2.5.- SAX/DOM.

2.5.1. Procesamiento de documentos XML.

Un *analizador* o *parser XML* es una herramienta encargada de leer documentos XML [17], poder acceder a sus elementos y comprobar si el documento es sintácticamente válido. Estas herramientas son módulos, bibliotecas o programas que se ocupan de transformar un archivo XML en una representación interna. Entre esos analizadores o parsers cabe destacar:

- SAX (Simple API for XML): se encarga de recorrer la estructura del documento generando eventos que corresponden a los elementos que se va encontrando.
- DOM (Document Object Model): representa el archivo en una estructura tipo árbol que usará para leer el documento.

SAX y DOM permiten analizar el lenguaje XML y definir la estructura de un documento. La validación del documento XML consiste en comprobar que el documento, además de estar bien formado de acuerdo a las reglas de XML, responde a una estructura definida en una Definición del Tipo de Documento (DTD) o en un XMLSchema.

Este procesamiento del documento XML se subdivide en dos fases:

1) Procesamiento de entrada XML:
   - Analizar y validar.
   - Reconocer/buscar información importante basándonse en su localización o en su etiquetado.
   - Extraer la información importante una vez que se ha localizado.
   - Opcionalmente, mapear/unir la información recuperada a objetos de negocio
2) Precesamiento de salida XML:
   - Construir un modelo del documento.
   - Aplicar hojas de estilo XSLT o serializar directamente a XML.

Estas fases son diferentes en SAX y en DOM. La Tabla 36 muestra las fases para cada uno de los dos analizadores:

Procesamiento	Fase	SAX	DOM
Entrada XML	Analizar y validar	Interno	Interno o basado en SAX
	Reconocer y buscar	Capturar eventos con manejadores de eventos	Buscar en el árbol con buscadores.
	Extraer	Capturar eventos	Obtener valores de atributos, contenidos de nodos: métodos del API
	Mapear/unir	Crear objetos de negocio desde la información extraída	Crear objetos de negocio desde la información extraída
Salida XML	Construcción	No hay soporte por defecto.	Parte implícita del modelo: API de métodos factoría
	Serialización	No hay soporte por defecto, pero puede hacerse con manejadores de eventos	Soporte de implementación específico, o a través de transformaciones XSLT.

Tabla 36.- Fases del procesamiento de un documento XML para SAX y DOM.

2.5.2. API SAX.

2.5.2.1. Introducción: modelo de la API SAX.

Simple API for XML (SAX), es una interfaz simple que se encarga de procesar o analizar la información del documento XML por eventos. SAX lee el documento secuencialmente de principio a fin, sin cargarlo en memoria, de forma que cuando encuentra un elemento se encarga de lanzar su evento sociado. Cuando el evento es lanzado éste puede ser capturado para realizar una función determinada. Esta API está definida en el paquete: *javax.xml.parsers.*

Para que estos eventos se puedan capturarse y realizar las operaciones que se deseen se debe usar un manejador de eventos. Un manejador es una clase con una serie de métodos y cada método se ejecutará cuando el analizador capture su evento asociado. Estos eventos se producen al leer un documento (al comienzo del documento, apertura o cierre de un elemento, al encontrar una instrucción de proceso o un comentario, etc.).

SAX 2.0 define cuatro interfaces básicas de manejadores de eventos:

- ContentHandler: se utiliza para tratar eventos generales del documento, como apertura y cierre de etiquetas o cuando aparecen bloques de texto.
- DTDHandler: invocado para tratar eventos relacionados con las DTD's.
- EntityResolver: se utiliza para resolver referencias a entidades externas.
- ErrorHandler: maneja los errores y warnings.

Para poder hacer uso del analizador primero se debe obtener una instancia de una factoría de analizadores (SAXParserFactory). Con esta factoría se crea el analizador (SAXParser, *javax.xml.parsers*) que encapsula un objeto de la interfaz

Parser (*org.xml.sax*). A este analizador se le asocia el/los manejador/es, que poseen los métodos que se deben ejecutar al capturar un evento lanzado por el analizador. Y, por último, se le pasa al analizador el documento para empezar a leerlo y validarlo.

