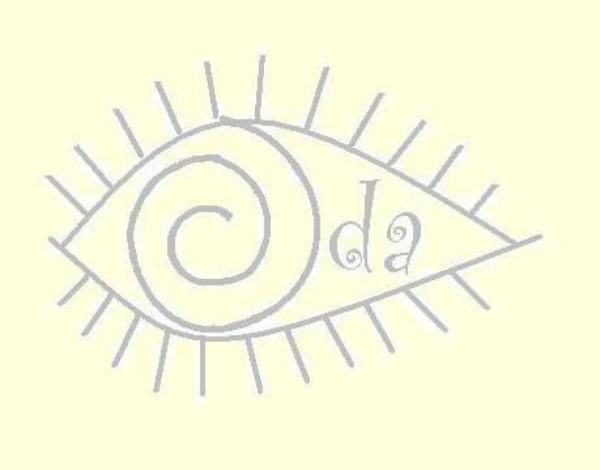
PROYECTO DE GRADO

ODA

Asistente Informático para la Generación de Objetos de Aprendizaje



Tutora: Regina Motz

Andrea Rodríguez
Raquel Sosa

Informe de Proyecto de Grado presentado al Tribunal Evaluador como requisito de graduación de la carrera Ingeniería en Computación. Montevideo, 2006

Resumen

Este proyecto presenta un Asistente Informático que ayuda a los docentes a preparar materiales ya existentes para ser usados en Plataformas de Educación a Distancia. El objetivo principal es obtener en forma semi-automática Objetos de Aprendizaje (ODA) desde materiales publicados en los sitios web de los cursos. Además, los ODA generados son enriquecidos con metadatos siguiendo el estándar LOM. Los ODAs son empaquetados de acuerdo al estándar SCORM. Disponer de Objetos de Aprendizaje ricos en metadatos, permite que estos objetos sean consultados en repositorios y que puedan ser adaptados de acuerdo al público que los consulta.
El Asistente Informático ODA está desarrollado en Java utilizando herramientas de procesamiento de lenguaje natural como Balie que utiliza QTag como Part-of-Speech Tagger y hace un uso intensivo de XML.

Palabras clave: Objetos de Aprendizaje, SCORM, LOM, Extracción de Metadatos.

Índice

Resumen 2
1. Introducción 5
1.1. Objetivos del Trabajo 7
1.2. Resultados 7
1.3. Metodología de trabajo 8
1.4. Organización del informe 8
2. Contexto de trabajo 10
2.1. Objetos de Aprendizaje 11
2.1.1. SCORM 13
2.2. Web Semántica 15
2.2.1. Ontologías 15
2.2.2. XML 17
2.2.3. XPath 17
2.3. Extracción automática de información 17
2.3.1. Wrappers y Mediadores 17
2.3.2. Procesamiento de Lenguaje Natural 19
2.3.3. Herramientas de Part-of-speech Taggers 20
2.3.4. Herramientas de aplicación de reglas contextuales 21
3. Análisis y diseño de la solución 23
3.1. Requerimientos 23
3.2. Ejemplo 24
3.3. Arquitectura de la solución 26
3.3.1. Marcado y organización de ODAs 26
3.3.2. Inferencia de metadatos 28
3.3.3. Diagrama de Componentes 28
3.3.4. Diseño de Componentes 30
4. Implementación y Testeo 37
4.1. Herramientas usadas para la implementación 37
4.2. Testeo Realizado 38
4.3. Métricas 44
5. Conclusiones y Trabajos Futuros 46
Asistente Pedagógico 46
Referencias 50

Índice de Figuras

Figura 1 - Problema a resolver....6
Figura 2 - Esquema del Proyecto Educa....10
Figura 3 - Características de Objetos de Aprendizaje....12
Figura 4 – SCORM 2004 (Bookshelf)....13
Figura 5 - Ontología de contenido de curso....16
Figura 6 - Representación en UML del pattern Wrapper....18
Figura 7 - Arquitectura del Web Warehouse....18
Figura 8 - Prueba del Wrapper 2004 con la Ontología de contenido de cursos....19
Figura 9 - Página de ejemplo....24
Figura 10 - Archivo con índice de temas....24
Figura 11 - ODAs de la página de ejemplo de la Figura 9....25
Figura 12 - Arquitectura de la solución....26
Figura 14 - Modulo de Marcado Automático - Diagrama de clases....29
Figura 15 - Ejemplo archivo de salida de marcado de temas....30
Figura 16 - Ejemplo de RGL para marcar Definiciones....31
Figura 17 - Ejemplo de salida del marcado POS....32
Figura 18 - Texto correspondiente a una definición....32
Figura 19 - Texto correspondiente a una definición....33
Figura 20 - Texto correspondiente al comienzo de un Teorema....33
Figura 21 - Prácticos 0 y 2 de Programación 1....40
Figura 22 - Esquema del resultado del marcado en Componentes Básicos....41
Figura 23 - Resultados de Reconocimiento de Definiciones....42
Figura 24 - Resultado de Reconocimiento de Ejercicios....42
Figura 25 - Resultado del reconocimiento de ejemplos....42
Figura 26 - Página correspondiente al Práctico 2....44
Figura 27 - Esquema de la aplicación del marcado al Practico 2....44
Figura 28 - Mapa Conceptual de una Unidad Didáctica....47
Figura 29 - Mapa conceptual de las actividades....49

Índice de Tablas

Tabla 1 – Esquema de metadatos culturales....14
Tabla 2 - Relevamiento de Herramientas de marcado POS....21
Tabla 3 - Tipo de Captura de Metadatos de LOM....35
Tabla 4 – Resultado del testeo del módulo de aplicación de reglas....39

Anexos

I – Documento de Análisis y Diseño
II – Manual de Usuario

1. Introducción

En la actualidad muchos docentes publican en sitios web sus contenidos como complemento a los cursos presenciales. Al querer llevar estos cursos presenciales a cursos a distancia (o semi-presenciales) vuelven a generar todo el material desde cero. Ocasionalmente pueden encontrar asistencia en la plataforma de educación a distancia que usan, para ingresar sus materiales, como se puede hacer en ATutor [4] al disponer de un editor de páginas HTML o un generador de paquete SCORM[13] como RELOAD[5].

Dentro de la realidad que conocemos –principalmente el caso del InCo – identificamos el proceso de generación de materiales, como el cuello de botella para la adopción de las plataformas de educación a distancia. Una forma de agilizar este proceso es extraer de forma automática los Objetos de Aprendizaje (ODA) desde los materiales disponibles.

Las distintas plataformas de educación a distancia han evolucionado a un formato estándar de intercambio de contenidos, este formato es SCORM. Este estándar permite que: 1) los diversos documentos electrónicos (páginas html, imágenes, videos, etc.) puedan ser entendidos por las distintas plataformas de e-learning; 2) el material quede clasificado; 3) se pueda reutilizar un curso o parte del mismo; 4) sea más fácil transformar el material a otro medio distinto y propio de otra plataforma LCMS (Learning Content Management System).

Sobre estos objetos de aprendizaje nos interesa poder tener algo más de información además de su propio contenido, este tipo de información es lo que llamamos Metadatos y que permite a un docente conocer las propiedades de un contenido educativo sin tener que leer todo el material. El estándar de metadatos de objetos de aprendizaje sugerido por SCORM es LOM. A su vez, estos metadatos permiten que los objetos de aprendizaje puedan ser "manipulados" automáticamente por sistemas de software.

Tradicionalmente, la característica de poder manipular el material automáticamente, está asociado al concepto de *metadato*, que significa "dato sobre los datos" o "información sobre la información". En nuestro contexto una definición más precisa sería "información relevante de los objetos de aprendizaje, procesable automáticamente". Según la ontología de contenido de curso [6], un ODA puede ser, por ejemplo, un curso, un tema, un ejercicio, una definición o incluso un ejemplo.

El problema a resolver consiste en reconocer y extraer ODAs de páginas HTML correspondientes a cursos. La Figura 1 muestra un esquema del procesamiento a realizar: los materiales elaborados para un curso presencial se toman como entrada del sistema ODA-Web (el asistente recibe como entrada la ubicación del curso presencial). El sistema reconoce y marca los ODAs presentes en las páginas HTML. A la vez extrae también los metadatos que tengan dichas páginas. El sistema permite que el usuario edite el marcado automático. Luego, el paquete obtenido del sistema ODA-Web puede ser colocado en un repositorio de paquetes SCORM o ser levantado en una plataforma de educación a distancia como ATutor o Moodle[37].

Una vez finalizado este proceso, el usuario revisa el marcado y puede modificarlo. También se le permite al usuario editar los metadatos para agregar información faltante. Por ejemplo, si el sistema no pudo obtener de forma automática el nombre del autor del curso de las páginas HTML, el usuario deberá ingresarlo antes de generar el paquete SCORM.

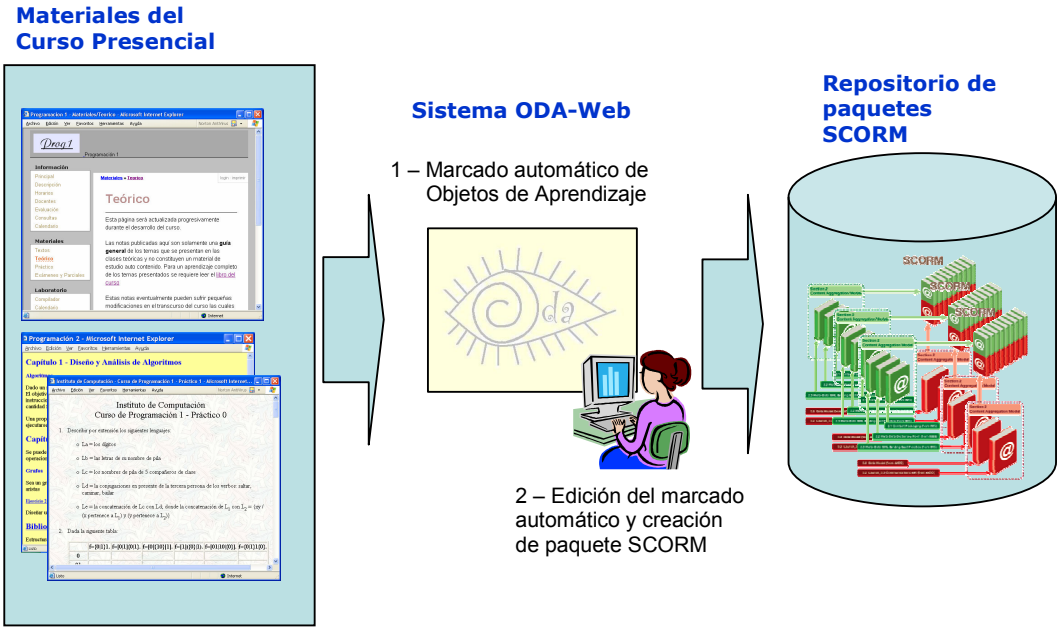


Figura 1 - Problema a resolver

El asistente ODA-Web marca temas y dentro de los mismos, las definiciones, ejemplos, ejercicios y otros objetos que especificaremos más adelante.

Para extraer estos ODAs, deberíamos entender para cada oración y/o párrafo, en que contexto estamos y cual es la función de ese bloque de texto. El lenguaje natural es de por si, ambiguo a diferentes niveles: i) a nivel léxico una misma palabra puede tener varios significados y el significado correcto se deduce a través del contexto gramatical; ii) la resolución de anáforas y catáforas implica determinar la entidad lingüística previa o posterior a que hacen referencia; iii) a nivel estructural, se requiere de la semántica para resolver las ambigüedades de dependencia de los sintagmas preposicionales; iv) a nivel pragmático, una oración, a menudo, no significa lo que realmente se está diciendo. Elementos tales como la ironía tienen un papel importante en la interpretación del mensaje.

