical Report IR.198, University of Massachusetts, CIIR, 2000.

[Swan00b] R. Swan, D. Jensen. "TimeMines: Constructing Timelines with Statistical Models of Word Usage". KDD-2000 Workshop on Text Mining, 2000.

[TIP] Proyecto TIPSTER.
"http://www.nist.gov/itl/div894/894.02/related_projects/tipster"

[TDT] TDT Home page.
httpd://www.nist.gov/speech/tests/tdt/index.html

[TDT98] National Institute of Standards and Technology. "The Topic Detection and Tracking evaluation phase 2 (TDT2)", 1998.
http//morph.ldc.upenn.edu/TDT/

[TDT00] National Institute of Standards and Technology. "The Year 2000 Topic Detection and Tracking task definition and evaluation plan (TDT2000)", 2000.
http//morph.ldc.upenn.edu/TDT/

[Tom94] A. Tomasic, H. Garcia Molina and k. Shoens. "Incremental Updates Of Inverted Lists For Text Document Retrieval". ACM Sigmod Vol. 5, 1994.

[Tra95] J. Tramullas Saz. "Introducción a la Informática Documental". Apuntes Ciencias de la Computación Universidad de Zaragoza, 8, Zaragoza pp. 6-10, 1995.
Http://jabato.unizar.es/elec_doc/info_doc.html

[Tur99] J. Turmo, N. Catalá and H. Rodriguez. "An Adaptable IE System to New Domains". Applied Intelligence, Vol. 0 (2-3), pp.255-246, 1999.

[Yang98] Y. Yang, T. Pierce and J. Carbonell. "A study on retrospective and Online Event Detection". In Proceedings of the 21st Annual International ACM SIGIR (SIGIR'98), Melbourne, 1998.

[Yan98b] Y. Yang, J. Carbonewll, R. Brown, T. Pierce, B.T. Archibald and X. Liu. "Learning approaches for Detecting and Tracking News Events". Proceedings of the 21st Annual International ACM SIGIR (SIGIR'98), Melbourne, 1998.

[Yan00] Y. Yang ,T. Ault, T. Pierce, and C.W. Lattimer. "Improving text categorization methods for event tracking". Proceedings of the 23st Annual International ACM SIGIR (SIGIR'2000), Athens, 2000.

[Yan01] Y. Yang. "Document Clustering". IR handout. 2001.

[Veg99] J. Vegas Hernandez. "Un sistema de recuperación de información sobre estructura y contenido". Tesis doctoral Departamento de Informática de la Universidad de Valladolid, 1999.

Apéndices

A. Colección de Periódicos

Clase	Cod_arti	Titular
0	i0	Milosevic dice aceptar las condiciones del G-8
31	i1	Solana se niega a recibir al líder de la guerrilla kosovar en su visita 'privada' a la OTAN
32	i2	Chirac y Schröder dicen que el procesamiento de Milosevic justifica la intervención militar
33	i3	La OTAN envía al Adriático una flota contra minas para 'pescar' sus bombas
0	i4	Calendario del conflicto Día
49	i5	Fidel Castro destituye a Robaina en Exteriores, en un clima de vuelta a la línea dura y ortodoxa
49	i6	Un colaborador a la sombra del 'comandante'
50	i7	Yeltsin desata una nueva crisis al destituir al 'zar económico' a los tres días de nombrarle
50	i8	La estrategia del cambio de sillas
51	i9	Los primeros ministros de India y Pakistán hablan por teléfono para rebajar la tensión
52	i10	Los militares colombianos exigen a Pastrana que endurezca el diálogo de paz
102	i11	Dimite el ministro de Defensa portugués por filtrar un informe sobre el espionaje
61	i12	Solana se perfila como favorito en la UE para el puesto de 'míster PESC'
62	i13	Ankara impide a cuatro letrados españoles asistir al juicio de Ocalan
63	i14	El IRA empieza a entregar cuerpos de desaparecidos en el conflicto irlandés
103	i15	Centroamérica recibirá 9.000 millones de dólares para superar los estragos del 'Mitch'
66	i16	El Gobierno chileno dice que Pinochet debe volver porque hay garantías de que será juzgado
76	i17	América Latina busca en la UE la salida a las crisis financieras cíclicas
92	i18	El Congreso de Venezuela sufre una 'fuga' masiva de parlamentarios
0	i19	Los enviados a Belgrado debaten hasta última hora el plan de paz que presentarán a Milosevic
0	i20	El Ejército yugoslavo advierte de que no aceptará tropas de la OTAN dentro de Kosovo
34	i21	Muere un general serbio en un bombardeo
35	i22	La ONU se gasta en atender a los refugiados 1.600 millones de pesetas semanales
0	i23	La OTAN hace una evaluación triunfal de los 70 días de bombardeos
28	i24	Clinton dice que Europa asumirá el peso de la intervención y la reconstrucción de Kosovo
28	i25	Santer duda de que la UE tenga dinero suficiente para los Balcanes
0	i26	Calendario de conflicto Día
62	i27	Ocalan implica a su ex mujer en el asesinato de Olof Palme
62	i28	Restricciones a la prensa
69	i29	'Fui allendista a mucha honra, pero mi mundo es distinto al de Allende'
105	i30	La Democracia Cristiana de Chile examina su revés electoral
53	i31	Pastrana dirige la búsqueda de los secuestrados por el
51	i32	India intensifica su campaña militar en Cachemira sin renunciar a la vía diplomática
72	i33	Mandela pide a Suráfrica votar hoy sin violencia en las elecciones de su despedida
72	i34	La promesa de mayor disciplina fiscal atrae al mundo del dinero
72	i35	El último gesto de
50	i36	Un nuevo 'Rasputín' en el Kremlin
74	i37	Israel dice a sus aliados del sur de Líbano que no les abandonará tras la retirada de sus tropas
106	i38	Un reformista aliado de Jatamí, elegido nuevo alcalde de Teherán
76	i39	Francia trata de retrasar la negociación de un pacto comercial de la UE con América Latina
107	i40	Cuba reclama a Washington 181.000 millones de dólares por su 'agresión' desde 1959
108	i41	El Salvador recibe con optimismo la llegada al poder del conservador Francisco Flores
109	i42	Un juez argentino acusa a Massera de robar tierras a desaparecidos
78	i43	La oposición pide al presidente mexicano que diga si hubo dinero negro en su campaña
110	i44	El asesinato de dos jueces siembra el miedo en el Caribe nicaragüense
0	i45	El Parlamento yugoslavo debate el plan de paz
1	i46	Clinton deja abiertas todas las opciones

Clase	Cod_arti	Titular
1	i47	La OTAN anuncia que la fuerza de paz está lista para actuar 'en unos días
30	i48	El Tribunal Penal Internacional de La Haya rechaza la demanda serbia
28	i49	Alemania intentará lanzar el Pacto de Estabilidad para Europa del Sur
0	i50	Calendario del conflicto Día
72	i51	Thabo Mbeki, nuevo presidente electo de Suráfrica
72	i52	El regreso de Winnie
72	i53	Sharpeville, entre el recuerdo y la esperanza
61	i54	Los Quince abordan en Colonia una política de defensa común
61	i55	Solana y Solbes, candidatos a dirigir la política exterior y el BEI
79	i56	La generación de Tiananmen, diez años después
79	i57	Tian Ge sigue buscando
79	i58	'La gente sólo piensa ahora en hacer negocios'
51	i59	India descarta el uso de armas nucleares en el conflicto de Cachemira
62	i60	El PKK apoya la llamada de Ocalan a abandonar la lucha armada en Turquía
53	i61	Pastrana decide romper el diálogo con los guerrilleros del ELN tras el secuestro de Cali
54	i62	La violencia se extiende a varias regiones del país
111	i63	Al menos nueve muertos en un accidente aéreo en Arkansas
112	i64	El féretro fue arrojado al Atlántico
113	i65	Los choques tribales en la región petrolífera de Nigeria causan más de 100 muertos
78	i66	El espionaje de EE UU vincula a dos altos políticos de México con el narcotráfico
78	i67	La encarnación del viejo sistema del
78	i68	Un ataque directo al círculo íntimo del presidente
49	i69	El nuevo canciller cubano niega ser un 'ortodoxo' del régimen de Castro
2	i70	Milosevic entrega Kosovo al control de la OTAN
2	i71	Los aliados advierten de que sólo pararán los ataques cuando verifiquen la retirada
2	i72	La OTAN consigue imponer sus principales objetivos
2	i73	El plan para la paz en Yugoslavia y el regreso de los refugiados
2	i74	La fuerza de paz internacional puede desplegarse en 48 horas
2	i75	La Alianza estima en 5.000 los muertos en el Ejército yugoslavo
2	i76	Cronología de 72 días de ataques
2	i77	El líder serbio tendrá que responder ahora ante un país derrotado y destruido
2	i78	Chernomirdin asegura que el fin de los ataques es 'cuestión de días
2	i79	Los refugiados desconfían de las buenas palabras de Belgrado
2	i80	Aznar dice que espera ver un Kosovo 'liberado de la represión'
2	i81	Kofi Annan cree que es prematuro 'ponerse a dar saltos de alegría'
37	i82	'No hemos cometido errores'
3	i83	Calendario del conflicto Día
36	i84	¿Hacia qué nuevo orden mundial?
61	i85	Los líderes europeos nombran a Solana 'míster PESC', el 'ministro' de Exteriores de la
61	i86	El órgano de defensa común de la UE se creará en el 2000
900	i87	Prodi se arroga el poder de destituir a cualquiera de sus comisarios europeos
72	i88	Mbeki roza la mayoría aplastante que necesita para reformar la Constitución de Suráfrica
72	i89	Asignatura aprobada
72	i90	Ambiente de fiesta en las urnas y escasos episodios de violencia
79	i91	China rescata el mito del peligro exterior en el aniversario de Tiananmen
115	i92	Yeltsin conmuta todas las penas capitales y vacía los 'corredores de la muerte' en Rusia
66	i93	Londres decide hoy la fecha del proceso de extradición de Pinochet
116	i94	Dólares y negocios raros en Panamá
78	i95	México exige a EE UU pruebas de que hay políticos del PRI vinculados al narcotráfico
92	i96	La falta de dinero amenaza los próximos comicios en Venezuela
53	i97	Las FARC ayudaron en el secuestro de la iglesia de Cali
86	i98	Activistas civiles piden a la OEA el fin de la impunidad en Guatemala
117	i99	La oposición ve factible unirse para las presidenciales mexicanas del 2000
3	i100	Los generales yugoslavos se resisten a firmar el plan para la retirada de sus tropas
3	i101	Un observador ruso se incorpora a la reunión
3	i102	La Alianza Atlántica aprueba hoy el plan para el despliegue militar en Kosovo
21	i103	EE UU teme que se produzca un éxodo de población serbia
3	i104	Milosevic guarda silencio mientras Belgrado regresa lentamente a la normalidad
3	i105	Los países del G-8 redactan hoy la fórmula del plan de paz que debe aprobar la ONU
902	i106	El presidente de Finlandia aplaza su viaje a China
18	i107	Comienza la campaña de información a los refugiados para su vuelta
38	i108	Violenta pelea entre deportados instalados en Macedonia
3	i109	Calendario del conflicto Día
80	i110	El brazo armado del FIS argelino anuncia el cese definitivo de sus acciones
81	i111	Los socialdemócratas alemanes triunfan en las elecciones de Bremen
82	i112	Los indonesios acuden hoy a las urnas en busca de una ruptura radical con el pasado
82	i113	'Gane quien gane, casi no tendrá periodo de gracia'
82	i114	Yusuf Habibie: 'Dirán que fui yo quien trajo la democracia'
82	i115	Megawati Sukarnoputri: La gran esperanza de la oposición
82	i116	Amien Rais: Un tifón político al frente de los estudiantes
82	i117	Adburrahman Wahid: Un buen aspirante con mala salud
2001	i118	'Los micronacionalismos amenazan a Europa'
2002	i119	Un político inclasificable con patente de corso de Jospin
83	i120	El Papa redobla los llamamientos a la ortodoxia en la Polonia profunda
53	i121	En el país de la guerrilla
55	i122	El presidente de Colombia acepta desmilitarizar una zona para que el ELN libere a los rehenes
120	i123	Venezuela repatria a más de 3.000 colombianos que huían de los paramilitares
86	i124	Los países de América discuten cómo hacer frente a las crisis financieras de la región
2003	i125	El presidente de México niega haber pagado su campaña con dinero de un polémico banquero
2004	i126	El presidente de Brasil toma medidas para asegurar el futuro de la Amazonia
74	i127	Líbano recupera Yesín 15 años después
51	i128	India prosigue los bombardeos contra posiciones rebeldes en Cachemira
3	i129	Las objeciones de Rusia dificultan el fin de la guerra
3	i130	Siete discrepancias iniciales
3	i131	Clinton y Yeltsin discuten fórmulas para acelerar la negociación
3	i132	La OTAN recrudece los bombardeos a la espera de una solución diplomática definitiva
3	i133	Belgrado intenta no dar la impresión de que ha capitulado

Clase	Cod_arti	Titular
3	i134	Silencio en Kumanovo tras el parón en las conversaciones
3	i135	Los puntos de la discordia
3	i136	La Casa Blanca no confía en la palabra de Milosevic
39	i137	Aznar pide ante Rugova que no se creen falsas ilusiones sobre la independencia de Kosovo
18	i138	La ONU ve imposible el regreso de los refugiados antes del otoño
40	i139	Los aliados ya han gastado 1,13 billones de pesetas en la guerra
41	i140	Fuerzas británicas detienen en Bosnia a un serbio por crímenes de guerra
3	i141	Calendario del conflicto Día
82	i142	La oposición laica se atribuye la victoria en las elecciones en Indonesia
82	i143	La incógnita del Ejército
82	i144	Sólo un primer paso hacia la normalización
80	i145	Ruido de paz en Argelia
83	i146	El Papa rinde homenaje a los mártires de la reciente historia polaca
122	i147	La verdad y la ciencia
72	i148	Mbeki se queda a un solo escaño de los dos tercios del Parlamento surafricano
123	i149	Secuestrado otro español en una 'pesca milagrosa' en Colombia
85	i150	Conmoción en México por el asesinato a tiros de un famoso presentador
124	i151	El riesgo del periodista
125	i152	350 presos se fugan por la puerta principal de una cárcel en Brasil
126	i153	Derrotado en las urnas el hijo del general Bussi, acusado de genocidio durante la dictadura
68	i154	El presidente de Argentina destituye a un alto militar acusado de violar derechos humanos
127	i155	Fujimori no acata una sentencia que ordena a Perú repetir un juicio por falta de garantías
86	i156	Las propuestas de EE UU levantan la polémica en la reunión de la OEA
4	i157	La ONU inicia la cuenta atrás hacia la paz
4	i158	La secuencia de la pacificación
4	i159	La OTAN cree cumplidos sus objetivos y reanuda el diálogo con el Ejército serbio
4	i160	La sensación de alivio se impone en Yugoslavia, aunque todavía caen las bombas
902	i161	Los aliados multiplican las atenciones con China para evitar un veto
4	i162	Proyecto de resolución de la ONU
4	i163	Calendario del conflicto / Día
68	i164	Un juez chileno procesa a cinco jefes militares en la mayor causa abierta contra la dictadura
94	i165	Cuatro magistrados son asesinados en Líbano en plena sesión de un juicio
901	i166	Schröder y Blair presentan un manifiesto para la modernización de la izquierda
82	i167	Los primeros resultados anuncian una holgada victoria opositora en Indonesia
129	i168	Ahora, Timor Oriental
95	i169	Amnistía Internacional se alza en EE UU contra el castigo eléctrico a presos
95	i170	Los fabricantes del 'cinturón aturdidor' dicen que es 'más humano' que las porras
130	i171	Condenado uno de los policías que violó a un inmigrante en Nueva York
62	i172	La acusación confirma la petición de pena de muerte para el líder kurdo Ocalan
83	i173	El Papa critica los desequilibrios sociales en Polonia
76	i174	El portazo de la UE a Mercosur amenaza el futuro de las relaciones con América Latina
56	i175	El Parlamento de Colombia deniega a Pastrana poderes especiales
85	i176	El espionaje mexicano afirma que el cómico asesinado tenía vínculos con el narcotráfico
131	i177	Las televisiones espolean la indignación contra Cárdenas
1015	i178	Alemán y Ortega negocian una reforma electoral en Nicaragua que les beneficia
1015	i179	Doña Violeta: 'Que Dios nos proteja'
86	i180	La OEA fuerza a EE UU a retirar una propuesta por 'intervencionista'
4	i181	Los serbios tienen 11 días para dejar Kosovo
8	i182	Solana anuncia el cese de los bombardeos en cuanto se verifique el inicio de la retirada
4	i183	El Consejo de Seguridad de la ONU sellará hoy el alto el fuego
4	i184	Clinton se alegra del avance hacia los 'objetivos' aliados en Kosovo
9	i185	Las tropas británicas serán las primeras en intervenir con la misión de ocupar Pristina
8	i186	Repliegue en cascada
2005	i187	España prepara el envío de hasta 1.200 legionarios
2006	i188	La Iglesia, el Ejército y los políticos piden a los serbios de Kosovo que no huyan de la provincia
21	i189	'Si se marchan los militares, el ELK se vengará'
4	i190	Calendario del conflicto Día
68	i191	Un juez deja en manos del Ejército el procesamiento de cinco oficiales chilenos
88	i192	Chile usa a España contra un español cuyos bienes expropió Pinochet
88	i193	El Gobierno de Aznar justifica su actuación en la defensa de las inversiones
66	i194	Repercusión política del caso
66	i195	El Supremo dice que la actitud de Fungairiño en el 'caso Pinochet' puede ser 'criticable'
21	i196	La Iglesia, el Ejército y los políticos piden a los serbios de Kosovo que no huyan de la provincia
2007	i197	'Si se marchan los militares, el ELK se vengará'
133	i198	Mueren en Bagdad seis militantes de la oposición iraní por un coche bomba
94	i199	La policía libanesa implica a radicales palestinos en la matanza de jueces
84	i200	El escaso interés anuncia unas elecciones europeas marcadas por la abstención
134	i201	Italia condena a cadena perpetua a un capitán nazi que fusiló a 15 partisanos
901	i202	Jospin se declara menos atlantista y librecambista que Blair y Schröder
64	i203	Londres y Dublín quieren una cumbre final para impulsar la paz en el Ulster
85	i204	La muerte de Stanley crispa el ambiente político en México
2008	i205	'Se creó un ambiente de histeria colectiva'
60	i206	El jefe máximo del ELN acude al Vaticano a pedir perdón por el secuestro de Cali
86	i207	La OEA reelige al colombiano Gaviria como secretario general
137	i208	Suiza extraditará al ex director de Aeroméxico, acusado de fraude
4	i209	La OTAN pone fin a los bombardeos
8	i210	Retirada por escalas y franja de seguridad
4	i211	La ONU asume la administración de Kosovo
4	i212	Milosevic se proclama victorioso por haber preservado la integridad territorial yugoslava
42	i213	¿Pero hubo alguna vez una guerra?
9	i214	Las tropas aliadas prevén entrar hoy o mañana en Kosovo
16	i215	Estados Unidos se resiste a dar a Rusia una zona de mando propia
28	i216	El G-8 aprueba el plan de reconstrucción para los Balcanes
6	i217	La 'guerra limpia' de la OTAN tuvo múltiples errores, pero permitió una victoria aplastante
6	i218	El arma de la cohesión interna

Clase	Cod_arti	Titular
43	i219	La crisis de los Balcanes costará 50.000 millones a España este año
44	i220	Aznar dice que el próximo paso debe ser colaborar con el Tribunal Internacional
19	i221	Los refugiados carecen de papeles para regresar y recuperar sus casas
20	i222	Ahtisaari cree que volverán con la Kfor
30	i223	Los investigadores del Tribunal de La Haya se preparan para ir a Kosovo
6	i224	78 días de guerra y más de 5.000 muertos en las filas yugoslavas
6	i225	Calendarido del conflicto Día
66	i226	Pinochet se personará en la causa que instruye Garzón por genocidio, terrorismo y tortura
88	i227	El Congreso y el 'caso Pey'
66	i228	'Embustes de los señores de España'
138	i229	Cuba destituye a varios altos cargos vinculados a empresas que promocionan el turismo sexual
2009	i230	El área del dólar, en la mirilla
83	i231	La visita del Papa a Polonia abre dudas sobre sus planes tras el 2000
140	i232	Numerosos eurodiputados aprovechan viajes y dietas para redondear ingresos
84	i233	Británicos, daneses y holandeses iniciaron ayer las elecciones europeas
99	i234	América Central espera este año nueve huracanes similares a 'Mitch'
2010	i235	Policías en apuros
11	i236	Tres serbios armados mueren en choques con las tropas aliadas en Prizren y Pristina
11	i237	'Es rabia, es odio'
22	i238	Desbandada serbia en Kosovo
8	i239	La OTAN confirma la retirada de 11.000 soldados serbios
16	i240	Rusia se compromete a no enviar más tropas a Kosovo mientras no haya acuerdo con la OTAN
7	i241	El Pentágono pide a los serbios que se rebelen contra Milosevic
10	i242	Tanques cubiertos de flores
9	i243	Casas quemadas, animales muertos
9	i244	'Nos hemos visto sobrepasados, desbordados'
9	i245	Entra en Kosovo el primer convoy de la ONU con ayuda humanitaria
45	i246	Un anónimo 'Schindler' en Pristina
97	i247	La crisis de la dioxina le cuesta la mayoría al Gobierno de Bélgica
84	i248	Los populares de la UE ganan los comicios por primera vez en 20 años
84	i249	Schröder, dispuesto a utilizar la derrota para impulsar las reformas económicas en Alemania
84	i250	Los laboristas de Blair podrían perder la mitad de sus diputados europeos
82	i251	El partido de Suharto reconoce su derrota en las elecciones de Indonesia
87	i252	La vuelta de Salinas solivianta a México
60	i253	La guerrilla colombiana aplaza la liberación de los secuestrados de Cali
143	i254	34 niños marcados por el general Videla
144	i255	Las mafias salvadoreñas utilizan a los antiguos 'escuadrones de la muerte'
87	i256	El ex gobernador mexicano prófugo acusa a Zedillo de 'revanchismo'
12	i257	Los soldados de la Kfor descubren al sur de Kosovo tres fosas con cerca de 200 cadáveres
16	i258	El mando de la OTAN deja a los soldados rusos el control del aeropuerto de Pristina
16	i259	Clinton y Yeltsin acercan posiciones
7	i260	Los radicales ultranacionalistas abandonan el Gobierno serbio
12	i261	La llegada de las tropas aliadas saca de su escondite a miles de albanokosovares
22	i262	Caravanas de serbios abandonan Prizren entre insultos y pedradas
22	i263	Pánico y brotes de venganza
1018	i264	Un oficial serbio denuncia que en la provincia sólo queda una étnia
13	i265	Una avanzadilla de 50 militares españoles viaja esta semana a la zona
73	i266	La seducción del hombre blanco
97	i267	El primer ministro belga admite la derrota y presenta la dimisión
84	i268	La abstención y el voto de castigo favorecen la mayoría conservadora en la eurocámara
84	i269	Equilibrio dominante
84	i270	Los conservadores intensifican su campaña contra el euro tras la derrota de Blair
84	i271	Schröder ofrece un plan de reformas fiscales como respuesta a su sonada derrota electoral
149	i272	La 'izquierda plural' francesa sale reforzada en las urnas
84	i273	Aumenta la fragmentación del mapa político italiano
84	i274	Los socialistas portugueses arrasan con Soares, que se acerca a la mayoría absoluta
84	i275	La derecha gana en Grecia, pero con el mismo número de escaños que el PASOK
83	i276	Juan Pablo II alarga su viaje con una escala en Armenia
87	i277	Salinas abandona México tras una polémica visita de dos días para defender sus logros
57	i278	Los paramilitares colombianos matan a cinco civiles
60	i279	La guerrilla colombiana exige que Alemania medie en el secuestro de Cali
153	i280	Un sacerdote acusa al jefe de la policía de Brasil de torturador
93	i281	El presidente venezolano coloca a su esposa al frente de sus candidatos a la constituyente
29	i282	Los guerrilleros del ELK intenta llenar el vacío dejado en Kosovo ante la pasividad de la Kfor
12	i283	Nuevos hallazgos de fosas comunes y restos de matanzas serbias
20	i284	Dos refugiados mueren al pisar una mina en el inicio del regreso de los kosovares a sus casas
1000	i285	El Pentágono confirma 20 muertos en escaramuzas en Kosovo
7	i286	La Iglesia ortodoxa serbia pide la dimisión de Milosevic y su Gobierno
20	i287	'Lo peor sería un regreso desordenado de los refugiados'
13	i288	Las tropas españolas salen el día 23 con destino a Kosovo
7	i289	Soros cree que un plan de ayuda provocará la caída de Milosevic
28	i290	Francia se opone a que Europa pague la reconstrucción de los Balcanes
83	i291	El Papa tiene que suspender todos los actos en Cracovia por una gripe
83	i292	Cancelaciones frecuentes
89	i293	Un fuerte terremoto sacude México y causa al menos 14 muertos y 200 heridos
98	i294	Corea del Sur hunde un barco norcoreano en su primera batalla naval desde 1953
154	i295	Woodward y el 'martirio' de Hillary
155	i296	La tercera derrota electoral en tres años acentúa el aislamiento de Chirac
156	i297	El Supremo de Italia anula una condena a Craxi por corrupción
145	i298	Acoso a la inexperta policía de El Salvador
145	i299	Más muertos al año que en Estados Unidos
157	i300	Una escisión de Sendero Luminoso asesina a ocho personas en Perú
158	i301	La beneficiencia se hace cargo de la comida de los presos dominicanos
20	i302	Miles de albanokosovares desbordan a la fuerza internacional en su desordenado regreso
20	i303	'No podemos volver a una vida normal'
22	i304	Cruz Roja calcula que 40.000 serbios han huido en la última semana
8	i305	Los últimos soldados serbios dejan Pristina bajo control de la Kfor
29	i306	La OTAN no desarmará al ELK hasta la total retirada serbia
7	i307	Milosevic pasa a la ofensiva ante la campaña de la oposición y la Iglesia para echarle del poder
12	i308	El Pentágono asegura que la Kfor ha identificado más de 90 puntos con fosas comunes
20	i309	Los aliados quieren desminar Kosovo antes del invierno
16	i310	Alemania propone a Rusia crear una zona en Kosovo con mando compartido por turnos