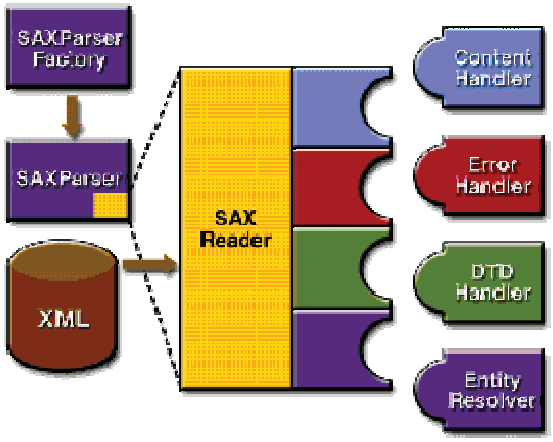

Ilustración 18.- Modelo de la API SAX.

Los métodos del manejador no están implementados, luego se tendrán que codificar aquellos que se quieran utilizar. Representan la lógica que se debe ejecutar al llevarse a cabo cada evento, entre ellos destacan:

Método	Lanzamiento del evento
startDocument()	Invocado cuando comienza un documento.
endDocument()	Invocado cuando finaliza un documento.
startElement(nombre, atributos)	Invocado cuando se abre una etiqueta.
endElement(nombre)	Invocado cuando se cierra una etiqueta.
characters(texto)	Invocado al leer el texto contenido en cada elemento.

Tabla 37.- Métodos del manejador.

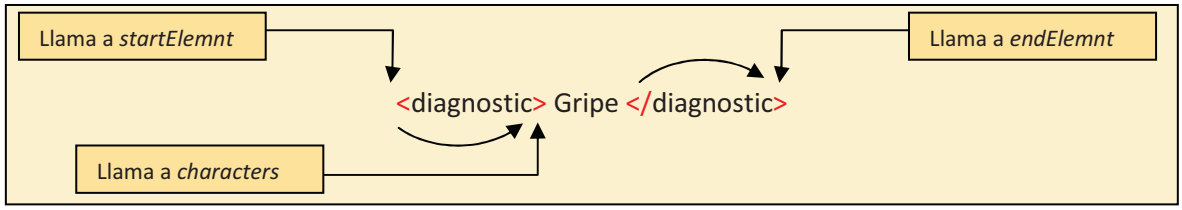

Ilustración 19.- Ejemplo de llamada de eventos en una línea de un documento XML.

Lo explicado anteriormente es lo que se ha usado en este proyecto para realizar la validación y lectura de los ficheros de configuración y de GPC así como la validación de los documentos XML de Guías de Práctica Clínica con respecto a un XML-Schema.

2.5.2.2. Características de la API SAX.

Las características generales que definen la API SAX se resumen en:

- SAX define una API para un analizador basado en eventos.

- El analizador lee un documento XML desde el principio hasta el final, y cada vez que reconoce una síntaxis de construcción, se lo notifica a la aplicación que lo está ejecutando.
- Procesamiento de un documento fuente como un stream de eventos.
- Los eventos son disparados mientras se analiza el documento como un flujo continuo de retrollamadas e invocaciones a métodos.
- Los eventos están anidados de la misma forma que los elementos en el documento, por lo tanto, no se crea ningún modelo de documento intermedio.
- Es más eficiente en cuanto al tiempo y la memoria empleados en el análisis ya que SAX no necesita la creación de un modelo de objeto como DOM.
- Permite la validación de un documento XML.

Estas características hacen que SAX sea apropiado para la lectura y validación de documentos en los que sólo sea necesario procesarlo una sola vez, ya que SAX no crea un modelo y lo mantiene en memoria para su posterior uso. También es ideal en el caso de tener archivos de gran tamaño ya que lee el documento sin ocupación de memoria.

SAX, con estas características, es suficiente para las labores de validación del documento XML de GPC y de los ficheros de configuración así como para la lectura de estos últimos. Por ello SAX con un menor gasto de memoria y mayor rapidez es la mejor elección para el uso en este servicio de gestión.