Para la detección y marcado de ODAs, tomaremos los enfoques i) y iii) El reconocimiento de ODAs se realiza a través de la búsqueda de palabras claves en un contexto gramatical y/o de la identificación de ciertas estructuras gramaticales. Algunas de las palabras claves a buscar son las provenientes de la ontología de curso y del mapa conceptual de la Figura 3.

1.1. Objetivos del Trabajo

El objetivo general de este trabajo es:

Desarrollar un sistema para la transformación semi-automática de material en formato HTML hacia objetos de aprendizaje para ser usados en Educación a Distancia.

Para esto marcamos los siguientes objetivos particulares:

- Investigar el estado actual de la extracción automática de información para la reutilización de estas tecnologías en la generación automática de Objetos de Aprendizaje.

- Procesar automáticamente las páginas HTML para reconocer elementos de curso y delimitar los Objetos de Aprendizaje (ODAs) y extraer metadatos.

- Brindar una solución sencilla de empaquetamiento de los Objetos de Aprendizaje en paquetes SCORM que puedan ser ingresados directamente en una plataforma de Educación a Distancia.

1.2. Resultados

Se realiza un relevamiento de Wrappers y herramientas de Procesamiento de Lenguaje Natural para la investigación del estado actual de la extracción automática de información y su aplicación a la extracción de Objetos de Aprendizaje.

Se obtiene un asistente para la transformación semi-automática de material educativo en formato HTML, a Objetos de Aprendizaje. La herramienta también extrae de forma automática metadatos de LOM, para el curso procesado, y estadísticas de las cantidades de definiciones, enunciados, ejercicios, ejemplos y otros. Este asistente requiere la participación del docente en diversos momentos para validar/corregir las acciones realizadas por el sistema.

Se deja el material del curso de Programación 1 en una plataforma que importa SCORM

Las estadísticas que se obtienen junto con el reconocimiento de los ODAs, aportan al proyecto CSE: “ODA: Asistente Informático y Pedagógico para generar Objetos de Aprendizaje”, ya que permiten comparar la cantidad de material expositivo con la cantidad de material práctico existente en un curso.

Se realizan aportes en el análisis de requerimientos para la definición de un asistente pedagógico.

En Agosto y Setiembre, se presentan además dos artículos que muestran parte del trabajo, para las conferencias de LAWEB06 [2] en México y otro para LACLO [3] en Ecuador. En octubre se realizan las respectivas presentaciones en las conferencias.

1.3. Metodología de trabajo

El trabajo se realiza siguiendo una metodología de trabajo iterativa e incremental que consta de tres iteraciones principales. En la primera iteración se estudian trabajos previos de extracción de información. Se busca y prueba exhaustivamente varias herramientas de extracción y marcado de información y herramientas de procesamiento de lenguaje natural. En la segunda iteración se obtiene un prototipo que extrae objetos de aprendizaje de páginas HTML y permite al usuario modificar el marcado automático. En la tercera iteración se generan paquetes SCORM con la información extraída de las páginas web, cumpliendo así con los objetivos planteados.

Se utilizó BSCW como espacio de trabajo para compartir versiones intermedias de los documentos generados y prototipo. Los documentos finales y la versión final de la plataforma se encuentran en la página oficial del proyecto [1]

Se mantuvieron reuniones periódicamente con la tutora en principio con el fin de comprender con claridad el problema a resolver, y luego mostrar el avance del proyecto y discutir las decisiones importantes. El equipo de trabajo mantuvo reuniones semanales, para la discusión de temas y la toma de decisiones, así como la coordinación de las actividades y la evaluación de los resultados de cada semana.

Se realizaron además tres reuniones con las Licenciada en Educación Nancy Peré y la Licenciada en Comunicación Virginia Rodes que ayudaron a definir una posible extensión del sistema ODA-Web y entender conceptos pedagógicos, aplicables al proyecto. El mapa conceptual presentado en el Capítulo 2 muestra parte del resultado de estas reuniones aplicable al proyecto. En el capítulo 5 se presenta como trabajo futuro esta extensión.

1.4. Organización del informe

El presente informe está organizado en 5 capítulos que describen los aspectos más relevantes del trabajo realizado durante el desarrollo de este proyecto de grado. Se anexa además material complementario para que el lector pueda profundizar en los temas. Los capítulos están organizados de la siguiente forma:

Capítulo 1: este capítulo es una introducción, presenta objetivos, forma de trabajo y la organización del informe.

Capítulo 2: describe el área en la que se enmarca este trabajo. Realiza una recorrida por los temas relacionados, definiciones previas y trabajos relacionados.

Capítulo 3: explica brevemente los requerimientos y profundiza en las decisiones de diseño y la arquitectura del asistente ODA-Web.

Capítulo 4: detalla la implementación del asistente, herramientas y bibliotecas utilizadas, pruebas realizadas, resultados prácticos obtenidos y mediciones.

Capítulo 5: presenta las conclusiones del proyecto, producto obtenido y trabajos futuros.

Finalmente se presentan las referencias consultadas para la realización del proyecto.

2. Contexto de trabajo

A continuación presentamos a grandes rasgos los temas de investigación, herramientas y estándares que fueron base para este proyecto.

Este trabajo está enmarcado dentro del proyecto Red Educa [9], cuyo objetivo es desarrollar un sistema para facilitar la búsqueda de cursos ofrecidos en modalidad a distancia que han sido creados por distintas universidades utilizando diferentes herramientas. La Figura 2 muestra un esquema de lo que abarca el proyecto Educa.

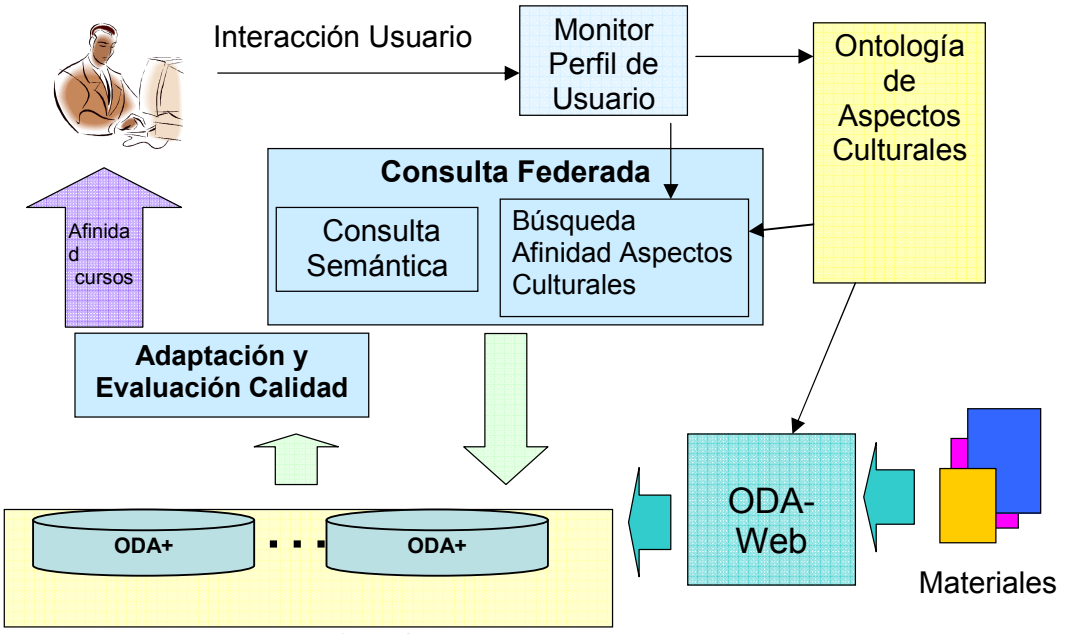


Figura 2 - Esquema del Proyecto Educa

El principal objetivo de EduCa es ofrecer un sistema federado de búsqueda de cursos, que maneje características culturales. Esta Federación estará formada por distintas universidades que ofrecen cursos a sus estudiantes. Los estudiantes tienen un perfil en el sistema que considera sus características culturales (región, idioma, forma de aprendizaje, etc). Ante una búsqueda de un estudiante sobre determinado tema, el sistema buscará objetos de ese tema que además cumplan con las características culturales del estudiante. Al devolver el resultado de la búsqueda el sistema avisará al estudiante cuánto se ajustan los ODAs encontrados a su solicitud y a sus características; además tratará de adaptar el ODA a las características del estudiante. Para que este sistema pueda funcionar se necesita una forma sencilla de generar ODAs ricos en metadatos y con sus características culturales, para esto se crea el asistente ODA-Web.

Un trabajo similar es el desarrollado por la Fundación ARIADNE [10], que cuenta con una red de repositorios de materiales educativos, aunque no posee características culturales.

2.1. Objetos de Aprendizaje

En este proyecto es central el concepto de Objeto de Aprendizaje, por esto presentamos a continuación las definiciones más usuales. Un Objeto de Aprendizaje es:

- "cualquier objeto, digital o no, que puede ser utilizado, reutilizado, o referenciado en el proceso de enseñanza apoyada por la tecnología." [38]

- "cualquier recurso digital que puede usarse repetidamente para facilitar el aprendizaje." [39]

- "un recurso digital basado en un sólo objetivo de aprendizaje que comprende presentación, actividad, evaluación y otros recursos complementarios. Pueden ser adaptados, editados o recombinados de acuerdo con los propósitos específicos de enseñanza aprendizaje" [40]

Las definiciones anteriores coinciden en que los Objetos de Aprendizaje deben ser reutilizables. Esta propiedad es importante porque se generan contenidos educativos sobre los mismos temas en distintas organizaciones, replicando el esfuerzo que esta generación lleva. También se dice que un objeto es reutilizable si no es sensible al contexto y no referencia a otros objetos de aprendizaje. El no ser sensible al contexto, se refiere a que no debe ser parte de una situación válida para un hecho circunstancial, por ejemplo, una noticia. Por otro lado, un objeto no debe referenciar a otros objetos, ya que se espera que cada objeto pueda ser usado como una unidad independiente.[35]

El concepto de Objeto de Aprendizaje no especifica una granularidad, y esto es razonable ya que podría llegar a ser tanto un ejemplo que ocupe un párrafo como un módulo completo. La granularidad es dejada a criterio del autor. En nuestro caso, los tipos de objetos de aprendizaje buscados son los especificados en la ontología de curso, como se verá en la sección 2.2.1.

De las reuniones con Virginia Rodes y Nancy Peré surgieron varios mapas conceptuales. La Figura 3 es un mapa conceptual de un Objeto de Aprendizaje, donde se identifican en distintos niveles Cursos, Unidades Didácticas y Sesiones de Trabajo.

Los elementos que contienen las Unidades Didácticas son la metodología, los objetivos, la evaluación, los contenidos, medios de comunicación y los materiales utilizados. Una unidad didáctica se concreta en sesiones de trabajo. Finalmente un curso, está conformado por Unidades Didácticas y dentro del mismo se identifican además Contrato Didáctico y el Programa del curso.

El sistema ODA-Web solo identifica los contenidos de las sesiones de trabajo de un curso. Por un asunto de practicidad, de aquí en más nos referiremos a ellos simplemente como contenidos de curso.

