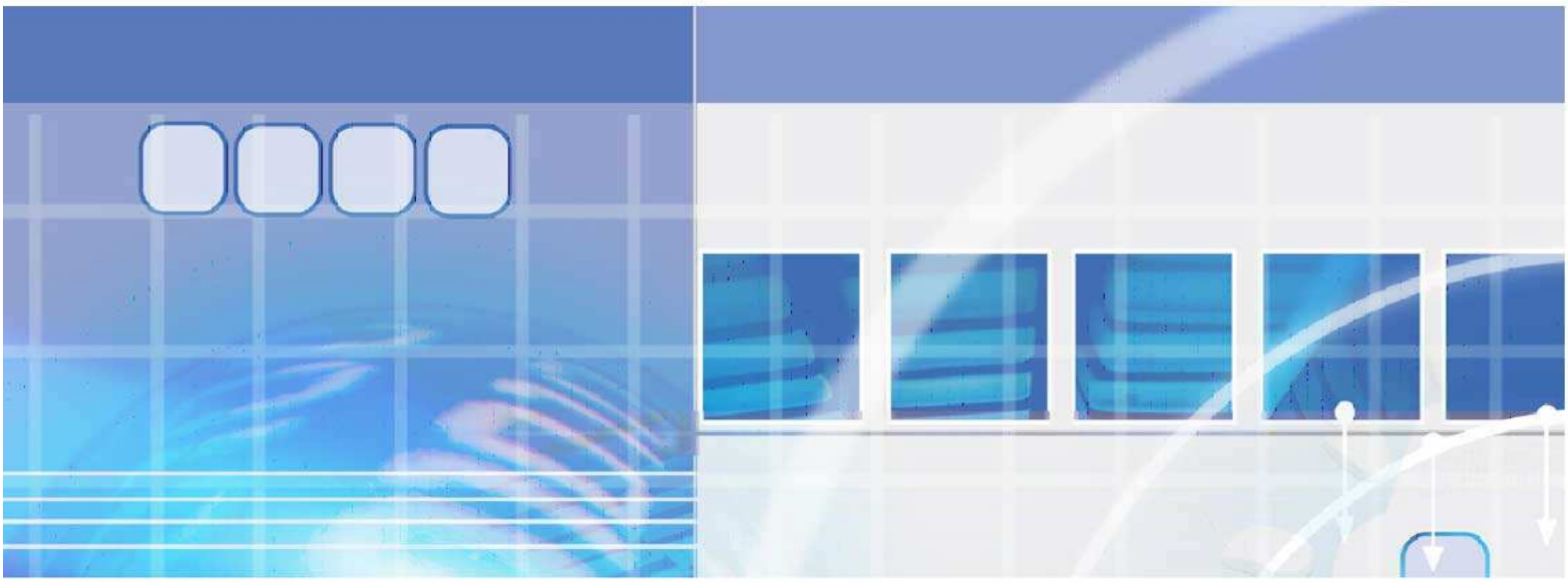


XBRL
eXtensible Business Reporting Language
ESPAÑA

LIBRO BLANCO

Grupo de Trabajo de Tecnología
2006

ÍNDICE

Prólogo

Han pasado casi dos años desde que con mucho esfuerzo e ilusión se presentó la primera edición de este libro. Con la facilidad didáctica que le caracteriza, Don Enrique Bonsón Ponte exponía en su prólogo los avances que este nuevo estándar, llamado a la revolución del intercambio de información financiera, había ido sufriendo desde su aparición allá por el año 1998.

Cualquier cambio, conlleva un proceso de elaboración de lo diferente, una adaptación a lo nuevo, si a esto unimos que XBRL (*eXtensible Business Reporting Language*) es un estándar que afecta a sectores especialmente sensibles como el financiero y el bancario, nos encontramos ante un escenario especialmente complejo. Pero sin duda, si echamos la vista atrás y realizamos un análisis de lo acontecido, podemos afirmar que el XBRL ha pasado de ser un ente abstracto a convertirse en una realidad. Hemos pasado de un marco XBRL teórico a implementaciones concretas en el mundo real, cada día podemos añadir más instituciones y compañías que se unen a la utilización de este estándar.

Hemos podido comprobar la aparición de nuevas taxonomías como LENLOC, ES-BE-COREP, ES-BE-CB, DGI… También se ha avanzado en la aprobación por parte del grupo de tecnología de una primera estrategia de versionado de las taxonomías, que resuelve el siempre importante y crítico problema de la compatibilidad entre versiones.

El continuo avance en las herramientas de conversión, validación y sobre todo visualización dentro del entorno de XBRL, ha permitido separar los aspectos más técnicos de la lógica de negocio, acercando el uso de este estándar a usuarios sin un gran conocimiento de la tecnología empleada. Para finalizar, no me gustaría pasar por alto un agradecimiento especial al desinteresado esfuerzo que están realizando los miembros de la Asociación XBRL España, sin ellos este libro no sería posible, y aprovechar la oportunidad para animar a todos aquellos que aún no formáis parte de la asociación a uniros a este gran proyecto que crece día a día.

Enrique Muñoz García

Director Relaciones Institucionales
Informática El Corte Inglés.

Prólogo a la primera edición

Los avances tecnológicos de las últimas décadas han configurado un nuevo escenario digital para la información financiera. Ahora bien, digital no quiere decir compatible, por lo que un documento (balance, cuenta de resultados, etc.) generado por un programa informático determinado no es directamente legible por otro programa distinto. Esto es debido a que cada fabricante utiliza sus propios formatos de almacenamiento de datos.

La solución tradicional a este problema ha sido la exportación del fichero deseado a formato ASCII (fichero de texto en el que cada campo de información se separa del siguiente por medio de caracteres delimitadores), para, posteriormente, recuperar el fichero ASCII desde el programa que va a incorporar los datos. Con esta solución, se producen pérdidas de tiempo porque es frecuente tener que realizar ciertas modificaciones en el fichero ASCII debido a que los delimitadores que genera el programa exportador no siempre son reconocidos por el programa que realiza la importación y esto puede provocar desplazamientos en las tabulaciones de datos. Otra solución es diseñar un programa específico capaz de realizar la conversión de datos entre dos aplicaciones pero la solución ideal es la utilización de un estándar para el intercambio.

La necesidad de un estándar digital para el intercambio de información contable entre aplicaciones de software se acentúa si de lo que se trata es de integrar múltiples datos procedentes de estados financieros publicados en diversos formatos (pdf, xls, html, doc, etc.). Ese estándar es hoy el XBRL, estándar ampliamente aceptado por la comunidad contable internacional, desarrollado por XBRL.org, un consorcio internacional de empresas y organizaciones, patrocinado por el AICPA (American Institute of Certified Public Accountants), entre las que se encuentran las grandes de la informática, de la contabilidad y la consultoría, e instituciones como el IASB (Internacional Accounting Standards Board), el IMA (Institute of Management Accountants), el CICA (Canadian Institute of Chartered Accountants) o el ICAEW (Institute of Chartered Accountants in England and Wales) .

En España, el XBRL es introducido en 2001 por AECA (Asociación Española de Contabilidad y Administración de Empresas), que a través de su Comisión de Nuevas Tecnologías y Contabilidad comienza a estudiar el estándar y sus posibilidades de utilización en nuestro país y crea un grupo de trabajo que, con la participación de la Central de Balances del Banco de España, Informa, el Instituto de Auditores Censores Jurados de Cuentas de España, Navision (hoy Microsoft), PriceWaterhouseCoopers y la Universidad de Huelva, se constituye como jurisdicción provisional de XBRL Internacional en febrero de 2002, dando como resultado la creación de la jurisdicción definitiva en abril de 2004 mediante la constitución de la Asociación XBRL España para la divulgación de estándares de tecnología[1] , que bajo la presidencia del Banco de España y con AECA como entidad facilitadora ha asumido el desarrollo del XBRL en el estado español.

La creación de la jurisdicción definitiva ha supuesto la consolidación del proyecto XBRL en España y el desarrollo de las primeras taxonomías de ámbito nacional. Estas taxonomías son: La DGI (Datos generales de identificación) que recoge todos los elementos de

[1] http://www.xbrl.org.es

identificación contenidos en los documentos financieros utilizados por la CNMV, la Central de Balances del Banco de España, los Registros Mercantiles y otras entidades o empresas de agregación/distribución de información financiera, la IPP (Información pública periódica de empresas con valores admitidos a cotización oficial) que incorpora todos los elementos de información que las empresas cotizadas remiten periódicamente a la CNMV, la PGC90 (Plan General Contable de 1990) que contendrá todos los elementos de las cuentas anuales del Plan General de Contabilidad de 1990 y las taxonomías coordinadas por el Banco de España (Información remitida por las sociedades y servicios de tasación; Información financiera por parte de las entidades de crédito y sistema de intercambio de información financiera; COREP, Common Reporting, del Comité Europeo de Supervisores Bancarios y SEPBLAC, Servicio Ejecutivo de la Comisión para la Prevención del Blanqueo de Capitales e Infracciones Monetarias).

En este libro, elaborado por el grupo de trabajo de tecnología de la Asociación XBRL España, se recogen los aspectos fundamentales del estándar de una forma clara y concisa. En él se abordan las principales cuestiones que cualquier lector interesado necesita conocer para comprender el alcance y las implicaciones del XBRL, por ejemplo los conceptos básicos, la especificación v2.1, la generación y/o captura de información XBRL, las taxonomías, el control de errores y la validación de informes, el envío, recepción y almacenamiento de informes XBRL, la recepción de información y su tratamiento, la seguridad, el versionado o los productos y servicios XBRL por citar algunas.

En definitiva se trata de un excelente documento de referencia -el primero de estas características escrito en lengua castellana- de obligada lectura no sólo para los profesionales responsables de la adopción e implantación del XBRL en sus empresas u organizaciones, sino también para todos los que necesiten conocer los fundamentos de este estándar para el intercambio de información de creciente difusión dentro de la comunidad financiera internacional.

Para finalizar estas líneas me gustaría destacar el desinteresado esfuerzo realizado por los miembros del grupo de tecnología de la Asociación que han hecho posible que este documento sea una realidad. A todos ellos, muchas gracias.

Huelva, Junio de 2005

Enrique Bonsón Ponte
Catedrático de Economía Financiera y Contabilidad de la Universidad de Huelva.
Vicepresidente de XBRL España.

1 Introducción

La adopción del estándar XBRL para la presentación de información financiera y empresarial por parte de una entidad requiere siempre un esfuerzo, ya que supone integrar una nueva tecnología en los sistemas de información existentes en la entidad. Este esfuerzo varía mucho dependiendo del grado de integración de XBRL en los mencionados sistemas, y va desde la simple preparación de un informe de acuerdo con una taxonomía elaborada por otra entidad, a una profunda reestructuración de procesos o de sistemas, pasando por el desarrollo, aprobación y publicación de taxonomías.

Muchas entidades habituadas a trabajar con XML, no van a ver XBRL como algo especialmente novedoso, ya que se trata de una particularización de XML, pero otras seguramente agradecerán un libro como éste que les facilite el primer contacto con el estándar. Este libro ha sido el primer trabajo abordado por el grupo de Tecnología de XBRL España y se ha pretendido recoger en él las respuestas a las principales preguntas que se plantea el futuro usuario de éste estándar universal para intercambio de informes.

En el capítulo 3 del libro se presenta XBRL como un lenguaje de etiquetado de datos que responde a las necesidades de un lenguaje estándar de cobertura universal para la elaboración de informes empresariales. Se incluyen en el capítulo los conceptos básicos de XML y XBRL y se señalan las particularidades que los diferencian.

En el capítulo 4 se amplían los conceptos XBRL detallándose los elementos que componen una taxonomía y los que componen un informe XBRL. Se tratan las distintas posibilidades de envío, recepción, almacenamiento y recuperación de informes XBRL y se facilita una arquitectura funcional y técnica de implantación de XBRL para que sirva de referencia. Como elemento imprescindible de control en el desarrollo y mantenimiento de una taxonomía, se incluyen unas indicaciones sobre buenas prácticas de versionado. También se hacen consideraciones sobre rendimiento de aplicaciones XBRL y se dan recomendaciones para optimizar ese rendimiento. Aunque la seguridad es algo que está fuera de la especificación XBRL, se ha creído interesante tocar también en este capítulo los aspectos de seguridad en la generación, almacenamiento y envío de informes.

El capitulo 5 contiene un relación de soluciones y herramientas existentes en el mercado, tanto para desarrollo de taxonomías como para la elaboración y manejo de informes XBRL, indicando sus características y prestaciones.

El capítulo 6 está dedicado a la formación, pero sólo con la idea de informar sobre los mínimos conocimientos que deben de poseer los técnicos de un equipo que se proponga abordar un proyecto XBRL. En este capítulo se incluye una referencia al documento de "Formación y Buenas Prácticas", preparado por el Grupo de trabajo de Desarrollo y Formación de XBRL España, en el que si se estudian en profundidad los requisitos de formación y la forma de desarrollar un proyecto de este tipo.

Finalmente, el capitulo 7 contiene algunos ejemplos de implantación de XBRL que pueden servir de referencia a los nuevos usuarios de este estándar.

La versión en red de este libro puede también encontrarse en www.xbrl.org.es donde se publicarán así mismo las actualizaciones periódicas correspondientes.

2 Objetivos del documento

Este Libro Blanco pretende facilitar a todos los interesados en el estándar XBRL el primer acercamiento a esta tecnología. Es objetivo del libro mitigar el riesgo a lo desconocido y aportar una serie de datos y recomendaciones útiles para cualquier entidad que se plantee implantar XBRL. Puede resultar de especial interés para los técnicos que pretendan iniciar un proyecto XBRL de cualquier tipo ya que en él pueden encontrar una referencia a todas las piezas tecnológicas que van a necesitar.

Ha sido una preocupación constante en el desarrollo del libro el elaborar un documento completo, que aportara toda la información imprescindible, presentando con claridad los conceptos que se manejan en el mundo XBRL, pero que al mismo tiempo resultara fácil de leer, incluso para los no familiarizados con la materia. Intencionadamente se ha evitado profundizar mucho en los distintos temas, optándose por el contrario por incluir muchas referencias externas a documentos y sedes web donde el lector interesado puede ampliar sus conocimientos hasta el extremo que desee

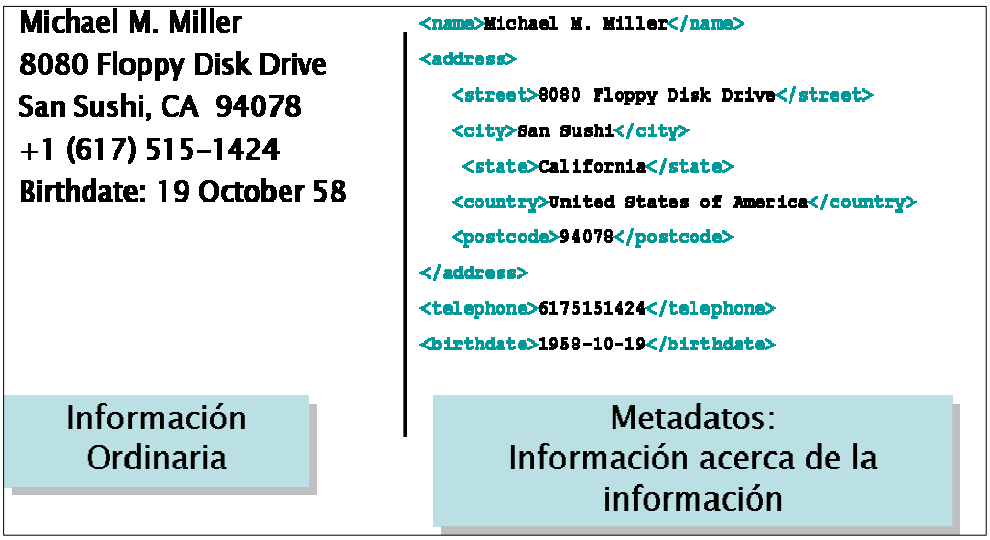
3 XML y XBRL conceptos básicos

3.1 XML como lenguaje universal

XML, acrónimo de eXtended Markup Language o Lenguaje de Marcado Extendido, es un lenguaje de marcado universal y estándar, definido por el World Wide Web Consortium, W3C, para el formateo de información etiquetada.

El formato de etiquetas XML proporciona un significado adicional a la información ordinaria a intercambiar de forma que las aplicaciones informáticas que consumen la información sean capaces de entender dicho significado, son los llamados metadatos de las etiquetas.

Ejemplo 1: La información ordinaria se acota entre etiquetas para dotarlas de significado adicional



Como podemos observar en el ejemplo anterior una aplicación informática capaz de interpretar los metadatos de las etiquetas podría "entender" el nombre de la persona "Michael M. Miller" y distinguirlo separadamente de su dirección postal y de esta forma automatizar el procesamiento de estos datos.

Desde su creación en 1998, XML ha servido como base para la construcción de otros lenguajes según diversos aspectos:

- Orientados al intercambio y extracción de información: SOAP, WSDL, XQuery, XPath, SAX, DOM
- Orientados a formar "vocabularios" específicos de negocio: MathML, MusicML, OTA, HL7, XBRL
- Orientados al formato o presentación de la información: XHTML, XForms, WML, SVG
- Orientados para tratar y transformar el propio XML: XSLT, XSL-FO, XML-Schema, RelaxNG, XLink, XPointer

3.2 Necesidades de un lenguaje estándar de reporting empresarial

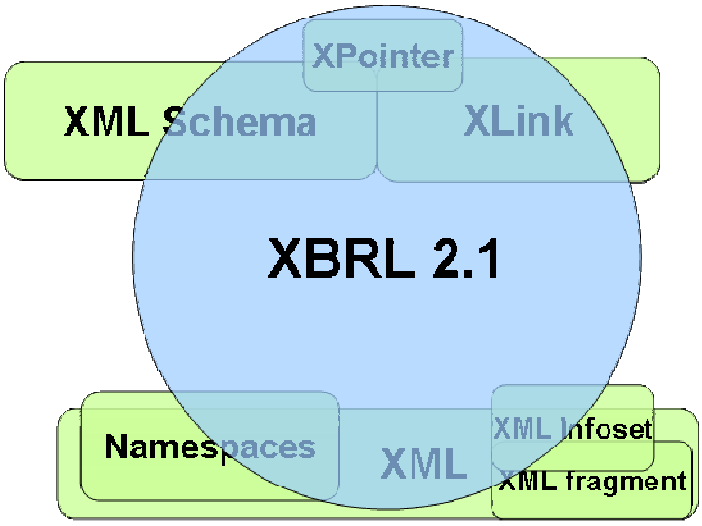
Cuando se sentaron las bases para describir un lenguaje de reporting se pensó en una sintaxis que alcanzase los siguientes requisitos:

- Basado en un formato universal y abierto: eXtended Markup Language, XML
- Las definiciones de los metadatos a intercambiar fuesen definiciones estándar, es decir, que un término como por ejemplo "Caja y depósito en Bancos Centrales" significase siempre lo mismo independientemente de las aplicaciones que usaran dicho término: este es el pilar en el que se sustentan las taxonomías, diccionarios comunes de datos expresados en lenguaje XBRL.
- Además, otro requisito necesario es que estas taxonomías fuesen fácilmente extensibles de forma que diversas industrias, compañías y analistas fueran capaces de publicar

definiciones a medida (siempre independientemente de las aplicaciones informáticas)

- Por último, al no estar implementadas en las aplicaciones informáticas dichos diccionarios de conceptos, la forma de las colecciones de datos pueden variar, consiguiendo un lenguaje con el que expresar datos de calidad guiados por Reglas de Negocio, que puedan ser usados por distintas aplicaciones.

De esta forma, y tras varias revisiones desde 1998 hasta la fecha actual, se construye XBRL, eXtended Business Reporting Language, como un lenguaje cuya sintaxis ha sido diseñada para el intercambio de informes empresariales, basado en XML y otros estándares del W3C complementando a XML como son la especificación de espacios de nombres (Namespaces), la definición de esquemas de datos en XML (XMLSchema) y la definición de recursos enlazados mediante XML (XLink).



3.3 XBRL como un lenguaje de etiquetado expresando reglas

Para conseguir la extensibilidad y garantizar la unicidad en la definición de los conceptos el etiquetado de información XML no es suficiente. Hace falta poder añadir reglas a esa información.

La motivación de dotar al lenguaje etiquetado con reglas es la siguiente:

*El significado de estos conceptos puede variar de acuerdo con los distintos principios contables que se apliquen al objeto de la información
**El nombre de otros conceptos idénticos de otra forma varía de una jurisdicción a otra, por no mencionar la variación en significado al cambiar de un idioma nativo a otro.

Por todo esto se definen reglas que consiguen dotar de semántica a la información expresada en las taxonomías.

Por tanto, en el lenguaje XBRL, adicionalmente a la definición de los conceptos a reportar se expresan unos metadatos de los propios conceptos de la taxonomía, esto son, las relaciones existentes entre los conceptos, y se expresan mediante reglas con sintaxis XML, son las llamadas linkbases, que se describen con más detalle en el siguiente capítulo.

Para hacernos una idea de las reglas que definen estas relaciones, citaremos por ejemplo las siguientes:

- Definir relaciones de cálculo simple entre dos conceptos, El concepto TotalEmpleados se obtiene como la suma entre el valor del concepto EmpleadosFijos con EmpleadosTemporales.
- Relaciones de negocio, El concepto TotalCostesIndirectos es similar a TotalGastosIndirectos.
- Relaciones de presentación, estableciendo dependencias jerárquicas que ayudan a la hora de entender la información reportada.
- Definir relaciones entre conceptos y sus referencias en textos escritos, por ejemplo documentos legales, etc.
- Establecer relaciones entre conceptos y distintos literales de texto que lo describan en diversos formatos e idiomas.

4 XBRL conceptos y recomendaciones

4.1 Especificación XBRL versión 2.1

4.1.1 Descripción de la especificación XBRL versión 2.1

El documento utilizado para realizar esta edición es:
XBRL-RECOMMENDATION-2003-12-31+Corrected-Errata-2005-03-24.doc

Este documento es la base de XBRL versión 2.1. Ha sido realizado por la comunidad que compone el consorcio XBRL (subgrupo de especificación) y se encuentra disponible en Internet en la pagina Web http://www.xbrl.org.

La especificación está organizada en los siguientes capítulos:

Capítulo 1: Introducción, en la que se comenta el propósito de la especificación, la relación con otros documentos, terminología, niveles de conformidad con XBRL y los espacios de nombres utilizados en el resto del documento.

Capítulo 2: Cambios respecto de la especificación 2.0a: en este capítulo se enumeran las diferencias entre las distintas versiones de XBRL, XBRL 2.0a y XBRL 2.1.
La especificación 2.0a se describió en un documento de menos de 40 páginas, en algunos aspectos la especificación 2.0a era muy abierta o demasiado flexible, lo cual llevó a que ciertos editores de taxonomías creasen documentos válidos de acuerdo a la especificación pero sin embargo incompatibles entre sí. La especificación 2.0a ha estado "viva" durante un año (2003) ya que a finales de dicho año se aprobó la nueva versión 2.1 que soluciona todos los problemas que se encontraron en los primeros proyectos de adopción de XBRL.

Capítulo 3: Entorno XBRL: se analiza a continuación el entorno en el que trabaja XBRL. En este capítulo se describe por primera vez un informe y una taxonomía XBRL sin entrar a discutir los detalles. Se describe la función del DTS (Discoverable Taxonomy Set) lo cual es nuevo en XBRL 2.1 y se hace mención explícita de la intención de mantener separados los requerimientos de seguridad de las aplicaciones de la función de XBRL como formato para el contenido de informes empresariales.

XBRL no impone, limita o impide el uso de cualquier otro mecanismo de seguridad que se requiera en cada momento. XBRL y seguridad son tos temas independientes y la especificación por tanto no entra a discutir este entorno.

Los siguientes apartados de este capítulo describen la validación de documentos XBRL y de Taxonomías así como el uso de XLink en XBRL. XLink es una tecnología basada en XML para definir relaciones entre elementos que se encuentran en diferentes ficheros de la misma forma que en HTML podemos definir enlaces entre documentos.

La primera versión de XBRL (1.0) al igual que la mayoría de los lenguajes basados en XML presentaba una estructura jerárquica y anidada de elementos.

INFORME XBRL VERSIÓN 1.0 (ILUSTRATIVO) **EJEMPLO 1**

```
<balance>
        <activo>
                <inmovilizado>1000</inmovilizado>
                …
                <totalActivo>1000000</totalActivo>
        </activo>
        <pasivo>
                <capitalSocial>2000</capitalSocial>
                …
                <totalPasivo>1000000</totalPasivo>
        </pasivo>
</balance>
```

Rápidamente se descubrió que este sistema tenía sus limitaciones. Sólo se podía representar una estructura del balance. No se podían definir fácilmente reglas de negocio que operasen con los valores de los elementos. Y tampoco fue nunca un requerimiento de negocio hacer coincidir la representación de un balance con la estructura en XML. En definitiva eran más los problemas con los que nos encontrábamos por este camino que adoptando una estructura diferente.

XBRL 2.1 presenta todos los datos en una estructura plana.

INFORME XBRL VERSIÓN 2.1 (ILUSTRATIVO) **EJEMPLO 2**

```
<xbrl>
  <schemaRef xlink:href="taxonomia.xsd"/>

  <inmovilizado>1000</inmovilizado>
  …
  <totalActivo>1000000</totalActivo>
  <capitalSocial>2000</capitalSocial>
  …
  <totalPasivo>1000000</totalPasivo>
</xbrl>
```

Este informe se complementa con la información que aparece en la taxonomía donde se describen los elementos utilizados en el informe (inmovilizado, totalActivo, capitalSocial y totalPasivo) así como todas las posibles jerarquías entre elementos que los autores de la taxonomía quieran realizar.

Ejemplos de múltiples estructuras de relaciones entre los elementos pueden ser: Formato de balance por orden de liquidez, balance según el orden establecido en el plan de cuentas… así como estructuras de cálculo entre los elementos, etiquetas en múltiples idiomas y relaciones con los textos legales que ayuden a interpretar el significado y uso de los elementos y relaciones de dependencia entre los elementos.

La tecnología que nos permitirá relacionar los elementos en estas estructuras se denomina XLink. XLink aparece por primera vez en XBRL 2.0 y desde entonces se ha utilizado extensivamente en XBRL hasta el punto de que la especificación XBRL tiene un capítulo específico explicando el uso de XLink en XBRL. No es probable que XBRL abandone el uso de XLink dado que proporciona un mecanismo perfecto para relacionar información de la misma manera que las páginas web mantienen enlaces entre ellas.

4.1.2 Informes XBRL

El siguiente capítulo de la especificación trata en profundidad sobre el formato de los informes XBRL 2.1 y de sus posibilidades.

Con XBRL se puede modelar cualquier tipo de informe empresarial. XBRL no está diseñado para modelar información Financiera exclusivamente. La especificación XBRL 2.1 no presupone que se vaya a representar ningún tipo de informe en concreto.

Es cierto que XBRL ha nacido en el seno de AICPA http://www.aicpa.org (La asociación americana de censores y auditores de cuentas) y que el mayor avance en XBRL existe hoy en día para información financiera. Sin embargo ya en el año 1998 y 1999 XBRL se llamó XFRML (Extensible Financial Reporting Markup Language) el nombre actual (Extensible Business Reporting Language) se adoptó posteriormente, cuando se comprobó que XBRL se puede utilizar no sólo para la información financiera sino para cualquier otro tipo de informes empresariales que se deseen modelar e intercambiar.
Comienza la especificación por mostrar cómo se ha definido el elemento <xbrl> en el esquema correspondiente xbrl-instance-2003-12-31.xsd y muestra un informe de ejemplo que pasamos a describir a continuación:

INFORME XBRL VERSIÓN 2.1 **EJEMPLO 3**

```
<xbrl xmlns="http://www.xbrl.org/2003/instance"
     xmlns:xlink="http://www.w3.org/1999/xlink"
     xmlns:link="http://www.xbrl.org/2003/linkbase"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xmlns:ci="http://www.xbrl.org/sp/general/2005/taxonomia-pgc-2005"
     xsi:schemaLocation="
http://www.xbrl.org/sp/general/2005/pgc-2005
http://www.xbrl.org.es/general/2005/pgc-2005.xsd">
  <link:schemaRef xlink:type="simple"
           xlink:href="http://www.xbrl.org.es/general/2005/pgc-2005.xsd"/>
  <ci:activo precision="3" unitRef="u1" contextRef="c1">727</ci:activo>
  <!-- ... otros elementos de la taxonomía con sus valores ... -->
  <ci:pasivo precision="3" unitRef="u1" contextRef="c1">727</ci:pasivo>
  <context id="c1"><!-- ... --></context>
  <unit id="u1"><!-- ... --></unit>
</xbrl>
```

El elemento xbrl normalmente contendrá las definiciones de los espacios de nombres que se utilizarán en el resto del documento. Esto nos evita tener que definir el espacio de nombres

muchas veces. El texto xmlns:link="http://www.xbrl.org/2003/linkbase" que aparece como atributo del elemento xbrl asocia el prefijo "link" al espacio de nombres "http://www.xbrl.org/2003/linkbase" de forma que cada vez que usamos "link:" en el resto del fichero nos estamos refiriendo a un elemento definido en el espacio de nombres anterior sin tener que escribir todo el texto.

