CENTRO DE AMPLIACIÓN DE ESTUDIOS

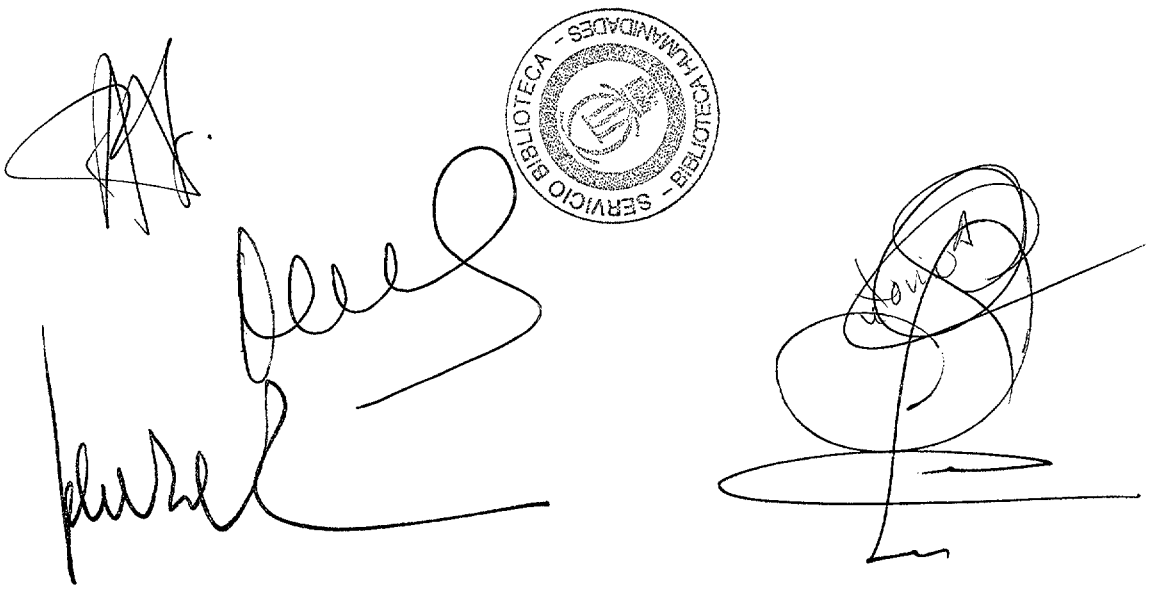
D. BONIFACIO MARTÍN GALÁN
D.N.I. 404.431 E

A U T O R I Z A :

Que su tesis doctoral con el título:
"Tratamiento y difusión en Internet de información jurisprudencial mediante tecnologías XML: aplicación al caso del Tribunal Constitucional" pueda ser utilizada para fines de investigación por parte de la Universidad Carlos III de Madrid.

Getafe, 14 de febrero de 2002.

Fdo.: D. Bonifacio Martín Galán.



UNIVERSIDAD CARLOS III DE MADRID

FACULTAD DE HUMANIDADES, COMUNICACIÓN Y DOCUMENTACIÓN

DEPARTAMENTO DE BIBLIOTECONOMÍA Y DOCUMENTACIÓN

TRATAMIENTO Y DIFUSIÓN EN INTERNET DE INFORMACIÓN JURISPRUDENCIAL MEDIANTE TECNOLOGÍAS XML: APLICACIÓN AL CASO DEL TRIBUNAL CONSTITUCIONAL

TESIS DOCTORAL

(TOMO II)

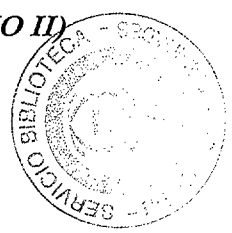
Presentada por:
Bonifacio Martín Galán

Dirigida por:
Prof. Dr. J. Tomás Nogales Flores

Getafe (Madrid), diciembre de 2001

TRATAMIENTO Y DIFUSIÓN EN INTERNET DE INFORMACIÓN JURISPRUDENCIAL MEDIANTE TECNOLOGÍAS XML: APLICACIÓN AL CASO DEL TRIBUNAL CONSTITUCIONAL

(TOMO II)



PARTE IV

TRATAMIENTO PRÁCTICO DE LA JURISPRUDENCIA CONSTITUCIONAL MEDIANTE TECNOLOGÍAS XML

INTRODUCCIÓN A LA PARTE

Esta tesis doctoral se complementa, como se expuso en la Introducción, con un desarrollo práctico de aplicación de diversas tecnologías XML a las Sentencias emanadas del Tribunal Constitucional para la creación de un sitio web de interés para los profesionales y estudiosos del Derecho, así como para el ciudadano común.

Resulta conveniente insistir nuevamente en el hecho de que el desarrollo realizado y aquí expuesto no pretende dar una solución completa y global al tratamiento integral de la documentación jurisprudencial emanada de este Tribunal, tarea ésta más propia de un gran proyecto de investigación aplicada, sino, más bien, ser un primer acercamiento al uso y aplicación de las tecnologías XML para el tratamiento y difusión en la WWW de este tipo de documentos. Aún así, el sistema desarrollado y en perfecto funcionamiento ofrece unas altas prestaciones y capacidades, sirviendo perfectamente de base para una posible y futura ampliación del mismo, tanto en su cobertura documental, como en aquellos aspectos tecnológicos no desarrollados dentro del mismo.

A lo largo de esta cuarta y última parte se presentarán de forma detallada los distintos pasos y procesos que ha sido necesario realizar para llevar a cabo el sistema proyectado. De este modo, se irán explicando todos estos procesos en su sucesión cronológica, desde las primeras fases de recogida y análisis de los documentos jurisprudenciales de dicha institución hasta la publicación y presentación de los mismos en un servidor web XML. Cada una de estas fases ha entrañado el empleo de una tecnología XML particular aumentando con ello la dificultad de su desarrollo y obligando a profundizar en la aplicación práctica de las mismas.

Partiendo del estudio y análisis de la estructura lógica de las Sentencias del Tribunal Constitucional, así como de los contenidos textuales existentes en las mismas, se procederá a establecer una definición formal de dicha estructura mediante el modelo de la DTD de XML. Las serias limitaciones del mismo harán que resulte de mayor interés decantarse por el nuevo estándar del W3C para llevar a cabo esta función, el *XML Schema*. A partir de ahí,

las Sentencias podrán ser marcadas según el nuevo lenguaje creado, ajustándose a una estructura determinada y a unos tipos de datos definidos a tal fin.

La asignación directa de un formato de presentación en la Web de estos documentos a través del lenguaje CSS, aunque interesante, no está exenta de problemas, principalmente procedentes de las limitaciones que los actuales navegadores web tienen para la correcta interpretación de las reglas de estilo creadas mediante dicho lenguaje. Este hecho obligará a replantearnos el modelo inicial deseado y hará que el uso de la tecnología XSLT para la transformación de formatos de documentos electrónicos cobre una especial importancia.

Se tratarán igualmente otros aspectos relacionados con el almacenamiento y gestión de la información contenida en estos documentos, analizando las diversas orientaciones que existen en la actualidad para el desarrollo de bases de datos XML. Frente al modelo netamente orientado a los datos, el propio de las bases de datos relacionales, existe el modelo de base de datos XML nativa, en donde el contenido de los documentos es indizado de forma automática para la generación de ficheros inversos capaces de proporcionar con posterioridad unos buenos resultados en la búsqueda de la información deseada.

Algunas de las tecnologías XML que se han ido exponiendo en la Parte III de esta tesis doctoral serán analizadas desde su lado más práctico, tratando de integrarlas en el sistema desarrollado. En algunos casos esta integración no será posible o, en otros, no se estimará oportuna para un desarrollo básico como es el presente.

Finalmente, todo el producto elaborado será publicado en un servidor web capaz de trabajar con documentos XML, desarrollando para ello los diferentes elementos de interés para la navegación por el sitio web, así como para la localización de las Sentencias del Tribunal Constitucional que pudieran ser de interés para el usuario. A pesar de no ser un producto completamente cerrado, sí que ofrece una visión lo suficientemente amplia para entender y comprobar en la práctica las ventajas aportadas por las tecnologías XML al tratamiento de documentos jurídicos, y, en el caso concreto de esta tesis doctoral, de carácter jurisprudencial.

CAPÍTULO IV.1

DESARROLLO INICIAL

IV.1.1. LÍMITES ESTABLECIDOS AL DESARROLLO PRÁCTICO

IV.1.1.1. DELIMITACIÓN DE OBJETIVOS

Crear un sistema integral de tratamiento y difusión de documentos electrónicos basado en las tecnologías XML es, como ya se ha ido exponiendo a lo largo de la anterior parte de esta tesis doctoral, ciertamente complejo. Los requerimientos profesionales y tecnológicos que ello requiere hacen que, en algunos casos, muchas organizaciones desistan de esta idea aún conociendo los innumerables beneficios que ello les puede reportar. Aunque en principio, la base de esta tecnología no resulta excesivamente difícil de aprender y manejar, no es menos cierto que la gran cantidad de piezas que componen el complejo puzzle de estas tecnologías lo convierten en una tarea que requiere de buenos profesionales procedentes de diversos campos de actuación y con suficientes conocimientos técnicos para enfrentarse a las mismas.

Todo proyecto serio de implantación de tecnologías XML para el tratamiento de documentos de contenido preferentemente textual debería contar con la participación de profesionales procedentes de la Documentación científica, capaces de tratar y analizar adecuadamente los documentos que vayan a ser incorporados al sistema con las operaciones e instrumentos propios de la profesión. Asimismo, han de estar capacitados para poder definir adecuadamente las estructuras lógicas que de los documentos tratados se desprenden a fin de poder establecer con posterioridad las diversas definiciones formales que establecerán el vocabulario correcto para el marcado del documento electrónico. La incorporación de profesionales de la informática a estos proyectos resulta igualmente vital pues serán ellos los que establezcan los requisitos necesarios de hardware y software para llevar a buen puerto el sistema desarrollado. Su participación es igualmente trascendental en el establecimiento del sistema de gestión de la base de datos que ha de almacenar la

información anteriormente tratada, así como la incorporación de los mecanismos necesarios para la interrogación de esta base de datos y otras cuestiones relativas a la publicación de esta información en Internet u otros medios de difusión electrónica. Por último, y no menos importante, es necesaria la presencia en estos equipos de profesionales del campo de actividad desde el que se generan la documentación tratada. En nuestro caso, la incorporación de juristas y otros profesionales del campo del Derecho (personal administrativo y auxiliar, documentalistas jurídicos, responsables de la informatización de las oficinas judiciales, docentes universitarios, etc.) resulta, por tanto, vital para la correcta ejecución de un proyecto de estas características, aportando sus conocimientos en el quehacer jurídico diario y señalando todos aquellos aspectos de vital importancia para el correcto tratamiento de la información jurídica, como son las necesidades reales de información que tienen los juristas, el tipo de datos más relevantes para este colectivo, las formas y métodos de acceso a esta documentación, la normalización de las citas jurídicas incorporadas a los textos, así como un sinfín de cuestiones complejas relativas al tratamiento, difusión y acceso a la información jurídica.

Finalmente, no habría que pasar por alto otro tipo de temas relativos a los aspectos económicos, tecnológicos y los puramente operativos para el desarrollo de un proyecto de estas características.

Resulta obvio por todo lo aquí expuesto, que el desarrollo práctico que se ha elaborado en esta tesis doctoral y se expondrá a continuación, no pretende alcanzar tales cotas de proyección pues el objetivo a alcanzar no es ése, sino bien, , como ya se comentó en la introducción, es hacer un primer acercamiento al tratamiento de los documentos jurisprudenciales emanados del Tribunal Constitucional mediante el uso de las tecnologías XML que demuestre a los colectivos profesionales jurídico, documental e informático los beneficios que estas tecnologías aportan para el tratamiento y difusión de tales documentos.

De igual forma, existen otros objetivos parciales que este desarrollo práctico espera cubrir, tal y como fue anunciado en los prolegómenos de la tesis. Entre ellos, y con el fin de volver a remarcarlos, estarían los siguientes:

- Analizar el panorama tecnológico actual en la aplicación de las tecnologías XML a los textos jurídicos, y de forma especial a los textos jurisprudenciales emanados del Tribunal Constitucional, estableciendo el estado actual de la cuestión, así como la viabilidad real de dichas tecnologías con los medios informáticos con los que actualmente se cuenta para llevar a cabo esta labor.
- Analizar los textos jurisprudenciales del Tribunal Constitucional, dilucidando la estructura de composición de los mismos y aquellas áreas susceptibles de ser destacadas mediante el marcado XML, para su posterior difusión y recuperación mediante las tecnologías de Internet.
- Continuar el proceso evolutivo en las investigaciones que se han venido desarrollando dentro de este Departamento desde hace ya varios años en la aplicación de los lenguajes de marcado usados en la *World Wide Web* a los textos jurídicos, consolidando dichas investigaciones con las tesis doctorales puestas en marcha sobre esta materia.
- Servir de base conceptual, metodológica y tecnológica para futuros proyectos de investigación aplicada que a buen seguro han de surgir en años venideros en colaboración con otras instituciones del ámbito jurídico.
- Y, en definitiva, seguir consolidando el grupo de Tecnologías de la Información del Departamento de Biblioteconomía y Documentación de la Universidad Carlos III como uno de los centros de excelencia a escala nacional en el tratamiento y difusión de documentos electrónicos procedentes de diversos campos de producción mediante el uso de las tecnologías XML.

Por todo ello, el desarrollo práctico expuesto en esta tesis doctoral no tratará de dar una solución global al complejo tema de la informatización de las oficinas judiciales de cualquier Tribunal o Juzgado de nuestro país, cuestión ésta, aunque sumamente interesante, enmarcada dentro de la informática de gestión. Este desarrollo tan sólo tiene el propósito de aportar una experiencia en el tratamiento de los documentos jurisprudenciales del Tribunal Constitucional mediante las tecnologías XML, proponiendo una serie de mecanismos considerados de utilidad para el marcado de los textos contenidos en dichos

documentos mediante la definición formal de un tipo documental establecido (desarrollo de una DTD y un Esquema XML).

Esta definición formal del tipo documental analizado (Sentencias del Tribunal Constitucional), como se verá en apartados posteriores, no pretende de igual forma sentar *cátedra* o constituirse en una propuesta definitiva y normalizada del vocabulario de elementos a emplear para el marcado XML de documentos jurisprudenciales. De hecho, como se observará en la propia DTD y, en mayor grado, en el esquema XML elaborados, no se hace uso ni mención a un nombre local (*namespace*) del Tribunal Constitucional español pues ello supondría que se cuenta con su aceptación para que el modelo aquí planteado pudiese constituirse como norma de marcado para sus resoluciones.

Se trata única y exclusivamente de un primer acercamiento, de una propuesta lanzada desde aquí para la definición estructural y semántica de este tipo de documentos jurídicos con el ánimo de poder establecer futuros debates en cuanto a su validez y contrastarla con otros desarrollos similares en los que ya se está trabajando en distintas partes del mundo, especialmente en organizaciones internacionales dentro de este campo de actuación y en diversos Tribunales de Justicia.

De igual modo, se pretende ofrecer un breve estado de la cuestión en la aplicación práctica de otras tecnologías XML asociadas al núcleo principal de la misma, estableciendo las posibilidades actuales y reales de utilización e incorporación, entre ellas las que posibilitan, el establecimiento de formatos de presentación o estilo para los documentos electrónicos (CSS y XSL), la incorporación de enlaces hipertextuales potentes y versátiles (XLink), la asociación de metadatos a estos documentos (RDF), la transformación necesaria de los documentos XML a otros formatos más adecuados de publicación (XSLT) y la interrogación y localización de información precisa dentro de estos documentos (XQuery).

Con todo ello analizado y desarrollado en el plano práctico, el producto obtenido se incluirá dentro de un servidor Web, capaz de tratar y servir documentos XML, para su publicación en el espacio electrónico de Internet y consulta por parte de los profesionales interesados en este desarrollo.

IV.1.1.2. DELIMITACIÓN DOCUMENTAL Y ESPACIO-TEMPORAL

No pretendemos ser exhaustivos en la incorporación y tratamiento de documentos jurisprudenciales en esta aplicación práctica desarrollada pues su finalidad no es, la construcción de una base de datos de información jurisprudencial completa y actualizada. Esta labor supondría, cuanto menos, un esfuerzo titánico, abordable únicamente desde la base de un gran proyecto de investigación aplicada en la que, como decíamos con anterioridad, interviniesen diversas instituciones y profesionales de cada uno de los campos de actuación puestos en juego. Por ello, y dados los objetivos que se pretenden cubrir, sirva el conocido dicho de *como muestra, un botón* para delimitar claramente la cobertura documental de lo aquí tratado.

Dado que sólo las Sentencias que emanan del Tribunal Constitucional, de cualquiera de sus Salas o del Pleno, constituyen la jurisprudencia del mismo, nos ceñiremos a este tipo concreto de documentos dejando, por tanto, de lado los otros tipos de documentos en que se plasman los dictámenes producidos en este órgano: Providencias y Autos.

Dentro del conjunto documental formado por las Sentencias de este Tribunal se hará un estudio riguroso de la estructura de los elementos de que se componen, analizando para ello cada uno de los grandes bloques en los que se divide el texto, así como las diversas subdivisiones jerárquicas que se pueden encontrar dentro de los mismos. Igualmente, resultará fundamental para nuestro estudio el orden de aparición de cada uno de estos bloques y sus correspondientes subdivisiones a fin de determinar las posibles variaciones que se pudieran producir en la muestra de documentos analizados.

Para proceder a esta primera fase de análisis documental resulta obvio que en primer lugar hay que establecer la fuente y el mecanismo para la obtención de los textos de dichas Sentencias del Tribunal Constitucional. En la primera parte de esta tesis doctoral se analizaron las diversas bases de datos comerciales existentes en España con información

jurisprudencial emanada de este Tribunal, así como los actuales servicios y portales jurídicos en la WWW que incorporan y distribuyen esta información de manera gratuita.

La Biblioteca de la Universidad Carlos III de Madrid proporciona acceso a las principales bases de datos comerciales de contenido jurídico analizadas en esta tesis doctoral, por lo que hubiese sido factible la recuperación de los documentos jurisprudenciales para su análisis. Estas de bases de datos jurídicas fueron empleadas en un primer momento para contrastar los diversos mecanismos empleados por cada uno de ellos en cuanto al análisis y tratamiento documental a los que se ven sometidos los documentos allí incorporados, los campos o bloques considerados como imprescindibles en los que el documento jurisprudencial podría ser dividido y de interés para una posterior recuperación, así como para otras cuestiones relativas a la presentación de estos documentos jurídicos. De igual forma, fueron usadas cuando fue necesario dilucidar la asignación de descriptores de materia a los documentos y la normalización a la que habían sido sometidas las citas jurídicas contenidas en las Sentencias del TC.

Esto mismo sucedió con los servicios y portales jurídicos existentes en Internet, analizando aspectos relacionados con su cobertura documental y temporal, los campos en los que el texto de las sentencias ha sido estructurado para su posterior recuperación, así como, y dado que nos encontramos en un espacio electrónico con capacidad hipertextual, el establecimiento de relaciones de este tipo, si las hubiera, entre los documentos que estas bases de datos jurídicas en la Web contienen. Esto mismo es aplicable a la información jurisprudencial que el Boletín Oficial del Estado pone de forma gratuita a disposición de los ciudadanos para su consulta a través de la web. Pero el principal inconveniente que nos encontramos en este caso es el hecho de que no se trata de texto electrónico sino de una imagen digital de las páginas de dicho Boletín, cuestión ésta que complica sobremanera su utilización y procesamiento, amén de la incomodidad del proceso de descarga de estos ficheros de imagen digital página a página hasta completar el texto completo de la Sentencia.

Pero fueron, sin duda, los documentos electrónicos existentes en el sitio Web del TC los que más se emplearon para el desarrollo de esta parte práctica de la tesis doctoral, constituyendo el espacio del cual se obtendría la Sentencia de partida desde la cual iniciar

todo el proceso. Esta decisión se debió en gran medida a una serie de factores como la fiabilidad de los textos que, aunque en este sitio web se advierte expresamente de su invalidez jurídica en el caso de existir discrepancias con lo plasmado en los medios impresos tradicionales (en concreto, con lo publicado en el BOE), ofrecían las suficientes garantías para nuestro desarrollo. Igualmente se tuvieron en cuenta para tomar esta decisión la gratuidad en el acceso a dichos textos electrónicos, así como el hecho de tratarse de una sencilla conversión de formatos desde el documento de MS-Word, en el que originalmente se redactan las sentencias, al formato HTML para su publicación en la Web, donde no se incorporan, por tanto, otros elementos textuales añadidos (fruto de las labores de tratamiento y análisis documental a los que se ve sometido el texto y que, en otros sistemas analizados, quedan incorporados y reflejados al principio del mismo). Todas aquellas Sentencias citadas en el documento de partida que no estuviesen contenidas en este sitio web serían tomadas de la base de datos de Jurisprudencia Constitucional publicada por el BOE.

El conjunto documental de Sentencias marcadas mediante XML en el desarrollo práctico que presentamos es ciertamente reducido pues, como decíamos, no se trataba de crear un hábeas documental amplio que diese solución a las posibles necesidades informativas de juristas, investigadores o, simplemente, interesados en la jurisprudencia constitucional. Además, el verdadero interés de esta tesis doctoral y, por extensión, del desarrollo práctico aquí contemplado se situaba en la definición formal de la estructura lógica a la que se ajustan las Sentencias del TC, así como en otras cuestiones relativas al empleo de tecnologías XML asociadas, como ya se ha venido exponiendo reiteradamente a lo largo de esta tesis doctoral. Una vez establecidas todas estas cosas, el marcado de los documentos en formato electrónico mediante el vocabulario XML resulta ya una cuestión, en buena medida, meramente mecánica (salvando las actividades intelectuales relativas a la asignación de descriptores de materias y normalización de otros aspectos), simplemente limitada por la disponibilidad del personal y el tiempo necesario para llevar a cabo esta tarea.

Con todo ello, se ha procedido, tomando como base inicial de este desarrollo una Sentencia dictada por el Tribunal Constitucional en el año 2000 (la Sentencia 1/2000, de 17 de enero), a marcar su contenido con las marcas establecidas en la definición formal del tipo de documento y a realizar las otras tareas asociadas a la misma, como la asignación de características de presentación o estilo, de relaciones hipertextuales, de metadatos y así como el mecanismo de transformación automática del documento XML al formato HTML para su difusión en la Web. Debido a que este sistema hace uso de la tecnología de los hipertextos en la Web, los vínculos que desde esta Sentencia se pueden establecer con otras disposiciones legales y judiciales son ciertamente numerosas. Como se expondrá en el apartado relativo a la asignación de enlaces hipertextuales a las Sentencias del TC, resultaba del todo imposible extender éstas a muchos de los documentos jurídicos citados en la misma pues ello nos hubiera obligado a definir y marcar de igual forma otros tipos de documentos jurídicos no contemplados en esta tesis doctoral, amén de llegar *ad infinitum* en dicha labor. Por ello, tan sólo se desarrollarán hasta un primer nivel de profundidad algunas de las Sentencias citadas dentro del texto de la Sentencia de partida (las tres primeras que aparecen contenidas en el apartado de Antecedentes y las tres primeras que aparecen contenidas en el apartado de Fundamentos Jurídicos), tal y como se verá en posteriores apartados de este desarrollo práctico.

Las Sentencias del TC tratadas en esta apartado práctico de la tesis doctoral constituyen, por tanto, un total de siete documentos electrónicos marcados mediante el lenguaje XML, comprendidos entre los años 1982 y 2000.

Sin embargo, el conjunto documental del sitio web XML desarrollado incluirá, como se irá viendo a lo largo de esta la exposición de esta cuarta y última parte de la tesis doctoral, más de 30 documentos electrónicos de diversos formatos, según la tecnología XML empleada en cada caso (DTD, XML *schema*, XSLT, CSS). A esta cifra habría que añadir todos aquellos documentos XML tomados como banco de pruebas de otras tecnologías finalmente no utilizadas (XLink y RDF) y que, por tanto, no han sido integrados dentro del mismo.

IV.1.1.3. DELIMITACIÓN TECNOLÓGICA

Las limitaciones tecnológicas del desarrollo práctico llevado a cabo en esta tesis doctoral se pueden agrupar den dos grandes categorías: las derivadas de las propias tecnologías XML y las relacionadas con los requerimientos y disponibilidad tanto de hardware como de software para este desarrollo.

En cuanto a las limitaciones propias de las tecnologías XML, éstas vienen en un primer momento asociadas a la falta de estandarización definitiva de algunos de los denominados estándares XML asociados. Si bien es cierto que el núcleo de las tecnologías XML está fuertemente asentado (especialmente, la sintaxis XML y el modelo de Definición de Tipo Documental), otros estándares asociados han sido consolidados como Recomendación del W3C sino en fechas recientes o, en algunos casos, aún siguen en fase de estudio (caso, por ejemplo del estándar XForm para la creación de formularios en documentos XML). Esta situación nos ha exigido un esfuerzo constante en el conocimiento, comprensión y manejo de cada una de las tecnologías XML puestas en juego, así como el seguimiento en la evolución normativa y tecnológica de aquellas otras que han ido variando en su especificación a lo largo de estos últimos años.

Todos aquellos profesionales que se han venido introduciendo en el mundo de las tecnologías XML en estos últimos años, o los que se han incorporado al mismo recientemente, saben de las dificultades que entraña el conocimiento y manejo de tan abundantes y complejas tecnologías. Si la especificación primitiva de 1998, la cual constituía el núcleo de estas tecnologías, no requería grandes esfuerzos para su comprensión y puesta en aplicación, con la aparición de los diversos estándares asociados que fueron surgiendo desde entonces hasta nuestros días, el dominio completo de todas estas tecnologías se ha vuelto algo prácticamente inalcanzable sin la ayuda de especialistas en cada uno de dichos desarrollos. Aún así, desde esta tesis doctoral se ha pretendido ofrecer y aplicar todas aquellas tecnologías XML consideradas necesarias para el logro de los objetivos inicialmente propuestos.

Pero, como ya se ha ido exponiendo a lo largo de diversos capítulos de esta tesis, el acento de la misma se ha puesto en los mecanismos existentes en XML para la definición estructural de los mecanismos, esto es, los modelos para el reconocimiento de las estructuras lógicas subyacentes en los documentos y su definición formal a través de la DTD y del Esquema XML. Esto es debido a diversos factores tales como los propios intereses y la importancia asignada por parte del doctorando y de su director de tesis a este tipo de aplicaciones, consideradas como fundamentales ante cualquier intento de abordar el tratamiento de documentos electrónicos de cualesquiera tipo y condición mediante el uso de las tecnologías XML y, de igual forma, la existencia de otras tesis doctorales en fase de elaboración en el mismo Departamento que inciden de manera específica sobre algunas de las tecnologías XML asociadas, caso de XLink y de RDF.

En cuanto a las limitaciones correspondientes a los requerimientos y disponibilidad del hardware y del software necesario para el correcto tratamiento de los documentos XML, la situación es aún si cabe más compleja.

Como ya se ha expuesto, la tecnología XML se concibe desde sus orígenes como una tecnología abierta, libre y de acceso gratuito para todos aquellos que deseen emplearla. Este hecho garantiza tanto su gratuidad de uso como la posibilidad de ser implementada en cualquiera de los actuales sistemas operativos, así como en cualquier ordenador con unas mínimas prestaciones técnicas. De igual modo, a lo largo de estos años han venido apareciendo una serie de programas informáticos capaces de tratar de forma adecuada los documentos XML, tanto de forma global como en algunos aspectos específicos de los mismos (analizadores sintácticos, validadores, editores, servidores Web, etc.).

El principal problema procede, como sucede en cualquier otro campo de aplicación de la informática, del hecho de que los programas más interesantes y potentes para el trabajo con las diferentes facetas que se engloban dentro de las tecnologías XML no resultan gratuitos, sino que, más bien todo lo contrario, son notablemente caros. Este hecho es debido al mercado al cual se orientan, principalmente a las grandes compañías e instituciones que tienen presencia en la Web y que desean por diversas razones tener un completo y eficaz sistema de almacenamiento, gestión e intercambio de datos. De este modo, estas

herramientas profesionales para el tratamiento de documentos XML resultan inalcanzables económicamente desde las premisas de las que parte el desarrollo práctico que estamos exponiendo. Ello no ha sido óbice para que aquellos productos informáticos considerados de interés para esta tesis no hayan sido analizados y evaluados, siempre y cuando dichos programas ofreciesen un tiempo gratuito de prueba (programas *shareware*), como se comentará dentro de cada uno de los apartados específicos de esta última parte de la tesis doctoral.

Es necesario aclarar, igualmente, el soporte real que muchas de estas aplicaciones dan en la actualidad al tratamiento de documentos XML. Hoy en día se puede afirmar de forma categórica que no existe ningún producto informático que haga un tratamiento completo y adecuado de todas las tecnologías XML. Si bien algunos funcionan perfectamente para la tarea que tienen encomendada (los programas específicos que tratan una única tecnología XML), no resultan igualmente satisfactorios aquellos otros que cubren un gran número de tecnologías pero, a menudo, de forma incompleta. Aunque esto se suele dar en todo tipo de aplicaciones, los mayores problemas los encontraremos en el soporte que los actuales navegadores web ofrecen a la tecnología XML, siendo en muchos casos, como ya se señaló con anterioridad en capítulos precedentes, bastante reducida, lo que ha condicionado en gran medida la elección del modelo de tratamiento al cual se han de ver sometidos los documentos XML en este apartado práctico.

Por todo ello, se ha tratado de emplear en la medida de lo posible programas informáticos XML gratuitos y de libre distribución (programas *freeware* o, en otros casos, *adware*) que permitan obtener los mejores resultados posibles para el tratamiento, de forma global o específica para cada tecnología XML, de los documentos electrónicos que se han manejado en este desarrollo práctico. Algunos de estos productos no dejan de ser realmente interesantes y potentes a pesar de su gratuidad, constituyendo en algunos casos la mejor alternativa dentro del amplio abanico de programas existentes, tanto comerciales como de libre distribución. Este ha sido el caso, por ejemplo, del servidor Web para documentos XML conocido por el nombre de Cocoon, como veremos en su momento.

Señalaremos, por último, en este primer apartado de limitaciones puramente tecnológicas, la existencia en Internet de servicios web orientados a proporcionar información sobre los desarrollos informáticos que existen en la actualidad para el trabajo con documentos XML, tanto de forma global como por áreas de aplicación y desarrollo de estas tecnologías. De forma general, todos los grandes portales o servicios de XML que existen en la Web incluyen algún apartado dedicado a tratar de forma más o menos detallada estas cuestiones. Sin ánimo de ser exhaustivos en el siguiente listado, no queremos pasar por alto el señalamiento de algunos de ellos, como son el caso del sito *XMLSOFTWARE*[1], creado y mantenido por James Tauber y otros investigadores, ya mencionado con anterioridad en el apartado III.4.4., el apartado dedicado al análisis de estos programas dentro del sitio *XML.com*[2], la inmensa lista comentada y perfectamente actualizada de software tanto para SGML como para XML, denominado *Public SGML/XML Software*[3], elaborada por Robin Cover, el directorio *Free XML tools and software*[4], iniciativa de Lars Marius Garshol o, finalmente, la relación establecida por el propio W3C dentro de la página dedicada al desarrollo de las tecnologías XML[5] .

Un segundo apartado de limitaciones puramente tecnológicas vendría constituido por los problemas surgidos para la implementación de este sistema en algunos de los cuatro ordenadores de los que dispone nuestro Departamento de apoyo a la docencia y a la investigación, capacitados para trabajar como servidores de información (tres de ellos corresponden a máquinas SUN con sistema operativo Solaris dentro del Laboratorio de Informática del Departamento y la otra, una Hewlett-Packard con sistema operativo NT dedicada a dar soporte a los PCs del Aula Informática propia del Departamento). Sin

[1] http://www.xmlsoftware.com/

[2] http://www.xml.com/

[3] http://www.oasis-open.org/cover/publicSW.html

[4] http://www.garshol.priv.no/download/xmltools/

[5] http://www.w3.org/XML/

embargo, diversos problemas surgidos en relación con estos equipos que no viene al caso detallar nos hicieron replantearnos la instalación del software necesario y el desarrollo de esta aplicación práctica sobre las citadas máquinas hasta fechas posteriores en las que todos estos problemas pudieran estar solventados. Con todo ello, el sistema se ha tenido que instalar en los ordenadores personales existentes en los respectivos despachos, así como en los de uso particular. Este hecho ha provocado serias limitaciones para el desarrollo del sistema pensado en origen debido a las reducidas capacidades de procesamiento, así como a la necesidad de tener que localizar software específico (en especial el que ha de ejercer de servidor web y de gestión de los datos) para el sistema operativo instalado en estos equipos (Windows 95/98/Millenium). Estas condiciones, lógicamente, no son las más adecuadas para llevar el peso de estas tareas.

En cualquier caso, el sistema ha sido probado en diversos ordenadores personales, y de forma especial en un ordenado portátil Compaq Presario (64 MB de memoria RAM, 6 GB de disco duro y procesador Pentium Celeron 600 Mhz) con sistema operativo Windows Millenium, lo que al menos nos ha permitido transportarlo cuando ha sido necesario.

IV.1.2. ESQUEMA GENERAL Y FASES DEL SISTEMA DESARROLLADO

Antes de comenzar a detallar cada uno de los aspectos abordados en el desarrollo de la aplicación práctica de la tesis doctoral, resulta conveniente hacer una breve descripción del sistema en su conjunto. Ello facilitará con posterioridad una mejor comprensión de cada una de las piezas que han entrado en juego y que se exponen de forma pormenorizada.

Dentro de este desarrollo práctico que se ha considerado necesario realizar en esta tesis doctoral sería más apropiado hablar de dos sistemas planteados, ciertamente distintos en su filosofía, como se irá viendo a lo largo de esta exposición.