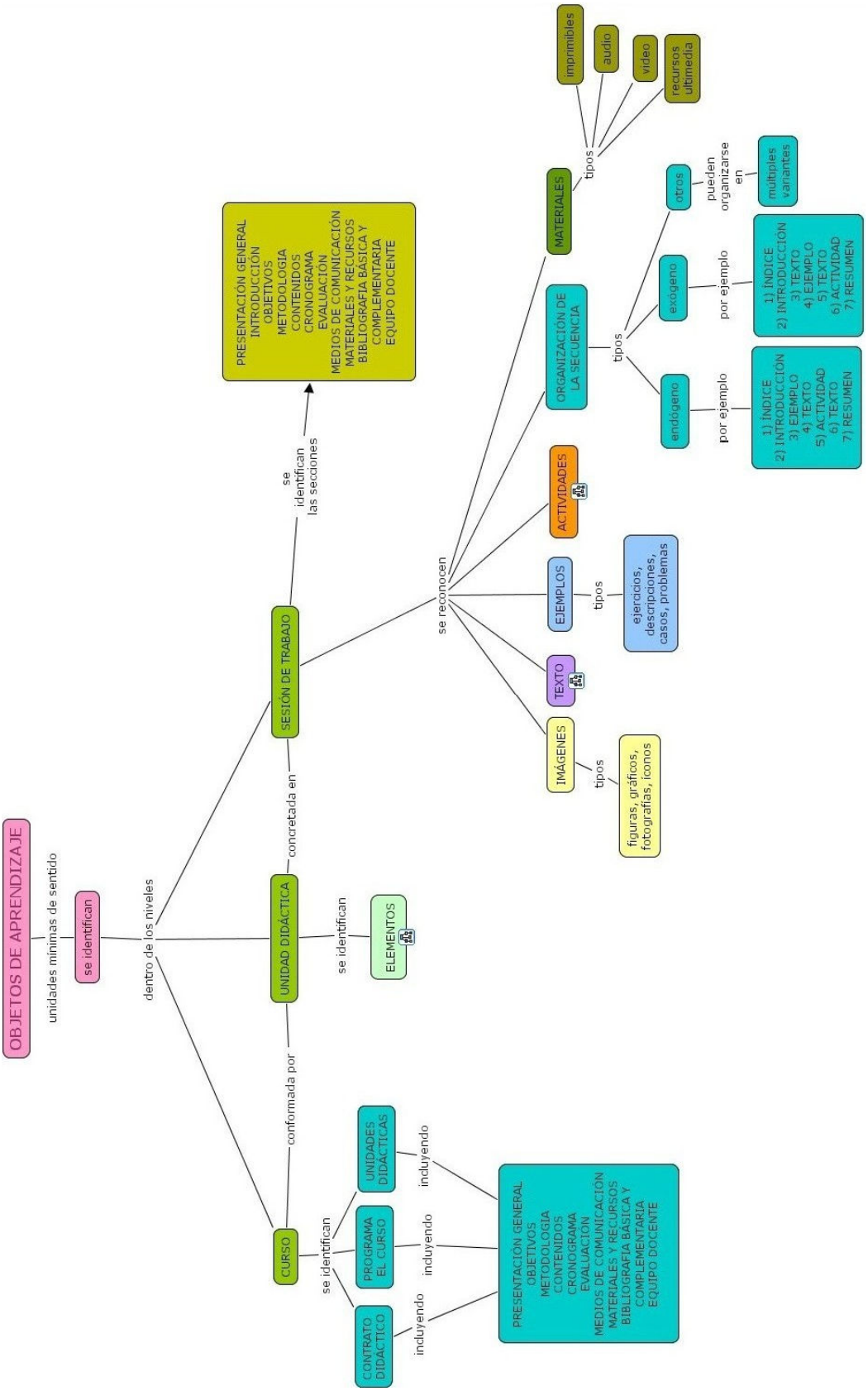


Figura 3 - Características de Objetos de Aprendizaje

Para que los Objetos de Aprendizaje puedan ser generados, compartidos, usados y re utilizados tienen que ser representados en formatos estándar. Veremos estos estándares a continuación.

2.1.1. SCORM

SCORM (Shareable Content Object Referente Model) [13] es un estándar que permite al usuario creador de un curso, generar material para múltiples LCMS. Los Objetos de Aprendizaje en formato SCORM pueden ser portados entre distintas plataformas que cumplan con el estándar.

El empaquetamiento SCORM es un estándar de intercambio de materiales de Objetos de Aprendizaje generado por ADL (Advanced Distribute Learning) [14] para permitir la interoperabilidad, accesibilidad y reutilización de contenidos educativos en un contexto de Educación a Distancia. Permite además que el material electrónico pueda ser entendido por distintas plataformas (LCMS) e incluso transformado en un medio electrónico distinto y específico de la herramienta LCMS. Sin embargo para que un LCMS pueda manejar paquetes SCORM debe al menos implementar el Run-Time Environment especificado por SCORM.

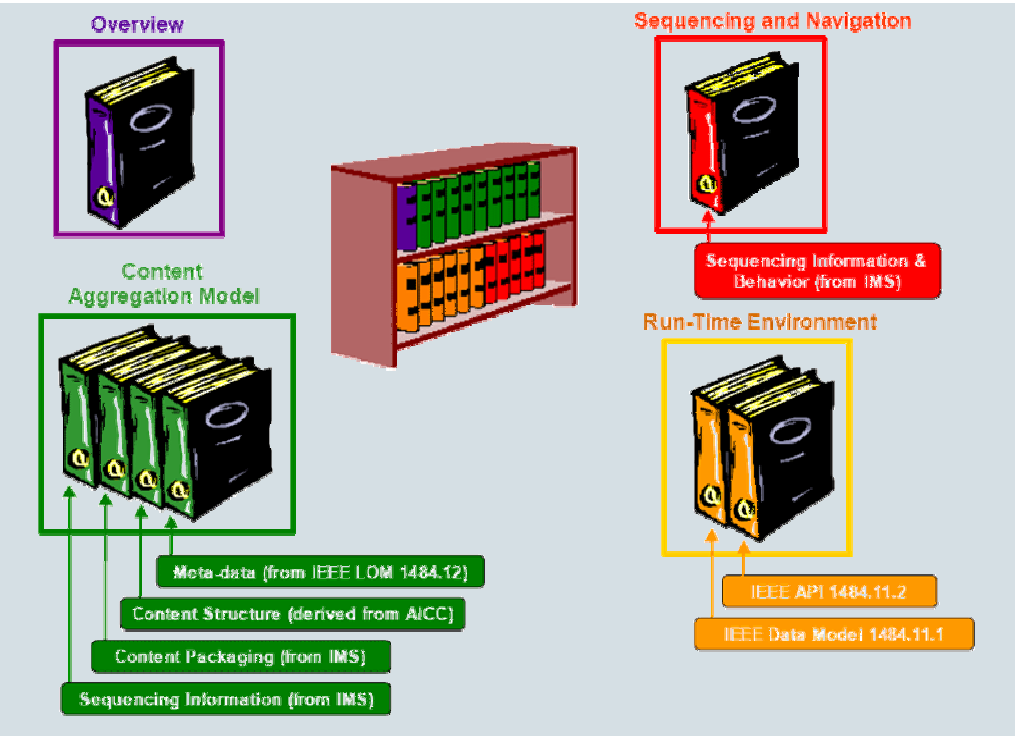


Figura 4 – SCORM 2004 (Bookshelf)

La Figura 4 muestra un esquema de los distintos componentes del estándar SCORM. En particular nos interesa el Content Aggregation Model (CAM) que especifica una forma estándar de referenciar y describir los materiales educativos que contiene el paquete. Además toma en cuenta el grado de complejidad de los objetos de aprendizaje. Se

pueden definir tanto elementos básicos (Asset) como puede ser un texto, una imagen, una página web o elementos que son conjuntos de elementos básicos (SCO – Sharable Content Object).

LOM [7] (Learning Object Metadata) es el estándar de metadatos sugerido por SCORM. Según el estudio de Rouyet y Martín [19] LOM contiene más metadatos que otros estándares conocidos, como Dublin Core que tiene solo 15 metadatos. SCORM sugiere seguir una versión reducida de LOM.

Los elementos de metadatos LOM se organizan en nueve categorías: General, Ciclo de vida, Meta-metadatos, Aspectos Técnicos, Aspectos Educativos, Derechos, Relación con otros recursos, Anotación, Clasificación.

Los perfiles de aplicación de LOM son extensiones o simplificaciones de este estándar, creados según ciertas necesidades, por ejemplo CanCore (http://www.cancore.ca/documents.html) pretende estar entre un punto intermedio entre la forma estructurada de enfocar los metadatos de LOM y el punto de vista minimalista de Dublin Core; UK Learning Object Metadata Core (UK LOM) tiene como objetivo ser el estándar básico de metadatos para recursos de aprendizaje, en el Reino Unido, para asegurar la interoperabilidad entre sistemas. Por esta razón establece elementos básicos del estándar LOM (obligatorios, opcionales y recomendados) y da guías sobre cada uno de los elementos, e indica los vocabularios controlados y los estándares recomendados para establecer los valores de estos elementos.

En el proyecto Educa se propone una extensión de LOM que agrega metadatos culturales. Los aspectos culturales son preferencias y formas de comportamiento de cada cultura, frente a actividades de aprendizaje.
La Tabla 1 muestra los campos correspondientes a los metadatos culturales.

Tabla 1 – Esquema de metadatos culturales

1	Grado de Impaciencia
2	Grado de Positividad
3	Actitud (Activo, Pasivo, Reactivo)
4	Lenguaje a. Expresiones Idiomáticas b. Uso común de tiempos verbales
5	País
6	Estilos de aprendizaje a. Tipo de estilos (verbal holístico, visual holístico, verbal analítico, visual analítico) b. Grado de generalidad (alta, media, baja) c. Uso de un caso real o no.
7	Actividad a. Tipo de actividad (crear soluciones, seleccionar soluciones, leer, escuchar, ver) b. Tipo de participación (individual, grupal, colectiva) c. Herramientas (foros, chat, simulación) d. Metas (evaluación, practica) e. Uso de recursos (textos, diagramas, ejercicios)

2.2. Web Semántica

Actualmente la información en la Web no sólo es accedida por usuarios, sino que también es accedida por sistemas de software. Para esto se requiere de interoperabilidad a nivel semántico, lo que significa que los sistemas de software no solo deben manejar la información que hay disponible, sino también su significado. Para lograr esta interoperabilidad se necesitan estándares para representar la información y para representar la semántica de ésta. Algunos estándares recomendados por la W3C son: HTML, XML/XML Schema, RDF/RDF Schema, OWL, XPath. El sistema ODA-Web propuesto agrega información semántica a los contenidos de las páginas web de cursos. A continuación mencionamos los distintos estándares utilizados en este proyecto.

HTML, el lenguaje utilizado en la web para publicar información, tiene como característica que permite que la información quede semi-estructurada, al disponer de etiquetas que indican cómo se presentará cada parte de una página web. Por ejemplo, es posible reconocer títulos y de esta forma comienzo y fin de temas, como se explicará en la sección 0.

Muchas de las páginas publicadas no necesariamente están bien formadas. Debido a esto antes de procesar páginas web se necesita arreglar su codificación en HTML, con este fin usamos Tidy [27], una herramienta que permite generar HTML válido (y XML) a partir de código HTML común.

2.2.1. Ontologías

Una ontología es un modelo de datos que representa un dominio y que usamos para razonar sobre él, es una forma de representa conocimiento.
Según Gruber [26][28] es la "especificación explícita de una conceptualización, es una representación formal y declarativa que incluye vocabulario para referirse a términos en el área de interés y sentencias lógicas que describen qué son los términos, cómo están relacionados, y cómo pueden o no estar relacionados". También agrega [28]: "es una descripción de conceptos y relaciones que existen para un agente o una comunidad de agentes". Esto nos lleva a que se define un vocabulario compartido, para darle un sentido exacto a cada término dentro de nuestra área.