El DTS (Discoverable Taxonomy Set) de este documento lo compone el elemento
<link:schemaRef
xlink:type="simple"
xlink:href="http://www.xbrl.org.es/general/2005/pgc-2005.xsd"/>

En él podemos ver que sólo existe una única taxonomía referenciada pgc-2005.xsd sin embargo nada nos impide utilizar elementos de muchas taxonomías en un mismo informe siempre y cuando todas ellas estén identificadas mediante los elementos <link:schemaRef> correspondientes.

Una taxonomía también puede incorporar elementos de otras taxonomías. En este caso la taxonomía que aparece en el DTS del informe es sólo la que se encuentre al final de la jerarquía.
A continuación, en el ejemplo anterior tenemos dos elementos de la taxonomía (ci:activo y ci:pasivo) con sus valores correspondientes (727 en los dos casos). Estos elementos contienen la siguiente información:

"precision=3" significa que los tres caracteres de la izquierda del número son significativos a la hora de utilizar el valor 727. Para obtener más información respecto del uso de "precision" y "decimals" se debe consultar la especificación XBRL en el capítulo 4.6.3 y siguientes, donde se explica detalladamente y con ejemplos el valor que se debe interpretar en cada caso.

A continuación nos encontramos con unitRef="u1" que hace referencia a la unidad de medida del valor 727. En XBRL todos los valores numéricos deben tener una unidad de medida. Esto se consigue mediante la utilización combinada del atributo unitRef y de sucesivos elementos <unit id="u1">…</unit> en el documento. La sintaxis de definición de unidades permite definir unidades simples del tipo iso4217:EUR para identificar Euros y unidades complejas como m/s2 para definir la aceleración si esto fuera necesario. El capítulo 4.8 de la especificación XBRL detalla cómo definir unidades simples y complejas en XBRL.

El siguiente atributo de los elementos de ejemplo es contextRef="c1". Este atributo permite relacionar el elemento en el que nos encontramos con el contexto dimensional en el que debe ser interpretado. En XBRL los contextos tienen como mínimo dos dimensiones, tiempo y entidad que reporta, sin embargo el contexto es extensible de forma que podemos añadir información relativa a la unidad dentro de la organización (departamento, persona, etc.) y escenario de interpretación del informe (estimado, real,…) Esta información se creará utilizando elementos <context id="c1">…</context> y su sintaxis está descrita en el capítulo 4.7 de la especificación.

4.1.2.1 Elementos simples y complejos: Items y tuplas

En el ejemplo anterior se están utilizando dos elementos de la taxonomía: ci:activo y ci:pasivo. Estos dos elementos, dentro de un determinado contexto tienen por sí mismos toda la información necesaria para ser utilizados. El valor de estos elementos (el número) tiene sentido por si mismo, por tanto podemos decir que son elementos simples.

Existen muchos casos a la hora de elaborar informes en los que la información debe mantenerse agrupada para que pueda ser utilizada. Pongamos como ejemplo que queremos crear un cuadro de directivos de la empresa con el Nombre, Cargo, Salario Fijo y Salario Variable. Para representar esta estructura de datos compleja surge en XBRL lo que denominamos tupla.

La tupla es una estructura de datos que agrupa elementos simples que no proporcionan información si se encuentran dispersos. En el ejemplo anterior, la tupla se podría llamar directivo y dentro de ella existirá un elemento Nombre, un elemento Cargo, un elemento Salario Fijo y otro para el Salario Variable.

En un informe XBRL los datos se verían más o menos así:

TUPLA EN UN INFORME XBRL **EJEMPLO 4**

```
<ci:directivo>
  <ci:nombre contextRef="c1">Juan Ramón Martínez</ci:nombre>
  <ci:cargo contextRef="c1">Director Financiero</ci:cargo>
  <ci:salarioFijo unitRef="Euro" contextRef="c1" precision="INF">45000</ci:salarioFijo>
  <ci:salarioVariable unitRef="Euro" contextRef="c1"
precision="INF">15000</ci:salarioVariable>
</ci:directivo>
<ci:directivo>
  <ci:nombre contextRef="c1">José Fernandez</ci:nombre>
  <ci:cargo contextRef="c1">Director General<ci:cargo>
  <ci:salarioFijo unitRef="Euro" contextRef="c1" precision="INF">55000</ci:salarioFijo>
  <ci:salarioVariable unitRef="Euro" contextRef="c1"
precision="INF">20000</ci:salarioVariable>
</ci:directivo>
```

4.1.2.2 Notas aclaratorias en XBRL

Una de las características más significativas de los informes XBRL es la posibilidad de incorporar notas a los elementos que aparecen en los informes. En la vida real las notas aparecen en prácticamente todos los informes financieros corporativos y a menudo contienen información muy valiosa y que no debe ser ignorada.

XBRL incorpora la posibilidad de que los informes incorporen las notas utilizando para ello la potencia de XLink.

NOTA AL PIE DEL ELEMENTO CUENTASACOBRAR **EJEMPLO 5**

```
<ci:cuentasACobrar id="item1" unitRef="Euro" contextRef="c1" precision="INF">455680</
ci:cuentasACobrar>

<link:footnoteLink
  xlink:type="extended" xlink:title="Notas"
  xlink:role="http://www.xbrl.org/2003/role/link">
  <link:footnote
    xlink:type="resource"
    xlink:label="footnote1"
    xlink:role="http://www.xbrl.org/2003/role/footnote"
    xml:lang="sp">Cuotas, eventos y cursos por cobrar (del ejercicio).- Variación producto
del traspaso de los saldos de 2004 a ejercicios anteriores, de la provisión por cuotas no
pagadas en el plazo establecido por algunos miembros de la XXXX así como eventos
realizados pendientes de pago.
<link:footnote>
  <link:loc xlink:type="locator" xlink:label="fact1" xlink:href="#item1"/>
  <link:footnoteArc
     xlink:type="arc"
     xlink:from="fact1" xlink:to="footnote1"
     xlink:title="ver nota aclaratoria"
     xlink:arcrole="http://www.xbrl.org/2003/arcrole/fact-footnote"/>
</link:footnoteLink>
```

En el ejemplo vemos cómo el elemento cuentasACobrar tiene un identificador id="item1" el cual es utilizado en el localizador <link:loc …/> para referenciar el elemento al que pertenece la nota. El texto de la nota se introduce como un recurso y la relación entre el elemento y el texto de la nota se define en el arco footnoteArc.
Se pueden crear notas en muchos idiomas de forma que el usuario pueda acceder a la información que entiende. En el caso anterior bastaría con crear un elemento

```
<link:footnote
   xlink:type="resource"
   xlink:label="footnote1"
   xlink:role="http://www.xbrl.org/2003/role/footnote"
   xml:lang="en">Payments - Variation of the amount … including … excluding …
<link:footnote>
```

Dentro del elemento <link:footnote> anterior.
Se puede consultar más información respecto de las notas en el capítulo 4.11 del documento de la especificación.

4.1.3 Taxonomías XBRL

Según la teoría de la comunicación, para que se pueda intercambiar un mensaje entre un emisor y un receptor, debe existir un código que sea conocido por los participantes. Este es el papel de las Taxonomías XBRL.

Una Taxonomía contiene:

Esquema: El conjunto de elementos que pueden aparecer en los informes y la estructura de los mismos. Este conjunto lo denominaremos el diccionario de términos definidos.

- **Linkbase de etiquetas**: Las etiquetas o textos asociados a los elementos del diccionario que pueden utilizarse en distintos idiomas y con distintos propósitos a la hora de construir representaciones de los informes.
- **Linkbase de referencias**: Las referencias a textos legales o normativas que fundamentan la base legal del concepto a modelar. Estas referencias juegan un papel muy importante a la hora de aclarar la utilización de los conceptos cuando se van a crear los informes.
- **Linkbase de presentación**: Las reglas para construir una representación del informe que se pretende modelar.
- **Linkbase de cálculo**: Las reglas de cálculo (sumas y restas) entre elementos de la taxonomía que permiten validar los informes XBRL.
- **Linkbase de definición**: Reglas adicionales que permiten documentar relaciones entre elementos de la taxonomía y que se utilizarán para validar los informes.

El capítulo 5 de la especificación se dedica por completo a explicar las taxonomías XBRL.

Toda Taxonomía XBRL está basada en un esquema XML. Las reglas y limitaciones de los esquemas XML también se aplican a las taxonomías XBRL.

Una taxonomía XBRL puede incluir otra taxonomía XBRL. Esta característica de XBRL es fundamental para implementar el modelo de extensibilidad. Una empresa que quiera proporcionar más información en sus informes siempre podrá crear una extensión de la taxonomía original en la que incluya los elementos y relaciones que no estén creadas en la taxonomía anterior.

4.1.3.1 Definición de conceptos en las taxonomías XBRL

Los conceptos se definen en las taxonomías creando elementos que tengan un tipo de datos definido en los esquemas de XBRL y que pertenezcan al grupo de elementos de item o tupla.

DEFINICIÓN DE UN ELEMENTO EN UNA TAXONOMÍA XBRL **EJEMPLO 6**

```
<element
id="tax_CajaYBancos"
name="CajaYBancos"
xbrli:periodType="instant"
  type="xbrli:monetaryItemType"
  substitutionGroup="xbrli:item" nillable="true"/>
```

En el ejemplo tenemos la definición en el esquema de un concepto simple de la taxonomía llamado CajaYBancos.

A continuación mostramos cómo este elemento se utilizará en un informe XBRL.

```
<tax:CajaYBancos unitRef="u1" contextRef="c1"
precision="7">1523554</tax:CajaYBancos>
```

Todos los elementos que se definan en taxonomías XBRL deben tener un atributo xbrli:periodType="instant" o "duration" lo cual permitirá identificar la relación que el concepto tiene con la coordenada tiempo.
Los conceptos de tipo "instant" son válidos en un momento concreto del tiempo. Los conceptos de un balance de situación por ejemplo son de tipo "instant".

Los conceptos de tipo "duration" tienen un rango de fechas (fecha de comienzo y fecha de finalización) para los cuales el dato es válido. Los datos de una cuenta de resultados hacen siempre referencia a conceptos de tipo "duration".

El valor normal que debemos asumir a la hora de realizar taxonomías es que los conceptos son de tipo "duration" a no ser que exista una razón para que sean de tipo "instant".

En XBRL hay algunos atributos adicionales que se pueden definir para algunos conceptos. Es el caso del atributo balance="credit" o "debit" que se puede utilizar para identificar si un elemento aparece en la parte izquierda o derecha.

En el capítulo 4.9 de la especificación XBRL tenemos documentación y ejemplos de cómo crear una tupla de datos en una taxonomía y cómo introducir datos en un informe XBRL.

Se pueden definir en una taxonomía conceptos abstractos. Los cuales no se pueden utilizar en los informes, pero añaden claridad a la hora de mostrar la estructura de la taxonomía. Por ejemplo, el concepto "Balance de Situación" no es ningún elemento que contenga datos por si mismo. Sin embargo, en la linkbase de presentación podremos hacer que este elemento sea el padre del Activo y del Pasivo, y así mantener la taxonomía organizada de forma cómoda.

4.1.3.2 Linkbase de etiquetas

Las taxonomías XBRL mantienen meta-información sobre los conceptos definidos en el esquema que ayudan a la hora de documentar los conceptos y también a la hora de hacer aplicaciones informáticas.

Este es el caso por ejemplo de la linkbase de etiquetas. Todas las linkbases se implementan en ficheros separados del esquema de la taxonomía, las relaciones entre la información de la linkbase y los conceptos de las taxonomías se hacen mediante XLink.

La linkbase de etiquetas proporciona los textos que aparecen en la parte izquierda de los datos. Los humanos interpretamos fácilmente que el dato corresponde al concepto que aparece en la misma fila. Si en un informe aparece por ejemplo:

Caja y Bancos…… 1000 €

Una persona identificará el concepto mediante el texto de la izquierda, y el valor en la columna de la derecha.

El texto "Caja y Bancos" aparecerá en XBRL en la linkbase de etiquetas, de esta forma podremos tener textos en múltiples idiomas y textos distintos para adaptarnos a los dispositivos en los que se tenga que formatear la información.

Las etiquetas tienen un rol asociado. Es posible definir roles propios de una taxonomía y luego definir las etiquetas que un concepto tendrá de acuerdo con lo distintos roles. Un ejemplo de utilización de roles puede ser el texto asociado al concepto "resultado de explotación". Si es positivo será "Beneficio de explotación" si es negativo será "Pérdidas de explotación".

El capítulo 5.2.2 de la especificación proporciona más información respecto de la utilización de la linkbase de etiquetas.

4.1.3.3 Linkbase de referencias

Una de las linkbases más importantes en la definición de una taxonomía de uso público es la linkbase de referencias.

Esta linkbase proporciona información respecto de los textos legales en los que podemos encontrar información adicional sobre el concepto. Esta linkbase será de mucha utilidad a la hora de localizar los conceptos que deben usarse para elaborar informes XBRL que utilicen la taxonomía del IFRS por ejemplo.

El procedimiento más sencillo para elegir un concepto de la taxonomía sería el siguiente:

1 Partiendo del concepto que aparece en el documento contable localizaremos las referencias en el IFRS mediante la documentación proporcionada por los auditores de la empresa o el libro de cuentas corporativo. Esta información será por ejemplo IAS-20 lo cual significa que el concepto utilizado en el informe se encuentra documentado en el libro de normativa internacional de contabilidad IAS, párrafo 20.
2 El siguiente paso sería buscar en la linkbase de referencias usando IAS, 20 qué conceptos están referenciados.

Este procedimiento sencillo permitirá crear un mapa entre los conceptos contables utilizados en una empresa y la normativa internacional de contabilidad.

El capítulo 5.2.3 de la especificación XBRL define la sintaxis a utilizar en las taxonomías para documentar los elementos que definen las relaciones con los textos legales. Las taxonomías son muy flexibles a la hora de crear nuevos conceptos que se puedan utilizar para este propósito.

4.1.3.4 Linkbase de presentación

Esta linkbase tiene un doble propósito: por un lado sirve para que las herramientas de creación o visualización de taxonomías nos muestren el contenido de la misma de forma más amigable que una simple lista de conceptos. Por otro lado sirven de base para que las aplicaciones que formatean los informes de forma automática tengan un punto de partida por el que empezar a construir las plantillas que mostrarán los datos.

La linkbase de presentación tiene una estructura jerárquica. Se construye relacionando los elementos hijos con los elementos padre usando XLink.

El capítulo 5.2.4 de la especificación XBRL proporciona más información sobre la utilización de esta linkbase.

Existe un atributo opcional llamado preferredLabel que aplica al arco XLink que une dos elementos en una linkbase de presentación. Este atributo puede contener el valor del rol que el elemento juega en ese momento específico. Es muy habitual que a la hora de modelar algunos apartados de la memoria nos encontremos con:

Movimientos en las Reservas:
Reservas al inicio del ejercicio ……10000 €
Incrementos o disminuciones de las reservas……2500 €
Reservas al final del ejercicio ……12500 €

En XBRL esto se debe modelar de la siguiente manera:

Existen tres conceptos en la taxonomía. Uno denominado Reservas, otro denominado Incrementos o disminuciones de las reservas y un concepto abstracto denominado Movimientos en las Reservas

En concepto Reservas tiene tres etiquetas:

"Reservas" para el rol "etiqueta normal" (se utilizará en el balance por ejemplo)
"Reservas al inicio del ejercicio" para el rol "inicio del ejercicio"
"Reservas al final del ejercicio" para el rol "final del ejercicio"

El concepto Reservas será de tipo "instant".

El concepto Incrementos o disminuciones de las reservas será de tipo "duration".

El concepto Movimientos en las Reservas es de tipo abstracto, por lo que no es significativo que sea "instant" o "duration".

En la linkbase de presentación, dentro de algún apartado de la memoria figurará que el concepto Reservas es el primer hijo del elemento abstracto Movimientos en las reservas, y que el preferredLabel es "inicio del ejercicio".

En la misma linkbase de presentación, aparecerá el elemento Incrementos o disminuciones de las reservas como segundo hijo del elemento abstracto Movimientos en las reservas.

Por último, aparecerá otra vez el elemento Reservas como tercer hijo del elemento abstracto Movimientos en las reservas pero en esta ocasión la preferredLabel será "final del ejercicio".

4.1.3.5 Linkbase de cálculo

Esta linkbase permite crear relaciones entre elementos similares a las de la linkbase de presentación. En esta ocasión, los elementos padre serán el resultado de las operaciones aritméticas que se deban realizar con los elementos hijos.

Estas operaciones sólo podrán ser sumas o restas de los elementos hijos.

La especificación define en el capítulo 5.2.5 la forma de definir estas redes de cálculo.

Los documentos XBRL se pueden validar con respecto a estas redes de cálculo. Los documentos XBRL que presenten errores en los cálculos no serán válidos desde el punto de vista de XBRL. Las aplicaciones informáticas podrán tener en cuenta estos errores de cálculo para admitir o rechazar automáticamente los documentos entrantes.

4.1.3.6 Linkbase de definición

La última de las linkbases definidas en la especificación XBRL 2.1 es la linkbase de definición. El contenido de esta linkbase es establecer relaciones entre los elementos de una taxonomía que permitan explicar o documentar relaciones así como añadir ciertas reglas que pueden ser importantes a la hora de validar documentos XBRL.

Las relaciones entre elementos definidas en la especificación son 4:

"general-special": Define relaciones de lo general a lo específico. El ejemplo que aparece en la especificación es que zipCode es una especialización de postalCode.
"essence-alias": Este arco se suele utilizar a la hora de relacionar conceptos de diferentes taxonomías o dos conceptos en la misma taxonomía para indicar que ambos son esencialmente el mismo.
"similar-tuples": se utiliza de forma similar al "essence-alias" pero para las tuplas Existen diferencias con respecto al arco "essence-alias" que vienen documentadas en la especificación.
"requires-element": Este arco se utiliza para obligar a que exista un elemento en un informe en el caso de que exista otro elemento.

Las linkbases son también extensibles. Nada en la especificación impide que se desarrollen linkbases propietarias para relacionar modelos de datos internos con elementos de taxonomías si bien esas linkbases deberán ser privadas dado que no existe una especificación aprobada en el consorcio para que todos los procesadores XBRL las entiendan.

4.1.4 Referencias

Especificación XBRL 2.1:

XBRL-RECOMMENDATION-2003-12-31+Corrected-Errata-2005-03-24.doc

4.2 Almacenamiento

4.2.1 Almacenamiento XML / XBRL

La aparición de nuevas tecnologías suele aportar soluciones a problemas existentes y abrir nuevas posibilidades de desarrollo, pero también suelen traer consigo nuevos requerimientos.

La implantación de lo que podríamos llamar tecnología XML ha resuelto determinados problemas sobretodo en el ámbito de las comunicaciones entre plataformas diversas, pero también ha traído consigo requerimientos, tales como el almacenamiento de los documentos y el acceso a sus contenidos, estructurados de manera distinta a la información generalmente hasta ahora tratada.

XBRL surge como una versión potente y flexible del XML, definida específicamente para satisfacer las exigencias de la información financiera y empresarial. Si bien XBRL desde un punto de vista funcional plantea una especialización con respecto a XML, desde un punto de vista tecnológico no plantea ningún cambio 'estructural' que impida que la mayor parte de las herramientas y plataformas de gestión, administración y explotación de contenidos diseñadas para XML no puedan ser utilizadas con XBRL.

Este documento pretende identificar los requerimientos de los contenidos XBRL y las técnicas actuales que plantean soluciones a esos requerimientos, desde el punto de vista de su almacenamiento, gestión y explotación. Para ello, se analizan los distintos métodos de almacenaje, se identifican los principales productos que existen en el mercado que implementan dichos métodos y se describen las herramientas destinadas a la administración, gestión y explotación de los contenidos XBRL.

4.2.1.1 Almacenamiento en ficheros

Los documentos o instancias XBRL se estructuran y articulan normalmente en ficheros de texto 'plano', sobre esta base, la primera opción a plantearse sería almacenar los contenidos XBRL en su 'envase' habitual, es decir, el propio fichero.

Ventajas:

- El documento no es tratado y por tanto su contenido (información) no se ve sometido a ninguna alteración o distorsión.
- 'Facilidad' para gestionar los documentos a nivel de archivo, siempre que se articule una estructura de ficheros adecuada que permita clasificarlos atendiendo a los criterios establecidos por la propia organización.

Desventajas:

- Se presentan los problemas típicos asociados a la gestión de ficheros: falta de concurrencia, comprobación de integridad, seguridad...

Productos / Herramientas:

Partiendo del supuesto, de que los contenidos XBRL son almacenados para su posterior consulta, publicación o explotación por parte de otros sistemas, se hace necesario, contar

con herramientas que permitan acceder, localizar y extraer información. En el caso de un almacenamiento en ficheros, algunas de las herramientas disponibles son las siguientes:

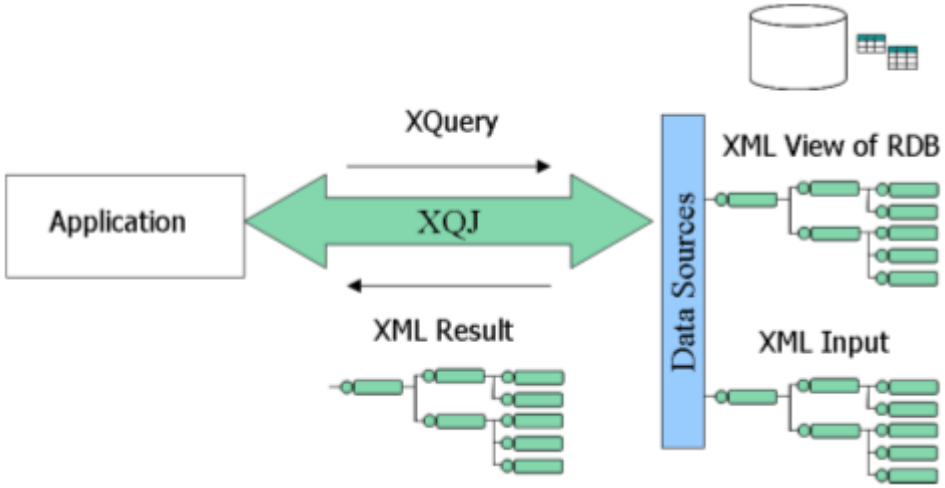
- Motores de búsqueda para ficheros XML/XBRL cuya sintaxis de consulta se basa en Xquery. La mayor parte de estos motores de búsqueda proporcionan un API que permite el acceso y uso del motor desde aplicaciones externas (normalmente Java o C++).

  - **XQEngine** (http://xqengine.sourceforge.net).

- Desde aplicaciones desarrolladas en Java, C++... se pueden invocar motores de búsqueda XML/XBRL partiendo de librerías (APIs) ya desarrolladas y disponibles en el mercado que implementan motores de búsqueda basados en Xquery sobre archivos XML/XBRL.

  - **XQJ**: es el API de Xquery para JAVA. Lo que es JDBC para SQL, lo es XQJ para Xquery. Es una manera de estandarizar el acceso y que cada producto no tenga que tener su propio API.

Está siendo desarrollado por la Java Community Process (JCP especificación 225), los detalles de su desarrollo se pueden ver en http://jcp.org/en/jsr/detail?id=225. Actualmente se encuentra disponible la versión 2.3 que puede ser descargada desde esta página.



  - **Microsoft.Xml.Xquery.dll**: librería de Microsoft para implementar consultas en archivos usando Xquery.
  - Existen también entornos de desarrollo que integran edición y búsquedas Xquery y Xpath proporcionando además un API que integra XQJ:
  - **DataDirect Xquery** (http://www.stylusstudio.com)

4.2.1.2 Almacenamiento en Bases de Datos Relacionales

Muchas empresas que se planteen la disyuntiva del almacenamiento de los documentos XBRL, optaran por la reutilización de los sistemas de almacenamiento ya existentes en su organización. En este caso lo más habitual serán las bases de datos relacionales. La pregunta a plantearse en ese momento será: ¿Cómo almaceno contenidos XBRL en una base de datos relacional?. La respuesta pasa por una serie de opciones:

- Transformando los contenidos XBRL al modelo relacional.
- Almacenando los contenidos íntegros en columnas de un tipo específico.

Ventajas:

- Las bases de datos relacionales son productos muy robustos y que podrían considerarse 'maduros' teniendo en cuenta su evolución e implantación en el mercado.
- La gran mayoría de las aplicaciones actuales que acceden, consultan, gestionan, analizan, publican... contenidos, se encuentran cimentadas entorno a sistemas de bases de datos relacionales, por lo que acomodar la información XBRL a estos sistemas trae consigo la posibilidad de reutilizar estas aplicaciones.

Desventajas:

- La conversión y transformación de los datos XBRL a un modelo relacional exige una manipulación, en mayor o menor medida, de la información. Esta manipulación conlleva en determinados casos que no se pueda garantizar la integridad, ni asegurar que lo que se muestra o recupera sea exactamente lo mismo que el documento original recibido en XBRL.
- Si el documento XBRL no ha sido generado a partir de un esquema relacional, la adecuación posterior a un modelo relacional no resulta sencilla sobretodo en la conversión de determinados elementos:

  - Elementos anidados.
  - Elementos que se repiten (atributos multivaluados).

4.2.1.2.1 Transformación de contenidos XBRL a modelos relacionales

El proceso de transformación conlleva la 'fragmentación' del contenido del documento, es decir, los datos que contiene el archivo XBRL son extraídos y almacenados en entidades de la Base de Datos.

Para empezar, conviene aclarar que utilizando este método de almacenamiento no se guarda información en formato XBRL como tal en la Base de Datos, el documento XBRL es completamente ajeno a la BDD y una vez es utilizado para extraer la información es descartado.

El hecho de recuperar un documento significa consultar a la BDD y construir un documento XBRL con los resultados obtenidos.

Una consecuencia importante de utilizar este sistema es que existe información referente al documento XBRL que no llega a almacenarse en la BDD y se pierde como puede ser el orden en que aparecen los elementos en el documento.

El modelo relacional será eficiente en la medida que los datos sean altamente estructurados y tengan un esquema conocido. Este modelo aportará la ventaja de que permite hacer consultas de la manera tradicional, aunque éstas debido a la estructura del XML requerirán, a menudo, una gran cantidad de joins.
Sin embargo esta opción no es la mejor si existen elementos anidados o elementos que se repiten, ya que su uso obliga a usar representaciones en árbol o a almacenar la relación entre elementos de nivel superior e inferior.

La manera más sencilla de usar este tipo de almacenamiento es definir un mapeo entre los datos del archivo XML y las tablas de la base de datos. De esta manera se pueden cargar datos de manera masiva. También este mapeo se utilizaría para el proceso inverso, es decir tenemos datos en tablas y queremos generar el XML a partir de ellos.

Productos / Herramientas:

- **SQL Server 2005** (http://www.microsoft.com)

Proporciona una herramienta destinada al 'Mapeo' entre almacenamiento XML/XBRL y relacional. Esta herramienta es conocida como Annotated XML (AXSD).

AXSD permite descomponer un documento XML/XBRL en columnas de una o más tablas, preservando la fidelidad de los datos a nivel relacional – se preserva la estructura jerárquica, mientras que se ignora el orden entre los elementos. El esquema no puede ser recursivo.

Al definir un mapeo entre los esquemas XML y las tablas en la base de datos, se crea una "vista XML" de los datos persistentes. La carga de XML puede ser utilizada para poblar las tablas subyacentes utilizando la vista XML. Se puede consultar la vista XML utilizando Xpath 1.0, la consulta es traducida a las consultas SQL en las tablas. De manera similar, las actualizaciones se propagan a esas tablas.