Clase	Cod_arti	Titular
16	i311	Washington ve 'progresos' para llegar a un acuerdo con Moscú
16	i312	El Gobierno alemán cree que la OTAN no debió revelar su plan de despliegue
1001	i313	La mayoría de edad
89	i314	El terremoto de México deja 24 muertos y 200 heridos
89	i315	El patrimonio de Puebla ha sufrido graves daños
73	i316	Mandela pone fin a su histórica carrera política y deja la presidencia en manos de Mbeki
73	i317	Reyes y ciudadanos
159	i318	El anuncio oficial de la candidatura de Gore abre la campaña presidencial en Estados Unidos
160	i319	El control de armas deriva en un debate moral
83	i320	El Papa reaparece en relativa buena forma, pero suspende su visita a Armenia
75	i321	El primer ministro israelí se dispone a formar Gobierno con la ayuda de los ultraortodoxos
36	i322	Las guerras del futuro tendrán su origen en la violación de los derechos humanos
60	i323	La guerrilla colombiana pone en libertad a 33 de los rehenes secuestrados en una iglesia
58	i324	La guerrilla asesina a 12 campesinos colombianos junto a la zona neutral
67	i325	Amnistía Internacional destaca el 'caso Pinochet' como hito de los derechos humanos
67	i326	Documentos oficiales secretos chilenos pueban la existencia de la Operación Cóndor
90	i327	El perdón de la deuda externa no es suficiente, según la Iglesia
98	i328	EE UU envía buques de guerra a la zona de conflicto entre ambas Coreas
20	i329	Columnas de refugiados vuelven a colapsar los puestos fronterizos en su viaje de regreso
12	i330	El avance de la Kfor deja al descubierto la cadena de horrores serbios contra los kosovares
23	i331	La Iglesia ortodoxa denuncia ataques contra monasterios por parte del ELK
1003	i332	Los albaneses quieren vivir con los 'serbios buenos'
22	i333	Un refugiado serbio, asesinado cuando volvía a Kosovo
13	i334	Los 2.500 soldados italianos desplegados se ven desbordadas para controlar
22	i335	'No hemos conseguido que se quede la población civil serbokosovar'
22	i336	Milosevic intenta detener la desbandada serbia de Kosovo
28	i337	Clinton reitera en París su compromiso con la reconstrucción de los Balcanes
1004	i338	La Duma pide por unanimidad juzgar a Solana como criminal de guerra
16	i339	EE UU presenta una nueva propuesta a Rusia
90	i340	La situación en Yugoslavia amenaza con relegar otros temas en la cumbre del G-8
163	i341	'Vamos a hacer política, no estamos sólo para administrar'
76	i342	Francia culpa a España del bloqueo de la negociación entre la UE y Mercosur
68	i343	Inquietud entre los generales chilenos por los nuevos sumarios contra cargos de la dictadura
67	i344	Baltasar Garzón amplía en 34 casos la acusación por torturas contra el ex dictador Pinochet
67	i345	La desaparición de personas era una 'industria', según un informe
73	i346	Mbeki deja a Winnie Mandela fuera del nuevo Gobierno de Suráfrica
83	i347	El Papa se despide de Polonia con la sugerencia de que tal vez regrese
89	i348	México destinará 4,5 millones de dólares para los damnificados
60	i349	La guerrilla colombiana del ELN pospone la liberación de los rehenes
91	i350	Un cadáver, clave para esclarecer el 'caso Gerardi' en Guatemala
164	i351	Bush rechazó el aplazamiento de otra ejecución en el Estado de Tejas
17	i352	3.600 soldados rusos se unirán a la Kfor, pero no controlarán ningún sector de Kosovo
13	i353	El primer grupo de militares españoles sale hacia Macedonia
24	i354	Los aliados hallan un muerto y 15 apaleados en un centro de torturas del ELK
12	i355	Los británicos descubren otro puesto serbio para la represión
29	i356	La guerrilla discute con la Kfor un desarme parcial
29	i357	'El ELK es un problema que la OTAN debe resolver rápidamente'
12	i358	La OTAN da más poder a las tropas para detener a los criminales de guerra
13	i359	Investigación española
8	i360	Los soldados serbios dejan un rastro de minas y casas quemadas en su retirada
7	i361	La oposición yugoslava alerta en Viena del peligro de una guerra en Montenegro
7	i362	Milosevic se lanza a una campaña de autopromoción personal
90	i363	Rusia intenta aprovechar la crisis de Kosovo para dar un nuevo paso en su vinculación al G-8
90	i364	La cumbre apoya a Schröder en su política sobre Chernóbil
90	i365	Los 'siete grandes' perdonan dos terceras partes de la deuda de los 41 países más pobres
90	i366	El acuerdo, insuficiente para los partidarios de la condonación total
68	i367	El Ejército analiza el procesamiento de mandos militares en Chile
165	i368	Un alto tribunal de Chile confirma la libertad de los directivos de Planeta
60	i369	La guerrilla colombiana libera a otros ocho rehenes en medio de violentos combates
166	i370	EE UU deporta a 99 balseros cubanos en un solo día
167	i371	Mejora el estado de salud del ex presidente argentino Alfonsín tras un accidente de coche
168	i372	Acusan al presidente de lucrarse con un futuro canal en Nicaragua
77	i373	Río estará tomada por las fuerzas de seguridad durante la cumbre de los 49 jefes de Estado
153	i374	El jefe de la policía de Brasil dimite a las 48 horas de jurar el cargo
2011	i375	Choque de poderes
172	i376	El 40 % de los delincuentes procesados en Perú eran policías
173	i377	Barak quiere unir los territorios palestinos
174	i378	Rusia aprueba una amnistía que liberará a cerca de 100.000 presos
175	i379	El Parlamento turco suprime los jueces militares en los tribunales de seguridad
1005	i380	Clinton condiciona la ayuda a Macedonia y Albania a su democratización
7	i381	El opositor Draskovic defiende una convocatoria urgente de las elecciones
13	i382	La zona en la que se desplegará el contingente español soporta un estallido de violencia
13	i383	La Legión se instalará donde los italianos no quieren quedarse
1006	i384	'¡Qué extraña sensación de libertad!'
1007	i385	La loca de Shajkov
46	i386	Una bomba de la OTAN mató a los dos soldados de la Kfor
17	i387	Yelstin somete al Senado el envío de tropas a Kosovo
1008	i388	El primer ministro esloveno se opone a una futura independencia de Kosovo
2012	i389	El obispo católico de Kosovo pide que 'el pasado no adelante al futuro'
2013	i390	El Fondo de Desarrollo presta ayuda a los países fronterizos
84	i391	Blair defiende el Nuevo Laborismo frente a las críticas de los izquierdistas del partido
84	i392	El euro desaparece de los discursos
65	i393	Críticas a la excarcelación del militante del IRA que atentó contra Thatcher
76	i394	'Difícilmente se nos podrá tildar de 'Europa fortaleza'
176	i395	El presidente ruso muestra en público su desdén por el alcalde de Moscú
2014	i396	EE UU someterá al detector de mentiras a sus científicos nucleares
82	i397	El partido de Habibie ya es segundo en el lento escrutinio de Indonesia
93	i398	El presidente Chávez decreta la enseñanza militar de niños y jóvenes en Venezuela
178	i399	El presidente de Chile sustituye a los ministros de Exteriores y Defensa
88	i400	Aclaración sobre el 'caso Pey'
59	i401	La guerrilla de Colombia ataca por sorpresa el cuartel general de los paramilitares
101	i402	Una comisión oficial reconoce que la policía y el Ejército continúan torturando en México

Clase	Cod_arti	Titular
101	i403	El papel de las
179	i404	Banzer cambia la mitad del Gobierno boliviano tras un grave escándalo de corrupción
153	i405	Cardoso desafía los tabúes al elegir a un abogado negro como jefe de la policía en Brasil
96	i406	La reforma de la Constitución levanta polémica en Nicaragua
47	i407	Suiza congela las cuentas de Milosevic y de otros cuatro acusados por crímenes de guerra
1009	i408	Cuatro ministros europeos prometen en Pristina a los kosovares que se hará justicia
1010	i409	Los 'marines' matan a un francotirador en el sur de Kosovo
1011	i410	Los supervivientes de la última matanza serbia
14	i411	Una requisa de armas, primera acción de los españoles en Pec
7	i412	El Gobierno yugoslavo teme una conspiración para derrocar a Milosevic
7	i413	'El Estado que representa Milosevic ya no existe'
1012	i414	'La noticia de mi asesinato me la dieron unos amigos'
28	i415	Economistas serbios cifran en 4,5 billones de pesetas el coste de la reconstrucción
17	i416	Annan negocia con Moscú la administración civil de Kosovo
65	i417	Blair advierte de que el acuerdo del Viernes Santo es la oportunidad final de paz en el Ulster
65	i418	Adams considera que la situación 'puede escapar a cualquier control'
75	i419	Laboristas y Likud negocian en Israel formar una coalición fuerte
181	i420	Favorable acogida al retraso del referéndum en Timor Oriental
59	i421	70 muertos en combates entre el Ejército colombiano y las FARC
182	i422	Cuba vigilará a los militantes comunistas para evitar la creciente corrupción
69	i423	González expone en Chile su desacuerdo con el 'caso Pinochet' y apoya a Lagos
183	i424	El narcotráfico destruye a los miskitos
183	i425	Entre la 'coca' y el Estado
93	i426	El presidente inicia su campaña para refundar Venezuela
185	i427	La cifra de niños contaminados con plomo en México se eleva a 8.000
1013	i428	Kosovo aclama a Solana como su salvador
30	i429	EE UU ofrece 800 millones a quien dé pistas que ayuden a capturar a Milosevic
7	i430	El Parlamento yugoslavo pone fin a tres meses de estado de guerra con la OTAN
2015	i431	Coordinador en los Balcanes
1014	i432	La tragedia inacabable de la familia Perteshi
2016	i433	'Burim pisó una bomba'
25	i434	Asesinados a tiros tres serbios en la Universidad de Pristina
26	i435	Saqueos en Kosovo
900	i436	España presenta una lista de seis carteras de la UE para De Palacio y Solbes
187	i437	'Los blancos creen que sus hijos serán discriminados'
99	i438	Las catástrofes naturales causan más refugiados que los conflictos bélicos
99	i439	Malos augurios
75	i440	Ocho muertos y más de 50 heridos en varios bombardeos israelíes sobre Líbano
65	i441	Blair viaja a Belfast en un esfuerzo decisivo por salvar la paz
188	i442	La huelga en la Universidad de México desborda las aulas y toma tintes políticos
189	i443	El vínculo zapatista
190	i444	El PRI impide una gran reforma electoral para los comicios del 2000
191	i445	Demandadas en Florida 24 empresas por prácticas laborales discriminatorias en Cuba
59	i446	El Ejército de Colombia controla la zona de combate con los rebeldes
192	i447	D'Alema apoya a la oposición a Menem en su visita a Argentina
91	i448	Un ex magistrado implica al Ejército en el asesinato del obispo Gerardi
2017	i449	Sin pistas sobre el 'asesino del tren'
27	i450	Los serbios abandonan Kosovo por centenares ante la ineficaz protección de la OTAN
1019	i451	Miembros del ELK denuncian varios asesinatos y purgas cometidos por los líderes de la guerrilla
1019	i452	La violencia del Serpiente
17	i453	Moscú envía generales para defender los intereses rusos
7	i454	Milosevic inicia la batalla por mantenerse en el poder
2018	i455	Inquisición serbia en Kosovo
2019	i456	'Era peor que el peor policía'
7	i457	Nueva asignatura para Belgrado
65	i458	Blair pide al Sinn Fein un compromiso de entrega de armas en el año 2000
75	i459	Israel refuerza la frontera con Líbano ante posibles represalias por el bombardeo de Beirut
75	i460	Barak se apresura a formar Gobierno
70	i461	Un militar chileno revela que el piloto de Pinochet arrojó al mar a detenidos
70	i462	El Ejército empieza a dar datos sobre la represión
2020	i463	Alemania asume su pasado con la aprobación de un monumento al Holocausto
71	i464	La Internacional Socialista apoya los cambios de gobierno en el Cono Sur
2021	i465	EE UU cierra seis Embajadas en África por temor a ataques terroristas
2022	i466	El nuevo presidente de Suráfrica promete privatizaciones en su plan de liberalizaciones
2023	i467	Apaleado hasta la muerte ante la pasividad de la policía colombiana
2024	i468	Ruiz Massieu pierde otra batalla en su lucha por no ser entregado a México
59	i469	Los últimos combates en Colombia muestran la complicidad entre Ejército y paramilitares
2025	i470	Juicio en Brasil a los 150 policías que asesinaron a 19 campesinos
2026	i471	Marruecos impide que Amnistía Internacional se reúna en Rabat
2027	i472	China ejecuta a 71 personas convictas por tráfico de drogas
2028	i473	Detenido en Corea del Norte un estadounidense por 'violar las leyes'
900	i474	Calvario de Prodi para formar Comisión
2029	i475	Javier Solana hace acopio de ideas
2030	i476	La ONU y la UE pugnan con la guerrilla independendista para administrar Kosovo
28	i477	Los Quince pagarán al menos la mitad de la reconstrucción
2031	i478	La difícil vida del soldado Bojan Krstic en el frente de Kosovo
2032	i479	Condones con sabor a banana
2033	i480	'Soy veterano y sólo me faltan 125 días para licenciarme'
14	i481	Indicios de que los serbios fusilaron a presos de la cárcel de Istok tras el bombardeo aliado
7	i482	'El pueblo serbio también es una víctima del régimen'
901	i483	Tres vías nacionales para un solo socialismo europeo
71	i484	La IS condiciona el cambio social a la estabilidad económica
65	i485	Mo Mowlan cree que se alcanzará un acuerdo en el Ulster antes del miércoles
2034	i486	La guerra sucia de los ayatolás
2035	i487	13 judíos esperan en el corredor de la muerte

Clase	Cod_arti	Titular
2036	i488	El matón a sueldo prospera en Rusia
80	i489	Buteflika anuncia la libertad de miles de islamistas sin delitos de sangre
77	i490	Europa y Latinoamérica pactarán en Brasil una alianza estratégica para el siglo XXI
77	i491	La UE trata de consolidar su presencia en el continente frente al empuje de EE
77	i492	Aznar no cree posible que los Reyes adelanten su viaje a Cuba
77	i493	Chile ve 'buena voluntad' europea en el 'caso Pinochet'
77	i494	El presidente peruano será el más protegido por el riesgo a un atentado
14	i495	Los españoles hallan las tumbas de 97 presos de Istok fusilados por los serbios
2037	i496	La venganza del ELK alcanza a la comunidad gitana
7	i497	El presidente de Montenegro exige la revisión de las relaciones en la Federación Yugoslava
2038	i498	Asesinados un miembro de una ONG y su intérprete
29	i499	La Legión controlará un depósito de armas del ELK
48	i500	España se integrará en el Grupo de países Amigos de Kosovo
2039	i501	Espectacular fuga en helicóptero de un jefe de la mafia marsellesa
80	i502	Argelia reconoce 100.000 muertos en la guerra con los islamistas
2040	i503	La izquierda pierde la alcaldía de Bolonia tras 54 años en el poder
2041	i504	Miles de manifestantes se echan a la calle en Bogotá contra la violencia y los secuestros
2042	i505	Los dos periodistas que filmaron un apaleamiento tienen que huir de Colombia
2043	i506	Wall Street invierte en la guerrilla
93	i507	El Congreso de Venezuela desafía a Chávez al vetar el ascenso de 35 oficiales
2044	i508	Una investigación revela que la mitad de los mexicanos vive en la pobreza
2045	i509	Ortega acalla las protestas sandinistas contra sus pactos con la derecha de Nicaragua
77	i510	La UE, Mercosur y Chile se comprometen a crear sin plazo fijo un área de libre comercio
70	i511	Europa y América Latina hablan sin contar con EE
70	i511	Europa y América Latina hablan sin contar con EE
77	i511	Europa y América Latina hablan sin contar con EE
77	i511	Europa y América Latina hablan sin contar con EE
77	i512	Fidel Castro estrecha relaciones con Aznar, al que califica de 'valiente', 'sabio' y 'afectuoso'
77	i513	Aznar reclama el protagonismo en la cumbre
7	i514	Iglesia ortodoxa, oposición e intelectuales se unen contra Milosevic por la pérdida de Kosovo
20	i515	Comienza el retorno organizado de los refugiados
14	i516	El primer contingente de tropas españolas llega a Kosovo e instala su cuartel general en Istok
29	i517	La guerrilla kosovar inicia la entrega de sus armas a la OTAN
2046	i518	La Alianza reconoce que causó pocos daños al Ejército serbio
17	i519	Moscú envía a Pristina más soldados y equipos para activar el aeropuerto
65	i520	Diálogo 'in extremis' para salvar la paz del Ulster ante el plazo definitivo que vence mañana
77	i521	Estados Unidos solicita la ayuda de España para acceder al procedimiento sobre Chile
77	i522	España acepta la idea de una salida humanitaria al 'caso Pinochet'
1020	i523	Procesado el alcalde de París por la financiación ilegal de los gaullistas
1020	i524	Objetivo, la presidencia
2047	i525	La izquierda italiana sufre una contundente derrota en las elecciones locales
75	i526	El Likud se niega a participar en el nuevo Gobierno de coalición israelí
2048	i527	Una fuerte escolta militar traslada al Tíbet al niño escogido por China como Panchen Lama
2049	i528	Una investigación culpa a un reducido grupo de militares de la matanza de Tlatelolco
2050	i529	Un escándalo de narcotráfico en Bolivia salpica al Ejecutivo de Banzer
1023	i530	Turquía condena a morir en la horca al líder kurdo Ocalan por traición y separatismo
1023	i531	Sombras legales en el proceso al 'enemigo público número uno'
1023	i532	El pueblo sin Estado del 'turco de las montañas'
1023	i533	Movilización internacional para impedir que Ankara ejecute la sentencia del jefe del PKK
1023	i534	30.000 muertos tras 15 años de rebelión
2116	i535	10.000 personas salen a la calle en el centro de Serbia para exigir la dimisión de Milosevic
2117	i536	La Legión toma el control de la zona fronteriza con Serbia
2118	i537	El Pacto para los Balcanes se presentará en Sarajevo en julio
80	i538	Buteflika liberará a 5.000 islamistas el lunes en el aniversario de la independencia argelina
77	i539	Latinoamérica pide un tratamiento de igualdad como socio comercial de Europa y EE
77	i540	Una solución rápida al 'caso Pinochet'
2119	i541	España organizará en el 2002 la segunda cumbre
63	i542	Blair anuncia avances en la negociación para el desarme en Irlanda del Norte
63	i543	Clinton cree que se alcanzará una solución
2120	i544	Una investigación internacional revela un caos en la Administración palestina
2121	i545	El Papa anuncia su deseo de viajar a Tierra Santa y a Irak en el 2000
2122	i546	'Lo sórdido de la política exterior de EE UU sale a la luz en Chile'
2123	i547	El juez Garzón podrá interrogar a un agente vinculado al 'caso Letelier'
2124	i548	El clima de guerra y de inseguridad provoca el éxodo de los más acomodados de Colombia
2125	i549	Asesinados dos miembros de la guardia del presidente de México
2126	i550	La oposición pide que se investigue a fondo la matanza de Tlatelolco
2127	i551	La proliferación de armas sin control agrava la elevada tasa de violencia en El Salvador
2128	i552	Se cambian pistolas por comida
2129	i553	EE UU retira las minas antipersona que protegían Guántanamo
2051	i554	Diputados laboristas piden a Blair recuperar las esencias de izquierda
2052	i555	Las negociaciones sobre el plan de paz del Ulster se prolongan más allá del plazo fijado
2053	i556	Un general en busca del desarme, no de la rendición
2054	i557	Clinton presiona a las partes en Irlanda del Norte para que lleguen a un acuerdo
2055	i558	Los países Amigos de Kosovo se plantean acuñar una moneda propia para la provincia
2056	i559	Justicia para la familia Bala
1016	i560	Milosevic ofrece a la oposición participar en el Gobierno
2057	i561	Soros plantea excluir a Yugoslavia de la primera fase de ayuda
1016	i562	El régimen moviliza a sus partidarios
70	i563	EE UU desclasifica documentos que implican a Pinochet en el sistema represivo de Chile
	i564	'Está volando como un cóndor'
65	i565	Los partidos del Ulster buscan una fórmula que haga simultáneo el desarme y el autogobierno
65	i566	Alternativas para la entrega de armas
2058	i567	Un Ejecutivo orientado a la cooperación con el
2059	i568	Escocia marca distancias con la reina Isabel en la inauguración de su Parlamento autonómico
2060	i569	Las competencias de la nueva Asamblea

Clase	Cod_arti	Titular
2061	i570	20 muertos al caer un teleférico desde una altura de 80 metros en los Alpes franceses
2062	i571	Avalanchas, incendios, inundaciones...
1016	i572	'Si Milosevic no dimite, hay riesgo de guerra civil en Serbia'
29	i573	Las fuerzas de la Kfor y el ELK pugnan por controlar los archivos públicos de Kosovo
1016	i574	El presidente montenegrino rechaza el intento de acercamiento de Milosevic
2063	i575	El Senado norteamericano califica a Yugoslavia de Estado terrorista
2064	i576	Quejas por el retraso de las promesas de la UE al Este
2065	i577	La CIA definió a Pinochet como el 'líder' que postulaba en Chile ejecuciones sumarias
69	i578	Una visita de González a Santiago
2066	i579	Cinco muertos en Turquía en el primer atentado tras la condena a Ocalan
2067	i580	Alemania cierra medio siglo de historia en la sede del Parlamento en Bonn
2068	i581	Bush, favorito para la presidencia de EE UU al recaudar el doble de dinero que Gore
2069	i582	36 heridos en Indonesia al disparar la policía contra quienes pedían la dimisión del presidente
2070	i583	Fallece Joshua Nkomo, artífice de la independencia de Zimbabue
2071	i584	'Me siento predicando la virtud de la virginidad en un burdel'
2072	i585	Washington cree que Colombia está perdiendo la guerra de la droga
1018	i586	Las elecciones en el Estado de México marcan el arranque de la carrera por la presidencia
2073	i587	La policía dispara contra un diputado mexicano para intervenirle el coche
2074	i588	Fujimori se niega a acatar un fallo de la Corte Interamericana de Derechos Humanos
93	i589	La ministra de Hacienda venezolana rompe con Chávez
2075	i590	Muere la líder de la lucha por los desaparecidos en Chile
2076	i591	Un tribunal francés condena a 10 años de cárcel al general Noriega
2077	i592	Una estatua con 40 años de censura
65	i593	Londres y Dublín dan un ultimátum para constituir el Gobierno del Ulster el día 15
65	i594	Clinton celebra el acuerdo y pide que no se destruya lo logrado
65	i595	La Comisión cree que el desarme puede terminar en mayo del 2000
65	i596	Londres envía refuerzos militares a la provincia ante el comienzo de las marchas protestantes
2078	i597	La OTAN detiene a once soldados serbios que actuaban en dos misiones dentro de Kosovo
1016	i598	El general Clark asegura que Milosevic planea una acción contra Montenegro
2079	i599	El francés Kouchner, nombrado administrador provisional de la ONU
1016	i600	El partido de Draskovic rechaza entrar en el Gobierno
2080	i601	La Fiscalía suiza no encuentra las cuentas de Milosevic
2081	i602	40 inmigrantes mueren al chocar sus barcazas frente a las costas de México
2082	i603	Kissinger fue avisado antes del asesinato de Letelier de que Pinochet actuaría en el exterior
2083	i604	El ex dictador consulta su caso a la diplomacia española
2084	i605	El ex prefecto de Córcega deja la cárcel y amenaza
2085	i606	El 90 % de las muertes causadas por las guerras en la actualidad son civiles
2086	i607	Todos contra el neoliberalismo en México
93	i608	El presidente de Venezuela se declara en 'guerra' con el Congreso
2087	i609	EE UU investigará si hubo malos tratos a los balseros en Miami
80	i610	El presidente argelino indulta a miles de islamistas antes de su debate en el Parlamento
2088	i611	Reconciliación sin verdad
2089	i612	Los orangistas de Portadown concluyen su marcha más pacífica de los últimos años
65	i613	Trimble recibe presiones de Blair y de la cúpula unionista
1021	i614	Rusia tacha de 'provocación' el bloqueo aliado al envío de sus tropas
2090	i615	El opositor Djindjic regresa a Belgrado y propugna un cambio democrático
2091	i616	La Legión confiaba en que el ELK protegiera a los serbios asesinados
	i617	La desclasificación del horror
80	i618	Cientos de islamistas son liberados en Argelia como gesto de reconciliación
80	i619	Punto final a siete años de estado de excepción
65	i620	El líder unionista dice que el plan de Blair permite al IRA vetar el proceso de paz
1021	i621	Moscú y la OTAN eliminan los obstáculos para el despliegue militar ruso en Kosovo
1018	i622	Primera victoria de la oposición mexicana sobre el PRI con una candidatura conjunta
2092	i623	El difícil camino hacia la unidad
1022	i624	Barak se compromete a poner fin a cien años de conflicto árabe-israelí
2093	i625	La familia de Letelier pide a EE UU que colabore con el juez Garzón
2094	i626	El 'caso Leighton'
2095	i627	Muere una militante kurda en Turquía en un atentado suicida con 14 heridos
2096	i628	Los islamistas rechazan el pacto de Pakistán con Clinton sobre Cachemira
2097	i629	Bélgica prepara un Gobierno 'arcoiris' liberal, socialista y ecologista
2098	i630	Yeltsin confirma que la momia de Lenin será retirada de la plaza Roja
2099	i631	El Partido Popular Europeo exige la presidencia de la Eurocámara
93	i632	Varios diputados solicitan que el presidente Chávez sea procesado
2100	i633	El Gobierno de Ecuador decreta el estado de emergencia ante la huelga de transportes
2101	i634	Colombia y Perú son los países donde más niños actúan como soldados
2102	i635	Comienza en Cuba un juicio por daños y perjuicios contra EE
2103	i636	Los peronistas se imponen en las elecciones locales de Tierra del Fuego
1022	i637	Barak se marca la reanudación del proceso de paz como eje de su programa de gobierno
1022	i638	Rusia y Siria creen que el nuevo primer ministro abre una buena oportunidad
2104	i639	Periodistas, verdugos y víctimas en Kosovo
2105	i640	Clark afirma que las tropas rusas en Kosovo estarán bajo mando de la OTAN
2106	i641	Las fuerzas de la Sfor detienen al ex vicepresidente serbobosnio
65	i642	Trimble busca el apoyo de los católicos moderados para gobernar sin el Sinn Fein
80	i643	Los militares antiislamistas intentan marcar el ritmo de la política reconciliadora en Argelia
2107	i644	Hillary Clinton formaliza su candidatura al Senado de EE
2108	i645	El buen tiempo favorece una nueva ola de balseros cubanos hacia EE
2109	i646	Seis disidentes cumplen un mes de ayuno
2110	i647	Florida aplaza la primera ejecución en la silla eléctrica de este año
1018	i648	Inquietud en el PRI tras el triunfo electoral de la oposición en un Estado mexicano
2111	i649	Voces contra la unidad
2112	i650	Asesinado por sus compañeros el jefe máximo del 'cartel del Golfo'
2113	i651	Defensores de los derechos humanos dicen que Guatemala prepara una 'ley de punto final'
2114	i652	Indignación en Nicaragua por el trato a sus emigrantes en Costa Rica
2115	i653	El diálogo de paz en Colombia se pospone de nuevo hasta el 20 de julio

B. TagTimex

Módulo de codificación semántica de las expresiones temporales. Está implementado mediante dos ficheros tagtimex.pl y base_lexico.pl en el lenguaje Sictus Prolog.