1. PRESENTACIÓN DIRECTA DE LOS DOCUMENTOS XML

Un sistema de tratamiento y difusión de documentos XML en Internet se puede hacer desde una perspectiva netamente *documental*, que es la que en un principio se ha deseado adoptar. Esto es, existe una presentación directa del documento XML (junto con su hoja de estilo asociada) en la ventana principal del navegador web del usuario. Desde esta perspectiva, la construcción de un sistema XML de este tipo no resulta una tarea excesivamente compleja dado que las tecnologías web empleadas no son numerosas y, en cierta medida, son sencillas de manejar: los documentos XML marcados según una determinada DTD o esquema; una hoja de estilo CSS para asignar el formato de presentación a dichos documentos; un servidor Web para la publicación de los mismos; un software que nos permita la indexación del texto y un motor de búsqueda integrado en el sistema que permita al usuario realizar búsquedas complejas de información según el

contexto estructural en el que se encuentren los datos. Esta es la visión u orientación al texto de XML, de la cual hemos venido hablando a lo largo de este trabajo científico.

El siguiente gráfico ilustra de forma simple y general el funcionamiento del sistema desde este planteamiento cuando un documento XML es solicitado por el usuario:

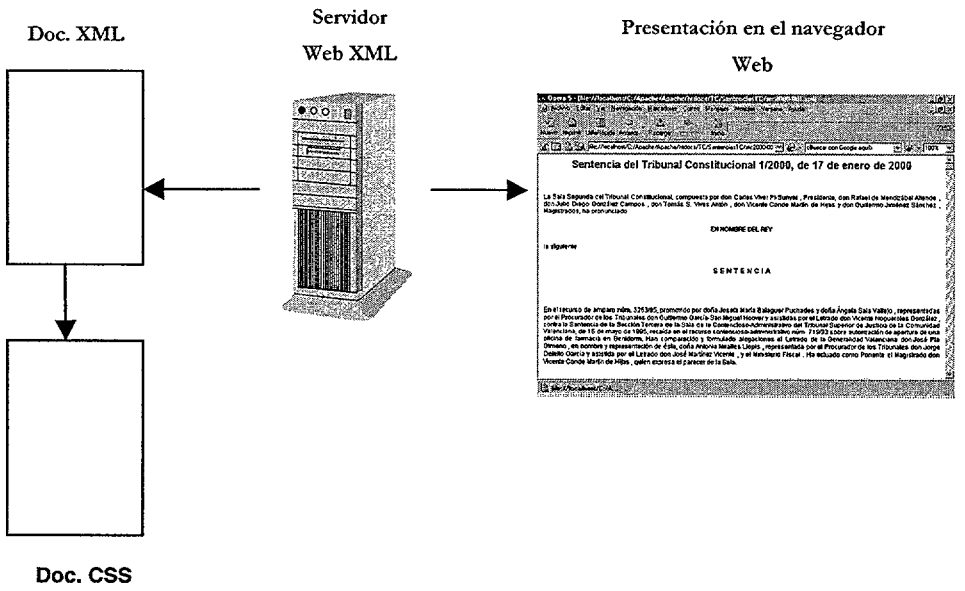


Figura IV.1: Diseño general del desarrollo práctico desde su orientación al texto.

2. TRANSFORMACIÓN DEL DOCUMENTO XML AL FORMATO HTML

Antes de explicar este segundo modelo de tratamiento de los documentos XML de nuestro desarrollo práctico veamos el siguiente gráfico que, al igual que antes, ilustra de forma sencilla el funcionamiento del sistema:

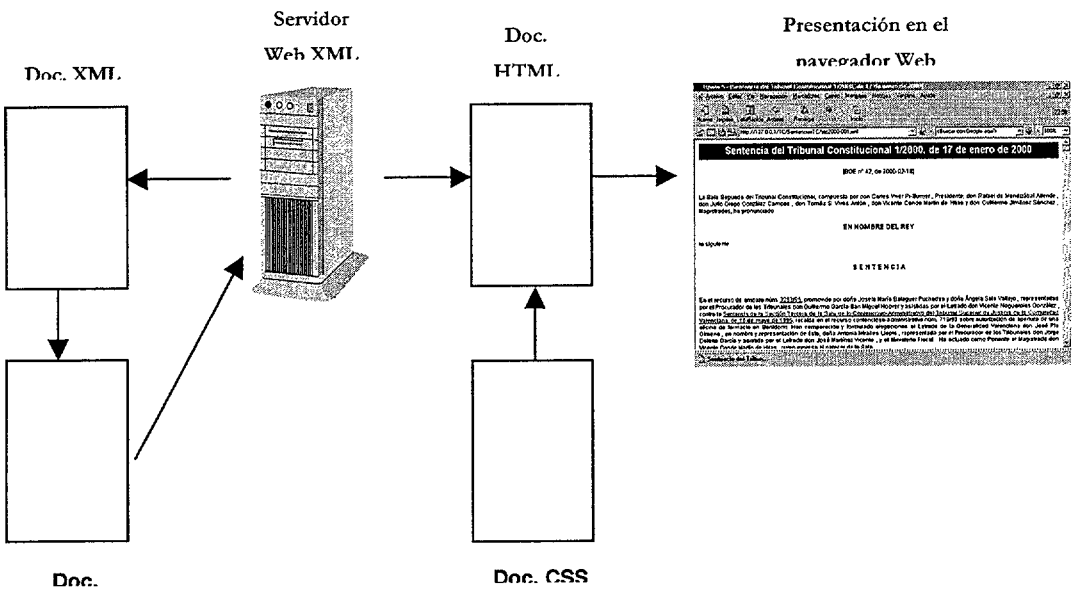


Figura IV.2: Diseño general del desarrollo práctico desde su orientación a los datos.

La explicación a grandes rasgos de las líneas básicas del funcionamiento de este modelo de transformación de los documentos XML al formato HTML es la siguiente:

- Los documentos XML que contienen las sentencias del TC son almacenados en el ordenador en el cual está instalado el servidor Web XML junto con otros documentos necesarios para realizar la transformación del formato (documentos XSLT, además de otros que serán detallados posteriormente).

- Ante una solicitud de presentación de un documento XML por parte del usuario, el programa servidor se encargará de efectuar la conversión de este documento al formato HTML, interpretable por cualquier navegador web existente en la actualidad. De este modo, el servidor genera y presenta un documento HTML *virtual* (sólo existe de forma temporal en la ventana del navegador del usuario, mientras está siendo consultado).
- Este documento HTML virtual lleva asociado desde el momento de su transformación una hoja de estilo CSS en la cual se detalla el formato de presentación asignado a cada elemento del documento HTML (fuente del texto, tamaño de los caracteres, colores, comportamiento hipertextual, etc.).

- Finalmente, es este documento HTML virtual, con una hoja de estilo externa asociada al mismo, el que se le presenta al navegador web del usuario que ha solicitado la presentación del documento XML inicial.

Este modo de proceder no está carente, como hemos venido señalando, de cierta polémica pues son diversas las teorías y tendencias que en la actualidad existen sobre el modo en el cual los documentos XML han de ser procesados y presentados ante el usuario. Dicha polémica, aunque ya se comentó algo al respecto en el Capítulo III.3. de esta tesis doctoral, se sitúa en la misma base de la concepción y del uso que de las tecnologías XML se puede hacer; esto es, XML orientado de forma exclusiva al tratamiento de datos frente a XML orientado al tratamiento de documentos con contenido principalmente textual.

Desde la óptica del tratamiento de datos resulta obvio y justificable que XML sólo sirva de marco en el cual se han de insertar éstos para su almacenamiento y gestión por parte de los sistemas de bases de datos relacionales capacitadas para trabajar con estas tecnologías. De igual forma, la base de datos es capaz de generar nuevos documentos XML a partir de ciertos datos contenidos en ella. En este caso, los datos contenidos en los documentos XML de salida que el sistema genera han de ser transformados a otro formato más adecuado para la presentación ante el usuario, formato que normalmente suele ser el HTML (aunque también resulta frecuente su conversión al formato PDF). De este modo, se consigue mantener perfectamente la independencia de los datos frente a la forma en la cual se desea que se presenten los mismos, normalmente insertos dentro de tablas, cuadros u otro tipo de marco que le sea de interés a la organización que ha de suministrar dicha información[6]. De igual forma, se garantiza que el usuario que ha de consultar la información suministrada a través de la Web no tenga ningún tipo de problema con la visualización de dicha información pues esta se encuentra ahora en un formato electrónico, el HTML, interpretable correctamente por la mayoría de los navegadores web.

Desde el punto de vista de los que defienden que defienden los modelos de sistemas XML orientados al texto, corriente de opinión procedente del ámbito del tratamiento de textos electrónicos mediante los lenguajes de marcado (principalmente, del campo de aplicación de SGML), estos procesos intermedios para el suministro de documentos XML no deberían ser, en este contexto de producción documental, exigibles. El contenido eminentemente textual que se encuentra marcado en estos documentos XML es el que se desea presentar al usuario y, por tanto, no se debería exigir ninguna transformación de formato, tan sólo la asignación de características de presentación de los elementos señalados en el marcado mediante la correspondiente asociación de una hoja de estilo (CSS o XSL). El problema no estaría, por tanto, en las tecnologías XML, pues éstas satisfacen

[6] Un símil bastante oportuno es lo que se produce con sistemas de bases de datos tan extendidos entre los usuarios actuales, como es el MS Access. Los datos pueden ser presentados de múltiples formas a través del modelo de informe que más le interese al usuario. La presentación cambia según nuestros deseos pero los datos se mantienen siempre inalterables.

perfectamente las necesidades de definición estructural, de contenidos y de presentación de la información, sino, como señalábamos, en el escaso respaldo que la mayoría de los actuales navegadores web ofrecen a las tecnologías XML (errores en los programas analizadores y validadores de documentos XML, escasa interpretación del lenguaje CSS y casi nula en el caso de XSL, falta de soporte para otros estándares como XLink, XForm, etc.).

Siendo así, no se trata de cambiar parte de la filosofía heredada de SGML en la tecnología XML y sí, por el contrario, de presionar a las diversas compañías en el desarrollo de estas herramientas informáticas para que desarrollen navegadores capaces de interpretar íntegra y correctamente los documentos XML y las diversas tecnologías asociadas a los mismos.

En nuestro caso, se ha tratado de combinar ambas posturas desarrollando, como se verá con posterioridad, propuestas que recojan las principales ideas procedentes de uno y otro lado. Ante todo, este desarrollo práctico ha querido ser un banco de pruebas de todas estas orientaciones que puedan ser de utilidad para futuros proyectos de investigación en las que se apliquen las tecnologías XML.

Lamentablemente, y debido a los problemas mencionados por parte de los actuales navegadores web, se ha tenido que optar por el segundo modelo, la transformación de formato, a fin de facilitar la correcta visualización en el espacio electrónico de la WWW de la información contenida en los documentos XML. Sin embargo, con esta decisión hemos podido obtener algunas ventajas añadidas a nuestro sistema, como son la posibilidad de presentar la información contenida en los documentos XML de diversas formas (texto completo, documento abreviado, generación de listados, etc.) sin tener por ello que confeccionar manualmente nuevos documentos XML.

El anterior gráfico general se puede detallar aún más para dar una visión mucho más exacta de las cuatro fases seguidas en el proceso de construcción del sistema desarrollado. Se han tomado como muestra dos documentos XML (Doc1.XML y Doc2.XML) pero el modelo, lógicamente, se aplicaría a todo el conjunto de documentos XML que se integraran en el sistema. Gráficamente representadas, estas cuatro fases serían las siguientes:

Figura IV.3: Esquema pormenorizado de las Fases 1, 2 y 3 del desarrollo práctico.

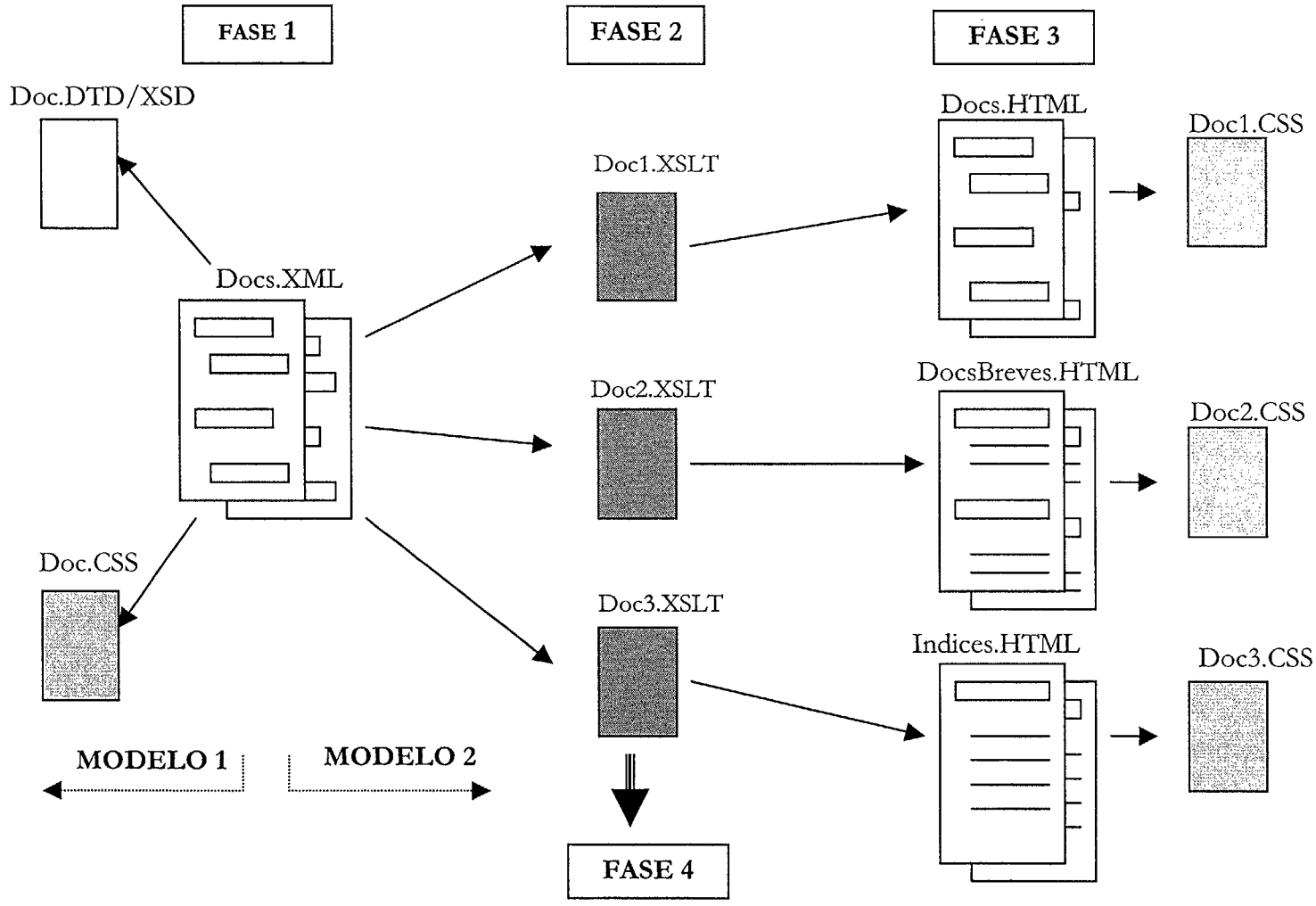
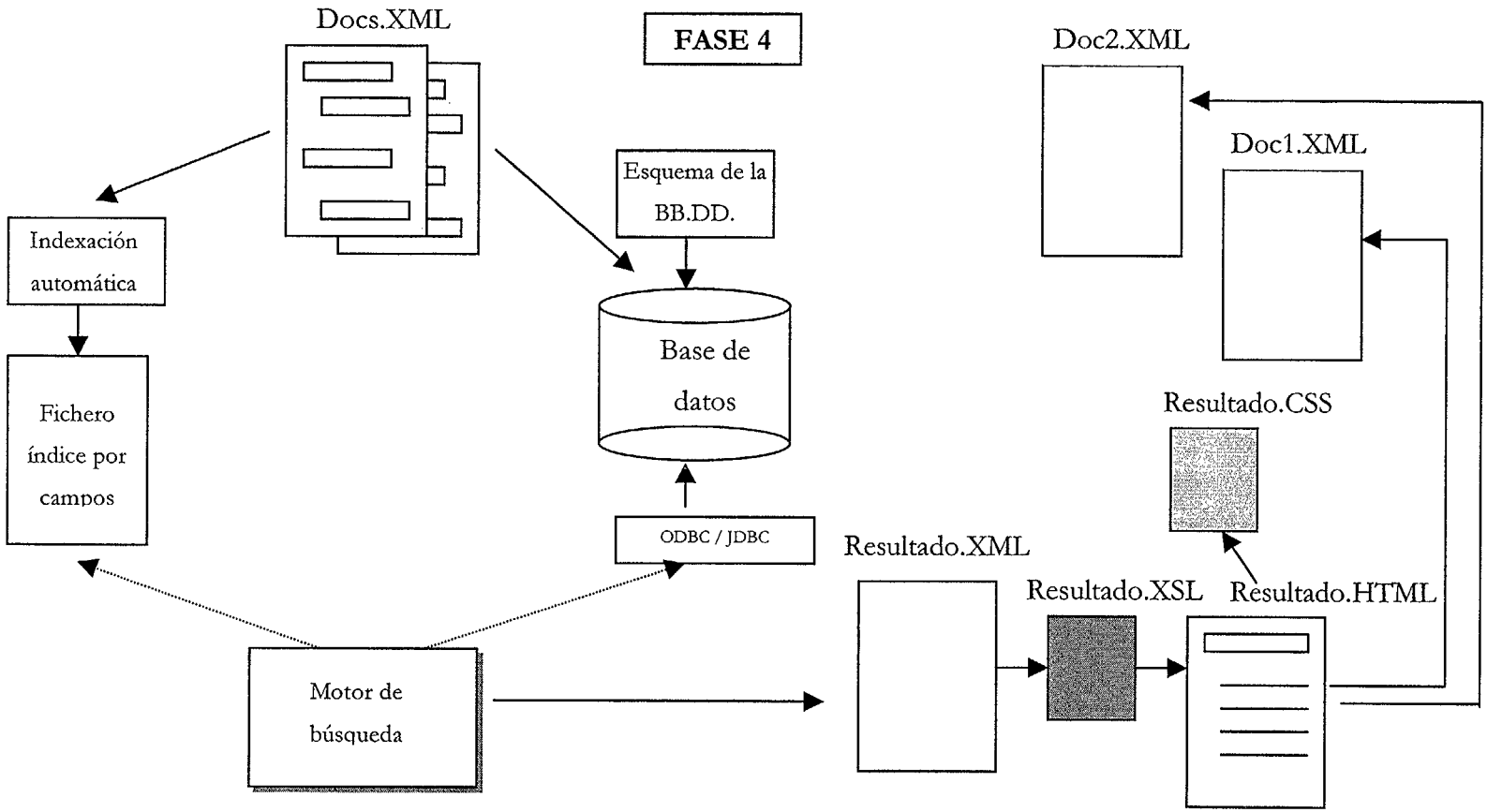


Figura IV.4: Esquema pormenorizado de la Fase 4 del desarrollo práctico.



En estos esquemas no se ha representado la página de inicio o acceso a las sentencias del Tribunal Constitucional. Este primer documento XML, denominado *STC.xml* (Sentencias del Tribunal Constitucional), carecería de definición formal de tipo de documento y tan sólo contendría los elementos textuales y gráficos de presentación del modelo práctico realizado, así como los enlaces a cada uno de los índices y el correspondiente enlace al formulario de búsqueda documental.

Como se ha expuesto anteriormente, el sistema desarrollado ha pretendido integrar la mayor parte de las tecnologías XML, tratando de ser, asimismo, lo más normativo posible. De este modo, todos los documentos XML deberán llevar algún tipo de descripción de metadatos a través del modelo RDF. Con ello se está contemplando, el hecho de que cuando se lleguen a implementar motores de búsqueda en la WWW que sean capaces de indexar documentos XML, nuestro conjunto documental ya estará en buenas condiciones de ser tratado y, consiguientemente, recuperado por aquéllos. Para llevar a cabo esta implementación de RDF en documentos XML, tal y como se verá en el apartado correspondiente, existen diversas opiniones entre los propios especialistas en este campo, como se comentó en el apartado correspondiente del Capítulo III.5. En el modelo propuesto se ha optado, como ya se señaló, por la incorporación de esquemas RDF (y referencia a vocabularios específicos) dentro del propio esquema XML, referenciando desde éste al vocabulario *Dublin Core.*

A continuación se describen en detalle cada una de las fases descritas en los esquemas anteriores:

- **FASE 1:**

– Se analizará el contenido textual así como la estructura física y lógica de las Sentencias del Tribunal Constitucional (TC) para, de este modo, poder establecer la definición formal de la estructura lógica genérica.

- La definición formal del tipo documental se realizará, en un principio, por medio del modelo de la DTD de XML. Este modelo es relativamente fácil de elaborar y ofrece una primera aproximación a la estructura y modelos de contenido de cada uno de los elementos puestos en juego. Sin embargo, se pretende transformar esta DTD al modelo definido en el estándar *XML Schema Definition Langage* (XSD) pues éste permite hacer todo lo que posibilita el modelo de DTD pero cuenta con la ventaja añadida de poder establecer potentes estructuras documentales así como tipos de datos más variados y complejos. El fichero con el esquema formal de los documentos XML será uno común para todos ellos, bien en el modelo de DTD o de Esquema XML (Doc.DTD/XSD). Como ya se ha comentado, dentro del mismo se contemplará la posibilidad de combinar este esquema con el esquema RDF y el vocabulario Dublin Core.
- Los ficheros HTML de las Sentencias del TC son descargados desde el sitio web del propio TC. Aunque la transformación del marcado HTML al marcado XML específico establecido con la anterior DTD o Esquema XML se ha tratado de automatizar al máximo en la medida de lo posible haciendo uso, para ello, de herramientas informáticas adecuadas como, por ejemplo, a través del editor profesional HTML *DreamWeaver* o el empleo de macros, las dificultades que se han encontrado en el proceso han sido múltiples haciendo, por tanto, desaconsejable su empleo. De igual modo se hicieron pruebas para la conversión de los ficheros electrónicos originalmente redactados con MS-Word a XML mediante el programa *eXportXML*[7]. Aunque este programa resulta de cierto interés por algunas de las capacidades que incluye (proceso rápido de transformación de formatos basado en el uso de estilos en documentos Word, así como la posibilidad de asignación de metadatos al documento XML generado), fue desestimado al no poder realizar la conversión según el marcado establecido en la DTD o esquema XML. Por todo ello, y teniendo en cuanta las malas experiencias sufridas en relación con la conversión de formatos electrónicos en proyectos anteriores, se consideró que lo más rápido y operativo sería la transformación de cada documento HTML a un formato de texto llano (TXT) e insertar,

[7] http//www.schultz.dk/exportxml/

posteriormente, las marcas XML correspondientes para cada uno de los elementos a través de un editor específico para esta labor.

- Los errores que se puedan producir en el marcado de los documentos XML, detectados en la validación de los mismos por el editor empleado, serán subsanados manualmente. De igual forma, se incluirán dentro de cada documento las instrucciones de procesamiento necesarias para la transformación de dicho documento a HTML a través del *parser* XSLT, así como las instrucciones para el posterior procesamiento por parte del servidor Web Apache/Cocoon.
- Como ya se ha señalado, para poder llevar a cabo los planteamientos definidos en el modelo 1, sería necesario establecer y desarrollar un fichero CSS, único para todos los documentos XML, que contendría las reglas oportunas para la asignación del formato de presentación de cada uno de los elementos de estos documentos. La vinculación de los documentos XML con esta hoja de estilo CSS externa se realizará a través de la correspondiente inclusión en aquéllos de la instrucción de procesamiento específica para llevar a cabo esta función, tal y como se verá en apartados posteriores de este capítulo. Aunque esta forma de proceder se adoptó para establecer las oportunas conclusiones sobre este modelo, no sería, como ya se comentó, el adoptado finalmente.

- **FASE 2:**

- Para desarrollar el modelo 2 era necesario, por tanto, desarrollar una serie de ficheros XSLT que habrían de servir para que con posterioridad, y una vez añadida la instrucción correspondiente para el servidor Web Apache/Cocoon, se pudiera producir la transformación del formato XML de los documentos al formato HTML, visualizado por el navegador del usuario.
- Dado que se deseaba generar diversos tipos de documentos HTML (documento completo, documento abreviado y diversos listados) a partir de los datos contenidos en los documentos XML, sería necesario establecer diferentes ficheros XSLT para cada uno de estos tipos de documentos. Este proceso, aunque se ha simplificado mucho en

la figura IV.3., ha sido uno de los más complejos dado que han tenido que intervenir en algunos casos otros documentos XML con la función de intermediar entre los documentos XML primitivos y los ficheros XSLT que han de generar las diversas transformaciones de formato. En el apartado correspondiente al empleo de XSLT se detallarán todas estas cuestiones.

- Se establecen los procesos oportunos de conversión en los ficheros XSLT, en los cuales se incluye la plantilla de elementos que compondrá el documento HTML y el lugar desde donde se han tomar los datos de los documentos XML primitivos (por ejemplo, el contenido del encabezamiento de nivel 1 –H1– será extraído automáticamente del valor que contiene el atributo *entrada* del elemento *TitDocumento*).
- De igual forma, dentro de cada uno de estos ficheros XSLT creados se establecen las instrucciones necesarias para que dentro de la cabecera del documento HTML *virtual* que se generará automáticamente se incluya el elemento de enlace a la hoja de estilo CSS externa. Esta hoja de estilo será la encargada de dar el formato de presentación adecuado a cada uno de los elementos HTML.

- **FASE 3:**

- Se establecen y desarrollan cada uno de los ficheros CSS que han de asignar el formato de presentación a los distintos tipos de documentos HTML generados, que en este caso será única (Doc.CSS) para todos los documentos XML generados en esta fase (pues todos llevarán prácticamente los mismos elementos HTML producidos por la XSLT).
- Es importante volver a señalar que los documentos HTML que se producen en esta transformación por parte del servidor Apache/Cocoon son documentos HTML *virtuales* dado que en ningún momento quedan almacenados físicamente en el espacio de publicación del servidor (el usuario carga un documento XML pero se le presenta *virtualmente* como un documento HTML, incluyendo la visualización del código fuente del documento en este formato).

- Cada uno de los conjuntos formados por el documento XSD, los documentos XML, los documentos XSLT y los documentos CSS se envían a sus correspondientes directorios de publicación dentro del servidor Apache/Cocoon y se procede a la comprobación del resultado final.

- **FASE 4:**

- La fase cuarta está definida *conceptualmente* pues ha sido la que mayores dificultades ha entrañado en su realización debido a problemas tales como los de acceso al software adecuado (programas comerciales), instalación y configuración del mismo y funcionamiento general de estas aplicaciones. En la figura IV.4., se ha optado por incluir dos posibles alternativas para el procesamiento de la información contenida en los documentos XML con vistas a su posterior recuperación por parte del motor de búsqueda.
- En el primer caso, y desde un prisma más cercano a los documentalistas, se ha hecho uso de una herramienta de indexación de los documentos XML que produce el correspondiente fichero inverso de términos contenidos en los documentos (de forma similar a lo que sobre documentos HTML realizan motores de búsqueda en Internet, como AltaVista, Google y tantos otros), donde cada uno de estos términos contiene la referencia al campo (elemento) al cual pertenece, y a los que jerárquicamente se sitúan por encima del mismo. Esta forma de proceder sencilla y ciertamente interesante desde nuestra particular visión será analizada con mayor detenimiento en el apartado IV.2.5.2 de este capítulo sobre creación de la base de datos.
- El segundo caso representa el modelo más común hoy en día, esto es, una base de datos de tipo relacional en donde todos los datos, o aquellos que son considerados de mayor interés, se almacenan en los correspondientes campos de las diversas tablas que se integran en dicha base de datos. La estructura o esquema de la base de datos debería reflejar en gran medida la estructura propia de los documentos XML que ha de almacenar. Por tanto, esta estructura debería ser una adecuación de lo establecido en el

esquema XML (como ya se comentó en anteriores los capítulos II.3 y III.4 de esta tesis, existen fuertes lazos y un gran paralelismo entre el esquema de definición de este tipo de bases de datos con los esquemas XML).

- Esta segunda opción está, en estos momentos, fuera de nuestro alcance pues se requiere personal informático especializado en la gestión de bases de datos XML además del software necesario para el almacenamiento y gestión de los datos. Este tipo de software constituye, en la mayoría de los casos, la piedra angular de los actuales sistemas de gestión de documentos XML por lo que el precio con el que se comercializan suele ser bastante elevado.
- En la actualidad, la conexión a este tipo de bases de datos se suele realizar haciendo uso bien de conectores ODBC (*Open DataBase Connectivity*), propio del entorno operativo de Windows, o bien mediante conectadores JDBC (*Java DataBase Connectivity).* Esta labor de conexión a una base de datos relacional, ya se emplee cualquiera de los dos modelos posibles, requiere amplios conocimientos en la materia, propios de los informáticos. En cualquier caso, se ha realizado una serie de pruebas con el sistema de bases de datos Access de la compañía Microsoft aunque, como se verá con posterioridad, los resultados ofrecidos han sido ciertamente pobres.
- En cualquiera de los casos planteados, lo que sí resulta imprescindible es la existencia de un motor de búsqueda potente que sea capaz de ofrecer al usuario distintos mecanismos para la interrogación y recuperación de la información contenida en los documentos XML. Esta aplicación debería, por tanto, ofrecer unas altas prestaciones en la búsqueda y recuperación documental mediante el uso de formularios que de forma directa o a través de una conversión, permitiesen la interrogación a través del lenguaje XML existente para ello, el XQuery.
- De igual forma, y desde un plano meramente especulativo, el sistema debería de ser capaz de producir una respuesta en formato XML, el cual sería transformado posteriormente, como en anteriores ocasiones, al formato HTML. En este caso, se haría nuevamente uso de una transformación XSLT con su correspondiente hoja de estilo CSS asociada. Como en el caso anterior, desde cada ítem recuperado accederíamos a los documentos XML materia de nuestro interés.

- Todo esto, insistimos, será materia de estudio y desarrollo en futuras aplicaciones que se pudieran derivar de proyectos de investigación por parte de nuestro Departamento relacionados con el tratamiento, difusión y recuperación de documentos XML en la WWW. En nuestro caso, y dada la complejidad de estos temas, simplemente nos hemos limitado a analizar y aplicar una serie de herramientas de nuestro interés disponibles en el mercado para poder establecer futuros criterios más precisos de valoración de este tipo de herramientas informáticas.

Algunas de las cuestiones planteadas a lo largo de los puntos que se integran en cada una de las fases del desarrollo del sistema aquí planteado serán expuestas en los siguientes apartados.

IV.1.3. ANÁLISIS Y DESCRIPCIÓN DEL TIPO DOCUMENTAL

IV.1.3.1. MODELIZACIÓN DE LA ESTRUCTURA LÓGICA

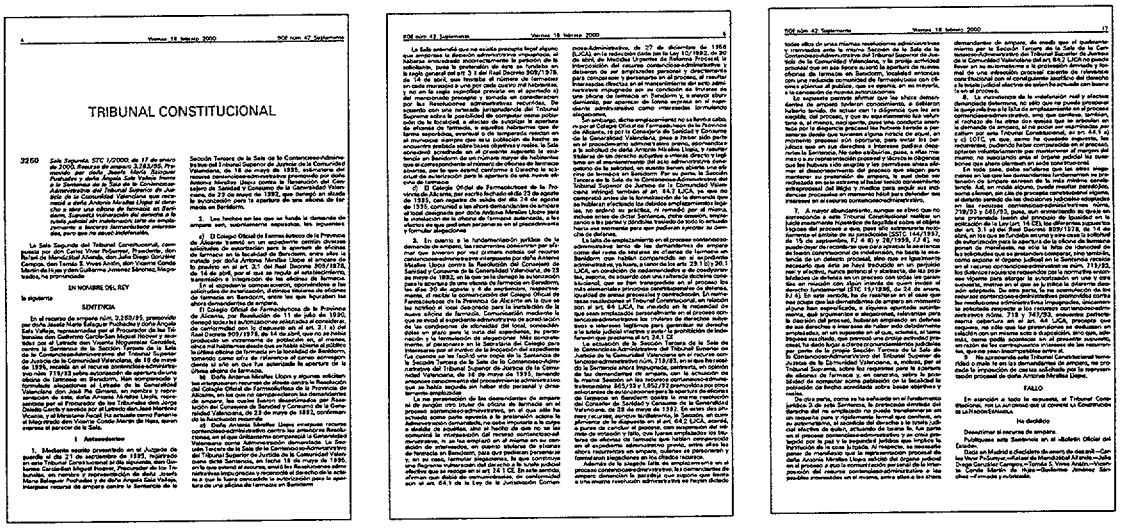
Ya se señaló con anterioridad que una de las primeras tareas realizadas de la primera fase de este desarrollo práctico consistió en el estudio y análisis de la estructura lógica subyacente en las Sentencias del Tribunal Constitucional. Para llevar a cabo esta tarea se consultaron diversas fuentes electrónicas, principalmente bases de datos jurídicas y sitios web, que almacenan y recopilan jurisprudencia de dicho Tribunal. Se analizó de forma exhaustiva un gran número de sentencias, así como su contenido y su estructura en diferentes periodos para poder determinar las posibles variaciones que se hubieran producido a lo largo del tiempo. Debido a que se detectaron algunos cambios en la redacción de las mismas a lo largo de la existencia del TC, decidimos ceñirnos a los últimos años de producción en los que se apreciaba de forma evidente una uniformidad y normalización tanto estructural como de contenido. Por este motivo, el estudio quedó restringido a los documentos electrónicos (en formato HTML) existentes en el sitio web del TC que, como ya se comentó, constituirían la base textual con la que se trabajaría en el posterior marcado XML.

Asimismo, fueron consultados diversos documentos impresos en los que de forma directa se hace mención a las cuestiones estructurales, de forma y contenido, de estas Sentencias, tales como el documento de trabajo interno del TC para la elaboración estos documentos jurídicos (Apéndice 1), así como la información contenida al respecto en la monografía de J. R. Mateo Maciá y en los Apéndices de la obra de F. Caamaño Domínguez y otros, ambas ya citadas en la Parte primera de esta tesis doctoral.