Esta tecnología es útil cuando:

- Se desea tener un modelo de programación centrado en XML utilizando vistas XML sobre sus datos relacionales existentes.
  - Tiene un esquema (XSD, XDR) para sus datos XML, el cual puede haber sido provisto por un socio de negocios externo.
  - El orden no es importante en sus datos, o sus datos disponibles para consultas no son recursivos, o la profundidad máxima de recursividad se conoce con antelación.
  - Desea consultar y modificar los datos mediante la vista XML utilizando Xpath 1.0.
  - Desea cargar datos XML y descomponerlos en las tablas subyacentes utilizando la vista XML.

- **DB2 XML Extender** (http://www.ibm.com)

IBM incorpora a su DB2-UDB a partir de la versión 7.0, un nuevo módulo denominado 'DB2 XML Extender'. Se trata de una herramienta destinada a la integración de XML y DB2, que permite almacenar documentos XML en bases de datos atendiendo a un método denominado: 'Método XML Collections'.

Método XML Collections:
Este método permite mapear la estructura de un documento XML a tablas de DB2. De esta forma se pueden generar documentos a partir de datos

almacenados en la BDD, y viceversa, fragmentar documentos XML y guardarlos en la BDD.

Una XML Collection está formada por un conjunto de entidades de la BDD que han sido mapeadas contra un documento XML por medio de un documento de tipo DAD (Data Access Definition).

Este tipo de fichero, generado en xml, es el que permite al XML Extender construir sentencias de tipo INSERT y SELECT que permitan publicar xml o fragmentarlo en la BDD. Digamos que posibilita un mapeo bidireccional. Para publicar datos xml usando XML Extender, se ha de llamar a una serie de procedimientos almacenados de base de datos creados para tal efecto.

IBM recomienda el uso de las XML Collections en los siguientes casos:

- Si se poseen datos previamente en la BDD y a partir de ellos se pretende generar documentos en xml.
- Si se poseen documentos xml cuya estructura posibilita un mapeo óptimo con entidades de la base de datos.
- Si lo que interesa almacenar de los documentos xml son tan solo sus datos dejando a un lado sus tags.
- Si se prevén actualizaciones frecuentes de partes de esos documentos.
- Si los documentos exceden de 2 gigabytes de tamaño.

- **Oracle XML DB** (http://www.oracle.com)

Oracle XML DB es la base de datos Oracle 9i con soporte para XML, concebida para la gestión de contenidos XML y el desarrollo de aplicaciones basadas en dicho formato

Esta BDD implementa un tipo de datos específico para XML (XMLType) que, entre otras cosas, posibilita la fragmentación en estructuras relacionales de contenidos xml para un mejor aprovechamiento de las capacidades SQL. Para ello, el tipo xml se asocia a un esquema que defina su estructura.

Los elementos del esquema XML se mapean como objetos en los que cada elemento anidado de tipo simple es representado por un atributo de un tipo nativo lo más acorde posible con el tipo del esquema: si es un número con NUMBER, si es texto con VARCHAR ...

El XMLType tiene toda una serie de métodos ya definidos para crear, extraer, indexar, etc., información xml almacenada en la BD, que están disponibles a través de las APIs para PL/SQL y Java.

Entre las características más destacables de este producto:

- Se preserva el estándar DOM a la hora de fragmentar y almacenar documentos. Esto quiere decir que el proceso de almacenamiento no afecta al orden de los elementos, la presencia de namespaces, etc., salvo los espacios en blanco.

- Da la posibilidad de crear tablas y tipos a través de un esquema xml dado, forzando a que un documento xml que se pretende almacenar sea válido para ese esquema.
- Permite actualizaciones parciales de información mediante el uso de Xpath.
- Posibilita el uso de la sintaxis del Xpath mediante SQL embebido para poder consultar contenido xml residente en la BDD.
- Con Xpath se pueden indexar partes de un documento para optimizar búsquedas.
- Permite la creación de vistas Xml para crear estructuras permanentes de información a través de fragmentos de varios documentos xml o de objetos relacionales de la base de datos, mostrándolos como documentos xml.
- Incorpora un repositorio donde poder ver el contenido xml almacenado a través de una estructura jerárquica de directorios, de manera opuesta al método tradicional de las bases de datos relacionales, donde cada directorio es un nodo en la jerarquía que contiene un conjunto de recursos.

4.2.1.2.2 Almacenamiento de documentos íntegros en columnas de tipo 'XML'

Una solución para el almacenamiento de XML/XBRL es el empleo de las columnas de tipo XML que los distintos fabricantes de software (Bases de Datos) han incluido en sus productos.

También se debe tener en cuenta que al tratarse de contenidos basados en 'texto plano', siempre existe la posibilidad de guardar el XML en un campo de tipo VARCHAR, con el inconveniente de que esto es útil sólo si se opta por almacenar y recuperar el texto íntegro sin necesidad de búsquedas interiores. No obstante, en este caso se debe tener en cuenta que:

- Se puede combinar con las columnas de tipo XML para mantener una copia exacta (aunque exista redundancia), por ejemplo para documentos legales.
- Se puede convertir en tipo XML en tiempo de ejecución para ejecutar Xquery por ejemplo, aunque penalizaría bastante el rendimiento.

Lo más habitual será no utilizar el tipo VARCHAR sino el tipo de datos XML Este uso será aconsejable cuando:

- No se conoce la estructura de los datos, o la estructura de sus datos puede cambiar significativamente en el futuro.
- Los datos representan jerarquía de contención (de manera opuesta a las referencias entre entidades) y muchos son recursivos.
- El orden es inherente en sus datos.
- Se necesita que los datos sean validados por el motor de la BDD contra un esquema, un DTD o una taxonomía.

Resumiendo, el almacenamiento en columnas de tipo XML es útil cuando tiene documentos XML con una variedad amplia de estructuras, o documentos XML conformes con esquemas complejos o diferentes que son muy difíciles de mapear con estructuras relacionales.

Un tipo de datos XML permite que los datos sean indexados, que puedan ser consultados o modificados por medio de Xquery y XML DML (extensión para la modificación de datos).

Productos / Herramientas:

- **SQL Server 2005** (http://www.microsoft.com)

Los datos son almacenados en una representación interna que preserva el contenido XML de los datos, como por ejemplo la jerarquía de contención, orden del documento, valores de elementos y atributos, y así sucesivamente. De manera específica, se preserva el contenido InfoSet de los datos XML (para más información acerca de InfoSet, visite http://www.w3.org/TR/xml-infoset). Puede no ser una copia exacta del texto XML, ya que la siguiente información no es retenida: espacios en blanco insignificantes, orden de atributos, prefijos de nombres, y declaraciones XML.

Una columna de tipo de dato XML puede ser creada en una tabla conteniendo columnas relacionales, o en una tabla separada con una relación de clave externa a una tabla principal. Se aconseja que se cree una columna de tipo de dato XML en la misma tabla cuando se cumple una de las siguientes condiciones:

- La aplicación realiza extracción de datos en una columna XML y no requiere un índice XML en la columna XML.
- Se desea construir un índice XML en una columna de tipo de dato XML, y la clave principal de la tabla principal es la misma que la clave de clustering.

Se aconseja crear una columna de tipo de dato XML en una tabla separada si se da alguna de las siguientes condiciones:

- Se desea construir un índice XML en la columna de tipo de dato XML, pero la clave primaria de la tabla principal no es la misma que la clave de clustering, o la tabla principal no cuenta con una clave primaria, o la tabla principal no cuenta con clave de clustering. Esto puede ser verdadero si la tabla principal ya existe.
- No se desea que el escaseado de tabla disminuya debido a la presencia de la columna XML en la tabla, lo cual ocupa espacio si se encuentra almacenado en la fila o fuera de la fila.

Las columnas de tipo de datos XML, variables y parámetros pueden ser tipados utilizando esquema XML pero no utilizando DTD.

Los índices XML pueden ser creados en una columna de tipo de datos XML. Los mismos indexan todas las etiquetas, valores y paths sobre las instancias XML en la columna, y beneficia el rendimiento de la consulta. Sus aplicaciones pueden beneficiarse de un índice XML bajo las siguientes condiciones:

- Las consultas en columnas XML son comunes en su carga de trabajo. Debe ser tenido en cuenta el costo de mantenimiento de índices XML durante la modificación de datos.

- Sus valores XML son relativamente grandes y las partes recuperadas son relativamente pequeñas. Construir el índice evita analizar los datos durante runtime, y beneficia la búsqueda por índices para el procesamiento efectivo de consultas.

SQLServer implementa dos tipos de datos XML denominados 'XML sin tipo' y 'XML con tipo'. La diferencia más importante entre ambos tipos se sustenta en la validación o no de los datos XML contra un esquema dado.

Microsoft recomienda el uso de 'XML sin tipo' bajo las siguientes condiciones:

- No se cuenta con un esquema para sus datos XML.
- Se cuenta con esquemas pero no desea que el servidor valide los datos. Este suele ser el caso cuando una aplicación realiza validación del lado del cliente antes de almacenar los datos al servidor, o almacena de manera temporal datos XML inválidos de acuerdo con el esquema, o utiliza componentes de esquema no soportados por el servidor (por ejemplo, key/kevref).

Microsoft recomienda el uso de 'XML con tipo' bajo las siguientes condiciones:

- Se cuenta con esquemas para los datos XML y desea que el servidor valide sus datos XML de acuerdo a sus esquemas XML.
- Se quieren aprovechar las mejoras de almacenamiento y consultas inherentes a este tipo de datos.
- Se quieren aprovechar las mejoras implementadas para este tipo de datos en lo referente a la compilación de las consultas.

- **DB2 XML Extender** (http://www.ibm.com)

Esta herramienta de DB2-UDB implementa un 'método' denominado 'XML Columns' que permite almacenar documentos XML de manera íntegra en DB2. Los documentos se insertan en columnas de un tipo XML sobre las que se pueden realizar búsquedas, actualizaciones...

XML Extender diferencia entre varios tipos de datos que pueden albergar contenidos XML:

- XMLVARCHAR (almacena documentos de 3 K como tamaño máximo).
- XMLCLOB (permite almacenar documentos de hasta 2 GB de tamaño).
- XMLFILE (almacena el nombre del fichero, no su contenido, permaneciendo el documento en el file system. Válido para documentos almacenados fuera de la BDD).

Todos los documentos que se almacenen en una determinada columna de tipo XML deben ajustarse a un mismo esquema XML.

A tener en cuenta que los espacios en blanco son respetados, algo importante para el xml, así como el orden de los elementos, comentarios, etc.

La indexación de este tipo de datos se puede llevar a cabo mediante el uso de lo que se conoce como "side tables", que son una especie tablas satélite que almacenan partes concretas del documento (elementos y atributos) que serán consultados con frecuencia. Contiene una clave foránea que apunta a la tabla y columna que contiene el documento en formato integro. Esto permite agilizar tanto las consultas sobre determinados valores como las que recuperan todo el documento. A su vez, cuando el documento almacenado en la columna xml es actualizado, los valores de estas tablas auxiliares también lo serán automáticamente.

Para especificar que valores del documento son los que tienen que ser extraídos y almacenados en este tipo de tablas se debe crear un fichero tipo DAD, que a su vez es otro archivo xml, aunque con una estructura diferente al que utilizado por las XML collections.

El uso de estas tablas es la manera más rápida para recuperar la información almacenada en una columna xml y por tanto se aconseja su utilización.

XML Extender proporciona toda una serie de funciones definidas que permiten:

- Importación de documentos del file system a la BD
- Realización de búsquedas
- Lanzar actualizaciones
- Exportación de datos de la BDD al file system

Este tipo de almacenamiento permite albergar intactos los documentos XML/XBRL, preservando su integridad, siendo especialmente eficaz cuando los datos de los documentos son consultados con asiduidad y raramente actualizados.

Este sistema se muestra también especialmente eficaz en el caso de que se trabaje sobre documentos de gran extensión y se pretenda realizar búsquedas a la vez que se quiera preservar la integridad de los mismos intacta.

Este sistema permite almacenar el documento de manera externa a la BDD (en un sistema de ficheros), bien sea local o remoto, y usar DB2 para las operaciones de mantenimiento y consulta.

- **Oracle XML DB** (http://www.oracle.com)

A partir de un nuevo tipo de datos específico para XML (XMLType) esta BDD permite almacenar contenidos XML en una columna de una tabla a través de un objeto CLOB que representa al documento como una instancia.

Entre las características más destacadas, cabe señalar:

- Proporciona métodos que permiten operaciones comunes como validaciones de esquemas.
- Puede ser usado como cualquier otro tipo de dato en una tabla relacional.

- El tipo XMLType proporciona constructores que permiten que sea creado a partir de un tipo VARCHAR o CLOB, y proporciona un número de métodos específicos XML para operar con los objetos XMLType.
- Se preserva el estándar DOM a la hora de fragmentar y almacenar documentos. Además se conservan los espacios en blanco.

4.2.1.3 Almacenamiento en Bases de Datos XML Nativas

Introducción y Antecedentes:

Como se ha visto en los apartados anteriores, las BDD Relacionales suponen una posibilidad para el almacenamiento de datos XML, sin embargo, no están diseñadas específicamente para almacenar estructuras de tipo jerárquico como son los documentos XML:

- Las BDD relacionales tienen una estructura regular frente al carácter heterogéneo de los documentos XML.
- Los documentos XML suelen contener muchos niveles de anidamiento mientras que los datos relacionales son "planos".
- Los documentos XML tienen un orden intrínseco mientras que los datos relacionales son 'no ordenados'.
- Los datos relacionales son generalmente "densos" (cada columna tiene un valor) mientras que los datos XML son "dispersos" pueden representar la carencia de información mediante la ausencia del elemento.

Una alternativa a los Sistemas Relacionales, son los Sistemas de Bases de Datos Orientados a Objetos (SGBDOO). Los SGBDOO soportan un modelo de objetos puro, en la medida en que no están basados en extensiones de otros modelos más clásicos como el relacional. Estos sistemas ofrecen características que los hacen especialmente interesantes de cara al almacenamiento de información XML:

- Considera toda la información del documento como objetos de clases predefinidas interconectadas mediante enlaces que preservan la estructura del documento XML.
- Emplean la aproximación DOM y permite tratar documentos que están bien-formados.
- La opción de una SGBDOO, es empleada por varias bases de datos nativas, pero los mecanismos de indexación, optimización, procesamiento de consultas, etc. son las del propio SGBDOO, y por lo general, no son específicos para el modelo XML.

Como ejemplos de SGBDOO existentes en el mercado:

Poet (http://www.poet.com)

Jasmine (http://www.cai.com)

ObjectStore (http://www.odi.com)

GemStone (http://www.gemstone.com)

BBDD XML Nativas:

La organización XML:DB Initiative for XML Databases (http://www.xmldb.org) describe una base de datos XML Nativa como un: "modelo lógico para documentos XML que almacena y recupera documentos de acuerdo a dicho modelo".

A diferencia de las bases de datos relacionales cuya operatividad gira alrededor de los 'datos atómicos', las base de datos nativas en XML carecen de campos, no centrándose en el almacenamiento de datos atómicos sino en el de documentos (XML).

Este tipo de BDD almacenan información en formato XML sirviéndose de unos repositorios que podríamos catalogar de 'tipo XML' como son DOM o Infoset. En estos repositorios se almacenan también los índices generados y asociados a cada documento XML.
Algunas de las características que distinguen a las BBDD XML nativas son:

- Emplear como unidad lógica de almacenamiento el documento XML.
- Preservar el orden del documento, las instrucciones de procesamiento, los comentarios, las secciones CDATA y las entidades, es decir, responde a un esquema (DTD, XML Schema).
- La mayoría de las BBDD XML nativas soportan uno o más lenguajes de consulta. Uno de los más populares es Xquery.
- Validación de los documentos.
- Almacenamiento de documentos en colecciones. Las colecciones juegan en las bases de datos nativas el papel de las tablas en las BDD relacionales. Los documentos se suelen agrupar, en función de la información que contienen, en colecciones que a su vez pueden contener otras colecciones.
- Indexación XML. Permiten la creación de índices que aceleren las consultas más habituales.
- Creación de identificadores únicos. A cada documento XML se le asocia un identificador único por el que será reconocido dentro del repositorio.
- Soportar API's de programación (SAX, DOM, JDOM,...)
- No tienen ningún modelo de almacenamiento físico subyacente concreto.

Como se ha señalado, las BBDD XML nativas existentes en la actualidad no poseen un modelo de almacenamiento físico subyacente concreto, es decir, son construidas sobre diversas estructuras de BBDD: relacionales, jerárquicas, orientadas a objetos o bien mediante formatos de almacenamiento propietarios.

Ventajas:

- Se trata de sistemas de almacenamiento diseñados específicamente para almacenar contenidos XML, con lo que ello supone:

  - No es necesario la conversión de la información a estructuras relacionales u otro tipo ajeno a las estructuras propiamente XML.
  - Implementa herramientas de gestión y consulta de contenidos expresamente diseñadas para este tipo de información, con los consiguientes beneficios en lo que respecta a su operatividad, eficacia y rendimiento.
  - Implementan herramientas específicas para este tipo de información, como puede ser los módulos de validación contra esquemas, DTDs o taxonomías.
  - Implementan una indexación específica y adecuada a los contenidos XML, por lo que su eficacia en la localización, tratamiento y recuperación de la información es mayor.

Productos / Herramientas:

- **Tamino XML Server (versión 4.2)** (http://www.softwareag.com)

Atendiendo a la denominación del fabricante, podríamos denominar este producto más que como una simple base de datos XML nativa, como un servidor de información íntegramente XML, capaz de almacenar, gestionar, publicar y servir de plataforma de intercambio de documentos XML.

Este servidor de información, está basado en estándares abiertos.

Entre las características más destacables de este producto cabe destacar:

- Alto Rendimiento y disponibilidad ya que su "motor XML" es multi-proceso, permitiendo grandes volúmenes de peticiones y usuarios. Admite acceso mediante URL, soportando todos los estándares de acceso Web (HTTP, ISAPI, NSAPI, Apache API).
- Multiplataforma. Tamino puede invocarse desde cualquier cliente que tenga una máquina virtual Java instalada. También puede invocarse desde componentes basados en ActiveX o C++.
- Soporta Xquery como lenguaje de consulta y además un lenguaje definido por el propio fabricante, basado en Xpath y denominado Tamino X-Query.
- Cabe destacar su integración con BBDD relacionales o Adabas de forma que este servidor de información disponibiliza los contenidos de dichas bases de datos para aplicaciones basadas en él.
- Alta escalabilidad.

En lo referente a su arquitectura, el servidor está cimentado en cinco módulos especializados:

- X-Machine: es el núcleo de alto rendimiento encargado del almacenamiento de objetos XML. "X-Machine" administra todas las estructuras de datos bien estén en formato XML o SQL, imágenes, archivos de audio, vídeo o datos recuperados a través de "X-Node".
- X-Node: implementa el acceso a fuentes de información con estructuras de datos tradicionales como BBDD relacionales o Adabas. "X-Node" proyecta estos datos en estructuras XML.
- SQL Engine permite el almacenamiento de datos SQL en Tamino. Esos datos pueden ser bien parte de documentos XML o utilizados por aplicaciones que sólo trabajan con datos SQL, pero que se ejecutan en el mismo servidor que Tamino.
- X-Manager: permite al administrador de Tamino proyectar todos los objetos XML a estructuras internas y administrar el sistema completo.
- Data Map: contiene 'meta-datos' como los DTDs, hojas de estilos, esquemas relacionales, etc. "Data Map" también determina como serán proyectados los documentos XML en las estructuras físicas de datos.

Otras de las características destacables de este producto son:

- Soporte de múltiples APIs y herramientas de desarrollo de terceros, al poder utilizar J2EE o '.NET'.

- Soporte de los estándares abiertos propuestos por W3C (http, xml, Xquery, Xpath, XML Schema, Namespaces,...), aportando flexibilidad en el diseño y desarrollo de aplicaciones y productos.
- Múltiples plataformas soportadas: Windows, Solaris, Linux, Unix System V.

- Semansys Tamino XBRL Suite (http://www.semansys.com – http://www.softwareag.com)

Aunque no se trata de un nuevo sistema de almacenamiento, ya que como tal utiliza el producto anteriormente reseñado (Tamino XML Server), merece la pena destacarlo en este documento, ya que este 'suite' está pensado y diseñado expresamente para el tratamiento, gestión, intercambio y almacenamiento de contenidos XBRL.

Este producto, surge a partir de una reciente alianza entre Software AG y Semansys Technologies con el objetivo de ofrecer al mercado un paquete (Suite) cerrado y centrado en XBRL. Semansys Tamino XBRL Suite, integra los productos para el procesado, validación y manipulación de XBRL de Semansys, apoyándose en la BDD XML nativa, Tamino XML Server. Este producto está constituido por varios módulos especializados:

- *'Semansys XBRL Composer'* es utilizado para recopilar la información de diferentes fuentes y formatos, y transformarlo en XBRL.
- *'Semansys XBRL Integrator'* es el gestor de datos para usar esa información guardada en XBRL.
- *'Semansys XBRL Reader'* permite acceder a los documentos XBRL así como su manipulación.
- *'Software AG Tamino'* es utilizada como BDD XML nativa.

- Xindice (versión 1.1.b4) - (http://xml.apache.org)

Xindice es la continuación de un proyecto anterior llamado dbXML Core. El código fue donado por el grupo dbXML (http://www.dbxml.com) a Apache Software Foundation en Diciembre de 2001.

Entre las características más destacables de este producto:

- Utiliza Xpath como lenguaje de consulta y XML:DB XUpdate como lenguaje de actualización en la BDD.
- Proporciona una implementación de XML:DB API para la invocación desde desarrollos Java.
- Mediante XML-RPC plugin es posible acceder a Xindice desde otros entornos o lenguajes.
- Los estándares de XML que soporta Xindice son los recomendados por W3C.
- Es un producto muy ventajoso desde el punto de vista económico ya que se sustenta sobre el tipo de licencia: Apache Software License.

- eXist (versión 1.0.b2) - (http://exist.sourceforge.net/)

Se trata de una versión 'Open Source' de una Base de Datos XML nativa. Entre las características más destacables de este producto:

- Soporta Xquery 1.0 como lenguaje de consulta, indexación y procesado de la información. También soporta Xupdate pero de una forma limitada.
- Implementa un API para la invocación desde aplicaciones Java.
- Implementa extensiones para el soporte de HTTP, manipulación y transformaciones XSL.
- Incorpora proyectos "extra" para la explotación de los datos que almacene, como puede ser 'Cocoon'.
- Incorpora un framework del lado del servidor para la creación de aplicaciones web, basadas totalmente en XML.
- Desde un punto de vista económico, resulta un producto muy ventajoso al sustentarse sobre licencias de tipo: GNU LGPL .

BBDD NATIVAS XML **LISTADO DE PRODUCTOS / FABRICANTES:**

Nombre	Fabricante	Tipo Licencia	Sistema Almac.	APIs
Centor Interaction Server	Centor Software Corp. http://www.centor.com	Comercial	Propietario	C++, Java
DXML	http://www.dbxml.com/	GNU LGPL	Propietario	Java, Javascript
GoXML DB	The Xenos GoXML™ Integration Solution http://www.xmlglobal.com/solutions/prod_goxml_overview.htm	Comercial	Propietario	Java
Infonyte DB	Infonyte http://www.infonyte.com/en/products.html	Comercial	Propietario	Java
Ipedo XML Database	Ipedo http://www.ipedo.com/html/ipedo_xml_database.html	Comercial	Propietario	Java
Tamino XML Server	Software AG http://www2.softwareag.com/Corporate/products/tamino/default.asp	Comercial	Propietario	Java
TeamXML/ TeamSite	Interwoven http://www.interwoven.com/products/features/xml/teamxml.html	Comercial	Propietario	Java
Virtuoso	OpenLink http://www.openlinksw.com/virtuoso/	Comercial	Propietario	
Xindice	Apache Software Foundation http://xml.apache.org/xindice/	Apache Software License	Propietario	Java
XDBM	http://sourceforge.net/projects/xdbm/	GNU LGPL	Propietario	
DBDOM	http://sourceforge.net/projects/dbdom	GNU LGPL	Relacional	Java
EXist	http://exist.sourceforge.net/	GNU LGPL	Relacional	Java
XDB	MindSuite http://xdb.wiredminds.com/	Comercial	Relacional	C++, Java
Birdstep RDM XML	Birdstep Technology http://www.birdstep.com/database_technology/rdm_server.php3	Comercial	Orientado Objetos	C, C++, Java
Ozone	http://www.ozone-db.org/frames/home/what.html	Open Source, Ozone Library License	Orientado Objetos	Java

4.2.2 Lenguajes de consulta XML

4.2.2.1 Xquery - Xpath (http://www.w3.org/XML/Query)

Xquery es un lenguaje que se encuentra aún en fase de desarrollo, aunque el estado de ésta se puede considerar avanzado, siendo el objetivo del grupo de trabajo finalizar una versión completa en el año 2005. Las características de la última versión de este lenguaje se recogen en el informe (working draft) del 29 Octubre 2004 que se puede encontrar en la siguiente dirección: http://www.w3.org/TR/Xquery.

Xquery se presenta como un lenguaje funcional, en vez de ejecutar comandos como lo haría un lenguaje procedural, cada consulta es una expresión a ser evaluada. Las expresiones se pueden combinar para hallar nuevas expresiones.

Xquery se basa en los siguientes lenguajes:

- Xpath y XQL: De ellos adopta la sintaxis para navegar por documentos jerárquicos.
- SQL: (lenguaje relacional). Las cláusulas FLWR y operadores como el join.
- XML-QL: De este lenguaje adopta la idea de asignar variables para crear nuevas estructuras.

Xquery hace un uso intensivo de Xpath (un lenguaje utilizado para seleccionar porciones de XML); de hecho algunos ven a Xquery como un superconjunto de Xpath. La nueva versión de Xpath (2.0) está siendo desarrollada por el mismo grupo de trabajo y de manera conjunta a la versión 1.0 de Xquery.

En la página del Consorcio W3C aparece una lista de fabricantes de software que ya implementan soluciones basadas en Xquery. Algunos de estos fabricantes:

- BEA's Liquid Data
- Berkeley Lab's Nux, an open source XQuery extension to XOM.
- Blackpearl's Blackpearl 4 platform, with an embedded XQuery engine
- Bluestream Database Software Corp.'s XStreamDB
- Cerisent's XQE
- Cognetic Systems's XQuantum
- GNU's Qexo (Kawa-Query)

  Compiles XQuery on-the-fly to Java bytecodes. Based on and part of the Kawa framework. An online sandbox is available too. Open-source.
- Ipedo's XML Database v3.0
- Microsoft's SQL Server 2005 Express, with XQuery support
- Neocore's XML management system (XMS): http://www.neocore.com/products/products.htm
- Oracle's Xquery Technology - Preview
- Renmin University of China's OrientX
- Sleepycat's Berkeley DB XML 2.0, an embedded native XML database with support for XQuery 1.0 (July 2004 draft), implemented in C++, with interfaces for Java, Python, Perl and PHP. Open source.
- Software AG's - Tamino XML Server
- Sonic Software's - Stylus Studio 5.0 (XQuery, XML Schema and XSLT IDE)
- Sourceforge's eXist. Open-source.
- Sourceforge's XQEngine. Open-source.
- Sourceforge's XQuench. Open-source.
- Worcester Polytechnic Institute's RainbowCore. Java.

4.3 Envío y recepción de informes XBRL

Dado que XBRL es un modelo de XML, el intercambio de ficheros se puede realizar utilizando distintas alternativas. A continuación se enumeran y describen brevemente.

Los servicios web permiten que las aplicaciones compartan información y que además invoquen funciones de otras aplicaciones independientemente de cómo éstas se hayan creado sea cuál sea el sistema operativo o la plataforma en que se ejecutan y cuáles los dispositivos utilizados para obtener acceso a ellas. Existen muchos servicios web implementados con php (lenguaje de programación no marcado que se ejecuta en el servidor.

Los protocolos que soportan los servicios web se comunican normalmente por el puerto 80, y basándose en HTTP, métodos GET y PUT. Esto hace que podamos acceder a ellos al igual que lo hacemos en una página web. La diferencia entre una página web y un Servicio Web, es que la página la visita cualquier individuo interesado, mientras que el servicio sólo lo visitan programas que lo requieren.