Existen varios lenguajes para especificar ontologías como DAML+OIL (http://www.daml.org) y OWL (http://www.w3.org/2004/OWL/). En este proyecto se usó la ontología de Actividades como marco conceptual para entender qué conceptos se hallan en un curso y como se compone un material educativo.

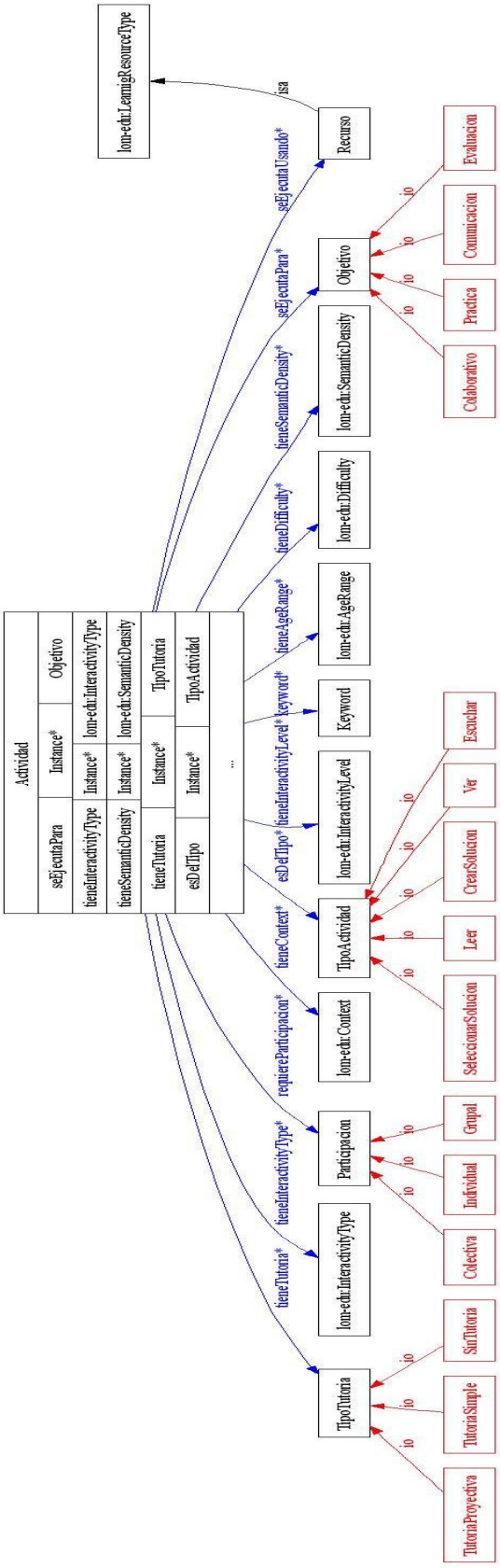


Figura 5 - Ontología de contenido de curso

Los elementos del curso son los presentados en la ontología de contenido de curso (http://www.fing.edu.uy/inco/grupos/csi/esp/Proyectos/OntologiasOwl/Actividad.owl) La Figura 5 es una representación gráfica de la misma.

2.2.2. XML

XML significa eXtensible Markup Language. Es un lenguaje de etiquetas diseñado para describir datos. No tiene etiquetas predefinidas, sino que cada uno debe definir sus propias etiquetas de acuerdo a los datos que se quieran representar.

En XML se usa una Definición de Tipo de Documento (DTD) o un esquema XML (XML-Schema) para describir la estructura de la información a representar. XML se auto-describe con un DTD o un esquema XML.

ODA-Web utiliza XML como formato para uno de los archivo de entrada y para guardar los marcados intermedios que realiza internamente. Además este lenguaje se utiliza para escribir el IMS-Manifest necesario para el paquete SCORM.

En [24] se encuentra un Tutorial de XML.

2.2.3. XPath

XPath [25] es un lenguaje para encontrar información y navegar a través de elementos y atributos en los documentos XML. Permite definir reglas declarativas para luego realizar las búsquedas. Una de sus características más importantes es que permite hacer consultas tanto sobre los datos guardados en un XML, como sobre su estructura (a través de sus etiquetas, atributos y jerarquía).

El sistema ODA-Web guarda sus procesamientos intermedios en XML y aplica reglas gramaticales en XPath, para las búsquedas contextuales, como explicaremos más adelante en la sección 3.3.4.

2.3. Extracción automática de información

Vamos a separar en dos grupos las herramientas de extracción automática de información que se evaluaron para este proyecto: a) Wrappers y mediadores; b) Herramientas de procesamiento de lenguaje natural.

2.3.1. Wrappers y Mediadores

Un Wrapper es un componente particular definido en el Patrón de diseño Wrapper o Adaptador. El Wrapper enmascara el acceso a un servicio ofreciendo una nueva interfaz de acceso y realizando cierto procesamiento. Como se ve en la Figura 6, el adaptador o Wrapper oculta la interfaz de ‘Adaptee’.

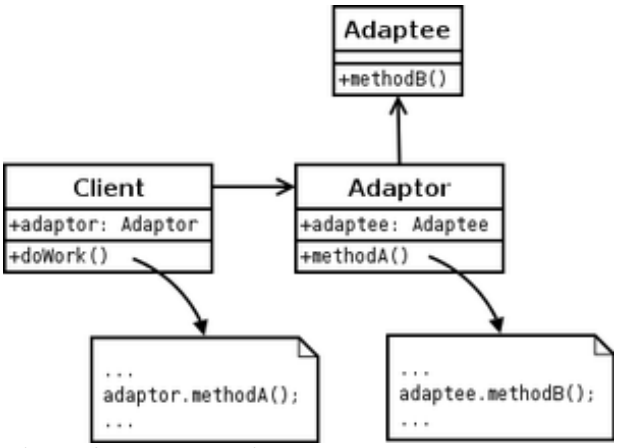


Figura 6 - Representación en UML del pattern Wrapper

Los Wrappers analizados son un esquema propuesto para obtener información para un Web Data Warehouse.

En esta arquitectura [29] el objetivo de un Wrapper es extraer información relevante a partir de una fuente de datos y presentar la información en un formato especificado. Luego, un mediador integra la información producida por diferentes wrappers u otros mediadores. La Figura 7 describe esta arquitectura.

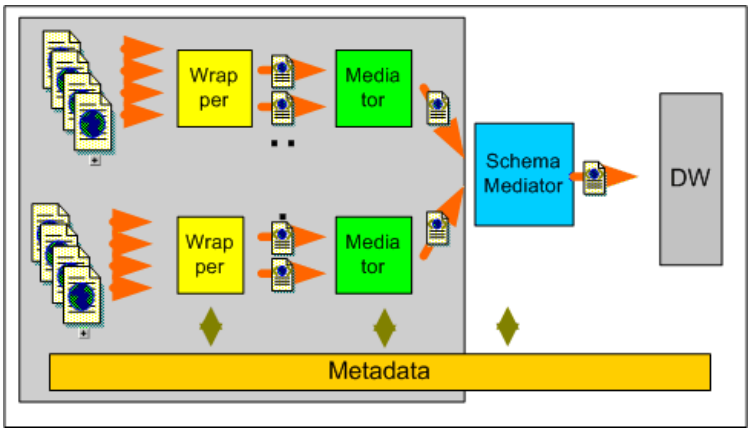


Figura 7 - Arquitectura del Web Warehouse

El principio del proyecto consistió en estudiar los Wrappers implementados en los proyectos de grado 2004 [16] y 2005 [17], que para la extracción de información aprovechan la estructura de las páginas Web. Se probó con las páginas del curso presencial de programación 1 y se le pasó por parámetro una ontología de curso.

La Figura 8 muestra el resultado del Wrapper generado con el proyecto 2004. Extrae las palabras del dominio buscado (definidas en la Ontología). La figura corresponde al archivo unido.xml. Como se ve en la figura, el wrapper encuentra un ejemplo y extrae parte de la información. En este caso sirve para indicar la presencia de un “Ejemplo de programa en Pascal” en el tema Componentes Básicos y de un “Ejemplo con recursión” en el tema Entrada y Salida.

En ambas herramientas el objetivo es extraer información específica, estructurada, desechando el resto del contenido de las páginas webs; por esto el enfoque no es apropiado para el reconocimiento de ODAs.



Figura 8 - Prueba del Wrapper 2004 con la Ontología de contenido de cursos

Como resultado del estudio realizado se destaca como aporte importante el enfoque de aprovechar la estructura de HTML en el reconocimiento de distintas secciones en las páginas web.

2.3.2. Procesamiento de Lenguaje Natural

El Procesamiento del Lenguaje Natural (PLN) es una de las piedras angulares tempranas de la inteligencia artificial. Los objetivos del PLN son de tres tipos: interfaces en lenguaje natural, procesamiento de textos, traducción automática.

Lo que utiliza el sistema ODA-Web son algunas herramientas de procesamiento de textos.

Como el lenguaje natural es de por sí, ambiguo. El PLN se convierte en una tarea compleja. Por ejemplo la palabra *ser* puede utilizarse tanto como verbo como sustantivo. La forma de saber el significado real de esta palabra es a través del "contexto oracional" y ciertas reglas gramaticales que definen que en ese contexto la palabra es un verbo o un sustantivo.

Es necesario, por lo tanto una herramienta que detecte el contexto aplicando reglas gramaticales y otra herramienta capaz de buscar ciertas palabras clave dentro de este contexto.

Aparte de la ambigüedad léxica, hay otros tipos de ambigüedades lingüísticas que resolver (referenciales y estructurales) pero no nos ocupamos de estas dentro de este proyecto.

2.3.3. Herramientas de Part-of-speech Taggers

Part-of-speech (POS) Tagging o Marcado de Parte-del-Discurso, es un proceso de marcado de palabras basado en la función que cumple en un texto particular. Se basa tanto en las definiciones como en el contexto, es decir la relación de una palabra con las palabras adyacentes relacionadas. Una forma simplificada de esto es lo que hacen los niños de escuela al marcar dentro de un texto sustantivos, verbos, adjetivos, adverbios, etc.

Dentro de las herramientas de PLN, esta es una de las más básicas, pero a su vez es la que permite procesamientos subsiguientes, al poder agregarle información a los textos que queremos procesar.

Para este proyecto se busca una herramienta o biblioteca de código abierto, desarrollada en java para que sea más fácil su integración, y lo más importante es que realice marcado POS para idioma español.

En la Tabla 2 se ve un estudio comparativo de algunas de las herramientas disponibles para cumplir con esta función.

Nombre	Idiomas	Implementado en	Tipo de Licencia	Evaluación
Stanford POS tagger	Ingles	Java	Open Source	Es una suite completa de procesamiento de lenguaje natural y contiene un módulo de Part-Of-Speech Tagger.
TreeTagger	Inglés, Alemán y Francés		Open Source	Basado en árboles de decisión.
Maximum Entropy part of speech tagger		Java	Open Source pero están disponible solo los .class	Es una biblioteca de procesamiento de lenguaje natural pero la última actulización en Source Forge es del 2004.
ACOPOST (formerly ICOPOST)		C	Open Source	
YamCha		C/C++	Open Source	
Balie-QTAG Part of speech tagger	Ingles, Español, Rumano, Frances, Alemán	Java	Balie es Open Source, Qtag está disponible para bajar pero no el código fuente, solo los class	Es una suite de procesamiento de lenguaje natural y contiene un módulo de Part-Of-Speech Tagger (Qtag) y está disponible para español
Brill's Transformation-based learning Tagger	Ingles	C		
Original Xerox Tagger		Lisp	Usa Xerox	
Lingua-EN-Tagger	Ingles	Perl		

Tabla 2 - Relevamiento de Herramientas de marcado POS

Se eligió utilizar Balie que utiliza QTag [22] para el marcado POS, está implementado en Java y está disponible para español. La Tabla 2 muestra una lista de las herramientas de marcado POS analizadas.