Fichero tagtimex.pl

```
:-use_module(library(queues)). :-use_module(library(system)).
:-dynamic   stop/1.

es_listachar([A|B]):-simple(A).

bonito([]).
bonito(Cadena):-format("~s",Cadena).

%%%Busco los periódicos del directorio ORI y los analizo con la producción 'newspaper'
%%%guardando en el directorio DATOD la salida y siendo D el directorio de trabajo

newspapers:- working_directory(D,D),
Dato='.\\..\\news\\datos',Ori='.\\..\\roma\\news\\ori',
    working_directory(X,Ori), directory_files('199906*',FileList),
    newspaper(FileList,Ori,Datos,D), working_directory(X,D).
newspaper([],A,B,C):-bonito("FIN").
newspaper(Lista,Ori,Datos,Home):-  append([X],R,Lista), open(X,read,Stream),
    working_directory(B,Home), working_directory(A,Datos), tell(X),
    get_lines(Stream,WLista), close(Stream), told, working_directory(E,Home),
    working_directory(C,Ori),bonito("1"), newspaper(R,Ori,Datos,Home).

%%%Producción que lee todo un fichero y lo analiza con la producción par

get_lines(SSS,[]):- at_end_of_stream(SSS),!. get_lines(SSS,W):-
get_chars(SSS,Pal0), par(Pal0,Result),!,bonito(Result),
    nl, get_lines(SSS,Rest).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% GRAMATICA PARA EL EXTRACTOR DE FRASES TEMPORALES
%%reconoce un párrafo con una posible  sentencia temporal
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

par(A,B):-((par0(A,C),B=C);(B=A)). par0(X,Y):-tok("<TEX",X,B),Y=X.
par0(X,Y):-parra(X,[],Y). parra("",J,Y):-Y=J.
parra(X,[],Y):-par1(A,X,Z),!,parra(Z,A,Y).
parra(X,J,Y):-par1(A,X,Z),!,junta(A,J,P), parra(Z,P,Y).
par1(X,Y,J):- (( separador(D,Y,J),X=[D]);
noexptemp(D,Y,J),((J=="",X=Y);(append(E,J,Y),X=E)));
    ( buscafech(B,Y,R),J=R,X=B); (coso(A,Y,J),((J=="",X=Y);append(X,J,Y)))).

%%coso  cadena sin espacios que termina con un espacio

coso(X)-->(separador(P),{X=[P]});[C],((coso(Y),{X=[C|Y]});{X=[C]}).

partedias(H)-->(((((por;en),spc);{true}),((a;arti),spc),word(W),{partedia(W,M)},
    ((termgn,{H="dp"});{H=M})); (word(W),{partedia(W,M)},termgn,{H="dp"});(mod(C),
    ((spc,(arti;d_e),spc);(({C=="",!,fail});spc)), word(W),!,{partedia(W,M),junta(M,C,H)})).

buscafech(A,B,R):- grupo_exp(D1,B,X0),!,((d_e(X0,X00),spc(X00,X),!,junta(D1,"R",D));(X=X0,D=D1)),
    append(S,X,B),junta(D,"<TIME Value=",L),junta("> ",L,M),junta(S,M,V),junta(" </TIME>",V,A),R=X.
```

```
grupo_exp(P)-->(((((d_e;en),spc);{true}),noexptemp(C),!,{fail});{true}),(((interval,spc,
   {S="I"});{S=""}),exp_simple(M0),!,{junta(M0,S,M)}),((spc,((y,spc,{M1=M});((a;hasta),spc,
   {junta(M,"I",M1)})),grupo_exp(N),!,{ junta(N,",",N2), junta(N2,M1,H)});(spc,(d_e;en),spc,
   grupo_exp(N),!,{junta(N,"##",O),junta(O,M,H)});{H=M}),({junta_comas(H,N1)};{N1=H}),!,
   ((spc,modifi(L1),!,spc,((word(W),(tok("s");tok("r")),spc,{P=N1});{junta(N1,L1,P)}));
   ((spc,mod(L1)),{junta(N1,L1,P)});{P=N1}).

exp_simple(M)-->(tok("(");{true}),((partedias(W11),((spc,d_e,spc,{B=""});{B=W11}));(((modifi(L0),
   spc);{L0="",true}),((mod(L),((spc,(d_e;artis;a),spc);(coma,cadena(P),coma,spc);spc));(word(W0),
   {verbos(W0,C1)},(spc,{L=C1};(spc,word1(X),spc,{L=C1})));{L=""}),!,{append(L,L0,L1)},
   ((partedias(W11),((spc,d_e,spc,{B=""});{junta(W11,L1,B)}));{B=""})),!,(({B==""},!,
   ((noexptemp(B1),!,{fail});expabs0(B1);exprel0(B1);((word(W);word1(W)),{nucleo(W,B1)},((word(W1),
   {fail});{true}))),{junta(B1,L1,B2)});(spc;{true}),{B2=B}),!,
   ((separador(C),partedias(C1));{true}),  {M0=B2,((es_listachar(M0),M=[M0]);M=M0)}.

%palabras separadas por espacios coma,cadena(P),coma

subexp(C)-->cadena(P),!,coma,spc.
cadena(C1)-->word(C2),!,cadena(C3),!,{append(C2,C3,C1)}.
cadena(C1)-->spc,cadena(C2),!,{append(" ",C2,C1)}. cadena([])-->!.

%%ESPRESION TEMPORAL

diasem(N)-->word(A),!,{diasem(A,N)},((spc,mayuscula(D),!,{fail});{true}).
dia(D)--> diasem(D); ( dia0,spc,cuant(M),{junta(M,"d",D)});(arti,spc,(  (diasem(N),{D=N});
    (num(M),(spc,( (dia0;(word(W),tok("s"),spc)),!,{fail});((separador(W);coma),{junta(M,"d",D)}))))) .
mes(M)  --> (((mes0,spc,de,spc);{true}),word(W),!,{nmes(W,M)},((spc,mayuscula(D),!,
    {fail});{true}));(mes0,spc,cuant(N),{junta(N,"m",M)}).

l_dias(N)  --> ((dias0;dia0),spc,l_cuant(M),!,{junta(M,"d",N1),N=N1});(dia(D),(((tok(",");(spc,y)),
    spc,(l_dias(M);(cuant(X), {junta(X,"d",X1),M=[X1]})),{append([D],M,N)}); {N=[D]})) .
l_mes(N)-->((mes(A),!,(((tok(",");(spc,y)),spc,l_mes(R),{append([A],R,N)});
     ({N=[A]}))); ((meses0;mes0),spc,((l_cuant(N1),!,{junta(N1,"m",N)});(de,spc,l_mes(N))))).
l_any(H) -->(spc;{true}),( ( (en;d_e),spc,((artis,spc,{C="1"});{true}));{true}),((num0(A)),!,
    {length(A,P)},(({P==4,junta(A,"#y",B)});({P==2,A==[A1|A2],A1\=="o",C=="1"},((spc,word(S),
    ((tok("s"),spc,!,{fail});{true}) ); {true} ),{junta(A,"#y19",B)})),{H=B}).
l_any0(N)  --> (anos0,spc,((l_num(A),!,{junta(A,"#y19",B),N=B});(cuant(H),{junta(H,"#z",N)});
    (l_cuant(A),!,{junta(A,"#y",B),N=B}))); (ano0,spc,(cuant(A);num0(A)),!,{length(A,B)},
    ({B<2,!,fail};{true}),{junta(A,"#y",N1),N=N1});( num0(A1),!,{junta(A1,"y",A)},
    ((spc,de,!,{fail});{true}),((tok("-"),num0(B),{length(A,Aa),length(B,Ba),Aa>1,Ba>1,
    junta(B,"y",B1),append([A],[B1],N)});({length(A1,P),P==4},( ((spc;{true}), (tok(",");y ),spc,
    l_any0(N1),{append([A],N1,N)}); {N=[A]}))))).

granos(N)-->word(S),!,(({(append(X,"es",S);append(X,"s",S)),grano(X,C),
    junta("p",C,N)}); ({grano(S,N)})).
mod(M)-->(d_e,spc;{true}),(mods(C);(word(W),!,{modificador(W,C)});{fail}),((((spc,(d_e;artis));
    {true}),spc,mod(D),{junta(D,C,M)});(word(W2),{fail}); {M=C}).

%% Lectura desde el fichero

get_chars(S,[]) :- at_end_of_stream(S),!.
get_chars(S,Pal):-get0(S,C),!,(C\==10,Pal=[C|New],get_chars(S,New);
Pal=[]).

pasomin(M,M2):-M=[C|Rest],  convierte(C,C2),M2=[C2|Rest].
convierte(C,C2):- (is_upper(C),to_lower(C,C2); acentos_espanyol(C,C2);  C=C2),!.

%% Gramáticas para Nombre propios y terminos
separador  --> spc;tok(",");([C],{separadores(L),memberchk(C,L)}).
separador(C) -->(spc,{[C]=" "});(tok(","),{[C]=","});([C],{separadores(L),memberchk(C,L)}).

index_term(W) --> nombre(W); suceso(W);  sigla(W),{length(W,S),S>1}.

sigla(W)-->  mayuscula(C),lista_mayusculas(L),separador, {append([C],L,W)}.
```

```
lista_mayusculas([C|L]) --> sp,mayuscula(C),lista_mayusculas(L).
lista_mayusculas([]) --> [].
up_word(W)    --> mayuscula(C1),minuscula(C2),word(Resto), {W=[C1,C2|Resto]}.
lw_word(W)    --> minuscula(C),word(Resto),{W=[C|Resto]}.

romano([H|L]) -->  num_rom(H), l_rom(L).
l_rom([H|L])  --> num_rom(H), l_rom(L). l_rom([])     -->  [].
num_rom(C)    -->  [C],{memberchk(C,[0'X,0'I,0'C,0'L,0'V,0'M])}.

word(C)-->word1(X),{length(X,X1),((X1<1,fail);(pasomin(X,Y),C=Y))}.
word1([H|L])   -->  letter(H),(word1(L);word0(L)).
word0([])-->  []. letter(C)-->[C],{\+separador(A,[C],B)},{\+coma([C],A)}.

mayuscula(C)  --> [C],{es_may(C)}.
minuscula(C)  --> [C],{es_min(C)}.
cualquiera(W) --> (spc,{W=" "});([A],{W=A}).

coma --> spc,coma. coma --> tok(",").

%junta datos de 2 listas separadas por ','

append_lists(L,Out):- append_lists1(L,[],Out),!.
junta_comas(A,B):-(es_listachar(A),B=A);juntacomas(A,[],B).
juntacomas([],Out,Out).
juntacomas([Cadena|Resto],[],D):-(juntacomas(Resto,Cadena,D);D=Cadena).
juntacomas([Cadena|Resto],C,D):-append(C,",",Y),append(Y,Cadena,X),!,(juntacomas(Resto,X,D);D=X).

junta(L,"",Out):-Out=L. junta("",B,Out):-Out=B.
junta(L,B,Out):-es_listachar(B),es_listachar(L),append(B,L,Out).
junta([L|Rest],B,Out):-(Rest==[],junta(L,B,C),Out=C);(junta(L,B,B1), junta(Rest,B,A1),
    ((es_listachar(A1),A2=[A1]);A2=A1), ((es_listachar(B1),B2=[B1]);B2=B1),append(B2,A2,Out)).
junta(L,[B|Rest],Out):-(Rest==[],junta(L,B,C),Out=C);(junta(L,B,B1), junta(L,Rest,A1),
    ((es_listachar(B1),B2=[B1]);B2=B1), ((es_listachar(A1),A2=[A1]);A2=A1),append(B2,A2,Out)).
junta1([],B,Out,Out).
junta1([L|Rest],B,Old,Out):- (Rest==[],junta(L,B,Out));
         (append(B,L,M), append(Old,[M],New), junta1(Rest,B,New,Out)).

append1(A,B,C):-append(A,B,C).

%genero una lista con dos listas de caracteres

append2(A,B,C):- ((es_listachar(A),A1=[A]);A1=A),((es_listachar(B),B1=[B]);B1=B),
    append(A1,B1,C).

%junta los datos de la 2 lista en la primera y los devuelve en out

juntalistas([],B,Out):- Out=B. juntalistas(L,[],Out):- Out=L.
juntalistas(L,B,Out):- (junta(L,B,Out));juntalistas1(L,B,[],Out).
juntalistas1(L,[],Out,Out).
juntalistas1(L,[B|Rest],Old,Out):- junta(L,B,M),((es_listachar(M),M1=[M]);M1=M),
    ((Old==[],New=M1);junta(M1,Old,New)), juntalistas1(L,Rest,New,Out).

%número o lista de números
l_cuant(N) --> l_num(N);(cuant(A),!,((spc,y,spc,cuant(B),{append(C,"0",A),
     length(B,D),D<2,append(C,B,N)}); ( ( (coma,spc,((artis,spc);{true}));
     tok("-");(spc,(a;y),spc,((artis,spc);{true})))),l_cuant(R),{append([A],R,N)});{N=[A]})).

%lista de números de menos de 3 cifras
l_num(N) --> num(A),!,((((coma,spc,((artis,spc);{true}));(spc,(y;a),spc,((artis,spc);{true}));
         tok("-")),l_num(R),!,{append([A],R,N)}); {N=[A]}).

%cuantificador
cuant(M) --> (un,spc,cuant(N),{N=M});((num(N),((tok(th),{append(N,'o',M)}) ;
```

```
    {M=N}));  ( word(W),!, {cuantificador(W,M)})).

indef(M)--> word(W), !,{indefinido(W,M)}.
nonum(N)-->num0(A),(((tok(".");tok(",")),num0(B));tok("%")),coso(C).
num(N)-->num0(A),!,{length(A,N1)},(({N1>2},!,{fail});
     ( (( ((tok(".");tok(",")),num0(B));tok("%")), !,{fail});{N=A})).
num0([H|L])  --> number(H), num_rest(L).
num_rest(N) --> num0(N). num_rest([])--> [].
number(C) --> [C], {is_digit(C)}.

mayuscula --> [C], {es_may(C)}.
punto     --> ".".
guion     -->"-".
tok([])    --> []. tok([H|L]) --> [H],!,tok(L).

spc  --> [32],lspc. lspc --> [32],lspc. lspc --> [].
sp  --> " ", sp. sp  --> [].

is_sep(C)  :- separadores(L),memberchk(C,L). non_sep(C) :-
separadores(L),non_member(C,L).

separadores([151,161,32,10,13,34,39,0'.,0'«,0'»,0'&,0'!,0'?,0'¿,
         0'/,0';,0':,0'-,0'_,0'(,0'),0'[,0'],0'=]).

es_min(X) :- X >= 0'a, X =< 0'z,!;
    memberchk(X,[0'ñ,0'á,0'é,0'í,0'ó,0'ú,0'ä,0'ë,0'ï,0'ü,0'ã]).

es_may(X) :- X >= 0'A, X =< 0'Z,!; memberchk(X,[0'Ñ,0'Á,0'É,0'Í,0'Ó,0'Ú]).

append_lists(L,Out):- append_lists1(L,[],Out),!.
append_lists1([],Out,Out).
append_lists1([L|Rest],Old,Out):- append(Old,L,New),!,  append_lists1(Rest,New,Out).
```

Fichero Base_lexico.pl

```
% BASE DE HECHOS
acentos_espanyol(0'Á,0'á).
acentos_espanyol(0'É,0'é).
acentos_espanyol(0'Í,0'í).
acentos_espanyol(0'Ó,0'ó).
acentos_espanyol(0'Ú,0'ú).
acentos_espanyol(0'Ñ,0'ñ).

interval -->tok("a partir de");tok("desde");tok("entre"); tok("A partir de");
   tok("Desde");tok("Entre").
de --> (spc;{true}),tok("de").
d_e --> (spc;{true}),(tok("del");(tok("de"),((spc,artis);{true}))).
por --> (spc;{true}),tok("por").
y-->(spc;{true}),(tok("y");tok("o");tok(",")).
hasta -->(spc;{true}),(tok("hasta");tok("Hasta")),((spc,artis);{true}).
a --> (spc;{true}),(tok("al");tok("Al");tok("A");tok("a")),((spc,artis);{true}).
medio-->tok("medio");tok("media").
en-->(spc;{true}),(tok("en");tok("En")).
un-->tok("un");tok("Un").
ahora -->tok("Ahora");tok("recientemente");tok("últimamente");
```

```
     tok("Ya");tok("Todavía").

dia0 -->tok("día").
mes0 -->tok("mes").
ano0 -->tok("año").
dias0 -->tok("días").
meses0 -->tok("meses").
anos0 -->tok("años").

% terminaciones palabras numero o genero

termgn -->tok("os").
termgn -->tok("es").
termgn -->tok("as").
termgn -->tok("s").
termgn -->tok("a").
termgn -->tok("o").
termgn -->tok("e").

arti -->(spc;{true}),(tok("la");tok("La");tok("El"); tok("el")).
artis -->(spc;{true}),(tok("las");tok("los");tok("la");
        tok("el");tok("Las");tok("Los");tok("La"); tok("El")).

%Expresiones no temporales
noexptemp(N) --> (mayuscula(N),word(B),spc,mayuscula(P));nonum(M);
    (tok("Día"),spc,num(P)); tok("hoy día");    tok("hoy en día");
    tok("hoy por día");    tok("punto final");    tok("del día siguiente");
    (a,spc,granos(C),((spc,(cuant(C1);mod(C1)),!,{fail});{true}));
    ((( tok("tod"),termgn,spc,artis);((tok("parecen");tok("al cabo"))
       spc,d_e)),spc,(granos(M);partedias(M)));
    ((((l_cuant(B);indef(B);num0(B)),spc);{true}),(tok("horas");
       partedias(M);granos(A)),spc,(tok("tras");tok("a");tok("en");y),
       spc,(partedias(M1);granos(A1);(word(M1), {nucleo(M1,M)})));
    (((l_cuant(X);indef(X)),spc),partedias(M));
    (((cuant(X),spc);{true}),de,spc,partedias(M),((a,spc,partedias(O));{true}));
    (granos(M),spc,(tok("como");de),spc,partedias(B));
    (tok("a la edad de "),cuant(P),spc,granos(B));
    ((a;de;{true}),spc,(cuant(P);indef(P)),spc,granos(M),spc,de,spc,tok("edad"));
    ((a;tok("cada");tok("ningún");tok("Cada")),spc,((cuant(C),spc);{true}),
       (granos(M);(word(M1),{nucleo(M1,M);partedia(M1,M)})));
    (tok("paso"),spc,de,spc,artis,spc,granos(M)).

%%%Modificadores
modifi(M)-->(((tok("Antes");tok("Después");tok("antes");tok("después")),
     spc,d_e);((tok("Anteriores");tok("anteriores");tok("posterior");
     tok("Posteriores")), spc,a)),spc, {M="R"}.

mods(M) --> tok("cuestión de"),{M="I+"}.
mods(M) --> tok("han sido"),{M="-"}.
mods(M) --> tok("va de"),{M="nO"}.
```

```
mods(M) --> tok("a lo largo de tod"),termgn,artis,{M="0"}.
mods(M) --> tok("a lo largo "),d_e,{M="0"}.
mods(M) --> tok("que viene"),{M="+"}.
mods(M) --> tok("poco más"),spc,de,{M=""}.
mods(M) --> tok("poco más tarte"),{M="+"}.
mods(M) --> tok("más tarde"),{M="+r"}.
mods(M) --> tok("dentro de"),{M="+"}.
mods(M)-->tok("más"),spc,d_e,{M=""}.
mods(M)-->tok("menos"),spc,d_e,{M=""}.
mods(M)-->tok("al menos"),{M=""}.
mods(M)-->tok("un par"),spc,de,{M="2"}.
mods(M)-->tok("ha sido"),{M="P-"}.
mods(M)-->tok("de antelación"),{M="r-"}.
mods(M)-->tok("desde"),spc,tok("entonces"),{M="r"}.

%%%Expresiones relativas
exprel0(H)-->((artis;a),spc),exprel0(H).
exprel0(H)-->(tok("después");tok("después")),spc,d_e,spc,exprel0(M),{junta(M,"I0-",H)}.
exprel0(H)-->(tok("Antes");tok("antes")),spc,d_e,spc,exprel0(M), {junta(M,"I0+",H)}.
exprel0(H)-->en,spc,exprel0(M),{junta(M,"I0",H)}.
exprel0(H)-->(tok("Tras");tok("tras")),spc,exprel0(M),((spc,d_e);{true}),{junta(M,"I-",H)}.
exprel0(H)-->d_e,spc,exprel0(M),spc,d_e,{junta(M,"-",H)}.
exprel0(H)-->(tok("hace");tok("hacía");tok("Hace");tok("Hacía")),spc,exprel0(M),spc,
    de,{junta(M,"-",H)}.
exprel0(H)-->mod(C),!,spc,exprel0(M),{junta(M,C,H)}.
exprel0(H)-->d_e,spc,granos(M),{junta(M,"0",H)}.
exprel0(H)-->exprel(H).
exprel(H)-->((l_cuant(B);indef(B);num0(B)),spc,((mod(C),!,spc,{junta(B,C,C1)});{C1=B})),
     (granos(M);(tok("jornadas"),{M="dp"})),({B=[],H=M};((spc,y,spc,medio,{junta(M,C1,M1),
     junta(M1,"0.5",H)});({junta(M,C1,H)}))).
exprel(H)-->medio,spc,(granos(M);(tok("jornadas"),{M="dp"})),{junta(M,"0.5",H)}.
exprel(H)-->((mod(C),!,spc);{C=""}),(granos(M);(tok("jornadas"),{M="dp"})),{junta(M,C,H)}.
exprel(H)-->((mod(A),!,spc);{A=""}),(((por;en),spc);{true}),((((de;a;artis),spc);{true}),
    ((cuant(C1);num(C1);indef(C1)),spc,{junta(C1,A,C)});{C=A}),word(W),tok("s"),
    ({partedia(W,M1)},!,((word(W1),{!,fail});{true}),((spc,d_e,spc,
    {junta(M1,"R",M)});{M=M1})),{junta(M,C,M2)},((spc,noexptemp(C1),!,{fail});
    (spc,mod(A1),!,{junta(M2,A1,H)});{H=M2}).

%%%Expresiones absolutas
expabs0(H)-->(((((a;d_e;hasta;en;por),spc);{true}),((artis,spc);{true}),!,
    (expabs(H);(mod(C),spc,{bonito("22")},expabs0(H1),{junta(C,H1,H)})))); expabs(H)).
expabs(H) -->l_cuant(D),!,(spc;separador(C1)),((((de,spc);{true}),l_mes(M));
       (l_num(M0),{junta(M0,"m",M)})), {junta(M,"#",X),junta(D,"#d",S),
       junta(S,X,Z)},((((spc,de,spc);separador(C2)),
       (l_any(Y);(num0(N1),{junta(N1,"#y",Y)})), {junta(Z,Y,H)});{H=Z}).
expabs(H) --> l_mes(M),!,(( spc,((d_e,spc);{true}),(
      (l_any(Y),{junta(M,"#",N),junta(N,Y,H)});
      (num(N),{junta(N,"#y",X),junta(X,M,H)}) ));
      (separador(C),l_cuant(N),{junta(N,"#d",X),junta(X,M,H)}); {H=M}).
expabs(H)-->(l_any0(H);l_any(H);l_dias(H)).
```

```
expabs(H) --> tok("siglo"),spc,romano(N),((tok(.),romano(M),{fail});{junta(N,"s",H)}).
expabs(H) --> tok("década de los"),spc,l_cuant(N),{junta(N,"z",H)}.
expabs(H) --> tok("siglos"),spc,romano(N),subexp(C),romano(M),spc,
     {junta(N,"s",N1),junta(M,",s",M1),junta(M1,N1,H)}.
expabs(H)--> granos(C),!,spc,num(N),{junta(N,C,H)}.
expabs(H)-->arti,spc,granos(M),{junta(M,"",H)}.

%%BASE DE HECHOS: palabras importantes en expresiones temporales

grano("día","d").
grano("semana","w").
grano("mes","m").
grano("trimestre","t").
grano("cuatrimestre","c").
grano("semestre","e").
grano("año","y").
grano("década","z").
grano("siglo","s").
grano("milenio","l").

diasem("lunes","0w#d1").
diasem("martes","0w#d2").
diasem("miércoles","0w#d3").
diasem("jueves","0w#d4").
diasem("viernes","0w#d5").
diasem("sábado","0w#d6").
diasem("domingo","0w#d7").

partedia("anochecer","rd").
partedia("amanecer","rd").
partedia("madrugada","rd").
partedia("mañana","rd").
partedia("mediodía","rd").
partedia("tarde","rd").
partedia("noche","rd").
partedia("momento","rd").
partedia("víspera","r-1d").
partedia("jornada","rd").

nucleo("jornadas","rdp").
nucleo("hoy","ny#nm#nd").
nucleo("ayer","-1ny#nm#nd").
nucleo("anoche","-1ny#nm#nd").
nucleo("mañana","+1ny#nm#nd").
nucleo("anteayer","-2ny#nm#nd").
nucleo("Ahora","0w").
nucleo("recientemente","I0-3dp").
nucleo("últimamente","I0-3dp").
nucleo("Ya","0w").
nucleo("todavía","w").
```

```
nucleo("carnaval","kOw#d2").
nucleo("purificación","Oy#m12#d10 ").
nucleo("purísima","Oy#m12#d10 ").
nucleo("cuaresma","kpcuaresma").
nucleo("navidades",["IOy#m12#d25","+r1y#m1#d5"]).
nucleo("primavera",["IOy#m3#d22","Oy#m6#d21"]).
nucleo("verano",["IOy#m6#d22","Oy#m9#d21"]).
nucleo("otoño",["IOy#m9#d22","Oy#m12#d21"]).
nucleo("invierno",["IOy#m12#d22","+r1y#m03#d21"]).
nucleo("cincuentenario","k").
nucleo("lustro","k").
nucleo("cuaresma","kpcuaresma").
nucleo("pascua","kpcuaresma").

nmes("enero","Oy#m1").
nmes("febrero","Oy#m2").
nmes("marzo","Oy#m3").
nmes("abril","Oy#m4").
nmes("mayo","Oy#m5").
nmes("junio","Oy#m6").
nmes("julio","Oy#m7").
nmes("agosto","Oy#m8").
nmes("septiembre","Oy#m9").
nmes("octubre","Oy#m10").
nmes("noviembre","Oy#m11").
nmes("diciembre","Oy#m12").

indefinido("demasiados","").
indefinido("algunos","").
indefinido("algunas","").
indefinido("varios","").
indefinido("varias","").
indefinido("unos","").
indefinido("decenas","").
indefinido("unas","").
indefinido("los","").
indefinido("las","").
indefinido("pocos","").
indefinido("pocas","").
indefinido("muchas","").
indefinido("muchos","").
indefinido("tantos","").
indefinido("exactamente","").

cuantificador("medio","0.5").
cuantificador("media","0.5").
cuantificador("par","2").
cuantificador("quince","15").
cuantificador("docena","12").
cuantificador("cien","100").
```

```
cuantificador("ciento","100").
cuantificador("cincuenta","50").
cuantificador("cuarenta","40").
cuantificador("mil","1000").
cuantificador("sesenta","60").
cuantificador("setenta","70").
cuantificador("ochenta","80").
cuantificador("noventa","90").
cuantificador("una","1").
cuantificador("uno","1").
cuantificador("un","1").
cuantificador("dos","2").
cuantificador("tres","3").
cuantificador("cuatro","4").
cuantificador("cinco","5").
cuantificador("seis","6").
cuantificador("siete","7").
cuantificador("ocho","8").
cuantificador("nueve","9").
cuantificador("diez","10").
cuantificador("once","11").
cuantificador("doce","12").
cuantificador("trece","13").
cuantificador("catorce","14").
cuantificador("quince","15").
cuantificador("dieciséis","16").
cuantificador("diecisiete","17").
cuantificador("dieciocho","18").
cuantificador("diecinueve","19").
cuantificador("veinte","20").
cuantificador("veintiún","21").
cuantificador("veintidós","22").
cuantificador("veintitrés","23").
cuantificador("veinticuatro","24").
cuantificador("veinticinco","25").
cuantificador("veintiséis","26").
cuantificador("veintisiete","27").
cuantificador("veintiocho","28").
cuantificador("veintinueve","29").
cuantificador("treinta","30").
cuantificador("primer","o1").
cuantificador("primero","o1").
cuantificador("primera","o1").
cuantificador("segunda","o2").
cuantificador("segundo","o2").
cuantificador("tercer","o3").
cuantificador("tercera","o3").
cuantificador("tercero","o3").
cuantificador("cuarto","o4").
cuantificador("cuarta","o4").
```

```
cuantificador("quinto","o5").
cuantificador("quinta","o5").
cuantificador("sexta","o6").
cuantificador("sexto","o6").
cuantificador("séptima","o7").
cuantificador("séptimo","o7").
cuantificador("octava","o8").
cuantificador("octavo","o8").
cuantificador("novena","o9").
cuantificador("noveno","o9").
cuantificador("décimo","o10").
cuantificador("décima","o10").

modificador("saliente","0").
modificador("entrante","+01").
modificador("antes","r-").
modificador("atrás","-").
modificador("después","r+").
modificador("actual","n").
modificador("corriente","0").
modificador("presente","0").
modificador("ya","P-").
modificador("fines","PF").
modificador("finales","PF").
modificador("final","F").
modificador("fin","F").
modificador("principios","PA").
modificador("principio","A").
modificador("comienzo","A").
modificador("comienzos","A").
modificador("inicios","PA").
modificador("inicio","A").
modificador("primeros","PA").
modificador("primeras","PA").
modificador("últimos","PF").
modificador("últimas","PF").
modificador("último","-").
modificador("última","-").
modificador("recientes","F-").
modificador("apenas","F").
modificador("sólo","").
modificador("casi","").
modificador("durante","").
modificador("para","").
modificador("siguiente","+").
modificador("siguientes","+").
modificador("nuevo","+").
modificador("nueva","+").
modificador("anterior","-").
modificador("posterior","+").
```

```
modificador("aquel","Or").
modificador("próximos","PO+").
modificador("próximo","O+").
modificador("próximas","PO+").
modificador("próxima","O+").
modificador("este","n").
modificador("esta","n").
modificador("estas","Pn").
modificador("estos","Pn").
modificador("transcurrido","-").
modificador("transcurridos","-").
modificador("transcurrida","-").
modificador("transcurridas","-").
modificador("anteriores","P-").
modificador("posteriores","P+").
modificador("ese","Or").
modificador("esa","Or").
modificador("esos","POr").
modificador("esas","POr").
modificador("aquel","Or").
modificador("aquella","Or").
modificador("aquellos","POr").
modificador("aquellas","POr").
modificador("pasado","O-").
modificador("pasada","O-").
modificador("pasadas","O-").
modificador("pasados","O-").

verbos("está","O+").
verbos("hace","O-").
verbos("hacen","O+").
verbos("hacía","IO-").
verbos("lleva","IO-").
verbos("llevaba","IO-").
verbos("llevaban","IO-").
verbos("llevará","IO+").
verbos("llevan","IO-").
verbos("llevo","O-").
verbos("faltan","O+").
verbos("dejará","O+").
verbos("fué","-").
verbos("dura","I+").
verbos("durará","I+").
verbos("duraron","Ir").
verbos("tardaron","Ir").
verbos("tardarán","IO+").
verbos("han","O-").
verbos("harán","O+").
verbos("tendrán","O+").
verbos("tienen","O+").
```