2.5.3. API DOM.

2.5.3.1. Introducción: modelo de la API DOM.

El API "Document Object Model" (DOM) es un conjunto de interfaces que describen una estructura abstracta para un documento XML. DOM carga el documento XML entero en memoria con una estructura tipo árbol. Cada elemento del documento XML se representa con un nodo (DOMNode). DOM está definido en los paquetes *org.w3c.dom* y *javax.xml.parsers.*

El árbol jerárquico de información en memoria permite que a través del manejador pueda manipularse la información: crear o eliminar información de un nodo en cualquier punto del árbol, acceder o cambiar su contenido y mover la herencia de nodos.

Para poder hacer uso del parser se debe obtener una instancia de una factoría analizadora (DocumentBuiderFactory). Con esta factoría se crea el analizador (DocumentBuilder) que es capaz de producir un nodo Document que cumple la especificación DOM, es decir, crear el inicio del árbol. Se puede crear un nodo Document vacío con el método *newDocument()* o crear el árbol completo de un

documento XML pasándole éste al analizador con el método *parser(documento_XML)*. A este analizador se le asocia/n el/los manejador/es, que indicarán las operaciones a realizar al capturar un evento lanzado por el analizador. Estas operaciones a realizar se encuentran definidas en sus métodos. Y, por último, se le pasa al analizador el documento para empezar a leer el documento y validarlo.

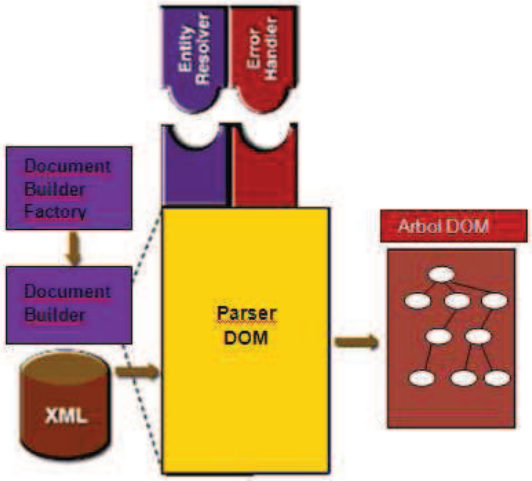

Ilustración 20.- Modelo DOM.

2.5.3.2. Características de la API DOM.

Las características principales de la API DOM son las siguientes:

- DOM representa en memoria el documento XML mediante una estructura tipo árbol.
- Cada elemento del documento XML se representa con un nodo dentro del árbol.
- Se proporcionan gran variedad de funciones para navegar a través del árbol DOM.
- Permite manipular el árbol en memoria, añadiendo un nuevo elemento o eliminando uno existente, actualizándolo o únicamente consultarlo.
- Permite la validación de un documento XML.

DOM permite disponer de la estructura del documento XML en memoria, luego es apropiado para el manejo de documentos XML que no sean de gran tamaño, ya que implicaría un gasto de memoria considerable. Está orientado a aplicaciones en las que se quiere consultar el documento varias veces o incluso modificarlo gracias, también, a que el árbol permanece en memoria. Sin embargo hay que tener en cuenta que el almacenamiento del documento XML en memoria mediante la estructura en árbol requiere de un coste en tiempo adicional además del coste en memoria.

2.5.4. Diferencias entre SAX y DOM.

La principal diferencia entre DOM y SAX es que mientras el primero tiene acceso al documento completo, es decir, que todos los elementos y atributos están disponibles a la vez, en SAX sólo está disponible el elemento actual. En las siguientes figuras se muestra el proceso de cada analizador:

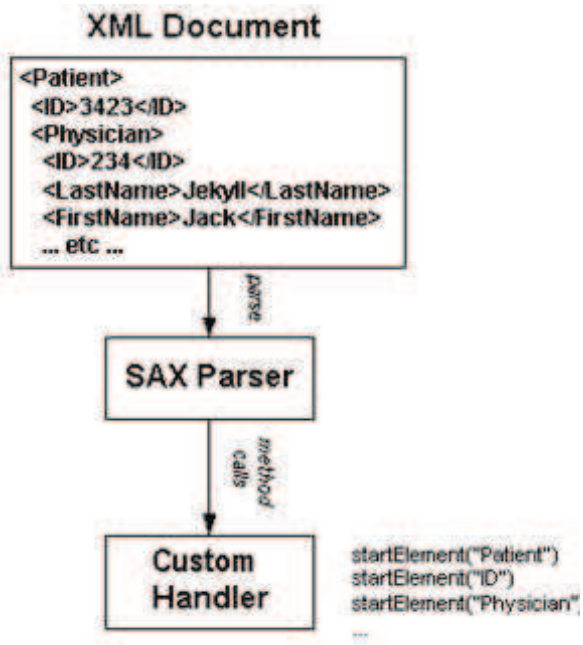

Ilustración 21.- Proceso al parsear con SAX.

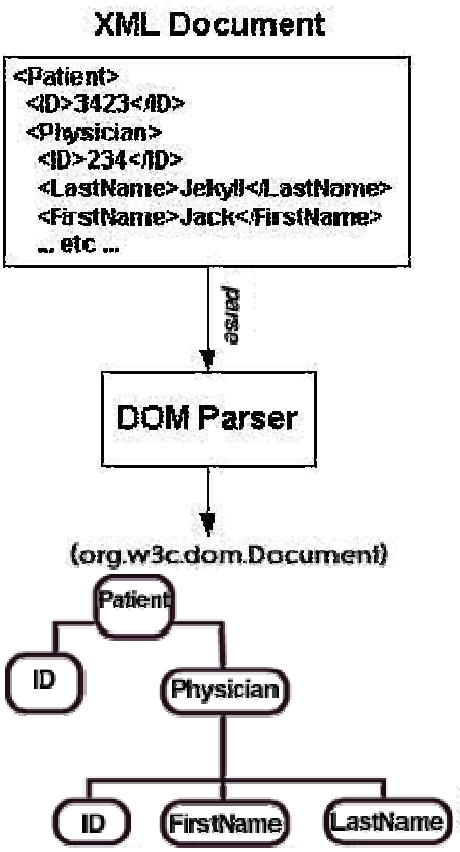

Ilustración 22.- Proceso al parsear con DOM.

Las Tabla 38 se ilustran las diferencias entre las APIs de SAX y DOM:

SAX	DOM
Modelo basado en eventos	Estructura de datos tipo árbol
Acceso serie (flujo de eventos)	Acceso aleatorio (estructura de datos en memoria)
Bajo uso de memoria (sólo se generan eventos)	Alto uso de memoria (todo el documento se carga en memoria)
Para procesar partes del documento (capturar eventos importantes)	Para editar el documento (procesar la estructura de datos en memoria)
Para procesar el documento sólo una vez (flujo de eventos temporal).	Para procesar el documento múltiples veces (documento cargado en memoria).

Tabla 38.- Diferencias entre SAX y DOM.

Y por último, la siguiente tabla muestra cuándo es más útil utilizar cada uno de los analizadores:

SAX	DOM
Cuando no haya una modificación estructural del documento.	Para modificar el documento.
Menor gasto de memoria y mayor rapidez.	Si se necesita realizar múltiples procesados.
Si sólo se necesitan partes de documentos	Evita tener que volver a analizar el documento
Para documentos XML grandes, en donde sólo haya que procesar una pequeña parte de la información.	Para documentos XML pequeños que necesiten ser procesados en su práctica totalidad.
Permite recorrer secuencialmente un documento XML y responder a una serie de eventos.	Evita tener que construir tu propio árbol.

Tabla 39.- Casos en los que usar SAX o DOM.

2.5.5. Conclusión.

Los analizadores son herramientas para el manejo de documentos XML, desde la lectura o modificación del documento hasta la validación del mismo.

A lo largo de los distintos apartados se han ido mostrando las características y funcionamiento de cada uno de los analizadores. En ellos se aclara que SAX es el mejor para el servicio que se va a implementar: leer y validar los fichero XML de configuración y de GPC.

Esta elección ha sido la seleccionada principalmente porque se trata de documentos que en esos casos sólo van a ser consultado una vez, luego el coste en memoria y tiempo va a ser positivo.