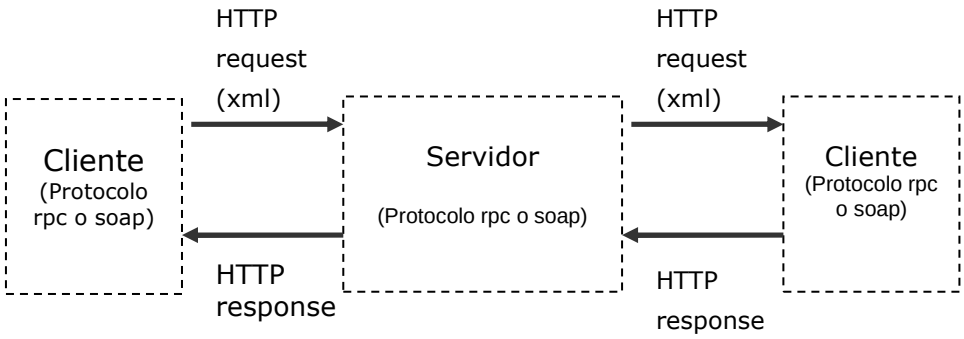
Se puede realizar un servicio web sencillo en el sistema, pero posiblemente éste no cumplirá con estándares de comunicación, es por eso que debemos de entender que para realizar una correcta función de nuestros servicios web es necesario estandarizarlos por medio de protocolos.

Existen dos grandes tendencias: XML-RPC protocolo de llamada de procedimientos remotos (RPC: Remote Procedure Call) y SOAP (Simple Object Access Protocol, Protocolo de acceso a objetos simple). A la hora de programar un servicio web, hay que decidir qué protocolo usar, porque un protocolo es incompatible con el otro. De modo que si programamos el servicio web con XML-RPC, no podremos invocarlo desde un lenguaje de programación que trabaje con SOAP, como por ejemplo .Net de Microsoft.

La diferencia entre SOAP y XML-RPC es su complejidad. XML-RPC está diseñado para ser sencillo. SOAP por el contrario está creado con idea de dar un soporte completo y minucioso de todo tipo de servicios web.

Un Servicio Web se invoca enviando una solicitud web en lenguaje XML y la respuesta es una página web cuyo código es XML. Para efectuar dicha invocación, se realiza directamente desde el código de programación de la aplicación cliente y se utilizan rutinas del protocolo cliente (SOAP o XML-RPC) que traducen a XML nuestra llamada, invocan por HTTP el Servicio Web, descargan el XML resultante, y lo procesan convirtiéndolo en variables resultado.

La siguiente figura muestra de manera simplificada el proceso de envío de un fichero xml. Como puede observarse cualquier cliente realiza la invocación del servicio web alojado en el servidor, enviando el fichero en formato xml. Dicho fichero es tratado en el servidor y devuelto en formato xml al cliente. Una vez el cliente esta en posesión del fichero de resultado deberá proceder a realizar las acciones de recepción y tratamiento definidas en posteriores apartados.



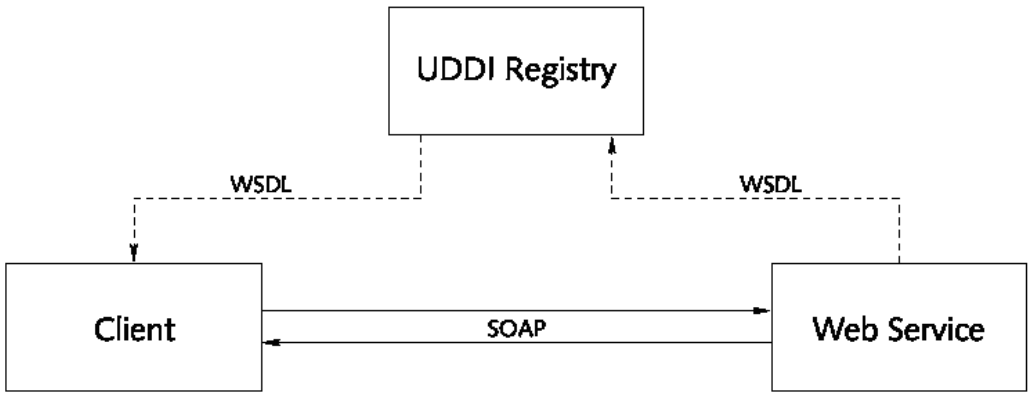
Si entramos un poco más en detalle diremos que en el caso de enviar un mensaje de XML-RPC, éste es una petición del HTTP-POST. El cuerpo del mismo está en XML, un procedimiento es ejecutado en el servidor y el valor que devuelve está en formato XML.

En el caso de SOAP, éste es un protocolo basado en XML que consiste de tres partes: la primera define cuál es el mensaje y cómo procesarlo, la segunda es un sistema de reglas de codificación para expresar tipos de datos definidos y una tercera parte para representar respuestas de llamadas por parte de procedimientos remotos.

SOAP a diferencia de XML-RPC, incluye una infraestructura a su alrededor. No es un mero protocolo de comunicación entre ordenadores, sino que además se rodea de términos como WSDL y UDDI. Veamos que significan exactamente.

- **WSDL** (Web Services Description Language) es un fichero en formato XML que indica al ordenador que lo consulta, qué servicios dispone en su sede. Además de dar una referencia precisa sobre ellos, para poder invocarlos usando los parámetros adecuados.
- **UDDI** (Universal Description, Discovery, and Integration). UDDI es un Servicio Web en línea que se puede utilizar desde las aplicaciones para descubrir de forma dinámica otros servicios en línea, todos ellos perfectamente integrados en una interfaz XML simple.

En la siguiente figura se muestra la relación de las tecnologías:



La especificación UDDI, junto con Extensible Markup Language (XML), Simple Object Access Protocol (SOAP) y Web Services Description Language (WSDL), están ganando un amplio soporte en el marco de trabajo de los servicios Web.

Esta tecnología puede ser muy útil en ciertos casos concretos de programación, además de ser una tecnología flexible. Pero por el contrario, a pesar de ser una tecnología que intenta repartir la carga de trabajo por Internet, sobre el servidor del Servicio Web, no deja de utilizar el paradigma cliente-servidor (filosofía de sistemas centralizados). Esto nos lleva a que si por ejemplo, si se viniera abajo el registro UDDI, aunque el sistema este replicado ocasionaría un cuello de botella en los servidores.

Dicho esto, cabe mencionar las siguientes ventajas e inconvenientes:

Ventajas:

- Aportan interoperabilidad entre aplicaciones de software independientemente de sus propiedades o de las plataformas sobre las que se instalen.
- Los servicios Web fomentan los estándares y protocolos basados en texto, que hacen más fácil acceder a su contenido y entender su funcionamiento.
- Al apoyarse en HTTP, los servicios Web pueden aprovecharse de los sistemas de seguridad firewall sin necesidad de cambiar las reglas de filtrado.

Inconvenientes:

- Para realizar transacciones no pueden compararse en su grado de desarrollo con los estándares abiertos de programación distribuida.
- Su rendimiento es bajo si se compara con otros modelos de programación distribuida, tales como RMI o Corba. Es uno de los inconvenientes derivados de adoptar un formato basado en texto.
- Al apoyarse en HTTP, pueden esquivar medidas de seguridad basadas en firewall cuyas reglas tratan de bloquear o auditar la comunicación entre programas a ambos lados de la barrera.

Otra opción es la utilización del correo electrónico como herramienta de intercambio. Esta herramienta es la aplicación Internet que más se usa en el entorno empresarial debido a su rapidez, facilidad de uso y comodidad.

Todo sistema de correo electrónico debería garantizar, al menos, que los mensajes lleguen a su destinatario (en ambas direcciones) y que pasen por un punto de recepción-control que gestione el apartado de seguridad (antivirus, firewall). Además debería almacenarse una copia se seguridad de cada mensaje recibido en un depósito de gestión centralizado que permita la recuperación de los documentos.

Podemos decir, que este sistema tiene muchas ventajas entre las que se encuentran las siguientes:

- Es una herramienta muy extendida
- Fácil de utilizar y de bajo coste.
- No requiere una infraestructura excesivamente compleja.
- Es un sistema asíncrono, por lo que no se requiere la presencia simultánea de los sistemas comunicantes.

Por el contrario, este sistema suele ser bastante inseguro con lo que las políticas de seguridad deben hacer especial hincapié para evitar la manipulación no desea de la información. Esto es, se deben implementar métodos que aseguren la información como la

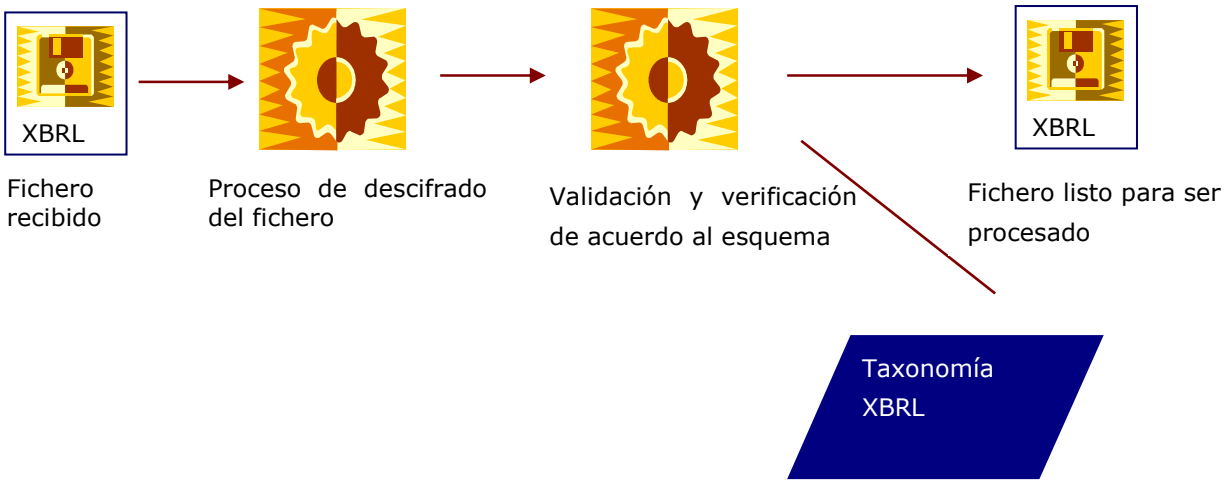
utilización de encriptación y/o firmas digitales que cumplan con los estándares (véase apartado de seguridad). También cabe la posibilidad sería configurar un servidor seguro de correo. Estos servidores son totalmente funcionales y de alto rendimiento además permiten usar un completo abanico de modernas tecnologías y protocolos que mejoran su eficiencia, robustez, flexibilidad y seguridad.

Una tercera posibilidad es el uso de un cliente FTP. Éste es un protocolo de transferencia de ficheros muy utilizado y fiable, síncrono, más seguro que el correo electrónico ya que necesitamos los permisos adecuados para realizar las acciones oportunas tales como visualizar, descargar, copiar, etc. y evita la mayoría de errores producidos por la codificación de los distintos protocolos derivados de los programas de correo. Además, existen muchos clientes ftp con una interfaz visual que facilita el uso de los mismos.

4.4 Recepción de información y tratamiento

El proceso de recepción de ficheros XBRL conlleva una serie de acciones anteriores a su tratamiento y dado que la información tratada es de carácter sensible deberían tomarse todas las medidas de seguridad razonables para evitar la manipulación no autorizada de los datos (sería recomendable que los documentos recibidos vinieran firmados digitalmente por la empresa, auditor o entidad certificadora).

En el gráfico que se muestra a continuación se indican de manera simplificada las acciones a realizar una vez se ha recibido el fichero xbrl.



El fichero recibido deberá ir cifrado en base a las especificaciones de seguridad descritas en el apartado correspondiente a este tema. Por lo tanto, el primer paso a realizar sería el proceso de descifrado que permita obtener un fichero legible para ser tratado. A continuación, se procederá a realizar la validación del documento. Para llevar a cabo este proceso se necesita disponer del esquema xml que describa la validez de las etiquetas definidas dentro del documento y los documentos xlink para validar la lógica del mismo. A esto hay que añadirle una taxonomía definida o conjunto de archivos XML que describe los

elementos lineales en un informe financiero. (Estos conceptos se encuentran definidos más detalladamente en apartados anteriores en este documento).

Una vez que el fichero ha sido validado de acuerdo a las especificaciones se puede proceder a su almacenamiento y/o tratamiento.

El almacenamiento puede realizarse en una base de datos relacional o en algún directorio dentro del sistema. Quizás nos interese almacenar el propio fichero una vez recibido o en cambio, primero proceder a su tratamiento y una vez transformado en otro fichero de resultado almacenarlo como un registro de un tabla en base de datos. Para obtener información a cerca del proceso de almacenamiento véase el apartado 4.5. de este mismo documento que lo describe en detalle.

Por otra parte, para realizar el tratamiento del fichero recibido y dado que XBRL se basa en los conceptos de XML se podrá utilizar la especificación XSLT (Xml Stylesheets Transformation Language). XSLT es un lenguaje para la transformación de ficheros xml, que se apoya en XPATH, que es un lenguaje que permite escribir expresiones para la búsqueda de nodos dentro del árbol XML.Dicha transformación puede generar ficheros tanto en el propio formato xml como html o txt. Además, se permite la generación de PDFs o rtf utilizando el lenguaje de especificación de estilos (xsl-fo).

En el caso de estar utilizando tecnología JAVA disponemos además de dos APIs que nos permiten tratar un fichero xml. A continuación se exponen brevemente las características fundamentales de cada una de ellas

- **SAX** (Simple API for XML). Consta de una serie de clases que nos permiten trabajar con un documento XML desde un programa escrito en Java, pudiendo acceder a los datos, comprobar si está bien formado y si tiene un formato válido.
    - La principal característica es que el documento se lee secuencialmente de principio a fin, sin cargar todo el documento en memoria.
    - Ventaja: la eficiencia en cuanto al tiempo y la memoria empleados en el análisis.
    - Desventaja: no disponemos de la estructura en árbol de los documentos
    - Se necesita un analizador SAX.

- **DOM** (Modelo de Objetos de Documento). Consta de una serie de clases que permiten trabajar con documentos XML desde programas escritos en Java.
    - La principal característica de DOM es que el documento con el que se trabaja se carga entero en memoria, en una estructura de árbol.
    - Ventaja: al disponer de la estructura del documento, permite acceder a los datos en función de la jerarquía de elementos, así como modificar el contenido de los documentos e incluso crearlos desde cero.
    - Desventaja: el coste en tiempo
    - Se necesita un analizador DOM.

Del proceso de transformación del fichero origen obtendremos un fichero de resultado con el formato requerido según cada caso. El almacenamiento de dicho fichero se realizará en base a las políticas de almacenamiento descritas en el apartado 4.5.

4.5 Seguridad

4.5.1 Seguridad en el aplicativo:

Restricción de acceso: Se debería dar acceso al aplicativo sólo a los usuarios con permisos para acceder a dicha información. Se deberían considerar las siguientes medidas y controles para dar soporte a los requisitos de restricción de accesos:

- Establecer menús para controlar los accesos a las funciones del sistema del aplicativo.
- Restringir el conocimiento de información y de funciones del aplicativo a cuyo acceso los usuarios no estén autorizados.
- Controlar los derechos de acceso de los usuarios.
- Asegurarse que las salidas del aplicativo que procesen la generación de las instancias, sólo contienen esta información y que se envían únicamente a los terminales o lugares autorizados.

Identificación y Autenticación: En el apartado de la generación de instancias deberá existir un mecanismo que permita la identificación de forma inequívoca y personalizada de todo aquel usuario que intente acceder a este apartado. Para ello se podrá tener una relación de usuarios con acceso a la aplicación, clasificados en función de su perfil, pudiendo ser uno de los perfiles el acceso a dicho apartado.

Además se debería limitar la posibilidad de intentar reiteradamente el acceso no autorizado a este apartado.

Distintos procedimientos de autenticación pueden usarse para materializar la identidad pedida a un usuario. Las contraseñas son una forma común de conseguir la identificación y la autenticación de usuario. Esto mismo también se puede conseguir por medios criptográficos y protocolos de autenticación.

También puede conseguirse Identificación y Autenticación con objetos como tarjetas inteligentes, minicalculadoras con claves almacenables o bien con tecnologías biométricas que utilizan características o atributos únicos de un individuo. Una combinación de tecnologías y mecanismos, relacionados mediante el establecimiento de un enlace seguro, pueden proporcionar una autenticación reforzada o más robusta.

Seguimiento de accesos al aplicativo: Cuando se genere la instancia se deberá de guardar un registro donde se debería identificar el usuario que ha realizado la instancia, la fecha y hora en que se realizó dicha instancia, identificación del terminal donde se ha generado la instancia si eso es posible, registro de los intentos aceptados y rechazados de acceso al aplicativo y si se ha generado de manera satisfactoria o no la misma.

Ubicación del Aplicativo: El aplicativo de generación de instancias debería estar ubicado en un Servidor con acceso restringido.

Además únicamente el personal autorizado tendrá acceso a las instalaciones donde esté ubicado el Servidor.

Limitación del tiempo de conexión: Las restricciones en los tiempos de conexión ofrecen seguridad adicional para las aplicaciones de alto riego. Limitar el periodo de tiempo durante el que se aceptan conexiones desde un terminal reduce la "ventana" de oportunidad para accesos no autorizados.

Backup: Se deberían de realizar regularmente copias de Seguridad del Servidor donde esté ubicado el Aplicativo.

Plan De Contingencias: El Plan de Contingencias implica un análisis de los posibles riesgos a los cuales pueden estar expuestos nuestro aplicativo y la información contenida en los diversos medios de almacenamiento, por lo que se debería realizar un análisis de los riesgos, cómo reducir su posibilidad de ocurrencia y los procedimientos a seguir en caso que se presentara el problema.

Pese a todas las medidas de seguridad puede ocurrir un desastre, por tanto es necesario que el Plan de Contingencias incluya un Plan de Recuperación de Desastres, el cual tendrá como objetivo, restaurar el Servicio de Cómputo en forma rápida, eficiente y con el menor costo y pérdidas posibles.

4.5.2 Seguridad en la instancia generada:

Acceso lógico Restringido: Se debería dar acceso al directorio donde se guarden las instancias sólo a los usuarios con permisos para acceder a dicha información. Se deberían considerar las siguientes medidas y controles para dar soporte a los requisitos de restricción de accesos:

- Restringir el conocimiento de esta información a los usuarios autorizados.
- Controlar los derechos de acceso de los usuarios.

Deberá existir un mecanismo que permita la identificación de forma inequívoca y personalizada de todo aquel usuario que intente acceder a este directorio. Para ello se podrá tener una relación de usuarios con acceso al directorio en función de su perfil.
Además se debería limitar la posibilidad de intentar reiteradamente el acceso no autorizado a este directorio.

Distintos procedimientos de autenticación pueden usarse para materializar la identidad pedida a un usuario. Las contraseñas son una forma común de conseguir la identificación y la autenticación de usuario. Esto mismo también se puede conseguir por medios criptográficos y protocolos de autenticación.

Backup De La Carpeta: Se deberían hacer regularmente copias de seguridad de las instancias que se han generado. Además se debería de comprobar que las copias de seguridad realizadas han finalizado correctamente.

Seguridad Física del Equipo: Se debería de tener un control sobre la Máquina física donde se han guardado las instancias generadas. Para ello se debería de usar perímetros de seguridad para proteger este recurso. El acceso a la Sala donde se encuentre el recurso debería ser controlado y restringido únicamente al personal autorizado. Se deberían usar controles de

autenticación, por ejemplo, tarjetas con número de identificación personal, para autorizar y validar el acceso. Se debería de mantener un rastro auditable de todos los accesos, con las debidas medidas de seguridad.

Se deberían de revisar y actualizar regularmente los derechos de accesos al lugar donde este ubicado dicho recurso.

Además el recurso debería estar físicamente protegido de las amenazas y riesgos del entorno para protegerlo contra pérdidas o daños. También se debería de considerar su instalación y disponibilidad. Deberían requerirse medidas o controles especiales contra riesgos de accesos no autorizados y para proteger los sistemas de apoyo, como la alimentación interrumpida o la infraestructura de cableado.

Cifrado de la Instancia: Es una técnica criptográfica que puede utilizarse para proteger la confidencialidad de la información.

El nivel adecuado de protección se debería basar en una evaluación del riesgo y tendrá en cuenta el tipo y calidad del algoritmo de cifrado y la longitud de las claves criptográficas que se usarán.

En la implantación de la política criptográfica se deberán tener en cuenta las regulaciones y restricciones nacionales que se aplican en distintas partes del mundo para el uso de las técnicas criptográficas y el cifrado de las transmisiones internacionales de los datos.

Plan De Contingencias: El Plan de Contingencias implica un análisis de los posibles riesgos a los cuales pueden estar expuestos nuestro directorio y la información contenida en el mismo, por lo que se debería realizar un análisis de los riesgos, cómo reducir su posibilidad de ocurrencia y los procedimientos a seguir en caso que se presentara el problema.

Pese a todas las medidas de seguridad puede ocurrir un desastre, por tanto es necesario que el Plan de Contingencias incluya un Plan de Recuperación de Desastres, el cual tendrá como objetivo, restaurar el Servicio de Cómputo en forma rápida, eficiente y con el menor costo y pérdidas posibles.

4.5.3 Seguridad en la transmisión de las instancias:

Para proceder a tener un canal Seguro es necesario lo siguiente:

- Autenticación en los extremos
- Confidencialidad: Asegurar que la información es accesible solo para aquellos usuarios autorizados.
- Integridad: Garantía de la exactitud y completitud de la información y los métodos de su procesamiento. En la práctica asegurar que la información suministrada en destino es idéntica a la generada en origen.
- Disponibilidad: Asegurar que los usuarios autorizados tienen acceso cuando lo requieran a la información y sus activos asociados.
- No Repudio: Cualidad por la que ninguno de los extremos de una comunicación pueda rechazar la autoría de un envío así como del contenido del mismo.

Mecanismos de seguridad:

Certificado Digital: Fichero con información sobre el titular del mismo que le permite identificarse ante un sistema o aplicación. El Certificado Digital proporciona Autenticación.

Cifrado: Se transforman los datos para que sólo sean inteligibles a los usuarios autorizados.
Firma Digital: Secuencia de datos digitales añadidos a una transmisión / instancia que permiten identificar unívocamente al autor de la misma. Proporciona autenticación e Integridad.

Control de Accesos: No permiten el acceso físico o lógico a la información a usuarios no autorizados.

Mecanismos de Integridad: Información de control añadida en una transmisión que permite detectar si se ha producido alguna modificación o error en el contenido de la información enviada.

Control de encaminamiento: Se utilizan los sistemas de encaminamiento para proteger la información.

La siguiente tabla relaciona los mecanismos con los Servicios de seguridad:

Servicios	Autenticación	Control Accesos	Confidencialidad	Integridad	No repudio	Disponibilidad
Mecanismos						
Certificado Digital	X					
Cifrado	X		X	X		
Firma Digital	X			X	X	
Mecanismos de Integridad				X	X	X
Control de accesos		X				X
Encaminamiento			X			

Criptografía

Los mecanismos cifrado, firma digital, Certificado digital e integridad utilizan criptología para su implementación.

La criptografía es la Ciencia que hace uso de métodos y técnicas matemáticas con el objeto principal de cifrar y/o proteger un mensaje o instancia por medio de un algoritmo, usando una o más claves. Esto da lugar a diferentes tipos de sistemas de cifrado que permiten asegurar estos cuatro aspectos de la seguridad informática: la confidencialidad, la integridad, la disponibilidad y el no repudio de emisor y receptor.

Las técnicas de criptografía se pueden clasificar en dos según el tipo de clave utilizado:

- Criptografía simétrica
- Criptografía de clave pública o asimétrica

Simétrica:

Se caracteriza por usar la misma clave para cifrar y descifrar.

Toda la seguridad está basada en la privacidad de esta clave secreta, llamada simétrica porque es la misma para el emisor y el receptor. Estos sistemas sólo permiten confidencialidad y no autenticación ni firma digital.

Los algoritmos simétricos son más sencillos que los asimétricos, por ese motivo los procesos son más simples y rápidos. Los algoritmos más utilizados son:
DES, TDES, IDEA, RC5, RIJNDAEL.

Asimétrica:
Está basada en utilizar claves distintas para cifrar y descifrar; una de ellas se hace pública y la otra es privada de cada usuario. Así todos los usuarios tienen acceso a las claves públicas, pero únicamente ellos conocen su clave privada. Estos sistemas se pueden utilizar para confidencialidad, autenticación y firma digital.

El principal inconveniente de estos sistemas es la dificultad de implementación y la lentitud de proceso.

La ventaja es que implementan servicios de autenticación y firma, y además no tiene problemas con la distribución de claves: la clave pública puede ser visible por cualquiera y la privada no se transmite nunca.

El algoritmo más utilizado es el *RSA*.

Únicamente para firma digital también se utiliza el algoritmo DSS que ha sido adoptado como estándar por el NIST.

Para distribuir claves simétricas también se utiliza el algoritmo Diffie-Hellman, pero no sirve para confidencialidad, autenticación ni firma digital.

4.5.4 Soluciones para el canal de comunicación:

4.5.4.1 VPNs (Virtual Private Networks)

Una red privada virtual (VPN) es la extensión de una red privada que incluye vínculos de redes compartidas o públicas como Internet. Con una red privada virtual, se puede enviar datos entre dos equipos a través de una red compartida o pública de forma que emula un vínculo privado punto a punto. Esto se consigue encapsulando los datos con un encabezado que proporciona la información de enrutamiento que permite a los datos recorrer la red pública hasta alcanzar su destino como si de un túnel se tratara. Los datos se cifran para asegurar la confidencialidad en la misma parte de la conexión en la que se encapsulan.

Gracias al acceso remoto y a las conexiones enrutadas, una organización puede utilizar conexiones VPN para realizar conexiones a larga distancia o líneas concedidas para conexiones locales o con un Proveedor de servicios de Internet (ISP).
La seguridad en la VPN de acceso remoto se mejora a través de una autenticación segura utilizando el esquema de certificados electrónicos.

Tipos de VPNs:

- **Sistemas Basados en Hardware**: Estos sistemas son routers que se cifran. Son seguros y fáciles de usar. Ofrecen un gran rendimiento, porque no malgastan ciclos de procesador haciendo funcionar un Sistema Operativo. Es hardware dedicado, muy rápido, y de fácil instalación.
- **Sistemas Basados en Cortafuegos**: Éstos se implementan con software de cortafuegos (Firewall). Tienen las ventajas de los mecanismos de seguridad que utilizan los cortafuegos, incluyendo el acceso restringido a la red interna. También realizan la traducción de direcciones (NAT). Estos satisfacen los requerimientos de autenticación fuerte.
- **Sistemas Basados en Software**: Estos sistemas son ideales para las situaciones donde los dos puntos de conexión de la VPN no están controlados por la misma organización, o cuando los diferentes cortafuegos o routers no son implementados por la misma organización. Con este tipo, el tráfico puede ser enviado a través de un túnel, en función de las direcciones o protocolos, en cambio en los VPNs por hardware, todo el tráfico es enrutado por el túnel. Podemos hacer un enrutamiento inteligente de una manera mucho más fácil.

Las dos tecnologías más utilizadas para crear VPN's, en realidad son diferentes protocolos o conjuntos de protocolos, son PPTP y L2TP.

PPTP: Point to Point Tunneling Protocol

PPTP es un protocolo desarrollado por Microsoft y disponible en todas las plataformas Windows. Es sencillo y fácil de implementar pero ofrece menor seguridad que L2TP.
Hay tres áreas en la seguridad PPTP que son la autenticación, encriptación de datos y el filtrado de paquetes PPTP.

- **Autenticación**
- **Encriptación de datos**: Los paquetes de red son encriptados en la fuente, viajan a través del túnel y son desencriptados en el destino. Como todos los datos en una conexión fluyen dentro del túnel, los datos son invisibles al resto del mundo. La encriptación de datos dentro del túnel da un nivel adicional de seguridad.
- **Filtrado de Paquetes PPTP**: Esta opción incrementa el rendimiento y fiabilidad de la seguridad de red si está activada en el servidor PPTP.

L2TP: Layer Two Tunneling Protocol

Se trata de un estándar abierto y disponible en la mayoría de plataformas Windows, Linux, Mac, etc. Se implementa sobre IPSec y proporciona altos niveles de seguridad. Se pueden usar certificados de seguridad de clave pública para cifrar los datos y garantizar la identidad de los usuarios de la VPN.