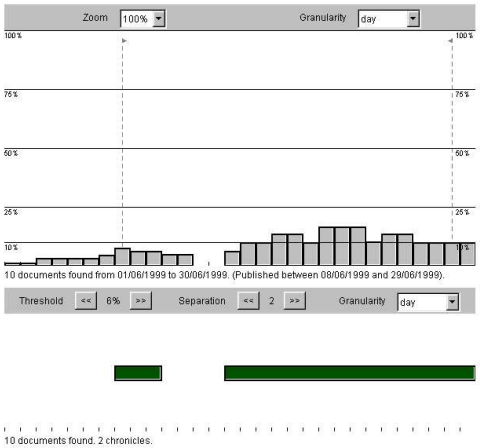
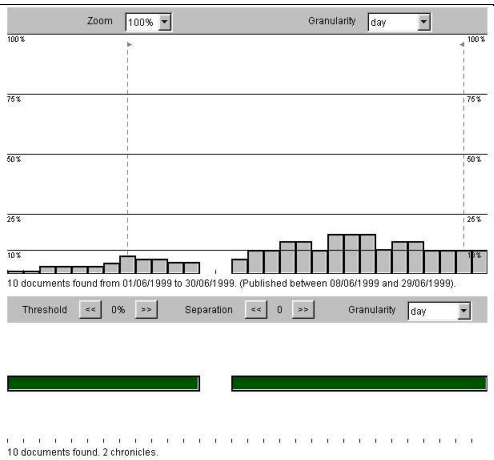
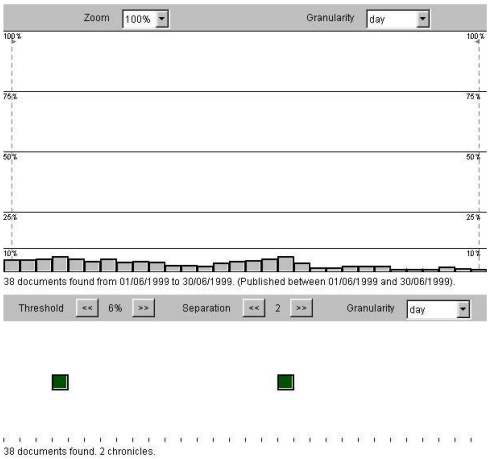
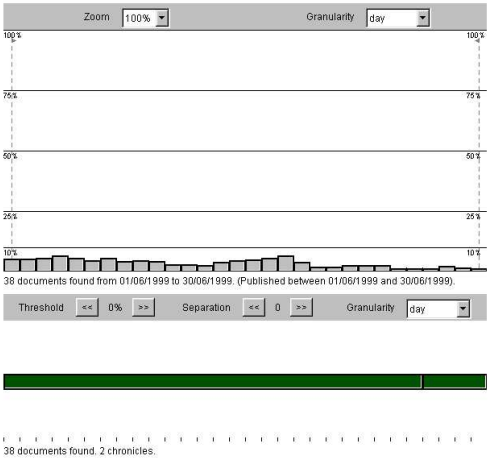
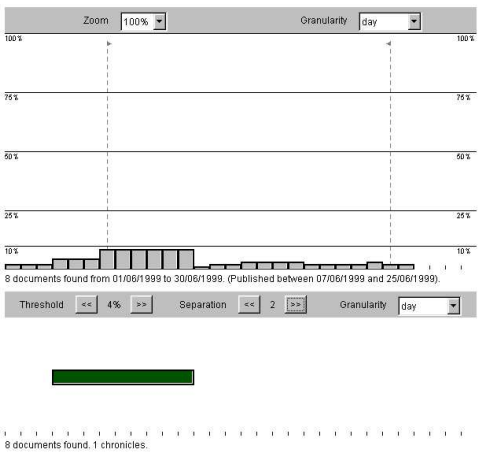
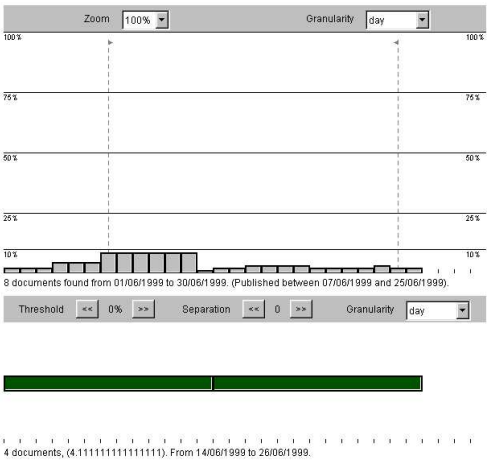
```
verbos("fueron","-r").
```

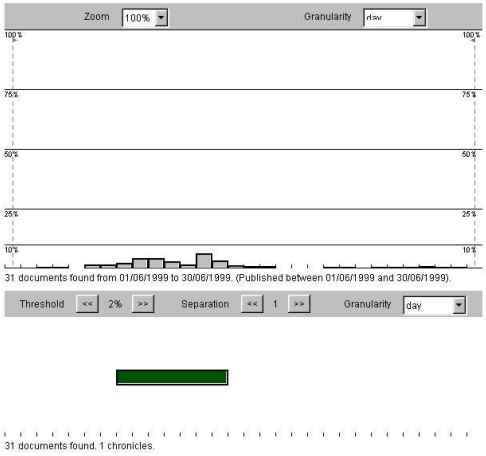
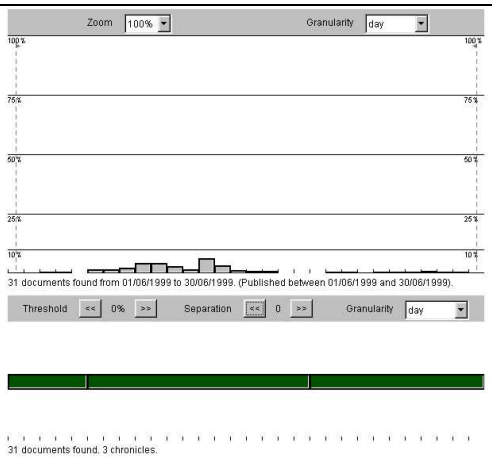
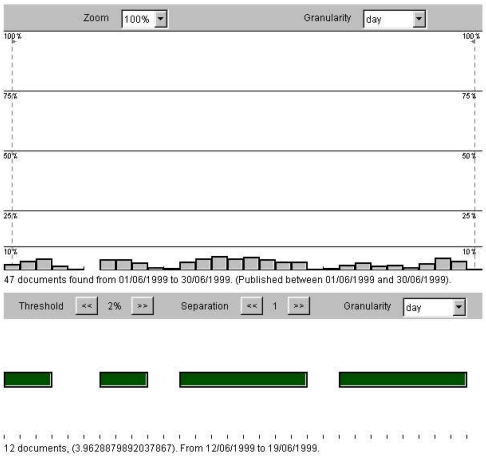
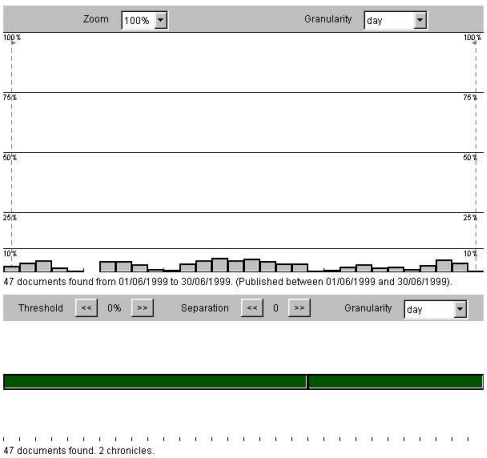
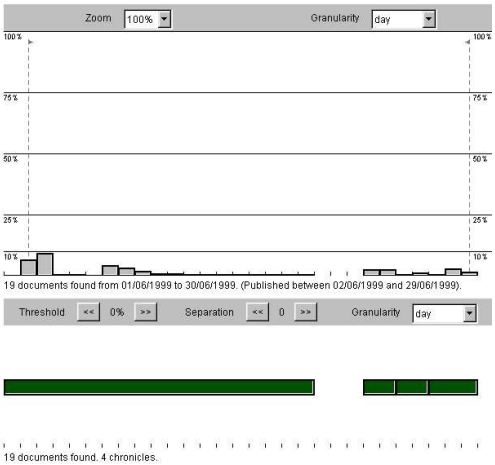
C. Palabras de la lista de parada

él	ésta	éstas	éste	éstos	última
últimas	último	últimos	a	añadió	aún
actualmente	adelante	además	afirmó	agregó	ahí
ahora	al	algún	algo	alguna	algunas
alguno	algunos	alrededor	ambos	ante	anterior
antes	aparece	aparecen	apenas	aproximadamente	aquella
aquellas	aquello	aquellos	aquí	así	aseguró
aunque	ayer	bajo	bien	buen	buena
buenas	bueno	buenos	cómo	cabe	caben
cabía	cabían	cada	casi	cerca	cierto
cierta	ciertamente	cinco	claro	claramente	comentó
como	con	confío	confía	confiar	conocer
consideró	considera	contra	cosas	cree	creemos
creer	creo	creó	cual	cuál	cuales
cualquier	cuando	cuanto	cuánto	cuatro	cuenta
da	dado	dan	dar	de	debe
deben	debió	debido	decir	dejó	del
demás	dentro	depende	desde	después	dice
dicen	dicho	dieron	diferente	diferentes	dijeron
dijo	dio	donde	dos	durante	ése
ésa	ésas	e	ejemplo	el	ella
ellas	ello	ellos	embargo	en	encuentra
entonces	entre	era	eran	es	esa
esas	ese	eso	esos	espero	esperamos
espera	está	están	esta	estaba	estaban
establece	establecen	estado	estamos	estar	estará
estas	éste	este	esto	estos	estoy
estuvo	ex	existe	existen	explicó	expresó
fin	final	finalmente	fue	fuera	fueron
gracias	gran	grandes	ha	había	habían
haber	habrá	hace	hacen	hacer	hacerlo
hacia	haciendo	han	has	hasta	hay
haya	he	hecho	hechos	hecha	hechas
hemos	hicieron	hizo	hoy	hubo	hubiera
igual	incluso	indicó	informó	inmediatamente	inmediato
inmediata	intenta	intentan	jamás	junto	la
lado	las	le	les	llega	llegan
llegó	lleva	llevar	lo	logró	los
luego	lugar	más	manera	manifestó	mayor
me	mediante	mejor	mencionó	menos	mi
mientras	misma	mismas	mismo	mismos	momento
mucha	muchas	mucho	muchos	muy	nada
nadie	ni	ningún	ninguna	ningunas	ninguno
ningunos	no	nos	nosotras	nosotros	nuestra
nuestras	nuestro	nuestros	nueva	nuevas	nuevo
nuevos	nunca	o	ocho	otra	otras
otro	otros	para	parece	parecen	parte
partir	pasada	pasado	pasando	pero	pesar
poca	pocas	poco	pocos	podemos	podrá
podrán	podría	podrían	poner	por	porque
posible	precisamente	próximo	próximos	primer	primera
primero	primeros	principalmente	propia	propias	propio
propios	pudo	pueda	puede	pueden	pues
qué	que	queda	quedan	quedó	queremos
quería	querías	quién	quien	quienes	quiere

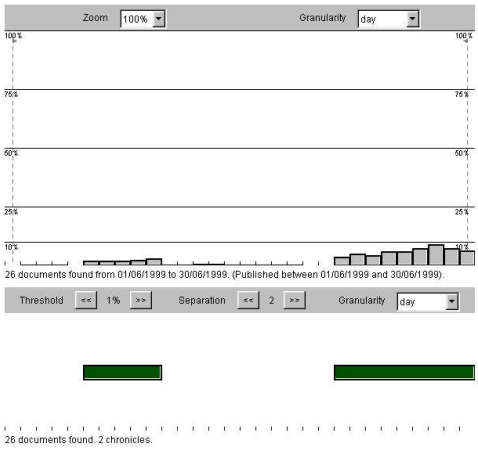
realizó	realizado	realizar	respecto	resulta	resultan
sí	sólo	sabe	saben	se	señaló
sea	sean	seguimos	según	segunda	segundo
seis	ser	será	serán	sería	si
sido	siempre	siendo	siete	sigue	siguiente
sin	sino	sobre	sola	solamente	solas
solo	solos	son	su	sus	tal
también	tampoco	tan	tanto	tenía	tenías
tendrá	tendrán	tenemos	tener	tenga	tengo
tenido	tercera	tiene	tienen	toda	todas
todavía	todo	todos	total	tras	trata
través	tres	tu	tú	tuvo	tuyo
un	una	unas	uno	unos	usted
va	vamos	van	varias	varios	ve
veces	ver	vez	voy	y	ya
yo	primavera	verano	otoño	invierno	Purísima
navidad	afelio	Carnaval	Cuaresma	antruejo	lunes
martes	miércoles	jueves	viernes	sábado	domingo
enero	febrero	marzo	abril	mayo	junio
julio	agosto	septiembre	octubre	noviembre	diciembre
fecha	festividad	fiesta	purificación	semana	víspera
periodo	trimestre	bimestre	cuatrimestre	semestre	cincuentenario
década	días	día	jornadas	jornada	año
aniversario	cuaresma	siglo	milenio	mes	hora
minuto	segundo	tiempo	periodo	ahora	antaño
anteayer	ayer	entonces	mañana	hoy	anoche
tarde	madrugada	pronto	inmediato	entonces	noche
mediodía	medianoche	hasta	al	cada	cada
cada	cualesquiera	cualquier	de	del	el
ene	las	la	los	lo	ningún
su	tal	tu	bastante	cualesquiera	cualquiera
a	al	aproximadamente	bastante	cada	cualesquier
cualesquiera	cualquier	cualquiera	de	del	el
en	hasta	la	las	lo	los
ningún	su	tal	anterior	antes	atrás
desde	hasta	pasada	pasado	último	dentro
después	posterior	próximo	próxima	siguiente	actual
corriente	esta	este	presente	durante	en
transcurrido	algún	alguna	alguno	cada	entre
final	finales	inicio	poca	poco	principio
tras	santo	mediados	ahora	pasada	par
cien	ciento	cincuenta	cuarenta	mil	sesenta
setenta	ochenta	noventa	una	uno	un
	dos	tres	cuatro	cinco	seis
siete	ocho	nueve	diez	once	doce
trece	catorce	quince	dieciséis	diecisiete	dieciocho
diecinueve	veinte	veintiún	veintidós	veintitrés	veinticuatro
veinticinco	veintiséis	veintisiete	veintiocho	veintinueve	treinta
primer	primera	primera	segunda	segundo	tercer
tercera	tercero	cuarto	cuarta	quinto	quinta
sexta	sexto	séptima	séptimo	octava	octavo
novena	noveno	décimo	décima	i	ii
iii	iv	v	vi	vii	viii
ix	x	xi	xii	xiii	xiv
xix	xviii	xv	xvi	xvii	xxi

D. Crónicas

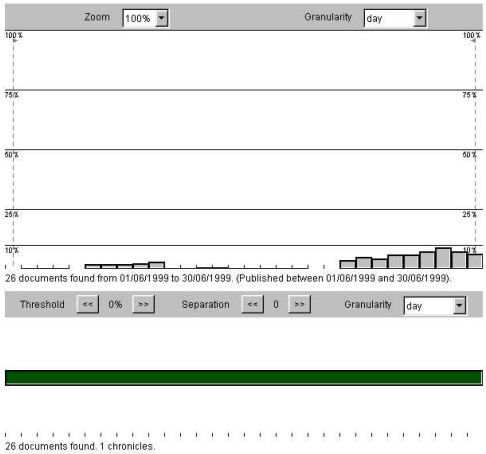
crónica	cronica-event
	
Tropas españolas en Kosovo	
	
Colombia	
	
Indonesia	

crónica	cronica-event
	
Elecciones Europeas	
	
México	
	
Narco	

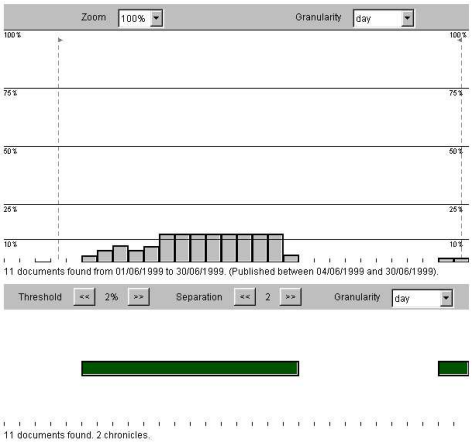
crónica

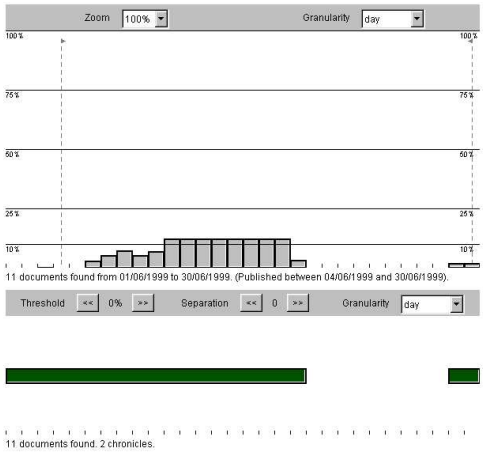


cronica-event



Ira





Papa

Pinochet

clase	F_{topic}	R_{topic}	P_{topic}	F_{et}	R_{et}	P_{et}	Query	F_{pd}	R_{pd}	P_{pd}	Query	F_{event}	R_{event}	P_{event}	Query	F_{notime}	$Rnotime$	P_{notime}	Query
0	0,3636	0,3636	0,3636	0,0909	0,0909	0,0909	Colombia	0,125	0,2	0,0909	Colombia	0,0455	0,0303	0,0909	Colombia	0,0417	0,027	0,0909	Colombia
1	0,8889	1	0,8	0,8889	1	0,8	Indonesia	0,75	1	0,6	Indonesia	0,8889	1	0,8	Indonesia	0,8889	1	0,8	Indonesia
2	0,8	1	0,6667									0,5714	0,5	0,6667	EU	0,1429	0,08	0,6667	Ira
3	0,5455	1	0,375	0,9412	0,8889	1	Papa	0,6667	1	0,5	Papa	0,8889	0,8	1	Papa	0,8421	0,7273	1	Papa
4	0,4	1	0,25	0,1739	0,1818	0,1667	Colombia	0,2353	0,4	0,1667	Colombia	0,5333	0,3636	1	Colombia	0,4898	0,3243	1	Colombia
5	1	1	1	0,25	0,1818	0,4	Colombia	0,2	0,2	0,2	Colombia	0,2222	0,1538	0,4	narco	0,1905	0,1081	0,8	Colombia
6	0,6667	1	0,5	0,1667	0,1	0,5	Ira	0,6667	1	0,5	Ira	0,0741	0,04	0,5	Ira	0,0741	0,04	0,5	Ira
7	0,75	1	0,6									0,2222	0,25	0,2	EU	0,0588	0,0345	0,2	EU
8	1	1	1	0,4615	0,3	1	México	0,6	0,4286	1	México	0,25	0,1538	0,6667	narco	0,1905	0,1111	0,6667	narco
9	0,8	0,6667	1	0,75	0,75	0,75	Pinochet	0,5	0,5	0,5	Pinochet	0,75	0,75	0,75	Pinochet	0,2581	0,1481	1	Pinochet
10	0,6667	1	0,5	0,8	0,6667	1	Pinochet	0,2857	0,3333	0,25	Pinochet	0,6154	0,4444	1	Pinochet	0,2581	0,1481	1	Pinochet
11	0,8	1	0,6667	0,2353	0,1429	0,6667	EU	0,3333	0,3333	0,3333	EU	0,2	0,1429	0,3333	EU	0,1875	0,1034	1	EU
12	0,5714	1	0,4	0,3333	0,2857	0,4	México	0,2	0,2	0,2	México	0,2	0,12	0,6	Ira	0,2	0,12	0,6	Ira
13	0,6667	1	0,5	0,25	0,1667	0,5	Pinochet	0,3333	0,25	0,5	Pinochet	0,1818	0,1111	0,5	Pinochet	0,069	0,037	0,5	Pinochet
14	0,4286	0,3333	0,6	0,2	0,2	0,2	kosovo	0,2857	0,5	0,2	Kosovo	0,2	0,2	0,2	kosovo	0,1333	0,1	0,2	kosovo
15	0,2857	1	0,1667	0,8462	0,7857	0,9167	EU	0,4	1	0,25	EU	0,7333	0,6111	0,9167	EU	0,5854	0,4138	1	EU
16	0,6	0,75	0,5	0,75	0,6	1	Colombia	0,1818	0,2	0,1667	Papa	0,3077	0,1818	1	Colombia	0,2791	0,1622	1	Colombia
17	0,4286	0,3333	0,6	0,2	0,2	0,2	kosovo	0,2857	0,5	0,2	Kosovo	0,2	0,2	0,2	kosovo	0,1333	0,1	0,2	kosovo
18	0,4	1	0,25									0,0909	0,0556	0,25	EU	0,0606	0,0345	0,25	EU
19	0,8	0,6667	1	0,75	0,75	0,75	Pinochet	0,5	0,5	0,5	Pinochet	0,75	0,75	0,75	Pinochet	0,2581	0,1481	1	Pinochet
20	0,6667	1	0,5									0,3333	0,25	0,5	Indonesia	0,3333	0,25	0,5	Indonesia
21	0,3333	0,3333	0,3333									0,0645	0,04	0,1667	Ira	0,0645	0,04	0,1667	Ira
22	0,5	1	0,3333									0,0833	0,0556	0,1667	EU	0,0571	0,0345	0,1667	EU
23	0,8	1	0,6667	0,4	0,25	1	México	0,5455	0,375	1	México	0,2069	0,1154	1	México	0,125	0,0667	1	México
24	0,5	1	0,3333	0,5455	0,6	0,5	kosovo	0,2857	1	0,1667	Kosovo	0,5455	0,6	0,5	kosovo	0,375	0,3	0,5	kosovo
25	0,8571	0,75	1									0,0714	0,04	0,3333	Ira	0,0714	0,04	0,3333	Ira
26	1	1	1									0,1	0,0556	0,5	EU	0,0645	0,0345	0,5	EU
27	0,6667	1	0,5	0,087	0,0526	0,25	México					0,087	0,0526	0,25	México	0,0408	0,0222	0,25	México
28	0,4615	0,375	0,6	0,2	0,2	0,2	Pinochet					0,1429	0,1111	0,2	Pinochet	0,0625	0,037	0,2	Pinochet
29	0,5455	1	0,375	0,9412	0,8889	1	Papa	0,6667	1	0,5	Papa	0,8889	0,8	1	Papa	0,8421	0,7273	1	Papa
30	1	1	1	0,4	0,25	1	México	0,3636	0,25	0,6667	México	0,2069	0,1154	1	México	0,125	0,0667	1	México
31	1	1	1	0,5714	0,4	1	Pinochet	0,8	0,6667	1	Pinochet	0,3636	0,2222	1	Pinochet	0,1379	0,0741	1	Pinochet
32	0,6667	1	0,5	0,125	0,0833	0,25	México					0,25	0,25	0,25	Indonesia	0,25	0,25	0,25	Indonesia
33	0,6667	1	0,5	0,8	0,6667	1	Pinochet	0,2857	0,3333	0,25	Pinochet	0,6154	0,4444	1	Pinochet	0,2581	0,1481	1	Pinochet
34	0,6667	1	0,5	0,1667	0,1	0,5	Colombia					0,0571	0,0303	0,5	Colombia	0,0513	0,027	0,5	Colombia
35	0,5	1	0,3333	0,1053	0,1	0,1111	Ira	0,1053	0,1	0,1111	Ira	0,0588	0,04	0,1111	Ira	0,0588	0,04	0,1111	Ira
36	0,8333	0,8333	0,8333	0,75	0,6	1	Ira	0,75	0,6	1	Ira	0,3871	0,24	1	Ira	0,3871	0,24	1	Ira
37	0,8889	1	0,8	0,8889	1	0,8	Indonesia	0,75	1	0,6	Indonesia	0,8889	1	0,8	Indonesia	0,8889	1	0,8	Indonesia
38	0,8571	1	0,75	0,2857	0,2	0,5	Colombia					0,2162	0,1212	1	Colombia	0,1951	0,1081	1	México
39	0,8333	0,8333	0,8333	0,75	0,6	1	Ira	0,75	0,6	1	Ira	0,3871	0,24	1	Ira	0,3871	0,24	1	Ira
40	0,8	1	0,6667	0,6	0,4286	1	México	0,75	0,6	1	México	0,25	0,1538	0,6667	narco	0,1905	0,1111	0,6667	narco
41	0,8	1	0,6667									0,4	0,2857	0,6667	EU	0,125	0,069	0,6667	EU
42	0,5	1	0,3333					0,1429	0,0909	0,3333	Pinochet	0,1429	0,0909	0,3333	Pinochet	0,0667	0,037	0,3333	Pinochet
43	0,8	1	0,6667									0,4	0,2857	0,6667	EU	0,125	0,069	0,6667	EU
44	0,4	1	0,25	0,4444	0,4	0,5	kosovo	0,4	1	0,25	Kosovo	0,4444	0,4	0,5	kosovo	0,2857	0,2	0,5	kosovo
45	0,5	0,8333	0,3571	0,7826	1	0,6429	Pinochet	0,72	0,8182	0,6429	Pinochet	0,72	0,8182	0,6429	Pinochet	0,439	0,3333	0,6429	Pinochet
46	0,4	1	0,25	0,4444	0,4	0,5	kosovo	0,4	1	0,25	Kosovo	0,4444	0,4	0,5	kosovo	0,2857	0,2	0,5	kosovo
47	1	1	1	0,1905	0,1053	1	México	0,3333	0,2	1	México	0,1905	0,1053	1	México	0,0851	0,0444	1	México
48	0,8	1	0,6667	0,6	0,4286	1	México	0,75	0,6	1	México	0,25	0,1538	0,6667	narco	0,1905	0,1111	0,6667	narco