Finalmente, es necesario destacar, debido a la importancia que ello tuvo para el desarrollo del trabajo científico aquí expuesto, los comentarios y las sugerencias recibidas

en las diversas entrevistas personales y conversaciones mantenidas a través del correo electrónico con profesionales que realizan su actividad en el Tribunal Constitucional, como Dª Mª Jesús Cuesta, Jefa de Documentación, Dª Pilar del Pozo, Directora de la Biblioteca, D. Juan Luis Requejo, Letrado de Biblioteca y Documentación, y, especialmente, D. Ignacio Borrajo, Letrado de los Servicios Informáticos.

Con toda esta información, más los análisis previos que se fueron estableciendo a lo largo de las etapas iniciales de este desarrollo, se realizó una primera aproximación a los grandes bloques que componen la *estructura lógica primaria* de toda Sentencia del TC, algunos de ellos obligatorios y por tanto siempre presentes, y otros opcionales, dependiendo de las circunstancias en las que se hubiera producido una determinada Sentencia. La siguiente figura muestra algunas de las partes de una Sentencia del TC (la que posteriormente será tomada como base del marcado inicial XML), tal y como aparece publicada en el Boletín Oficial del Estado.



TRIBUNAL CONSTITUCIONAL

Figura IV.5: Sentencia del Tribunal Constitucional publicada en el BOE.
Fuente: *BOE*. http://www.boe.es/boe/dias/2000-02-18/seccionT.html#00000

El resultado obtenido en este primer nivel general de estructuración fue el siguiente:

- Dentro de toda Sentencia del Tribunal Constitucional se encuentran de forma obligatoria y por este orden de exposición los siguientes grandes bloques que constituyen la estructura de la misma: **Preámbulo**, **Antecedentes**, **Fundamentos Jurídicos** y **Fallo**. En algunas Sentencias es posible encontrar un bloque destinado a los **Votos**, en donde quedan reflejadas las discrepancias y puntualizaciones de los votos particulares que uno o varios Magistrados hayan podido formular en relación con la resolución adoptada por la Sala o el Pleno del TC. La representación gráfica de este primer nivel de división estructural de la Sentencia quedaría de la siguiente forma:

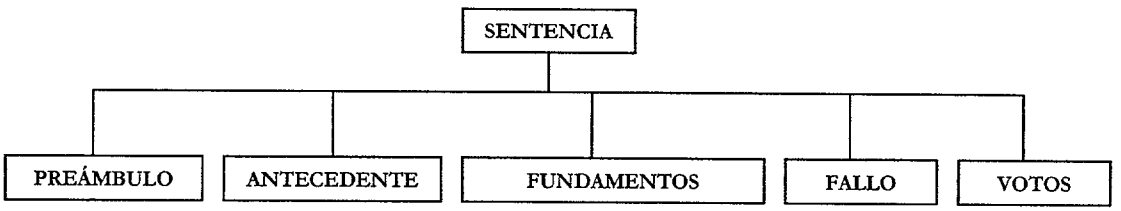


Figura IV.6: Árbol lógico de la estructura general de una Sentencia del TC.

A partir de aquí, se fueron extrayendo todos aquellos otros elementos de relevancia contenidos en estos bloques principales, susceptibles de ser destacados a través del marcado XML por su interés tanto para la descripción del contenido (*estructura lógica orientada al contenido*) como para la presentación (*estructura lógica orientada a la presentación*) del mismo. La definición estructural en esta segunda fase de descripción daría lugar al detalle particularizado de cada uno de los grandes bloques anteriormente definidos, que en su conjunto conformarán la **estructura lógica genérica**. Así:

- Dentro del bloque dedicado al *Preámbulo* de la Sentencia se detectaron dos partes principales: la **Composición** de la Sala o el Pleno que había dictado la Sentencia y un bloque de **Introducción** a la misma en donde se detallaba cierta información relativa a

las partes en litigio y la causa tratada, entre otras cuestiones. A su vez, dentro de cada una de estas partes se detectaron una serie de elementos característicos de los mismos. Así, dentro de la *Composición* se detallaba de forma obligatoria la **Sala** (o el Pleno) que ha tratado este asunto y ha dictado la Sentencia, el **Presidente** de la misma y el nombre de cada **Magistrado** que ha participado en este acto. De igual forma, dentro de la *Introducción* es posible encontrar los siguientes elementos (algunos de ellos siempre presentes y otros no, dependiendo de la causa tratada): el tipo de **Recurso** interpuesto y que ha dado origen a la Sentencia, la persona o entidad que ha sido el **Promotor** del Recurso, la persona o entidad que ejerce de **Representante** del Promotor, la persona o entidad que ejerce de **Defensor** del mismo, la **Resolución Afectada** en dicho Recurso, el **Motivo** por el cual se pide amparo al TC, la persona o entidad que **Comparece** a lo largo de este proceso, la persona o entidad que **Interviene** a lo largo del mismo, y, finalmente, el Magistrado que ha ejercido de **Ponente** en esta causa. A su vez, alguno de estos elementos incluye otros subelementos, para determinar la información relativa a la *Persona* o *Institución* que corresponde. La representación gráfica de todo ello quedaría de la siguiente forma:

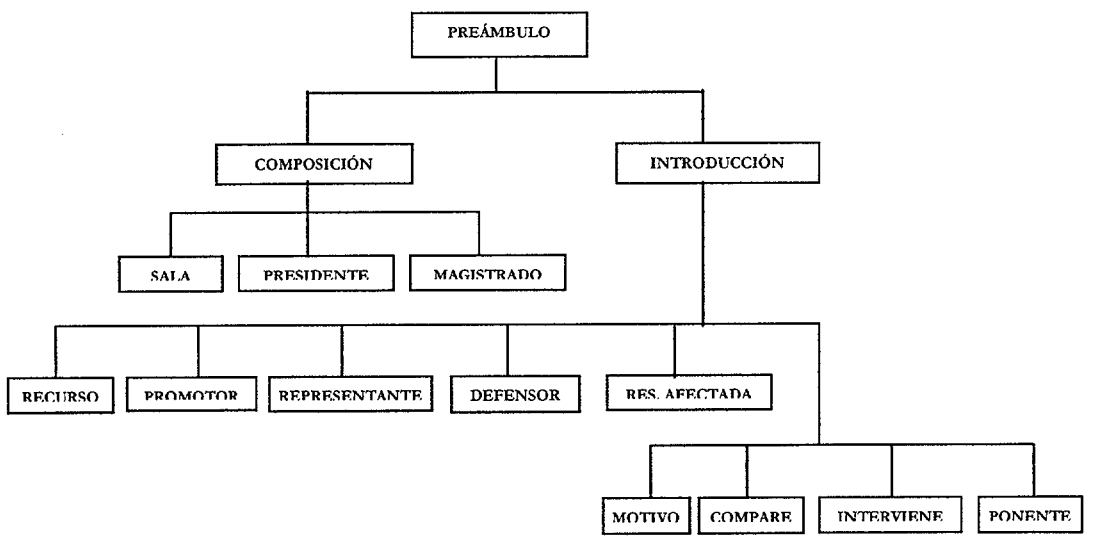


Figura IV.7: Árbol lógico de la estructura particular del Preámbulo de una Sentencia del TC.

- Dentro del bloque principal dedicado a los *Antecedentes* de hecho, lugar en el que se relata la historia del asunto judicial tratado por el TC hasta llegar a su revisión por parte de la Sala o el Pleno, se detectaron un **Encabezamiento** para dar entrada a los mismos y la relación de cada **Antecedente**. A su vez, cada *Antecedente* estaría compuesto de un **Párrafo** o más, siendo esto último lo habitual. La representación gráfica de todo ello quedaría de la siguiente forma:

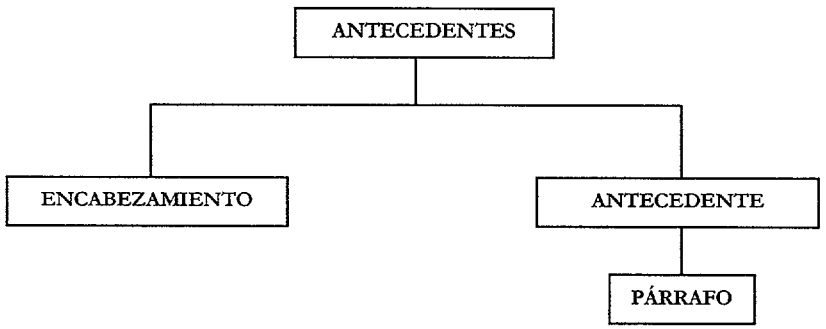


Figura IV.8: Árbol lógico de la estructura particular de los Antecedentes de una Sentencia del TC.

- De forma similar se estructura el bloque principal dedicado a los Fundamentos Jurídicos, apartado éste dedicado a la fundamentación jurídica que se aplica al caso tratado, indicándose para ello los principios doctrinales que se han invocado así como la normativa jurídica aplicada en esta Sentencia. Así, se detectaron un **Encabezamiento** que les da entrada, al que le sigue la relación de cada **Fundamento**. A su vez, cada *Fundamento* estaría compuesto de un **Párrafo**, habitualmente. La representación gráfica de todo lo expuesto quedaría así:

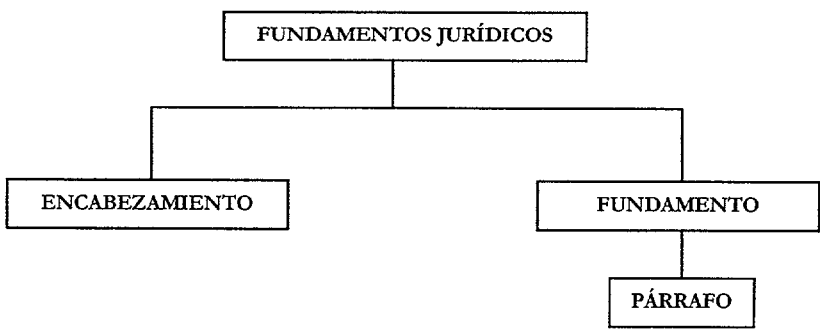


Figura IV.9: Árbol lógico de la estructura particular de los Fundamentos Jurídicos de una Sentencia del TC.

- De igual forma se puede establecer una división dentro del bloque principal correspondiente al *Fallo*, espacio de la Sentencia en donde la Sala o el Pleno del TC dicta el fallo de la misma y hace su correspondiente disposición, distinguiéndose un **Encabezamiento** para la entrada de este bloque, la correspondiente **Decisión**, un apartado para el exhorto a la **Publicación** de la sentencia dictada y la **Fecha de Aprobación** de ésta. Dentro del apartado de *Decisión* es posible encontrar dos elementos: el primero de ellos dedicado a la **Decisión General** adoptada y otro, en algunos casos, al **Desarrollo de la Decisión**. Cada uno de estos dos elementos pueden contener un **Párrafo** o más. La representación gráfica de a estructura particular de este bloque sería la que sigue:

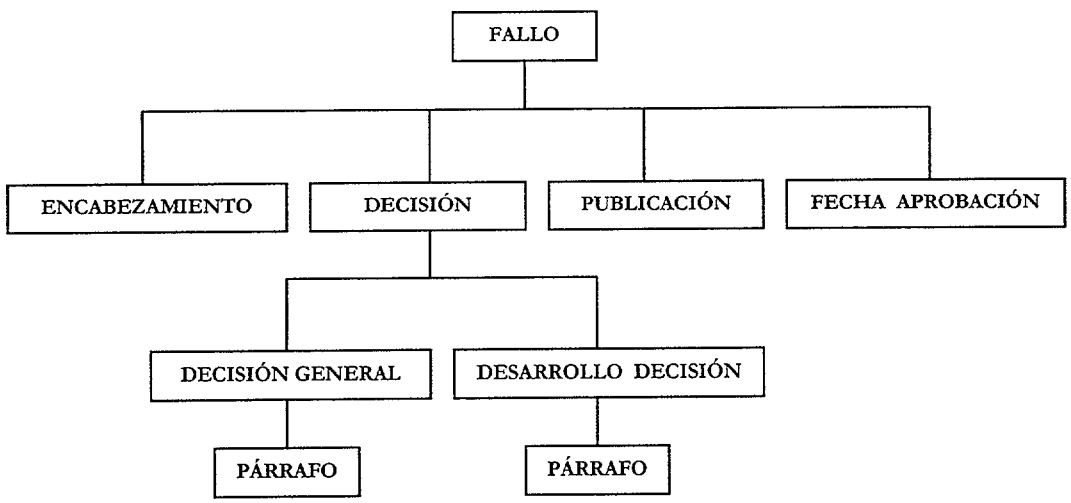


Figura IV.10: Árbol lógico de la estructura particular del Fallo de una Sentencia del TC.

- Finalmente, dentro del bloque principal dedicado a los *Votos*, que sólo existiría en el caso de que alguno de los Magistrados que componen la Sala o el Pleno haya discrepado de la resolución adoptada y desee que quede constancia de ello, es posible, por tanto, encontrar cada **Voto Particular** emitido. La estructura de este apartado está compuesta de dos elementos: una **Cabecera del Voto** y un **Párrafo** o más, en donde se desarrolla la fundamentación jurídica aportada por el Magistrado para establecer su

discrepancia del resto. Dentro de esta *Cabecera del Voto* es posible destacar un elemento que describe al Magistrado que ha **Formulado** el voto particular, así como la **Adhesión** a dicho voto por parte de otros Magistrados. La representación gráfica de todo lo expuesto quedaría de la siguiente forma:

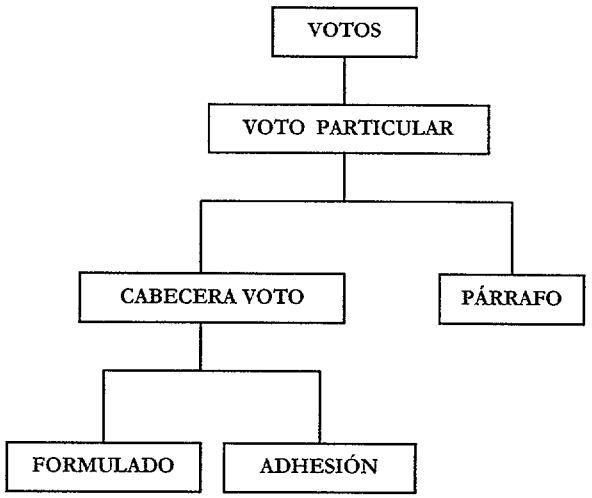


Figura IV.11: Árbol lógico de la estructura particular de los Votos de una Sentencia del TC.

Dentro de algunos de estos bloques y apartados descritos existen igualmente otros elementos que no han sido destacados aquí para no alargar en exceso esta parte dedicada a la descripción de la estructura lógica de toda Sentencia del TC. Estos elementos, que serán expuestos con posterioridad a la hora de definir formalmente el tipo documental a través del modelo de la DTD de XML, hacen referencia a otros aspectos que puede resultar de interés destacar para su posterior descripción formal en aras a facilitar la recuperación de la información contenida en los mismos (Sentencias del TC citadas, otras Sentencias de otros Tribunales, disposiciones legales citadas, nombres y apellidos de Magistrados y personas que han intervenido en el proceso, etc.).

A continuación se detalla la descripción formal o DTD de XML realizada para describir la estructura y el contenido de las Sentencias del TC.

IV.1.3.2. DESCRIPCIÓN FORMAL DEL TIPO DOCUMENTAL

La descripción formal del tipo documental (DTD) propio de las Sentencias del Tribunal Constitucional comprende, como ya se ha venido explicando a lo largo de esta tesis doctoral en sucesivos capítulos, la redacción mediante la gramática y la sintaxis establecidas en la especificación XML necesaria para describir los nombres de los elementos que se van a emplear en el marcado del texto de las Sentencias (las instancias de los documentos XML), así como los modelos posibles de contenido para cada uno de éstos.

Es importante señalar aquí que aunque en un principio esta descripción formal se empezó a formular teniendo en cuenta los criterios propios que habíamos ido estableciendo a lo largo de las anteriores fases, dicha descripción iría sufriendo una serie de transformaciones a lo largo del proceso de elaboración de la tesis doctoral. Estas variaciones fueron motivadas por múltiples factores, como fueron la detección de los primeros errores de descripción, las aportaciones realizadas por diferentes miembros del grupo de Tecnologías de este Departamento, y, de forma especial, las distintas propuestas que desde diversos organismos e instituciones nacionales e internacionales han ido apareciendo en estos años. A dichas propuestas volvemos a remitir para contrastar lo aquí desarrollado.

Dentro de estas variaciones influyó igualmente la evolución que han venido sufriendo los diversos estándares XML acompañantes, en especial aquellos que fijan los mecanismos para hacer la descripción formal de los contenidos de los documentos XML. Así, como ya se detalló, es posible realizar esta descripción formal siguiendo cualquiera de los dos modelos que existen en la actualidad dentro de las tecnologías XML para llevar a cabo esta tarea. En primer lugar, es recomendable hacer una aproximación al modelo de la DTD de XML por ser éste más sencillo de establecer, aunque no exento de cierta complejidad, que el modelo del estándar de Esquema XML.

Para el desarrollo práctico del sistema aquí presentado se empezó, por tanto, elaborando la correspondiente Definición de Tipo Documental o DTD para las Sentencias del TC.

• DEFINICIÓN DEL TIPO DOCUMENTAL

En una primera aproximación a la definición formal de las Sentencias del TC mediante el mecanismo de la DTD de XML se trasladó de una forma más o menos fiel el resultado obtenido en la fase anterior de descripción de la estructura lógica genérica. Así, se tomaron los nombres de cada uno de los bloques y apartados constituidos para asignar el correspondiente nombre (identificador genérico) a cada uno de los elementos y se definió el modelo de contenido de cada uno de ellos, teniendo en cuenta su estructuración jerárquica, así como el orden y frecuencia de aparición de los mismos.

Desde aquí se fueron definiendo una serie de atributos necesarios para delimitar o profundizar en la descripción de algunos aspectos de ciertos elementos. En la mayoría de los casos, estos atributos pretenden la normalización de la información contenida por el elemento o, en otros casos, la codificación de ciertos valores para la mejora en el procesamiento y la recuperación posteriores de esta información.

De igual forma, como ya hemos adelantado, se definieron algunos contenidos nuevos para ciertos elementos, en especial para el elemento Párrafo. Y es lógico que así sea, pues dentro de los párrafos que se encuentran a lo largo del texto de una Sentencia se encuentra información de sumo interés y que es susceptible de ser destacada y, por tanto, tratada como elemento aparte. Así, se estableció que para cada uno de los párrafos existentes sería conveniente ofrecer la posibilidad de marcar las normas legales y las sentencias judiciales citadas, así como otras referencias a personas u entidades que pudieran aparecer. En un principio no se pretendía llegar hasta este nivel de profundidad pero de este modo, al ser elementos opcionales, dejábamos la puerta abierta para su posible aplicación (en el desarrollo final sólo se marcarían, por razones de operatividad, las referencias a otras Sentencias del TC).

Antes de dar paso a los diagramas de contenido de cada uno de los bloques constituyentes de esta DTD, es preciso señalar brevemente que se estableció un bloque inicial nuevo, el denominado **Cabecera**. Dentro de esta cabecera se incluiría toda aquella información que en cierto modo resume los aspectos principales de la Sentencia y, en gran medida, no se encuentra directamente reflejada en el texto de la misma y que resulta de vital importancia para su futuro procesamiento y recuperación de metainformación. Tales son los casos, por ejemplo, de los elementos de *Extracto* y *ResumenFallo* (obtenidos del propio servidor web del TC, donde el listado de cada Sentencia contiene estas informaciones), de la fecha de publicación en el *BOE* (obtenida del mismo sitio, donde además el atributo de *fecha* de este elemento, al igual que en otros elementos que se hace uso de él, se ha normalizado según lo establecido por la *ISO 8601: Representations of dates and times*, esto es, en formato año-mes-día) y de cada uno de los *Descriptores* dentro del elemento *Materia*. En este último caso se emplearon directamente los descriptores contenidos en la base de datos de jurisprudencia constitucional del BOE.

Por otro lado, para este control de descripción de la metainformación no se optó por el uso del modelo RDF, dado que el modelo de la DTD de XML no permite incluir de forma cómoda descripciones de metadatos basadas en este estándar (pues, como se ha explicado, la DTD no está pensada para hacer uso de los *namespaces*), lo cual obligaría a establecer nuevos documentos XML externos y relacionados con los originales (con sus correspondiente DTD) basados en la sintaxis definida por RDF. Este modo de proceder complicaría en exceso el modelo final, existiendo como veremos una solución más interesante y razonable integrando el modelo de esquema de RDF dentro de la definición del esquema de XML de nuestros documentos.

Señalaremos por último, que la labor de redacción y validación de la DTD desarrollada se realizó con el editor de documentos XML profesional XML Spy a través del módulo disponible dentro del mismo, editor del cual se hablará más tarde en este capítulo.

Con todo ello, a la DTD final que se redactó le dimos el nombre de **SentenciaTC.dtd**, disponible para su consulta en el Apéndice 7 de esta tesis doctoral. Su representación gráfica global es la que figura en el siguiente diagrama[8]:

[8] Hemos de advertir que en los diagramas que aparecen a continuación no quedan reflejados los atributos de los elementos que los incorporan, así como sus posibles valores cuando de una lista de ellos se trata, (caso, por ejemplo, del elemento *Recurso* el cual incluye un atributo denominado **codigo** para normalizar el tipo del mismo, el atributo **tipo** del elemento *Parrafo* para establecer si es un párrafo normal, o si se ha de comportar como ítem de listas ordenadas o desordenadas, o los niveles establecidos para el atributo **codigo** que ha de emplearse con el elemento *Enfasis*, y otros tantos), pues su explicación alargaría en exceso esta tesis doctoral. Remitimos, pues, al lector a la consulta del mencionado Apéndice dentro del cual se encuentran todos los atributos empleados en aquellos elementos que lo han necesitado, así como una explicación de su uso.

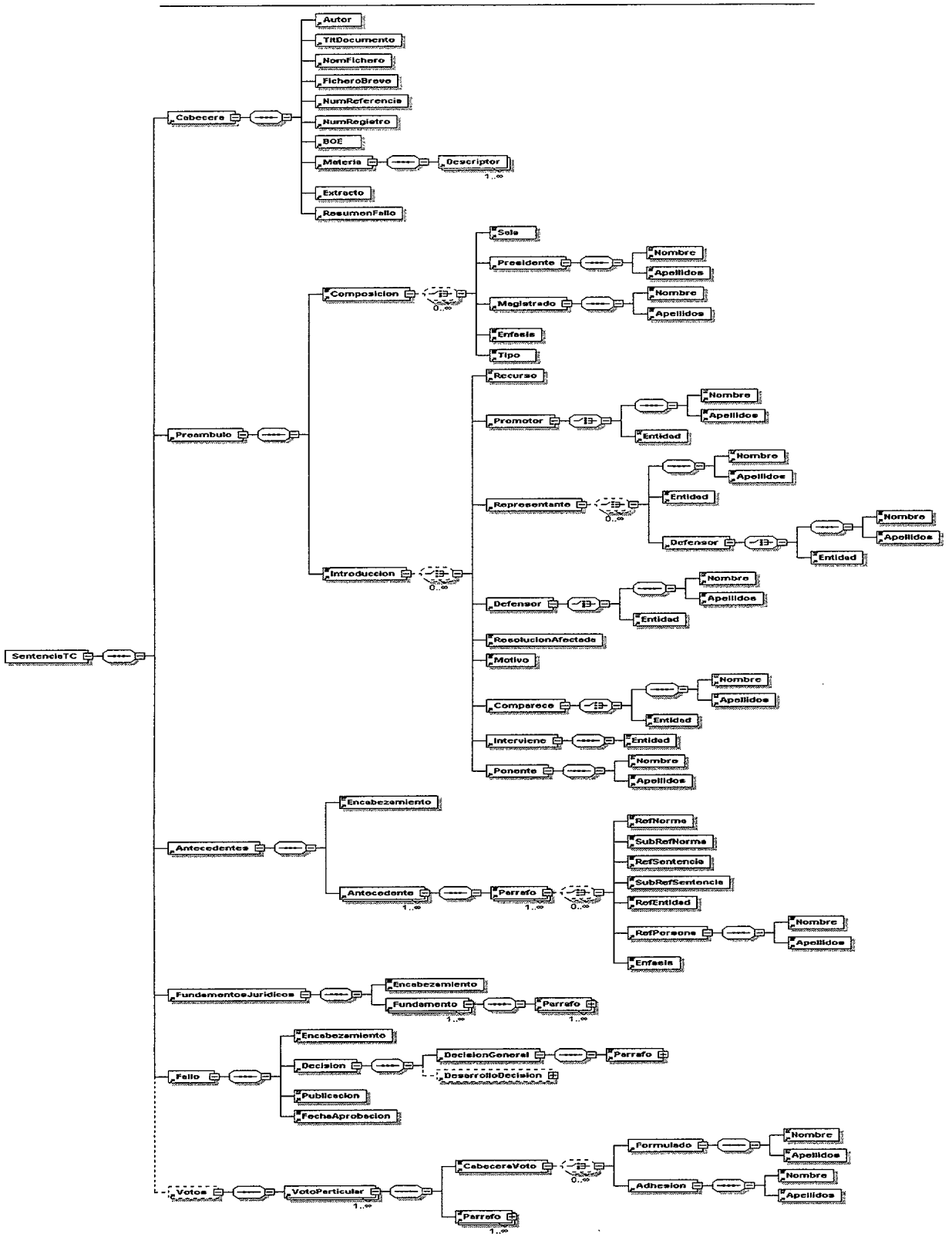


Figura IV.12: Diagrama general de una Sentencia del TC definida en SentenciaTC.dtd.

Analizando cada una de estas partes de forma más detallada, los bloques de esta DTD quedarían de la siguiente forma establecidos:

- El primer estadio de división estructural, como ya se vio con anterioridad quedaría como es representado gráficamente a continuación:

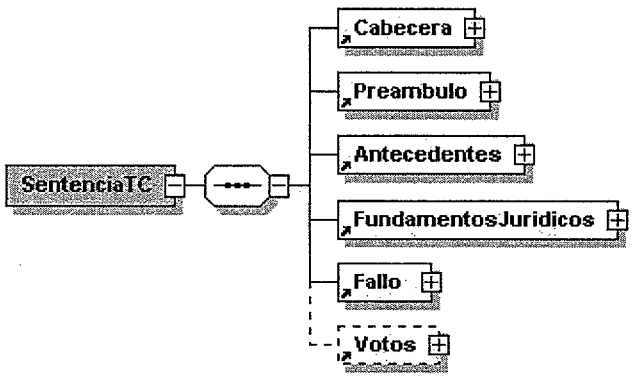


Figura IV.13: Diagrama particular del primer nivel de estructuración definido en SentenciaTC.dtd.

- La **Cabecera** incluye, como decíamos, información que no encontrándose directamente en el texto del documento puede resultar de utilidad para el correcto procesamiento del documento XML y el control de metainformación de interés para la búsqueda y recuperación documental. La cabecera incluye, en esta primera aproximación, los siguientes apartados: autor, título del documento, nombre del fichero, nombre del fichero abreviado (existe una versión reducida de cada Sentencia, como se explicará luego), número de referencia, número de registro, fecha de publicación en el BOE, Materias (con sus descriptores), extracto y resumen del fallo. Gráficamente, todo ello quedaría de la siguiente forma:

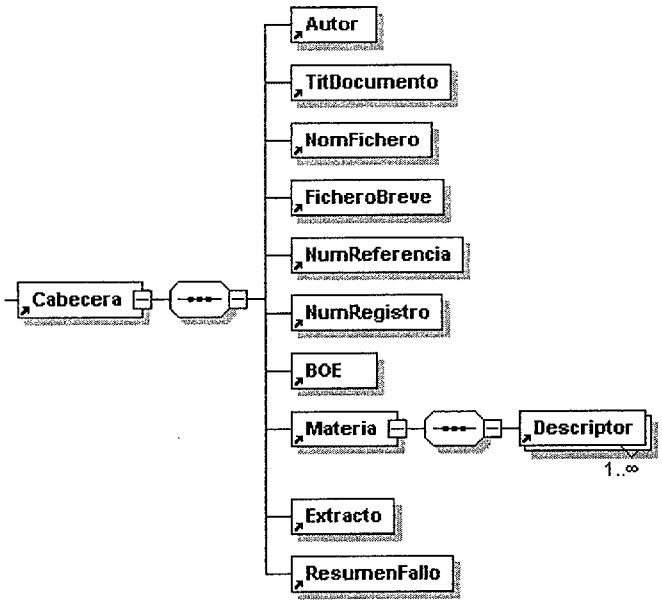


Figura IV.14: Diagrama particular de estructuración del elemento **Cabecera** definido en SentenciaTC.dtd.

- El **Preámbulo** contiene dos bloques estructurales obligatorios: la composición de la sala o el pleno y la introducción a la sentencia. La **Composición** consta de los siguientes elementos: la sala, el presidente, los magistrados, elemento de énfasis (con fines puramente de presentación -negrita, salto de línea, cursiva, etc.-) y el tipo documental tratado (Sentencia, Auto o Providencia, si bien en este desarrollo práctico, como se ha explicado, sólo se contemplarán las primeras). La **Introducción** consta igualmente de un gran número de elementos: tipo de recurso, promotor del mismo, su representante, su defensor, la resolución que se ve afectada, el motivo de la misma, los que comparecen (con sus correspondientes representantes y defensores), los que intervienen y el magistrado que ejerce de ponente de la misma. Todo elemento referido a una persona viene a su vez diferenciado con dos subelementos para la distinción del **nombre** frente a los **apellidos**. Gráficamente quedaría así:

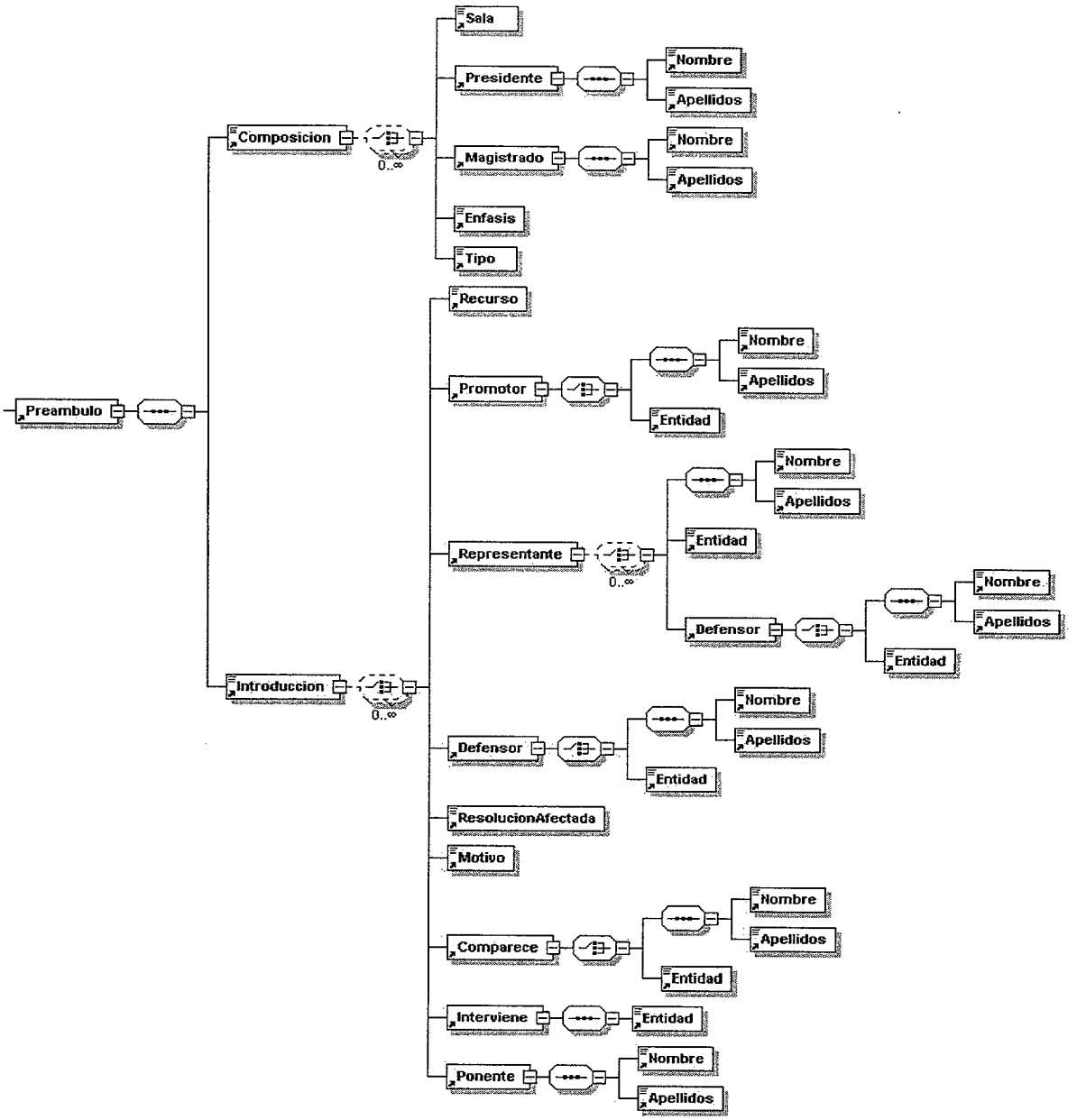


Figura IV.15: Diagrama particular de estructuración del elemento **Preámbulo** definido en SentenciaTC.dtd.