Comparativa entre PPTP y L2TP

- Con PPTP, el cifrado de datos comienza después de que la conexión se procese (y, por supuesto, después de la autentificación PPP). Con L2TP/IPSec, el cifrado de datos empieza antes de la conexión PPP negociando una asociación de seguridad IPSec.
- Las conexiones PPTP usan MPPE, un método de cifrado basado en el algoritmo de encriptación Rivest-Shamir-Aldeman (RSA) RC-4, y usa llaves de 40, 56 o 128 bits. Las conexiones L2TP/IPSec usan Data Encryption Standard (DES), con llaves de 56 bits para DES o tres llaves de 56 bits para 3-DES. Los datos se cifran en bloques (bloques de 64 bits para el caso de DES).
- Las conexiones PPTP requieren sólo autentificación a nivel de usuario a través de un protocolo de autentificación basado en PPP. Las conexiones L2TP/IPSec requieren el mismo nivel de autentificación a nivel de usuario y, además nivel de autentificación de máquina usando certificados digitales.

Tecnología de VPN: Seguridad sobre VPN:

- Ipsec con encriptación
- SSL 3.0 ó TLS con encriptación

- **IPSEC con ENCRIPTACIÓN**: Es el más popular dentro de las VPNs. Soporta una amplia variedad de algoritmos de encriptación(DES, 3DES,AES, RC4), así como los mecanismos de integridad de los datos(MD5, SHA-1). También soporta Certificados digitales X.509.
Tiene una amplia variedad de soluciones en múltiples modos: Cliente, Servidor y Gateways.

Ventajas y Desventajas del uso de VPN IPSEC:

Ventajas:

- Todos los tipos IP y Servicios son soportados.
- Una vez que el intercambio de clave se completa, muchas conexiones pueden utilizar el túnel establecido.
- Misma tecnología base para trabajar entre cliente-cliente, cliente-site, site-site.
- Soporta tecnología de autenticación fuerte e integración del directorio.

Desventajas:

- Para las VPN Cliente se requiere una instalación software en el cliente; No todos los sistemas operativos requeridos por el cliente pueden soportarlo.
- Para las VPN Cliente se requiere la configuración cliente antes de que el túnel sea establecido.
- La conectividad puede ser afectada negativamente para NAT o mecanismos proxy entre el cliente y el gateway.
- La conectividad puede ser afectada negativamente para firewalls entre el cliente y el gateway.

- **SSL VPN (CLIENTLESS VPNs)**: Actualmente la tecnología SSL VPN (también llamada Clientless VPN) está en competencia con la tradicional IPSec VPN para el acceso remoto o el uso frecuente de la extranet. SSL VPN es básicamente la posibilidad de realizar VPN desde

una interface Web, con la posibilidad de acceder en forma cifrada, a determinados recursos de la empresa.
Soporta una amplia variedad de algoritmos de encriptación (DES, 3DES, AES, RC4, RSA, DSA), así como los mecanismos de integridad de los datos (MD5, SHA-1).

Ventajas y Desventajas del uso de SSL VPN:

Ventajas:

- Https cliente es soportado por todos los sistemas operativos.
- La mayoría de las aplicaciones mas populares de cliente / Servidor soportan SSL.
- Opera transparentemente a través de NAT y mecanismos proxy.

Desventajas:

- Solo soporta servicios TCP, y a menudo solo http ó POP3/IMAP/SMTP sobre SSL.
- No usado para VPNs site-to-site.
- Los datos son seguros en transito, pero no hay control sobre la seguridad de los datos en el sistema del cliente.
- Múltiples Sesiones SSL pueden ser requeridas para una sesión.

La pregunta que se plantean es "Debo utilizar SSL VPN o IPSEC VPN ?". Es por ellos que hemos elaborado esta breve lista con las ventajas de una y otra tecnología:

SSL VPN es útil cuando:

- El tráfico es HTTP o e-mail.
- Se requiere acceso a la información desde cualquier dispositivo (laptop, PC, PDA) desde cualquier parte del mundo (oficina, casa, cybercafé).
- No es posible instalar software de acceso VPN en el PC remoto.

IPSEC VPN es útil cuando:
- La organización necesita disponer de una infraestructura que permita el tráfico de gran variedad de protocolos, no solo HTTP o e-mail
- La organización tiene control sobre los dispositivos de acceso remoto.
- Existen fuertes restricciones de seguridad en el acceso remoto (P.Ej: no permitir el acceso desde cybercafés).

4.5.4.2 Web cifrada:

Permite el establecimiento de un canal seguro de comunicación sobre http en el ámbito de una aplicación.

En la actualidad el protocolo más utilizado es el SSL.

SSL proporciona sus servicios de seguridad sirviéndose de dos tecnologías de cifrado distintas: criptografía de clave pública (asimétrica) y criptografía de clave secreta (simétrica). Para el intercambio de los datos entre el servidor y el cliente, utiliza algoritmos de cifrado simétrico, que pueden elegirse típicamente entre DES, triple-DES, RC2, RC4 o IDEA. Para la autenticación y para el cifrado de la clave de sesión utilizada por los algoritmos anteriores, usa un algoritmo de cifrado de clave pública, típicamente el RSA. La clave de

sesión es la que se utiliza para cifrar los datos que vienen de él y van al servidor una vez establecido el canal seguro.

Para establecer una comunicación segura utilizando SSL se tienen que seguir una serie de pasos.

Primero se deben establecer los parámetros que se utilizarán en la sesión a través de un handshake o intercambio de claves. Durante el handshake se cumplen varios propósitos: se hace autenticación del servidor y opcionalmente del cliente a través de certificados, se determina qué algoritmos de criptografía serán utilizados y se genera una clave secreta para ser utilizada durante el intercambio de mensajes subsiguientes durante la comunicación SSL:

- **Client Hello**: El "saludo de cliente" tiene por objetivo informar al servidor qué algoritmos de criptografía puede utilizar y solicita una verificación de la identidad del servidor. El cliente envía el conjunto de algoritmos de criptografía y compresión que soporta y un número aleatorio. El propósito del número aleatorio es para que en caso de que el servidor no posea un certificado para comprobar su identidad, aún se pueda establecer una comunicación segura utilizando un conjunto distinto de algoritmos. Dentro de los protocolos de criptografía hay un protocolo de intercambio de clave que define cómo cliente y servidor van a intercambiar la información, los algoritmos de clave secreta que definen qué métodos pueden utilizar y un algoritmo de hash de una sola vía. Hasta ahora no se ha intercambiado información secreta, solo una lista de opciones.

- **Server Hello**: El servidor responde enviando su identificador digital el cual incluye su clave pública, el conjunto de algoritmos criptográficos y de compresión y otro número aleatorio. La decisión de qué algoritmos serán utilizados está basada en el más fuerte que tanto cliente como servidor soporten. En algunas situaciones el servidor también puede solicitar al cliente que se identifique solicitando un identificador digital.

- **Aprobación del Cliente**: El cliente verifica la validez del identificador digital o certificado enviado por el servidor. Esto se lleva a cabo descifrando el certificado utilizando la clave pública del emisor y determinando si éste proviene de una entidad certificadora de confianza. Después, se hace una serie de verificaciones sobre el certificado, tales como fecha, URL del servidor, etc. Una vez se ha verificado la autenticidad de la identidad del servidor. El cliente genera una clave aleatoria y la cifra utilizando la clave pública del servidor y el algoritmo criptográfico y de compresión seleccionado anteriormente. Esta clave se le envía al servidor y en caso de que el handshake tenga éxito será utilizada en el envío de futuros mensajes durante la sesión.

- **Verificación**: En este punto ambas partes conocen la clave secreta, el cliente porque la generó y el servidor porque le fue enviada utilizando su clave pública (La única forma posible de descifrarla es utilizando la clave privada del servidor). Se hace una última verificación para comprobar si la información transmitida hasta el momento no ha sido alterada. Ambas partes se envían una copia de las anteriores transacciones cifrada con la clave secreta. Si ambas partes confirman la validez de las transacciones, el handshake se completa, de otra forma se reinicia el proceso.

Ahora ambas partes están listas para intercambiar información de manera segura utilizando la clave secreta acordada y los algoritmos criptográficos y de compresión. El handshake se realiza sólo una vez y se utiliza una clave secreta por sesión.

- Intercambio de datos: Ahora que se ha establecido un canal de transmisión seguro SSL, es posible el intercambio de datos. Cuando el servidor o el cliente desean enviar un mensaje al otro, se genera un resumen (utilizando un algoritmo de hash de una vía acordado durante

el handshake), cifran el mensaje y el resumen y se envía, cada mensaje es verificado utilizando el resumen.
- Terminación de una sesión SSL: Cuando el cliente deja una sesión SSL, generalmente la aplicación presenta un mensaje advirtiendo que la comunicación no es segura y confirma que el cliente efectivamente desea abandonar la sesión SSL.

Ventajas y Desventajas del uso de web cifrada:

Ventajas:

- Facilidad en el Despliegue de esta solución.
- Mínimos requisitos (No necesidad de Cliente) para el establecimiento de la sesión.

Desventajas:

- Sólo utilizable en el ámbito de una aplicación Web (No puede utilizarse para cifrado de otros protocolos como POP3, FTP, ETC.).

4.5.5 Herramientas

Algunas herramientas para la seguridad sobre XML:

- **XML Security suite** (https://secure.alphaworks.ibm.com/tech/xmlsecuritysuite)

Es una herramienta que proporciona características de la seguridad tales como firma digital, cifrado, y control de acceso para los documentos de XML.

Estas características están más allá de la capacidad de los protocolos de la seguridad del nivel transporte SSL.

¿Cómo trabaja? Tres tecnologías se incluyen en XML Security suite:

- Puesta en práctica de la firma Digital basada en " "XML-Signature Syntax and Processing" " por la puesta en práctica del cifrado de W3C/IETF.
- XML basada en "XML Encryption Syntax and Processing"
- XML Access Control Language and implementation

- **VORDEL DIRECTOR** (http://www.vordel.com/products/vordeldirector/index.html)

VordelDirector proporciona los servicios de seguridad discretos que se utilizan a través de la red. Estos servicios de seguridad incluyen el cifrado, firma digital, amenaza-análisis de XML, de la emisión y de la validación simbólica de la seguridad, de la autentificación, de la autorización, y del logging.

Los servicios de seguridad proporcionados por VordelDirector se diseñan para ser parte de los servicios web en una Arquitectura orientada a Servicios (SOA). Estos servicios de seguridad son utilizados por usos múltiples, incluyendo las entradas de XML, los usos basados NET, y los usos basados Java.

VordelDirector envía con dos aplicaciones que hagan uso de sus servicios de seguridad. El primero es un proxy que actúa en tráfico de XML en la red. El segundo uso es un plug-in del web server que intercepta XML en un web

server. Ambas aplicaciones hacen cumplir reglas de la seguridad sobre tráfico de XML.

- **VORDEL SECURE** (http://www.vordel.com/products/vordelsecure/index.html)

  VordelSecure es un gateway en XML, que es un elemento intermediario en línea que filtra el tráfico XML según un sistema de reglas configurables de la seguridad. Protege sistemas de XML, tales como servicios web, asegurándose de que solamente los usuarios y aplicaciones correctas - enviando los datos correctos - pueden conectar con ellos. Cuando está desplegada directamente detrás de cortafuegos de red en una red DMZ (zona desmilitarizada), una entrada de XML a menudo se llama un "cortafuego de XML".

  VordelSecure se puede desplegar como software o como aplicación del hardware, disponible a través de los socios de la aplicación de la compañía.

4.5.6 Bibliografía

- VPN Technologies: Definitions and Requirements.
- IPSEC Versus "Clientless" VPN for Remote Access.
- Hackers 3: Secretos y soluciones para la seguridad de redes.
- Criptografía y Seguridad en Computadores. Manuel José Lucena López
- Análisis de Seguridad de la familia de protocolos TCP/IP y sus servicios asociados. Raúl Siles Peláez.
- UNE-ISO/IEC 17799: Código de buenas prácticas para la Gestión de la Seguridad de la Información.
- Virtual Private Networks (VPN /PPTN)
  http://www.helmig.com/j_helmig/vpn.htm
- Understanding Virtual Private Networks (VPN)
  http://www.sans.org/infosecFAQ/encryption/understanding_VPN.htm
- Understanding PPTP and VPNs
  http://www.aliceinwondeland.com/library/website_archives/rhino9/pptp.htm
- PoPToP – The PPTP Server for Linux
  http://poptop.lineo.com
- PPTP
  http://www.cas.mcmaster.cal.~wmfarmer/SE-4C03-01/papers/Silva-PPTP.htm

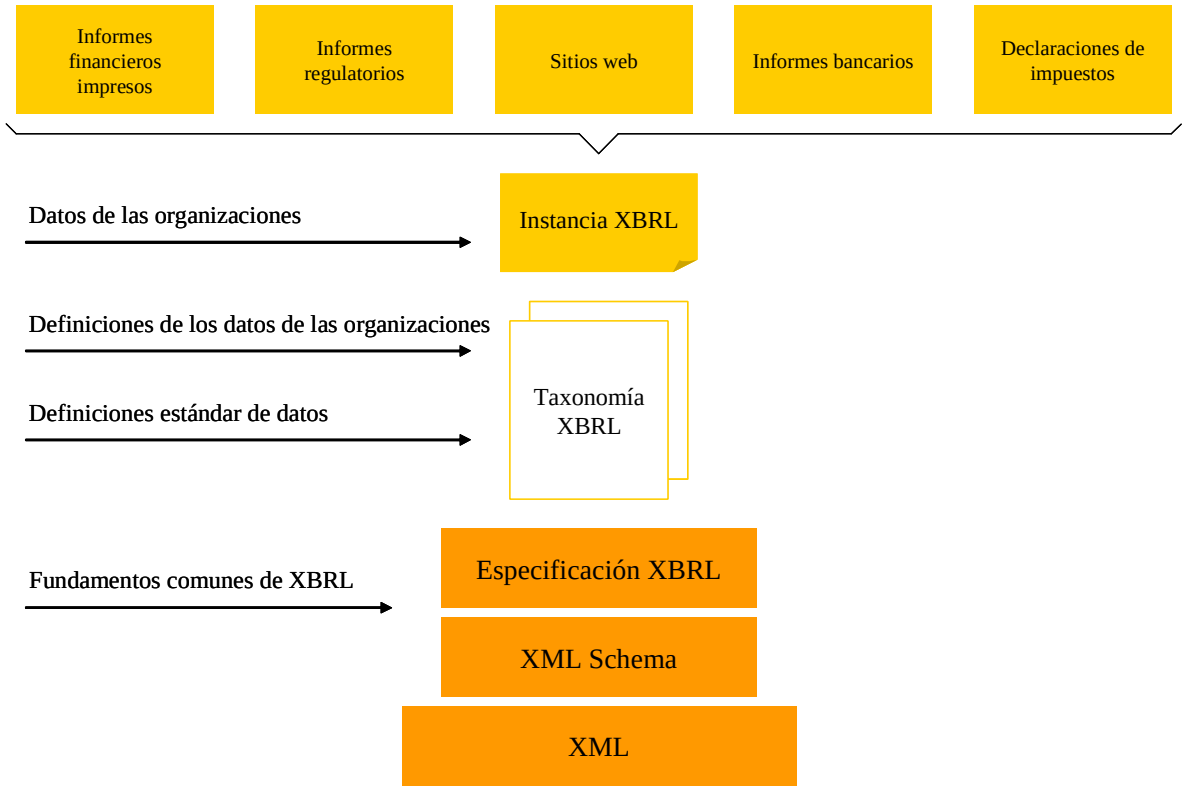
- Acceso remoto por VPN (Red Privada Virtual)
  http://www.uv.es/ciuv/cat/vpn
- Cisco – Layer 2 Tunnel Protocol
  http://www.cisco.com/warp/public/cc/pd/iosw/tech/l2pro_tc.htm
- VPN FAQ
  http://kubarb.phsx.ukans.edu/~tbird/vpn/FAW.html

4.6 Arquitectura XBRL

4.6.1 Arquitectura de referencia

XBRL es una derivación de XML (eXtensible Mark-up Language). XML, como formato estándar de comunicación independiente de las plataformas, ha sido diseñado para facilitar el intercambio de información entre aplicaciones a través de las redes corporativas o en Internet. De esta manera, soportado prácticamente por todos los proveedores de software, XML se está imponiendo como el principal facilitador para la transferencia de datos.

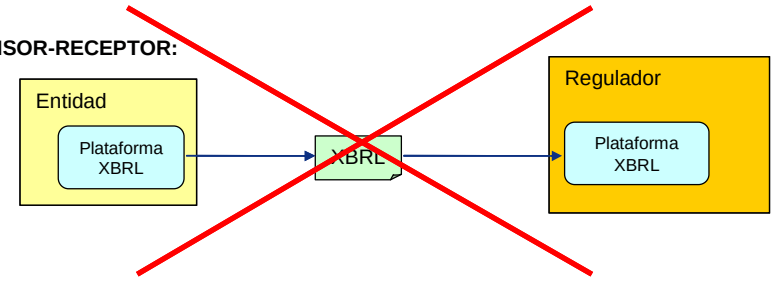
XBRL se ha creado específicamente para optimizar la tecnología XML en su aplicación a la cadena de distribución de información financiera.



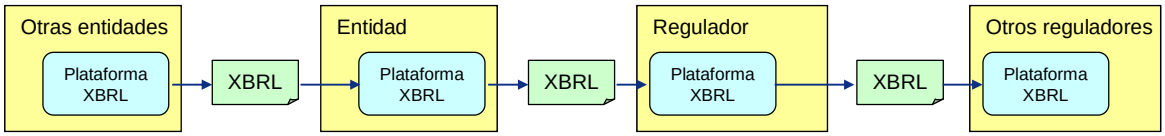
Una aproximación inicial basada en los primeros pasos que se están dando para la implantación de XBRL como estándar para intercambio de información financiera, fundamentalmente impulsada por los organismos reguladores, podría hacer pensar en un escenario en el que hay entidades que reciben datos XBRL (reguladoras) y otras que son las que envían dichos datos (reguladas).

Sin embargo, un análisis más profundo pone de manifiesto que aquella entidad que ha de reportar a un regulador, a su vez puede estar recibiendo datos de sus subsidiarias, y que aquel organismo regulador que recopila informes de sus entidades reguladas, ha de consolidar datos y reportarlos a otro regulador de ámbito más global. Es decir, cualquier organización que procesa datos financieros es en potencia un eslabón de una cadena de distribución de información financiera. O sea, podría recibir, generar, manipular, consolidar y publicar informes financieros en formato XBRL. Por lo tanto, las organizaciones requieren sistemas que respondan completamente a estas necesidades.

ENFOQUE EMISOR-RECEPTOR:



ENFOQUE CADENA DE DISTRIBUCIÓN:



Por lo tanto, cualquier arquitectura diseñada para la gestión y procesamiento de datos en formato XBRL debe soportar las fases esenciales del proceso de reporting financiero:

- Creación, distribución, obtención y manejo de múltiples taxonomías.
- Creación, publicación, recepción, validación e interpretación de instancias.
- Repositorio para almacenamiento y búsqueda.

Adicionalmente, la arquitectura puede incluir elementos complementarios:

- Mecanismos de seguridad: Si bien es cierto que la especificación XBRL no proporciona ninguna directiva ni referencia sobre mecanismos de seguridad, las comunicaciones de la información que se intercambia en formato XBRL requieren, en la mayor parte de los casos, cumplir con estrictos requerimientos de seguridad que garanticen la confidencialidad, la integridad, la autenticidad y el no repudio de los datos.
- Herramientas para el desarrollo de taxonomías en un entorno colaborativo, permitiendo gestionar completamente el ciclo de vida de las mismas.
- Herramientas de monitorización y control de los procesos de tratamiento de informes XBRL.
- Adaptadores para la extracción y conversión de datos en formato no XBRL a XBRL.
- Funciones de análisis de la información en XBRL.
- Proceso en batch de instancias XBRL.

Gráficamente:

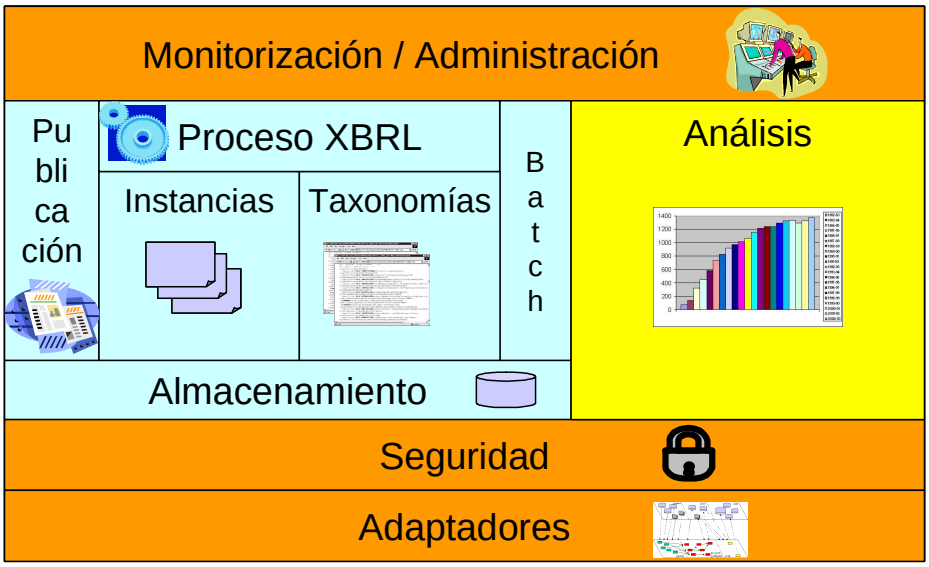
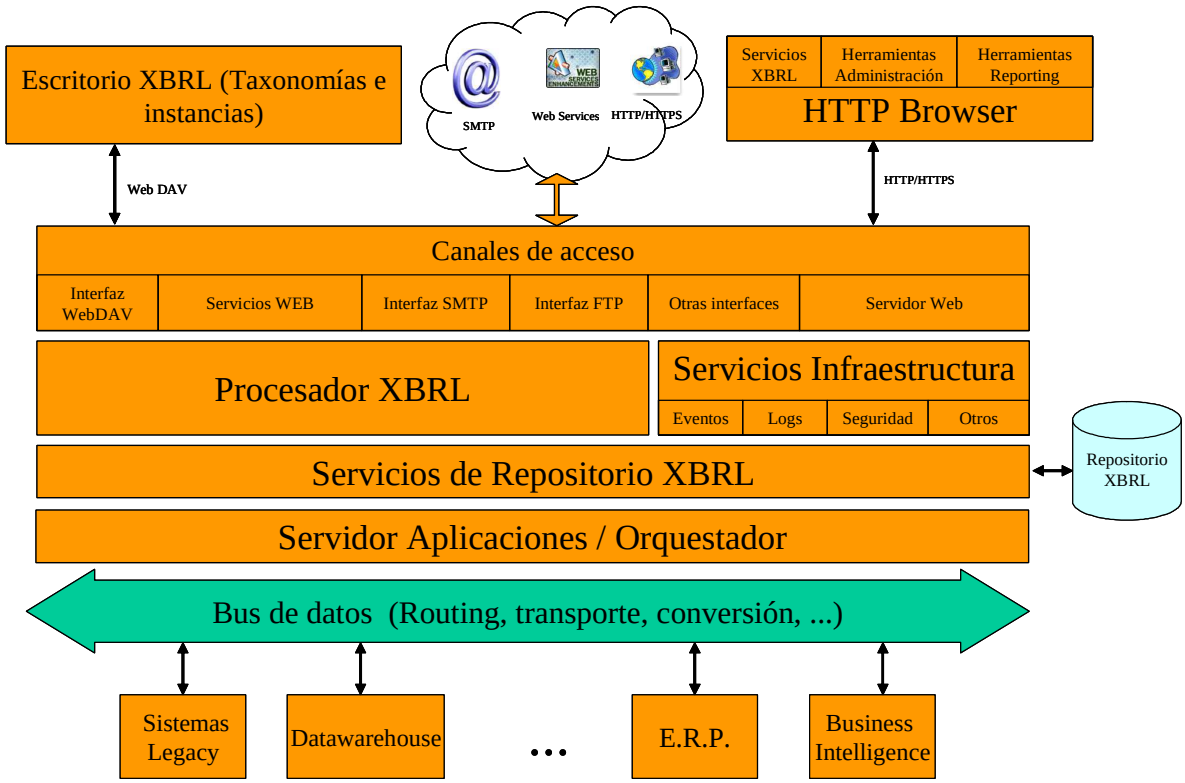


Figura 4.9-3

La arquitectura funcional descrita anteriormente ha de estar sustentada por una arquitectura técnica que responda a los requerimientos de escalabilidad y rendimiento necesarios. A continuación se presenta una propuesta de arquitectura técnica de referencia que cubre dichos requisitos. Se caracteriza por su diseño modular y por su orientación a servicios.



Desglosando la arquitectura por niveles:

- **Escritorio XBRL**: Agrupa e integra herramientas XBRL ejecutables en puesto cliente para la creación y edición de taxonomías e instancias, así como otras utilidades relacionadas. Estas herramientas pueden trabajar en modo local o en conexión con el repositorio de documentos XBRL residente en la parte servidora (WebDAV).

- **Cliente Web**: Interfaz de usuario principal, proporcionando:
  - Gestión y control de los servicios y operaciones directamente relacionados con el proceso de documentos XBRL.
  - Herramientas para el diseño, generación y manejo de informes.
  - Opciones de administración y operación del sistema.
- **Canales de acceso**: Mecanismos implementados para la recepción y envío de instancias XBRL, así como para la descarga y distribución de taxonomías XBRL. Debe soportar protocolos tales como SMTP, FTP y HTTP(S) y proporcionar un conjunto de servicios web para la distribución de datos.

- **Procesador XBRL**: Es el núcleo de la arquitectura; provee los mecanismos fundamentales para el procesado de los documentos XBRL.

- **Servicios de infraestructura**: Agrupa todos aquellos servicios que soportan las funcionalidades no estrictamente XBRL de la arquitectura tales como:
  - Notificaciones, eventos y logs.
  - Mecanismos de seguridad y control.

- **Servicios de repositorio XBRL**: Almacenamiento de documentos XBRL, tanto taxonomías como instancias, con funciones de gestión de históricos, control de versiones y buscador de datos. La disponibilidad y características de estos servicios dependen del tipo de almacenamiento implementado como repositorio, que puede ser:
  - Repositorio de objetos.
  - Base de datos relacional.
  - Repositorio de datos XML.
  - Sistema de ficheros.

El nivel de granularidad proporcionado por el repositorio también depende del tipo de almacenamiento. Así puede proporcionar granularidad a nivel de fichero (esquema y linkbases) o llegar al nivel de elemento de taxonomía.

Debe contemplarse también el almacenamiento de casos de prueba de las taxonomías, en el mismo repositorio en el que éstas residan o en otro específico.

- **Servidor de aplicaciones/Orquestador**: El uso de un servidor de aplicaciones u orquestador es opcional, dependiendo de que los módulos incluidos en la arquitectura lo requieran o no.

- **Bus de datos**: Bajo el control del servidor de aplicaciones u orquestador, aglutina todas las interfaces con sistemas externos, tales como Mainframe, sistemas de Datawarehouse, ERPs, Business Intelligence, etc. Proporciona mecanismos de routing, transporte y adaptadores para el mapeo, extracción de datos no XBRL y su transformación a XBRL.

4.6.2 Rendimiento

Las arquitecturas diseñadas para el tratamiento de informes en formato XBRL deben ser capaces de procesar grandes volúmenes de datos proporcionando un rendimiento adecuado.

Consideraciones a tener en cuenta respecto al volumen de datos:

- En la definición de taxonomías se tiende a crear DTSs (Discoverable Taxonomy Set) con múltiples niveles de importación y, en algunos casos, con modelos de datos complejos.
- Se están definiendo taxonomías internacionales con muchos elementos (por ejemplo, la taxonomía IFRS-GP tiene más de 4.000).
- Se pueden construir instancias con múltiples contextos.
- Para un mismo informe financiero, el reporting en XBRL representa un incremento en volumen sobre el reporting tradicional (texto plano) en una relación del orden de 3 a 1.
- Si bien los procesos de reporting financiero no implican un tráfico muy intenso de datos, si que tienden a concentrarse en unas fechas concretas.