clase	Titular
0	Clinton y Yeltsin discuten fórmulas para acelerar la negociación
1	Los indonesios acuden hoy a las urnas en busca de una ruptura radical con el pasado
2	Solana se perfila como favorito en la UE para el puesto de 'míster PESC'
3	El Papa tiene que suspender todos los actos en Cracovia por una gripe
4	En el país de la guerrilla
5	Los países de América discuten cómo hacer frente a las crisis financieras de la región
6	El IRA empieza a entregar cuerpos de desaparecidos en el conflicto irlandés
7	Mbeki se queda a un solo escaño de los dos tercios del Parlamento surafricano
8	Conmoción en México por el asesinato a tiros de un famoso presentador
9	El Gobierno chileno dice que Pinochet debe volver porque hay garantías de que será juzgado
10	Inquietud entre los generales chilenos por los nuevos sumarios contra cargos de la dictadura
11	Schröder y Blair presentan un manifiesto para la modernización de la izquierda
12	América Latina busca en la UE la salida a las crisis financieras cíclicas
13	Chile usa a España contra un español cuyos bienes expropió Pinochet
14	La Iglesia, el Ejército y los políticos piden a los serbios de Kosovo que no huyan de la provincia
15	Schröder ofrece un plan de reformas fiscales como respuesta a su sonada derrota electoral
16	La guerrilla colombiana pone en libertad a 33 de los rehenes secuestrados en una iglesia
17	La Iglesia, el Ejército y los políticos piden a los serbios de Kosovo que no huyan de la provincia
18	El G-8 aprueba el plan de reconstrucción para los Balcanes
19	El Gobierno chileno dice que Pinochet debe volver porque hay garantías de que será juzgado
20	América Central espera este año nueve huracanes similares a 'Mitch'
21	Desbandada serbia en Kosovo
22	Rusia se compromete a no enviar más tropas a Kosovo mientras no haya acuerdo con la OTAN
23	La vuelta de Salinas solivianta a México
24	Una avanzadilla de 50 militares españoles viaja esta semana a la zona
25	Mbeki deja a Winnie Mandela fuera del nuevo Gobierno de Suráfrica
26	La 'izquierda plural' francesa sale reforzada en las urnas
27	El presidente Chávez decreta la enseñanza militar de niños y jóvenes en Venezuela
28	Los guerrilleros del ELK intenta llenar el vacío dejado en Kosovo ante la pasividad de la Kfor
29	El Papa tiene que suspender todos los actos en Cracovia por una gripe
30	Un fuerte terremoto sacude México y causa al menos 14 muertos y 200 heridos
31	Amnistía Internacional destaca el 'caso Pinochet' como hito de los derechos humanos
32	El perdón de la deuda externa no es suficiente, según la Iglesia
33	Inquietud entre los generales chilenos por los nuevos sumarios contra cargos de la dictadura
34	Un cadáver, clave para esclarecer el 'caso Gerardi' en Guatemala
35	Nueva asignatura para Belgrado
36	Blair pide al Sinn Fein un compromiso de entrega de armas en el año 2000
37	Los indonesios acuden hoy a las urnas en busca de una ruptura radical con el pasado
38	La guerrilla de Colombia ataca por sorpresa el cuartel general de los paramilitares
39	Blair pide al Sinn Fein un compromiso de entrega de armas en el año 2000
40	La oposición pide al presidente mexicano que diga si hubo dinero negro en su campaña
41	España presenta una lista de seis carteras de la UE para De Palacio y Solbes
42	La Internacional Socialista apoya los cambios de gobierno en el Cono Sur
43	España presenta una lista de seis carteras de la UE para De Palacio y Solbes
44	El primer contingente de tropas españolas llega a Kosovo e instala su cuartel general en Istok
45	Aznar no cree posible que los Reyes adelanten su viaje a Cuba
46	El primer contingente de tropas españolas llega a Kosovo e instala su cuartel general en Istok
47	Una investigación culpa a un reducido grupo de militares de la matanza de Tlatelolco
48	La oposición pide al presidente mexicano que diga si hubo dinero negro en su campaña