- El bloque principal de **Antecedentes** diferencia entre un encabezamiento de entrada y los diversos antecedentes que se encuentran en el texto. Cada **Antecedente** está

formado por una serie de párrafos diversos (los ordenados numéricamente, los ordenados alfabéticamente y los párrafos sin ordenación). Cada **Párrafo** existente puede contener, además del texto del antecedente, referencias a otras sentencias, o normas, a entidades, a personas (con su nombre y apellidos) y elementos de énfasis; el marcado de las sentencias y las normas referenciadas servirán para establecer los elementos de ancla para los enlaces hipertextuales a dichos documentos. Gráficamente quedaría del siguiente modo:

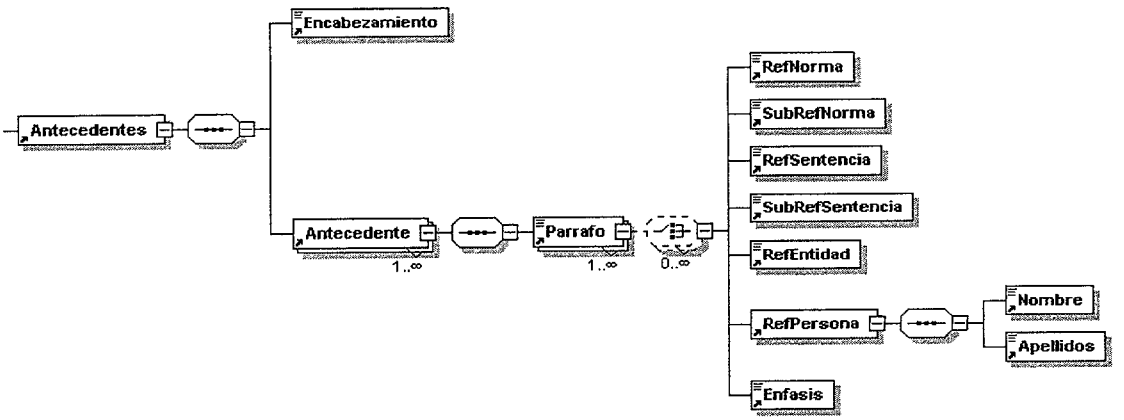


Figura IV.16: Diagrama particular de estructuración del elemento **Antecedentes** definido en SentenciaTC.dtd.

– El caso del bloque de **Fundamentos Jurídicos** es muy similar en cuanto a estructura al anterior; esto es, en un primer nivel se encuentra un encabezamiento del bloque y la relación de fundamentos. Cada **Fundamento** contiene, igualmente, una serie de párrafos de diverso tipo (numéricos, alfabéticos y normales). Cada **Párrafo** puede contener alguno o todos los elementos descritos con anterioridad para los antecedentes. Esta sería su representación gráfica:

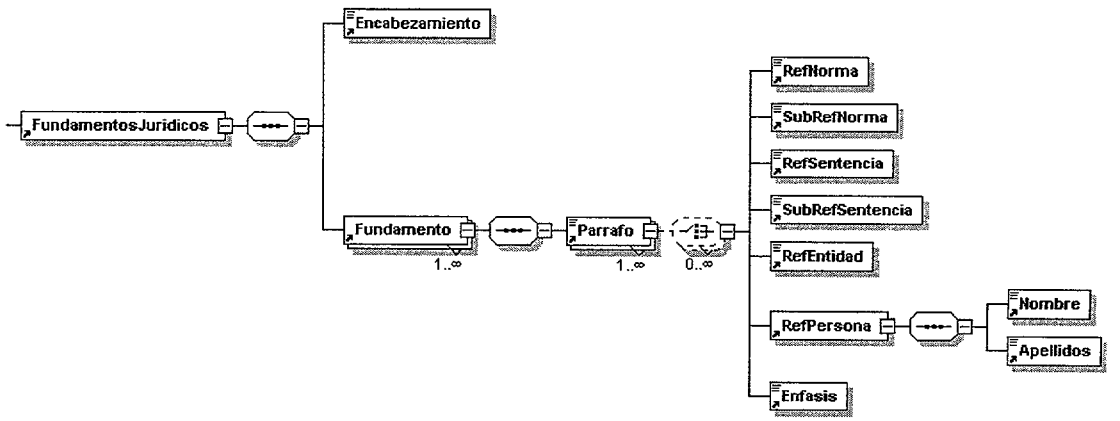


Figura IV.17: Diagrama particular de estructuración del elemento **Fundamentos Jurídicos** definido e SentenciaTC.dtd.

- El **Fallo** de la sentencia está compuesto de los siguientes elementos: un encabezamiento (al igual que en los anteriores casos), la decisión, la mención a la publicación de la sentencia y el lugar y fecha de aprobación de la misma. El elemento **Decisión** se divide a su vez en otros dos subelementos: la decisión general y, si procede, un desarrollo de dicha decisión. La **Decisión general** está compuesta de un párrafo y, si hubiese un **Desarrollo de la decisión**, también estaría compuesto de uno o más párrafos (que podrían contener todos los subelementos descritos con anterioridad). Gráficamente todo ello quedaría del siguiente modo:

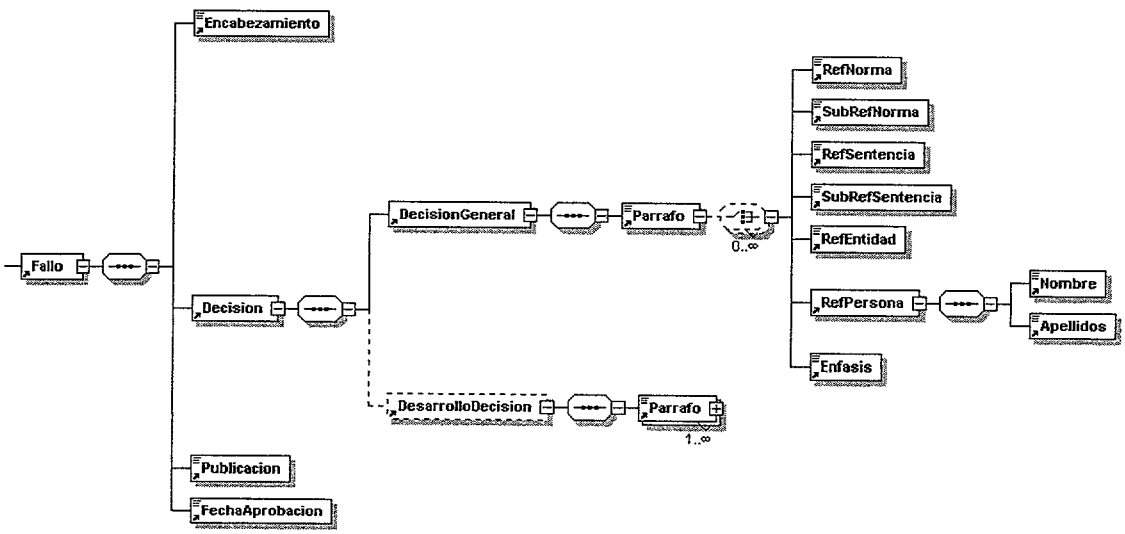


Figura IV.18: Diagrama particular de estructuración del elemento **Fallo** definido en SentenciaTC.dtd.

- Finalmente, el bloque dedicado a los **Votos**, elemento no obligatorio pues muchas sentencias no tienen votos particulares, se estructura, en caso de aparecer, con uno o más votos particulares. El elemento que identifica a cada **Voto particular** viene a su vez dividido en un encabezamiento del voto y uno o más párrafos. El **Encabezamiento del voto** es distinto a los anteriores encabezamientos pues en este caso se trata de un texto en el cual se marcan el magistrado que ha formulado el voto particular y, en caso, los magistrados que se han adherido a dicho voto. Gráficamente quedaría como muestra la figura:

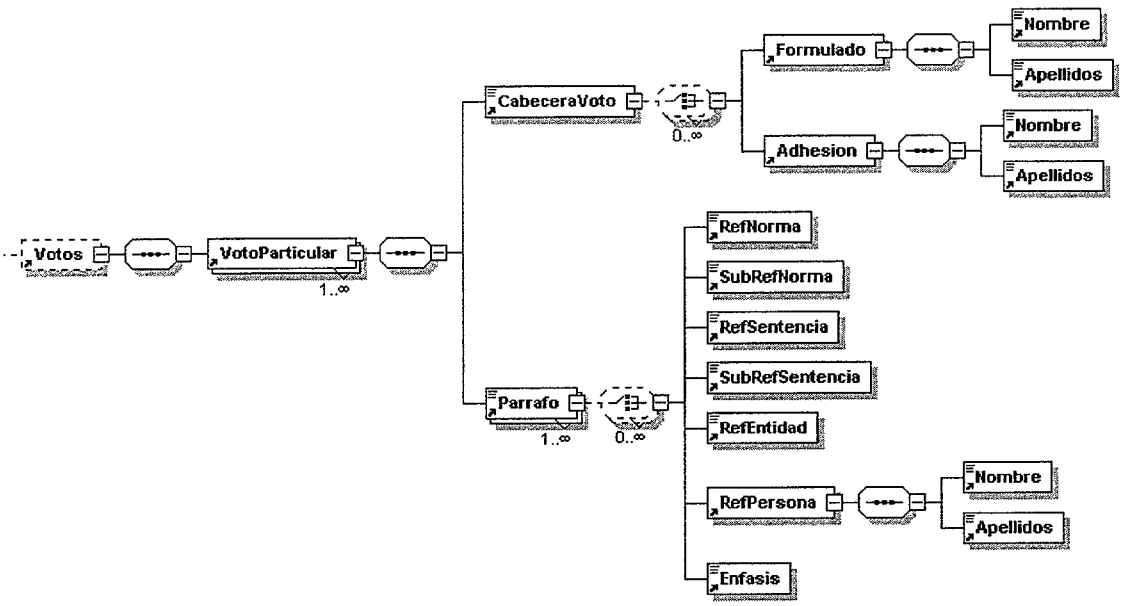


Figura IV.19: Diagrama particular de estructuración del elemento **Votos** definido en SentenciaTC.dtd.

Establecida la DTD se procedió en una siguiente fase al marcado XML del documento electrónico que constituía la base de pruebas del tratamiento de las Sentencias del Tribunal Constitucional, fase ésta que se detalla en el último apartado del presente Capítulo.

IV.1.4. MARCADO INICIAL DE LAS SENTENCIAS DEL TRIBUNAL CONSTITUCIONAL

El marcado de los documentos electrónico constituye aún hoy en día una labor compleja, lenta y, en gran medida, tediosa. El principal motivo de esta afirmación está en la falta de herramientas informáticas adecuadas para llevar a cabo dicha tarea, máxime cuando se trabaja siguiendo una *orientación al documento* y no a los datos.

Como ya se ha comentado, la construcción de documentos XML cuya única función sea la incorporación de datos, con una estructura fija y perfectamente definida, resulta relativamente sencilla con los actuales editores XML existentes en el mercado. En este caso, se trata de herramientas con interfaces gráficas para la creación de aplicaciones XML dinámicas en donde toda su potencialidad reside en la integración de datos y la gestión de contenidos. Muchos de ellos incluyen, una vez cargada la correspondiente DTD o esquema, una interfaz de entrada de datos, normalmente a través de una serie de campos, que facilitan enormemente esta labor (las correspondientes etiquetas son automáticamente incluidas por el programa). Tal es el caso de editores XML tan conocidos y empleados por los que trabajamos en este entorno tecnológico, como *XML Spy*[9] o *Turbo XML*[10] (integrado por tres módulos independientes: *XML Authority*, para la creación y gestión de DTDs y esquemas, *XML Instance*, para el marcado de los documentos, y *XML Console*, para la creación de proyectos y gestión de todo el conjunto), entre otros muchos.

Sin embargo, cuando se está trabajando con modelos de contenido más abiertos, como son los documentales, donde dentro de un texto pueden aparecer destacados algunos elementos (modelos de contenido mixto), la situación se torna más compleja. Ya no se trata de introducir datos dentro de un modelo perfectamente estructurado, similar a los campos de una base de datos, sino, más bien, de ir combinando texto con el marcado de algunos elementos incluidos dentro del mismo de una forma más ambigua. Los editores XML que

[9] http://www.xmlspy.com/

[10] http://www.tibco.com/products/extensibility/solutions/turbo_xml.html

mejor están preparados para enfrentarse a esta labor son todos aquellos que proceden del entorno de producción de documentos SGML donde, como ya se expuso, esta orientación al documento es la que ha venido predominando. Así, dentro de este grupo de editores se encuentran programas tan interesantes como *epcEDIT*[11], *WordPerfect 9*[12] (incluido dentro del paquete *WordPerfect Office 2000*) o *XmetaL*[13], entre otros.

Todos estos editores profesionales (así como otros de interés pero con menores prestaciones, como son los casos *XML Pro*[14] o *XMLwriter*[15], por ejemplo) han sido probados y analizados en las versiones de demostración disponibles, y ninguno de ellos ha satisfecho de un modo íntegro el ideal de editor XML del que hubiéramos deseado disponer.

De forma general, se puede decir en primer lugar que todos ellos tienen un elevado precio para el usuario final. En el caso de los editores XML profesionales orientados a los datos resultan más sencillos de manejar y ofrecen interfaces similares a las que se podrían encontrar en formularios de entrada de datos en los programas gestores de bases de datos. Suelen ser más respetuosos con respecto a los estándares que difunde el W3C, dando soporte en la mayoría de los casos a las últimas versiones en vigor. En estos programas tienen una vital importancia los módulos de definición formal de tipos documentales (tanto la validación de DTDs como de Esquemas XML) y, de forma especial, el dedicado a establecer transformaciones de los documentos XML (de los datos contenidos) a otros formatos de publicación en la Web (HTML, PDF) a través del uso del estándar XSLT. Por el contrario, los editores XML profesionales orientados al texto suelen utilizar mecanismos propios y, por tanto, poco normalizados para el trabajo con estos documentos electrónicos. Suelen dar cobertura únicamente al modelo de DTD de XML y ponen especial énfasis en la

[11] http://www.epcedit.com/

[12] http://www3.corel.com/

[13] http://www.xmetal.com/

[14] http://www.vervet.com/

[15] http://xmlwriter.com/

asociación y definición de estilos de presentación de los documentos XML a través de plantillas de estilo para su transformación a alguna versión de HTML, caso, por ejemplo, de *WordPerfect 9*[16], o directamente en lenguaje CSS, caso, por ejemplo, de *XmetaL*. Sin embargo, estos editores resultan más adecuados y potentes si la institución va a trabajar exclusivamente con un número finito de DTDs pues ofrecen potentes mecanismos de programación para incorporar botones en la barra de herramientas que facilitan el marcado de los textos en un entorno de trabajo WYSIWYG.

En cualquier caso, en futuros desarrollos que pudieran derivarse de proyectos de investigación u otro tipo de actividades científicas, para el tratamiento de documentos XML de carácter textual sería conveniente y recomendable hacer un estudio más profundo de todos estos editores profesionales, y disponer de los más útiles en sus versiones completas.

Teniendo en cuenta todo lo expuesto, se decidió emplear el editor *XML Spy* (en las diferentes versiones que han venido apareciendo hasta la fecha) que, aunque con una clara orientación al tratamiento de datos en documentos XML, estaba disponible en el Departamento y existía una ya larga tradición de uso por parte de los diferentes miembros del grupo de Tecnologías de la Información. No se trata de un editor orientado a la presentación (WYSIWYG), por lo menos no hasta la versión 4.0 aparecida en fechas recientes, por lo que su manejo exige del usuario un perfecto conocimiento de la sintaxis y dinámica de marcado de los documentos electrónicos (elaboración de DTD y/o Esquema XML, marcado del texto, creación de documentos XSLT para la transformación de formatos, etc.). Pero su potencia de trabajo, la perfecta cobertura y actualización que se ofrece para el trabajo global con las diversas tecnologías XML, así como otros factores de interés, le convierten en uno de los programas para la edición profesional de documentos XML más útiles existentes en el mercado.

[16] Véanse al respecto de este tema y de este producto en concreto, las consideraciones que se hacen en Greg Kohn, Michel Rodríguez. *Using WordPerfect 9 to Edit XML* [documento HTML]. XML.com, May 31, 2000. Disponible en http://www.xml.com/print/2000/05/31/wordperfect/index.html (consultado el 15 de junio de 2000).

Establecido todo esto, se procedió a marcar el documento de partida para este desarrollo, la Sentencia 1/2000, de 17 de enero, haciendo uso esta herramienta informática (*XM Spy* en su versión 3.5) y según lo establecido en la DTD anteriormente creada. El documento XML generado tomó el nombre de **stc2000-001.xml**, y su contenido completo se encuentra disponible en el Apéndice 8 de esta tesis doctoral.

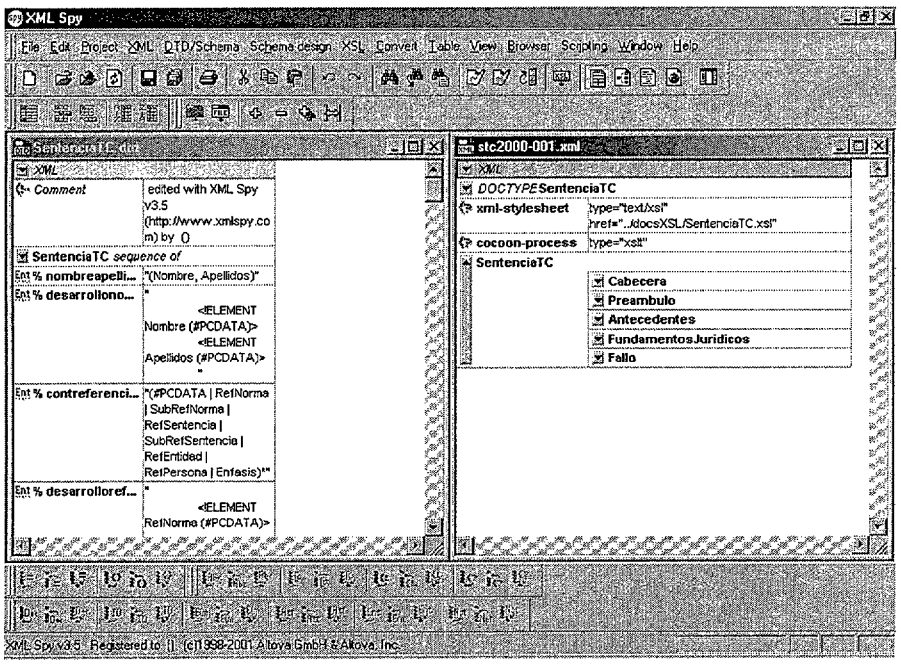


Figura IV.20: Editor XML profesional XML Spy con SentenciaTC.dtd y Sentencia2000-001.xml.

Esta labor de marcado del texto no resultó nada sencilla, como ya se comentó, debido a la gran extensión del texto original (más de 20 páginas) y, en especial, a la minuciosidad con la que se tuvo que hacer dicho marcado para aquellos elementos que requerían un mayor tratamiento. Baste resaltar unos cuantos ejemplos de interés en la asignación de las etiquetas de marcado a algunos de los elementos definidos en la DTD, como la delimitación de relaciones entre los promotores del Recurso y las diferentes partes que intervienen en el mismo, o las relaciones entre las Sentencias del TC, citadas tanto en el apartado de Antecedentes como en el apartado de Fundamentos Jurídicos, y las citas a

partes concretas de las mismas (a un determinado Antecedente o a un Fundamento Jurídico).

Para el primer caso, la DTD establecía la incorporación de diversos atributos al elemento *Recurso*, como son el *codigo* (normalización del tipo de Recurso interpuesto), *numero* (identificador ID para posteriores relaciones con el mismo) y *href* (para establecer una relación de enlace hipertextual al documento que contiene el Recurso, en el caso de que se desease implementar un sistema de acceso global a los documentos electrónicos que se integran dentro del expediente judicial). Cada uno de los promotores, marcados con las etiquetas del elemento *Promotor*, así como el *Representante* y el *Defensor*, contienen el atributo *idref* para establecer una relación directa con el Recurso al cual se adscriben (piénsese que existen Sentencias del TC que pueden agrupar diversos Recursos, cada uno de éstos con sus correspondientes promotores, representantes y defensores). Este modo de proceder nos permitirá con posterioridad realizar y presentar en un documento XML abreviado de la Sentencia (el cual se explicará más adelante) agrupaciones de personas que han intervenido en cada uno de los distintos Recursos relacionados en la Sentencia.

El siguiente código extraído del documento XML que se ha realizado ilustra lo expuesto:

```
[...] En el recurso de amparo núm. <Recurso codigo="RA"
numero="RA1" href="../Recursos/RA/OtrosRecursos.xml">
3263/95</Recurso>, promovido por doña <Promotor idref="RA1">
<Nombre>Josefa María</Nombre><Apellidos>Balaguer Puchades
</Apellidos></Promotor>
[...]
representadas por el Procurador de los Tribunales don
<Representante idref="RA1"><Nombre>Guillermo</Nombre>
<Apellidos>García-San Miguel
Hoover</Apellidos></Representante>y asistidas por el Letrado
don <Defensor idref="RA1"><Nombre>Vicente
</Nombre><Apellidos>Nogueroles
González</Apellidos></Defensor>
[...]
```

Algo muy similar ocurre con las Sentencias del TC citadas en el texto del documento con respecto a las referencias concretas a algunos Antecedentes y/o Fundamentos Jurídicos de dichas Sentencias citadas. Así, una Sentencia del TC citada en el texto es marcada con las

correspondientes etiquetas del elemento *RefSentencia*, incluyendo dentro de la etiqueta inicial los atributos *codigo* (para establecer si se está citando una Sentencia -valor STC- o un Auto -valor AUT-, pues el sistema, si funcionase de modo global contemplaría también este tipo de documentos no jurisprudenciales), *id* (identificador único para cada Sentencia) y *href* (para establecer la relación hipertextual a la Sentencia citada). Cuando se cita de forma particular un Antecedente o un Fundamento Jurídico específico de alguna de estas Sentencias, lo que es bastante habitual, se establecen de igual forma las oportunas relaciones entre estas partes. Así, una cita específica se marca con las etiquetas del elemento *SubRefSentencia*, y en la etiqueta inicial se incluyen los valores que les correspondan a los atributos *codigo* (para establecer si se cita un Antecedente -AN- o un Fundamento Jurídico -FJ-), *idref* (para hacer referencia al identificador asignado con anterioridad a la Sentencia del TC a la que está aludiendo) y *href* (para el posicionamiento hipertextual exacto al Fundamento o Antecedente de la Sentencia referida al navegar por el sistema). Todo ello nos permitirá luego, al igual que en el anterior caso, establecer agrupaciones lógicas de Sentencias del TC citadas, indicando los Antecedentes y/o Fundamentos Jurídicos referidos de cada una, en el correspondiente documento XML abreviado.

El siguiente código extraído del documento XML ilustra lo que se acaba de exponer:

```
<Parrafo tipo="normal"> [...] así como requiere del órgano
judicial el examen de las actuaciones administrativas para
comprobar que se han efectuado los emplazamientos necesarios
y ordenar, en su caso, que se practiquen si se advirtiera
que son incompletos (SSTC <RefSentencia codigo="STC"
id="STC65-94" href="stc1994-065.xml">65/1994</RefSentencia>,
de 28 de febrero, FJ <SubRefSentencia codigo="FJ"
idref="STC65-94" href="stc1994-
065.xml#FJ3">3</SubRefSentencia>;<RefSentencia codigo="STC"
id="STC105-95" href="stc1995-105.xml">105/1995
</RefSentencia>, de 3 de julio, FJ <SubRefSentencia
codigo="FJ" idref="STC105-95" href="stc1995-
105.xml#FJ3">3</SubRefSentencia>).</Parrafo>
[...]
son, entre las más significativas, aunque no exclusivas,
causas o hechos determinantes de la valoración y juicio que
la infracción procesal pueda y deba merecer desde la
perspectiva de aquel derecho fundamental (STC 65/1994, de 28
de febrero, FJ <SubRefSentencia codigo="FJ" idref="STC65-94"
href="stc1994-065.xml#FJ3">3</SubRefSentencia>) [...]
```

Como ya se ha venido señalando, gran parte de este marcado asocaido al documento XML (y diseñado en la DTD) ha intentado combinar de la mejor manera la orientación propia del tipo de documento de que se trata (predominio de contenido textual) con la orientación al señalamiento y marcado de datos concretos. Esta última orientación facilita en gran medida el almacenamiento de datos característicos de estos documentos (fechas, nombres, instituciones, normas jurídicas, etc.) siguiendo los modelos tradicionales de bases de datos, pero, de igual forma, nos permite generar productos secundarios derivados del documento XML de base. Aunque las posibilidades son infinitas desde este último punto de vista (documentos abreviados, listados de todo tipo, etc.), para esta fase de desarrollo del apartado práctico se consideró oportuno disponer de un documento XML abreviado para cada Sentencia tratada, con los datos que caracterizan en mayor medida a las Sentencias del TC.

Este proceso de construcción de documentos XML abreviados se realizaría de forma automática a través de una plantilla única confeccionada a tal fin mediante la tecnología XSLT. Es decir, se elaboraría un documento XSLT único que aplicado a cada documento XML base creado creara de forma automática su correspondiente documento XML abreviado, al cual sólo habría que asignarle un nombre para ser almacenado y en su caso presentado a través del servidor Web XML.

Aunque sobre esta interesante capacidad de la tecnología XSLT se hablará en el correspondiente apartado, comentaremos ahora que para dicho documento XML abreviado de la Sentencia se estableció que contendría los datos más característicos de cada uno de los bloques principales del texto (el resumen de la Sentencia, la composición de la Sala o el Pleno, los datos sobre el recurso interpuesto y partes intervinientes, las Sentencias del TC citadas tanto en los Antecedentes como en los Fundamentos Jurídicos, el Fallo o decisión final y los votos particulares, en el caso de que los hubiera). Asimismo, se establecería en la presentación web de las Sentencias completas el correspondiente enlace a estos resúmenes y desde éstos, igualmente, al documento completo.

Para finalizar este apartado, señalaremos brevemente que el sistema desarrollado exigía una correcta estructuración de directorios dentro del espacio del disco destinado por el

servidor Web XML a la publicación de los documentos XML. Dado que este desarrollo práctico exigiría la elaboración de múltiples documentos electrónicos procedentes de diversas tecnologías Web, habría que diferenciar todo ello perfectamente incluyendo dichos documentos en sus correspondientes carpetas. Los nombres de cada una de las carpetas creadas hace mención al tipo de documento que va a contener, según se detalla en la siguiente relación:

- La carpeta general que contendría a todas las demás se denominaría *TC* (Tribunal Constitucional).
- Los diversos documentos donde se especifican diversas cuestiones asociadas al marcado de los documentos XML que contienen el texto de las Sentencias tendrían su carpeta correspondiente: los ficheros CSS para la asignación de estilo en *docsCSS*, los ficheros con las definiciones formales del tipo de documento en *docsDTD*, los esquemas XML en *docsXSD*, los ficheros para la transformación de formatos en *docsXSL*, con un subdirectorio para otros ficheros relativos a la transformación de un documento XML en otro nuevo en *generarXML*.
- Las imágenes en la carpeta *imagenes*.
- Los ficheros con los distintos índices generados en la carpeta *Indices*.
- Los textos electrónicos de las normas jurídicas citadas en *OtrasN* (no se llegaría a utilizar).
- Los textos de las Sentencias citadas de otros Tribunales de Justicia inferiores en *OtrasS* (no se llegaría a utilizar).
- Los textos electrónicos de cada uno de los Recursos interpuestos ante el TC en *Recursos*, y con subcarpetas propias según el tipo de recurso de que se trate (no se llegaría a utilizar).
- Y los documentos XML con las Sentencias del TC tratadas en la carpeta *SentenciasTC*, tanto a texto completo como los documentos abreviados, en este caso dentro de la carpeta *SentenciasTCbreves*. Otros documentos XML empleados como base para realizar transformaciones posteriores de formato se ubicarían dentro de la carpeta *docsXMLbase*. Los nombres de cada uno de los ficheros XML que contienen las Sentencias tratadas

toma la forma de *Sentencia* seguida del *año* (cuatro cifras) y, separado por un guión, el *número* de la misma (tres dígitos); por ejemplo: *Sentencia2000-001.xml*

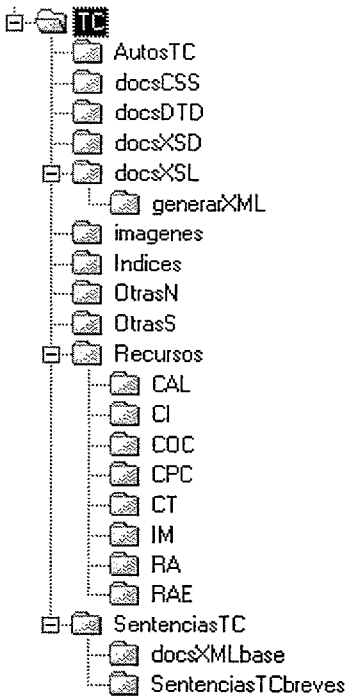


Figura IV.21: Representación gráfica del árbol de carpetas y subcarpetas creado en el desarrollo práctico.

CAPÍTULO IV.2

DESARROLLO FINAL

IV.2.1. EL ESQUEMA XML DE LAS SENTENCIAS DEL TRIBUNAL CONSTITUCIONAL

Si redactar una buena DTD es, como ya hemos venido comentando a lo largo esta tesis doctoral, una labor ciertamente compleja debido a la gran cantidad de factores que hay que tener en cuenta. Redactar la definición formal de un tipo de documento siguiendo el modelo de Esquema XML exige, en un grado muy superior, grandes conocimientos en esta materia.

Deseamos volver a insistir aquí en la importancia que ha adquirido el uso de esquemas XML en detrimento del modelo de DTD para la definición formal de tipos documentales. Como ya se expuso en el Capítulo III.4, el uso de esquemas XML permite corregir en gran medida las limitaciones y la ambigüedad de algunos aspectos de la DTD, principalmente aquellos que hacen referencia al establecimiento de los modelos de contenido de los elementos y a la diversidad de tipos de datos que pueden ser empleados.

Antes de dar paso a este proceso de construcción del esquema XML resulta necesario hacer unas cuantas observaciones de gran importancia al respecto:

- La construcción de este esquema XML se ha realizado siguiendo la especificación de la Recomendación Candidata del *XML Schema* del W3C, de octubre de 2000 y no con la versión de la Recomendación final de este estándar, de mayo de 2000, por varias razones, en especial la imposibilidad de trabajar con dicha Recomendación final por parte del validador de esquemas XML *Xerces* (véanse los comentarios al respecto en el apartado III.4.4 sobre programas analizadores de esquemas XML), con el cual trabaja el servidor Web XML Apache/Cocoon implantado en nuestro sistema. Por este motivo, y de igual forma, se ha tenido que trabajar con la versión 3.5 del programa editor *XML Spy* aunque, como comentábamos en el mencionado apartado, éste sí es capaz de trabajar ya con la Recomendación final desde su versión 4.0. En cualquier caso, esto

tampoco resulta un factor crítico ni para el sistema desarrollado ni para el entendimiento de lo aquí expuesto pues las variaciones que existen entre dichas versiones del *XML Schema* son mínimas. Además, una vez que aparezcan nuevas versiones del *parser* Xerces y sea, por tanto, actualizado nuestro servidor Web XML, la adaptación de la versión antigua a la nueva es un proceso que la mayoría de los editores profesionales de DTD y esquemas XML hacen de forma automática.

- Conviene señalar, igualmente, que en la actualidad existen diversos programas de edición profesional de DTDs, entre ellos *XML Spy* (empleado en todos los procesos realizados en este desarrollo práctico) que permiten transformar una DTD determinada en un modelo de esquema XML. Si bien esto es cierto, dicha transformación resulta poco adecuada y, por tanto, nada útil pues tan sólo se limitan a traducir la sintaxis de la DTD y lo allí definido a la sintaxis propia de los esquemas XML. Esto supone una traslación directa de las declaraciones de los elementos y las definiciones de los modelos de contenido ya existentes a este nuevo lenguaje. Así, si un modelo de contenido contenía grandes dosis de ambigüedad en la DTD (por ejemplo, modelos de contenido mixto donde se entremezcla el texto con otros subelementos), esto mismo se traslada al nuevo modelo. Otro tanto hay que decir del tipo de datos que pueden contener los elementos y/o los atributos: se trasladan de igual forma los tipos de datos establecidos en la DTD al esquema. De este modo, si un elemento tenía definido en su modelo *#PCDATA* (el único posible) la traslación al esquema se hace, sencillamente, asignando el valor *string* (cadena de caracteres) al tipo de dato, cuando sabemos de la riqueza que en este aspecto comporta el uso de esquemas. De igual forma sucede con la definición de los datos contenidos en los atributos de los elementos; se trasladan tal cual (*CDATA* por *string*, *NMTOKEN* por *NMTOKEN*, *ID* por *ID*, etc.).

Todo esto supone, como es fácil de entender, que se desaprovecha la potencialidad que los esquemas XML aportan a las definiciones de tipos documentales.

No deseamos ahondar nuevamente en las diferencias que se establecen entre estos dos modelos, pues ya se expusieron en el Capítulo III.4., al cual remitimos. En este caso,

trataremos de demostrar todas esas ventajas aportadas por el modelo de esquema XML del W3C frente al modelo de la DTD haciendo uso de varios ejemplos extraídos del esquema aquí desarrollado para las Sentencias del TC.

Dado que explicar todo el esquema XML elaborado, cuyo nombre es **SentenciaTC.xsd**, sería una labor compleja y alargaría en exceso esta tesis doctoral, remitimos a la consulta del Apéndice 9, en el cual se encuentra el texto completo del mismo.

La posibilidad de definir con precisión el tipo de datos que han de contener los elementos y los atributos de los documentos XML es, como hemos venido señalando de forma reiterada, de extrema importancia para cualquier sistema de tratamiento de información electrónica basado en estas tecnologías. Y no sólo para la traslación de estos esquemas al esquema propio de la base de datos que ha de extraer y almacenar los datos de los documentos XML sino también para el control de los errores que se pudieran producir a la hora de redactar nuevos documentos XML. Este último hecho puede ser analizado en el siguiente ejemplo de nuestros modelos de DTD y de esquema XML:

- En la DTD se definieron un par de elementos que contenían entre sus atributos uno destinado a señalar una determinada fecha (en el caso del elemento *BOE*, para señalar la fecha de publicación de la Sentencia, y en el caso del elemento *FechaAprobacion*, para señalar la fecha en la que ésta fue aprobada por la Sala o el Pleno del TC). El modelo de la DTD no permite indicar este tipo especial de dato (sólo como *CDATA*, datos de carácter) pero sí en el modelo de esquema XML, identificado como tipo de dato *xsd:date*. Por tanto, con el uso de la DTD sería fácil poner cualquier valor, numérico y/o alfabético, no siendo por ello incorrecto para el *parser*. Sin embargo, al hacer uso de ese tipo de dato para el atributo fecha en el esquema, ahora los posibles errores se minimizan. Éstos, es cierto que no son evitables al cien por cien pues aunque ahora nos debamos ceñir al esquema de la ISO 8601 para la expresión de fechas (año-mes-día), siempre es posible cometer algún error sintácticamente válido (por ejemplo, en vez de señalar el año 2000 indicar que es el año 3000). De igual forma, se puede ver en este mismo ejemplo el caso de la identificación y acotación que se ha realizado al atributo

numero del elemento *BOE*, en donde se ha especificado que es un número entero y que, además, sólo puede estar comprendido entre los valores 1 y 320 (no se publican más de dicho número de Boletines al año).