Para la extracción de información, el sistema ODA-Web aprovecha la estructura de las páginas web y utiliza técnicas de procesamiento de lenguaje natural. Los mecanismos utilizados se verán en el capítulo 3.

2.3.4. Herramientas de aplicación de reglas contextuales

Una técnica de procesamiento de lenguaje natural en la que no sólo se reconoce lo que buscamos porque lo tengamos en un diccionario, sino por el contexto en el que aparece es llamada aplicación de reglas contextuales. En este proyecto se detectó que algunas de las entidades que se busca reconocer pueden ser caracterizadas mediante reglas contextuales: esto es secuencias de palabras con determinadas características en determinado orden.

Al investigar herramientas de aplicación de reglas surge en primer lugar Jess Rules [23], un motor de reglas en Java desarrollado por Sandia National Laboratories (no es Open Source). Este motor apunta a la aplicación de reglas de negocio en las que se describe

una situación y una acción a tomar cuando se detecta esa situación. Debido a que Jess no está orientado a la aplicación de Reglas Contextuales en lenguaje natural fue dejado de lado.

Se decidió usar XPath para expresar las reglas contextuales, porque la información en pasos intermedios se almacena en XML. Más adelante veremos en detalle la forma de las reglas contextuales usadas.

3. Análisis y diseño de la solución

En este capítulo se especifican requerimientos, se muestra un caso de ejemplo para entender mejor las funcionalidades de la aplicación y se comentan las decisiones de diseño.

El documento de análisis de requerimientos y el documento de diseño pueden verse en los anexos I y II respectivamente.

3.1. Requerimientos

El requerimiento principal del sistema ODA-Web es ofrecer al docente la posibilidad de reciclar el material que tiene generado y publicado en un sitio web, para educación a distancia.

Dentro del sistema se identificó un solo tipo de usuario: docente o encargado de la elaboración del curso, quien deberá darle formato a los contenidos de un curso para que puedan ser "entendidos" por una plataforma de de educación a distancia.

Para esto son requerimientos del sistema:

a) Reconocer en forma automática objetos de aprendizaje del curso.

b) Permitir que el usuario corrija en forma manual los objetos de aprendizaje detectados en forma automática.

c) Ofrecer al usuario la posibilidad de:
    a. Enriquecer en forma manual los metadatos del curso que no pudieron ser detectados en forma automática.
    b. Generar a partir de su material, paquetes SCORM.

d) Extraer a partir de los contenidos, otros datos (cantidad de definiciones, ejercicios, ejemplos) que puedan servir en un futuro para determinar si el tipo de metodología empleada es expositiva o participativa o si para cada unidad didáctica los contenidos son conceptuales o prácticos.

e) Fácilmente extensible para que pueda reconocer otro conceptos.

Se asume que el proyecto contiene un solo curso. Dentro del mismo puede existir uno o más temas. La salida del sistema es un paquete SCORM por tema extraído. Sin embargo el sistema ODA-Web reconoce ODAs de menor granularidad que están contenidos en los temas. Estos son definiciones, ejercicios, ejemplos, bibliografía, enunciados y teoremas. Algunos de estos surgen de la ontología de contenido de cursos y otros se reconocieron como necesarios a partir del mapa conceptual de la Figura 3.

El paquete SCORM generado por el asistente tiene una organización jerárquica de acuerdo a los temas del curso. Este paquete no tiene secuenciación como está definida

según el Standard de SCORM 2004. De aquí que para empaquetar los contenidos de curso, será suficiente con la versión 1.2 de SCORM. Sin embargo, para poder representar todos los metadatos que se desea, el paquete deberá soportar LOM.

El sistema también permite agregar en forma manual los metadatos de características culturales.

Otro requerimiento no funcional es probar el asistente ODA-Web para transformar el material de Programación 1 a un curso en SCORM.

3.2. Ejemplo

La Figura 9 muestra un ejemplo de una página web de curso que contiene varios ODAs pero no puede ser usado en una plataforma de e-Learning.

Para poder usar el sistema ODA-Web el autor debe proveer esta página al sistema junto con un índice de temas como se ve en la Figura 10.

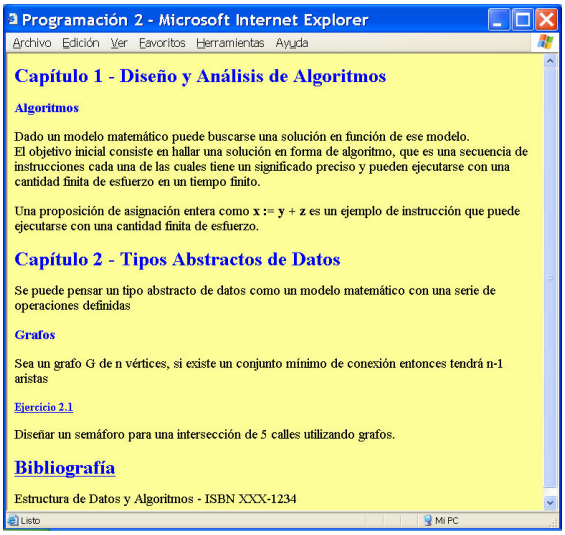


Figura 9 - Página de ejemplo

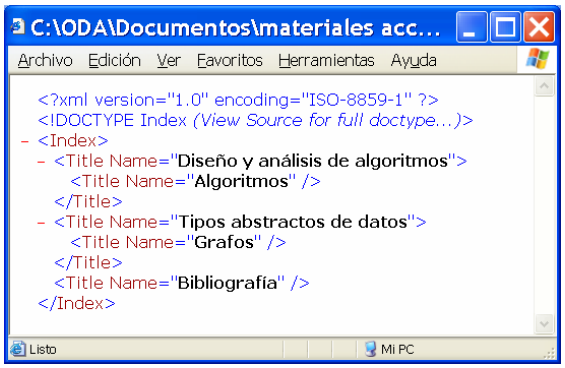
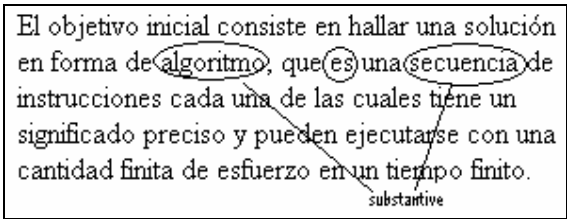
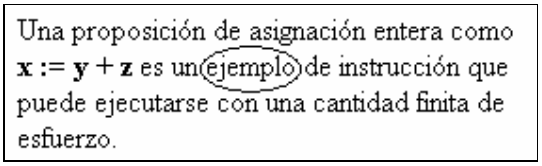


Figura 10 - Archivo con índice de temas

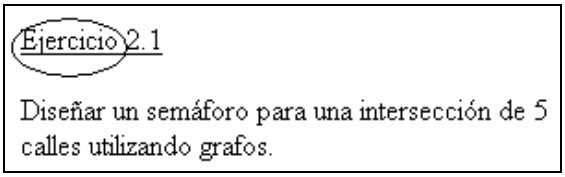
Usando el criterio de marcado en las siguientes figuras, el sistema ODA-Web identifica los ODAs que se muestran en la Figura 8.
Primero se realiza la delimitación de los temas y sub-temas. Luego se realiza la búsqueda contextual de palabras clave.



ODA: Definición.



ODA: Ejemplo



ODA: Ejercicio



ODA: Bibliografía.

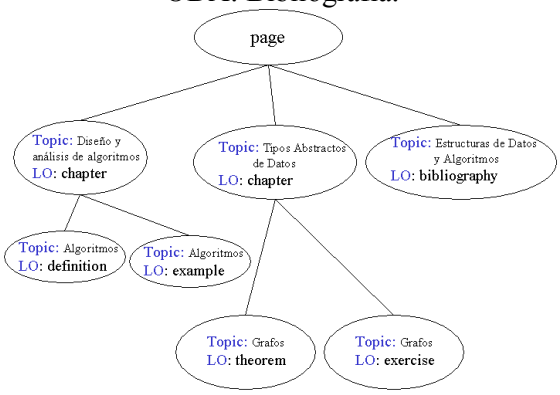


Figura 11 - ODAs de la página de ejemplo de la Figura 9

Para la definición se busco la palabra clave “es” como verbo, seguida y precedida por sustantivos. En el caso del ejercicio se busco la palabra “Ejercicio” como título.

La Figura 11 muestra los ODAs detectados en forma automática dentro de la página de ejemplo. Muestra además la jerarquía de los ODAs: El capítulo 1 – Diseño y Análisis de algoritmos incluye al sub-tema Algoritmos y este a su vez incluye un ODA definición y un ODA ejemplo.

3.3. Arquitectura de la solución

El asistente ODA-Web convierte materiales disponibles en sitios web, en paquetes SCORM enriquecidos con metadatos. La figura 5 muestra la arquitectura de la solución descripta en los siguientes pasos:

1) Toma como entrada las páginas web de cursos y un índice de temas en formato XML. El sistema identifica los temas con el módulo de marcado de temas, y marca además objetos de aprendizaje dentro de estos.
2) El usuario valida el marcado automático y modifica lo que considera necesario.
3) El usuario le pide al sistema que genere los paquetes SCORM.
4) El paquete SCORM puede ser agregado a un repositorio de paquetes SCORM.

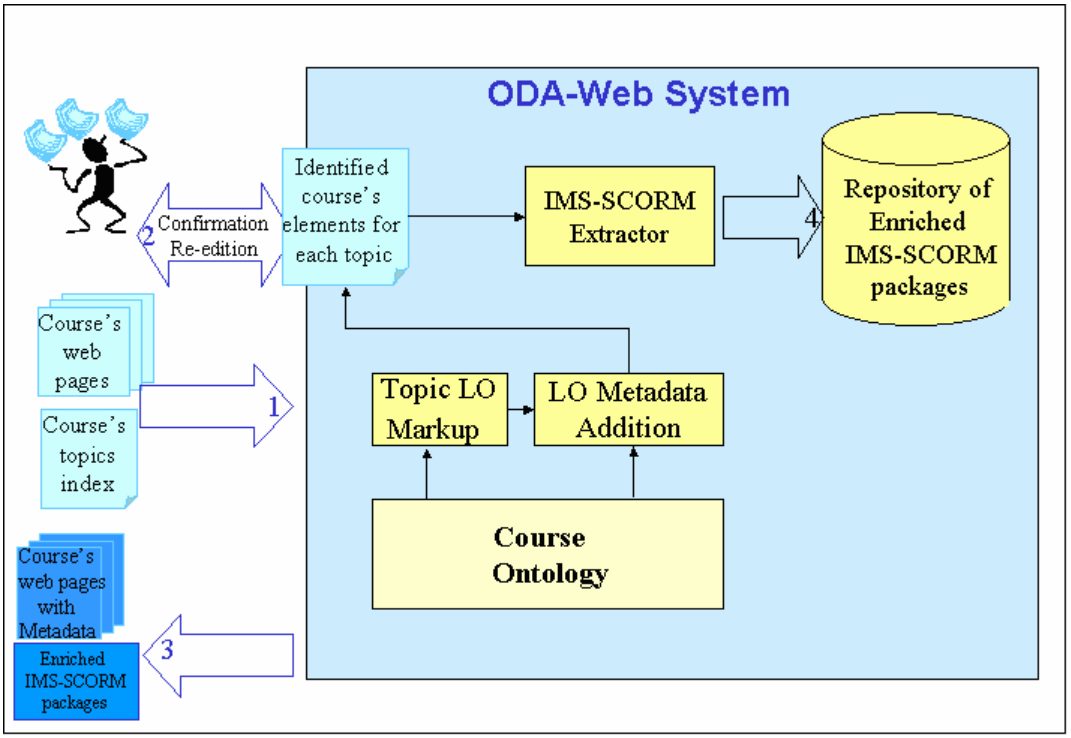


Figura 12 - Arquitectura de la solución

3.3.1. Marcado y organización de ODAs

El sistema ODA-Web reconoce ODAs en diferentes jerarquías, puede reconocer Temas, sub-temas dentro de estos y Objetos dentro de los temas y sub-temas.