Cuadro 1: Titular del primer documento de las clases teóricas

E. Programa CodTimex

```
 #! python2.0
''' Program que genera indices,
indice de referencias(lartidb)codigo=[scode,titulo,ambito,fechas]
indice invertido de fechas(tidb)fecha=[codigo-articulos donde aparece(repetidos)]
indice invertido de palabras(pidb=) clave= [codigo ar,frecuencia]
lartir=bsddb.btopen(d+'referencias.db','c')
'''

import os, bsddb,re,calendar,glob,sys
from DocyCluster import Documento
from os import popen
from Alib_fechas import espalabra,convminus,solapar,cargafich,CS,sumagra,fecha_standar,DistanciaDias
from string import joinfields,find,split,upper
from math import sqrt
import DepuraFechas
from Asimi_ambitos import *

MAY='[A-ZÁÉÍÓÚÑ]'
ALFA='[A-Za-zÑáÁéÉíÍóÓúÚ]'

def palinlista(pal,lista):
       esta=0
       for p in lista:
              if re.search('^'+pal+'.?$',p):
                  esta=p
                  break
       return esta

def CargaDocumentoAurora(Doc,Codigo,lpal,SCodeF,Lugar,Titulo,CadFechas):
# Carga las variables del documento en la clase de Aurora
       Doc.Codigo = Codigo
       Doc.VectorFrec.update(lpal)
       Doc.SCode = SCodeF[8:]
       Doc.FechadePublicacion = SCodeF[0:8]
       if CadFechas[0]=="#":
           CadFechas=CadFechas[1:]
       Doc.Tiempo = Doc._Documento__CargaFechas(CadFechas)
       DepuraFechas.DepuraFechas(Doc.Tiempo, Doc.FechadePublicacion)
       Doc.Lugar = Lugar
       Doc.Titulo = Titulo
       Doc.Tematica = Doc.Codigo[:1]
def buscaclase(clases,valor):
    k=0
    while k <len(clases):
        if valor in clases[k]: return k
        k+=1
    return -1

def extraerpal(J,abstract,lipalabras):
    """ El texto J, devuelve una lista con las  palabras(extrae similitudes con tacat)
    que no estan en la lista de parada(variable global parada) y el numero de veces que aparece(tf),
    además del n° palabras y la suma de tf^2
    relevancia: nos permite desechar las palabras que aparecen menos del max/relevancia,
              o menos que el mínimo, si el max=min o relevancia =0 se toman todas
    abstract: tomamos solo las palabras que aparecen en frases con TIMEX si vale 1
    """
    global  parada
    pal_fech=''
    if abstract=='1':
      X=re.search('((<TIMEX){1}(.*?)(</TIMEX.*?>){1})',J)
      if X:
    frases=re.split('\n|\r',J)
    j=0
    for i in frases:
          if i:
            j+=1
            if j<5 and  not re.search('^\s*\S+\s*\S*\s*\S*\s*$',i):
                 pal_fech+=' '+i
            else:
               X=re.search('(<TIMEX){1}(.*?)(</TIMEX.*?>){1}',i)
               if X:
                   pal_fech+=' '+i
                   # Genera un fichero con fechas y palabras en mayúsculas
          if len(pal_fech)>100 :
            J=pal_fech
    J=re.sub('(<TIMEX){1}(.*?)(</TIMEX.*?>){1}',' ',J)
    J=re.sub('<.+?>',' ',J)
    P=re.split('(\n|\r|\.|\'|(-\s+))+',J)
    TPalabras=[]
    j=0 #para dar mas relevancia titular y resumen
    for  frases in P:
     if frases:
```

```
    frases=re.sub(',|\(|\)|:|;|\[|\]|%|\?|\¿|(\s+-)|(-\s+)|\!|\¡',' ',frases)
    frases=re.sub('^\s+','',frases)
    frases=re.sub('\s+',' ',frases)
    frases=re.sub('\s+$','',frases)
    if len(frases)>0:
        if  re.search('^\s*\S+\s*\S*\s*\S*\s*$',frases):
            j=10 #agui comienza el cuerpo de la noticia
            continue
        i1=frases.split()
        if len(i1)<0 : raw_input(frases)
        else : # paso a minúscula la primera palabra de cada frase
            i1[0]=convminus(i1[0])
        TPalabras+=i1
        j+=1

Vector = {}  #diccionario que guarda las palabras(clave) con su frecuencia(valor)
if len(TPalabras)==0:
    return [{},0]
if parada==[]:
    parada=cargafich('listaparada.txt')
Buscamos = []
for palabra in TPalabras:
  if len(palabra)<2 :continue
  palabra=re.sub('-$','',palabra)
  palabra=re.sub('^-','',palabra)
  palmin=convminus(palabra)
  i=1
  #doy el doble de relevancia a las palabras en mayúsculas
  if palmin=="":
      if not re.search('-',palabra):continue
      else:
             palmin=palabra
  if (palmin in Vector.keys()):
      Vector[palmin]+=i
  elif re.search('-',palmin):
         if not re.search('\d+-\d+',palmin):
                Vector[palmin]=i
         p=palmin.split('-')
         for p0 in p:
                if len(p0)>3 and not re.search('\d+',p0):
                       if p0 in Vector.keys():
                              Vector[p0]+=i
                       elif not (p0 in parada):
                              Vector[p0]=i
                              Buscamos.append(p0)
  elif (palmin not in parada):
  Vector[palmin]=i
  #si no es una palabra en maýuscula o nombre própio o no contiene '-'
  if (palmin not in lipalabras.keys()) and palmin==palabra and (palabra not in Buscamos):
      Buscamos.append(palabra)
P=popen('buscarpalabras.pl \''+ joinfields(Buscamos,' ')+'\'','r')
P=P.readlines()
xx=[]
xy=0
for pos in range(len(P)):
     palabra=Buscamos[pos]
     if len(palabra)==0 or len(P[pos])<2 :
            continue
     Acepciones=' '+P[pos]+' '
     Stem = palabra
     ProbarStem = re.search('\s+(\S+r)\s+V',Acepciones)
     esta=0
     #damos preferencia a los verbos (codigo V)
     if ProbarStem :
         Stem = ProbarStem.group(1)
         esta=1
     else:
         ProbarStem = re.search('\s+'+palabra+'{1}\s+',Acepciones)
         if (not ProbarStem) and (len(palabra)>3):
           try:
             Acepciones=Acepciones.split(' ')
             for k in range(1,len(Acepciones),2):
                 if find(Acepciones[k],palabra[0:len(palabra)-2])<>-1:
                      Stem=Acepciones[k]
                      break
             if Stem==palabra: Stem=Acepciones[1]
           except:
                  print 'error',Acepciones,k.groups()
                  raw_input()
                  pass
     #aumento el contador de los lemas en el indice
     palabra=convminus(palabra)
     if convminus(Stem)==palabra and esta<>1 and len(palabra)>3:
       p=palinlista(palabra[0:len(palabra)-1],  Vector.keys())
       if p<>0: Stem=p
       elif len(palabra)>6:
```

```
                Stem=palabra[0:len(palabra)-1]
                esta=0
                while len(Stem)>5 and esta==0:
                  Stem=  Stem[0:len(Stem)-1]
                  p=palinlista(Stem, Vector.keys())
                  if p<>0:
                         Stem=p
                         esta=1
                if esta==0: Stem=palabra
          if convminus(Stem)<>convminus(palabra):
            if Stem in Vector.keys():
              Vector[Stem]=Vector[Stem]+Vector[palabra]
            else:
              Vector[convminus(Stem)]=Vector[palabra]
            del(Vector[palabra])
    w2=0
    npalabras=float(len(Vector.keys()))
    for i in Vector.keys():
            w2+=Vector[i]
    return [Vector,w2]

def singlepassaurora(clasesA,Doc,cod,umbral,umbralambito,scodeF):
    max=umbral
    c=-1
    z=0
    for i in clasesA.keys():
        [RClas,a]=clasesA[i]
        z=RClas.Similaridad(Doc,'Medida3',0.5,0.5,0,float(umbralambito),float(umbral))
        if z>=max:
                max=z
                c=i
    if c==-1  :
                DocClas = Documento()
                DocClas.VectorFrec.update(Doc.VectorFrec)
                DocClas.SCode = Doc.SCode
                DocClas.FechadePublicacion = DocClas.FechadePublicacion[:]
                DocClas.Tiempo.update(Doc.Tiempo)
                DocClas.Lugar=Doc.Lugar
                DocClas.Titulo=Doc.Titulo
                Doc.Tematica=Doc.Tematica[:]
                clasesA[str(len(clasesA))]=[DocClas,[cod]]

    else:
        [RClas,a]=clasesA[c]
        RClas.VectorFrec=recalcularclase(RClas.VectorFrec,Doc.VectorFrec,len(a))
        RClas.Tiempo=recalcularclase(RClas.Tiempo,Doc.Tiempo,len(a))
        clasesA[c]=[RClas,a+[cod]]

def singlepass(clases,lpalabras,cod,ambito,umbral,umbralambito,time,scodeF):
    """
        Calcula clases con el metodo de single pas
        time==1 con fechas
        recal= max, min o suma (0,1,2)para recalculo de clases
    """
    FECHA=scodeF[0:8]
    lpal={}
    lpal.update(lpalabras)
    z=0
    if len(lpal)==0: return
    c=-1
    q=0
    max=umbral
    FI=ambito.split("-")
    if len(FI)<2:
        FF=FECHA
        FI=FECHA
    else:
        FF=FI[1]
        FI=FI[0]
    if len(clases)>0 :
      for i in clases.keys():
        l=clases[i][2][:]
        z1=0
        if int(time) ==0 :
           z=coseno(clases[i][0],lpal)
           if z>=max:
               max=z
               c=i
               fech=1
        else:
              z0=SimilEventTime(clases[i][2][0],clases[i][2][1],FI,FF)
              if z0 >=umbralambito :
                  z=coseno(clases[i][0],lpal)
              if round(z,5)>=round(max,5):
                 if z<=max+0.0003:
                      if z0>z1:
```

```
                        c=i
                        fech=l
                        q=0
                        max=z
                elif z>max:
                       c=i
                      fech=l
                      q=0
                      max=z
            if z0>z1: z1=z0
      if c==-1 :
              clases[str(len(clases))]=[lpal,[cod],[FI,FF]]
              clases[str(len(clases))]=[lpal,[cod],[FI,FF]]
      else:
                  j=re.search('desp\S*?\((\d+)\)',scodeF)
                  z,cod1,cod0,i=0,0,0,0

                  if j:
                     j=int(j.group(1))
                     cod0=cod[0:1]+str(int(cod[1:])-j)
                     if j>1: cod1=cod[0:1]+str(int(cod[1:])-1)
                     else: cod1=0
                     A=clases.keys()
                     A.reverse()
                     for i in A:
                       if cod0 in clases[i][1] or cod1 in clases[i][1] :
                          z=coseno(clases[i][0],lpal)

                  if z> 0.17:
                            c=i
                            clases[c][1]+=[cod]
                            clases[c][0]=recalcularclase(clases[c][0],lpal,len(clases[c][1]))
                            break
                           elif cod1==0: break
                  if c==-1:
                         clases[str(len(clases))]=[lpal,[cod],[FI,FF]]
    else:
        clases[c][0]=recalcularclase(clases[c][0],lpal,len(clases[c][1]))
        if int(fech[0])<int(FI): FI=fech[0]
        if int(fech[1])>int(FF): FF=fech[1]
        clases[c][1]+=[cod]
        clases[c][2]=[FI,FF]

def tomarNMAXdiccionario(diccionario):
# toma de un diccionario los 30 valores más altos
    L=diccionario.values()
    L1=diccionario.values()
    L2=diccionario.keys()
    lpal=[]
    L.sort()
    L.reverse()
    lpal={}
    if len(L)>30:
      for i in L[0:30]:
        a=L1.index(i)
        lpal[L2[a]]=i
        L1.pop(a)
        L2.pop(a)
    else: lpal=diccionario
    return lpal

def maxim(a,b):
    if a>b: return a
    else: return b

def recalcularclase(reclase,a_pal,n):
#toma las palabras de la clase y del artículo
#y recalcula el representante con la media, suma o maximo segun el valor recal =2,1,0
    clasea={}
    clasea.update(reclase)
    for j in clasea.keys():
         clasea[j]=clasea[j]*n
    for j in a_pal.keys():
    if clasea.has_key(j):clasea[j]=clasea[j]+a_pal[j]
    else: clasea[j]=a_pal[j]
    for j in clasea.keys():
         clasea[j]=clasea[j]/(n+1)
    return clasea

def coseno(c_pal,a_pal):
   """
     c_pal= palabras de la clase con los pesos
     a_pal= palabras del articulo con los pesos
     a_w=suma tf cuadrados
   """
   z=0.0
```

```
   c_w=0.0
   a_w=0.0
   f=0.0
   for j in a_pal.keys():
       a_w+=a_pal[j]**2
   for i in c_pal.keys():
       c_w+=c_pal[i]**2
       if a_pal.has_key(i):
           z+=c_pal[i]*a_pal[i]
   if a_w==0 or c_w==0:

       print "cos",c_pal, a_pal,"\n",a_w,c_w
       raw_input("cos")
       z=0
   else: z=z/sqrt(a_w*c_w)
   return z

def similitudl(e,larti,lipalabras,lpal,cod,ambito,w,p_2):
    """Calcula la similitud entre un documento caracterizado por una lista de palabras que incluye la frecuencia
    y  la Bd de documentos 'db', siendo cod la etiqueta que caracteriza el articulo
    y simil es la lista de similitudes entre documentos
    e=directorio ficheros
    """
    global clases,clases1,lifechas
    foutf2=open (e+'similitudes.tx','a')
    if larti=={}:
    clases+=[[cod]]
    return
    X=''
    j=''
    b=''
    a=''
    K=[] # lista de pares de documentos con posible similitud
    #simil solo tiene aquellos pares de documentos que se solapan en el ambito
    #si está el par en K y no en simil es que no se solapan
    # si no esta en K-> veo si se solapan y si es asi lo añado a simil
    simil={}
    s1=cod[0]#seccio del articulo
    i=ambito.split('-')
    FI=''
    FF=''
    if len(i)==2:
        FI=i[0]
        FF=i[1]
        if 1990<int(FF[0:4])<2030:
            A=calendar.weekday(int(FF[0:4]),int(FF[4:6]),int(FF[6:8]))
            if 6>A>=4:
                FF= sumagra(FF,'d','+'+str(6-A))
    # como no tenemos fines de semana ampliamos los extremos de viernes o sabado
    #weekday: lunes =0, domingo=6
    if int(FF)==int(FI):
        FI=sumagra(FF,'d','-1')
    max=0
    maxart=''
    for i in lpal.keys():
        if i in lipalabras.keys():
            for c in lipalabras[i]:
                if s1==c[0][0]: # misma seccio del periodico
                    z=c[1]*lpal[i]
                    codsim=c[0]
                    j=codsim+'#'+cod
                if j  in simil.keys():
                          simil[j][0]=simil[j][0]+z
                elif  (j not in K) :
                          K=K+[j]
                          l=larti[codsim][2].split('-')
                          if len(l)==2:
                            if l[0]==l[1]: l[0]=sumagra(l[1],'d','-1')
                                 l[1]=fecha_standar(l[1])
                            if solapar(l[0],l[1],FI,FF):
                                 simil[j]=[z,c[2]]
    j=''
    lista=[]
    for i in simil.keys():
    z=simil[i][0]/(sqrt(simil[i][1])*sqrt(p_2))
        a=i.split('#')
    if z>max:
        max=z
        maxart=a[0]
    if z>0.278:
        j=j+'\n'+i+'#'+str(z)
        if z>0.37: lista.append(a[0])
    if z>=CS:
            l=0
            a=i.split('#')
            for k in range(len(clases1)):
```

```
                if  a[1] in clases1[k] :
                    l=1
                    if  (a[0] not in clases1[k]) :
                        clases1[k]=clases1[k]+[a[0]]
            break
                elif  a[0] in clases1[k] :
                    l=1
            if  (a[1] not in clases1[k]) :
                        clases1[k]=clases1[k]+[a[1]]
                    break
            if l==0:
                clases1=clases1+[[a[0],a[1]]]
    l=0
    if max>=0.28:
    k=buscaclase(clases,maxart)
        l=1
        clases[k]=clases[k]+[cod]
    if l==0:    clases=clases+[[cod]]
    for i in lista:
    k1=buscaclase(clases,i)
    if k1<>k:
        clases[k]=clases[k]+clases[k1]
        clases.pop(k1)
        if k>k1: k=k-1
    foutf2.write(j)
    foutf2.close()
    return j

def indicei(listai,claves,cod,w2):
    for i in claves.keys():
        tf=claves[i]
        if i in listai.keys()  :
            listai[i]=listai[i]+[[cod,tf,w2]]
        else:
            listai[i]=[[cod,tf,w2]]

def anyadirlista(l,pal):
    for i in pal.keys():
        if i in l.keys():l[i]=str(float(l[i])+pal[i])
        else:l[i]=str(pal[i])

def analizar(lpalabras,ambito,cod,scodef,titular,fechas,lugar):
    # Asigna a cada nuevo documento cod una clase
    #Vpalabras es una diccionario con las palabras del documento y su frecuencia relativa
    #lclases es una lista de listas de clases según un umbral
    #recal es unalista de enteros positivos 0,1,2
    global lclasesA,lclases,relevancia,threshold,time,abstract,aurora
    jj=0
    Vpalabras={}
    Vpalabras.update(lpalabras)
    #normalizo el vector
    m=0
    for i in Vpalabras.keys():
            m+=Vpalabras[i]
    if m==0: print Vpalabras,cod,'00'
    for i in Vpalabras.keys():
             Vpalabras[i]=float(Vpalabras[i])/m
    for k in relevancia.split(','):
      lpal={}
      lpal.update(Vpalabras)
      if aurora== 1:
             Doc=Documento()
             CargaDocumentoAurora(Doc,cod,lpal,scodef,lugar,titular,fechas)
      for i in threshold.split(','):
        if aurora==1:
            singlepassaurora(lclasesA[jj],Doc,cod,float(i),float(k),scodef)
            #donde k es el umbral de relevancia de los ambitos
        else:
          singlepass(lclases[jj],lpal,cod,ambito,float(i),float(k),time,scodef)
        jj+=1

def analizardesp(Vpalabras0,Vpalabras,ambito,cod,scodef,titular,fechas,lugar):
    # Asigna a cada nuevo documento cod una clase
    #Vpalabras es una diccionario con las palabras del documento y su frecuencia relativa
    #lclases es una lista de listas de clases según un umbral
    #recal es unalista de enteros positivos 0,1,2
    global lclasesA,lclases,relevancia,threshold,time,abstract,aurora
    jj=0
    m=0
    for i in Vpalabras0.keys():
            m+=Vpalabras0[i]
    if m==0: print Vpalabras0,cod,'00'
    for i in Vpalabras0.keys():
             Vpalabras0[i]=float(Vpalabras0[i])/m
    m=0
    for i in Vpalabras.keys():
```

```
            m+=Vpalabras[i]
    if m==0: print Vpalabras,cod,'00'
    for i in Vpalabras.keys():
             Vpalabras[i]=float(Vpalabras[i])/m
    z=coseno(Vpalabras0,Vpalabras)
    if j:
       j=int(j.group(1))"""
    cod0=cod[0:1]+str(int(cod[1:])-1)
    l=0
    for k in relevancia.split(','):
       for i in threshold.split(','):
         if float(z)>= 0.12:#float(i):

           for i1 in lclases[l].keys():
               if cod in lclases[l][i1][1]:
                 print '2',lclases[l][i1][1], i,cod,cod0
                 if len(lclases[l][i1][1])==1:
                     print '33'
                     del(lclases[l][i1])

                     for i1 in lclases[l].keys():
                       if cod0 in lclases[l][i1][1]:
                            lclases[l][i1][1]+=[cod]
                            print '1',cod,cod0,j

         else: print i,z,cod,cod0

         l+=1

def analizarperiodicos(e,e0,e1):
   global SEC
   larti={}
   lipalabras={}
   lartipal={}

   if e0=='0' :
          Q=glob.glob(e+'199906*.txt')+glob.glob(e+'199907*.txt')+glob.glob(e+'199905*.txt')
   else:
          Q=glob.glob(e+'1999'+e0+'*.txt')

   foutf1=open (e1+'coberturas.tx','w')
   faurora=open (e1+'arti.txt','w')
   Q.sort()
   for q in Q:
        print q,len(larti)
        FECHA=''
        LDADES=['','','']
        LFECHAS=[]
        dicnref={}
        dicnref[0]=0
        dicnref[1]=0
        dicnamb={}
        clasificacion={'-3':0,'-2':0,'-1':0,'0':0,'1':0,'2':0,'3':0,'4':0}
        Z='-' # Almacena el titular ,variable indica si es ='' indica que lo siguiente que se lea es el titular
        code='' #scode
        arti='' #fecha Scode
        coda=''  #almacena arti anterior
        texto=''
        lugar='-'
        LDADES=['','','']
        m=0 #num doc
        n=0 # numero de doc sin fechas
        f=open (q,'r')
        J=f.readline()
        lugar='-'
        J0=''
        ultima=0# variable para que coja el último artículo
        while J or ultima==0:
          if not J and ultima==0:
                ultima=1
                J=J0
          if Z=='':
            Z=J
            Z=Z.strip()
          elif lugar=='':lugar=J.strip()
          if len(J)<2 or re.search('TEXTEND',J):
                J=f.readline()
                continue
          elif re.search('<TEXTSTART.*\.(snam)\(',J): #nuevo artículo
                   J=f.readline()
                   while J and not re.search('TEXTEND',J):
                       J=f.readline()
                   continue
          elif re.search('<TEXTSTART',J):
            X=re.search('SCODE\s+=\s*\'(\S*?date\S*?)\',\s+FECHA\s+=\s*\'(\d{8})\'',J)
            if X: FECHA=X.group(2)
```

```
            else:
              X=re.search('SCODE\s+=\s*\'(\S*?plac\S*?)\',\s+FECHA\s+=\s*\'(\d{8})\'',J)
              if X:   lugar=''
              else:
        X=re.search('SCODE\s+=\s*\'(\S*?titl\S*?),\s+FECHA\s+=\s*\'(\d{8})\'',J)
        if X:
           J0=J
           m=m+1
           s1=''
           arti=J
           #arti=X.group(2)+'_'+X.group(1)
           s0=re.search('sect\((\d){1}\)',coda)
           if s0:
                       s1=Secciones[int(s0.group(1))]
                  X= re.search('SCODE\s+=\s*\'(\S*?titl\S*?)\',\s+FECHA\s+=\s*\'(\d{8})\'\s+VALUEAMBITO\s+=\s*([^>\s]*)\s*VALUEFECHAS\s*=\s*([^>]*)',coda) #nuevo ar

                  if X and s1==SEC:
                     ambito=X.group(3)
                     fechas=X.group(4)
                     #fechas son les dates que apareixen en el text amb la frecuencia
              SCODEF=X.group(2)+'_'+X.group(1)

              if ambito=='':

                        a=sumagra(FECHA,'d','-1')
                        if re.search(a,fechas): ambito=a+'-'+FECHA
                        else:
                           a=sumagra(FECHA,'d','+1')
                           if re.search(a,fechas): ambito=FECHA+'-'+a
                if ambito=='': ambito=FECHA+'-'+FECHA
              K={}
              [K,w2]=extraerpal(texto,abstract,lipalabras)
              codarti=s1+str(len(larti))

                      lartipal[codarti]=K
              analizar(K,ambito,codarti,SCODEF,Z,fechas,lugar)
              #he añadido w2 20020108
              faurora.write('\n'+codarti+'|'+SCODEF+'|'+str(w2)+'|'+str(ambito)+'|'+Z+'|'+lugar+'|'+fechas)
              larti[codarti]=[SCODEF,Z,ambito,fechas,lugar]
              for i in K.keys():
                            K[i]=float(K[i])/w2
                      indicei(lipalabras,K,codarti,w2)
                  Z=''
                  lugar='-'
                  texto=''
                  coda=arti
                  LDADES=['','','']
           else:
                 texto=texto+"\n"+J
           J=f.readline()
           foutf1.write('\n'+str(clasificacion)+':'+str(dicnref[0])+'--'+str(dicnref[1])+':'+str(dicnamb)+':total: '+str(m)+': nofech:'+str(n))
        f.close()
    foutf1.close()
    faurora.close()
    farti=bsddb.hashopen(e1+'larti.db','c')
    fartipal=bsddb.hashopen(e1+'lpal.db','c')
    fipalabras=bsddb.hashopen(e1+'pia.db','c')
    #fifechas=bsddb.hashopen(e+'fi.db','c')
    for i in larti.keys():
            farti[str(i)]=''
            for j in larti[i]:
                farti[str(i)]+= '|'+str(j)
    for i in lartipal.keys():
            fartipal[str(i)]=''
            for j in lartipal[i].keys():
                fartipal[str(i)]+='|'+j+' '+str(lartipal[i][j])[0:8]
    for i in lipalabras.keys():
            fipalabras[str(i)]=''
            for j in lipalabras[i]:
                fipalabras[str(i)]+='|'+j[0]+' '+str(j[1])[0:8]
    farti.close()
    fartipal.close()
    fipalabras.close()

def analizadb(e):
    global abstract
    farti=bsddb.hashopen(e+'larti.db','r')
    fartipal=bsddb.hashopen(e+'lpal.db','r')
    if abstract=='2':
               fartipalt=bsddb.hashopen(e+'06lpal.db','r')
    c=''
    SCODEF=''
    Z=''
    ambito=''
    fechas=''
    lugar=''
```

```
   codarti=''
   Vpalabras={}
   for pp in range(0,len(farti.keys())):
       codarti='i'+str(pp)
       b=farti[codarti].split('|')
       if len(b)==6: [c,SCODEF,Z,ambito,fechas,lugar]=b
       else: raw_input(b)
       lpal=[]
       #if not re.search('desp',SCODEF):Vpalabras0=Vpalabras
       Vpalabras={}
       lpal=fartipal[codarti].split('|')
       for i in lpal:
              if len(i)<2:continue
              p=i.split()
              Vpalabras[p[0]]=float(p[1])
       if abstract=='2':
         lpalt=fartipalt[codarti].split('|')
         for i in lpalt:
                if len(i)<2:continue
                p=i.split()
                if Vpalabras.has_key(p[0]):Vpalabras[p[0]]=float(p[1])+Vpalabras[p[0]]
                else: Vpalabras[p[0]]=float(p[1])

       analizar(Vpalabras,ambito,codarti,SCODEF,Z,fechas,lugar)
   farti.close()
   fartipal.close()

def creaficheroclases(clases,times,k,i0,k1,e):
       global farti
       foutf2=open (e+'E'+times+'-'+str(k)+'-'+str(i0)+'-similitudes.txt','w')
       grupo={}
       grupo.update(clases[k1])
       print len(grupo)
       for i in range(len(grupo.keys())):

            i=str(i)
            p=len(grupo[i][1])
            if len(grupo[i])==3:
                foutf2.write('\n   CLASE'+i+' '+ str(grupo[i][2]))
            else:
                foutf2.write('\n   CLASE'+i)
            for j in grupo[i][1]:
               foutf2.write( '\n'+str(j)+'; '+str(i)+'; '+str(p)+'; '+str(farti[j]))
       foutf2.close()

def creaficheroclases2(clases,times,k,i0,k1,e):
       global farti
       foutf2=open (e+'CLAS'+times+'-'+str(k)+'-'+str(i0)+'-similitudes.txt','w')
       grupo={}
       grupo.update(clases[k1])
       print len(grupo)
       for i in range(len(grupo.keys())):
            i=str(i)
            if len(grupo[i])==3 :
                       p=len(grupo[i][1])
                       if p>2:
                n=Distanciadias(grupo[i][2][0],grupo[i][2][1])
                f=grupo[i][2][0]
                for k in range(n):
                                foutf2.write('\n   CLASE'+i+' '+str(f)+' '+p+' '+str(grupo[i][1][0])+' "'+str(farti(grupo[i][1][0]))+'"')
                                f=sumagra(f,'d','+1')
       foutf2.close()

#PROGRAMA PRINCIPAL

if len(sys.argv)>1:
    time=sys.argv[1]

    indice=sys.argv[2]
    relevancia=sys.argv[3]
    abstract=sys.argv[4]
    threshold=sys.argv[5]
    aurora=sys.argv[6]
else:
    time=raw_input("Clases con fechas: Si(1)/ no(0)")

    indice=int(raw_input("con o sin indice? :1/0"))
    relevancia=raw_input("umbral tempora0<=1 ?")
    abstract=raw_input("si solo texto con fechas pulsa [1] sino otra tecla")
    threshold=raw_input("umbral de similitud 0<=1")
    aurora=int(raw_input("similitud con fechas(aurora) :1/0"))
    dir=raw_input("directorio datos? "+os.getcwd())
e=os.getcwd()+dir
d1=e+'2'+time+abstract
lclases=[]
lclasesA=[]
```

```
for j in range(len(relevancia.split(',',))):
  for i in range(len(threshold.split(',',))):
            lclases+=[{}]
            if aurora==1:lclasesA+=[{}]
Secciones=['','i','n','o','s','c','g','d','e']
parada=[]
e0=''
SEC=''
e0=raw_input("pulsa 0 para el estandar, si quieres un mes concreto teclea el número del mes")
SEC=raw_input("pulsa i-internacional,n-nacional,e-economia")
e1=e+e0+SEC+abstract
print abstract,e1,e,e0
if indice==1:analizarperiodicos(e,e0,e1)
else: analizadb(e1)
k1=0
grupo={}


farti=bsddb.hashopen(e1+'larti.db','r')
for k in relevancia.split(','):
      for i0 in threshold.split(','):

        if  len(lclases[k1])>0:
           creaficheroclases(lclases,'2'+str(time)+str(abstract),k,i0,k1,e1)
        if aurora==1 and len(lclasesA[k1])>0:
           creaficheroclases(lclasesA,'0'+str(time),k,i0,k1,e1)
        k1+=1
farti.close()
```

```
import re,calendar,time,string,os
from string import joinfields
from calendar import monthrange, leapdays, isleap

#from lexico import CGRANO
CGRANO=['d','w','m','t','c','e','y','z','s','l']

LETRA=map(chr,range(ord('A'),ord('Z')+1))+map(chr,range(ord('a'),ord('z')+1))+map(chr,range(ord('Á'),ord('ü')+1))+['-']
MINUS=map(chr,range(ord('a'),ord('z')+1))+['á','é','í','ó','ú','ñ']

#AA= corte del valor de la frecuencia de fechas en el ambito
AA=0.33 # valor extremo + periodo 30 dias.
#DD= dias maximo que se permite al intervalo
DD=7

#corte para similitud =0.23
CS=0.35
def anadedicc(dicc,dato,valor):
    if dicc.has_key(dato):
        dicc[dato]+=valor
    else:
        dicc[dato]=valor
def buscaverbo(pal):
    """
    Busca si una palabra es un verbo y en ese caso devuelve si se refiere a tiempo pasado - o futuro +
    requiere el tancat
    """
    P1=''
    pals=pal.split()
    for pal in pals:
     P2=convminus(pal)
     if P2<>0:
        '''pal=string.lower(pal)
        P2=os.popen('echo \''+pal+'\'|/usr/local/lib/upc/MACO/sp/bin/buscar.pl ','r')
        P2=P2.readlines()
        P2=P2[len(P2)-1]
        '''
        P2=os.popen('perl buscarpalabras.pl '+P2,'r').readline()
        if re.search(' (V(A|M){1}(I|S|M|C){1})+',P2):
            P2=re.search(' (V(A|M){1}(I|M|C|S){1}(S|F|P|I){1})\S*\s*$',P2)
            if P2:
                if P2.group(1):
                   if  P2.group(4)=='P':
               P1=''
               #parece que los verbos hay que poner - en el presente
               #pero lo pondré en cla_fechas.py
                       if re.search('M|S',P2.group(3)):
                          P1='0'
           elif P2.group(4)=='S'or P2.group(4)=='I':P1='-'
                   elif P2.group(4)=='F':P1='+'
                   break
    return P1

def fechaenfecha(F1,F2,gra2):
    #mira si F1 está recubierto por F2.
    #teniendo encuenta que gra2 es el índice de la granularidad minima de F2,
    #si el valor de gra2 es 10, se trata de un intervalo
    #la granularidad mínima de F1 debe ser menor que la de F2
    #y_1m_5d_3 en y_1999m_5w_2

    g1=gramin(F1)
    i=gra2
    if gra2<10:
        if g1<gra2:
            if i<6: i=6
            F11=actualgra(F1,CGRANO[gra2],0)
            if F11<>'-1':
                while i>=gra2 and actual(F11,CGRANO[i])==actual(F2,CGRANO[i]):

                    if i==gra2: return 1
                    i-=1
        else:return 0
        return 0
    else:
        I=F2.split(',')
        h20=gramax(I[0])
        h21=gramax(I[1])
        h1=gramax(F1)
        i=h20
        g=i
        if h20==h1 and h21==h20:
            #misma granularidad máxima
            g20=gramin(I[0])
            g21=gramin(I[1])

            if g21<g20 :g=g21
```

```
            else: g=g20
            if g1<=g:
                g=g1
            else:
                 X=refina(F1,CGRANO[g])
                 X=X.split(',')
                 i1=fechaenfecha(X[0],F2,10)==1
                 i2=fechaenfecha(X[1],F2,10)==1
                 if i1==1 and i2==1: return 1
                 else: return 0
            if g21>g:
                      X=refina(I[1],CGRANO[g])
                      X=X.split(',')
                      if len(X)>1:I[1]=X[1]
                      else: I[1]=X
            if g20>g:
                      X=refina(I[0],CGRANO[g])
                      X=X.split(',')
                      I[0]=X[0]
            if g1>g:
                      F1=actualgra(F1,CGRANO[g],0)
            i0=compara_fechas(F1,I[0])
            i1=compara_fechas(F1,I[1])
            if (i0 in[0,1]) and (i1 in[0,2]):return 1
            else: return 0
    return '-1'

def calculoambito(l_fechas,fech):
    #extrae el ambito y las veces que aparece cada fecha, llama a clasificación
    #FECHA=str(FECHA)
    #if len(LFECHAS)<2: return '--'
    ambito=''
    countdias=0
    count=0
    lambito={}
    lista={}
    l_int=[]
    t=0# numero total de referencias
    #calculo nº de días y nº de fechas
    if len(l_fechas)==1:
        a=l_fechas.keys()
        a=a[0]
        if re.search('-',a):
            ambito=a
        else: ambito=a+'-'+a
    else:
        for i in l_fechas.keys():
            v=float(l_fechas[i])
            t+=v
            j=i.split('-')
            if len(j)==2:

                r=DistanciaDias(j[0],j[1])
                if r==0: r=0.5
                if v>1:
                  lambito[i]=v

                  if r!= '-1' and (0<=r<=DD or 0<r<=7):
                        #si son menores de DD se pasan a días y se borran

                        v=float(v)/r
                        j1=j[0]
                        for k in range(0,r+1):
                            if lista.has_key(j1):lista[j1]+=v
                            else:     lista[j1]=v
                            j1=sumagra(j1,'d','+1')
                            j1=fecha_standar(j1)
                else:
                  #pongo los extremos
                  l_int.append(i)
                  if v>2 or  float(v)/r>0.2:
                    if lista.has_key(j[0]):lista[j[0]]+=v/2
                    else:     lista[j[0]]=v/2
                    if lista.has_key(j[1]):lista[j[1]]+=v/2
                    else:     lista[j[1]]=v/2
            else:
               #i=i+'-'+i
               if lista.has_key(i):lista[i]+=v
               else:     lista[i]=v

        #incremento el peso de los días que aparecen en otros puntos
        #con 0.2*apariciones
        for i in l_int:
            for k in lista.keys():
                j=i.split("-")
                if j[0]<k<j[1]:
```

```
                        lista[k]+=0.2*l_fechas[i]
        #tomo las fechas que superan el umbral de relevancia 2/DD
        min=2
        c={}
        for k in lista.keys():

            if lista[k] <(2.0/DD) :
                del(lista[k])
            else:
                if lista[k]<min: min=lista[k]
                c[k]=lista[k]

        #para cada intervalo
            #si el número de días para hacer matching es mayor de 6, entonces tendremos que pasar también las semanas a días
        min=2.0/DD
        b={}
        if len(lista.keys())>1:
                b=extraeambitosdias(lista,min)
                if len(b)>0:
                    lambito.update(b)

        for i in c.keys():
           if len(b)==0:
               lambito[i+'-'+i]=c[i]
           else:
              for k in b.keys():
                j=k.split("-")
                if len(j)<>2:
                    raw_input( j+'999')
                    j=j+j
                if not (j[0]<=i<=j[1]):lambito[i+'-'+i]=c[i]
        min=2+(1.0/DD)
        if len(lista)>0 and len(ambito)=='': print lista
        if len(lambito)>0:
                lv=lambito.values()
                lk=lambito.keys()
                lvr=[]+lv
                lvr.sort()
                lvr.reverse()
                posibilidades= lvr[0]
                if (posibilidades > 1 and len(l_fechas)<=3) or posibilidades>(2+(1.0/DD)):
                  i0=lvr[0]
                  k1=lv.index(lvr[0])
                  ambito=lk[k1]
                  while len(lvr)>1 and (int(i0)==int(lvr[1]) or lvr[1]>3)and not re.search(fech[0:6],lk[k1]):
                     del(lv[k1])
                     del(lk[k1])
                     del(lvr[0])
                     i0=lvr[0]
                     k1=lv.index(lvr[0])
                  if lk[k1]<>ambito and re.search(fech[0:6],lk[k1]):
                      ambito=lk[k1]
                      posibilidades=lvr[k1]
        if len(lambito)==1: ambito=lambito.keys()[0]
    return [ambito,lambito]
def extraeambitosdias(lfechas,min):
       # se le pasa una lista de fechas a nivel
       #de dias y devuelve el intervalo de fechas consecutivas y su
       #frecuencia
       lambito={}
       lf=lfechas.keys()
       lf.sort()
       c=lfechas[lf[0]]
       f0=lf[0]
       f1=''
       i=1
       while i<len(lf):
               r=DistanciaDias((lf[i-1]),(lf[i]))
               if (r<DD):
                       c+=lfechas[lf[i]]
                       f1=lf[i]
               else:
                   if f1<>'' and c>=min:
                       lambito[f0+'-'+f1]=c
                   f0=lf[i]
                   f1=''
                   c=lfechas[lf[i]]
               i+=1
       if f0<>f1 and f1<>'' :
           lambito[f0+'-'+f1]=c
       return lambito

def standar_fecha(FECHA1):
    #pasa de formato yyyymmdd a y_yyyym_mmd_dd
    FECHA2='-1'
    if re.search("^\d+$",FECHA1):
```

```
        if 6<len(FECHA1)<=8:
            FECHA2='y'+FECHA1[:4]+'m'+FECHA1[4:6]+'d'+FECHA1[6:8]
        elif 4<len(FECHA1)<=6:
            FECHA2='y'+FECHA1[:4]+'m'+FECHA1[4:6]
        else:FECHA2='y'+FECHA1[:4]
    else:FECHA2=FECHA1
    return FECHA2

def fecha_standar(F0):
    #pasa a formato yyyymmdd
    X=re.search("^\[*(\d+)\]*",F0)
    if X:
        F1=X.group(1)
    else:
        F1=''
        X=re.search("y(\d+)",F0)
        if X:
            F1=X.group(1)
            X=re.search("m(\d+)",F0)
            if X:
                m=X.group(1)

                if int(m)<10 and len(m)<2: m='0'+m
                F1+=m
                X=re.search("d(\d+)",F0)
                if X:
                    d=X.group(1)
                    if int(d)<10 and len(d)<2: d='0'+d
                    F1+=d
    if F1=='': F1='-1'
    if len(F1)<4:
        print F0,",",F1,"5454"
        #return -1
    return F1

def fecha_digitos(F0):
    #pasa una fecha a un formato de dígitos.
    #se utiliza para comparar fechas con la misma gramin y gramax.
    #y_1999m5=19990005 y y_1999w_5 = 19990005
    #para cada granularidad reservo 4 cifras que completo con 0
    F1=''
    if re.search("^\d+$",F0):
        F0=actualgra(F0,'d',0)
    X=re.search("^\D*?(\d+){1}(.*)$",F0)
    while X:
        a='0'*(4-len(X.group(1)))+X.group(1)
        F1+=a
        if X.group(2):
            F0=X.group(2)
        else: F0=''
        X=re.search("^\D*?(\d+){1}(.*)$",F0)
    return F1

def compara_fechas(F0,F1):
    #compara F0 con F1, 2 <, 0 =,1>
    #error '-1'
    if gramin(F0)==gramin(F1)and gramax(F0)==gramax(F1):
        F01=long(fecha_digitos(F0))
        F11=long(fecha_digitos(F1))
        if F01<F11: return 2
        elif F01==F11: return 0