En DTD (SentenciaTC.dtd)	En esquema XML (SentenciaTC.xsd)
[...] <!ELEMENT BOE EMPTY> <!ATTLIST BOE numero CDATA #REQUIRED **fecha CDATA #REQUIRED** > *[...]*	*[...]* <xsd:element name="BOE"> <xsd:complexType> <xsd:attribute name="numero" use="required"> <xsd:simpleType> **<xsd:restriction base="xsd:integer">** **<xsd:minInclusive value="1"/>** **<xsd:maxInclusive value="320"/>** </xsd:restriction> </xsd:simpleType> </xsd:attribute> <xsd:attribute name="fecha" type="xsd:date" use="required"/> </xsd:complexType> </xsd:element> *[...]*
[...] <!ELEMENT FechaAprobacion (#PCDATA)> <!ATTLIST FechaAprobacion **fecha CDATA #REQUIRED** >	*[...]* <xsd:element name="FechaAprobacion"> <xsd:complexType> <xsd:simpleContent> <xsd:restriction base="xsd:string"> **<xsd:attribute name="fecha" type="xsd:date" use="required"/>** </xsd:restriction> </xsd:simpleContent> </xsd:complexType> </xsd:element> *[...]*

- Existen otros cuantos ejemplos de similares características al anterior, donde se hizo uso directo de los tipos de datos definidos por el esquema del W3C, otorgando de este modo una mayor precisión a la descripción de los mismos. Tal es el caso, por ejemplo, de aquellos elementos definidos en nuestro esquema que debían contener dentro de uno de sus atributos un valor relativo a la ubicación de un determinado fichero referenciado (tanto a los integrados dentro de nuestro sistema como a los ficheros externos al mismo). Este es el caso del atributo *href* contenido en los elementos *Recurso*, *RefNorma*, *SubRefNorma*, *RefSentencia*, *SubRefSentencia* y *ResolucionAfectada*. Con la DTD tan sólo podíamos precisar, al igual que en el ejemplo anterior, que los datos contenidos en dicho atributo eran del tipo *CDATA*. Por el contrario, haciendo uso de los tipos de datos definidos en el esquema XML es posible precisar que se trata exactamente de

referencias a direcciones de ficheros (tipo de dato *uriReference*), tal y como queda patente en el siguiente cuadro para el caso de la Referencia a una Sentencia del TC (en este caso, se puede observar además la traslación de la lista de valores que puede tomar el atributo *codigo* de este elemento al modelo de esquema XML):

En DTD (SentenciaTC.dtd)	En esquema XML (SentenciaTC.xsd)
[...] <!ELEMENT RefSentencia (#PCDATA)> <!ATTLIST RefSentencia codigo (S | A | STC | ATC) #REQUIRED id ID #REQUIRED **href CDATA #REQUIRED** > *[...]*	*[...]* <xsd:element name="RefSentencia"> <xsd:complexType> <xsd:simpleContent> <xsd:restriction base="xsd:string"> <xsd:attribute name="codigo" use="required"> <xsd:simpleType> <xsd:restriction base="xsd:NMTOKEN"> <xsd:enumeration value="S"/> <xsd:enumeration value="A"/> <xsd:enumeration value="STC"/> <xsd:enumeration value="ATC"/> </xsd:restriction> </xsd:simpleType> </xsd:attribute> <xsd:attribute name="id" type="codResolucionTC" use="required"/> **<xsd:attribute name="href" type="xsd:uriReference" use="required"/>** </xsd:restriction> </xsd:simpleContent> </xsd:complexType> </xsd:element> *[...]*

- De igual forma, si los tipos de datos que han de contener ciertos elementos y/o atributos son particulares o característicos de nuestro modelo (variaciones o adaptaciones de los definidos en la Recomendación del *XML Schema*), es posible definirlos, restringiendo o adaptando los existentes en la Recomendación para que el valor que vayan a tomar dichos elementos y/o atributos se ajuste a las reglas ahí establecidas. Una vez definido y asignado un nombre a dicho tipo de dato, éste es referenciado a la hora de establecer el tipo de dato que pueden contener los elementos o atributos. En nuestro esquema XML se han definido gran número de tipos de datos propios (especialmente, para la asignación de códigos), por lo que sería poco práctico explicarlos todos aquí. Por ejemplo, toda Sentencia del TC que aparezca citada en el texto que se está marcando debe llevar invariablemente un identificador único (un ID) como valor del atributo *id*. En el caso de la DTD no es posible especificar nada más allá

sobre el contenido de ese dato (sólo las restricciones que el modelo de DTD impone a los valores de este tipo de dato). Sin embargo, en el modelo de esquema XML se puede ser mucho más precisos en esta descripción: debe empezar invariablemente bien por STC (para las citas a Sentencias) o por ATC (para las citas a Autos), continuar con una serie de dígitos para el número de la resolución, un guión y terminar con dos dígitos para señalar el año. El resto de Sentencias y Autos de otros Tribunales inferiores debería ir sólo con el código S y el código A, respectivamente. Esto mismo se aplica, como decíamos, a otra serie de códigos empleados en el marcado de los documentos XML, como es el caso de los votos particulares donde cada uno de ellos es identificado unívocamente por medio de un dato que ha de incluir las letras VP seguidas del número de voto que corresponda.

En DTD (SentenciaTC.dtd)	En esquema XML (SentenciaTC.xsd)
[…] <!ELEMENT RefSentencia (#PCDATA)> <!ATTLIST RefSentencia codigo (S | A | STC | ATC) #REQUIRED **id ID #REQUIRED** href CDATA #REQUIRED > *[…]*	*[…]* <xsd:simpleType name="codResolucionTC"> <xsd:restriction base="xsd:ID"> **<xsd:pattern value="\[S] d{} - d{2}"/>** **<xsd:pattern value="\[A] d{} - d{2}"/>** **<xsd:pattern value="\[STC] d{} - d{2}"/>** **<xsd:pattern value="\[ATC] d{} - d{2}"/>** </xsd:restriction> </xsd:simpleType> *[…]*
[…] <!ELEMENT VotoParticular (CabeceraVoto, Parrafo+)> <!ATTLIST VotoParticular **numero ID #REQUIRED** > *[…]*	*[…]* <xsd:simpleType name="codvoto"> <xsd:restriction base="xsd:ID"> **<xsd:pattern value="\[VP] d{}"/>** </xsd:restriction> </xsd:simpleType> *[…]*

- Otro aspecto fundamental en los esquemas XML es la definición de los modelos de contenido de los elementos. Ya se comentó en el apartado III.4.1 que los modelos de contenido mixto (texto entremezclado con otros subelementos), resultan muy ambiguos y poco prácticos en la realidad: al quedar tan abiertos (los subelementos pueden aparecer o no, en cualquier orden y un número de veces indeterminado) los errores que se pueden producir durante el marcado del texto serán, a buen seguro, numerosos. Con el esquema XML se puede definir con mayor precisión este tipo de modelo de contenido mixto, reduciendo notablemente esta ambigüedad. En algunos casos esto no es posible debido a las características ya señaladas de los documentos textuales

(frecuencia irregular de aparición de los subelementos, orden aleatorio en la aparición de los mismos, etc.). En nuestro caso, aunque existen situaciones en que no se puede evitar dicha ambigüedad (por ejemplo, en los párrafos es imposible establecer el orden y la frecuencia de aparición de las citas a otras Sentencias del TC o de otro Tribunal, a disposiciones legales, etc.), en otras sí es posible y conveniente evitar este problema. Por ejemplo, el elemento *Composicion*, a pesar de contener texto entremezclado con otros subelementos, éstos siempre deberán aparecer en una determinada secuencia y con una frecuencia concreta. Véase, por ejemplo, lo que sucede con el subelemento *Magistrado*, que puede aparecer bien cinco veces, si se trata de una Sentencia de alguna de las Salas (5 magistrados más un presidente) o bien once, si se trata de una Sentencia del Pleno (11 magistrados más un presidente).

En DTD (SentenciaTC.dtd)	En esquema XML (SentenciaTC.xsd)
[…] <!ELEMENT Composicion (#PCDATA | Sala | Presidente | Magistrado | Enfasis | Tipo)*> *[…]*	*[…]* <xsd:element name="Composicion"> <xsd:complexType mixed="true"> <xsd:sequence> <xsd:element ref="Sala"/> <xsd:element ref="Presidente"/> **<xsd:element ref="Magistrado" minOccurs="5" maxOccurs="11"/>** <xsd:element ref="Enfasis"/> <xsd:element ref="Tipo"/> </xsd:sequence> </xsd:complexType> </xsd:element> *[…]*

De similar importancia es lo sucedido con el elemento *CabeceraVoto*, donde su modelo de contenido definido en la DTD quedaba muy abierto y ambiguo, y, por tanto, susceptible de ser interpretado de forma errónea: si existen votos particulares en una Sentencia, el elemento *Formulado* debe aparecer siempre, sin embargo el elemento *Adhesion* puede no aparecer y si lo hace, no más de dos veces (no puede haber más de dos magistrados que se adhieran a un voto particular). La siguiente tabla muestra estas diferencias entre la DTD y el esquema XML elaborado:

En DTD (SentenciaTC.dtd)	En Esquema XML (SentenciaTC.xsd)
[...] <!ELEMENT CabeceraVoto (#PCDATA | Formulado | Adhesion)*> *[...]*	*[...]* <xsd:element name="CabeceraVoto"> <xsd:complexType mixed="true"> <xsd:sequence> **<xsd:element ref="Formulado"/>** **<xsd:element ref="Adhesion" minOccurs="0" maxOccurs="2"/>** </xsd:sequence> </xsd:complexType> </xsd:element> *[...]*

Todos estos ejemplos son lo suficientemente representativos para entender el proceso de construcción del Esquema XML que define a la Sentencia del TC que, como ya se ha señalado, resulta bastante más complejo, técnico y extenso que el empleado para la elaboración de la DTD. El empleo de un modelo u otro irá en función de las propias características del texto a tratar pero, en líneas generales, ha quedado demostrada la utilidad de esta tecnología XML cuando de lo que se trata es de garantizar el máximo control posible de los datos contenidos en la información tratada.

Finalmente, comentaremos que algunas otras cuestiones que aparecen tanto en la DTD como en el esquema XML, en especial ciertas instrucciones de procesamiento de documentos XSL y elementos referentes al control de metadatos RDF, serán tratadas posteriormente en sus respectivos apartados.

IV.2.2. HOJAS DE ESTILO CSS PARA LA PRESENTACIÓN DE LOS DOCUMENTOS

En el apartado IV.1.2 del anterior capítulo se expusieron de forma general los razonamientos que nos habían hecho plantearnos, no sin cierto pesar, la necesidad de transformar los documentos XML tratados al formato HTML.

Desde la óptica de los documentalistas especializados en este tipo de tecnologías informáticas para el tratamiento de documentos electrónicos, nos parecía erróneo tener que transformar el documento a otro formato distinto cuando lo que se desea mostrar en el navegador web está todo prácticamente definido en el documento XML correspondiente a la Sentencia del TC tratada; tan sólo habría que asignarle el formato o estilo de presentación de forma directa. Tal vez, la concepción más tradicional en la edición de textos electrónicos, donde a las diversas partes que componen el texto se les asignan una serie de propiedades presentación (piénsese en los procesadores de texto como MS Word o en la propia edición de documentos HTML), nos ha venido marcando a lo largo de todos estos años a los miembros del grupo de Tecnologías de la Información de este Departamento. Si las marcas establecidas al texto electrónico estaban ya perfectamente definidas a lo largo de este tiempo, sólo restaba, pues, crear una hoja de estilo externa definida mediante el lenguaje CSS (*Cascading Style Sheets*), crear la vinculación correspondiente en el documento XML y, con todo ello, el documento estaría listo para ser presentado a cualquier usuario a través de la WWW.

En el apartado III.3.1 comentamos los problemas existentes con la tecnología de las hojas de estilo en cascada, tanto para documentos HTML como para documentos XML. En principio, se trata de un lenguaje potente para la asignación de propiedades de estilo a documentos marcados con estos lenguajes que ha venido cubriendo las expectativas y necesidades de la mayoría de los autores de documentos electrónicos para la Web. Pero con la irrupción de la tecnología XML sus limitaciones se han hecho más evidentes en este

nuevo entorno tan exigente: además de no estar escrito con la misma gramática que los documentos XML, sus carencias técnicas (imposibilidad de asociar un comportamiento hipertextual a un elemento, imposibilidad de generar nuevos elementos, etc.) se hacen más palpables aquí pues a los elementos marcados mediante XML no se les puede asignar un formato de presentación preestablecido. Un navegador que interpreta documentos HTML puede no entender una determinada propiedad CSS pero, en ese caso, mostrará la que ha heredado de un elemento superior o, simplemente, le asignará la representación que tradicionalmente se ha establecido para cada elemento del lenguaje HTML.

A pesar de todos estos planteamientos en contra, se procedió a desarrollar el modelo concebido en un primer momento (modelo 1 reflejado en la figura IV.3.) tratando de reproducir lo más fielmente posible el aspecto general de la presentación impresa de las Sentencias del TC, así como el propio de cada uno de los elementos contenidos (textos destacados en negrita, centrados, en cursiva, etc.). Para ello se procedió a crear un nuevo documento electrónico (una hoja de estilo) que contuviera las propiedades CSS que se deseaba asignar a los elementos marcados en el documento XML de partida. Ya contábamos con el hecho de que las versiones antiguas de algunos navegadores (Netscape Navigator en sus diferentes versiones 4 o Internet Explorer 4, por ejemplo) serían incapaces de interpretar correctamente estas propiedades (amén de su nula capacidad para la interpretación directa de documentos XML) pero no nos resistíamos a intentar, al menos, proponer esta solución tecnológica pues en ella creíamos, y seguimos creyendo. Este nuevo documento electrónico, denominado con posterioridad **SentenciaTC(old).css**, se encuentra incluido dentro de la tesis en su Apéndice 10.

Para que esta hoja de estilo CSS pudiese ser incorporada a los documentos XML se añadió al prólogo de los mismos la necesaria instrucción de procesamiento, tal y como figura a continuación:

```
<?xml-stylesheet type="text/css"
href="../docsCSS/SentenciaTC(old).css"?>
```

Antes de dar paso a la valoración de los resultados obtenidos, mencionaremos brevemente que para llevar a cabo la confección de esta hoja de estilo CSS se evaluaron diversos programas editores específicos para este fin. Ya se tenía experiencia de uso con el editor profesional *TopStyle*[17], pues esta es la herramienta con la que habitualmente trabajamos los componentes del grupo de Tecnologías de la Información del Departamento, y cuyo manejo enseñamos a los alumnos en los diversos cursos de formación que sobre estas materias venimos impartiendo desde hace ya unos cuantos años. A pesar de ello, se analizaron otros editores existentes en el mercado, como fueron los casos de los programas *Style Master*[18] y de *Style Studio*[19]. El primero de ellos, resultaba bastante sencillo y no demasiado potente por lo que fue descartado rápidamente. En el segundo caso, se trata de un magnífico editor con unas prestaciones muy profesionales; pero no aportaba mejoras en relación con *TopStyle*.

En definitiva, para el desarrollo de ésta y otras hojas de estilo CSS (de las que se hablará en apartados posteriores) se empleó el editor *TopStyle* en su versión 2.0, tal y como se muestra en la siguiente figura:

[17] http://www.bradsoft.com/topstyle/

[18] http://www.westciv.com.au/style_master/

[19] http://www.xsstudio.net/

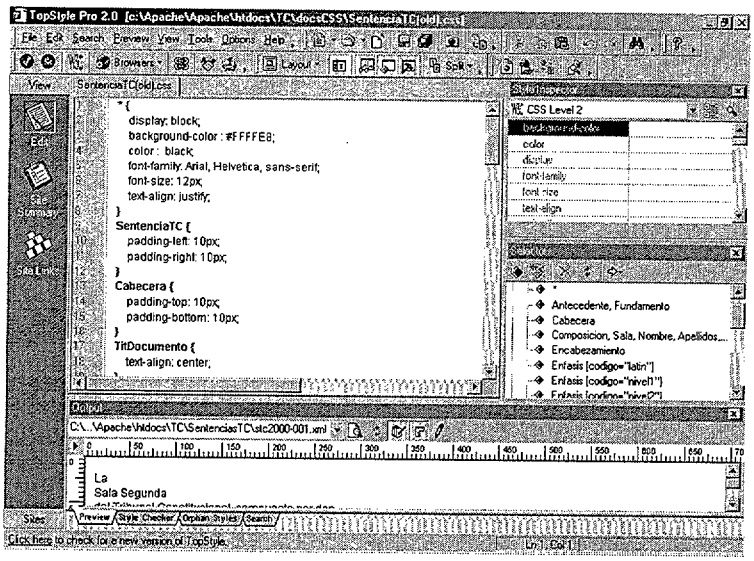


Figura IV.22: Presentación de *SentenciaTC(old).css* en el editor TopStyle 2.0.

Centrándonos en los resultados obtenidos, como era de esperar, éstos, aunque en general aceptables haciendo uso de las últimas versiones de los principales navegadores, no cubrían las expectativas pues incluso las últimas versiones de algunos de los navegadores más capacitados para trabajar con hojas de estilo CSS cometían ciertos errores de interpretación. Los resultados quedan patentes en las siguientes capturas de pantalla que se hicieron de estas pruebas[20]:

[20] Desde el grupo de Tecnologías de la Información de este Departamento se han venido realizando un gran número de pruebas sobre la interpretación que de las propiedades de las hojas de estilo CSS (en su versión 1 y 2) hacen los principales navegadores Web existentes en el mercado. Todo ello ha quedado reflejado y expuesto en los distintos cursos de formación que se han venido impartiendo sobre esta materia, para documentos tanto HTML como XML. En el caso que nos ocupa, se probaron las distintas versiones que han ido apareciendo hasta la fecha de los navegadores Internet Explorer, Netscape Navigator, Opera y Mozilla. Las últimas versiones de Netscape Navigator y Mozilla (ambos son prácticamente el mismo producto) y Opera han sido las más satisfactorias en este campo. En cualquier caso para una más amplia información sobre la cobertura proporcionada por estos navegadores al estándar CSS, recomendamos la consulta de los siguientes documentos:

- Keith Schengili-Roberts. *Core CSS*. Upper Saddle River, NJ: Prentice-Hall, 2000.

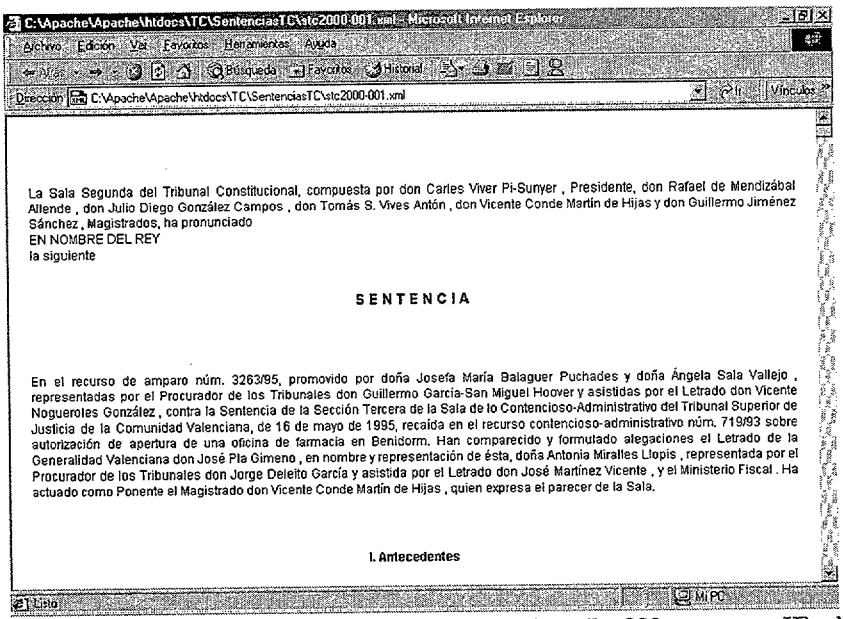


Figura IV.23: Presentación de *stc2000-001.xml* con hoja de estilo CSS externa en IExplorer 6.

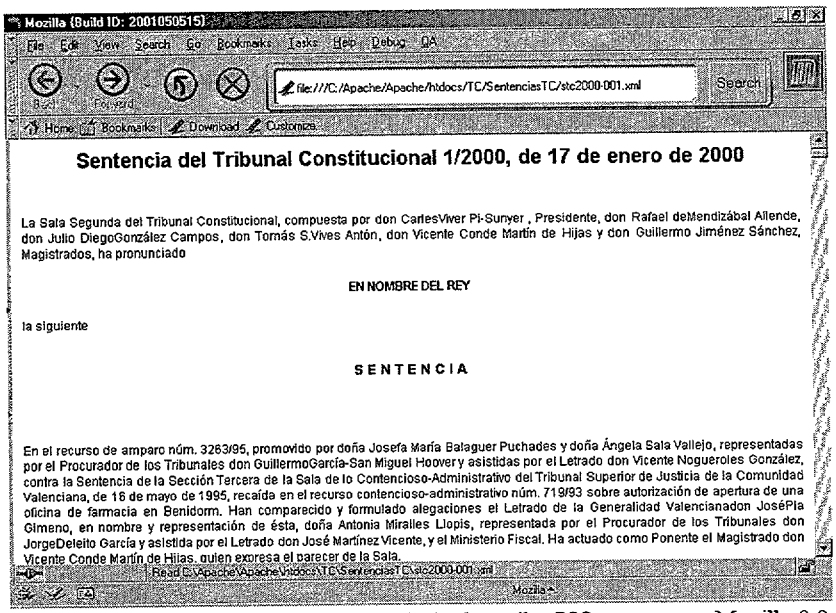


Figura IV.24: Presentación de *stc2000-001.xml* con hoja de estilo CSS externa en Mozilla 0.9/Navigator 6.

- Philip Shaw. *CSS Style Guide* [documento HTML]. Code Style, 2001. Disponible en http://www.codestyle.org/css/index.shtml (consultado el 24 de agosto de 2001).

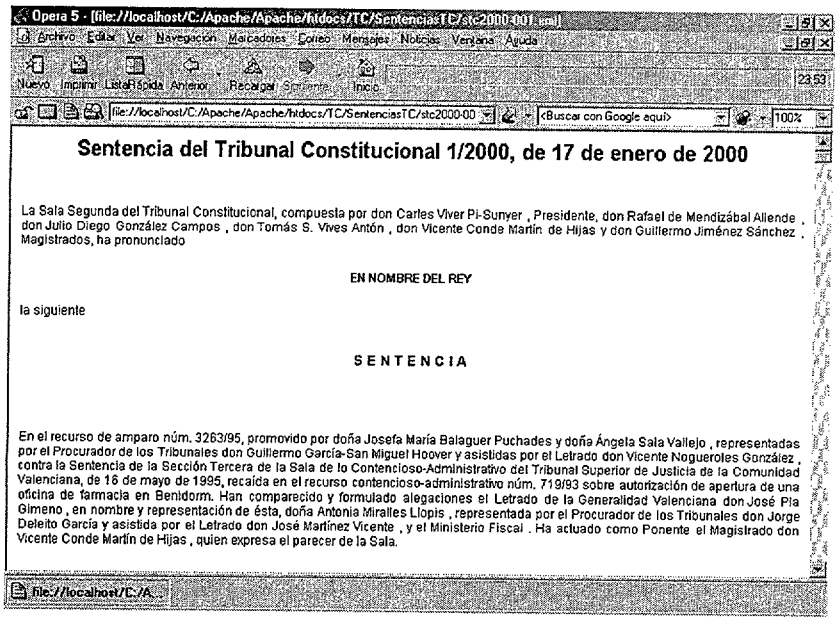


Figura IV.25: Presentación de *stc2000-001.xml* con hoja de estilo CSS externa en Opera 5.12.

Como se puede observar, el resultado ofrecido por cada uno de los navegadores analizados varía en algunos aspectos de la presentación del documento XML. Este es el caso del título o cabecera del documento que da entrada a la Sentencia del TC que el navegador IExplorer es incapaz de mostrar por la falta de cobertura que este navegador ofrece a la visualización a través de CSS de valores contenidos en los atributos de los elementos. Esto es, dado que el contenido del elemento *TitDocumento* que se desea mostrar se encuentra ubicado en el valor del atributo *entrada*, resulta necesario establecer la correspondiente propiedad CSS que permite representar el contenido de un atributo, propiedad ésta no reconocida por el IExplorer y sí por los otros navegadores. Las reglas CSS completas que se le asignaron al elemento *TitDocumento* para el establecimiento del formato de presentación del mismo fueron las siguientes:

- Eric A. Meyer. *WebReview.com's Style Sheet Reference Guide* [documento HTML]. WebReview.com, January 14, 2001. Disponible en http://www.webreference.com/style/css1/charts/mastergrid.shtml (consultado el 25 de mayo de 2001).

```
TitDocumento {
 text-align: center;}

TitDocumento:before {
 content:attr(entrada);
 font-size: 20px;
 font-weight: bold;
 padding-top: 5px;
 padding-bottom: 5px;}
```

Igualmente podemos establecer otra diferencia notable en la presentación del documento realizada por estos tres navegadorescon respecto al elemento *Enfasis*, con los valores de *nivel1* y *nivel2* para su atributo *codigo* (en las anteriores figuras tan sólo se visualiza lo sucedido con el nivel1, correspondiente al texto EN NOMBRE DEL REY). Dado que la presentación en pantalla de este elemento varía en función del valor que tome su atributo (para el *nivel1* y el *nivel2* el texto contenido debe aparecer centrado en la ventana del navegador y, sólo para el primer caso, en negrita), es necesario hacer uso en CSS de los denominados *selectores de atributos*. Este tipo especial de selector no es interpretado por el navegador IExplore. Las reglas CSS completas que se le asignaron al elemento *TitDocumento* para establecer el formato de presentación del mismo fueron las siguientes:

```
Enfasis[codigo="nivel1"] {
 display: block;
 padding-top: 20px;
 padding-bottom: 20px;
 text-align: center;
 font-weight: bold;}

Enfasis[codigo="nivel2"] {
 display: block;
 text-align: center;
 padding-top: 20px;
 padding-bottom: 20px;}
```

Otras observaciones de interés que se pueden hacer al método de presentación utilizado para desarrollar el modelo 1 de nuestro sistema están en la incapacidad de CSS para dotar

de comportamiento hipertextual a algunos elementos de los documentos XML y para generar nuevos elementos con sus correspondientes contenidos.

En el primer caso, en CSS, como decíamos, no se establece ninguna propiedad que haga posible que un determinado elemento se comporte como un ancla para el establecimiento de una relación hipertextual. En nuestro sistema resulta imprescindible que algunos elementos tengan dicha capacidad, como son todos los casos en los que se cita una norma jurídica o una resolución judicial, bien del propio TC o de otros tribunales inferiores. Por ejemplo, los elementos *RefSentencia* y *SubRefSentencia* han de ser presentados ante el usuario como anclas de origen de la relación hipertextual que se establece entre estos elementos y los documentos XML a los que está haciendo referencia a través del valor de su atributo *href*.

El navegador Opera, desde su versión 4.0, implementó a título particular esta asignación de hipertextualidad a los elementos XML a través de unas nuevas propiedades CSS, no normativas[21]. Para ello, si se deseara que los elementos antes mencionados poseyeran esta capacidad hipertextual, habría que haber establecido la siguiente regla:

```
RefSentencia, SubRefSentencia {
 color:blue;
 set-link-source:attr(href);
 use-link-source:current;}
```

Pero, insistimos, este modo de proceder sólo hubiera sido válido para la visualización de los documentos XML con el navegador Opera.

De igual modo, otra de las opciones posibles para solventar este problema hubiera podido venir del empleo del estándar *XML namespaces*. Como ya se señaló en el apartado III.2.3., este estándar XML asociado permite utilizar los elementos definidos en otras DTDs siendo, por tanto, factible hacer uso (y de hecho así se hace en numerosos casos) del elemento *A* (*anchor*) de la DTD del lenguaje HTML para lograr dicha hipertextualidad. Esta

[21] Véanse los comentarios ofrecidos al respecto de este asunto en Simon St. Laurent. *On Display: XML Web Pages with Opera 4.0* [documento HTML]. XML.com, April 19, 2000. Disponible en http://www.xml.com/print/2000/04/19/opera/index.html (consultado el 15 de junio de 2000).

solución, como ya se comentó igualmente en dicho apartado, no deja de ser en cierto sentido un pequeño truco no muy ortodoxo pues la integración de los *XML namespaces* en el modelo de la DTD no está contemplada y sí en el modelo de esquema XML. Además, de haber hecho uso de esta posibilidad hubiera sido necesario renombrar los elementos *RefSentencia* y *SubRefSentencia* (al igual que el resto que se deseara que se comportasen como anclas de hipertexto) por el nombre del elemento *A*. Las enormes pérdidas semánticas que ello hubiese supuesto hizo desaconsejable esta idea.

Por último, también podríamos haber optado por hacer uso del estándar asociado XLink para la creación de enlaces hipertextuales en los documentos XML. Este tema será tratado de forma más detallada en un posterior apartado.

En el segundo caso planteado, la imposibilidad de generar nuevos elementos, con sus correspondientes contenidos textuales, en la presentación que el usuario recibe en el navegador del documento XML resulta, igualmente, un serio problema con CSS.

Los documentos XML que han de ser presentados al usuario deben contener, además de todo el texto de la Sentencia del TC, otros elementos de ayuda para la navegación por el sitio web creado, esto es, los diversos enlaces hipertextuales que se pudieran establecer para que el usuario pueda, entre otras cosas, acceder al documento que contiene la Sentencia abreviada, a los diversos índices existentes, a la página de inicio del sitio, al motor de búsqueda, etc. Dado que gran parte de esta información (los URLs de estos destinos) no se encuentra contenida en los documentos XML de partida, sería del todo imposible a través de CSS incluirla en la presentación que de éstos se hace al usuario. Se podrían haber creado nuevos elementos (con sus correspondientes etiquetas) dentro de cada uno de los documentos XML que contemplasen estas opciones pero ello ocasionaría, entre otras cosas, una profunda alteración de la estructura lógica del documento, así como la necesidad de incluir manualmente en cada uno de estos documentos la información pertinente para establecer esas relaciones hipertextuales.

Lo mismo puede decirse respecto a la posible inclusión de elementos gráficos dentro de la presentación que de estos documentos se realiza.

Finalmente, señalaremos en este apartado de asignación de un formato de estilo a los documentos XML tratados que no se estimó conveniente el empleo del estándar XML asociado XSL-FO (*Extensible Stylesheet Language – Formating Objects*), comentado brevemente en el apartado III.2.3. de esta tesis doctoral. Como ya se señaló en aquel momento, se trata de una tecnología propia de XML de gran potencia y versatilidad para la asignación de formatos de presentación a los documentos XML. Pero, lamentablemente, su consolidación como estándar oficial del W3C no se ha producido hasta fechas muy recientes (octubre de 2001). Además, ninguno de los actuales navegadores web incorpora, aun mínimamente, capacidades para procesar e interpretar este método de asignación de características de presentación a los documentos XML. Por otro lado, el servidor web XML empleado, Apache/Cocoon, sí implementa estas capacidades avanzadas de integración de XSL-FO, pero desde la óptica de la transformación de formatos: los documentos XML se transforman mediante dicha tecnología al formato PDF, orientando, por tanto, esta presentación a un formato adecuado para su impresión.

Por todos estos motivos, el empleo de XSL-FO se desestimó para el desarrollo actual de nuestro sistema, pero con la esperanza de que la cobertura que de él se haga en futuras herramientas para la Web sea toda una realidad en poco tiempo pues, a diferencia de CSS, se trata de un estándar propio de la gran familia de lenguajes XML. Ello hubiera solucionado gran parte de los problemas con los que nos hemos ido encontrando a lo largo del desarrollo práctico aquí expuesto.

En cualquier caso, los problemas señalados con el uso de la tecnología CSS aplicada a nuestros documentos quedaron resueltos, en gran medida, con el cambio de planteamiento de nuestro sistema, expuesto con anterioridad: del modelo 1 pasaríamos al modelo 2 en donde jugará un papel de suma importancia el estándar XSLT, como veremos en el siguiente apartado.

IV.2.3. LA TECNOLOGÍA XSLT PARA LA TRANSFORMACIÓN DE LOS DOCUMENTOS

En apartados precedentes se han venido señalando los razonamientos que nos hicieron abandonar el modelo 1 de desarrollo y decidirnos, por tanto, por el empleo de la tecnología XSLT (*Extensible StyleSheet Language Transformation*) que hemos tratado en sus líneas generales en el apartado III.2.3. relativa a los estándares XML asociados.

Es necesario advertir antes de empezar con la explicación y desarrollo de este segundo y definitivo modelo que la Definición del Tipo Documental, el esquema XML y, por tanto, las marcas empleadas en un principio para este sistema sufrieron unas pequeñas transformaciones para adaptarlas a este nuevo entorno. Aunque no vamos a extendernos en este punto para no alargar en exceso la redacción del presente apartado, sí resulta conveniente señalar que la mayor parte de las modificaciones que se realizaron consistieron en la definición e inclusión de nuevos atributos en una serie de elementos. Con ello aumentábamos el control sobre la normalización de la información contenida en éstos y facilitábamos los posteriores procesos de extracción de los datos de interés que habrían de ser incorporados a la plantilla de elementos HTML.

Ya se señaló, quedando reflejado en el figura IV.3., que la aplicación de la tecnología XSLT a los documentos XML de partida se hizo para lograr generar de forma automática tres tipos distintos de documentos HTML: documentos a texto completo, documentos abreviados y los distintos índices de acceso a las Sentencias del TC (cronológico, por Salas y por materias).

Analicemos de forma más detallada el desarrollo de cada uno de estos productos generados a través de la aplicación de XSLT.