Por lo tanto, hay que prever el procesamiento simultáneo de instancias XBRL de gran tamaño.

Recomendaciones de diseño para optimizar el rendimiento de las aplicaciones XBRL:

- Emplear procesadores XBRL que proporcionen:

  - Posibilidad de reutilización de modelos de datos de taxonomías o, expresado de otra manera, mecanismos de caché de taxonomías en memoria: La creación en memoria del modelo de datos relativo a un DTS es uno de los procesos más pesados asociado al tratamiento de documentos XBRL. Mediante este mecanismo la taxonomía se carga una vez en memoria, estando disponible para procesar todas las instancias vinculadas a ella.

  - Mecanismos de serialización/deserialización: Una alternativa interesante a la caché de taxonomías es su almacenamiento como objetos serializados. Según estudios realizados, el tiempo de lectura de una taxonomía es directamente proporcional al cuadrado del número de los enlaces de sus linkbases, mientras que el tiempo de lectura de una taxonomía serializada (deserialización) es directamente proporcional al número de enlaces de sus linkbases.

  - Posibilidad de precompilación de esquemas XML.

  - Mecanismos de resolución eficaces para resolver el acceso y salvado de documentos XBRL.

  - Empleo del parser XML (DOM o SAX) más óptimo para cada situación.

  - Procesar únicamente los linkbases que sean necesarios para cada situación.

- En taxonomías que importen a otras, inhabilitar los cálculos heredados y que no tengan sentido en la nueva taxonomía (mediante el atributo use = "prohibited" en los arcos de los roles de cálculo a inhabilitar). De esta manera, aunque el tamaño del linkbase de cálculo aumente, el tiempo de validación de las instancias disminuye considerablemente.

- Seleccionar el tipo de repositorio a utilizar para el almacenamiento de documentos XBRL valorando las prestaciones de rendimiento que proporciona para el manejo de datos XML y XBRL.

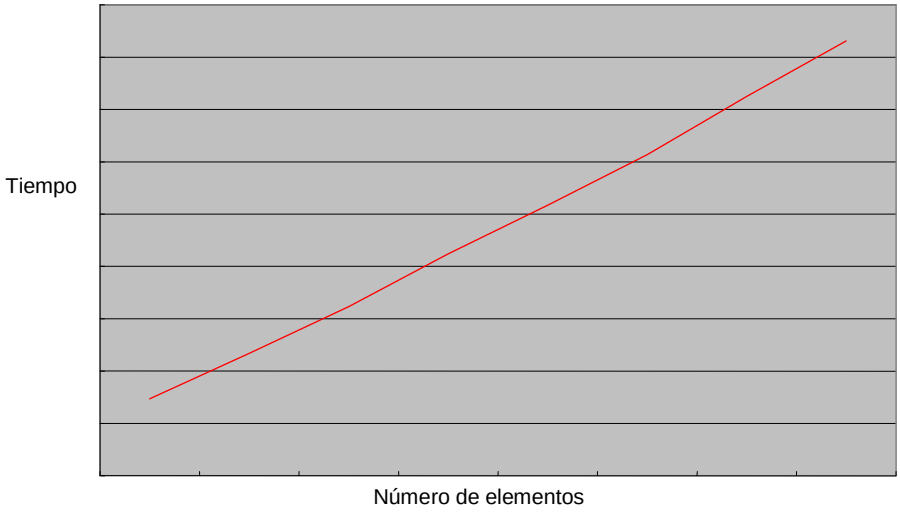
- Diseñar arquitecturas escalables.

- Minimizar la longitud de las etiquetas XML/XBRL en el diseño de hojas de estilo (XSLT) diseñadas para la transformación de documentos XBRL.

A continuación se muestra una serie de gráficas que presentan el comportamiento típico – en términos de tiempo de respuesta en función del número de elementos - de algunos de los procesos más habituales que se realizan con documentos XBRL:

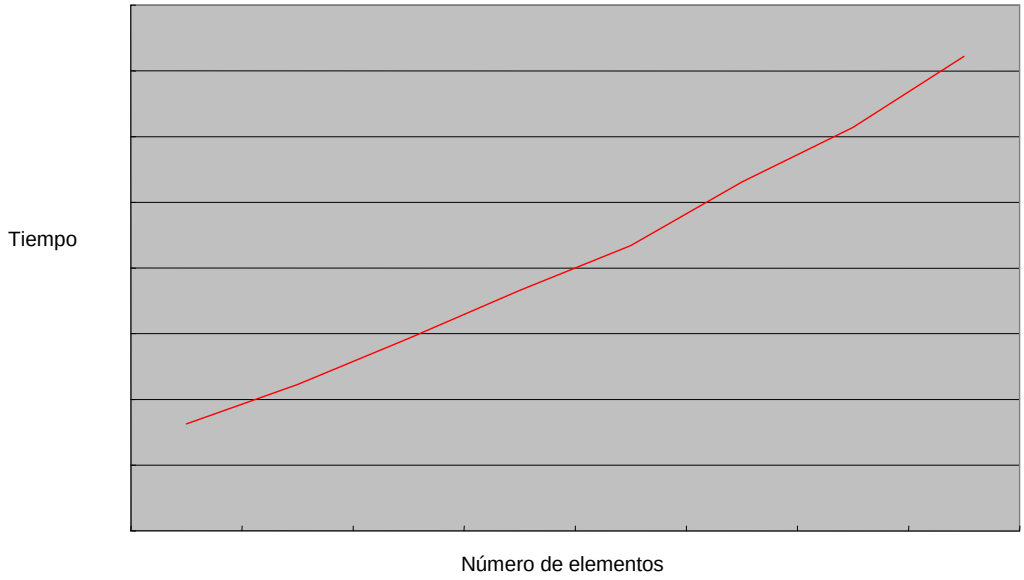
1 Carga (y validación) de taxonomía en memoria.

Figura 4.9-5



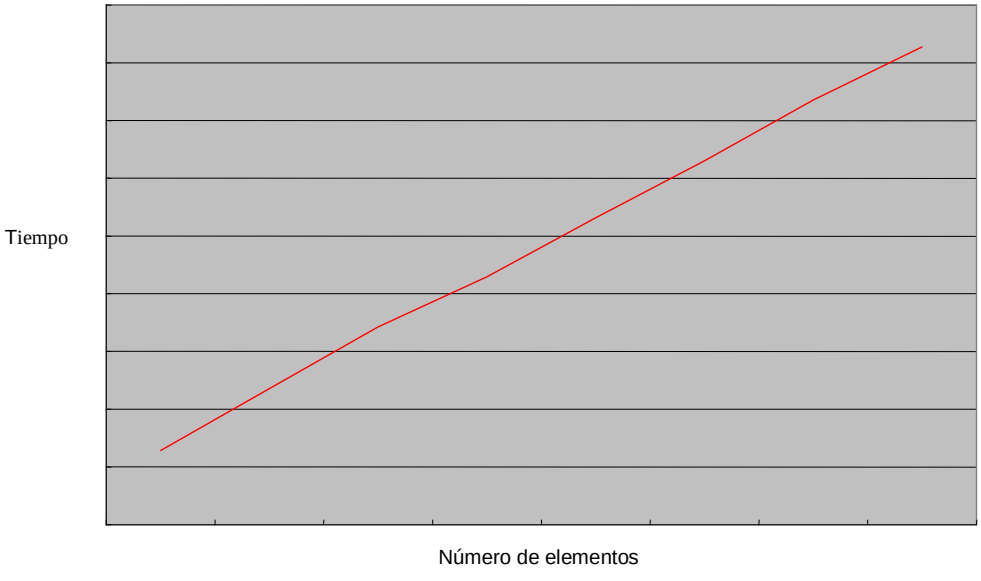
2 Carga (y validación) de instancia en memoria.

Figura 4.9-6



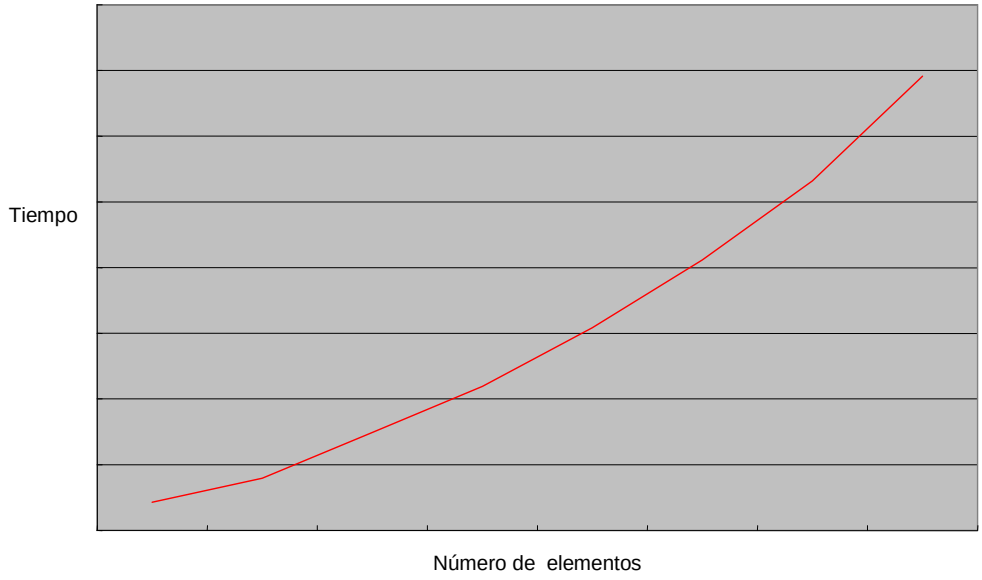
3 Serialización de una taxonomía.

Figura-4.9-7



4 Deserialización de una taxonomía.

Figura 4.9-8



4.7 Dimensiones

4.7.1 Introducción

La mayor parte de las taxonomías definidas hasta ahora estaban limitadas a estructuras bidimensionales, a través del uso de tuplas, o estructuras mono-dimensionales, mediante el uso listas jerárquicas de conceptos.

En Enero de 2005, el Comité Europeo de Supervisores Bancarios hizo público lo que se conoce como *Common Reporting framework* (COREP – COmmon REPorting) por el que las instituciones de crédito y compañías inversoras reportarán información relativa a sus ratios de solvencia bajo la nueva directiva de requisitos de capital.

La taxonomía COREP se caracteriza por un modelo de datos basado necesariamente en una estructura de taxonomía dimensional.

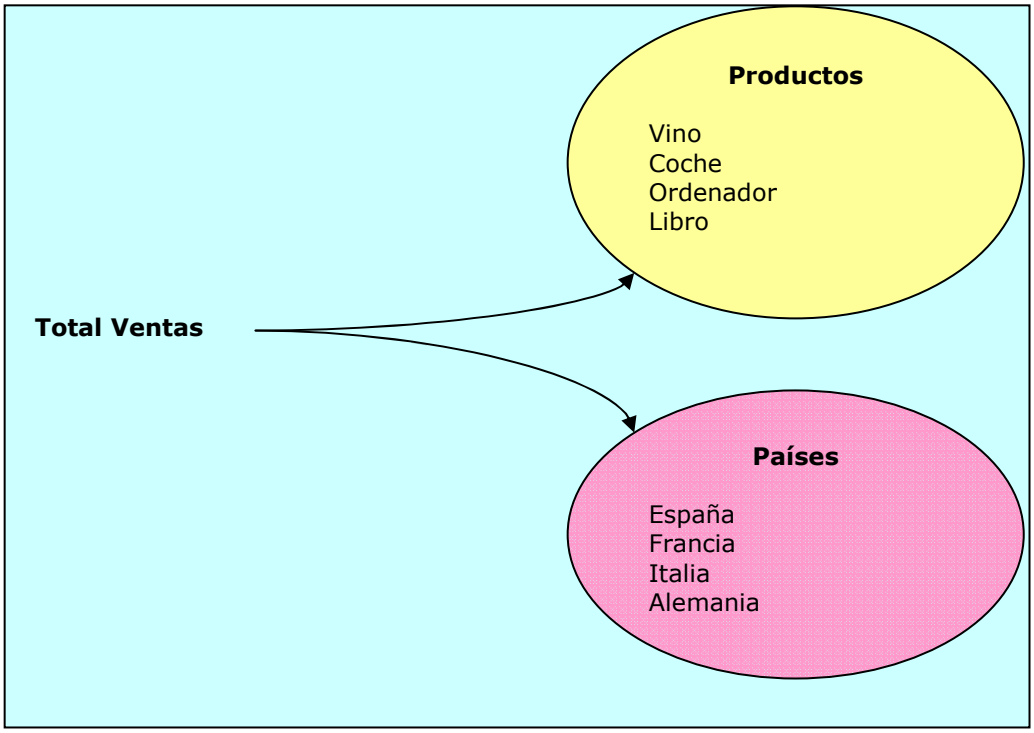
Ninguna taxonomía desarrollada en XBRL había presentado tales requisitos hasta ese momento.

Es por esto, por lo que el Consorcio de XBRL Internacional decidió desarrollar lo que hoy se conoce como la Especificación de Dimensiones.

Otras taxonomías en las que se hace uso de esta nueva especificación son FINREP, *FINancial REPorting framework* y PGC 90, Plan General Contable 1990.

Una taxonomía dimensional permite definir por cada concepto distintas dimensiones, las cuales representan jerarquías de información de un determinado aspecto.

EJEMPLO: REPRESENTACIÓN DEL CONCEPTO "TOTAL VENTAS" POR PRODUCTOS Y POR PAÍSES.



4.7.2 Descripción de la Especificación de dimensiones 1.0

Actualmente, la versión publicada de esta especificación es "XBRL Dimensions 1.0" – Candidata a Recomendación 3 (XDT-CR3-2006-04-26.rtf).

Desarrollada por el Consorcio de XBRL Internacional, se encuentra disponible en diferentes formatos en la Web: http://www.xbrl.org/SpecRecommendations/.

El documento se compone de las siguientes partes:

- Introducción, donde se explica el propósito del documento, relación con otros documentos, terminología así como los espacios de nombres (*namespaces*) utilizados en la especificación.

- Taxonomías dimensionales: en este apartado se describe la arquitectura utilizada para desarrollar taxonomías dimensionales. Asimismo, se definen todos los conceptos y relaciones a utilizar en una taxonomía dimensional.

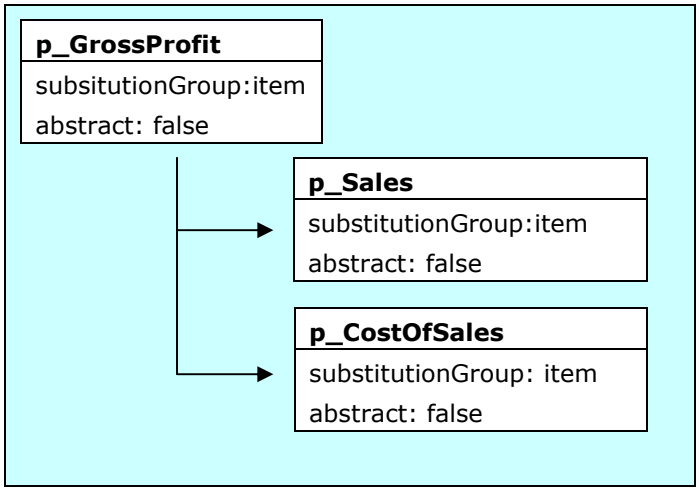
- Uso de dimensiones en instancias: una instancia XBRL cuyo DTS contenga relaciones dimensionales de acuerdo a los especificado en el apartado anterior, deberá ser validada de acuerdo a las reglas que se describen en este capítulo.

La especificación de dimensiones 1.0 es una extensión de la especificación XBRL 2.1. Está desarrollada de acuerdo a los requisitos de XBRL 2.1. Los documentos de instancia XBRL que contienen información dimensional pueden ser procesados sin ningún error por cualquier validador que es capaz de procesar correctamente XBRL.

4.7.3 Taxonomías Primarias (Primary Taxonomies)

Son taxonomías XBRL cuyo DTS no tiene elementos dimensionales ni arcos definidos en la Especificación de Dimensiones. Taxonomías que no contienen información dimensional pueden ser extendidas para añadir dimensiones.

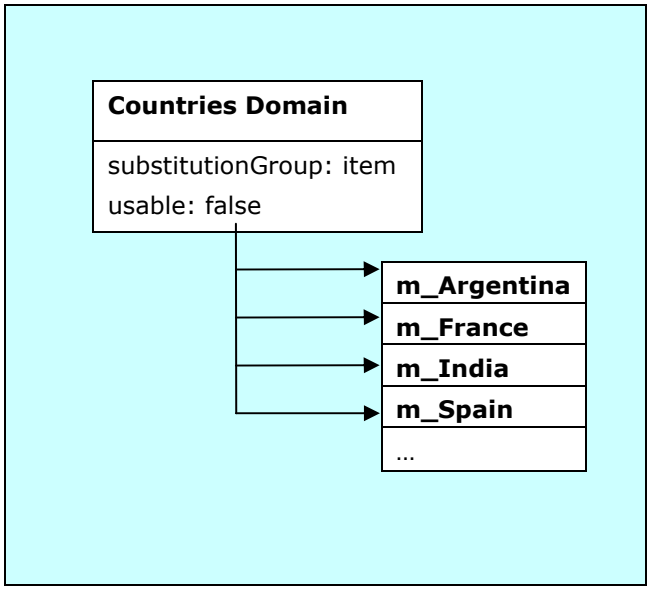
Con el término de taxonomía primaria se hace referencia a un DTS de elementos que pueden ser reportados en una instancia XBRL.



4.7.4 Taxonomías domain-member (Domain Member Taxonomies)

Existen dos tipos de taxonomías dimensionales:

- *Typed Dimensional Taxonomies*: definen restricciones de los contenidos en el segmento o en el escenario.
- *Explicit Dimensional Taxonomies*: son aquéllas cuyos elementos XBRL conforman un conjunto discreto de miembros. Este conjunto se denomina dominio.



Las instancias XBRL pueden un número indeterminado de dimensiones, posibilitando así cualquier combinación de de los miembros de sus dominios tanto en el escenario como en el segmento.

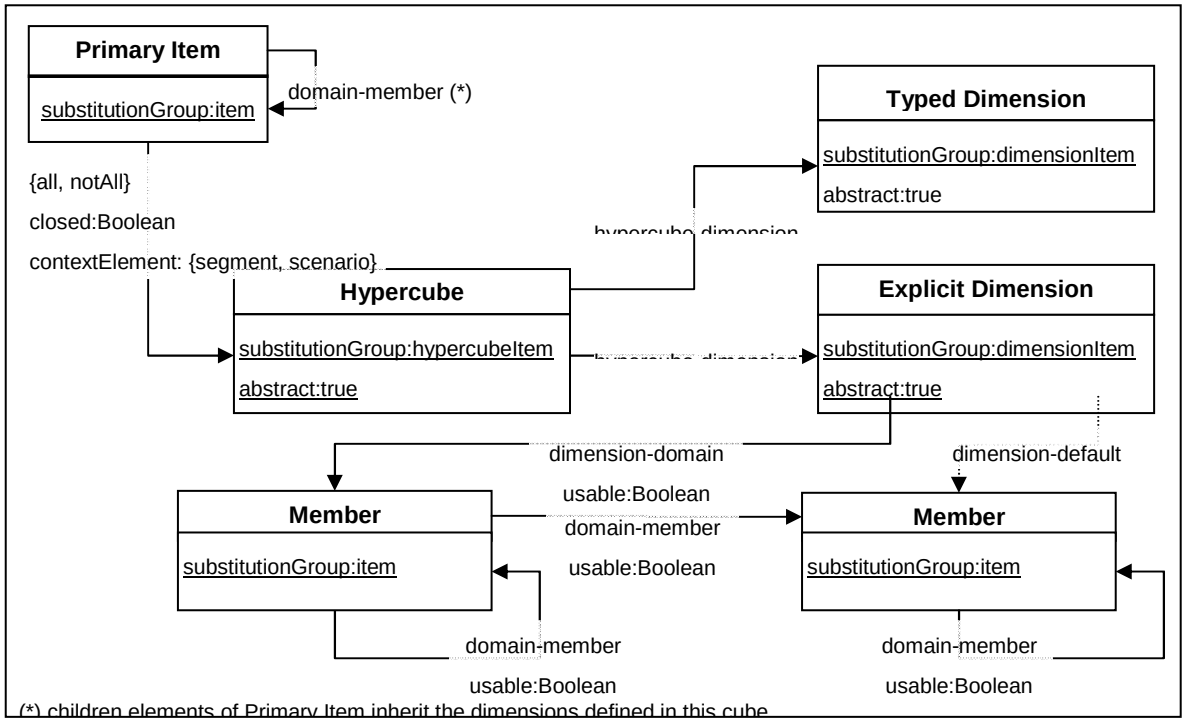
4.7.5 Taxonomías template (Template Taxonomies)

Importa tanto las taxonomías primarias como las taxonomías domain-member y añade las estructuras dimensionales que serán utilizadas en la instancia XBRL. Por convenio, una taxonomía que importa taxonomías primarias y domain-member y define toda la información dimensional necesaria se denomina Taxonomía Template.

Una taxonomía template define hipercubos. Un hipercubo describe el producto cartesiano de cero o más dimensiones. Cada dimensión está definida por cero o más dominios y los dominios están formados por miembros.

Ejemplo: Ingresos por provincias y por productos. Se trata de un hipercubo con tres dimensiones, con un elemento primario (Ingresos) y dos dimensiones explícitas (productos y provincias).

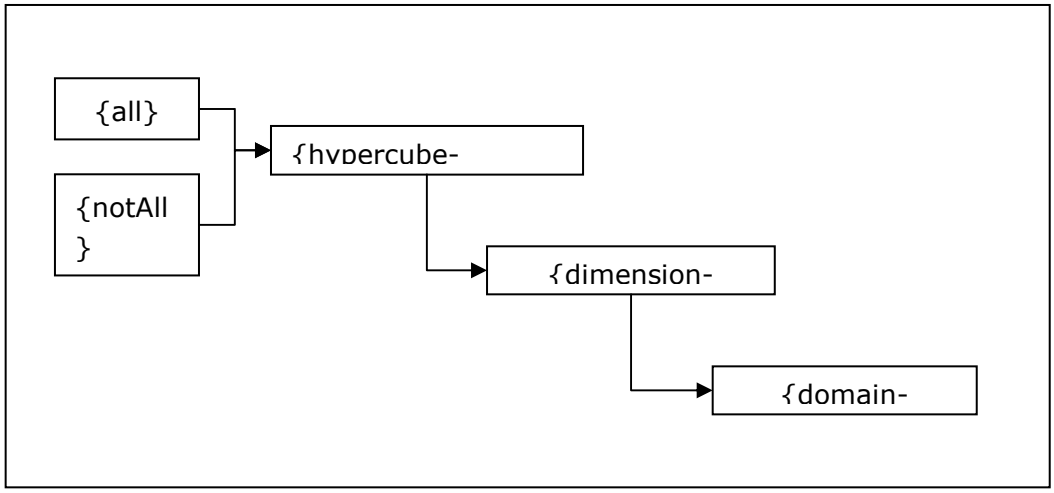
4.7.6 Modelo Conceptual



Primary Elements son elementos definidos en taxonomías XBRL de tipo xbrli:item y no xbrldt:hypercubeItem o xbrldt:dimensionItem.

Arcos consecutivos son dos arcos en secuencia. En la especificación se definen los siguientes:

{all} – {hypercube-dimension}
{notAll} – {hypercube-dimension}
{hypercube-dimension} – {dimension-domain}
{dimension-domain} – {domain-member}



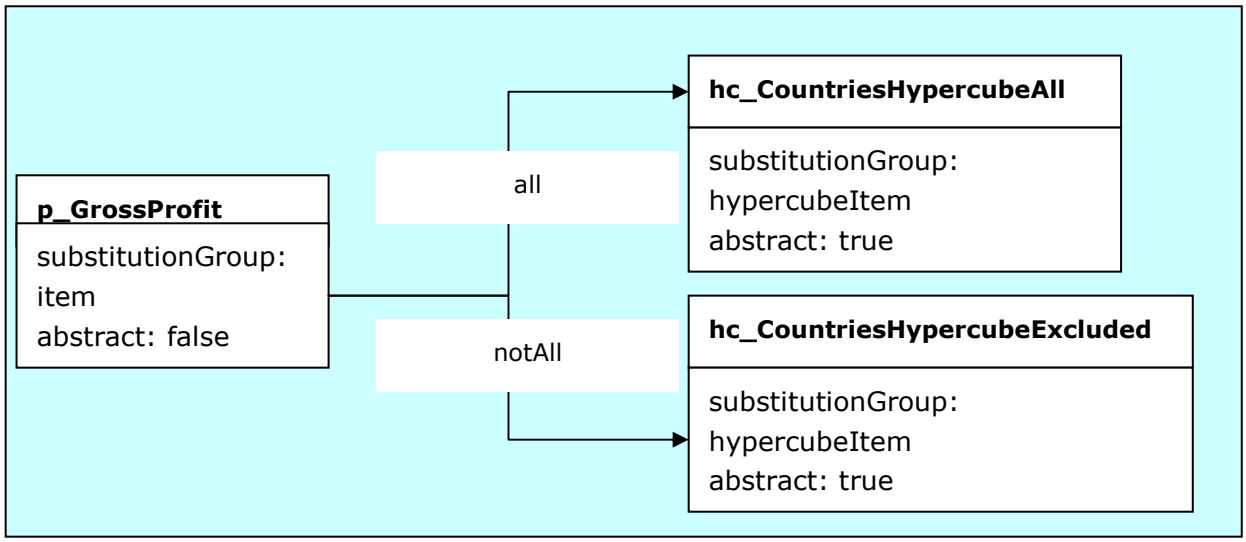
4.7.6.1 Arcos all y notAll

Para restringir una serie de contextos para un elemento primario dado, dicho elemento puede estar asociado a cero o más hipercubos.

Un conjunto de hipercubos se componen por la conjunción de arcos "all" y "notAll".

http://xbrl.org/int/dim/arcrole/all: arco cuyo origen es un *primary item* y destino un elemento hipercubo.

El arco **http://xbrl.org/int/dim/arcrole/notAll** es la versión negada del anterior.



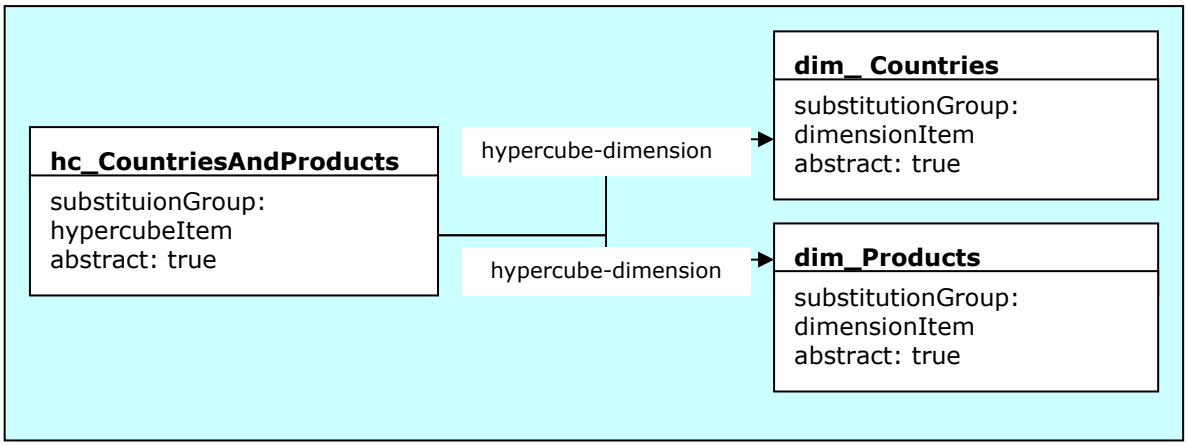
El elemento primario p_GrossProfit tiene asociado dos hipercubos. En una instancia, un contexto será válido con respecto al elemento primario sólo si el páis que aparece en el escenario es miembro del hipercubo hc_CountriesHypercubeAll y no lo es de hc_CountriesHypercubeExcluded.

Los arcos all y notAll requieren el atributo xbrldt:contextElement. Dicho atributo puede tomar dos valores, segmento y escenario.

El atributo *booleano* opcional xbrldt:closed se utiliza para especificar si el hipercubo es cerrado (xbrldt:closed = true) con respecto al escenario o al segmento del contexto. Este atributo toma *false* como valor por defecto.