        else :return 1
    else: return '-1'

def completafecha_digitos(F0,F2):
    #devuelvo un dia con la parte de F0 completada con F2
    #F2 debe ser una fecha standar yyyymmdd
    y=-1
    m=-1
    d=-1
    F1=''
    X=re.search("y(\d+)",F0)
    if X:
        F1=X.group(1)
        X=re.search("m(\d+)",F0)
        if X:
            m=X.group(1)

            if int(m)<10 and len(m)<2: m='0'+m
            F1+=m
            X=re.search("d(\d+)",F0)
            if X:
                d=X.group(1)
                if int(d)<10 and len(d)<2: d='0'+d
                F1+=d
            else:F1+=F2[6:8]
```

```
        else: F1+=F2[4:8]
    else:
        X=re.search("^(\d+)$",F0)
        if X:
            if len(X.group(1))==8:F1=X.group(1)
            elif len(X.group(1))==6: F1=X.group(1)+F2[6:8]
            elif len(X.group(1))<=4:
                F1=(4-len(X.group(1)))*'0'+X.group(1)+F2[4:8]
    return F1

def extraefecha(sent,O,FECHAI,PERIODOI,FECHA,ferr):
    """
    FECHA y FECHAI con 8 dígitos
    Periodo forma canónica
    funcion  que obtiene una FECHA a partir de una expresio temporal codificada(sent)
    FECHAI= fecha antes citada, PERIODOI=periodo antes citado,FECHA=fecha de publicacio

    """
    medio=''
    if re.search("0\.5",sent):
        sent=re.sub("0\.5","",sent)
        medio=1
    amb='3' #si no indico cantidad y granularidad es plural, supongo que es de 3
    if len(sent)<2 :
        print "2222",sent
        return '-1'
    A=''#cuantos añadir o quitar
    B=''#cantidad de granularidad
    C=''#granularidad
    cal=0 #global si fecha no calculada
    fecha0=''
    FF='-1' #Fecha resultado
    F=''
    GG=''
    GG=''
    FECHA1=FECHA
    FECHA2=FECHA1
    X=re.search("^(.*?(l|s|z|e|c|y|t|m|w|s|d){1}.*?)##+(.*)",sent)
    if X :
        sent=X.group(1)
        b=X.group(3)
        if len(sent)>1 :
              if len(b)<1: b='0'+b
        FF=extraefecha(b,O,FECHAI,PERIODOI,FECHA,ferr)
        if gramin(FF)==0:return (FF)
        FF=actualgra(FF,X.group(2),0)
    if re.search('k|x',sent):
        return '-1'
    sent0=sent
    if re.search('R',sent) :
          #R generalmente indica que depende de evento, pero
          #a veces pero se produce cierto error al poner esa codificación
          #si tenemos una granularidad fijada en la expresión tenemos una expresion
          # absoluta por lo tal sobra la R .
          # A veces esto es un error
          if not re.search('n|O|F|A|((l|s|z|e|c|y|t|m|w|s|d)\d+)|\d{4}',sent):
                  #no puedo extraer fechas si no tengo alguna granularidad de años

                  return '-1'

          else:
              sent=re.sub('R','',sent)
    if re.search("\+\S*?-",sent): sent=re.sub("\+","",sent,1)
    elif re.search("-\S*?\+",sent): sent=re.sub("-","",sent,1)
    if re.search("\+\S*?-",sent) or re.search("-\S*?\+",sent):raw_input("erree"+sent)
    y,m,d,p,w,r='','','','','',''
    X=re.search("^(\S*?)(\d*)(ny#*nm#*nd){1}\s*",sent)
    if X:
        B=0
        if len(X.group(2))>0:
            if re.search("-",sent): B="-"+X.group(2)
            elif re.search("\+",sent): B="+"+X.group(2)
        return sumagra(FECHA,'d',B)
    if re.search("(s+\w+)",sent):
        #print sigloXX no se decodificarlo
        return sent
    X=re.search("(y\d+#*m\d+#*d\d+)",sent)
    if X:
        X=re.sub("#","",X.group(1))
        return X
    if re.search('^\s*P?\d*('+string.joinfields(CGRANO,'|')+'){1}p?\s*$',sent):
        #duracion
        return '-1'
    sent0=re.search("^#*(\S*?)#*$",sent0)
    sent0=sent0.group(1)
    X=re.search("O(l|s|z|e|c|y|t|m|w|s|d){1}",sent0)
```

```
if X and re.search(X.group(1)+".*?"+X.group(1),sent0):
    sent=re.sub("O\w{1}#","",sent0)
    if sent0==sent: raw_input("weeoeeo")
    sent0=sent
    #print sent0,"11"
FC=re.split('#*',sent0)
if len(FC)>0:
  F1=""
  if gramin(FC[0])==-1:
    if len(FC)==1: return '-1'
    a=FC[0]
    FC.remove(FC[0])
    for i in range(len(FC)):
        FC[i]=a+FC[i]

  while FC<>F1 :
   j=gramin(FC[0])
   F1=[]+FC
   for i in range(1,len(FC)):
    k=gramin(FC[i])
    if k>j:
        k1=FC[i-1]
        FC[i-1]=FC[i]
        FC[i]=k1
    j=k
for sent0 in FC:
        if sent0==FC[0] and len(sent0)==1: continue
        A=''#cuantos añadir o quitar
        B=''#cantidad de granularidad
        C=''#granularidad
        D=''#plural
        p=''
        G=''
        O=''
        X=re.match('^#*((\S*?(n|P|r|I|\+|-|O|F|A)*)(o?\d*?)((l|s|z|e|c|y|t|m|w|s|d){1}(p)?)(o?(\d+))?)(\S*)$',sent0)
        if X and X.group(6):
            G=X.group(6)
            if X.group(7):
                D='p'
        else:
            break
        if X.group(2): A=X.group(2)
        if X.group(9):B=X.group(9)
        if X.group(4):C=X.group(4)
        if X.group(10):
            if len(X.group(10))>1:
              print "error codificacion",sent0,sent
              print X.groups()
              return sent
        if len(FC)>1:
            if C==''  and B==''  and re.search("O",sent0) :
              if not re.search("O",sent):
                  print sent,"3131"
                  raw_input()
              continue
        C1=re.match("(o?)(\d+)",C)
        if C1 and C1.group(1):
            if FF<>-1:
                B=C1.group(2)
                C=''
            else:
                print PERIODOI, FECHAI, "ordinales"
                raw_input(sent)
                C=C1.group(2)
        fecha0=fechref(G,FECHAI,PERIODOI,FECHA)
        sent1=X.group(1)
        if A.find('+') <> -1:
                          O='+'
        elif A.find('-') <> -1:
                          O='-'
        if A.find('P') <> -1:p=1
        if B=='':
            if A.find('n')<>-1:
                        FECHA1=FECHA
                        if A.find('r')<>-1: A=re.sub("r","",A)
                        if C<>'' and B=='' and O=='':
                            O='-'
            else:
              if A.find('R')<>-1:
                  #print "555",sent,sent0
                  return '-1'
              elif A.find('r')<>-1:
                  if len(FF)>4: FECHA1=FF
                  elif A.find('O')<>-1 and fecha0=='': FECHA1=FECHA
                  else:
                      if fecha0=='':
```

```
                print "444",FECHAI,'-',PERIODOI,'-',FF,'-',FECHA,sent
            FECHA1=fecha0

      else:
        if FF<>'-1': FECHA1=FF
        #elif FECHAI<>'':FECHA1=FECHAI
        else:
            if A.find('O')<>-1 and FECHAI<>''and  0<DistanciaDias(FECHAI,FECHA)<7:
                    FECHA1=FECHAI
            else: FECHA1=FECHA

elif FF<>'-1':
    FECHA1=FF
    if re.search(",",FECHA1):
                FECHA1=FECHA1.split(',')
                FECHA1=FECHA1[0]
elif  re.search("A|F",sent) and sent0==FC[len(FC)-1] and (not A.find("n") or B<>''):
    print "XXX"
    continue
if C=='' and B=='' and (O==''  or A.find("n")!=-1 ) :
    if not re.search("A|F",sent):

       if sent0.find('P')<>-1 or D=='p' :
            FF="-1"
            #estas semanas cojo el mes de la fecha si existe
            #esos últimos meses, estos últimos años, ultimos dias <> últimos dias de mayo...
            if A.find('P')==-1 and A.find('r')<>-1 and PERIODOI<>"":
                    #esas semanas cojo el periodo citado antes
                    FF=PERIODOI
            else:
                    G1=CGRANO[CGRANO.index(G)+1]
                    FF=actualgra(FECHA1,G1,'0')
                    #últimos días de mayo
                    if FF=='-1':
                        G1=CGRANO[gramin(FECHA1)]
                        FF=actualgra(FECHA1,G1,'0')
       else:
          FF= actualgra(FECHA1,G,'0')
else: FF='-1'
if FF=='-1':
    if B=='' and C=='' :
        cal=1
    if B == '' and (C<>'' or cal==1):
            if FECHA1=='':
                print "666",sent0,sent,fecha0,FECHAI
                return '-1'
            C=re.sub('o','',C)
            if medio=='1':
                C=str(int(C)+1)
            if O=='' and (cal<>1 or  re.search("n",sent0)):
                #if C=='' and re.search("(A|F)",A)<> -1: continue
                #'durante los ultimos 2 días
                #pero si los ultimos 2 días
                if A.find('P')<>-1:continue
            if O<>'':
                if cal==1:
                    if D=='p': C=amb
                    else: C='1'
                C=O+C
            else: continue
            if C=='': continue
            # esta última semana
            #estos 2 ultimos dias
            if G=='w' :
                S1=actual(FECHA1,'w')
                if S1<>'-1':
                   C=str(int(C)*7)
                   FF=sumagra(FECHA1,"d",C)
                else:
                   print "error",sent,G,FECHA1
                   return '-1'
            else:
                FF=sumagra(FECHA1,G,C)
            if FF=='-1':
                    G1=CGRANO[gramin(FECHA1)]
                    FF=actualdias(FECHA1,G1,'0')
                    if FF<>'-1':
                        FF=FF.split(',')
                        if O=='-' and len(FF)==2:
                            FF=FF[1]
                        else:FF=FF[0]
                        if cal<>1:
                            FF1=sumagra(FF,G,C)
                            if O=='-':
                              FF='['+FF1+','+FF+']'
                            else:
```

```
                                    FF='['+FF+','+FF1+']'
                    else:
                            FF=actualgra(FF,G,0)
                            print D,FF,p,cal
                            if (cal==1 and FF<>'-1' and D=='p') or(cal<>1 and D=='p')  or p==1:
                                print "WWSS",G
                                FFx=actualgra(FECHA1,G,0)
                                print FFx
                                if O=='-':
                                  FF='['+FF+','+FFx+']'
                                else:
                                    FF='['+FFx+','+FF+']'
                elif B<>'':
                 if G=='d' and re.search('w',sent):
                     #dia respecto a semana
                     #HAy que tener formato m_X y tambien
                     ##necesito una fecha de referencia
                     FF=pasocanonica(B,"d",FECHA1)
                     if not re.search("O",sent):
                         if O<>'': sent=O+sent
                         else: sent='-'+sent
                     if FECHA1==FECHA2 and O<>'':
                             F1=compara_fechas(fecha_standar(FF),fecha_standar(FECHA1))
                             if re.search("\+",sent):
                                 if F1 in [0,2]:  FF=sumagra(FF,'d','+7')

                             elif re.search("-",sent):
                                 if F1 in [0,1] : FF=sumagra(FF,'d','-7')

                 elif G=='w'and (not re.search('m',sent)):
                     #HAy que tener formato y_X
                     FF=pasocanonica(B,"w",FECHA2)
                 else:

                     [a,b]=ultimagra(G,FECHA1)
                     if int(a)>0 and  int(B)>int(a):
                         print "777",a,B,G,FECHA1,sent
                         return '-1'
                     if G=='d' and int(B)>31:
                         FF=FECHA1

                     elif G=='m' and int(B)>12: FF=FECHA1
                     elif  re.search("O",sent) and FECHA1==FECHA :
                      k1=len(FC)
                      if k1<2: k1=0
                      else: k1=1
                      if  sent0==FC[k1] :
                        C=re.search("(\+|-)\D+",sent)
                        if C:
                          if C.group(1)=="+":
                             if int(actual(FECHA1,G))>int(B):
                                FECHA1=sumagra(FECHA1,'y','+1')

                          elif int(actual(FECHA1,G))<int(B):
                             FECHA1=sumagra(FECHA1,'y','-1')
                     FF=actualgra(FECHA1,G,int(B))
        print FF
        if FF=='-1': continue
if re.search("F|A",sent) and not re.search(",",FF):
    #último día Fd
    #últimos meses, últimos 3 meses, a últimos de mes,final de mes
    #PFmp,PF3mp,Fm,PFm,Fm, a principios de los próximos 3 meses
    if not (re.search("\d+d{1}",sent) or re.search("d{1}\d+",sent)):
        #últimos 5 dias,
        G0=gramin(sent)
        gra=CGRANO[G0]
        if FF=='-1':
            FF=actualgra(FECHA,gra,0)
        if re.search("(\+|-){1}\d+",sent):
            raw_input("error-afsdasdf")
        cant=0
        X=re.search("(F|A)(\d+){1}",sent)
        if not X :
            if not re.search("P",sent):
                if not re.search("p",sent) :
                    if G0==0: cant=0 ##final del dia
                    else:

                         cant=2
                    G0='d'
                    #fin de semana, fin de siglo, final del día
                else:
                    raw_input("error número posiblemente"+ sent)
                    G0=CGRANO[G0]
                    #caso else no debe ocurrir final de años.... sería un error
            else:
```

```
                    if not re.search("p",sent) :
                         cant=3
                         if G0>=6:
                           #últimos de siglo, finales de año
                           G0=CGRANO[G0-1]
                         else:
                           G0='d'
                    else:
                        G0=CGRANO[G0]
                        #últimos meses,
                gra=CGRANO[gramin(FF)]
                FF=actualdias(FF,gra,'0')
                print gra,FF,cant
                if FF=='-1': return '-1'
                print "11",FF
                if len(FF)<>2 :
                    FF=FF+','+FF
                print FF,G0,"33"
                FF=FF.split(',')
                if G0<>'d':
                    FF[0]=actualgra(FF[0],G0,0)
                    FF[1]=actualgra(FF[1],G0,0)

                if re.search("A",sent):
                         FF=FF[0]
                         FF=FF+','+sumagra(FF,G0,'+'+str(cant))
                else:
                         FF=FF[1]
                         FF=sumagra(FF,G0,'-'+str(cant))+','+FF
            elif X:
                if re.search("A",sent):
                    FF=standar_fecha(FECHA)+','+sumagra(FF,gra,'+'+X.group(2))
                else:  FF=sumagra(FF,gra,'-'+X.group(2))+','+standar_fecha(FECHA)
            if len(FF)>4: FF="["+FF+"]"
    if len(FF)<2 and FF<>'-1':
        print FF,"1111111111",sent
        raw_input()
        #return '-1'
    pp=extraefechas(FF)
    if len(pp)==2 and re.search("I|F|A|[",sent):
        p=['','']
        a=gramin(pp[0])
        if a==0: p[0]=fecha_standar(pp[0])
        b=gramin(pp[1])
        if b==0 : p[1]=fecha_standar(pp[1])
        if a>b: a=b
        a=CGRANO[a]
        if p[0]=='':
            p[0]=actualgra(pp[0],a,0)
            p[0]=pp[0].split(',')[0]
        if p[1]=='':
            pp[1]=actualgra(pp[1],a,0)
            pp[1]=pp[1].split(',')
            if len(pp[1])==2: p[1]=pp[1][1]
            else: p[1]=pp[1][0]
        if p[0]>p[1]:
            print FF,pp,sent,PERIODOI,FECHAI
            raw_input('errordfad')
    return FF

def espalabra(a):
    #detecta si a es una palabra devuelve 1 si lo es, si no 0.
    # este método permite algunos caracters que no son letras
    # entre 91-96, 123-126,161-191, pero no me inporta suelen ser separadores
    i=0
    b=1
    while i<len(a) and b==1:
        if    ord(a[i])<65:    b=0
        i=i+1
    return b

def esmayuscula(c):
    castellano=['Á','É','Í','Ó','Ú','Ñ']
    # devuelve 0 si el caracter c no es mayúscula , 0 en caso contrario el caracter en minúscula
    if ('A' <=c<='Z') or (c in castellano): return ord(c)+32
    else: return 0

def convminus(a):
    global LETRA
    b=''
    for i in range(len(a)):
        if not (a[i] in LETRA):
            return ''
        c=esmayuscula(a[i])
        if c<>0:
            b=b+chr(c)
```

```
        else: b=b+a[i]
    return b


def solapar(a1,a2,b1,b2):
#min1 max1, min1,max2
  try:
    if not((int(a2)<int(b1))or (int(a1) >int(b2))):
        return 1
    else: return 0
  except:
        print a1,a2,b1,b2
        return 0

def cargafich(d):
     f=open (d,'r')
     J=f.readline()
     X=[]
     S=[]

     while J:
         X=X+J.split(';')
         J=f.readline()
     for i in X:
         Y=i.split('=')
         Y[0]=re.sub('\s+','',Y[0])

         S.append(Y[0])
     f.close()
     return S

def cargasinonimos(d):
     f=open (d,'r')
     J=f.readline()
     X={}
     S=[]
     k=0
     X1={}
     X2={}
     while J:
         J=J.strip()
         S=J.split(' ')
         if X2.has_key(S[0]):
             k=1
             X2[S[0]]=X2[S[0]]+[S[1]+S[2]]
         else:
             if k==1:
                for i in X1.keys():
                  X[i]=[X1[i]]+[X2[X1[i]]]
             k=0
             X1={}
             X2[S[0]]=[S[1]+S[2]]
         X1[S[1]+S[2]]=S[0]
         J=f.readline()
     f.close()
     return(X)

def fechaseg(T):
    if len(T)==6: T=T+'01'
    if len(T)==4: T=T+'01'
    if len(T)<>8:return -1
    try:
         return calendar.timegm([int(T[0:4]),int(T[4:6]),int(T[6:8]),0,0,1])
    except: return -2

def DistanciaDias(Fecha1, Fecha2): # Determina la cantidad de días entre las dos fechas simples.
# algoritmo diseñado por AURORA
# La variable C añadida por lola. De modo que el resultado sera´<0 si Fecha1>FECHA2
    #try:

    if (len(Fecha1)<>8 or len(Fecha2)<>8):
    return '-1'
    C=1
    if Fecha1 == Fecha2:
      return 0
    else:
      if Fecha1 > Fecha2:   #Pone en Fecha1 la más chiquita.
        C=-1
        Fecha_Aux = Fecha1
        Fecha1 = Fecha2
        Fecha2 = Fecha_Aux
      Ano1 = int(Fecha1[:4])
      Mes1 = int(Fecha1[4:6])
      Dia1 = int(Fecha1[6:8])
      Ano2 = int(Fecha2[:4])
      Mes2 = int(Fecha2[4:6])
```

```
      Dia2 = int(Fecha2[6:8])
      if not ((1970<Ano1<2038) and (1970<Ano2<2038)):
     return ((Ano2-Ano1)+1)*365*C
      if Mes1 < Mes2:
        DifAno = Ano2 - Ano1
        if Dia1 <= Dia2:
          DifMes = Mes2 - Mes1
          DifDia = Dia2 - Dia1
        else:
          DifMes = Mes2 - Mes1 - 1
          Mes = Mes2 - 1
          (PrimerDia, CantDiasdelMes) = monthrange(Ano2,Mes)
          DifDia = CantDiasdelMes - Dia1 + Dia2
      elif Mes1 > Mes2:
        DifAno = Ano2 - Ano1 - 1
        if Dia1 <= Dia2:
          DifMes = (12 - Mes1) + Mes2
          DifDia = Dia2 - Dia1
        else:
          DifMes = (12- Mes1) + Mes2 - 1
          if Mes2 <> 1:
            Mes = Mes2 - 1
            (PrimerDia, CantDiasdelMes) = monthrange(Ano2,Mes)
          else: # Es el mes de enero.
            Mes = 12
            Ano = Ano2 - 1
            (PrimerDia, CantDiasdelMes) = monthrange(Ano,Mes)
          DifDia = CantDiasdelMes - Dia1 + Dia2
      else:  # Mes1 == Mes2
        if Dia1 <= Dia2:
          DifAno = Ano2 - Ano1
          DifMes = 0
          DifDia = Dia2 - Dia1
        else:
          DifAno = Ano2 - Ano1 - 1
          DifMes = 11
          if Mes2 <> 1:
            Mes = Mes2 - 1
            Ano = Ano2
          else: # Es el mes de enero.
            Mes = 12
            Ano = Ano2 - 1
          (PrimerDia, CantDiasdelMes) = monthrange(Ano,Mes)
          DifDia = CantDiasdelMes - Dia1 + Dia2
      if (Ano1 <> Ano2) and isleap(Ano2) and (Mes1 == Mes2) and (Mes2 > 2):
        CantAnosBisiestos = leapdays(Ano1,Ano2) + 1  #porque hay que contar también a Ano2.
      else:
        CantAnosBisiestos = leapdays(Ano1,Ano2)
      #linea siguiente descomentada por loal
      CantDiasMes = 0              #Calculando la suma total de días que hay en los meses de diferencia.
      if DifAno == Ano2 - Ano1:
        Ano = Ano2
      else:
        Ano = Ano2 - 1
      for i in range(1,DifMes+1):
         Mes = Mes1+i
         if Mes > 12:
           Mes = Mes - 12
         (PrimerDia, CantDiasdelMes) = monthrange(Ano, Mes)
         CantDiasMes = CantDiasMes + CantDiasdelMes

      CantDias = 365 * DifAno + CantAnosBisiestos + CantDiasMes + DifDia
      return CantDias*C


def fechref(grano,FECHAI,PERIODOI,FECHA):
    """obtiene la fecha de referencia a partir de fechas ya analizadas
    """
    #global FECHAI,PERIODOI,FECHA
    salida=''
    if  grano=='d' :
        if FECHAI<>'' :salida= FECHAI
        else:salida=''

    else:
      if PERIODOI<>'':
        A=re.search('^\[*([^,]+),?([^,]+?)?\]*$',PERIODOI)
        P=[]
        if not A:
            print PERIODOI
            raw_input("errorrrrr 3")
            salida=PERIODOI
        else:
          if A.group(1):
            P.append(A.group(1))
          if A.group(2):
```

```
            P.append(A.group(2))

          if len(P)==1: salida=P[0]
          else:
            if  re.search(FECHA,PERIODOI):
                 P.remove(FECHA)
                 salida=P[0]
            else:
                if P[0][0:6]==FECHA[0:6]:
                   if P[1][0:6]!=FECHA[0:6] :
                        salida=P[1]
                   else:
                        salida=P[0]
        if CGRANO.index(grano)<=6:
              salida=completafecha_digitos(salida,FECHA)
      else: salida=FECHAI
    return salida

def extraefechas(T):
#se le pasa una cadena con 2 fechas separadas por ',' y devuelve una lista
    x=re.search('^[*(.*?),{1}(.*)]*$',T)
    a=[]
    if x:
        for i in x.groups():
            if len(i)>4 : a+=[i]
    if a==[]:a=[T]
    return a

def ultimagra(gra,F):
    #devuelve una lista con la última granularidad y la granularidad superior para las operaciones
    try:
        max=-1
        sup='y'
        if  gra=='m': max=12
        elif gra=='t' : max=4
        elif  gra=='c': max=3
        elif gra=='e' : max=2
        elif gra=='z' :
        max=100
        sup='s'
    elif gra=='d':
                sup='m'
                y=actual(F,'y')
                m=actual(F,'m')
                try: max=calendar.monthrange(y,m)
                except:max=calendar.monthrange(1999,m)
                max=max[1]
    elif gra=='w':
                sup='m'
                y=actual(F,'y')
                try: max=len(calendar.monthcalendar(y,actual(F,'m')))
                except: max=len(calendar.monthcalendar(1999,actual(F,'m')))
        else:
                max=0
                sup=0
        max=str(max)
        return [max,sup]
    except:
                print max,sup,F," 543"
                #raw_input('error'+gra+' '+F)
                return ['-1','-1']

def elmes(M):
    #entrada el mes ,devuelve el nfmero del mes
    MES =['enero','febrero','marzo','abril','mayo','junio','julio','agosto', 'septiembre','octubre','noviembre','diciembre','']
    if len(M) <1: return '0'
    A=0;
    while A<=12 and MES[A] <> M.lower() :
        A=A+1
    A=A+1
    if A >0 and A <13  :
        A=str(A)
        if len(A) < 2 :A='0'+A
        return A
    else :return '0'

def semanaperiodo(Fech):
    '''
    indica el periodo en dias de la semana W del año o mes (G) a partir de la fecha Fech
     obtener la semana 4 del mes de la fecha  semanaperiodo(4,'m',Fech)
     o bien la semana 4 del año de la fecha  semanaperiodo(4,'y',FECh)
    '''
    #print W,G,Fech
    W=int(actual(Fech,'w'))
    if W <1: return '-1'
    else:W=W-1
```

```
    y=actual(Fech,'y')
    m=actual(Fech,'m')
    if m<>'-1':
        if W>6: return'-1'
    else:
        if W>60:return('-1')
        m=0
        F=[]
        j=0
        while  (j<=W) and (m<12):
                    m=m+1
                    F=calendar.monthcalendar(y,m)
                    j=j+len(F)
        j=W-(j-len(F))
        if j>=len(F): print 'err2',W,Fech,F,m,j
        W=j
    F=calendar.monthcalendar(y,m)
    if W>=len(F):
        print 'error fecha',Fech
        raw_input()
        return '-1'

    F=F[W]
    i2=F[len(F)-1]
    i1=F[0]
    m2=m
    if i1==0 and m>1:
        m=m-1
        F1=calendar.monthcalendar(y,m)
        F1=F1[len(F1)-1]
        i1=F1[0]
    if i2==0 and m<12:
        m2=m2+1

        F1=calendar.monthcalendar(y,m2)
        F1=F1[0]
        i2=F1[6]

    i1=str(i1)
    i2=str(i2)
    F='y'+str(y)+'m'+str(m)+'d'+i1+',y'+str(y)+'m'+str(m2)+'d'+i2

    #  F,'2223'
    return F

def pasocanonica(W,G,FECHA1):
    '''
    año-semana
    devuelve la forma canonica y_m_w_
    semana-dia
    devuelve la forma canonica y_m_d
    '''
    n=-1
    Fecha='-1'
    y=actual(FECHA1,'y')
    if y=='-1': return ('-1')
    if G=='w' :
        W=int(W)-1
        if W>60:return('-1')
        m=0
        F=[]
        j=0
        while  (j<=int(W)) and (m<12):
                    m=m+1
                    F=calendar.monthcalendar(y,m)
                    j=j+len(F)
        j=W-(j-len(F))
        Fecha="y"+str(y)+"m"+str(m)+"w"+str(j+1)
    if G=='d':
        m=actual(FECHA1,'m')
        if m=='-1': return ('-1')
        W=int(W)
        d=actual(FECHA1,'d')
        if d=='-1':
            s=actual(FECHA1,'w')
            if s<>'-1':
                d=semanaperiodo("y"+str(y)+"m"+str(m)+"w"+str(s))
                d=d.split(',')
                FECHA1=d[0]
                d=actual(FECHA1,'d')
                n=calendar.weekday(y,m,d)+1
                if n>W: n=7-n+W
                elif n==W: n=0
                else: n=W-1
            else:
                print W,G,FECHA1,"333"
```

```
                raw_input(FECHA1)

        else:
            n=calendar.weekday(y,m,d)+1
            if n==W: n=0
            elif n>W:
                    n=n-W
                    n='-'+str(n)
            else:
                    n=W-n
        if n==0: Fecha=FECHA1
        else: Fecha=sumagra(FECHA1,'d',str(n))
    return Fecha

def refina(FECHA1,gra):
    #devuelve una expresión, que puede ser un intervalo en
    #forma canonica y_1999,y_3000
    c=gramin(FECHA1)
    if c=='-1': return  '-1'
    if gra=='d':
        FECHA2=actualdias(FECHA1,CGRANO[c],0)
        return FECHA2
    d=CGRANO.index(gra)
    FECHA2='-1'
    if c>d:
        #caso contrario seria abstraer, que se consigue con "actualgra"
        FECHA2=actualgra(FECHA1,'d',0)
        if FECHA2=='-1': return  '-1'
        max=ultimagra(gra,FECHA1)
        if max[1]<>0:
            if CGRANO[c]==max[1]:
                FECHA2=FECHA2+gra
                FECHA2=FECHA2+'1,'+FECHA2+max[0]
        else:
            FECHA2=actualdias(FECHA1,CGRANO[c],0)
            if FECHA2<>'-1':
                FECHA2=FECHA2.split(',')
                FECHA2=actualgra(FECHA2[0],gra,0)+','+actualgra(FECHA2[0],gra,0)
    return FECHA2

def actual(FECHA1,gra): # devuelve un número
    #devuelve el valor de la granularidad gra en FECHA1
    FECHA1=standar_fecha(FECHA1)
    if gra=='-1' or FECHA1=='-1': return  '-1'
    c=gramin(FECHA1)
    if c=='-1': return  '-1'
    d=CGRANO.index(gra)
    FECHA2='-1'
    #abstraer
    if c<=d:
        FECHA2='-1'
        X=re.search(gra+'(\d+)',FECHA1)
        if X:
            a=X.group(1)
            if gra=='y':
              if len(a)==2: y='19' +a
            return int(a)
        if d>6:
            y=str(actual(FECHA1,'y'))
            if gra=='y': FECHA2=y
            elif y<>'-1':
                if gra == 'l':
                    FECHA2=int(y[0:1])
                    if y[1:4]=='000': FECHA2=FECHA2-1
                elif gra == 's':
                    FECHA2=int(y[0:2])+1
                    if y[2:4]=='00': FECHA2=FECHA2-1
                elif gra == 'z':
                    FECHA2=int(y[2:3])
                    if y[3:4]=='0' and y[2:3]<>'0': FECHA2=FECHA2-1
        elif d>2:
            m=actual(FECHA1,'m')
            if gra== 't':
                 if m<=3: FECHA2=1
                 elif m <=6 :FECHA2=2
                 elif m <=9 :FECHA2=3
                 else: FECHA2=4
            if gra== 'c':
                 if m<=5: FECHA2=1
                 elif m <=9 :FECHA2=2
                 else: FECHA2=3
            if gra== 'e':
                 if m<=7: FECHA2=1
                 else: FECHA2=2
        elif d==1:
            try:
```

```
                F1=calendar.monthcalendar(actual(FECHA1,'y'),actual(FECHA1,'m'))
                d=actual(FECHA1,'d')
            except: return '-1'
            i=0
            while (i < len(F1)) and (d>=int(F1[i][0])):
                 i=i+1
            FECHA2=i
    return FECHA2

def sumagra(FECHA1,gra,n):
    """
    FECHA1 formato de dígitos
    Suma n granularidades de tipo gra a la fecha. Si gra<>'d' devuelve un periodo
    """
    #solo funciona con sumas de n granularidades de forma que |n|<2*max(grasup)
    # o sea para restar o sumar menos de 60 días o menos de 24 meses o...
    if FECHA1=='-1' : return '-1'
    n=int(n)
    if n==0: return FECHA1
    [max,sup]=ultimagra(gra,FECHA1)
    val=int(actual(FECHA1,gra))
    if max=='-1' or val<0: return '-1'
    max=int(max)
    FECHA2='-1'
    FECHA1=standar_fecha(FECHA1)
    Aux1=FECHA1
    if not re.search(gra,FECHA1): Aux1=actualgra(FECHA1,gra,val)
    a2=val+n
    a1=0
    #print max,a2,a1,sup,FECHA1,Aux1
    if not(0<a2<max) and sup<>0:
        gra1=sup
        while (a2-max)>0:
                 a2=a2-max
                 Aux1=sumagra(Aux1,gra1,'+1')
                 [max,sup]=ultimagra(gra,Aux1)
                 max=int(max)
                 a1=0
        while a2<=0:
                 a1=0
                 Aux1=sumagra(Aux1,gra1,'-1')
                 [max,sup]=ultimagra(gra,Aux1)
                 max=int(max)
                 a2=max+a2
    val=str(val)
    if a1<>0:
        #Aux1=actualgra(Aux1,gra,a1)
        a1=str(a1)
        Aux2=re.sub(gra+val,gra+a1,Aux1)
        if Aux1==Aux2:
          if  len(a1)<2 :
             a1='0'+a1
          if len(val)<2: val='0'+val
          Aux2=re.sub(gra+val,gra+a1,Aux1)
    else:
        a2=str(a2)
        if val<>a2:
          Aux2=re.sub(gra+val,gra+a2,Aux1)
          if Aux1==Aux2 :
             if len(a2)<2 : a2='0'+a2
             if len(val)<2 : val='0'+val
             Aux2=re.sub(gra+val,gra+a2,Aux1)
        else:
            Aux2=Aux1
            Aux1=''
    if Aux2==Aux1 :
        print val,a1,a2,Aux1,Aux2,gra,FECHA1,n
        raw_input("error")
    return Aux2

def actualdias(FECHA1,gra,VALOR):
# devuelve el dia o periodo indicado en formato calendario
# permite con Fecha1 obtener otra fecha segun el valor de la granularidad
# tambien permite refinara dias
# si quiero calcular la fecha del dia 1, len(FECHA1)< 8
# si no se sobreentiende la fecha del 1 dia de la semana, len(FECHA)==8
# FECHA1 fecha para coger datos que requiero Valor= valor de la granularidad
# casos útiles
#   refinar un año a dias: actualdias(X,'y',1999) devuelve[19990101,19991231]
#   o bien obtener el periodo del año de la fecha:
#
actualdias('19990601','y','0') devuelve [19990101,19991231] #
actualdias(199906,'d',06) devuelve 19990606
    VALOR=int(VALOR)
    if VALOR<=0:
        VALOR=actual(FECHA1,gra)
```

```
        if VALOR==’-1’: return ’-1’
        if gra==’d’:
            y=str(actual(FECHA1,’y’))
            m=str(actual(FECHA1,’m’))
            if y != ’-1’and m != ’-1’:
                return ’y’+y+’m’+m+’d’+str(VALOR)
            else:
                return -1
    FECHA2=’-1’
    if gra == ’l’:
        FECHA2=str(VALOR)+"0010101,"+str(int(VALOR)+1)+"0001231"

    elif gra == ’s’:
         A=str(int(VALOR)-1)
         A=’0’*(2-len(A))+A
         FECHA2= str(A)+"010101,"+str(int(A)+1)+"001231"
    elif gra == ’z’:
        VALOR=str(VALOR)
        if len(VALOR)>1:VALOR=VALOR[0:1]
        s=actual(FECHA1,’s’)
        A=str(s-1)+VALOR
        FECHA2= A+"10101,"+str(int(A)+1)+"01231"
    elif gra == ’y’:
         VALOR=str(VALOR)
         if len(VALOR)>=4: FECHA2=VALOR[0:4]
         else:
             c=4-len(VALOR)
             FECHA2=’0’*c+VALOR
         FECHA2=FECHA2+’0101,’+FECHA2+’1231’
    else:
        y=str(actual(FECHA1,’y’))
        if gra == ’m’:
              VALOR=str(VALOR)
              y=str(actual(FECHA1,’y’))
              if len(VALOR)==1: VALOR=’0’+VALOR
              FECHA2=y+VALOR
              FECHA2=FECHA2+’01,’+FECHA2+ultimagra(’d’,FECHA2)[0]
        elif gra == ’t’:
            if VALOR==1 : FECHA2=y[0:4]+’0101’+’,’+y[0:4]+’0331’
            elif VALOR==2 : FECHA2=y[0:4]+’0401’+’,’+y[0:4]+’0630’
            elif VALOR==3 : FECHA2=y[0:4]+’0701’+’,’+y[0:4]+’0831’
            elif VALOR==4: FECHA2=y[0:4]+’0901’+’,’+y[0:4]+’1231’
        elif  gra == ’c’:
            if VALOR==1:FECHA2=y[0:4]+"0101,"+y[0:4]+"0330"
            elif VALOR==2:FECHA2=y[0:4]+"0401,"+y[0:4]+"0831"
            elif VALOR==3:FECHA2=y[0:4]+"0801,"+y[0:4]+"1231 "
        elif  gra == ’e’:
            if VALOR==1:FECHA2=y[0:4]+"0101,"+y[0:4]+"0630"
            elif VALOR==2:FECHA2=y[0:4]+"0701,"+y[0:4]+"1231"
        else:
            m=str(actual(FECHA1,’m’))
            if (gra == ’w’):
                if m==’-1’: # semana x del año
                    FECHA2=semanaperiodo(’y’+y+’w’+str(VALOR))
                else:  #semana x del mes
                        FECHA2=semanaperiodo(’y’+y+’m’+m+’w’+str(VALOR))
            elif gra==’d’:# el miercoles
                d=actual(FECHA1,’d’)
                if d<>’-1’:
                    n=calendar.weekday(int(y),int(m),int(d))+1
                    if n==VALOR: FECHA2=FECHA1
                    elif n>VALOR:
                        n=n-VALOR
                        FECHA2=sumagra(FECHA1,’d’,’-’+str(n))
                    else:
                        n=VALOR-n
                        FECHA2=sumagra(FECHA1,’d’,str(n))
                else:   # dia 4
                    FECHA2=’y’+y+’m’+m+’d’+str(VALOR)
    if FECHA2==’-1’: print FECHA1,gra,VALOR,’0002’
    fechas=FECHA2.split(’,’)
    FECHA2=standar_fecha(fechas[0])
    if len(fechas)==2: FECHA2+=’,’+standar_fecha(fechas[1])
    return FECHA2

def gramin(FECHA): #devuelve el índice de la granularidad mínima
    g=-1
    X=re.search(’(’+joinfields(CGRANO,’|’)+’){1}p?\d*$’,FECHA)
    if X:
        g= CGRANO.index(X.group(1))
    else:
        X=re.search(’(’+joinfields(CGRANO,’|’)+’){1}p?\w*$’,FECHA)
        if X:
         g= CGRANO.index(X.group(1))
        else:
            X=re.search(’(\d{4,})’,FECHA)
```

```
            if X:
                c=len(X.group(1))
                if c==8: g=0
                elif c==6: g=2
                elif c==4: g=6
    return g

def gramax(FECHA): #devuelve el índice de la granularidad máxima
de una expresión canónica
    g=-1
    X=re.search('^\S*?('+joinfields(CGRANO,'|')+'){1}p?\d+',FECHA)
    if X:
        g= CGRANO.index(X.group(1))
    else:
        if re.search('^\d+$',FECHA):
            g=6
    return g

def actualgra(FECHA1,gra,VALOR):
#abstrae una fecha  a la granularidad gra si valor=0
#o bien , a la granularidad gra le pone el valor VALOR
# devuelve el dia o periodo indicado con la granularidad de gra
# si quiero calcular la fecha del dia 1, len(FECHA1)< 8
# si no se sobreentiende la fecha del 1 dia de la semana, len(FECHA)==8

    if gra=='-1' or gra==-1 or FECHA1=='-1': return '-1'
    FECHA2='-1'
    FECHA1=standar_fecha(FECHA1)
    if FECHA1=='-1' :

        return '-1'
    if int(VALOR)==0:
        if gra=='d':
            g=CGRANO[gramin(FECHA1)]
            val=actual(FECHA1,g)
            if g=='d': val='0'
            FECHA2=actualdias(FECHA1,g,val)
            return FECHA2
        VALOR=actual(FECHA1,gra)
        if VALOR=="-1":
            FECHA2=actualgra(FECHA1,'d','0')
            FECHA2=FECHA2.split(",")
            if len(FECHA2)==2:
                FECHA2=actualgra(FECHA2[0],gra,'0')+","+actualgra(FECHA2[1],gra,'0')
                return FECHA2
            return '-1'
    VALOR=str(VALOR)
    if VALOR=='-1' :return '-1'
    if gra == 'l':
        FECHA2=gra+VALOR
    elif gra == 's':
        FECHA2='s'+VALOR
    elif gra == 'z':
        FECHA2='s'+str(actual(FECHA1,'s'))+'z'+VALOR[0:1]
    elif gra == 'y':
         c=4-len(VALOR)
         FECHA2=c*'0'+VALOR
         FECHA2='y'+FECHA2
    elif 1<CGRANO.index(gra)<6 :
          y=actual(FECHA1,'y')

          if len(VALOR)==1: VALOR='0'+VALOR
          if y<>'-1': FECHA2='y'+str(y)+gra+VALOR
    elif (gra == 'w'):
        y=str(actual(FECHA1,'y'))
        m=str(actual(FECHA1,'m'))
        # semana x del año
        if y<>'-1':
           if m=='-1':
                FECHA2='y'+y+'w'+VALOR
           else:
        #semana x del mes
                FECHA2='y'+str(y)+'m'+str(m)+'w'+VALOR
    elif gra=='d':
        y=actual(FECHA1,'y')
        m=actual(FECHA1,'m')
        if VALOR<>'-1' and y<>'-1' and m<>'-1':
            FECHA2='y'+str(y)+'m'+str(m)+'d'+VALOR
    return FECHA2

def detdatagra(FECHA1,gra,n):
 #determina la fecha a nivel de gra  sumando a la FECHA1 n granularidades
 # Suma a una fecha n granularidades, y la convierte a dias
 # detdata(fecha,'t',0) devuelve el trimestre al que pertenece la fecha
 # supongo que cuando se dicen 2 semanas después es respecto a la fecha de referencia ->sumaré n*7
```

```
FECHA2='-1'
n=int(n)
if gra == 'd' :
    sumagra(FECHA1,"d",n)
else:
    max=0
    a=1
    if gra=='m': max=12
    elif gra=='t' : max=4
    elif  gra=='c': max=3
    elif gra=='e' : max=2
    if gramin(FECHA1)<CGRANO.index(gra):
        FECHA2=operadata(gra,max,n,FECHA1,0)
return FECHA2
```