IV.2.3.1. GENERACIÓN DE DOCUMENTOS HTML A TEXTO COMPLETO

La tecnología XSLT ha supuesto para los miembros del grupo de Tecnologías de la Información de este Departamento un gran hallazgo. A pesar de ser uno de los estándares asociados a XML que más pronto fue establecido como Recomendación del W3C (noviembre de 1999), su análisis y aplicación no habían sido abordados en profundidad por nuestro colectivo debido, en gran medida, al hecho de ser un estándar surgido desde esa orientación de XML a la que hemos venido denominado *a los datos*.

La profundidad que nos ha dado el estudio y aplicación de esta tecnología ha hecho que nuestra visión haya cambiado en gran medida debido a los innegables beneficios que de ella hemos obtenido. Además de las posibilidades que ofrece generar nuevos documentos XML a partir de ciertos datos contenidos en otros (transformación dentro del mismo formato), tema del cual se hablará en el apartado IIV.2.3.3, el mayor beneficio para el sistema desarrollado ha venido de su potencialidad para generar nuevos documentos con formatos distintos a los de partida: de XML a HTML[22].

Las limitaciones que comentamos en el apartado anterior dedicado al estándar CSS del W3C podrían ahora ser superadas en gran medida mediante esta transformación de formato pues se contaba ahora con los mecanismos necesarios para poder presentar el contenido textual existente en los atributos de los elementos. De igual modo, se podría otorgar una capacidad hipertextual a aquellos elementos del documento XML que así lo requiriesen. Asimismo, era posible añadir todos los elementos gráficos, textuales o hipertextuales que se considerase de interés incluir en la presentación del documento que el usuario recibiría a través de su navegador. Y, ante todo, se garantizaba, si así lo deseábamos, la integridad en la visualización del contenido del documento

[22] Ya se ha señalado en diversas partes de esta tesis doctoral que la tecnología XSLT permite igualmente establecer otras transformaciones del formato original XML, como son los casos del formato PDF y MS Word, entre otros. Para nuestro desarrollo práctico no ha sido necesario hacer uso de estos formatos tan comunes de documentos electrónicos por lo que la dinámica en la generación de los mismos no será explicada.

independientemente del desarrollador y la versión del navegador utilizado por el usuario. Empezaremos analizando esto último.

Como ya hemos señalado repetidamente a lo largo de este capítulo, la gran ventaja del formato HTML reside en su correcta interpretación[23] por parte de los navegadores web existentes en la actualidad, pues ése ha sido el fin con el que éstos han ido surgiendo a lo largo de estos años. Por tanto, al trasladar los datos contenidos en los documentos XML al formato HTML, aseguramos que éstos vayan a ser presentados ante el usuario, independientemente del navegador empleado. Bien es cierto que, como se ha señalado anteriormente y mostrado en los diferentes gráficos expuestos, en este segundo modelo se sigue haciendo uso del estándar CSS para asignar un estilo de presentación a cada uno de los elementos contenidos en los documentos HTML generados automáticamente. Ello podría nuevamente ocasionar problemas en la presentación final de los mismos pero a pesar de ello se ha decidido mantener esta solución debido, entre otras razones, a que no se deseaba hacer uso del atributo *style* dentro de los elementos HTML[24]. Por este motivo, y tratando de seguir los estándares con la mayor fidelidad posible, se decidió mantener el desarrollo de estas hojas de estilo asociadas a los documentos HTML. De todas formas, para minimizar en lo posible los errores que pudieran cometer los distintos navegadores web en la presentación de los documentos, las propiedades CSS contenidas en los ficheros desarrollados (pues como veremos con posterioridad fueron dos) corresponderían al nivel 1 de este estándar, interpretadas de forma correcta por la práctica totalidad de ellos.

[23] Esta afirmación habría que matizarla mucho más pues en realidad esta interpretación del lenguaje HTML no es tan completa como se ha afirmado. No deseamos extendernos en este punto para no alargar en exceso el desarrollo de este capítulo pero aún hoy hay que lamentar que existen algunos aspectos del lenguaje HTML que no son correctamente interpretados por algunos de los navegadores web, como es el caso, por ejemplo, de la inclusión de marcos flotantes por medio del elemento *iframe*, dentro de los documentos HTML, que algunos navegadores (caso de Opera) no reconocen y, consiguientemente, no lo muestran en la ventana principal de visualización.

[24] Su uso, aunque admitido por el W3C desde la versión 4 de este estándar, es el que resulta más pobre y menos útil al no poder incorporar selectores y tener que establecer los cambios de estilo, en el caso de ser necesario, elemento a elemento.

El funcionamiento de la tecnología XSLT para realizar esta transformación del formato XML al HTML, así como para la traslación de los contenidos de uno a otro, resulta, en líneas generales y para este tipo de aplicación, sencilla de entender. Como en este caso se deseaba trasladar la misma estructura de los documentos XML a los documentos HTML generados, el proceso debería de partir de la construcción de una plantilla general o armazón en el que se define la estructura de presentación de los contenidos en el formato HTML. A partir de él, y como si de una *matrioska* se tratase, se van definiendo desde lo más general a lo más particular otras subplantillas o bloques contenidos. Dentro de cada uno de éstos, y con las instrucciones oportunas, se va indicando el lugar y contenido concreto del documento XML del que se ha de extraerse la información que se ubicará en cada uno de los elementos de las citadas plantillas. Para la generación de cada una de éstasse emplea el elemento *xsl:template* junto con el atributo *match* para asignarle un nombre. Para indicar que en una determinada parte de esa plantilla se van a insertar otras subplantillas se hace uso del elemento vacío *xsl:apply-templates*. Igualmente, para indicar cuál ha de ser el contenido del documento XML que se ha de incrustar en un determinado elemento o atributo HTML se emplea el elemento *xsl:value-of* junto con el atributo *select*. Por último, si algún elemento HTML tiene que aparecer tantas veces como su homólogo en los documentos XML, se debe hacer uso del elemento *xsl:for-each* junto con el atributo *select*. La sintaxis que se ha de emplear para indicar el valor que toma este atributo (es decir, de dónde ha de tomar los datos) está basado en el estándar XPath.

Con estas ideas generales, se procedió a la elaboración del documento XSLT que serviría para realizar las transformaciones al formato HTML de cada uno de los documentos XML que contenían las diversas Sentencias del TC tratadas. Tras una serie de pruebas y modificaciones, la versión definitiva tomó el nombre de **SentenciaTC.xsl**. El contenido completo de este documento se encuentra disponible en el Apéndice 11 de esta tesis doctoral, al cual remitimos para un mejor entendimiento de los procesos descritos a continuación.

Todos los procesos realizados en relación con la construcción y validación de este documento XSLT (al igual que los restantes) fueron llevados a cabo con el módulo correspondiente del editor *XML Spy*.

Analicemos algunos de los apartados más sobresalientes de este documento elaborado.

Como ya se ha señalado, en primer lugar era necesario crear la armazón o plantilla general del documento HTML que sería presentado de forma *virtual* al usuario. Esto significa que deben definirse los dos bloques principales de todo documento HTML (cabecera y cuerpo) y dentro de éstos aquella información que ya podía incluirse de forma directa a través de su asociación con elementos y/o atributos de los documentos XML. Así, en la cabecera del documento HTML se pueden incluir directamente el título, las etiquetas meta y los enlaces a las hojas de estilo CSS que se iban a aplicar pues, en este último caso, se deseaba que este documento llevase asociadas dos hojas de estilo, cada una de ellas para un medio de salida distinto (pantalla de ordenador e impresora)[25]. El fichero CSS que define la presentación de los documentos en pantalla tomó el nombre de **SentenciaTC.css** y el correspondiente a la presentación a través de impresora tomó el nombre de **SentenciaTC2.css**. Ambos se encuentran incluidos dentro del Apéndice 12 de esta tesis doctoral.

Dentro de cada una de estas etiquetas del documento HTML se establecieron las instrucciones oportunas para indicar al procesador el lugar exacto dentro de los documentos XML de donde debería tomar dicha información cuando se procediese a la transformación y presentación de los mismos: para el título del documento (elemento *title*) se tomaría del valor existente en el atributo *entrada* del elemento *TitDocumento*, los valores

[25] No nos detendremos demasiado en la explicación de este hecho. Tan sólo aclarar que se consideró oportuno hacer uso de esta funcionalidad del lenguaje CSS, estableciendo una forma de presentar los documentos HTML a través de la pantalla (con todos los elementos gráficos, diversidad de colores, fuentes, etc.) y otra distinta para cuando dicho documento fuese impreso (sin los elementos gráficos, fuentes distintas, etc.). Los principales navegadores existentes en la actualidad interpretan estas características y, además, nuestras experiencias en cursos de formación sobre CSS habían sido muy positivas al respecto de este doble uso.

contenidos en las diversas etiquetas *meta* establecidas[26], las de autor, palabras clave y descripción del recurso, de cada uno de los atributos *entrada* de los elementos *Autor*, *Descriptor* y *Extracto*, respectivamente. Finalmente, se establece la ubicación de la carpeta y el nombre de los ficheros CSS que han de dar formato a estos documentos HTML. Todo ello queda reflejado en el código siguiente:

```
<head>
<title><xsl:value-of
select="Cabecera/TitDocumento/@entrada"/></title>
<meta name="Author" content="{Cabecera/Autor/@entrada}"/>
<xsl:for-each select="Cabecera/Materia/Descriptor">
<meta name="keywords" content ="{@entrada}"/>
</xsl:for-each>
<meta name="Description"
content="{Cabecera/Extracto/@entrada}"/>
<link rel="stylesheet" href="../docsCSS/SentenciaTC.css"
type="text/css" title="hoja de estilo para pantalla"
media="screen"/>
<link rel="stylesheet" href="../docsCSS/SentenciaTC2.css"
type="text/css" title="hoja de estilo para impresora"
media="print"/>
 </head>
```

Para desarrollar el cuerpo del documento HTML la operación es algo distinta: se establece la estructura global del cuerpo del documento dejando el espacio para que las diferentes plantillas se vayan ubicando dentro del mismo de un modo jerárquico, así como la descripción de aquellos otros elementos textuales y gráficos que, no estando contenidos en los documentos XML originales, son comunes a la presentación de todos los documentos HTML y que se desea mostrar ahora para enriquecer y completar la presentación que se realiza de los mismos. Éste es el caso de las barras horizontales, la tabla

[26] En realidad, resulta ilógico incluir los elementos meta dentro de estos documentos HTML *virtuales* dado que al poseer dicha característica nunca podrán ser analizados por los tradicionales robots de indexación que operan en el espacio Web. Los documentos HTML, insistimos, no tienen una realidad física, tan sólo los documentos XML. Así, aunque la visualización que haga de los mismos el usuario en su navegador (tanto del propio documento como de su código fuente) sea en formato HTML, el contenido de los mismos nunca podrá ser indexado. Sí el de los documentos XML que tienen realidad física en el sistema, pero eso está aún muy lejos de suceder, al menos en un futuro cercano. Las etiquetas meta se han incluido, por tanto,

que ha de contener las imágenes que enlazan a otras partes del sistema (documento abreviado, página de inicio, índices y motor de búsqueda) y otros elementos finales, como el nombre de los autores de este desarrollo práctico. El código establecido para ello quedaría, por tanto, de la siguiente forma:

```
<body>
<h2><xsl:value-of
select="Cabecera/TitDocumento/@entrada"/></h2>
<p class="boe">[BOE nº <xsl:value-of
select="Cabecera/BOE/@numero"/>, de <xsl:value-of
select="Cabecera/BOE/@fecha"/>]</p>

<xsl:apply-templates/>

<hr/>
<br/>
<table width="100%" border="0">
<tr>
<td align="center"><a href="../index.xml"><img
src="../imagenes/pagprincipal.jpg" alt="ir página principal"
border="0"/></a></td>
</tr>
</table>
<br/>
<br/>
<table width="100%" border="0">
<tr>
<td align="center" width="33%"><a
href="SentenciasTCbreves/{Cabecera/FicheroBreve/@entrada}"><
img src="../imagenes/docbreve.jpg" alt="ir doc. abreviado"
border="0"/></a></td>
<td align="center" width="33%"><a
href="../indices/indices.xml"><img
src="../imagenes/indices.jpg" alt="ir indices"
border="0"/></a></td>
<td align="center" width="33%"><a
href="../buscador.xml"><img src="../imagenes/buscador.jpg"
alt="ir buscador" border="0"/></a></td>
</tr>
</table>
<br/>
<hr/>
```

únicamente para experimentar con este elemento tan característico de los documentos HTML. Estas ideas volverán a ser retomadas al hablar con posterioridad de la asignación de metadatos al sistema desarrollado.

```
<br/>
<address>
<p>Bonifacio Martín Galán - J. Tomás Nogales Flores / Dpto.
Biblioteconomía y Documentación / Universidad Carlos III de
Madrid</p>
</address>
</body>
```

El resto del documento se va desarrollando de forma similar: se define la presentación de cada uno de los grandes bloques estructurales (Preámbulo, Antecedentes, Fundamentos Jurídico, Fallo y Votos) y dentro de éstos los siguientes subbloques o subplantillas que se han de integrar. Para no alargar en exceso esta explicación, se mostrará el caso de los votos particulares, donde se definen la forma de presentación del bloque *Votos*, la del elemento *VotoParticular* y, finalmente, la del elemento *CabeceraVoto*:

```
<xsl:template match="Votos">
<div class="votos">
<a name="votos"></a>
<xsl:apply-templates/>
</div>
<br/>
</xsl:template>

<xsl:template match="VotoParticular">
<div class="votoparticular">
<a name="{@numero}"></a>
<xsl:apply-templates/>
</div>
<br/>
</xsl:template>

<xsl:template match="CabeceraVoto">
<p class="cabeceravoto">
<xsl:apply-templates/>
</p>
</xsl:template>
```

El proceso de desarrollo integral de este documento XSLT no ha sido tarea fácil, dando lugar a numerosas pruebas y correcciones a lo largo del tiempo. Pero la eficacia lograda finalmente ha merecido la pena, máxime cuando, como sabemos, que se trata de una plantilla de aplicación a cualquier documento XML de Sentencia del TC (uno o un millón

de ellos) para su transformación y presentación automática en formato HTML. Tan sólo resta, por tanto, añadir el oportuno enlace a este documento XSLT dentro del prólogo de cada uno de los documentos XML existentes para lo que se emplea una instrucción de procesamiento muy similar al utilizado en el modelo 1 para enlazar los documentos XML con la hoja de estilo CSS; ahora esta se suprime y en su lugar se sitúa la correspondiente a XSLT:

```
<?xml-stylesheet type="text/xsl"
href="../docsXSL/SentenciaTC.xsl"?>
```

Es necesario señalar que al haber sido empleado el servidor web XML Cocoon para la presentación en la WWW de estos documentos XML, es necesario que todos ellos incluyan una nueva instrucción de procesamiento en su prólogo para el correcto procesamiento por parte del servidor, como se expondrá con posterioridad de forma más detallada en el apartado IV.2.5.1 del presente capítulo.

En cualquier caso, el resultado de todo este trabajo es que los documentos XML sean presentados ante cualquier navegador web de forma homogénea en un formato HTML, tanto en su versión para la presentación en pantalla como por impresora, mostrándose en las siguientes figuras el resultado obtenido para el primer medio mencionado:

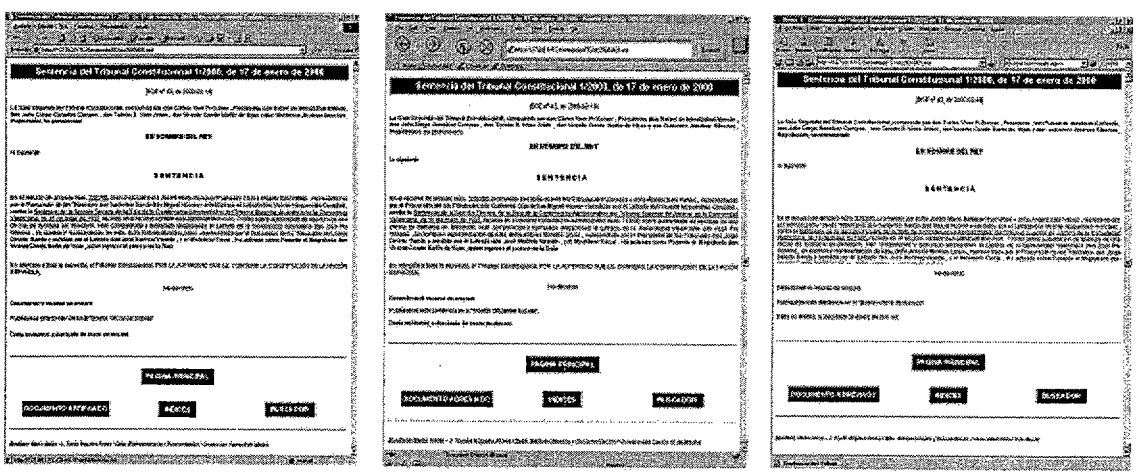


Figura IV.26: Presentación del documento HTML *virtual* a texto completo en los distintos navegadores web.

IV.2.3.2. GENERACIÓN DE DOCUMENTOS HTML ABREVIADOS

Otra de las aportaciones fundamentales del modelo práctico desarrollado ha sido fruto del deseo de ofrecer al usuario la posibilidad de acceder a la información contenida en las Sentencias del TC en un formato de presentación abreviado. Esta forma de acceso es característica de algunas bases de datos jurídicas (véase el caso, por ejemplo, de las bases de datos de La Ley), donde los datos más relevantes de las resoluciones de los tribunales son mostrados en un documento aparte, facilitando así, de un modo directo, la rápida comprensión de la información contenida.

XSLT nos ofrece nuevamente la posibilidad de llevar a la práctica esta idea. De este modo, partiendo de la información marcada en los documentos XML de origen, es posible extraer de ellos los datos más relevantes y presentarlos ante el usuario en un formato de documento HTML abreviado. Esta forma de operar está muy cercana a la denominada orientación a los datos pues ahora no resulta tan fundamental la presentación del texto de los documentos y sí la plasmación de los datos más característicos de los mismos.

En los modelos XML plenamente orientados al almacenamiento y gestión de datos, los documentos de partida suelen estar, como hemos venido señalando, perfectamente estructurados, y la rigidez de estas estructuras es un factor determinante para el buen desarrollo de las bases de datos en las que aquellos han de almacenar, así como para la futura presentación que de los mismos se haga a través de Internet. Estas estructuras documentales reflejan en gran medida las estructuras de las bases de datos que han de tratar los datos contenidos, por lo que resulta fundamental que éstos se agrupen de forma similar a como lo harían dentro de los campos y tablas de dichas bases de datos. En el caso de las estructuras de los documentos de carácter eminentemente textual, ya hemos señalado en anteriores ocasiones que las estructuras que los caracterizan son bastante más flexibles, lo que ocasiona ciertos problemas para su gestión a través de los tradicionales modelos de bases de datos relacionales y, por extensión, con las tecnologías XML que se basan en estos criterios. Y esto es lo que sucede con la aplicación de XSLT para este segundo desarrollo.

Si decimos que las estructuras existentes en los documentos de carácter textual suelen ser flexibles, significa que las agrupaciones de algunos de los elementos contenidos en los mismos lo son de igual forma. Y así es, como lo hemos venido señalando en el capítulo IV.1. Sabemos que el modelo de contenido de algunos elementos de nuestra DTD o esquema queda muy abierto, pudiendo aparecer o no algunos de los subelementos contenidos y con una frecuencia de aparición muy variable. Y será aquí, en el intento de agrupación de ciertos elementos de los documentos XML, donde nos encontraremos con mayores problemas, algunos de ellos insalvables. Todo ello obliga, en la medida de lo posible, a transformar la estructura ambigua de los documentos XML originales a otra estructura más rígida y estable, lo que invariablemente conlleva la generación de un nuevo documento XML antes de que se produzca la transformación definitiva al formato HTML de presentación de los datos.

El siguiente gráfico muestra de forma global los diferentes pasos necesarios para producir estos documentos abreviados.

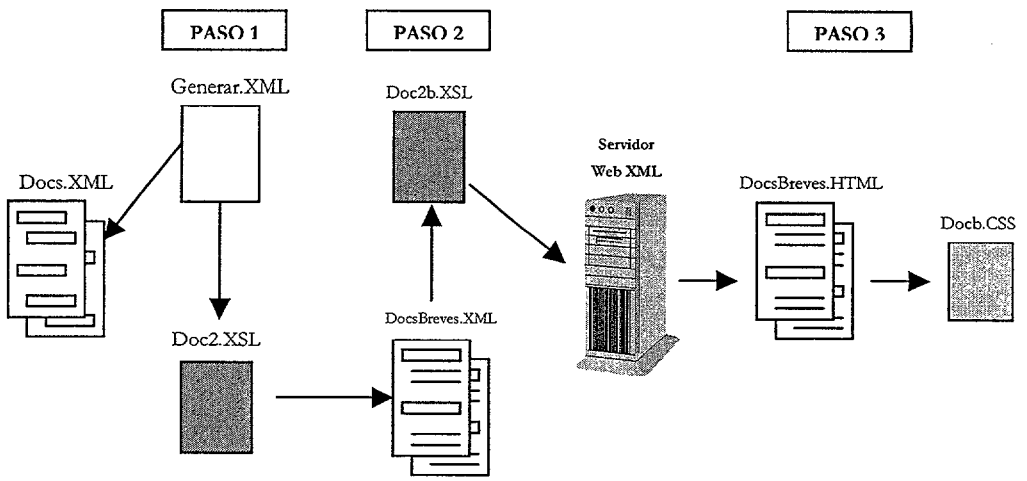


Figura IV.27: Esquema general para la generación de documentos HTML abreviados.

Dentro del Paso 1 se incluyen todas aquellas tareas necesarias para generar nuevos documentos XML desde los documentos XML con el texto íntegro de las Sentencias del

TC, con la finalidad de extraer sus datos más característicos y establecer, asimismo, una estructura más rígida y manejable de los mismos. Tal y como se puede observar en la figura anterior, se ha hecho uso de un nuevo documento XML para dar soporte al documento XSLT que ha de generar los documentos XML abreviados. Este documento XML tan sólo contiene la instrucción de procesamiento[27] que llama al fichero XSLT encargado de realizar la transformación, así como un único elemento vacío. Dicho elemento no tiene ninguna función pero para que sea válido y el parser pueda actuar necesitamos incluir al menos un elemento de este tipo. Este documento XML tomó el nombre dentro de **baseXMLbreve.xml** dentro de nuestro sistema, y dado la brevedad de su contenido lo exponemos a continuación:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- edited with XML Spy v3.5 (http://www.xmlspy.com) by
(Bonifacio Martín Galán, J. Tomás Nogales Flores) -->
<!-- Este documento sólo sirve de base o plantilla para la
transformación de los documentos XML que contienen las
Sentencias completas al documento abreviado correspondiente
-->
<?xml-stylesheet type="text/xsl"
href="../../docsXSL/generarXML/genXMLbreve.xsl"?>
<!-- Recordar que para cada nuevo documento XML abreviado
hay que cambiar la referencia al fichero afectado dentro de
la variable definida en el fichero XSLT (genXMLbreve.xsl) --
>
<Entrada/>
```

Como se puede observar, la instrucción de procesamiento está haciendo una llamada al documento XSLT encargado de realizar la transformación de los documentos XML

[27] Se podía haber incluido esa nueva instrucción dentro de cada uno de los documentos XML de las Sentencias del TC pero dado que ya existía una instrucción para la generación de los documentos HTML virtuales a texto completo, hubiera supuesto una incompatibilidad entre estas dos instrucciones de procesamiento (resulta tener dos instrucciones de procesamiento XSLT dentro de un mismo documento XML). Para no tener que estar ocultando una u otra dependiendo de lo que se desease realizar (o bien presentar el documento o bien generar uno nuevo), se consideró más oportuno realizar un nuevo documento

originales al formato XML abreviado. Este documento, denominado **genXMLbreve.xsl**, juega un papel de vital importancia en el desarrollo del sistema aquí expuesto. El contenido íntegro del mismo se encuentra disponible en el Apéndice 13 de esta tesis doctoral.

Este fichero XSLT es bastante más complejo que el elaborado con anterioridad pues ahora es necesario hacer un uso más intensivo de características avanzadas de este estándar, algunas de las cuales se han demostrado limitadas para poder llevar a cabo lo que se deseaba. Sin ánimo de extender esta explicación, señalaremos brevemente los aspectos más característicos de la misma:

- En primer lugar, y a diferencia del anterior fichero XSLT, es necesario especificar que la transformación que se va a realizar debe de producir un documento XML. Para ello se emplea el siguiente elemento con sus correspondientes atributos:

```
<xsl:output method="xml" version="1.0" encoding="ISO-8859-1"
indent="yes"/>
```

- Se ha creado una variable dentro de este documento para designar el documento XML original del cual ha de tomar los datos para producir la transformación. Lógicamente, como así se advierte en uno de los comentarios existentes, el nombre del fichero referenciado en esta variable hay que cambiarlo cada vez que se desee generar un documento abreviado para cada uno de los documentos XML que contienen los textos de las Sentencias del TC tratadas. Esta variable se define como sigue:

```
<xsl:variable name="sentencia" select="document
('../../SentenciasTC/stc2000-001.xml')"/>
```

- A partir de aquí, cada vez que se desee transformar los elementos y los atributos del documento de partida en otros nuevos, tan sólo será necesario hacer referencia a esta variable junto con la localización exacta de dichos elementos y atributos en el documento XML original. Por ejemplo, el bloque original del *Fallo* de las Sentencias del

XML que sirviese únicamente de soporte al documento XSLT que ha de generar los documento XML transformados.

TC se desea transformarlo de tal manera que ahora sólo se recoja dentro del mismo el elemento *Decision* y el *Resumen* del mismo (recordemos que dicho resumen se encuentra en los documentos XML de partida formando parte del elemento *Cabecera* y no dentro del propio *Fallo*); el código establecido quedaría, por tanto, de la siguiente forma:

```
<Fallo>
 <Decision><xsl:value-of
select="$sentencia/SentenciaTC/Fallo/Decision/@codigo"/></De
cision>
 <Resumen><xsl:value-of
select="$sentencia/SentenciaTC/Cabecera/ResumenFallo/@entrad
a"/></Resumen>
</Fallo>
```

- Pero, sin duda, los mayores problemas vinieron en el intento de establecer agrupaciones complejas de elementos y subelementos de los documentos de partida. La idea era en el plano teórico muy interesante: deseábamos extraer de los bloques de *Antecedentes* y de *FundamentosJuridicos* todas las referencias que existiesen a otros Autos y Sentencias del TC citadas y, una vez extraídas, agrupar todas las SubReferencias que existiesen a cada una de ellas (de una determinada Sentencia todos los Antecedentes y Fundamentos Jurídicos citados). Lamentablemente, esto se reveló imposible de realizar debido en gran medida a las limitaciones del propio estándar XSLT para la confección de este tipo de agrupaciones complejas de elementos y subelementos contenidos de forma dispersa y ambigua en los textos[28]. Por este motivo, sólo se pudo realizar una agrupación de Sentencias y Autos citados que, por otro lado, se revelaba más que suficiente para la futura presentación de los documentos en formato abreviado. Este nivel de agrupación obtenido de Sentencias y Autos del TC puede observarse en el caso, por ejemplo, de lo definido para el bloque de *FundamentosJuridicos*:

[28] Los diversos intentos que se realizaron han quedado reflejado en el propio texto del fichero XSLT, por lo que recomendamos la consulta del Apéndice 13 para un mayor detalle sobre este problema.

```
<FundamentosJuridicos>
 <xsl:for-each
select="$sentencia/SentenciaTC/FundamentosJuridicos/Fundamen
to/Parrafo/RefSentencia">
 <RefSentencia>
      <xsl:attribute name="href">
      <xsl:value-of select="@href"/>
      </xsl:attribute>
      <Codigo><xsl:value-of select="@codigo"/></Codigo>
      <Entrada><xsl:value-of select="."/></Entrada>
 </RefSentencia>
 </xsl:for-each>
</FundamentosJuridicos>
```

Una vez finalizada la confección de este documento XSLT, tan sólo restaba activar el documento XML de base antes realizado para llevar a cabo la transformación del documento XML original en un nuevo documento XML abreviado, entrando de este modo en el **Paso 2.**

Ahora, a diferencia de lo que sucede cuando transformamos de XML a HTML, sí que se desea que los nuevos documentos generados sean almacenados físicamente en el sistema, no de forma virtual, pues constituyen unconjunto documental que será finalmente presentado al usuario en formato HTML. De este modo, activando la aplicación XSLT contenida en el editor *XML Spy* se generaba de forma automática esta transformación, restando únicamente asignar un nombre al fichero generado y guardarlo físicamente en la carpeta creada a tal fin (*SentenciasTCbreves*). A pesar de la dificultad que supuso la creación del documento XSLT antes mencionado, los resultados que se obtuvieron con ello eran ciertamente importantes: ahora estábamos en disposición de transformar cualquiera de los documentos XML con el texto íntegro de las Sentencias del TC con un simple cambio en la referencia de la variable antes citada (para indicar el nombre del fichero a transformar) y un solo clic.

El documento XML abreviado correspondiente al documento XML completo que hemos venido tomando como ejemplo para el desarrollo de esta explicación (*stc2000-001.xml*) tomó el nombre de **stc2000-001b.xml**, encontrándose disponible para su consulta en el Apéndice 14 de esta tesis doctoral.

En la anterior transformación de documento completo a documento abreviado también se incluían una serie de instrucciones de procesamiento, la propia para que pueda ser interpretado por el servidor Web XML Apache/Cocoon, y la necesaria para que estos documentos XML abreviados fueran finalmente transformados al formato HTML para la presentación de los contenidos al usuario. Nuevamente, por tanto, hubo que confeccionar otro documento XSLT que llevase a cabo esta transformación, ahora sí, de formatos, que tomó el nombre de **SentenciaTCb.xsl** y se encuentra disponible para su consulta en el Apéndice 15 de esta tesis.

Este nuevo documento XSLT es muy similar al desarrollado para la transformación de los documentos XML originales al formato de presentación HTML a texto completo, por lo que no nos extenderemos en la explicación de su desarrollo. Tan sólo destacaremos que resultaba necesario tratar adecuadamente aquellos elementos que se deseaba que funcionaran como anclas de origen de relaciones hipertextuales con otros documentos electrónicos (el Recurso planteado ante el TC, así como los distintos Autos y Sentencias citadas a lo largo de la Sentencia recogida) a la hora de confeccionar este documento XSLT, tal y como se muestra en el siguiente ejemplo para dotar de esta capacidad hipertextual a cada uno de los Autos y Sentencias recogidas en el apartado de los Fundamentos Jurídicos:

```
<xsl:template match="FundamentosJuridicos">
<h3>FUNDAMENTOS JURÍDICOS</h3>
<table class="entrada" width="95%" border="0">
<tr>
<th width="200px" align="left" valign="top">SENTENCIAS
CITADAS</th>
<td>
<xsl:for-each select="RefSentencia">
<a href="{@href}"><xsl:value-of select="."/></a><br/>
</xsl:for-each>
</td>
</tr>
</table>
<br/>
</xsl:template>
```

Como se puede deducir de este ejemplo, la presentación de los distintos bloques de agrupación de datos principales de las Sentencias del TC se realizaría a través de la

definición de diversas tablas, algunas de ellas anidadas dentro de otras. Con esto podíamos lograr de forma sencilla una indentación o tabulación de todos los datos, facilitando la visualización y rápida comprensión de los mismos, tal y como se verá a continuación.

El **Paso 4** era ya, con la experiencia adquirida, sencillo de llevar a cabo: tan sólo restaba crear las dos hojas de estilo CSS (ya referenciadas igualmente en el documento XSLT) para la presentación de los elementos HTML descritos. Al igual que en ocasiones anteriores, se confeccionaron estas dos hojas de estilo para poder establecer una diferenciación en el formato de presentación de los documentos HTML abreviados según se realizase a través de la pantalla del ordenador o la impresora. Estas dos hojas de estilo CSS tomaron los nombres de **SentenciaTCb.css** (para la presentación en pantalla) y **SentenciaTCb2.css** (para la impresión), encontrándose disponibles en el Apéndice 16.

Tal y como se planteó para la presentación de los documentos HTML virtuales a texto completo, estos ficheros CSS deberían incluir propiedades sencillas del nivel 1 de este estándar, evitando de este modo los errores que los diversos navegadores web pudieran cometer por su deficiente interpretación de los niveles superiores de este lenguaje.

Como resultado final de todo este proceso se obtuvo una óptima presentación de estos documentos abreviados, como se puede observar en la siguiente figura para el navegador web IExplorer 6, si bien la presentación de este documento sería idéntica para el resto de navegadores web.

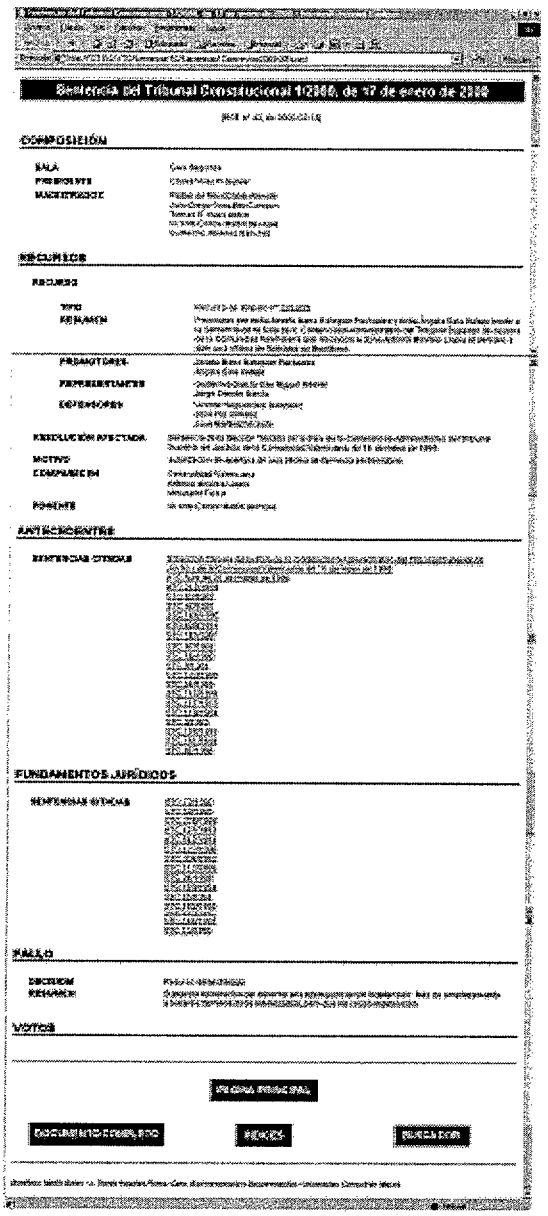


Figura IV.28: Presentación del documento HTML *virtual* en formato abreviado en el navegador IExplorer 6.

IV.2.3.3. GENERACIÓN DE INDICES DE ACCESO A LOS DOCUMENTOS

La generación de los diversos índices de acceso a los documentos XML comporta un mayor grado de complejidad si cabe que los anteriores procesos descritos debido en gran medida al número de piezas que han de intervenir en este nuevo proceso. Nuevamente deberá hacerse uso de la tecnología XSLT tanto para producir nuevos documentos XML en el formato abreviado que aglutine todos los datos requeridos de cada uno de los documentos XML de origen, como para, finalmente, generar cada uno de los diversos índices en formato HTML con sus correspondientes hojas de estilo CSS asociadas.

El siguiente gráfico muestra de forma global los diferentes pasos necesarios para producir los distintos índices en formato HTML para el acceso a los documentos XML:

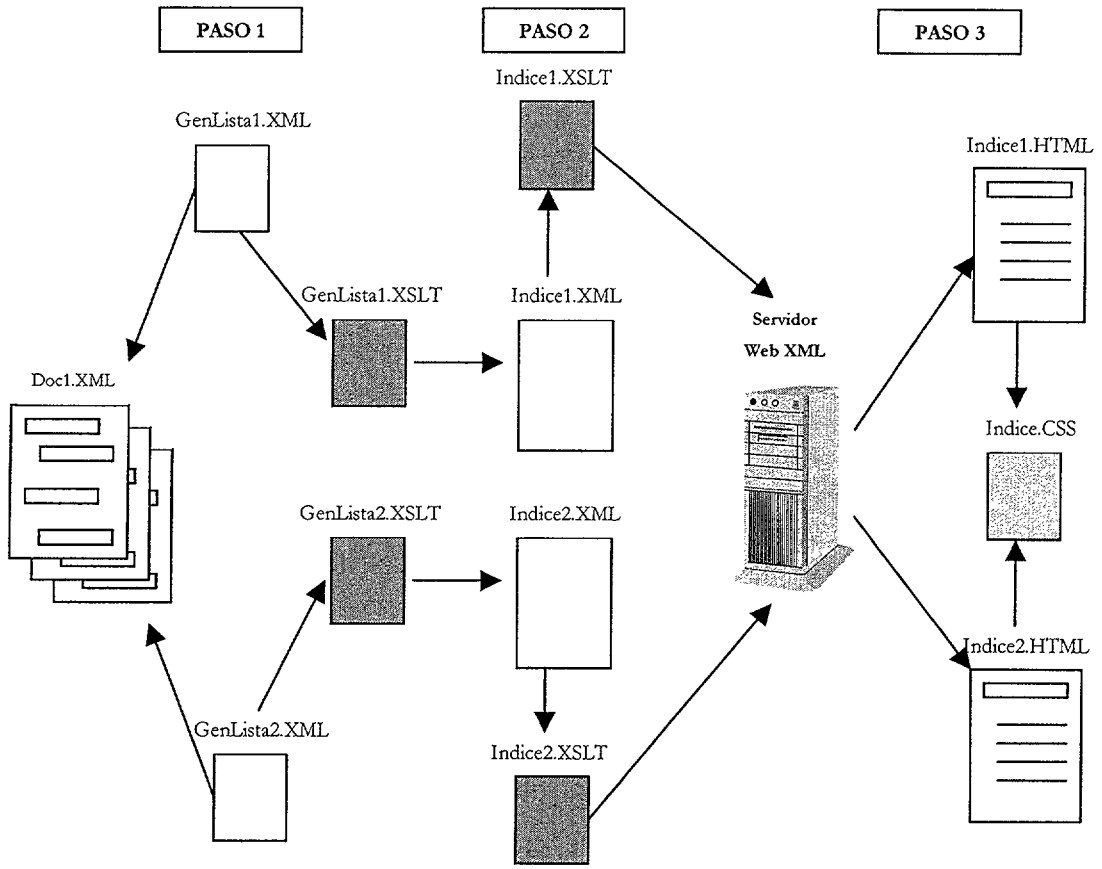


Figura IV.29: Esquema general para la generación de los diversos índices HTML de acceso a los documentos.

En la anterior figura se muestra el proceso que se ha de llevar a cabo para la confección de dos índices en formato XML y su posterior presentación al usuario en formato HTML. Es evidente que este mismo modo de proceder es válido para el resto de índices que se desearan crear y, consiguientemente, presentar en la WWW. Para no extendernos en exceso se comentarán aquí los diversos pasos seguidos para la realización de uno de los índices de acceso a los documentos XML del sistema, el primero y más común: el índice cronológico. La dinámica a seguir para el resto de índices que se desearan crear (por Salas, por descriptores de materias, por la decisión adoptada, por el tipo de recurso planteado, etc.) es prácticamente idéntica a la aquí descrita.

Para la creación del índice cronológico en el **Paso 1** es necesario generar nuevamente un documento XML que sirva de base o soporte al documento XSLT que ha de extraer los datos que posteriormente van a ser empleados para la presentación final de dicho índice. Este documento XML, con contenido mínimo al igual que en anteriores ocasiones, fue denominado **baseXMLlista1.xml**. En él se inserta la instrucción de procesamiento para llamar al documento XSLT que habrá de extraer los datos de cada uno de los documentos XML de las Sentencias del TC, así como un elemento vacío aleatorio, tal y como se muestra a continuación (contenido completo de dicho documento XML de soporte):

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- edited with XML Spy v3.5 (http://www.xmlspy.com) by
(Bonifacio Martín Galán, J. Tomás Nogales Flores) -->
<!-- Este documento sólo sirve de base o soporte para la
generación del documento XML (indiceXML1.xml) que contendrá
los datos necesarios para el desarrollo del índice
cronológico -->
<?xml-stylesheet type="text/xsl"
href="../../docsXSL/generarXML/genXMLlista1.xsl"?>
<Entrada/>
```

El verdadero peso recae nuevamente en el documento XSLT que ha de extraer la información de interés de cada documento XML. Este nuevo documento XSLT, denominado de **genXMLlista1.xml**, se encuentra disponible en el Apéndice 17 para su consulta.

La tecnología XSLT tiene una serie de limitaciones, como ya se comentó, y aquí podemos encontrar una de las principales , lo que nos ha obligado a crear un documento XML por cada uno de los tipos de índices que deseamos generar: si bien podemos realizar una llamada a un determinado documento XML desde el documento XSLT para la extracción de aquellos datos que son de nuestro interés, no es posible realizar esta función de forma genérica para un conjunto de documentos. La variable XLST *xsl:variable* aplicada al tratamiento de documentos externos no permite hacer indicaciones de forma global, para indagar y tratar, por ejemplo, todos los documentos XML que se puedan encontrar dentro de un determinado directorio. Esta gran limitación obliga a confeccionar un nuevo documento XML por cada índice que se desee crear pero, además, impide que dentro de

cada uno de ellos se incluyan instrucciones que puedan afectar de forma global a un conjunto de documentos; cada instrucción sólo puede ser relativa al contenido de un único documento. Ello obliga a crear una variable distinta para cada documento XML contemplado y repetir las mismas instrucciones para cada una de estas variables. Por ejemplo, para la Sentencia del TC que hemos venido utilizando de base de esta explicación (STC 1/2000), la variable definida dentro de este documento XSLT sería la siguiente:

```
<xsl:variable name="stc00-01" select="document
('../../SentenciasTC/stc2000-001.xml')"/>
```

Para la extracción de los datos de este documento y la generación de un nuevo documento XML para el índice cronológico, se crea la estructura deseada para cada documento y se establecen las diversas instrucciones para la traslación de datos. Dado que dentro de cada ítem del índice cronológico la información que se desea mostrar es mínima (título de la sentencia, la Sala que adoptó el fallo, el tipo y número de recurso, el resumen de dicho recurso, el resumen del fallo y otros datos de interés para la ordenación de dichos ítem y el acceso hipertextual tanto al documento abreviado como al completo), la estructura y los elementos extraídos quedarían de la siguiente forma para cada uno de los documentos XML tratados (dentro del elemento de documento denominado *DocumentoGlobal*):

```
 <STClista>
 <Titulo><xsl:value-of select="$stc00-
01/SentenciaTC/Cabecera/TitDocumento/@entrada"/></Titulo>
 <FechaAprobacion><xsl:value-of select="$stc00-
01/SentenciaTC/Fallo/FechaAprobacion/@fecha"/></FechaAprobac
ion>
 <FicheroBreve><xsl:value-of select="$stc00-
01/SentenciaTC/Cabecera/FicheroBreve/@entrada"/></FicheroBre
ve>
 <FicheroCompleto><xsl:value-of select="$stc00-
01/SentenciaTC/Cabecera/NomFichero/@entrada"/>
</FicheroCompleto>
 <Sala><xsl:value-of select="$stc00-
01/SentenciaTC/Preambulo/Composicion/Sala/@codigo"/></Sala>
 <Recurso>
      <xsl:attribute name="codigo">
```

```
      <xsl:value-of select="$stc00-
01/SentenciaTC/Preambulo/Introduccion/Recurso/@codigo"/>
      </xsl:attribute>
      <xsl:attribute name="numero">
      <xsl:value-of select="$stc00-
01/SentenciaTC/Preambulo/Introduccion/Recurso"/>
      </xsl:attribute>
  </Recurso>
 <ResumenRecurso><xsl:value-of select="$stc00-
01/SentenciaTC/Cabecera/Extracto/@entrada"/>
</ResumenRecurso>
 <ResumenFallo><xsl:value-of select="$stc00-
01/SentenciaTC/Cabecera/ResumenFallo/@entrada"/>
</ResumenFallo>
 </STClista>
```

De este modo, el procesador XSLT se encarga de extraer los datos reflejados en cada una de las instrucciones del documento *stc2000-001.xml*, los inserta dentro de los nuevos elementos establecidos y con todo ello genera un nuevo documento XML, que nos servirá para la presentación del índice en formato HTML.

El anterior código expuesto, volvemos a insistir, debe ser repetido para todos y cada uno de los documentos que han de ser procesados e incluidos en el índice. Lamentablemente, dicha inclusión no se puede realizar de forma automática al ser la tecnología XSLT incapaz de tratar de forma global todo el conjunto de documentos que se encuentran ubicados dentro de un determinado directorio. Así, y de forma poco ortodoxa, se debe copiar y pegar este bloque definido para cada nuevo documento XML y cambiar la referencia a la variable por la propia de ese documento. No es una tarea compleja pero, evidentemente, no es la solución ideal. Habrá que esperar a nuevas versiones del estándar XSLT que permitan dar solución a este grave problema o, en su defecto, incluir dentro de estos documentos XSLT sentencias de programación (*scripts* de Java, habitualmente) que permitan realizar esta función, aspecto éste no contemplado en el desarrollo práctico debido a su gran complejidad.

Con la activación de procesador XSLT contenido en el editor *XML Spy* comenzamos el **Paso 2** de este proceso. De este modo se genera, como decíamos, el documento XML que contendrá los datos de interés de cada uno de los documentos XML de partida y que

servirá para que con posterioridad podamos presentar al usuario el índice de estos documentos según una ordenación cronológica. Este documento se almacena físicamente dentro de su carpeta correspondiente (*Indices*) y se le asigna el nombre de **indiceXML1.xml**. Este documento XML se encuentra disponible a texto completo para su consulta en el Apéndice 18 de esta tesis doctoral.

Dentro de este fichero se encuentran ordenados según la nueva estructura definida todos los documentos XML de las Sentencias del TC que conformarán cada uno de los ítems del índice cronológico. De igual modo, dentro de este fichero se encuentra la instrucción de procesamiento que llama al documento XSLT encargado de ordenar de forma cronológica estos ítem y de presentarlos al usuario en formato HTML. Este nuevo documento XSLT que ha de generar el índice cronológico en formato HTML toma el nombre de **indiceXML1.xsl** y se encuentra disponible para su consulta en el apéndice 19 de esta tesis.

Este fichero XSLT, ciertamente extenso, contempla el marco HTML en el cual se van a insertar los datos procedentes de cada uno de los bloques de las Sentencias reflejadas en el documento *indiceXML1.xml*. Para ello, se estable el formato global de la estructura del cuerpo del documento HTML a generar, así como las instrucciones correspondientes para la selección de los datos que corresponden a cada uno de los años en los que se han emitido Sentencias por parte de este Tribunal (desde 1981 hasta el actual 2001). De igual modo, se establece que los ítem que correspondan a cada uno de esos años sean ordenados dentro de cada año de forma ascendente (desde el primero aparecido en dicho año hasta el último). Para llevar a cabo esta labor, se ha hecho uso, entre otras cosas, de dos instrucciones XSLT de gran interés: el establecimiento de condiciones o criterios para la selección de los ítems a través del elemento *xsl: if*, donde su atributo *test* establece que para cada año se deben seleccionar los ítems en los que el contenido del elemento *FechaAprobacion* comience con los cuatro dígitos propios de dicho año; la ordenación de los mismos se establece a través del elemento *xsl:sort*. Todo ello queda reflejado en el siguiente ejemplo extraído de dicho documento XSLT y correspondiente al tratamiento para el año 2000:

```
<h3>2000</h3>
 <br/>
 <xsl:for-each select="STClista">
 <xsl:sort select="FechaAprobacion" order="ascending"/>
 <xsl:if test="starts-with (FechaAprobacion,'2000')">
 <table width="100%">
 <tr>
 <td class="titulo"><xsl:value-of select="Titulo"/></td>
 <td class="fichero">[ <a
href="../SentenciasTC/SentenciasTCbreves/{FicheroBreve}">Fic
hero Breve</a> ]</td>
 <td class="fichero">[ <a
href="../SentenciasTC/{FicheroCompleto}">Fichero
Completo</a> ]</td>
 </tr>
 </table>
 <p class="resumen"><span><xsl:value-of
select="Sala"/></span>. <span><xsl:value-of
select="Recurso/@codigo"/></span> nº<xsl:value-of
select="Recurso/@numero"/>. <xsl:value-of
select="ResumenRecurso"/></p>
 <p class="resumen"><xsl:value-of
select="ResumenFallo"/></p>
 <br/>
 </xsl:if>
 </xsl:for-each>
```

Finalmente, con la creación de las hojas de estilo CSS para la presentación del documento HTML que se ha de generar de forma virtual, este índice cronológico está listo para ser presentado ante el usuario. Nuevamente, se definieron dos hojas de estilo CSS, en su nivel 1, tanto para la presentación en pantalla, **SentenciaTCindice.css**, así como para su salida por impresora, **SentenciaTCindiceb.css**, que se encuentran disponibles para su consulta en el Apéndice 20, que serán igualmente utilizados para el resto de índices en formato HTML que se deseen crear y presentar al usuario.

La presentación que del índice cronológico se realiza en formato HTML queda reflejada en la siguiente figura, utilizando el navegador IExplorer 6, si bien ésta similar en cualquier otro navegador web:

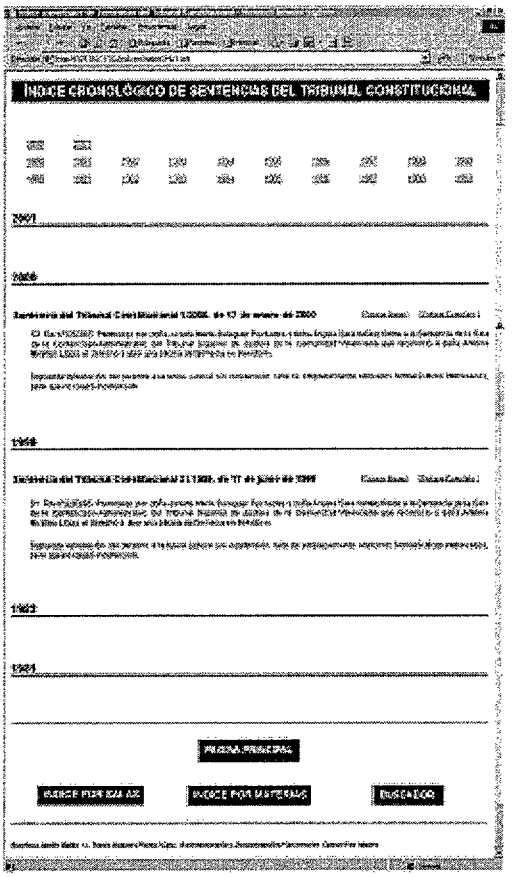


Figura IV.30: Presentación del índice cronológico en formato HTML en el navegador IExplorer 6.

En esta figura se puede observar que dentro de cada entrada del índice cronológico existen junto al título de las Sentencias del TC dos enlaces hipertextuales que dan acceso al documento abreviado y al documento completo de la Sentencia. De igual forma, en las partes superior e inferior de este documento HTML, se ofrecen otros accesos hipertextuales para la navegación por el sistema, tanto dentro del propio documento (acceso directo al año de interés) como a otros , entre ello la página principal, el motor de búsqueda y los otros dos índices planteados en un principio para este desarrollo práctico: el índice por Salas del TC y el índice por materias.

IV.2.4. INCORPORACIÓN DE OTRAS TECNOLOGÍAS XML COMPLEMENTARIAS

IV.2.4.1. HIPERENLACES AVANZADOS: XLINK

En el apartado III.2.3 relativo a los estándares XML acompañantes, se introdujo la tecnología XLink para la asignación de capacidades hipertextuales extendidas a aquellos elementos de un documento XML que así lo requiriesen. En ese momento, así como en otros apartados de esta tesis doctoral, se ha venido comentando que debido a la complejidad tecnológica que conlleva este potente modelo su implantación por parte de los actuales navegadores web era muy escasa, por no decir nula. Tan sólo los navegador Netscape Navigator, desde su versión 6, y el navegador Mozilla, desde su versión 0.9, son capaces de hacer una interpretación mínima de este estándar XML asociado: de los dos modelos generales de hiperenlaces definidos, los simples y los extendidos, estos navegadores ofrecen cobertura únicamente a los del primer tipo.

Los enlaces de tipo sencillo de XLink no se diferencian prácticamente nada de los enlaces habituales del entorno de HTML, siendo, por tanto, unidireccionales.

Esta falta de soporte real a XLink por parte de los actuales navegadores nos hizo plantearnos muy seriamente su inclusión dentro del sistema desarrollado. Además de no obtener ninguna ventaja añadida con respecto a los enlaces HTML que iban a ser empleados, nos encontrábamos con las trabas de tener que incluir un *namespace* dentro del modelo de DTD desarrollado para las Sentencias del TC que, como sabemos, no está capacitado para soportar de forma adecuada este tipo de integraciones.

El problema lo podríamos solventar con el esquema XML donde es perfectamente posible y válido hacer uso de elementos procedentes de otros lenguajes XML específicos, como es el caso de XLink. Esta idea hubiera sido razonable y a buen seguro hubiese prosperado si se hubiese desarrollado el modelo 1 planteado aquí, de presentación directa

en el navegador web del usuario de las Sentencias del TC en formato XML. En este caso, sólo los navegadores Netscape Navigator 6 y Mozilla 0.9 hubiesen sido capaces de señalar los correspondientes enlaces XLink simples, con la consiguiente pérdida de información y capacidad hipertextual en el resto de navegadores.

En el segundo modelo planteado, y desarrollado finalmente en nuestro sistema, nos encontraríamos con el problema de qué hacer finalmente cuando los documentos de las Sentencias del TC tuvieran que ser transformadas al formato HTML para su visualización. En este caso, todo lo definido para XLink (incluso si se tuviera la capacidad de crear enlaces extendidos y, por tanto, bidireccionales) hubiese servido de poco; dichos enlaces deberían ser transformados al formato de enlace sencillo propio del lenguaje HTML.

Finalmente otro tipo de consideraciones nos hicieron abandonar la idea de utilizar la tecnología XLink dentro de nuestro desarrollo práctico, especialmente la existencia de otra tesis doctoral en marcha dentro del Departamento de Biblioteconomía y Documentación de la Universidad Carlos III de Madrid, realizada por la profesora Carmen Arellano Pardo y dirigida, al igual que la presente, por el profesor Dr. J. Tomás Nogales Flores, en donde de forma profunda se analiza y emplea dicha tecnología para el tratamiento de los documentos legislativos.

En cualquier caso, y dado que nuestro interés de partida había sido el estudio y aplicación práctica de un gran número de tecnologías XML a los documentos jurisprudenciales emanados del Tribunal Constitucional español, se realizaron en las primeras fases de este desarrollo práctico una serie de pruebas con la tecnología XLink. Así, se definieron dentro de la DTD primitiva aquellos elementos característicos de XLink que podrían ser empleados en su modo sencillo dentro de aquellos elementos propios que deberían tener un comportamiento hipertextual. Para poder llevar a cabo esto era necesario en primer lugar incluir el *namespace* de XLink dentro del elemento raíz o de documento (*SentenciaTC*) de la DTD definida, quedando, por tanto, del siguiente modo:

```
<!ELEMENT SentenciaTC (Cabecera, Preambulo, Antecedentes,
FundamentosJuridicos, Fallo, Votos?)>
<!ATTLIST SentenciaTC
 xmlns:xlink CDATA #FIXED "http://www.w3.org/1999/xlink"
>
```

Tomando como ejemplo al elemento *RefSentencia* definido en nuestra DTD, se le incluyeron como atributos propios los diversos componentes definidos en la Recomendación XLink para el establecimiento de una capacidad hipertextual simple, quedando la declaración de su modelo de contenido como sigue:

```
<!ELEMENT RefNorma (#PCDATA)>
  <!ATTLIST RefNorma
     id ID #REQUIRED
     xlink:type (simple) #FIXED 'simple'
     xlink:href CDATA #REQUIRED
     xlink:show (new | replace | embed | other | none)
    #IMPLIED
     xlink:actuate (onLoad | onRequest | other | none)
   #IMPLIED>
```

De igual forma se procedió con el resto de elementos de la DTD que debían llevar asociada esta capacidad hipertextual; esto es, todos los que incluyeran referencias a normas y sentencias.

En el paso siguiente, habría que modificar el marcado del documento XML de la Sentencia e incluir, dentro de los elementos modificados en la DTD, los atributos correspondientes a la especificación XLink. Por ejemplo, en el caso de una cita a una Sentencia anterior del propio TC, ésta quedaría marcada del siguiente modo:

```
[...] Doctrina que ha sufrido diversas matizaciones en
cuanto ese deber de emplazamiento personal ha de
corresponderse, no sólo con una actitud diligente del
ciudadano para mostrase parte en el proceso, sino también
con la efectiva indefensión que se le pueda causar (STC
<RefSentencia id="STC133-86" xlink:href="../1986/stc133-
1986.xml" xlink:show="new"
xlink:actuate="onRequest">133/1986</RefSentencia>), pues la
protección ilimitada [...]
```

Como ya se ha señalado, la visualización de este documento XML en el cual se hacía uso de la tecnología XLink tan sólo produjo los resultados deseados en los anteriormente mencionados navegadores Web.

Finalmente, y para concluir este proceso de aplicación de la tecnología XLink, se procedió a integrarla en el modelo 2, es decir, a emplear un documento XSLT para realizar la transformación del documento XML al formato HTML. Convirtiendo los enlaces XLink en elementos *A* del lenguaje HTML, de manera que para el elemento que hemos usado como ejemplo, quedaría de la siguiente forma:

```
<xsl:template match="RefSentencia">
   <a href="{@xlink:href}" class="RefSentencia">
 <xsl:apply-templates/>
   </a>
```

Por todo ello, y con las demás razones expuestas en este punto, se desechó el empleo de la tecnología XLink, dejando su análisis pormenorizado y su implementación práctica a la mencionada tesis doctoral ya en marcha dentro de nuestro Departamento. A ésta remitimos, por tanto, para una mayor profundidad sobre todos estos aspectos de tanto interés en el entorno de producción documental mediante el uso de tecnologías XML.

IV.2.4.2. ASIGNACIÓN DE METADATOS: RDF

El tema de la asignación de metadatos a los documentos electrónicos existentes en el espacio de la WWW se explicó en profundidad en el capítulo III.5 de la presente tesis doctoral. En él se analizaban los diversos modelos existentes para llevar a cabo esta labor, así como las ventajas aportadas por la tecnología XML de RDF (*Resource Description Framework*). De igual modo, se expusieron las actuales limitaciones de este modelo para la asignación de metadatos a los objetos electrónicos de la Web, entre los que fundamentalmente predominaban la falta de cobertura proporcionada por los habituales servicios de búsqueda y recuperación de la información en este espacio y, de igual forma, la prácticamente nula integración de esta tecnología en los actuales navegadores web, ya que tan sólo los navegadores Netscape Navigator 6 y Mozilla 0.9 ofrecían un mínimo de interoperabilidad.

De igual modo se expuso en el apartado III.5.2 de dicho capítulo, las diversas corrientes de opinión surgidas a lo largo de estos años con respecto a la idoneidad de los distintos modelos para la definición semántica de los contenidos existentes en los documentos electrónicos, que iban desde el empleo de tecnologías como los *Topic Maps* (mapas de materias) aplicadas al entorno tecnológico de XML, hasta el desarrollo de esquemas y vocabularios específicos a través de la propuesta realizada por el esquema RDF, tanto de carácter generalista, como es el vocabulario del *Dublin Core* (DC), como de carácter más sectorial y puntual. En concreto, se analizaron y compararon el modelo de desarrollo de esquemas propuestos por RDF y el modelo de esquema propio de XML, estableciendo las ventajas e inconvenientes de ambos modelos. Se concluía dicho apartado exponiendo la creencia particular en que la fusión de ambas tecnologías podría representar una esperanza de futuro más que razonable para el establecimiento de los llamados metadatos dentro de los documentos XML, pues de este modo se combinaban las capacidades que los esquemas RDF tienen para establecer una definición semántica universal de los contenidos existentes en dichos documentos con la potencialidad del modelo de esquema XML del W3C para la definición y establecimiento de tipos de datos que fueran requeridos dentro de los documentos XML. Todo ello aplicado al entorno de la WWW podría llegar a producir la

tan ansiada idea de una *Web Semántica*, planteada hace unos años ya por T. Berners-Lee. Todos estos planteamientos serán expuestos más adelante en este apartado.

Si en un entorno de producción y gestión de documento XML cerrado, como puede ser el caso de las intranets corporativas, el desarrollo y uso de un esquema XML potente y bien definido podría ser más que suficiente para la posterior búsqueda y localización de la información deseada, en el mega-espacio virtual de la WWW el empleo de la tecnología RDF se hace absolutamente fundamental. De este modo, dado que el modelo práctico aquí desarrollado ha de ser expuesto en ambos espacios electrónicos, resultaba más que conveniente realizar una aproximación, aunque fuese mínima, a estos planteamientos de un modo práctico.

Antes de empezar con la explicación del uso que se hizo de la tecnología RDF en nuestro desarrollo práctico, es necesario comentar que igualmente existe otra tesis doctoral en marcha dentro de nuestro Departamento que analiza con mucha más profundidad y precisión todas estas cuestiones relativas al modelo RDF. Se trata de la tesis doctoral que está realizando la profesora Eva Mª Méndez Rodríguez y dirigida por el catedrático D. José Antonio Moreiro, y a la cual remitimos para una más amplia información sobre lo tratado en este punto.

Al igual que sucedía en el caso de la tecnología XLink, la inclusión de la tecnología RDF en nuestro modelo no iba a aportar en la práctica nada especialmente relevante, por lo menos en el nivel actual de aceptación de las diversas tecnologías XML existentes; y nos explicamos.

En primer lugar, la inexistencia de un verdadero vocabulario de carácter jurídico-jurisprudencial elaborado o adoptado por alguna institución u organización supranacional y suficientemente aceptado para el tratamiento de documentos XML con dichos contenidos, nos hizo establecer nuestra primera aproximación con el vocabulario más extendido y aceptado universalmente: el *Dublin Core*. Partiendo del conjunto de elementos de metadatos definidos por *Dublin Core* (DCMES), algunas instituciones de carácter jurídico habían

realizado una aproximación a los mismos de una forma práctica, añadiendo algunos nuevos para una mejor adecuación del modelo a los contenidos de los documentos jurídicos. Tal es el caso, como así se expuso en el apartado III.6.2.2, de la *Law & Justice Foundation of New South Wales* y otras instituciones australianas, en donde para el desarrollo del *Online Legal Access Project* (OLAP) se ha hecho uso del modelo del *Dublin Core* para la descripción de recursos electrónicos de carácter jurídico, especialmente documentos HTML, que permitan una recuperación eficaz de los mismos en Internet por parte de los ciudadanos de este estado. También es importante destacar a la sección germana de la organización Legal XML, la denominada LeXML, dado el especial interés que han demostrado en la tecnología RDF aplicada a la documentación jurídica y el apoyo que le han prestado, aunque sus trabajos no hayan ido más allá de las meras especulaciones teóricas.

Por lo demás, y tal y como se vino mostrando a lo largo del Capítulo III.6 de esta tesis doctoral, la mayoría de las iniciativas emprendidas en el mundo con respecto al uso de las tecnologías XML para el tratamiento de la información electrónica de carácter jurídico se han centrado en el desarrollo de definiciones de tipos documentales (DTD), más o menos aceptadas, que resultasen de interés para la estructuración de los contenidos en los intercambios de información electrónica que se producen en y entre los Tribunales de justicia y las partes implicadas en un proceso judicial (en el caso norteamericano, para el desarrollo de los denominados *Tribunales Electrónicos de Justicia*).

En los planteamientos realizados para llevar a cabo nuestro desarrollo práctico se decidió que tan sólo se haría uso de los elementos generales del *Dublin Core* pues nuestro fin último no iba encaminado al estudio, análisis y problemática del complejo tema de los metadatos en los documentos XML, aspecto éste, como ya se ha comentado, tratado con detenimiento en otras tesis doctorales que se están realizando en estos momentos dentro del Departamento de Biblioteconomía y Documentación de esta Universidad.

Centrándonos en las experiencias llevadas a cabo en este desarrollo práctico para la inclusión de información semántica dentro de los documentos XML tratados, se barajaron dos posibilidades que nos parecieron realmente interesantes. Desde un primer momento se descartó enlazar cada documento XML con un fichero RDF externo para la descripción

semántica de parte de los elementos de marcado definidos en nuestro modelo debido, en parte, al gran trabajo que ello supondría, máxime cuanto existen otros mecanismos más interesantes para ello: la asignación de dicho componente semántico a los elementos dentro del propio esquema XML desarrollado.