4.7.6.2 Arcos hypercube - dimension

El arco **http://xbrl.org/int/dim/arcrole/hypercube-dimension** tiene como origen un elemento declarado como hipercubo y como destino un elemento definido como dimensión.



4.7.6.3 Dimensiones

Una dimensión es un elemento abstracto cuyo atributo substitutionGroup toma valor dimensionItem.

Existen dos tipos de dimensiones:

- Dimensiones *typed*
- Dimensiones explícitas

4.7.6.3.1 Dimensiones typed

Una dimensión *typed* es un elemento dimensión cuyo dominio de miembros está definido en un elemento XML. La dimensión y el elemento XML se relacionan por el atributo xbrldt:typedDomainRef.

El atributo xbrldr:typedDomainRef es un xlink:href que apunta a un elemento XML donde está definido el dominio de la dimensión.

4.7.6.3.2 Dimensiones explícitas

Una dimensión explícita es un elemento dimensión que tiene arcos dimension-domain que apuntan a cero o más declaraciones domain-member.

Los miembros del dominio son elementos xbrli:item. Por tanto, heredan todas las propiedades de los elementos XBRL como etiquetas en distintos idiomas, referencias, estructuras de presentación…

4.7.6.4 Arco dimension-domain

Tiene como origen la declaración de una dimensión explícita y como destino la declaración de miembros del dominio de la dimensión.

4.7.6.5 Arco domain-member

La utilización de este arco permite la creación de conjuntos de miembros de dominio explícitos.

El atributo opcional xbrldt:usable se utiliza para evitar que ciertos miembros del dominio no aparezcan en un documento instanciado. Esto permite utilizar elementos para cuya finalidad es organizar la jerarquía del dominio.

Este atributo tiene como valor por defecto *true*.

4.7.6.6 Valores por defecto

El arco **http://xbrl.org/int/dim/arcrole/dimension-default** permite definir valores por defecto en dimensiones.

La relación dimension-default especifica que miembros del dominio actúan por defecto para una dimensión.

Los valores por defecto no deben aparecer en el contexto de la instancia.

Dicho arco tiene como origen un elemento dimensión y como fuente el miembro a utilizar por defecto.

4.7.7 Instancias

Las taxonomías primarias definen conceptos cuyos valores son reportados en una instancia XBRL.

Cada valor reportado en la instancia tiene un contexto asociado. En dicho contexto está contenida la información dimensional. Dependiendo del valor del atributo xbrldt:contextElement, esta información se encontrará bien en el escenario, bien en el segmento:

```
<p:GrossProfit  contextRef="c1" unitRef="eur" decimals="INF">10000</p:GrossProfit>
<p:Sales contextRef="c1" unitRef="eur" decimals="INF">60000</p:Sales>
<p:CostOfSales contextRef="c1" unitRef="eur" decimals="INF">50000</p:CostOfSales>
```

```
<context id="c1">
  <entity>
        <identifier scheme="http://www.example.com">example.com</identifier>
    <scenario>
          <xbrldi:explicitMember dimension="temp:dimProducts">d:Wine</xbrldi:explicitMember>
          <xbrldi:explicitMember
dimension="temp:dimCountries">d:Spain</xbrldi:explicitMember>
    </scenario>
   </entity>
   <period><forever/></period>
 </context>
```

Las dimensiones typed aparecen en la instancia de la siguiente manera:

```
<context id="c1">
  <entity>
        <identifier
scheme="http://www.example.com">example.com</identifier>
    <scenario>
        <xbrldi:typedMember dimension="tax:dCustomer">
                <cust>12345</cust>
         </xbrldi:typedMember>
    </scenario>
   </entity>
   <period><forever/></period>
 </context>
```

4.8 Fórmulas

4.8.1 Introducción

Dentro del marco de intercambio de información en XBRL, se necesita que las aplicaciones que producen y consumen la información sean capaces de aplicar validaciones sobre los tipos y valores de los datos, tanto comprobaciones de consistencia de la información y sus posibles correcciones como proporcionar suficiente información a la aplicación productora para detectar la gravedad de los errores encontrados.

Las aplicaciones que intercambian informes electrónicos, en general, suelen añadir valores calculados y realizar pruebas sobre sus propias salidas antes de enviar la información al resto de aplicaciones. Estas aplicaciones derivan normalmente en temas de mantenimiento tanto de los datos como de las reglas de negocio (fórmulas) asociadas a los criterios de validación de los datos.

Estas características que son comunes a cualquier aplicación que utilice XBRL como formato de intercambio hace que uno de los objetivos de estándar XBRL haya sido siempre permitir a las aplicaciones consumir información XBRL sin importar la fuente de origen.
La validación de los datos de entrada es un caso de uso para usar fórmulas que detecten errores.

La gestión de información para garantizar la calidad normalmente recomienda que las validaciones se comprueben, anexen y corrijan tan arriba en el flujo de suministro de la información como sea posible. Se recomienda también que las mismas pruebas realizadas sobre los datos se apliquen repetidamente a medida que la información viaja por los consumidores de la cadena.

El lenguaje XBRL desde sus comienzos expresa no únicamente hechos producidos sino también las relaciones existentes entre los hechos y relaciones generales. Es decir, las taxonomías son una forma de documentar el significado de los conceptos reportados.
De esta forma, otro caso de uso importante de las fórmulas es que sirven para documentar las relaciones de cálculo complejas entre los conceptos. De igual manera a como la linkbase de presentación establece el lugar jerárquico donde debe aparecer presentado un concepto, la linkbase de fórmulas indica la jerarquía dentro de una fórmula en la que un concepto aparece. Esto es especialmente útil a la hora de identificar aquellos conceptos que participan en el cálculo de un ratio de análisis, estando documentado en la linkbase.

Dentro del primer planteamiento se han ilustrado deficiencias en los esquemas utilizados actualmente y las limitaciones que tiene actualmente el lenguaje dentro de los patrones típicos de uso de XBRL para cubrir estas necesidades de expresar reglas de negocio y realizar validaciones avanzadas sobre los datos intercambiados.

Hasta la fecha la solución de la mayor parte de procesamiento de estas reglas de negocio queda relegado bien en las aplicaciones (por ejemplo utilizando XQuery como lenguaje de consulta), bien en implementaciones propias de estas reglas en esquemas adicionales XML o bien en estructuras de diseño de taxonomías que tratan de "exprimir" al máximo posible la especificación XBRL dada su flexibilidad:

Condiciones de existencia:

Para definir que el elemento A debe ser informado si aparece el elemento B

if exist(B) AND exist(A) AND (c-eq(A,B) AND u-eq(A,B)) then OK else ERROR

Otra opción podría ser utilizar la linkbase de definición con el role "requires-element":

If A requires-element B then OK else ERROR

Teniendo en cuenta la limitación de que al menos aparezca A para validarla

También se pueden definir elementos constantes en la taxonomía, que aparezcan en los informes XBRL de la siguiente forma:

<xsd:element id="my_CTE" type="xbrli:monetaryItemtype" fixed="xxx"/>

Donde xxx es el valor de la constante. Y así igualar valores reportados con sus limitaciones.

Condiciones de cálculo:

Multiplicar el valor de un concepto por una constante definida (ponderar un concepto)

Mediante la linkbase de cálculo puedo realizar multiplicaciones sin más que definir un arco de peso distinto de −1.0 ó 1.0: ej A*0.8 = A→weight="0.8"

Elaboración de condiciones más avanzadas:

If A > B check if exists(C)

A - B != 0 → true C != 0 true

If positiveInteger(A-B)=true then if nonZeroInteger(C) then true else error

Por todas estas necesidades nace la especificación de fórmulas XBRL como una ampliación del lenguaje XBRL que permita la declaración y publicación de un conjunto de fórmulas, para su procesamiento estandarizado.

4.8.2 Fórmulas XBRL

El presente capítulo pretende acercar a la problemática de formulación XBRL, desde su motivación inicial en el reporting descrita en los requerimientos de negocio de las fórmulas XBRL, hasta situar al lector en el marco actual de la especificación. Para ello resumiremos la estructura del borrador interno de las especificaciones de fórmulas y funciones, describiendo las distintas partes que la componen y daremos una orientación de cada una de ellas.

La especificación de Fórmulas XBRL nace tras la puesta en marcha de aplicaciones de intercambio de información en XBRL que demandan poder expresar reglas de negocio más avanzadas que las sumas aritméticas actualmente soportadas en la linkbase de cálculo.

A medida que estas aplicaciones producen y consumen informes XBRL, se detectan dos necesidades adicionales. Por un lado documentar en las taxonomías las relaciones de reglas de cálculo matemático avanzado existentes entre conceptos, y por otra parte declarar las relaciones que permitan realizar validaciones de consistencia de la información más avanzadas en función de los datos de los informes. Pasamos a continuación a describir ambos casos.

4.8.2.1 XBRL como forma de documentación estándar de reglas de negocio

El lenguaje XBRL desde sus comienzos expresa hechos producidos y además documenta las relaciones existentes entre los conceptos reportados.

Las taxonomías son una forma de documentar el significado de los conceptos reportados.

Mediante el mecanismo de linkbases se extiende el modelo de contenido de XML Schema disponiendo de categorías adicionales para expresar significado de los conceptos en forma de documentación:

- Label Linkbase: Etiquetas o literales con una amplia variedad de posibilidades e idiomas
- Reference Linkbase: referencias de importancia a la hora de encontrar soporte legal y escrito de los conceptos
- Presentation Linkbase: Declaran relaciones útiles por las aplicaciones consumidoras a la hora de clasificar jerárquicamente los conceptos estableciendo los padres e hijos en un informe presentado
- Calculation Linkbase: Declaran relaciones de sumas y restas sobre los conceptos.
- Definition Linkbase: Declaran relaciones más "semánticas" entre los conceptos como pueden ser especialización, similitud, pertenencia a un dominio de conocimiento, etc.

De esta forma, establecer una documentación entre conceptos que declare las relaciones de tipo formulado (cálculo complejas) que existen entre los conceptos, ayudará, de igual manera que hacen el resto de linkbases, a dotar a los conceptos de mayor significado.

Esto es especialmente útil a la hora de localizar dentro de un diccionario aquellos conceptos que participan en una fórmula, por ejemplo, un ratio de análisis financiero, a diferencia de aquellos que no participan.

4.8.2.2 XBRL para validar la información intercambiada

Una parte fundamental de las aplicaciones que producen y consumen información XBRL es realizar comprobaciones de consistencia de los datos intercambiados y sus posibles correcciones, proporcionando a todas las aplicaciones que participan en la cadena de intercambio (reporting) de información, datos suficientes para poder detectar los posibles errores o inconsistencias encontradas en los valores de los informes producidos.

Para este propósito la asociación XBRL Internacional, partiendo de los requerimientos de negocio identificados, está realizando una especificación de fórmulas que complemente la sintaxis actual de XBRL en su versión 2.1, y así poder definir reglas y validaciones de los datos de forma estandarizada e independiente de las aplicaciones que consuman los informes y poder automatizar más el consumo de informes XBRL, garantizando la calidad de los datos.

4.8.3 Requerimientos de fórmulas XBRL

Los requerimientos de fórmulas aquí comentados están basados en el documento:

Formula-Req-CR-2005-06-21.rtf

De forma general toda aplicación que utilice fórmulas XBRL debería ser capaz de cumplir los siguientes requerimientos:

Requerimientos de uso general

1. Comprobar la consistencia de los datos de los informes XBRL y emitir los fallos de una fórmula, de forma similar al funcionamiento actual de XBRL 2.1 en la linkbase de cálculo al reportar fallos de consistencia para los arcos de sumas "summation-item".
2. Comprobar la consistencia de informes XBRL y proporcionar mensajes detallados de cada fórmula que falle. En este caso, se considera que una salida general XML del procesador es lo más apropiado para el subsiguiente procesamiento de errores por la aplicación consumidora.
3. Crear un informe XBRL sin tuplas basado en otros documentos XBRL de entrada. En este caso, el procesador de fórmulas es sensible de producir un informe XBRL válido. Si las aplicaciones productoras verifican que su propia salida sera aceptada por aquellas aplicaciones consumidoras será posible obtener eficiencias significativas, sobre todo en un entorno distribuido como es Internet.
4. Transformar, adjuntar, o componer informes XBRL, incluyendo conceptos de tipo tupla donde el orden de creación es relevante a diferencia del caso 3.

Requerimientos relacionados con linkbases

1. Un DTS (conjunto referenciable de taxonomías) puede incluir definiciones de fórmulas. Es decir, debe ser posible definir un conjunto de fórmulas de tal forma que puedan formar parte de uno o más conjuntos referenciables de taxonomías (DTS's)
2. Las fórmulas podrían requerir el uso de diversos componentes de un DTS. Por ejemplo, las fórmulas pueden hacer referencia a items y tuplas que estén definidos en las taxonomías de las que forman parte o sobre las que aplican.
3. Las fórmulas podrían ser agrupadas dentro de conjuntos o familias
4. Una fórmula, en general, define en sí misma una relación entre dos o más hechos y por tanto un conjunto de formulas debe usar el mismo atributo xlink:role que se emplea en la especificación XBRL 2.1 para indicar mediante el role cual es la red de relaciones a utilizar.
5. Una fórmula puede incluir documentación, utilizando la sintaxis xlink de la especificación XBRL 2.1 para asociar recursos (etiquetas y/o referencias) a las fórmulas que se definan.
6. La declaración de fórmulas debería aparecer únicamente en linkbases asociadas con una taxonomía, de forma consistente al tratamiento de cálculos, y no con la instancia.

Requerimientos relacionados con el procesamiento:

A continuación se describen los requerimientos de la semántica de procesamiento de fórmulas relativa a las entradas y salidas XBRL.

1. El resultado de aplicar fórmulas a un informe que no es un documento XBRL válido no está definido

2. Los autores pueden escribir fórmulas de tal manera que esperen recibir un documento XBRL válido con respecto a un esquema taxonómico conocido.
3. La aplicación de un conjunto de fórmulas a un informe XBRL válido debe resultar en un documento XML bien formado tanto de resultado positivo como de fallo.
4. Cualquier número de fórmulas podrían calcular el valor de cualquier item. Las fórmulas podrían proporcionar distintas variantes para obtener un hecho. En la práctica la recomendación es que los autores eviten escribir fórmulas que se apliquen sobre el mismo conjunto de hechos para producir el mismo resultado, puesto que los hechos resultados podrían aparecer duplicados o incluso contradictorios.
5. Toda fórmula en un DTS debería ser procesada concurrentemente y sin excepción sin tener en cuenta la prioridad.

Requerimientos relacionados con las expresiones:

Identificaremos en este apartado tres tipos de expresiones:

Binding expressions: expresiones que asocien argumentos a hechos;
Precondition expressions: expresiones que puedan ejecutar una precondición booleana de una fórmula cuando sus argumentos se hayan encontrado; y
Result expressions: para expresiones que alberguen el resultado de procesar una fórmula.

1. Debería haber un único lenguaje para las expresiones.
2. El lenguaje de expresiones debería ser reconocible por un programador de nivel medio
3. El lenguaje de expresiones debería incluir operadores que puedan seleccionar cualquier nodo objetivo de asociación de variables que exista en documento instanciado.
4. Las expresiones podrían incluir constantes.
5. Las expresiones podrían incluir operaciones matemáticas

Requerimientos de asociación de hechos

1. Toda Fórmula podría filtrar los hechos de entrada a los que aplica de acuerdo a las relaciones que existan entre los hechos de uno o más contextos.
2. Los hechos asociados en una formula podrían tener diferentes contextos.
3. Las formulas podrían restringir los hechos sobre los que se aplican basándose en sus contextos. Este requerimiento también implica que el lenguaje de expresión proporcione una librería nativa para la manipulación de funciones de fecha tales como los especificados en XQuery y XPath [XQPFO].
4. Las fórmulas pueden restringir los hechos sobre los que aplican basándose en sus unidades y/o en su atributo de precisión o decimales según sea el caso.
5. Debería ser capaz de aplicar filtros en la entrada de los hechos dependiendo de la estructura intrínseca de una tupla.

Requerimientos de alertas

En lugar de producir únicamente un resultado Booleano, o indicar un fallo general como hemos descrito en la introducción, un procesador de fórmulas puede producir un aviso más detallado o una explicación de la ejecución realizada o del problema encontrado.

4.8.4 Especificación de fórmulas XBRL

En este apartado describiremos de forma resumida los trabajos del grupo XBRL Internacional basados en el documento interno del grupo:

Formula-Linkbase-Specification-IWD-2005-04-19.doc

Dicho documento es el resultado técnico del equipo de estandarización del grupo Specification dentro de XBRL International, realizado en base a los requerimientos anteriormente descritos para poder expresar mediante una linkbase de fórmulas la declaración de conceptos que expresen la problemática asociada a cálculos avanzados en el reporting XBRL.

Propósito

- Linkbase de fórmulas para realizar validaciones complejas sobre los datos reportados en el documento instanciado. Dichas validaciones pueden ser reglas de negocio como un Análisis de movimientos de Caja, por ejemplo.

- Linkbase de fórmulas para construir documentos instanciados complejos a partir de otros datos reportados más simples. Por ejemplo, reporting de conceptos de ratios a partir de valores financieros simples.

- Linkbase de fórmulas para resolver cálculos entre valores reportados en distintos contextos. Puede ocurrir que según el diseño de las taxonomías, exista una limitación a la hora de realizar comprobaciones de cálculo sobre los datos reportados si estos se encuentran englobados en diferentes contextos. Por ejemplo, en una taxonomía en la que hay una dimensión geográfica y/o de unidades de negocio sobre una taxonomía primaria financiera, el poder comprobar que los totales y subtotales a lo largo de la dimensión se cumplen implica poder transponer los cálculos del documento instanciado para aquellos contextos que se necesiten.

- Documentar las relaciones existentes entre los conceptos a los que se aplican reglas de negocio.

Estructura

La especificación está organizada en dos grandes bloques, el modelo implícito de procesamiento, donde se dan las recomendaciones de explotación de las fórmulas por un procesador, y así facilitar la interoperabilidad, y la sintaxis de la linkbase propiamente dicha, donde se detallan los fragmentos XML y atributos de los elementos para expresar fórmulas, de acuerdo a cada modelo de procesamiento.

4.8.4.1 Modelo de procesamiento General

Una fórmula en XBRL está expresada como un **recurso** (*xlink resource*).

Toda fórmula puede tener un conjunto de **variables** sobre las que utilizar datos que alimenten la fórmula. Por ejemplo, para la fórmula "*TotalActivo = ActivoMaterial + ActivoInmovilizado*" habría una variable para asignar los datos de "*ActivoMaterial*" y una variable para asignar los datos de "*ActivoInmovilizado*".

Puede ocurrir que el conjunto de variables de una fórmula resulte vacío, cuando sea el caso en el que los valores que produce una fórmula no se basen en ningún dato de entrada.

En las fórmulas se define el valor de **expresión** para indicar el resultado de evaluar la fórmula. Generalmente el valor de expression utiliza las variables como entradas para obtener el resultado, tras aplicarle una serie de **operadores o funciones**. Por ejemplo, para la formula "*TotalActivo = ActivoMaterial + ActivoInmovilizado*", los datos de "*ActivoMaterial*" serían asignados a la variable *$activoMaterial* y los datos de "*ActivoInmovilizado*" serían asignados a la variable *$activoInmovilizado*. El valor de la expresión para la fórmula sería "*$activoMaterial + $activoInmovilizado*" teniendo en cuenta el operador entre ambos (*suma*).

En general el motor de evaluación de una fórmula tendrá además en cuenta no sólo los operadores de expresión sino también el resto de **funciones relativas a las dimensiones** (tanto estándar como específicas de usuario). En el ejemplo anterior que los valores de las variables son *c-equal() y u-equal()* si no se especifica lo contrario.

De igual forma a como se documentan los conceptos en la taxonomía XBRL, las fórmulas pueden tener etiquetas y referencias (*label*, *reference*) para proporcionar literales legibles por las personas y referencias a literatura normativa acerca de las fórmulas.

Como **resultado** de evaluar fórmulas se puede indicar que se generen hechos XBRL. Este proceso se denomina **creación de hechos desde fórmulas**.

Las fórmulas se pueden utilizar para **producir mensajes** de comprobación de consistencia de los datos que se reciben en los documentos instanciados.

Las formulas pueden tener mensajes de procesamiento asociados con ellas. Éstos se utilizan para indicar una condición de error o informes internos de procesamiento de la fórmula.

Las fórmulas son procesadas dentro del alcance de un conjunto de taxonomías descubiertas (Discoverable Taxonomy Set, DTS). El DTS proporciona toda la información de soporte necesaria para la ejecución de la fórmula (localización de los conceptos del esquema, tipo de dato, relaciones, etc.).

4.8.4.2 Sintaxis de Linkbase de fórmulas.

Dentro del borrador interno de la especificación, se detallan los elementos que definen la sintaxis de la linkbase de fórmulas.

Para resumir la sintaxis sin entrar a detalle del borrador, podemos decir que se ha estructurado de forma similar a las demás linkbases de tipo jerárquico existiendo fragmentos XML (*formulaLink*) para englobar las fórmulas, sobre los que se especifican elementos de tipo Localizador (*Locator*), otros de tipo arco para establecer las relaciones (*formulaArc*), y otros de tipo recurso (*formula type xl:resource*), donde se especializan elementos y definen toda una serie de fragmentos XML de relevancia a la hora de realizar el procesamiento de fórmulas como son las variables (*Variables, itemVariable, tupleVariable*), los resultados (*instance*) y los mensajes (*message*).

Es importante en la ejecución de una fórmula la selección dentro del documento instanciado de los hechos que se asignan a una variable, teniendo en cuenta todas las características dimensionales del mismo (contexto, unidades, período temporal, otras dimensiones específicas), y también la utilización de filtros para los valores de los hechos. Para esta selección toda fórmula y variable tienen un atributo (*select)* que sirve para indicar una expresión en el lenguaje XPath para la evaluación de la variable en cuestión.

En ambos casos la especificación apunta a la utilización de fragmentos XML específicos dentro de una fórmula, que definen las variables y los filtros.

EJEMPLO DE VARIABLE

```
<df:itemVariable name="endOf2005">
  <df:period instant="2005-12-31"/>
</df:itemVariable>
```

EJEMPLO DE FILTRO GENERAL

Una variable de elemento item que filtra aquellos valores que estén por debajo de 110:

```
<df:itemVariable name="less110">
  <df:generalFilter filter=". < 110"/>
</df:itemVariable>
```

Dentro de la sintaxis de estos elementos se pueden declarar y utilizar funciones externas para su procesamiento, que realicen por ejemplo tareas de comprobación de valores de periodos, de unidades, o cualquier otra operación. Veremos en el siguiente apartado un resumen sobre el estado de las funciones más comunes que se están estandarizando de cara a la interoperabilidad de procesadores, y mantienen la declaración en la linkbase de fórmulas independientemente de su implementación.

Puesto que el procesamiento de las fórmulas aplica sobre documentos XBRL de entrada, es razonable pensar que un procesador de fórmulas puede producir como resultado documentos XBRL de salida y también mensajes resultado de las comprobaciones de consistencia de estos datos de entrada.

Para ello la especificación de fórmulas incluye elementos en la linkbase que declaran el resultado tanto de los conceptos producidos como de los mensajes resultado.

EJEMPLO DE MENSAJE DE PROCESAMIENTO

```
<f:message xlink:type="resource" xlink:label="formula_message">
      <f:onTrue>Aggregation is OK</f:onTrue>
      <f:onFalse>Aggregation do not match</f:onFalse>
</f:message>
```

Veamos a continuación a modo de ejemplo, unos fragmentos de lo que podría ser la linkbase de fórmulas en distintos casos de uso.

EJEMPLO DE CREACIÓN DE RESULTADOS EN FÓRMULAS.

```
<f:formulaLink xlink:type="extended" xlink:role="http://www.xbrl.org/2003/role/link"
        xmlns:f1="http://www.xbrl.org/2005/function/instance/check-element-exist"
        xmlns:acme="http://example.com/acme/results/2003-07-01"
        xmlns:m="http://www.xbrl.org/2005/function/instance/movement-analysis"
        xmlns:xbrli="http://www.xbrl.org/2003/instance">
        <!-- Declaración de módulo externo con funciones adicionales (en este caso en Xquery) -->
        <f:module xlink:type="resource" xlink:label="formula_label"
         namespace="http://www.xbrl.org/2005/function/instance/movement-analysis"
         url="xbrl-movement-analysis.xq"/>
        <!-- Declaración de un recurso fórmula que usa funciones del módulo anterior -->
        <f:formula xlink:type="resource" xlink:label="formula_resource"
         select="m:movement-is-zero(/xbrli:xbrl/acme:Cash,
                 QName('http://example.com/acme/results/2003-07-01', 'acme:IncDecCash'),
                 xdt:yearMonthDuration('P3M'))"/>
        <!-- Este arco conecta la fórmula con el recurso mensaje -->
        <f:formulaArc xlink:type="arc" xlink:to="formula_message" xlink:from="formula_label"
         xlink:arcrole="http://xbrl.org/int/formula/arcrole/instance-validation"/>
        <!-- Este es un mensaje con información acerca del resultado de ejecución -->
        <f:message xlink:type="resource" xlink:label="formula_message">
         <f:onTrue>Aggregation is OK</f:onTrue>
         <f:onFalse>Aggregation do not match</f:onFalse>
        </f:message>
        <!-- Este en un recurso de fórmula que produce un nuevo documento instanciado -->
        <f:formula select="xfi:create-instance($schemaRefs,$contexts,$units,$facts,())"
         xlink:type="resource" xlink:label="formula_resource_create_instance">
         <f:variable select="/xbrli:xbrl/link:schemaRef" name="schemaRefs"
                 bindAsSequence="true"/>
         <f:variable select="/xbrli:xbrl/xbrli:context" name="contexts"
                 bindAsSequence="true"/>
         <f:variable select="/xbrli:xbrl/xbrli:unit" name="units"
                 bindAsSequence="true"/>
         <f:variable select="/xbrli:xbrl/acme:revenueByGeographicSegment" name="facts"
                 bindAsSequence="true"/>
        </f:formula>
        <f:instance xlink:type="resource" xlink:label="instance_label"/>
        <f:formulaArc xlink:type="arc" xlink:to="instance_label"
         xlink:from="formula_resource_create_instance"
         xlink:arcrole="http://xbrl.org/int/formula/arcrole/instance-creation">
        </f:formulaArc>
</f:formulaLink>
```

EJEMPLO ANÁLISIS DE MOVIMIENTOS.

```
<f:formulaLink xlink:type="extended" xlink:role="http://www.xbrl.org/role/link"
        xmlns:acme="http://example.com/acme/results/2003-07-01"
        xmlns:m="http://www.xbrl.org/2005/function/instance/movement-analysis"
        xmlns:xbrli="http://www.xbrl.org/2003/instance">
        <!-- Declaración de módulo externo con funciones adicionales (en este caso en Xquery) -->
        <f:module prefix="m"
         namespace="http://www.xbrl.org/2005/function/instance/movement-analysis"
         url="xbrl-movement-analysis.xq"/>
        <!-- Declaración de un recurso fórmula que usa funciones del módulo anterior -->
        <f:formula xlink:type="resource" xlink:label="formula_resource"
         select="m:movement-is-zero(xbrli:xbrl/acme:Cash,
                 QName('http://example.com/acme/results/2003-07-01', 'acme:IncDecCash'),
                 xdt:yearMonthDuration('P3M'))">
        </f:formula>
        <link:loc xlink:type="locator" xlink:href="xbrl-formula-2006.xsd#onInstanceValidation"
         xlink:label="onInstanceValidation"/>

        <f:formulaArc xlink:type="arc" xlink:to="formula_resource" link:from="onInstanceValidation"
         xlink:arcrole="http://xbrl.org/int/formula/arcrole/instance-validation"/>
</f:formulaLink>
```

4.8.5 Módulo de Funciones XBRL

Los documentos de trabajo en los que se basa este apartado son:

XF-REQ-IWD-2005-04-23.doc	*Funciones. Requerimientos*
XF-IWD-2006-05-25.rtf	*Funciones. Especificación.*
xf-2006.xq	*Funciones. Implementación de referencia*

Siendo este último una librería con la implementación de referencia en lenguaje XQuery de las funciones que se detallan en la especificación XF.