F. Programa para la obtención automática de tópicos

```
#!\usr\local\bin\python2.0
import bsddb,os,re,glob
from Alib_fechas import *
from string import lower,joinfields

def ana_exp_temp(expresion,todasfechas):
    """Obtiene la fecha o periodo de toda la expresion temporal
    """
    if expresion=='-1': return '-1'
    global FECHA
    global ferr
    global FECHAI
    global PERIODOI
    O=''
    FECHAP=standar_fecha(FECHA)
    expre='-1'
    if expresion[len(expresion)-1]==',':
        expresion=expresion[:(len(expresion)-1)]
    #print expresion
    expresion=re.sub(',+',',',expresion)

    fechas=expresion.split(",")
    X=re.search('^(\D*)',expresion)
    if X:
        expresion0=X.group(1)
        X=re.search('(\+)|(-)|(O)',expresion0)
        if X:
            if X.group(1):O='+'
            elif X.group(2):O='-'
            elif X.group(3):O=''
    if len(fechas)>1:
        X=re.search("^(\D*?(l|s|z|e|c|y|t|m|w|s|d){1}.*)##+(.*)$",expresion)
        if X and not re.search(X.group(3),X.group(1)):
          for i in range(len(fechas)):
           fechas[i]=fechas[i]+"##"+X.group(3)
    LFECHAS=[]
    G=0
    yy=''
    for sent in fechas :
        if len(sent)<2:
            continue
        X=re.search("y(\d{4})\D+",sent)
        if X: yy=X.group(1)[0:2]
        X=re.search("y(\d{2})\D+",sent)
        if X:
            if yy!="": sent=re.sub("y"+X.group(1),"y"+yy+X.group(1),sent)
            else: sent=re.sub("y"+X.group(1),"y19"+X.group(1),sent)
        sent1=extraefecha(sent,O,FECHAI,PERIODOI,FECHA,ferr)
        if sent1<>'-1':
            X=re.search("y_*(\d{4})+",expresion)
            if X and X.group(1) and not re.search("y_*\d{4}",sent):
                if int(X.group(1))< int(sent1[1:5]): sent1=sent1[0]+X.group(1)+sent1[5:]
            G=1
            LFECHAS+=[sent1]
            if not(re.search("\[",sent1)) and re.search("d\d+",sent1) and not re.search('w',sent1) and not re.search("I",sent):
                FECHAI=fecha_standar(sent1)
    if G==1:
        expre=string.joinfields(LFECHAS,",")
        if re.search("I",expresion) and not(re.search("\[",expre)) :
            salida=''
            f1=''
            if not re.search(',',expre) and not re.search("r",expre)  :
             if re.search(FECHA[:4],expre):
                  g=gramin(expre)
                  if 0<=g<3:
                    if g==0:f1=expre
                    else:
                        g=CGRANO[g]
                        v=actual(expre,g)
                        fech=actualdias(expre,g,v)
                        if fech=='-1':f1=expre
                        elif re.search(',',fech):
                          if re.search("\+",expresion):
                            f1=fech.split(',')[1]
                          else:
                            f1=fech.split(',')[0]
                    if re.search(FECHAP,todasfechas) : F=FECHAP
                    else:
                        F=actualdias(FECHAP,'d','-1')
```

```
                    #if not re.search(F,todasfechas): F=''
                if f1<>'' and F<>'':
                    c=DistanciaDias(fecha_standar(f1),fecha_standar(F))
                    if c>0:
                        if c<0: expre=actualdias(F,'d',0)+','+f1
                        else: expre=f1+','+actualdias(F,'d',0)
                    expre="["+expre+"]"
            else:
                 anyo0=int(actual(expre,'y'))
                 anyo1=int(FECHA[:4])
                 if anyo0<>-1:
                     if anyo0<anyo1:
                        expre="["+expre+","+str(anyo1)+"]"
                     else: "["+str(anyo1)+","+expre+"]"
                 else: expre="["+expre+"]"
            else:

                expre="["+expre+"]"
                if re.search('d{1}\d+',expresion) or re.search('w{1}\d+',expresion):
                    expre=expre+','+expre
            if len(LFECHAS)>1 or re.search(",",expre):
                periodo=re.split("\],\[",expre)
                if len(periodo)<2: PERIODOI=periodo[0]
                else: PERIODOI=periodo[0]+']'

        elif len(LFECHAS)==1 and gramin(LFECHAS[0])>0 and gramax(LFECHAS[0])>=6:
            PERIODOI=LFECHAS[0]
        if not re.search(",",PERIODOI) and gramin(PERIODOI)==0:
            print PERIODOI,expre,expresion,"222333",LFECHAS
            raw_input()
    return(expre)


d=os.getcwd()
 d+=raw_input("directorio datos? "+os.getcwd())

Q=glob.glob(d+'*')
m=raw_input('mes?')
if len(m)<2: m=''
ferr=open (d+m+'calfech.err','w')
fevent=open (d+m+'eventos.txt','w')
ftema=open (d+m+'tema.txt','w')
fest=open (d+m+'ambitos.txt','w')

L_FECHAS={"0":{},"1":{},"2":{},"3":{},"4":{},"5":{},"6":{},"7":{},"8":{},"9":{},"10":{}} #lista con las fechas de cada artículo
fechas_vacio=L_FECHAS
if len(Q)>0:
    arti=''
    for q in Q:
      FECHA=re.search("\S*?(1999"+m+"\d+)(\.{1}\D+)?$",q)
      if FECHA:
        FECHA=FECHA.group(1)
        #d='./lola/news/htmlo/'
        f=open (q,'r')
        print d,FECHA
        fout=open (d+FECHA+'event.txt','w')
        a=fecha_standar(sumagra(FECHA,'d','-1'))
        b=fecha_standar(sumagra(FECHA,'d','+1'))
        SCODE=''
        SCODE0=''
        FECHAI=''
        PERIODOI=''
        L_FECHAS={"0":{},"1":{},"2":{},"3":{},"4":{},"5":{},"6":{},"7":{},"8":{},"9":{},"10":{}} #lista con las fechas de cada artículo
        J=f.readline()
        FI=''
        FF=''
        F=''
        K1=[]
        arti=''
        texto=''
        ultima=0
        todasfechas=''
        ambito=''
        while J or ultima==0:
            if not J and ultima==0:
                ultima=1
                J=J0
            if len(J)<2:
                J=f.readline()
                continue
            P=J
            todasfecha=''
            X=re.search('<TEXTSTART:\s*SCODE\s+=\s+\'(.*?(titl){1}.*)>',J)
            if X:
                J0=J
                ftema.write(J)
```

```
SCODE=SCODE0
SCODE0=X.group(1)
ambito0=ambito
FECHAI=''
PERIODOI=''
fechas=''
ambito=''
lista={}
cad=''
if SCODE<>'':
  if L_FECHAS<>fechas_vacio:
    #cad=str(L_FECHAS)
    #print L_FECHAS
    #solo fechas a nivel de dia
    for i in L_FECHAS['0'].keys():
        if  re.search(FECHA[0:4],i):
           lista[fecha_standar(i)]= L_FECHAS['0'][i]
    #puntos a nivel no de dia
    for j in range(1,10):
        for i in L_FECHAS[str(j)].keys():
          if  re.search(FECHA[0:4],i):
            ff=actualdias(i,CGRANO[j],'0')
            if ff=='-1':
                print i,j,'1232'
                raw_input("error días")
                continue
            ff=ff.split(',')
            lista[fecha_standar(ff[0])+'-'+fecha_standar(ff[1])]=L_FECHAS[str(j)][i]
    #intervalos
    for i in L_FECHAS['10'].keys():
        i0=i.split(',')
        if len(i0)<>2:
            print i,"error111"
            i0[1]=i[len(i0)-1]
        p=[]
        for k in [0,1]:
          if  re.search(FECHA[0:4],i0[k]):
            j=gramin(i0[k])
            if j>0:

               k1=actualdias(i0[k],CGRANO[j],'0')
               if k1=='-1':
                   print i,i0,j,k
                   raw_input("error días")
                   continue
               k1=k1.split(',')
               p.append(k1[k])
            else: p.append(i0[k])
        if len(p)==2:
             for k in [0,1]:
               p[k]=fecha_standar(p[k])
             if lista.has_key(p[0]+'-'+p[1]):lista[p[0]+'-'+p[1]]+= L_FECHAS['10'][i]
             else: lista[p[0]+'-'+p[1]]= L_FECHAS[str('10')][i]
    for i in lista.keys():
        j=i.split('-')
        if len(j)==2:cad+='#'+j[0]+','+j[1]+']:'+str(lista[i])
        else: cad+='#'+i+':'+str(lista[i])
  if lista=={}:
      if L_FECHAS==fechas_vacio: j='0'
      else: j='1'
      ambito=a+'-'+FECHA
      fest.write('\n'+SCODE+';'+ambito+'; '+j+';'+'1')
  else:
    fechas=str(L_FECHAS['0'])
    [ambito,lambito]=calculoambito(lista,FECHA)
    if  ambito<>'':
          jj=ambito.split('-')
          d0=DistanciaDias(jj[0],FECHA)
          if jj[0]==jj[1]:
              if not (-15<d0<15 ): ambito=''
          elif not (-15<d0<15 ):
              d1=DistanciaDias(FECHA,jj[1])
              if not (-15<d1<15): ambito=''
    if ambito=='' :
        for i in lambito.keys():
                c=i.split('-')
                if len(c)<>2:
                    raw_input(str(c)+'00')
                if a<=c[0]<=b or a<=c[1]<=b:
                    ambito=i
                    break
        if ambito=='':
                    #print  '---',lambito,L_FECHAS,FECHA,a,b
                    ambito=a+'-'+FECHA
                    fest.write('\n'+SCODE+';'+ambito+';'+cad+';'+fechas+';'+str(lambito)+';3')
        else:
```

```
                fest.write('\n'+SCODE+';'+ambito+';'+cad+';'+str(lambito)+';'+fechas+';'+'4')
            else:
                fest.write('\n'+SCODE+';'+ambito+';'+cad+";"+fechas+';'+str(lambito))
          pp=SCODE
          pp=re.sub("\(","\\\(",pp)
          pp=re.sub("\)","\\\)",pp)
          if cad=='': cad=fecha_standar(FECHA)+":1"
          texto=re.sub(pp,SCODE+" VALUEAMBITO ="+ambito+ " VALUEFECHAS ="+cad+" " ,texto,1)
          if not re.search('-',ambito):raw_input(ambito+'111'+a)
        fout.write(texto)
        texto=''
        L_FECHAS={"0":{},"1":{},"2":{},"3":{},"4":{},"5":{},"6":{},"7":{},"8":{},"9":{},"10":{}} #lista con las fechas de cada artículo
    else:
        P=J
        X=re.search('([^.]*?)(,[^.<]*?,\s*)?(<TIME\s*Value=(\S+)\s*>[^<]*?</TIME>){1}(\s*,*\s*(\S*)(\s*[^\.]*).*)$',P)
        while X:
            B=''
            B1=X.group(4)
            if not re.search("n|R|r|\-|\+|A|F|(s_\w)|(y\d+)|(m\d+)",B1):
              if re.search('(d|m|w){1}\d+',B1) or re.search('0[^\+\-]*(d|m|w){1}$',B1):
                texto0=''
                if len(X.groups())>=1:
                    texto0=X.group(1)
                    texto0=texto0.split()
                    texto0.reverse()
                    texto1=''
                    for p in texto0:
                          texto1+=p+' '
                    texto0=texto1
                    if len(X.groups())>=6:
                        texto0=X.group(6)+' '+texto0
                        if len(X.groups())>7:
                            texto0=texto0+' '+X.group(7)
                    if X.group(2):texto0=texto0+' '+X.group(2)
                    if texto0<>'' :
                          B=buscaverbo(texto0)
                          #parece que los verbos hay que poner - en el presente
                          if B=='0':B='-'
              elif len(X.groups())>=1 and not re.search("\d+",B1):
                    if X.group(2) and len(X.group(2))>3: texto0=X.group(2)
                    else:
                        texto0=X.group(1)
                        texto0=texto0.split()
                        if len(texto0)<2:texto0=X.group(1)
                        else:
                            texto0.reverse()
                            texto0=texto0[0]+' '+texto0[1]
                    B=buscaverbo(texto0)

            elif FECHAI=='' and re.search("r{1}",B1):
                  texto0=''
                  if len(X.groups())>=1:
                      texto0=X.group(1)
                      texto0=texto0.split()
                      texto0.reverse()
                      texto1=''
                      for p in texto0:
                            texto1+=p+' '
                      texto0=texto1
                      if len(X.groups())>=6:
                          texto0=X.group(6)+' '+texto0
                          if len(X.groups())>7:
                              texto0=texto0+' '+X.group(7)
                      if X.group(2):texto0=texto0+' '+X.group(2)
                      if texto0<>'' :
                            B=buscaverbo(texto0)
                            if B=='0': FECHAI=FECHA
                      B=''
            if B<>'' :
                B1=re.sub(',',',',B,B1)
                B1=B+B1
            Fech_ori=ana_exp_temp(B1,todasfechas)
            J=re.sub('TIME>',"TIMEX "+B1+">",J,1)
            if Fech_ori=='-1':
                  #raw_input("11"+X.group(4))
                  Y=re.search("^(\s*\S*\s*)(<even\S*>{1}([^'\.\,\(\)]*))",X.group(4))
                  if Y or re.search("R",B1):
                      J=re.sub('<TIME\s+',"<TIMEX type=EVENTO ",J,1)
                  else:
                      J=re.sub('<TIME\s+',"<TIMEX type=DURATION ",J,1)

            else:
                todasfechas+=Fech_ori+' '
                J00=re.sub("(\+){1}","\\\+",X.group(4))
                J1=re.sub('<TIME(\s+)Value=\s*'+J00,"<TIMEX type=DATE Value="+Fech_ori,J,1)
                if J==J1:
```

```
                        #print "22",X.group(4),"--",B,"-",Fech_ori,"-",FECHAI,",",PERIODOI
                        #if not (re.search("s_\w",Fech_ori)):
                        print Fech_ori,"22"
                        print X.groups()
                        raw_input(J)
                    J=J1
                    G=re.search("([^\.]*?(TIME){1}[^\.]*)",J)
                    if G:
                        ftema.write(G.group(1)+"\n")
                    if not re.search('\[',Fech_ori):
                        l=Fech_ori.split(',')
                    else:
                        l=[]
                        l0=re.split('\]+,+\[+',Fech_ori)
                        for l1 in l0:
                            if re.search('\]|\[',l1):
                                l1=re.sub('\[','',l1)
                                l1=re.sub('\]','',l1)
                                l+=[l1]
                            else:
                                l+=l1.split(',+')
                    for jj in l:
                        if  re.search(',',jj):
                            gra="10"
                            if L_FECHAS[gra].has_key(jj):L_FECHAS[gra][jj]+=1
                            else:     L_FECHAS[gra][jj]=1
                        else:
                          if len(jj)>2:
                            gra=str(gramin(jj))
                            if gra<>'-1':

                                  if L_FECHAS[gra].has_key(jj):L_FECHAS[gra][jj]+=1
                                  else:     L_FECHAS[gra][jj]=1
                            else:
                                    print 'errrr',Fech_ori,jj
                                    raw_input("222" )
                P=X.group(5)
                X=re.search('([^.]*?)(,[^.<]*?,\s*)?(<TIME Value=(\S+)\s*>[^<]*?</TIME>)(\s*,*\s*(\S*)(\s*[^\.]*).*)$',P)
            P=J
            X=re.search("((<TIMEX[^<]*?</TIMEX>[^<]*?)?(<even\S*>{1}([^<\'\.\,)]*)){1})(.*)$",P)
            while X:
                Y=re.search("(<TIMEX([^<]*?)</TIMEX>(\s*\S*\s*)(<even\S*>{1}([^\'\.\,]*)))",P)
                if Y:

                    fevent.write("22"+Y.group(1)+"#"+SCODE0+"\n")
                else:fevent.write("33"+X.group(3)+"#"+SCODE0+"\n")

                P=X.group(4)
                X=re.search("((<TIMEX[^<]*?</TIMEX>[^<]*?)?<even\S*>{1}([^<\'\.\,]*)){1}(.*)$",P)
        texto+=J+"\n"
        J=f.readline()
    f.close()
    fout.close()
fevent.close() ftema.close
fest.close() ferr.close() print "FIN"
```