Para ello, el esquema debería ahora hacer uso directo de ciertos elementos establecidos por el DC a través de la inclusión de su *namespace* y la importación de dichos elementos, establecido todo ello del modo siguiente:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2000/10/XMLSchema"
xmlns:dc="http://purl.org/dc/elements/1.1/">

<xsd:import namespace="http://purl.org/dc/elements/1.1/"/>
[...]
```

Lo siguiente hubiese sido hacer referencia directa a los elementos del DC dentro de la declaración de nuestros elementos en el esquema. El gran problema estaría entonces en el hecho que dichos elementos dejarían de tomar el nombre que se había elegido para ellos a favor de los nombres de los elementos del DC. Por ejemplo, para señalar el título de los documentos XML se había establecido el nombre de elemento *TitDocumento*, por lo que éste debería ser sustituido en este modelo por el elemento *title* del modelo del DC, quedando, por tanto, del siguiente modo:

```
<xsd:element ref="dc:title">
   <xsd:complexType>
 <xsd:attribute name="entrada" type="xsd:string"
use="required"/>
   </xsd:complexType>
</xsd:element>
```

De este modo, todos aquellos elementos de nuestro esquema que tuviesen un paralelismo claro con los elementos definidos por el DC (*creator*, *subject*, *description*, *publisher*, *date*, etc.) deberían ser sustituidos por éstos. Con ello ganábamos en normalización de la descripción semántica al ajustar el vocabulario de nuestro lenguaje de marcado al vocabulario del DC pero, lógicamente, perdíamos el control de poder establecer los nombres de los elementos

que deseásemos para marcar las Sentencias del TC. Por ello, este mecanismo para la asignación de metadatos fue desechado.

El siguiente planteamiento, ya anunciado y planteado en el apartado III.5.2 de esta tesis, toma como idea base lo expuesto por Jane Hunter y Carl Lagoze al respecto de este tema en la comunicación presentada al décimo congreso internacional de la WWW[29]. De las ideas ahí expuestas, estos autores resaltan el interés creciente que supone la combinación del modelo de esquema de XML con el modelo de esquema RDF. Si el primero resulta ser un potente mecanismo para definir estructuras documentales, frecuencias de aparición de los elementos, así como tipos de datos y sus restricciones, el segundo proporciona una base potente para el establecimiento de definiciones semánticas a través de la definición de tipos de jerarquías, así como para las relaciones que se dan entre las clases y las propiedades. La idea es sencilla pero eficaz: si para nuestro esquema el título de todo documento XML tratado debe estar contenido dentro del elemento *TitDocumento*, tan sólo resta hacer una equivalencia de éste con el elemento *title* del DC (a modo de diccionario bilingüe). Esta equivalencia se realiza dentro del elemento *xsd:annotation*, para establecer anotaciones dentro de los esquemas XML, y de forma más precisa dentro del subelemento *xsd:appinfo*, destinado a suministrar información de interés a la aplicación informática que ha de procesar el esquema XML.

Lamentablemente, a pesar de la importancia que esta forma de operar tendría para el control semántico de los documentos XML que vayan a poblar en un futuro cercano la WWW, no existen aún programas informáticos que sean capaces de procesar de forma conjunta ambos modelos de esquemas. En cualquier caso este planteamiento, al menos en su plano teórico, resulta más que interesante.

[29] Jane Hunter, Carl Lagoze. *Combining RDF and XML Schemas to Enhance Interoperability Between Metadata Application Profiles* [documento HTML]. Queensland: DSTC, University of Queensland, November 2000. Disponible en http://archive.dstc.edu.au/RDU/staff/jane-hunter/www10/paper.html (consultado el 5 de junio de 2001).

Siguiendo estas interesantes ideas, se procedió a integrar dentro de nuestro esquema XML el modelo de esquema RDF con la inclusión, nuevamente, del vocabulario del DC en todo este conjunto. Así, nuestro esquema debería hacer referencia a los distintos *namespaces* que entran en juego, del modo siguiente:

```
<xsd:schema
 xmlns:xsd="http://www.w3.org/2000/10/XMLSchema"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
>
```

A partir de ahí, se pueden incluir definiciones semánticas (clases o propiedades, con sus distintas subclases o subpropiedades) propias del modelo de esquema RDF dentro de nuestro esquema XML y haciendo mención a un dominio o vocabulario RDF concreto, en nuestro caso el del *Dublin Core*. Por ejemplo, el elemento *Autor* definido en nuestro esquema XML quedaría, siguiendo este planteamiento, del siguiente modo:

```
<xsd:element name="Autor">
  <xsd:complexType>
    <xsd:annotation>
     <xsd:appinfo>
      <rdf:Property ID="Autor">
       <rdfs:subPropertyOf
   rdf:resource="http://purl.org/dc/elements/1.1/#Creator"/>
      </rdf:Property>
     </xsd:appinfo>
    </xsd:annotation>
   <xsd:attribute name="entrada" type="xsd:string"
use="fixed" value="Tribunal Constitucional (España)"/>
 </xsd:complexType>
</xsd:element>
```

De este modo, establecemos una equivalencia entre el atributo *entrada* de nuestro elemento *Autor* y el elemento *Creator* del DC, por lo que si existiera un programa analizador en la WWW de este tipo de documentos, podría perfectamente establecer el paralelismo semántico. Los documentos marcados con este esquema podrían ser por tanto recuperados con un sistema de búsqueda de información en Internet sobre documentos XML que hiciesen uso de este paralelismo con el vocabulario de mayor aceptación, el *Dublin Core*.

De igual modo se puede proceder con el resto de elementos y/o atributos definidos en nuestro esquema que para los que interese establecer similares equivalencias con el vocabulario del Dublin Core, como pueden ser los casos del elemento *Descriptor* (equivalente al *subject* del DC), el elemento *FechaAprobacion* (equivalente al *date* del DC) y otros tantos.

En cualquier caso, como queda reflejado en el Apéndice 9 de esta tesis doctoral, en el cual se contiene el texto íntegro del esquema SentenciaTC.xsd aquí desarrollado, no se han incluido todas estas cuestiones debido al estado de experimentalidad aún en nuestros días de este tipo de desarrollo y, especialmente, a su falta de soporte por parte de los actuales programas que trabajan con XML.

Confiamos que en futuro próximo la asignación de metadatos RDF incluidos en esquemas XML sean toda una realidad. Dejamos, por tanto, su planteamiento teórico y el futuro desarrollo práctico a aquellas personas que dentro de nuestro Departamento están apostando y trabajando de forma decidida en la consecución de la llamada *Web Semántica.*

IV.2.5. PUBLICACIÓN DEL PRODUCTO FINAL EN INTERNET

IV.2.5.1. APACHE/COCOON: SERVIDOR WEB DE DOCUMENTOS XML

Resulta evidente que para llevar a cabo la publicación en la *World Wide Web* de todo el conjunto de documentos electrónicos desarrollados en nuestro sistema es necesario disponer de un equipo informático y de un software especial instalado en él.

En el apartado IV.1.1.3 del anterior capítulo ya se expusieron las limitaciones tecnológicas que habíamos sufrido a lo largo del desarrollo práctico aquí presentado. Entre éstas estaban las relativas a la disponibilidad de un ordenador potente en el que almacenar los documentos XML elaborados, así como los diversos programas necesarios para su publicación en la red Internet. Debido a los problemas ahí reseñados, el sistema tuvo que ser montado en diversos ordenadores personales, con las limitaciones que ello conlleva.

En cualquier caso, no sin ciertos problemas, como iremos exponiendo a continuación, el sistema desarrollado tuvo que ser montado en un ordenador portátil con sistema operativo Windows Millenium, con suficiente memoria en disco y temporal para poder trabajar con ciertas garantías con todos los programas informáticos puestos en juego.

Cualquiera de los servidores web (o *httpd*) existentes en el mercado es capaz de suministrar a través de Internet documentos XML pues, como ya se ha exponiendo a lo largo de esta tesis doctoral, estos documentos están confeccionados en texto plano, y, por tanto, son fácilmente transmisibles a través del protocolo HTTP propio de este tipo de servidores. El problema de estos servidores reside en el hecho de no ser capaces de interpretar lo dispuesto en un documento XML que haga una llamada a su DTD o esquema XML asociado, por lo que estas partes no serán cargadas o enviadas (en el caso de

que dicha definición de tipo documental esté inserta en un fichero externo). De este modo, si dentro de los documentos existen declaraciones de entidades, gráficas o de texto, éstas no serán mostradas por el navegador web del usuario.

Estos problemas se solucionan en gran medida con un servidor web específico para el tratamiento de documentos XML. Entre la pléyade de productos de este tipo que existen en el mercado[30] se optó por hacer uso del servidor web Apache junto con su extensión Cocoon para el tratamiento de este tipo de documentos, debido a la más que demostrada calidad de sus productos y su gratuidad para fines no comerciales.

El servidor web **Apache**[31] es fruto del consorcio internacional *Apache Software Foundation* para el desarrollo de tecnologías web de forma cooperativa y abierta. Se trata de un excelente producto que, además de su gratuidad, resulta perfecto para la publicación de documentos HTML en el espacio de la WWW. El grupo de Tecnologías de la Información ya tenía experiencia con este software para ordenadores SUN con sistema operativo Solaris pues es el servidor instalado en dos de las máquinas del Departamento.

Esta organización internacional puso en marcha en noviembre de 1999 el denominado **Apache XML Project**[32] mediante el cual se hacía una apuesta firme y decidida por la promoción de los sistemas de información electrónica basados en las tecnologías XML, así como por el desarrollo práctico de diferentes estándares relacionados con aquellas. Este gran proyecto está dividido en una serie de subproyectos, cada uno de ellos tratando un

[30] Dentro del sitio web de James Tauber, dedicado a la descripción de software de interés en relación con la tecnología XML, existe un apartado dedicado específicamente a los sistemas de publicación de este tipo de documentos en el espacio Web, en la dirección http://www.xmlsoftware.com/publishing/

[31] La dirección del sitio web en donde se encuentra disponible toda la información oficial relativa a este proyecto y dicho producto es http://www.apache.org Dentro del mismo, recomendamos para una más amplia información sobre este producto la consulta del documento *Apache Server Frequently Asked Questions* [documento HTML]. Apache.org, rev. February 28, 2001. Disponible en http://httpd.apache.org/docs-2.0/misc/FAQ.html (consultado el 13 de marzo de 2001).

[32] Toda la información relativa a este importante proyecto se encuentra en la dirección http://xml.apache.org/

aspecto distinto de dichas tecnologías. Así, las principales partes de desarrollo de este proyecto son las siguientes:

- **Xerces**: se trata del procesador o *parser* específico de documentos XML, desarrollado en Java y en C++. Esta aplicación es la encargada de validar los documentos XML del sistema por lo que ofrece una completa cobertura al modelo de la DTD de XML y, de forma parcial, al modelo de Esquema XML (completa en su fase de borrador de trabajo). Para ello, este *parser* implementa dichos estándares XML del W3C, así como una perfecta cobertura para el estándar DOM (en sus niveles 1 y 2) y el no oficial SAX (en su versión 2).
- **Xalan**: se trata del procesador o *parser* de hojas de estilo XSLT, desarrollado igualmente en Java y en C++. Esta aplicación ofrece una completa implementación de los estándares del W3C para llevar a cabo la validación y transformación de los documentos XML, como son, principalmente, el estándar XSLT y el estándar XPath. Asimismo, Xalan utiliza el denominado BSF (*Bean Scripting Framework*) para la implementación de instrucciones escritas en Java o a través de extensiones de *script*, para el desarrollo de múltiples extensiones en la presentación del documento. De igual modo, dentro de este proyecto se está trabajando en la interconexión de sistemas de bases de datos a través de extensiones para SLQ/JDBC.
- **Cocoon**: aplicación de la cual hablaremos posteriormente, se trata del marco de trabajo de esta organización para el desarrollo de un sistema de publicación de documentos XML en la WWW.
- **FOP**: se trata de la aplicación encargada de asignar un formato de estilo orientado a la impresión de los documentos XML derivado del uso del estándar *XSL Formatting Objects*. Se trata de una aplicación desarrollada completamente en Java que analiza el árbol estructural de los documentos XML y los transforma en documentos PDF.
- **Xang**: aplicación orientada al desarrollo de páginas web dinámicas, de especial utilidad para aquella aplicaciones Web que han de integrar fuentes de datos dispares. Esta herramienta separa los datos, en su estructura lógica y en su presentación. Desarrollado mediante JavaScript.

- **SOAP** (*Simple Object Access Protocol*): se trata de un desarrollo del protocolo puesto en marcha por parte del W3C para el suministro de documentos XML en la WWW. Se trata del nuevo protocolo que se desea implementar en este espacio electrónico, basado en el lenguaje XML, y capaz de interpretar el contenido del mensaje y el modo de procesarlo, así como la forma en la que los servidores han de establecer las peticiones de información a otras máquinas y las respuestas que han de producir.

El programa **Cocoon**[33], fruto de las investigaciones llevadas a cabo sobre las tecnologías XML por parte de esta misma organización internacional, es igualmente un programa gratuito desarrollado íntegramente en Java. Como se indica en su propia página web de inicio, se basa en la idea de que los contenidos de los actuales documentos electrónicos, tanto los lógicos como los puramente estilísticos, han de ser creados de forma independiente por grupos distintos. De este modo, Cocoon extiende esta idea a la producción de documentos XML, separando la construcción lógica de estos documentos (creación de DTD/Esquema y generación de las distintas instancias de documento) de las características de presentación de los contenidos. Para esta última labor, como ya se ha ido exponiendo a lo largo de este último capítulo, Cocoon hace uso de las capacidades para la transformación de formatos de la tecnología XSLT. Con esta idea, se separa el contenido del documento electrónico, establecido en formato XML, de la presentación del mismo en el espacio de la Web, empleando para ello la transformación a cualquiera de los formatos electrónicos más habituales en dicho entorno: principalmente, en HTML, pero igualmente posible en PDF (a través del uso del XSL-FO), o incluso el lenguaje propio de la tecnología WAP de los actuales teléfonos móviles, el lenguaje WML (*Wireless Markup Language*).

Cocoon utiliza dos instrucciones de procesamiento propias, sin las cuales los documentos XML no pueden ser procesados y transformados.

[33] Toda la información oficial relativa a esta herramienta informática se encuentra disponible en la dirección http://xml.apache.org/cocoon/index.html Igualmente recomendamos la consulta del documento Robin Cover. *Apache XML Project* [documento HTML]. OASIS, rev. September 21, 1999. Disponible en http://www.oasis-open.org/cover/apacheXML.html (consultado el 12 de septiembre de 2000).

La primera de ella se ha de incluir dentro de cada documento XML para indicar al programa el proceso al que se le va a someter, en nuestro caso mediante el uso de XSLT. Es por este motivo por lo que todos los documentos XML confeccionados llevan en su prólogo la siguiente instrucción de procesamiento:

```
<?cocoon-process type="xslt"?>
```

La segunda instrucción de procesamiento es la que se incluye en los documentos XSLT para indicar el formato al cual se va a transformar el documento XML. En nuestro caso se ha empleado en todos los casos la transformación a HTML, por lo que la instrucción de procesamiento sería:

```
<?cocoon-format type="text/html"?>
```

Esta instrucción de procesamiento se puede generar de forma directa a través del propio lenguaje XSLT, con la siguiente sintaxis (la que se ha empleado en todos los casos):

```
<xsl:processing-instruction name="cocoon-
format">type="text/html"</xsl:processing-instruction>
```

Otra característica ciertamente interesante del software Cocoon es la posibilidad de incluir más de una instrucción de procesamiento de hoja de estilo XSL dentro de los documentos XML. Cada una de estas instrucciones de procesamiento puede apuntar a diversos ficheros XSLT, cada uno de ellos optimizado para un navegador o visualizador diferente, lo que se consigue a través del uso del atributo *media* dentro de cada una de esas instrucciones de procesamiento. Cocoon ofrece seis medios distintos de salida: para el navegador IExplorer (valor del atributo igual a *explorer*), para el navegador Opera (*opera*), para el navegador Lynx (*lynx*), para navegadores Java (*java*), para móviles WAP de Nokia y móviles UP (*wap*) y para el navegador Netscape Navigator/Mozilla (*netscape*).

En nuestro desarrollo práctico no se ha hecho uso de esta característica tan interesante de Cocoon pues el código HTML y las hojas de estilo CSS asociadas eran fácilmente interpretables de forma semejante por todos los navegadores utilizados.

En cuanto a la instalación de la aplicación Cocoon en el ordenador que ha servido de banco de pruebas, comentar brevemente que dicha tarea no ha estado exenta de cierta complejidad pues son numerosos los componentes que han de integrarse para la puesta en marcha de esta aplicación. Además, como ya se comentó, trabajar con un ordenador personal con un sistema operativo no adecuado para ejercer de servidor de información supone otra limitación bastante importante, como veremos a continuación. Las diferentes piezas que tuvieron que ser descargadas de la red e instaladas en el ordenador (todas ellas para WIN32), fueron las siguientes[34]:

- Instalación de una máquina virtual Java: Para ello fue necesario instalar el programa JDK (*Java Development Kit*) en una versión superior a la 1.2, posibilitando de este modo que nuestro sistema operativo sea capaz de interpretar y, por tanto, trabajar con aplicaciones escritas en Java. Del sitio web de la compañía Sun Microsystems se descargó e instaló la versión 1.3.1 de dicho desarrollo[35].
- Instalación del servidor Web Apache: De igual forma, se descargó, instaló y puso en marcha el servidor web Apache, antes referido. Afortunadamente, la instalación de esta aplicación no entrañó excesivas complicaciones, existiendo además abundante

[34] Es necesario puntualizar aquí que la instalación de todos los programas referidos no fue tarea sencilla dado que dentro del propio sitio web del desarrollo Cocoon se ofrece muy pocas instrucciones para la instalación de todas estas piezas para el sistema operativo de Windows 95/98/Millenium (dicha información se encuentra disponible en la dirección http://x,l.apache.org/cocoon/install.html). Por este motivo, se hizo necesario recurrir a otros documentos electrónicos existentes en la red que ofreciesen un mayor detalle en todos los pasos a seguir. En especial destacamos dos documentos HTML, cuyas direcciones son las siguientes: http://www.electropubs.com/cocooninstall.htm y http://www.drtebi.com/windows98/

[35] En concreto, la versión para Windows 95/98/Millenium se encuentra disponible en la dirección http://java.sun.com/j2se/1.3/download-windows.html

información al respecto en el sitio web de la organización Apache[36]. La versión que finalmente se instaló de este servidor http fue la 1.3.20 para Win32.

- Instalación de un *Servlet Engine*: Dado que Cocoon es un *servlet* de Java (programa que corre en un servidor web para el tratamiento y difusión de datos) es necesario instalar un motor de este tipo para su correcto funcionamiento. Cocoon puede funcionar con un amplio número de motores *servlet*, tanto los desarrollados al amparo del proyecto XML de Apache (Apache JServ y Apache Tomcat) como los desarrollados por otras compañías. Finalmente, se optó por la instalación de Apache JServ en su versión 1.1.2[37].
- Instalación de Cocoon: Finalmente, se procedió a descargar esta aplicación e instalarla de forma integrada con el resto de aplicaciones anteriores, no sin ciertas complicaciones dado el gran número de líneas de comandos que hubo que modificar en diversos ficheros de cada una de las instalaciones realizadas. Se descargó para su instalación la versión 1.8.2, aunque en la actualidad ya se encuentra en su versión 2. Dentro del paquete de instalación de Cocoon venían contenidos los distintos ficheros (en formato *jar* propio de los desarrollos Java) necesarios para el correcto funcionamiento de todas de las piezas integrantes en este desarrollo. Igualmente hubo que conectar todas ellas con el sistema general.

Tras una serie de intentos fallidos, el sistema de publicación en la WWW de documentos XML Apache/Cocoon pudo ser puesto en marcha, empezando con ello las primeras pruebas prácticas y el desarrollo completo del sistema XML expuesto en esta parte práctica de la presente tesis doctoral. Todo empezaba al activar el servidor Apache/Cocoon y comprobar que la respuesta de este servidor era la esperada, tal y como se puede observar en la siguiente imagen de activación del servicio Cocoon en nuestro ordenador:

[36] Para su instalación en sistemas Win32, recomendamos la consulta de la información contenida en la dirección http://httpd.apache.org/docs-2.0/plataform/windows.html

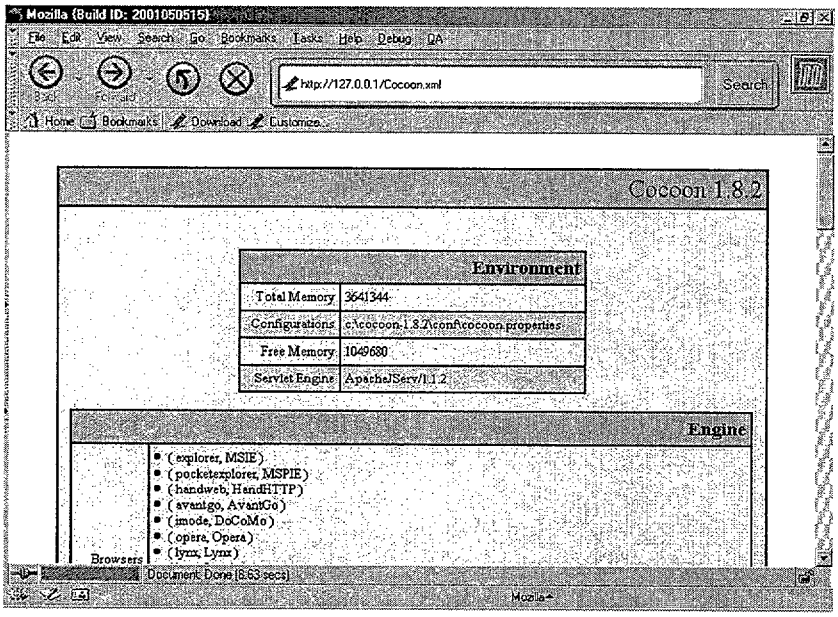


Figura IV.31: Pantalla de presentación y estado del servidor web Apache/Cocoon.

A partir de ese momento ya estábamos preparados para examinar de un modo directo todo lo que se había ido desarrollando durante estos últimos años, pudiendo de este modo contrastar de forma práctica y real todos los aciertos habidos y los errores cometidos en cada una de las etapas anteriores desarrolladas.

Ahora, tan sólo restaba ofrecer otra serie de complementos al sistema desarrollado tanto para la búsqueda y recuperación precisa de información como en la presentación global del sitio web XML desarrollado.

[37] La información existente en el sitio web de Apache para la instalación de esta aplicación se encuentra disponible en la dirección http://java.apache.org/jserv/install/howto.win32_install.html

IV.2.5.2. INDEXACIÓN DE LOS DOCUMENTOS E IMPLANTACIÓN DE UN MOTOR DE BÚSQUEDA

Antes de comenzar la exposición de este punto, es necesario volver a recordar que esta parte del desarrollo no ha sido implementado en la práctica en nuestro sistema fundamentalmente por la inexistencia de un software adecuado que se ajuste a los requisitos conceptuales que exigimos, y menos aún entre el gratuito o de dominio público. Aún así, se evaluaron una serie de herramientas, algunas de las cuales se aproximan a la idea general que tenemos sobre cómo deberían trabajar.

En la figura IV.4, así como en los comentarios vertidos en el apartado IV.1.2 del anterior capítulo, se expuso la doble visión que existe en la actualidad en el tratamiento y gestión de los documentos XML por parte de los actuales modelos existentes de bases de datos. En dicho apartado comentábamos que la orientación más habitual en nuestros días era la destinada al almacenamiento, gestión e intercambio de datos. Por ello, la mayor parte de los desarrollos que podemos encontrar hoy en día hacen uso de los tradicionales sistemas de bases de datos relacionales reconvertidos al entorno de producción de las tecnologías XML. Además, este entorno de producción viene desarrollado por empresas de todo tipo que hacen uso de las tecnologías XML para la comercialización de sus productos y servicios a través de la red Internet, por lo que es razonable suponer que el software que nos encontraremos para la puesta en marcha de dichos sistemas tenga, por lo general, un precio elevado.

De hecho, el campo de investigación y desarrollo de sistemas de gestión de bases de datos en el entorno de XML es hoy en día un negocio ciertamente lucrativo, como puede observarse en la amplia y variada oferta existente, y en la que toman partido las principales compañías informáticas del mundo[38].

[38] Recomendamos nuevamente visitar los sitios web de XMLSoftware y de XML.com, en los apartados correspondientes al software para la creación de bases de datos XML respectivamente, http://www.xmlsoftware.com/database/ y http://www.xml.com/pub/pt/15

Por otro lado, este tipo de programas informáticos ha de ser instalado, obviamente, en grandes ordenadores con sistemas operativos que sean capaces de ejercer de servidores (NT/2000, Sun Solaris, Red Hat Linux, IBM OS/390, etc.), lo que determina que por el momento y para la aplicación en nuestro desarrollo práctico, sea imposible hacer uso de estas aplicaciones de bases de datos pues ello excede las pretensiones de este apartado práctico.

De todas formas, como ya se comentó igualmente en el apartado antes mencionado, este modo de procesar la información contenida en los documentos XML no resulta adecuada para aquellos sistemas que tienen una orientación y vocación documental. Desde esta visión, y a diferencia de los sistemas de bases de datos tradicionales, no resulta necesario el desarrollo de las diversas tablas (con sus correspondientes campos) y el establecimiento de relaciones entre las mismas, sino que el conjunto de documentos XML tratados forma ya en sí mismo una base de datos (conocidas por el nombre de *native XML DataBase*) a la que lo único que le falta es la incorporación de un sistema de indexación automática de los contenidos y un motor de búsqueda capaz de recuperar dicha información. En los documentos XML marcados siguiendo esta visión, la información se encuentra ya inserta dentro de los campos, representados por los elementos empleados para marcar el contenido de los documentos. De este modo, si la estructura que define a los documentos tratados es más o menos rígida, un programa indexador debería ser capaz de generar los ficheros inversos correspondientes, asignando a cada palabra el elemento (o la línea hereditaria de elementos) o el atributo en el que se encuentra ubicada. Con posterioridad, el empleo de un motor de búsqueda potente debería ser capaz de facilitar los campos (elementos o atributos) existentes para realizar la búsqueda de información puntual, así como otras capacidades como la utilización de operadores booleanos, de proximidad o de adyacencia, etc.).

Este modo de proceder es el que tradicionalmente han venido siguiendo los buscadores de información en la WWW, como han sido los casos tan significativos de Lycos, Excite, AltaVista, Google, o, más recientemente, Wisenut.

Lamentablemente, no existen productos de este tipo que satisfagan de forma adecuada las expectativas que sobre ellos se tienen desde la visión documental en el tratamiento de documentos XML[39]. Muchos de ellos emplean ya la tecnología XML específica para la interrogación avanzada de información, a través del lenguaje XQL o, en su defecto, a través de XPath, pero su integración con el sistema y su interfaz de usuario para la búsqueda y localización de información dejan todavía mucho que desear. Otros, sin embargo, siendo en sí de gran interés por su potencia de indexación, su interfaz de usuario y capacidades de recuperación, emplean sus propios lenguajes de interrogación, no normativos.

En cualquier caso, se han analizado y probado una serie de programas de interés de este tipo, capaces de indexar los contenidos existentes en los documentos XML y de proporcionar los mecanismos necesarios para realizar potentes búsqueda de información. Entre ellos cabe destacar a programas tales como XYZfind[40], dtSearch[41], eXist[42], XML Query Engine[43], descartándose otros de gran interés, como TEXTML Server[44] o GoXML Search[45] por ser desarrollos comerciales que no ofrecen soporte para plataforma Win32 (o Java, en su defecto).

A continuación se exponen brevemente las características principales de los dos primeros productos citados.

---

[39] Recomendamos, nuevamente, la consulta de los apartados de los sitios web citados en los que se ofrece la relación de productos de mayor interés en la búsqueda y recuperación de información contenida en documentos XML. Así, véanse las direcciones http://www.xmlsoftware.com/search/ y http://www.xml.com/pub/pt/8

[40] http://www.xyzfind.com/

[41] http://www.dtsearch.com/

[42] http://exist.sourceforge.net/

[43] http://www.fatdog.com/

[44] http://www.ixiasoft.com/

[45] http://www.goxml.com/

- **XYZfind** es uno de los productos estrella en el campo de la indexación y recuperación de documentos XML bajo la filosofía antes expuesta. Así, este programa en su fase de indexación analiza los documentos XML existentes bajo un determinado directorio, infiere su estructura y de forma automática descompone el contenido de estos documentos de un único fichero de datos. En este proceso quedan establecidas todas las asociaciones que existen entre los elementos, atributos, contenidos, comentarios, espacios en blanco, nombres locales, etc., de cada uno de los documentos XML. Para la recuperación de la información almacenada este programa emplea un lenguaje propio de recuperación (el XYZQL) basado en gran medida en los desarrollos del W3C en esta materia, por lo que lo convierte en un lenguaje potente y robusto para la interrogación de información de la base de datos XML nativa: permite la búsqueda de palabras acotadas a los campos producidos en la anterior fase, soporta el uso de operadores booleanos, operadores de truncamiento, de proximidad, a texto completo, etc. De igual forma, permite establecer el modo en el que la información va a ser presentada, esto es, si se desea obtener el documento completo, bloques del mismo, elementos individuales o, simplemente, un listado de los resultados obtenidos. Finalmente, puede accederse a esta aplicación a través de un servidor web, utilizando para ello el propio protocolo http, una API de Java o el protocolo SOAP propio de la tecnología XML. A pesar de todas las ventajas de este producto, cuenta sin embargo con una serie de características que nos parecen inadecuadas, como son la imposibilidad de conocer por parte del usuario los campos por los que puede ser acotada la información, la utilización de un lenguaje de interrogación no normativo y de difícil aprendizaje por parte del usuario, una interfaz de interrogación bastante poco "amigable" y la presentación de los documentos XML recuperados con el texto y las marcas existentes, ignorando su asociación con una hoja de estilo. La instalación de esta aplicación se realiza en cualquiera de los sistemas operativos más extendidos con capacidad de ejercer de servidor (WindowsNT/2000, Solaris o Linux), lo que ha motivado que su análisis y evaluación haya tenido que efectuarse a través de la demostración que se realiza dentro de su sitio web y no, como hubiese sido nuestro deseo, instalando este programa de

forma local en el ordenador de trabajo. La siguiente captura de pantalla presenta una de esas pruebas realizadas:

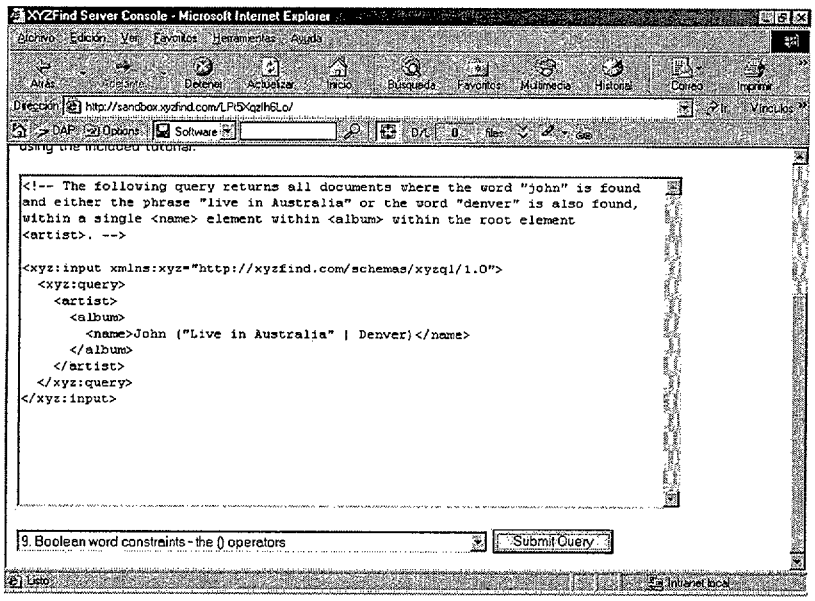


Figura IV.32: Pantalla de trabajo de la aplicación XYZfind.

- Otro de los productos analizados y considerados de mayor interés ha sido el programa **dtSearch**. Se trata de un producto dividido y comercializado en módulos independientes que posibilitan tanto la indexación de múltiples formatos de documentos electrónicos (HTML, XML, PDF, MS-Word, ficheros de bases de datos, mensajes de correo electrónico, etc.), la interrogación de los mismos a través de potentes mecanismos y la publicación de todo el conjunto documental del sistema tanto en el espacio de la WWW como en un soporte óptico (CD-ROM o DVD). Estos módulos son el Desktop, Network, Spider, Web, Publish y Text Retrieval Engine. El modo de proceder en la indexación es muy similar al del anterior producto analizado pero en este caso el proceso se realiza de forma más cómoda y sencilla a través de una interfaz de usuario en la que se pueden detallar aspectos tales como directorio que debe ser indexado, tipo de ficheros que se desean incluir, palabras que se desean eliminar del índice, posibilidad de generar un tesauro para el control terminológico, etc. En el caso

de la aplicación de este producto a los documentos XML, el programa crea un fichero de palabras asociando a cada una de ellas el elemento o elementos que la contienen. La siguiente figura ilustra el índice generado para los documentos XML tratados en nuestro desarrollo práctico.

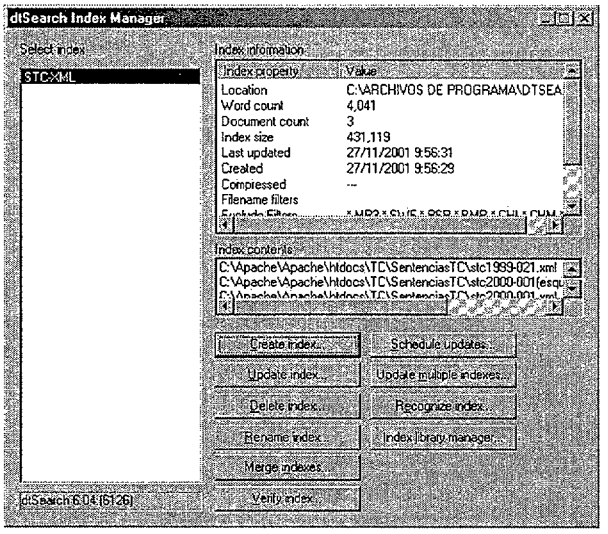


Figura IV.33: Módulo de indexación de la aplicación dtSearch.

– De este modo, y al igual que en el anterior caso, es posible acotar las búsquedas de información según su inclusión en un determinado campo (elemento), o a texto completo, utilizando operadores booleanos, de truncamiento, de proximidad (detallando el número de espacios entre palabras), etc., o combinando diversos criterios búsqueda dentro de una misma instrucción de búsqueda. Asimismo, el sistema ofrece en su versión cliente un listado de todos los campos surgidos tras la indexación de los documentos XML, lo que facilita enormemente la interrogación precisa de la base de datos. El formulario de interrogación existente en el módulo de consulta en la WWW resulta adecuado, con capacidades muy potentes, pues además de las ya señaladas es posible recuperar según la representación fonética de una palabra o haciendo uso de técnicas de recuperación procedentes de la lógica difusa.

Lamentablemente, la presentación de los documentos XML obtenidos tras la interrogación del sistema se produce, al igual que en el anterior caso, mediante la

visualización del código del documento, sin la interpretación de la hoja de estilo asociada al mismo. Para salvar esta dificultad, el programa puede hacer una llamada externa al navegador configurado por defecto. La siguiente figura ilustra un ejemplo de búsqueda documental con esta aplicación y la posterior presentación de los resultados.

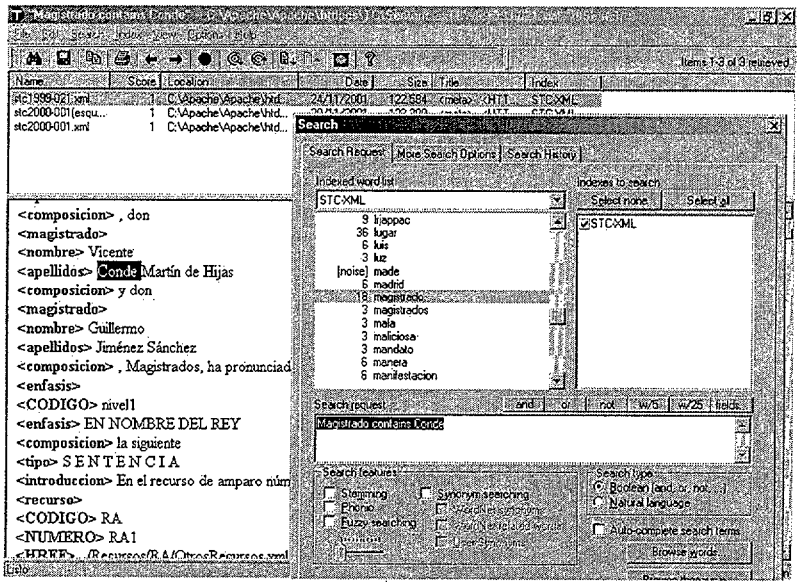


Figura IV.34: Búsqueda y presentación de los resultados obtenidos con la aplicación dtSearch.

Como ya se ha comentado al principio de este punto, aún queda mucho camino por recorrer hasta que existan herramientas que sean capaces de realizar de una forma correcta las tareas de indexación, recuperación y presentación de conjuntos documentales marcados mediante XML. El camino está abierto, como se ha podido observar con algunos productos aquí presentados, por lo que el seguimiento de su evolución será una de las tareas prioritarias en una posible ampliación futura del desarrollo práctico aquí presentado.