El sistema marca el texto del curso en bloques que delimitan los objetos de aprendizaje. Para saber donde empieza y donde termina un objeto de aprendizaje, el sistema utiliza como entrada: a) la estructura del documento html, y b) un índice temático.

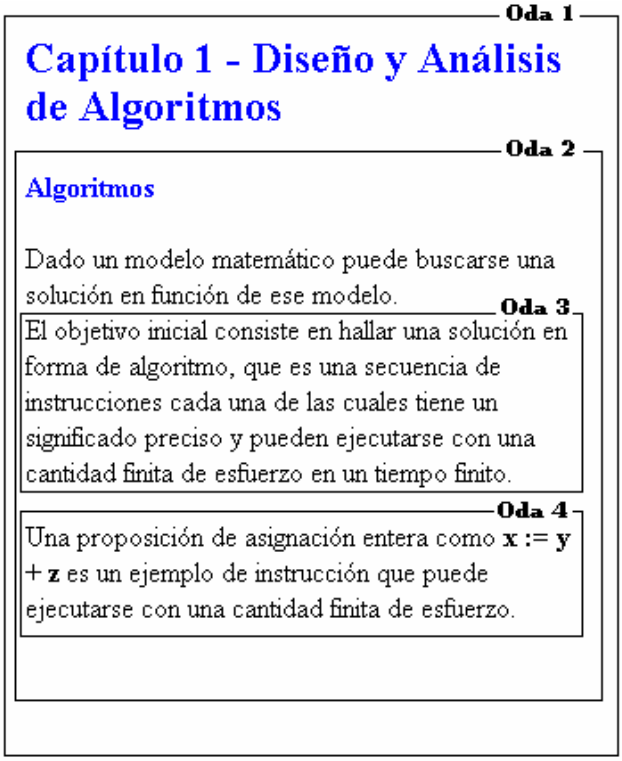


Figura 13 - Ejemplo de organización de ODAs

Explotando la estructura del documento –utilizando tags de párrafos y títulos (como <P> <H1> <H2>)- encuentra los títulos y sub-títulos indicados en el índice temático. Determina además donde empiezan y terminan estos ODAs. Con el índice temático se determinan objetos de aprendizaje jerarquizados. Luego, reconociendo determinadas estructuras y palabras en el texto, se delimitan ODAs (ejercicios, ejemplos) dentro de los temas y sub-temas. Aquí se enfrenta un problema importante y es que el autor de páginas web puede usar distintos tipos de etiquetas para mostrar lo mismo (por ejemplo un texto resaltado o jerarquizar un título).

La Figura 13 muestra el tema “Capítulo 1 – Diseño y Análisis de Algoritmos” que contiene un sub-tema “Algoritmos” y este a su vez contiene adentro una definición y un ejemplo. Se corresponde con los nodos mostrados en la Figura 11.

El marcado del Oda 1 y 2 se obtiene a través del índice temático. Los Odas 3 y 4 se marcan con las reglas creadas a partir de la ontología de dominio y la estructura del documento.

Para la búsqueda ODAs dentro de los temas se asume que un objeto termina donde empieza otro o donde termina el tema o sub-tema. Los Odas 3 y 4 quedan dentro del sub-tema “Algoritmo” (marcado como Oda 2).

En el caso de los objetos de aprendizaje jerarquizados, un ODA padre se puede ver como un metadato de los objetos de aprendizaje hijos. En la Figura 1, el Oda 1 y 2 son un metadato de los Odas 3 y 4. Por ejemplo, podemos decir que la definición y el ejemplo dados pertenecen al tema Algoritmos. Un ODA queda vinculado a un tema. Inicialmente la ordenación que tendrá será la propuesta en el material existente.

3.3.2. Inferencia de metadatos

El asistente ODA captura los metadatos que los materiales contengan (por ejemplo: las páginas pueden o no contener un metadato "author", si está presente el sistema lo obtiene en forma automática). Esto se hace aprovechando los metadatos que HTML tiene previstos.

Además se brinda la posibilidad de ingresar metadatos de LOM, que luego serán incorporados en el paquete SCORM que se brinda de salida.

Dado que tenemos Objetos de Aprendizaje en un formato estándar (más específicamente SCORM), es necesario conocer las características de estos, para poder evaluarlos, clasificarlos y decidir si son útiles para otros cursos, es decir para permitir su reutilización.

La caracterización de los objetos se hace a través de los metadatos. El estándar de metadatos para Objetos de Aprendizaje es LOM (Learning Object Metadata) [4]. Los paquetes SCORM incluyen los metadatos de LOM para caracterizar los ODA. En general los distintos LCMS solo permiten al usuario la posibilidad llenar un subconjunto de la información que LOM permite tener. Por esto es que es necesario adaptar cada LCMS de una forma particular para que puedan manejar la mayor cantidad de metadatos posible.

3.3.3. Diagrama de Componentes

A continuación vemos el diagrama de clases correspondiente al módulo de marcado automático de ODAs Temas y de Objetos de Aprendizaje dentro de los Temas.

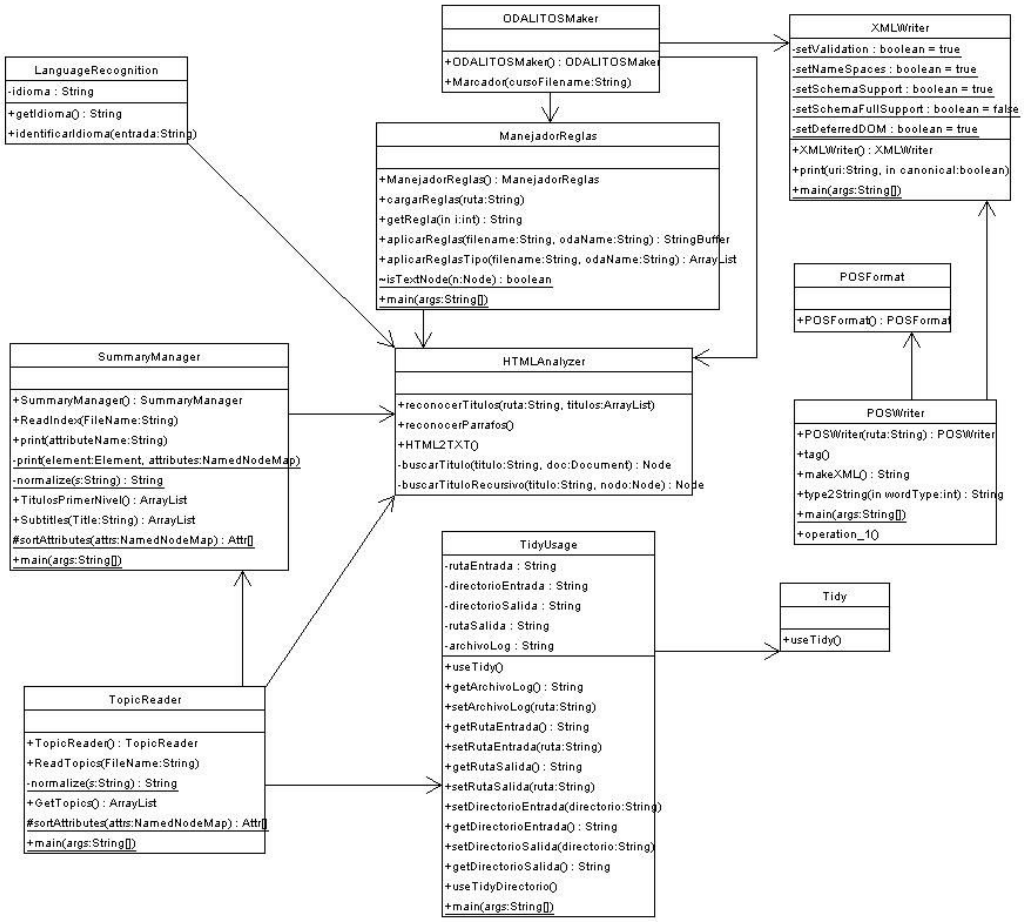


Figura 14 - Modulo de Marcado Automático - Diagrama de clases

Los módulos Tidy, XMLWriter, ManejadorReglas, SummaryManager y HTMLAnalyzer son módulos de servicios a los demás módulos. Los módulos que contienen la lógica de marcado automático son el TopicReader, y el módulo ODAlitosMaker. POSWriter, TidyUsage y LanguageRecognition son módulos de marcados intermedios.

TopicReader se encarga de reconocer los temas de un curso a partir de las páginas HTML y un índice dado en formato XML. Los archivos de salida quedan en un directorio distinto al de entrada. El manual de usuario especifica como configurar cual será el directorio de salida. La salida del TopicReader son archivos XML como se mostrará en la Figura 15.

ODAlitosMaker usa el módulo ManejadorReglas para aplicar las reglas en XPath y retorna archivos XML con los ODAs marcados dentro de cada Tema.

POSWriter lee cada uno de los temas reconocidos previamente y aplica el marcado de part-of-speech.

LanguageRecognition reconoce el idioma del curso.

TidyUsage corrige las páginas HTML para que puedan ser leídas como XML.

3.3.4. Diseño de Componentes

Modulo de Marcado Automático

El módulo de marcado automático se puede dividir en un módulo de Marcado de Temas (MMT) y un módulo de marcado de ODAs dentro de los temas (MMO).
El objetivo del Módulo de Marcado de Temas (MMT) es identificar comienzo y fin de temas. Los temas a ser identificados son provistos a través de un índice de temas. El principal problema que enfrenta este módulo es determinar el alcance de cada tema.
A continuación se describen las dos principales tareas que realiza este módulo: (i) identificar título de los temas, (ii) determinar el alcance de cada tema. La salida del MMT es utilizada por el Módulo de Marcado de ODAs (MMO).

Un índice de temas -como el que se ve en la Figura 10 - se implementa utilizando un archivo XML. El archivo XML es usado para buscar los nombres de los temas en el contenido de los tags de título de html (por ejemplo se buscan los nombres de los temas dentro de los tags h1, h2, h3, h4, etc).

En principio el módulo identifica temas dentro de los archivos HTML. El MMT, procesa archivos html y produce archivos XML con *tags* que identifican los ODAs para cada tema del curso.

La estructura HTML es usada para identificar las áreas de los documentos correspondientes al tema (i.e. los nombres de los temas son presentados en alguna estructura HTML que corresponde a los títulos).