El procesamiento XBRL y en particular el requerido para implementar la funcionalidad de la linkbase de fórmulas descrita en el apartado anterior, se soporta en una serie de operaciones comunes. Por ejemplo, determinar el final del período de un hecho puede ser visto como una funcionalidad común a más de una fórmula, por lo que sería razonable pensar en un conjunto de funciones para su implementación y reutilización.

La especificación XBRL 2.1 ya describe varios predicados de igualdad (sección 4.10) relativos a la identificación de items y tuplas duplicadas, que sirven actualmente a los procesadores para utilizar como funciones reutilizables en su procesamiento, a la hora de comprobar la consistencia de los informes XBRL. En concreto estos predicados son operadores

comparativos que trabajan sobre dos argumentos para devolver la evaluación de valor verdadero o falso:

TABLE 1. EQUALITY PREDICATE DEFINITIONS.

Argument Types	Predicates	Definition
Node	identical	Exactly the same XML node.
Sequence	s-equal, v-equal, c-equal, u-equal	Every node in one sequence is {s-equal, v-equal, c-equal, u-equal} to the node in the same position in the other sequence.
Set	identical, s-equal, v-equal, c-equal, u-equal	Set **X** is {identical, s-equal, v-equal, c-equal, u-equal} to set **Y** if: every node in set **X** can be paired with a node in set **Y** to which it is {identical, s-equal, v-equal, c-equal, u-equal} and the two sets have the same number of members.

De la misma forma que podemos identificar estas funciones, la especificación de funciones XF trabaja en extender a un conjunto variado de predicados de acuerdo con los requerimientos de procesamiento que aparecen tras la especificación de fórmulas.

4.8.5.1 Relación de Funciones por categorías:

Se lista a continuación la relación de las funciones que se encuentran en los documentos clasificadas por categorías:

Accessor Functions

Items->Decimals and precision

xfi:precision-of($item as xbrli:item) as xs:integer

Items->Contexts and units

xfi:context-of-item($item as xbrli:item) as xbrli:context

xfi:unit-of-item($item as xbrli:item) as xbrli:unit

Items->Units

xfi:unit-numerator($unit as xbrli:unit) as xbrli:measure*

xfi:unit-denominator($unit as xbrli:unit) as xbrli:measure*

xfi:measure-name($measure as xbrli:measure) as xs:QName

Contexts

Periods

xfi:is-period-duration($context as xbrli:context) as xs:boolean

xfi:is-period-forever($context as xbrli:context) as xs:boolean

xfi:is-period-instant($context as xbrli:context) as xs:boolean
xfi:period-start($context as xbrli:context) as xs:dateTime
xfi:period-end($context as xbrli:context) as xs:dateTime
xfi:period-instant($context as xbrli:context) as xs:dateTime

Segments and Entities
xfi:identifier-of-entity($context as xbrli:context) as xs:token
xfi:scheme-of-identifier($context as xbrli:context) as xs:anyURI
xfi:segment-content($context as xbrli:context) as node()*
xfi:scenario-content($context as xbrli:context) as node()*

Footnotes
DTS Related Functions
Facts and tuples
Predicados

Constructor Functions

Creating items
xfi:create-numeric-item($name as xs:QName, $value as xs:anyType, $context as xbrli:context, $unit as xbrli:unit, $precision as xs:integer?, $decimals as xs:integer?)
xfi:create-numeric-item($name as xs:QName, $value as xs:anyType, $context as xbrli:context, $unit as xbrli:unit, $precision as xs:integer?, $decimals as xs:integer?, $attrs as attribute()*)
xfi:create-non-numeric-item($name as xs:QName, $value as xs:anyType, $context as xbrli:context)
xfi:create-non-numeric-item($name as xs:QName, $value as xs:anyType, $context as xbrli:context, $attrs as attribute()*)
xfi:create-nil-item($name as xs:QName, $context as xbrli:context, $unit as xbrli:unit?)
xfi:create-nil-item($name as xs:QName, $context as xbrli:context, $unit as xbrli:unit?, $attrs as attribute()*)

Creating tuples
xfi:create-tuple($name as xs:QName, $facts as (xbrli:item | xbrli:tuple)*) as xbrli:tuple
xfi:create-tuple($name as xs:QName, $facts as (xbrli:item | xbrli:tuple)*, $attrs as attribute()*) as xbrli:tuple
xfi:create-wrapped-tuple($name as xs:QName, $instance as xbrli:xbrl) as xbrli:xbrl
xfi:create-wrapped-tuple($name as xs:QName, $instance as xbrli:xbrl, $attrs as attribute()*) as xbrli:xbrl

Creating contexts
xfi:create-start-instant-context($context as xbrli:context) as xbrli:context
xfi:create-end-instant-context($context as xbrli:context) as xbrli:context

xfi:create-duration-from-instant-contexts($left as xbrli:context, $right as xbrli:context, $id as xs:string) as xbrli:context

xfi:create-forever-context($identifier as xs:token, $scheme as xs:anyURI, $segment as node()*, $scenario as node()*, $id as xs:string) as xbrli:context

xfi:create-instant-context($instant as (xs:date | xs:dateTime), $identifier as xs:token, $scheme as xs:anyURI, $segment as node()*, $scenario as node()*, $id as xs:string) as xbrli:context

xfi:create-duration-context($startDate as (xs:date | xs:dateTime), $endDate as (xs:date | xs:dateTime), $identifier as xs:token, $scheme as xs:anyURI, $segment as node()*, $scenario as node()*, $id as xs:string) as xbrli:context

Creating units

xfi:create-product-unit($units xbrli:unit*, $id xs:string) as xbrli:unit

xfi:create-inverse-unit($unit xbrli:unit, $id as xs:string) as xbrli:unit

xfi:create-unit($numerator as xs:QName+, $denominator as xs:QName*, $id as xs:string) as xbrli:unit

Creating instances

xfl:create-schema-ref($location as xs:anyURI) as xbrll:schemaRef

xfl:create-linkbase-ref($location as xs:anyURI) as xbrll:linkbaseRef

xfl:create-linkbase-ref($location as xs:anyURI, $role as xs:string) as xbrll:linkbaseRef

xfi:merge-instances($xbrls as xbrli:xbrl*) as xbrli:xbrl

xfi:create-instance($refs as xl:simple+, $facts as (xbrli:item | xbrli:tuple)*, $contexts as xbrli:context*, $units as xbrli:unit*)

Taxonomy Functions

4.9 Versionado

Las taxonomías pueden cambiar a lo largo del tiempo debido a diferentes factores, entre ellos:

- Cambios en la normativa legal que origina el reporting: Circulares, Normativas Contables, etc.
- En el caso de ser una taxonomía extendida de otra, un cambio en la taxonomía superior, puede originar un cambio.
- Cambios debidos a incidencias correctivas originadas por el uso tecnológico de las taxonomías.
- Cambios en la normativas de Buenas Prácticas sobre las que se soporta (FRTA, ISO, etc.).
- Cambio en las versiones de la especificación XBRL y de las especificaciones XML de soporte (XLink, XML Schema, etc.).

El propósito del versionado de taxonomías consiste en permitir la comparación, el análisis y la correcta identificación de los datos que han cambiado en los documentos XBRL que forman las taxonomías. Para ello, es necesario documentar correctamente la información sobre las diferencias existentes entre ellas.

Los usuarios de la taxonomía que automatizan sus procesos de reporte, son afectados por los cambios de versión, en el sentido de que podrían necesitar efectuar cambios en sus sistemas internos. La documentación de los cambios en la taxonomía permite valorar los cambios realizados entre versiones y saber que elementos cambian y/o desaparecen.

Respecto al versionado se consideran buenas prácticas:

- Documentar los cambios realizados entre dos versiones distintas de la taxonomía, identificando cada cambio o revisión con un código unívoco al que poder hacer referencia en otros documentos. (Listas de Control de Cambios).
- Cambiar el namespace con cada versión nueva de una taxonomía. El namespace es el identificador biunívoco de la versión de la taxonomía. Dos taxonomías de dos versiones diferentes NO DEBEN tener el mismo namespace
- Identificar los cambios en elementos del diccionario de datos de la taxonomía con códigos como (copiado, eliminado, renombrado, movido, etc.) en la hoja de control de cambios
- Describir la fecha de lanzamiento de la taxonomía en el nombre de los ficheros de los documentos, y además en identificador de los namespaces.
- Identificar la versión en la cabecera de los documentos XML de la taxonomía: Diccionario de datos y linkbases referenciados
- Identificar exactamente qué elementos de los documentos de la taxonomía: linkbases y diccionario de datos han sido afectados por los cambios de versiones
- Especificar sobre qué versiones de normativas (FRTA Candidate Recomentation 5, FRIS, etc.) y especificaciones (XBRL 2.1., Xlink 1.0) está diseñada la nueva versión.
- En el caso de organismos regulatorios, poner a disposición de los usuarios un sitio público con un repositorio de las taxonomías donde se pueda consultar la vigencia de la versión de la taxonomía y disponer un repositorio de todos las taxonomías emitidas
- Asegurar la estabilidad, sin variaciones, de la taxonomía durante un periodo aconsejado de un ejercicio.

5 Productos y Servicios

A la hora de implantar XBRL en una organización, el mercado ofrece muchas soluciones y herramientas específicas para el tratamiento del lenguaje XBRL, además de las ya disponibles para trabajar con tecnologías XML. Estas herramientas pueden variar en función del nivel de la solución XBRL a implantar, y por tanto del uso o funcionalidad requerida. Por ejemplo:

- Herramientas específicas para la edición de taxonomías, generación de instancias, etc.
- Herramienta de desarrollo de aplicaciones a medida para el procesamiento de documentos XBRL.
- Plataforma global de reporting financiero XBRL para la generación, administración, validación, gestión y almacenamiento de documentos XBRL, así como la propia definición y generación de los informes.

Algunos ejemplos de los principales productores de herramientas XBRL:

Productos	Crear y Editar Taxonomías XBRL	Crear y Editar Instancias XBRL	Extracción de información de las instancias XBRL	Validación de Instancias XBRL	Validación de Taxonomías	Representación en Pantalla de XBRL (PDF, XML, HTML)	Conversión a Versión 2.0 -> 2.1	Control de Versión Integrado	Gestión de un repositorio interno	Integración Excel	Comparación entre taxonomías
Batavia XBRL		X	X	X		X					
Blast Radius (XMetal) XBRL Office Express		X	X	X	X					X	
Blast Radius (XMetal) XBRL Web Express				X	X	X					
DecisionSoft – True North Validator				X	X						
Edicom – Business Integrator XBRL	X	X		X	X	X		X	X		
Fujitsu Interstage XBRL Processor		X	X	X							
Fujitsu Business Rule Editor	X					X				X	
Fujitsu Converter (Tool List v2.1)							X				
Fujitsu Intance Creator (Tool List v2.0)		X								X	
Fujitsu Intance Creator (Tool List v2.1)		X				X				X	
Fujitsu Taxonomy Diff	X										X
Fujitsu Taxonomy Editor (Tool List v2.0)	X										
Fujitsu Taxonomy Editor (Tool List v2.1)	X				X					X	
Fujitsu Validator (Tool List v2.0)				X	X						
Fujitsu Validator (Tool List v2.1)				X	X						
Fujitsu Versioning & Mapping			X					X			
Fujitsu XBRL Adapter		X								X	
Fujitsu XBRL Manager	X				X			X	X	X	
Fujitsu XBRL Sheet Mapping		X								X	
Hitachi (XiRUTE) Taxonomy Editor	X				X					X	
Hitachi (XiRUTE) Instance Creator		X	X	X						X	
Hitachi (XiRUTE) Instance Viewer		X		X						X	
Hitachi Xinba		X	X							X	
Rivet Software Dragon Tag		X		X						X	
Semansys XBRL Composer	X	X		X	X						
Semansys XBRL Builder	X						X				
Semansys XBRL Integrator			X								
Semansys XBRL Analyzer			X								
Software AG - Digital Reporting Platform			X	X	X	X		X	X	X	
UB Matrix Taxonomy Manager											
UB Matrix Reporting Manager											
UB Matrix XBRL Processing Engine											
UB Matrix Advanced Processing Module											
UB Matrix XBRL Taxonomy Designer	X	X		X	X	X	X				
UB Matrix XBRL Translator Module											

Batavia www.batavia-xbrl.com

Batavia XBRL Products son una serie de productos para XBRL basados en Java. Editor Online de instancias, API Java para desarrollo sobre XBRL y la herramienta ReportLink de integración XBRL.

Blast Radius www.blastradius.com

Ha desarrollado flexible v2.1-compliant XBRL processor y otros productos asociados como XBRL Web Express (valida online instancias), XBRL Office Express para importar, crear, validar y exportar instancias XBRL a MS Excel. Y XMetal customization para hacer reporting financiero usando XBRL.

DecisionSoft www.decisionsoft.com

Dispone de productos y servicios para implementar XBRL, como True North para procesamiento y validación XBRL. También dispone de un API para desarrollo en XBRL.

Edicom www.edicom.es/xbrl

Edicom Bussines Integrator XBRL Edition. Una herramienta que cubre íntegramente todo el proceso transaccional: EBI, un completo EAI que se ocupará de implementar y gestionar el bus de datos efectuando las transformaciones necesarias (Mapping) así como la validación de instancias, un potente administrador de comunicaciones: gestión de interlocutores, seguridad (firma y encriptación), históricos, .. y un editor de taxonomías con soporte a linkbases.

Fujitsu software.fujitsu.com/en/interstage-xwand/activity/xbrltools/index.html

Ofrece amplia gama de herramientas para crear, editar taxonomías e instancias, procesamiento de XBRL, validación e integración XBRL.

Hitachi www.hitachi.co.jp/XBRL

XiRUTE ToolSet ofrece un API para desarrollar con el estándar XBRL DOM con la misma interfaz que el W3C DOM.

Microsoft www.microsoft.com/latam/office/solutions/xbrl/overview.mspx

Ofrece un prototipo de Herramienta de Ofiice para XBRL que puede utilizar Microsoft Office Word 2003 y Microsoft Office Excel 2003 para crear y analizar documentos en formato XBRL.

Rivet Software www.rivetsoftware.com

Dragon Tag 1.0 es una herramienta para crear fácilmente informes XBRL a partir de documentos MS Word y Excel.

Semansys Technologies BV www.semansys.com

Ofrece soluciones como Next Generation XBRL productivity tools, para creación, análisis y procesamiento de taxonomías. Ofrece también soluciones de integración XBRL y reporting financiero.

Software AG España www.softwareag.com/es

Ofrece soluciones para reporting, validación y análisis sobre XBRL. Su framework de desarrollo Digital Reporting Platform permite desarrollar proyectos de reporting XBRL.

Ubmatrix www.ubmatrix.com
Ofrece soluciones para reporting, validación y análisis sobre XBRL. Automator XBRL Professional permite crear taxonomías, UBS es un servidor Web para integración XBRL junto con XBRL Converter, y también ofrece un API para desarrollar sobre XBRL llamado XBRL ToolKit.

En la siguiente tabla se pueden apreciar algunas de las funcionalidades que soportan las herramientas XBRL. Se incluye una muestra de productos que ofrece el mercado.

Funcionalidad a evaluar:

Crear y editar taxonomías: Ayuda a la visualización, creación y modificación de taxonomías.
Crear y editar instancias XBRL: Permite crear y editar instancias XBRL que corresponden a una taxonomía.
Extracción de información de las instancias XBRL: Permite procesar la instancia para buscar información en ella y recogerla.
Validación de instancias XBRL: validación sintáctica conforme a especificación XBRL (conformidad mínima) y/o semántica conforme a la taxonomía (completa).
Validación de Taxonomías: validación sintáctica de la taxonomía.
Representación en pantalla de XBRL (XML-HTML): Es capaz de mostrar un informe HTML conforme a las linkbases de presentación.
Conversión a versión 2.0 -> 2.1: Capaz de convertir un documento de la especificación XBRL 2.0 a la 2.1.
Control de versión integrado: Mantiene una historia de los documentos XBRL usados, y de los cambios realizados en ellos.
Gestión de un repositorio interno: Capacidad para almacenar documentos XBRL en un repositorio.
Integración Excel: Capacidad para exportar / importar a MS Excel.
Comparación entre taxonomías: Permite ver diferencias entre una taxonomía y otra.

6 Formación

Tecnológicamente XBRL hace uso intensivo de las especificaciones que sobre XML se han ido construyendo. En los capítulos 3 y 4 del presente libro se han detallado conceptos básicos sobre XML y XBRL.

En este capítulo sólo glosaremos la estructura de conocimientos que de XML y las especificaciones accesorias, se debe poseer para afrontar un proyecto XBRL y la pauta básica de adquisición de estos conocimientos.

XBRL España ha elaborado un documento específico de Formación y Buenas Prácticas en XBRL, que se encuentra accesible en la Web de XBRL España, en el que se trata ampliamente las necesidades de formación en un proyecto XBRL.

Algunas empresas que ofrecen servicios de formación XBRL:

- Azertia http://www.azertia.com
- Edicom http://www.edicom.es/xbrl
- Fujitsu http://www.fujitsu.com/es/services/sectors/bank/solutionsbank/solop.html
- PricewaterhouseCoopers http://www.pwc.com/xbrl
- Software AG http://www.softwareag.es/institute

6.1 Tecnologías

XBRL ha utilizado la especificación XML Schema para poder construir la estructura de definición de elementos, que compone la taxonomía.

XML Schema, se puede encontrar en la sede de la organización del W3C, (http://www.w3.org/XML/Schema). De acuerdo a la información que la sede ofrece sobre la especificación, XML Schema permite expresar vocabularios, definir estructuras, contenido y "semánticas" para documentos XML. XML Schema fue aprobado como recomendación del W3C el 2 de mayo de 2001.

La especificación se estructura en dos documentos:

> **XML Schema Part 1**: Structures contiene el núcleo de la especificación, describe los mecanismos que XML Schema provee para poder representar estructuras XML, se puede acceder a él, en idioma inglés, en el siguiente vínculo http://www.w3.org/TR/xmlschema-1/ .
>
> **XML Schema Part 2**: Datatypes complementa el documento anterior con la definición de todos los tipos de elementos que son utilizables por la especificación, es accesible en idioma inglés en el vínculo http://www.w3.org/TR/xmlschema-2/

La especificación de espacios de nombres (en adelante Namespaces) se utiliza en tecnologías XML, para resolver los problemas de colisión en los documentos XML, documentos que contienen elementos con el mismo nombre pero diferente significado. XBRL la utiliza ampliamente.
Se puede consultar en: http://www.w3.org/TR/REC-xml-names/.

XBRL utiliza la especificación XLink para permitir las relaciones entre los elementos definidos en el Esquema y su utilización como recurso en las Linkbases XBRL. La especificación XLink se encuentra accesible en W3C, http://www.w3.org/XML/Linking

XPath es una especificación del W3C, que permite ser utilizado en XBRL, para procesar documentos XML obteniendo partes del documento. Es un lenguaje de consulta. La especificación se encuentra en http://www.w3.org/TR/xpath. XPath es una tecnología candidata a ser usado en la Linkbase de fórmulas.

XQuery es una especificación con status de Working Draft, que se utiliza para implementar intérpretes de lenguajes de consulta a los documentos XML.
Se puede localizar en http://www.w3.org/TR/2005/WD-xquery-20050404/.

6.2 Tecnologías de Manipulación XML

Los documentos XML necesitan ser manipulados para obtener la información que contienen. XML como especificación contiene información suficiente como para construir aplicaciones informáticas que puedan procesar la información contenida en ellos. Estas piezas de software se conocen como Intérpretes o "Parsers".

Los intérpretes, deberán comprobar que los documentos XML están bien formados, de acuerdo a la especificación. Y, si poseen información sobre su esquema XML podrán ser validados.

Existen dos tipos de intérpretes, Interpretes DOM e Interpretes SAX a continuación se describen ambos brevemente:

Intérpretes DOM (Document Object Model)

> Los intérpretes DOM transforman el documento XML, que está en forma textual en una estructura de árbol. Esta estructura es jerárquica y los intérpretes proveen de mecanismos (métodos) para navegar por el árbol. DOM es una especificación que se puede recuperar de la sede del W3C http://www.w3.org/DOM/.

Intérpretes SAX (Simple API for XML)

> Los intérpretes SAX, utilizan mecanismos de interpretación basados en eventos. El intérprete SAX informa a las aplicaciones que lo utilizan, de la ocurrencia de un determinado evento, transfiriendo el control a la aplicación. Más información sobre SAX en http://www.saxproject.org/.

Existen intérpretes en todas las tecnologías de la información, Java, .Net, PHP, etc. Entre ellos los siguientes:

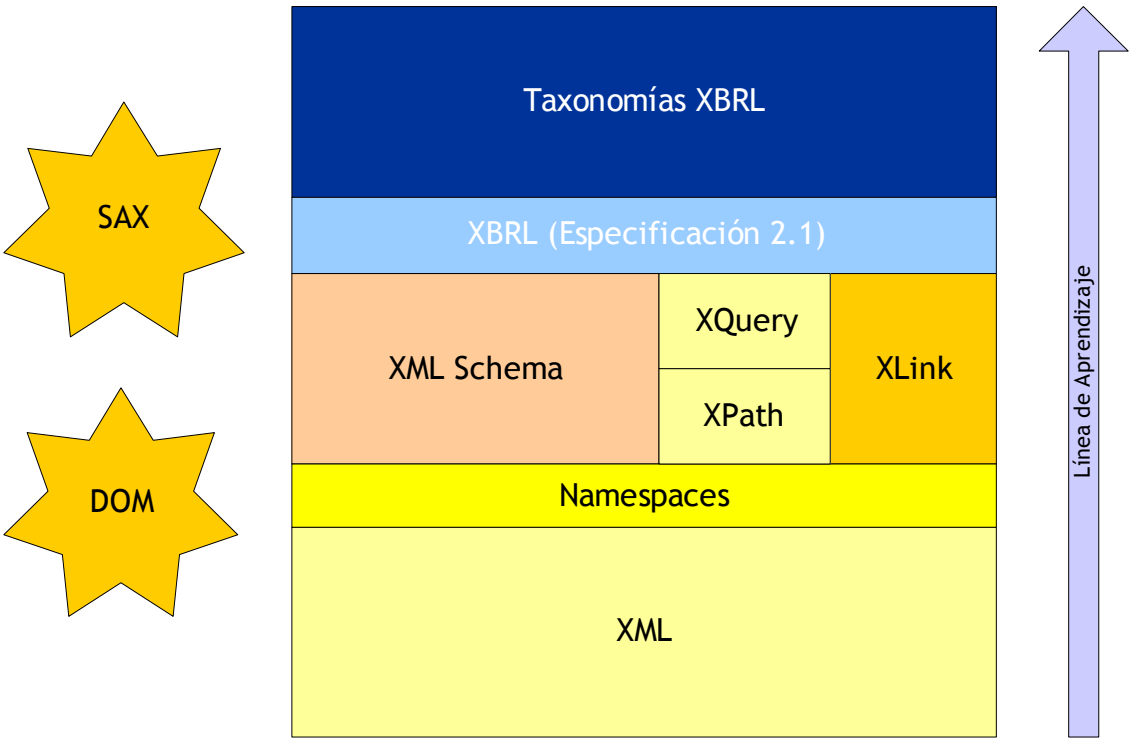
- Xerces (Apache)
- XJ (Data Channel)
- MSXML (Microsoft)

Los parsers XBRL pueden utilizar estos modelos de interpretación, añadiendo a las capacidades de los mismos, las necesidades especificas de XBRL.

Las tecnologías XML también poseen manipuladores que permiten la transformación de un documento XML a diferentes formatos. Esta tecnología es XSL. XSL es un lenguaje de comandos XML que un intérprete, el procesador de hojas de estilo, utiliza para procesar el documento XML. El resultado de este proceso puede ser otro documento XML o un tipo de archivo que es capaz de ser interpretado por otras aplicaciones, PDF, CSV, XHTML, etc. Estas tecnologías se pueden utilizar en la representación de instancias XBRL.

6.3 Esquema de Aprendizaje

A continuación se describe un esquema de Línea de Aprendizaje en XBRL.



7 Material de referencia y ejemplos

7.1 Esquema general de casos de uso e implementación de un sistema XBRL siguiendo una arquitectura orientada a servicios

Este ejemplo muestra una serie de roles y patrones de uso e implementación de una arquitectura XBRL que ha sido determinada con objeto de posibilitar una infraestructura automatizada de recepción de informes XBRL, definición y distribución de taxonomías y transformación de informes XBRL a formatos específicos de aplicaciones informáticas de back office.

Sin ánimo de realizar un trabajo exhaustivo dada la escasa experiencia en el uso de XBRL y con objeto de delimitar el ámbito de actuación se empezó por determinar los casos y roles de uso de XBRL implicados en el sistema y que, al margen de roles menores, fueron:

- Definición de taxonomías XBRL
- Construcción de taxonomías XBRL
- Emisión de informes XBRL
    - Emisión automatizada
    - Emisión manual
- Recepción de informes XBRL
- Recepción automatizada
- Recepción manual
- Análisis de información

Los dos primeros roles son realizados por usuarios que analizan los requerimientos de negocio y definen las necesidades de intercambio de información, la normalización de la

misma y la implementación de esos requerimientos en un modelado XBRL, en definitiva **el modelo de datos y su implementación**.

La experiencia ha demostrado que, si bien, para el trabajo de análisis de datos se requieren conocimientos profundos del negocio, la implementación en una terminología XBRL debe contar con especialistas en análisis de sistemas de información y el uso de herramientas propias del desarrollo de software:

- Análisis y normalización de datos
- Definición del modelo de datos
- Requerimientos versionado de taxonomías en desarrollo
- Requerimientos de trabajo en grupo
- Pruebas y validación de taxonomías en desarrollo
- Requerimientos de entornos de desarrollo, preproducción y producción
- …

En sentido estricto, estos dos primeros roles son los "Diseñadores XBRL" y necesitan herramientas propias de desarrolladores de software, adaptadas, en algunos casos, a la tecnología XBRL.

El resto de roles están directamente relacionados con el tratamiento de la información operativa, tanto desde el punto de vista de los emisores de este tipo de información como de los receptores y analistas de la misma y son los que de forma más propia pueden ser llamados "Usuarios XBRL".

A partir de la determinación de intervinientes en el sistema de información se definieron los servicios necesarios y las características de su implementación:

- Arquitectura informática orientada a servicios
- El sistema ha de ser altamente escalable y extensible ya que el volumen de la información intercambiada, almacenada y con necesidades de análisis es potencialmente enorme.
- Aislamiento y encapsulación de las especificidades de la tecnología XBRL para las aplicaciones informáticas de back end.
- Independencia de la especificación XBRL con objeto de que modificaciones en la misma no impacten en los sistemas informáticos diseñados.
- Independencia de canales de entrada y salida de informes XBRL con la propia infraestructura XBRL.
- Incorporación de los servicios XBRL en la arquitectura general de servicios de infraestructura generales.
- …

Siguiendo estas directrices se definieron una serie de servicios centrales y se dotó de herramientas especializadas a los intervinientes en el sistema que permitiera de forma cómoda la consecución de análisis y generación de las taxonomías.

Los servicios centrales se dirigieron principalmente a la automatización en el tratamiento de informes XBRL en sus diversas facetas (generación de informes XBRL, validación de informes XBRL, emisión y recepción segura de la información –cifrado y descifrado de la información-, almacenamiento de informes XBRL,…).

Los servicios desarrollados han sido:

- Servicios de taxonomías
    - Desarrollo de taxonomías

  - Publicación y suscripción de taxonomías
- Servicios de prevalidación y validación de informes XBRL
- Servicios de transformación de informes XBRL a formatos amigables.
  - Por role, idioma, …
  - A fichero plano, a formatos específicos de back office,..
- Servicios generales de infraestructura
  - Seguridad (Autenticación, firma electrónica, cifrado y descifrado, …)
  - Servicios de orquestación de procesos e integración de aplicaciones
  - Informes de usuario (HTML, PDF, Excel, …)
  - Servicios de persistencia de datos (Bases de datos de taxonomías y DTSs, base de datos de XBRL Reports.
  - Utilización del resto de servicios generales de la instalación.

Igualmente y teniendo en cuenta la bisoñez de la tecnología y el requerimiento político del proyecto de ser un acelerador y aglutinante en la implementación de la misma, se decidió externalizar parte de los servicios para su uso por los terceros emisores de información, permitiéndoles el acceso a los servicios de prevalidación, validación y transformación de informes XBRL sin que incurrieran en las inversiones que esta tecnología requiere.

El siguiente diagrama muestra la arquitectura de alto nivel de los servicios centrales del sistema.