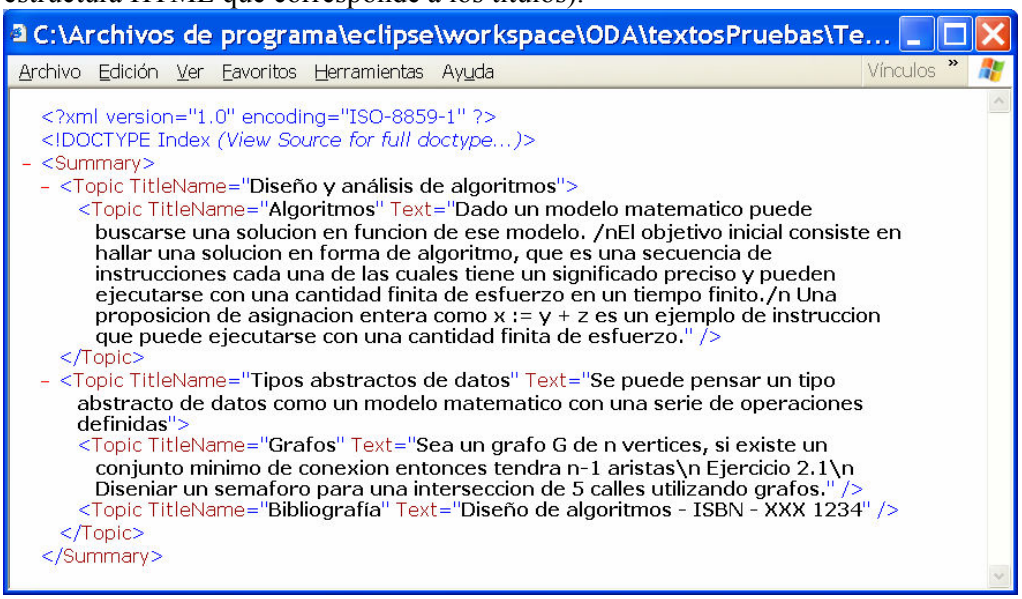


Figura 15 - Ejemplo archivo de salida de marcado de temas

El contenido entre los tags de html identificados, es marcado como el nodo contenedor de un tema. Un nodo contenedor –delimitado por los tags- puede contener otros nodos contenedores que serán considerados como sub-temas del primero. Un nodo contiene otros cuando existe una correspondiente estructura anidada de html. Los nodos contenedores son aquellos que son definidos por las siguientes etiquetas de html (o lo que usando heurísticas se corresponda a ese formato): Title, Table, P, DL, OL, UL, H1, H2, H3, H4, H5. Usando heurísticas, los formatos equivalentes son por ejemplo una palabra en letra negrita o más grande que el tamaño medio de la letra de la página.. La figura 4 muestra un árbol con los nodos contenedores identificados en el ejemplo.

Siguiendo con el ejemplo presentado en la sección 3.2, la Figura 11 muestra como se marcan internamente los temas.

Cuando las áreas del texto que se corresponde con cada tema son delimitadas el MMO debe reconocer los ODAs dentro cada tema. Para ejecutar este proceso el módulo usa dos metodologías combinadas (i) los términos, o sinónimos de estos, correspondientes a los elemento de la ontología de cursos (i.e. Ejercicio, Ejemplo, Definición, etc) son identificados en títulos o como parte de un texto; (ii) al no haber términos o sinónimos de los términos buscados, es necesario del uso de reglas gramaticales de lenguaje (RGL) que describan la forma contextual más frecuente de aparición, para indicar la posible existencia de estos elementos conceptuales.

Como en esta etapa ya están reconocidos los temas, nos olvidamos de aquí en más del problema de volver a encontrar el nombre del título en el medio de un párrafo y confundirlo nuevamente con el comienzo de un tema.

Las reglas deben ser especificadas en XML para el lenguaje que se está utilizando. Para idioma español, algunas de ellas son descriptas en la Figura 16.

1.	<substantive> + “es” + <substantive>
2.	<verb> + <substantive>+ “como” + <substantive> + [“que” + <verb>]

Figura 16 - Ejemplo de RGL para marcar Definiciones

Para aplicar el conjunto de reglas gramaticales usamos primero un marcador Part-of-speech, QTag de Balie [36], el cual identifica la categoría de cada palabra en una sentencia (sustantivos, verbos, adjetivos, etc), y el resultado es guardado por el MMO en un archivo XML. Luego aplicamos las reglas gramaticales que son escritas en XPath [25] como consultas.

Por ejemplo la siguiente regla en XPath se corresponde con la primera RGL mostrada en la Figura 16:

//sentence[word/text()='es' and word/@type='POS_VERB' and word[text()='es' and @type='POS_VERB']/preceding-sibling::[@type='POS_NOUN'] and word[text()='es' and @type='POS_VERB']/following-sibling::*[@type='POS_NOUN']]/@id*

La Figura 17 es un ejemplo del formato que tiene la salida del Part-Of-Speech Tagger:

```
- <sentence id="3">
    <word type="POS_DETERMINER">el</word>
    <word type="POS_VERB">objetivo</word>
    <word type="POS_NOUN">inicial</word>
    <word type="POS_VERB">consiste</word>
    <word type="POS_PREPOSITION">en</word>
    <word type="POS_VERB">hallar</word>
    <word type="POS_DETERMINER">una</word>
    <word type="POS_NOUN">solución</word>
    <word type="POS_PREPOSITION">en</word>
    <word type="POS_NOUN">forma</word>
    <word type="POS_PREPOSITION">de</word>
    <word type="POS_NOUN">algoritmo</word>
    <word type="POS_NOUN">,</word>
    <word type="POS_PRONOUN">que</word>
    <word type="POS_VERB">es</word>
    <word type="POS_DETERMINER">una</word>
    <word type="POS_NOUN">secuencia</word>
    <word type="POS_PREPOSITION">de</word>
    <word type="POS_NOUN">instrucciones</word>
    <word type="POS_DETERMINER">cada</word>
```

Figura 17 - Ejemplo de salida del marcado POS

XPath permite buscar palabras dentro de una categoría seguidas o precedidas por otra palabra en otra categoría.

Se decidió usar Balie porque tiene identificación del lenguaje, marcado Part-of-Speech para español y está implementado en Java.

Un ejemplo de uso del marcador Part-of-Speech se muestra en la figura 10 y 11. Donde la secuencia de sustantivos "algoritmo", "secuencia" y "tipo abstracto de datos", "modelo matemático" son encontrados por las reglas 1 y 2, respectivamente., para que los bloques sean identificados como *definiciones*

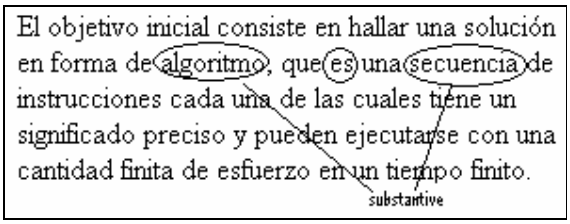


Figura 18 - Texto correspondiente a una definición

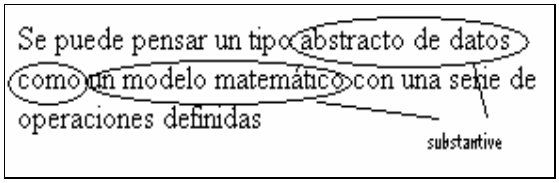


Figura 19 - Texto correspondiente a una definición

Otro ejemplo de uso de Part-of-Speech y RGL es el ejemplo mostrado en la figura 12. Para un teorema los elementos de un teorema, el módulo busca las palabras “hipótesis”, “tesis”, “demostración” o una sentencia con la siguiente estructura:

“si” + <expression> + “entonces” + <expression>

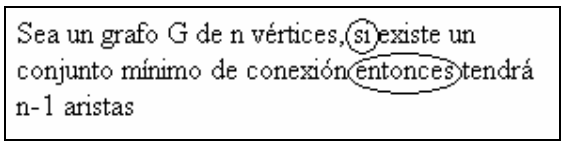


Figura 20 - Texto correspondiente al comienzo de un Teorema

La regla en XPath correspondiente a la Figura 20 es la siguiente:

//sentence[word/text()='si' and word/@type='POS_CONJUNCTION' and word[text()='si' and @type='POS_CONJUNCTION']/preceding-sibling::[text()='entonces'] or word[text()='si' and @type='POS_CONJUNCTION']/following-sibling::*[text()='entonces']]/@id*

Notar que las reglas en XPath devuelven el identificador de la oración. Este identifica la oración donde comienza el ODA. El fin del mismo será donde termine el tema o donde comience otro ODA. Notar además que las palabras a buscar están siempre en minúscula porque –como muestra la Figura 17 - la salida del POS tagger queda siempre en minúscula.

Modulo de agregado de Metadatos

Este módulo apunta a la provisión automática de valores de metadatos para cada ODA identificado. Con este fin, el módulo toma como entrada la página web y genera los metadatos para la identificación de los ODAs. La tabla 1 muestra que elementos de LOM pueden ser provistos automáticamente por el módulo. La estructura de los elementos es provista en orden lineal. El elemento *author*, es automáticamente extraído si y solo si el nombre del autor está presente en la página web. El *status* es definido por defecto como la primera versión liberada, porque consideramos que una página web es una versión pública, si no es este el caso el autor tiene la posibilidad de cambiar estos valores durante la fase de confirmación. Es esta fase, los ODAs temas identificados, con los valores de metadatos asignados están presentes para la posible confirmación o re-edición manual. Cuando estén confirmados el sistema deja la información disponible para la creación del paquete SCORM.

Dada una página web se pueden encontrar en ella algunos metadatos previstos en HTML[W3C]. Algunos de estos metadatos se corresponden a algunos de los metadatos especificados en LOM.

Un estudio realizado sobre la plataforma AdaptWeb[15], determinó un conjunto de atributos de LOM que se pueden inferir y completar en forma automática, cuales podrían ser obtenidos en forma semi-automática y cuáles requieren del usuario para poder disponer de ellos.

De los 64 tipos de metadatos especificados en LOM, el asistente ODA puede llegar a extraer 25 metadatos en forma automática y el resto en forma manual. La Tabla 3 presenta la forma de obtención de metadatos que realiza ODA.

De los metadatos obtenidos de forma automática: todos los *identifier* se generan para el metadato *entry* y con una constante para *catalog*.

General.Identifier.catalog es por omisión el catalogo del In.Co.de Facultad de Ingeniería, el valor será InCo-Fing-UdelaR, *General.Language* se obtiene utilizando una de las funcionalidades de Balie [36] que detecta el idioma; *General.Title* es el título del tema; *General.Description* es por defecto el título de tema también; *General.Keywords* se puede obtener –si existe- de la metadata de la página web;

Para la categoría Meta-Metadata: *Meta-metadata.Language* es constante, , se refiere al lenguaje de los metadatos, el valor por defecto es "español". *Meta-metadata.schema* se generará con el valor LOM v1.0.

En LifeCycle: *LifeCycle.version* es 1.0 porque consideramos que lo generado en forma automática es una primera versión del curso a distancia. *LifeCycle.Contribute* es el dato del autor se obtiene de la página web, si está presente. LifeCycle.status asumirá el valor "final".

Educational.LearningResourceType tomará como valor lo que detectamos con el marcado automático, es decir, si es una definición, tema, ejercicio, ejemplo, u otro y *Educational.Language* será el lenguaje en el que está hecho el objeto de aprendizaje.

Tabla 3 - Tipo de Captura de Metadatos de LOM

LOM element	Captura
1. general	
1.1 identifier	Automático
1.1.1 catalog	Automático
1.1.2 entry	Automático
1.2 title	Automático
1.3 language	Automático
1.4 description	Manual
1.5 keyword	Manual
1.6 coverage	Manual
1.7 structure	Automático
1.8 aggregation level	Automático
2. lifeCycle	
2.1 status	Automático
2.2 version	Automático
2.3 contribute	Semi-automático
2.3.1 role	Semi-automático
2.3.2 entity	Semi-automático
2.3.3 Date	Automático
3. metaMetadata	
3.1 identifier	Automático
3.1.1 catalog	Automático
3.1.2 entry	Automático
3.2 contribute	Automático
3.2.1 role	Automático
3.2.2 entity	Automático
3.2.3 date	Automático
3.3 metadata schema	Automático
3.4 language	Automático
4. technical	
4.1 format	Automático
4.2 size	Automático
4.3 location	Automático
4.4 requirement	Manual
4.4.1 orComposite	Manual
4.4.1.1 type	Manual
4.4.1.2 name	Manual
4.4.1.3 minimun version	Manual
4.4.1.4 maximum version	Manual

LOM element	Captura
4.5 installation remarks	Manual
4.6 otherPlatformRequirements	Manual
4.7 duration	Manual
5. educational	
5.1 interactivity type	Manual
5.2 learningResourceType	Automático
5.3 interactivity level	Manual
5.4 semantic density	Manual
5-5 intended end user role	Manual
5.6 context	Manual
5.7 typical age range	Manual
5.8 difficulty	Manual
5.9 typical learning time	Manual
5.10 description	Manual
5.11 language	Automático
6. rights	
6.1 cost	Manual
6.2 copyrightAndOtheRrestrictions	Manual
6.3 description	Manual
7. relation	
7.1 kind	Manual
7.2 resource	Manual
7.2.1 identifier	Automático
7.2.1.1 catalog	Automático
7.2.1.2 entry	Automático
7.2.2 description	Automático
8. annotation	
8.1 entity	Manual
8.2 date	Automático
8.3 description	Automático
9. classification	
9.1 purpose	Manual
9.2 taxonPath	Manual
9.2.1 source	Manual
9.2.2 taxon	Automático
9.2.2.1 id	Automático
9.2.2.2 entry	Automático

Dado que no es posible completar todos los metadatos de forma automática, el usuario tiene la posibilidad de completar los metadatos de su curso antes de generar el paquete SCORM. Por más información consultar el manual de usuario en los documentos anexos.

En principio se pensó en restringir que el usuario no pudiera generar el paquete SCORM hasta tener completados al menos los metadatos especificados en UK LOM [8], como restricción para una mejor calidad de los datos. La restricción fue por el momento levantada por considerarla en contra del objetivo de facilitar la tarea al docente.

Obtención de paquetes SCORM

La obtención de paquetes SCORM se realiza a través de RELOAD. El sistema ODA-Web genera para cada tema una página en el formato original, un archivo XML con los metadatos de LOM y los archivos con los objetos de aprendizaje que se reconocieron dentro de los temas, estos últimos sin el formato original. Con esto el usuario podrá crear un paquete SCORM con la metadata de LOM incluida usando la herramienta RELOAD.

Por la forma en que se genera la salida del sistema ODA-Web, sugerimos generar un paquete para cada tema. Los paquetes son retornados al autor en un archivo zip pudiendo integrarse a un repositorio.

4. Implementación y Testeo

En este capítulo explicaremos los detalles técnicos de la implementación, los mecanismos utilizados para el testing, las métricas y los resultados obtenidos de las mismas. Durante el desarrollo se siguió el enfoque propuesto por Vallespir en sus tesis de maestría "Actividades para Mejorar la Calidad" [42]

4.1. Herramientas usadas para la implementación

Para la implementación se elige java 1.4.2. por su portabilidad entre distintas plataformas y su extenso uso. Además se usan las siguientes bibliotecas:

- Balie para lo que tiene que ver con procesamiento de lenguaje natural
- HTMLParser [33] para el procesamiento de páginas HTML
- Xerces[41] para lo que tiene que ver con lectura y escritura de documentos XML.

Se uso Eclipse 3.1.0 como IDE (Integrated Development Environment). Para la creación de la interfaz gráfica se usó el plugin de eclipse, Visual Editor.

Con respecto a la implementación del reconocedor de temas, encontramos que muchas veces –dentro de una página HTML – un título de tema, está a un nivel más interno que los subtítulos. Por esta razón, el marcado de temas utiliza los tags de la estructura HTML para encontrar los nombres de los títulos pero no para delimitar un Tema.

El Part-Of-Speech de Balie – QTag – tiene inconvenientes en reconocer el fin de las oraciones cuando éstas están cargadas de caracteres no alfabéticos. Al encontrar un tag de HTML, QTag asume que finalizó la oración. Esto nos llevó a aplicar el marcado POS sobre texto plano. Como el reconocimiento de ODAs que están dentro de los temas utiliza la salida del POS, estos quedan en texto plano (sin énfasis, negritas, indentación, etc). Dejamos para futuras versiones resolver el problema de mantener la trazabilidad de los formatos originales del curso, para los ODAs que se encuentran dentro de un tema.

RELOAD es una herramienta para generar paquetes SCORM de licencia de uso gratuita y de código abierto. Originalmente se pensaba integrar RELOAD como un módulo más, es decir usarlo como librería para generar los paquetes SCORM. Al estudiar su arquitectura se determinó que no estaba diseñada para ser usada como biblioteca ya que los módulos que sirven para la generación de paquetes SCORM están altamente acoplados a su interfaz gráfica.

Actualmente existen dos versiones de RELOAD disponibles: la versión 2.0.2 –la última accesible en el CVS- contiene el código fuente completo, genera SCORM 1.2 y no tiene soporte para LOM; la versión 2.5 agrega SCORM 2004 y LOM pero no fue liberado aún el código fuente.

ADL - junto con la universidad de Carnegie Mellon - tomaron el código 2.0.2 de RELOAD y lo modificaron agregando SCORM 2004 con secuenciamiento simple y

LOM, esta aplicación fue llamada RELOAD 2004 y fue publicada en el sitio oficial de RELOAD recién en Noviembre de 2006 como RELOAD 2.5.2, sin publicarse aún su código fuente.

Una de las opciones que se manejó fue agregar LOM a RELOAD 2.0.2 y utilizar su interfaz grafica para agregar las funcionalidades de nuestro sistema, dicho de otro modo embeber nuestro sistema en RELOAD. Sin embargo, estaríamos programando sobre una versión que sería obsoleta en poco tiempo, por lo que no tiene sentido duplicar el esfuerzo para hacer lo que otros ya están haciendo (generar paquetes SCORM con LOM incluido). Nuestra decisión fue no trabajar sobre una versión vieja de RELOAD.

Se decide trabajar los metadatos siguiendo el estándar LOM, pero sin atarnos a una representación específica. El sistema ODA-Web genera los metadatos de LOM y las estadísticas como una extensión, dejándolos en un archivo XML fuera del ims-manifest.xml pero incluido en el paquete SCORM y relacionado con el tema al que corresponde. Luego el usuario puede usar la última versión de RELOAD para generar el paquete SCORM.

4.2. Testeo Realizado

En la segunda y tercera iteración se realizan pruebas con el material de curso presencial de Programación 1.

A continuación presentamos el testeo del módulo de aplicación de reglas. La

Tabla 4 muestra el resultado del testeo y las gráficas de las siguientes figuras esquematiza que tanto se separa en errores el marcado automático del marcado manual.

Aquí mostraremos el resultado de las pruebas con los siguientes archivos:
http://www.fing.edu.uy/inco/cursos/prog1/pm/uploads/Materiales/clase1.html
incluye en su totalidad el tema Componentes Básicos.
http://www.fing.edu.uy/inco/cursos/prog1/pm/uploads/Materiales/clase2.html
contiene el tema Entrada y Salida.
http://www.fing.edu.uy/inco/cursos/prog1/pm/uploads/Materiales/Practico0.html
es el correspondiente al Práctico 0.
http://www.fing.edu.uy/inco/cursos/prog1/pm/uploads/Materiales/Practico2.html
es el correspondiente al Practico 2.

Tabla 4 – Resultado del testeo del módulo de aplicación de reglas

Tema	Componentes Básicos	Entrada y Salida	Practico 0	Practico 2	Presentación
Bibliografía	0	0	0	0	1
Definiciones	11	9	1	13	1
Ejemplo	12	3	4	24	0
Ejercicio	0	0	2	28	3
Enunciados	2	0	0	1	0
Bibliografía	0	0	0	0	1
Definiciones	5	5	0	3	1
Ejemplo	12	3	4	24	0
Ejercicio	0	0	0	27	0
Enunciados	2	0	0	0	0
Bibliografía	0	0	0	0	1
Definiciones	5	5	0	3	1
Ejemplo	12	3	4	24	0
Ejercicio	0	0	6	27	0
Enunciados	2	0	0	0	0

Detectados en forma manual
Detectados automáticamente
Detectados automáticamente y correctos

"Componentes Básicos" y "Entrada y Salida" tienen un alto contenido de Definiciones, Ejemplos y no contienen ningún Ejercicio. Los temas Practico 0 y Practico 2 tienen un alto contenido de Ejercicios. Las cantidades exactas de Definiciones, Ejercicios, Ejemplos, Bibliografía y Enunciados aparecen en la Tabla 4 marcadas en amarillo.

Se eligen para mostrar los resultados con el Práctico 0 y el 2 porque son claramente distintos en la forma de delimitar los ejercicios. En el caso del práctico 0, del título principal se puede inferir que lo que veremos a continuación son ejercicios, en el caso del práctico 2, se indica el comienzo de cada ejercicio con la palabra "ejercicio". Como veremos en los resultados, eso hace una diferencia importante en la efectividad del reconocimiento de los ODAs, ya que una de las reglas en XPath para encontrar los ejercicios es buscar esta palabra en el texto.

Las reglas para encontrar ejercicios son:

```
//sentence[word/text()='ejercicio']/@id
//sentence[word/text()='ejercicios']/@id
//sentence[word/text()='práctico']/@id
//sentence[word/text()='practico']/@id
//sentence[word/text()='Problema']/@id
//sentence[word/text()='Pregunta']/@id
```

En la Figura 21 se puede observar las diferencias en la forma de redactar las páginas.

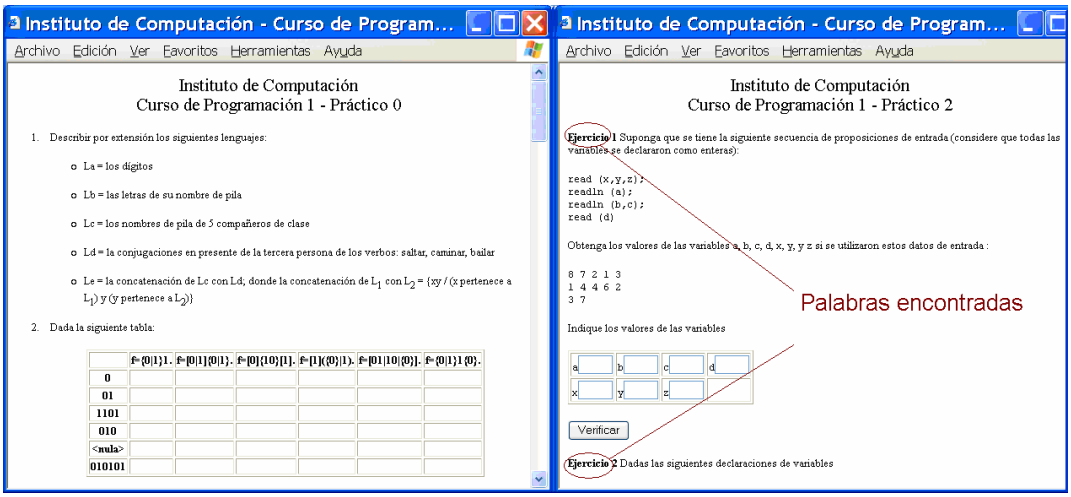


Figura 21 - Prácticos 0 y 2 de Programación 1

Las páginas correspondientes a las clases teóricas tienen todas una estructura similar y para todas se obtuvo porcentajes de error similares. Así que elegimos mostrar solo los resultados correspondientes a Componentes Básicos y Entrada y Salida que son representativas del resto del testeo.
La Figura 22 muestra un esquema del resultado del marcado de texto.

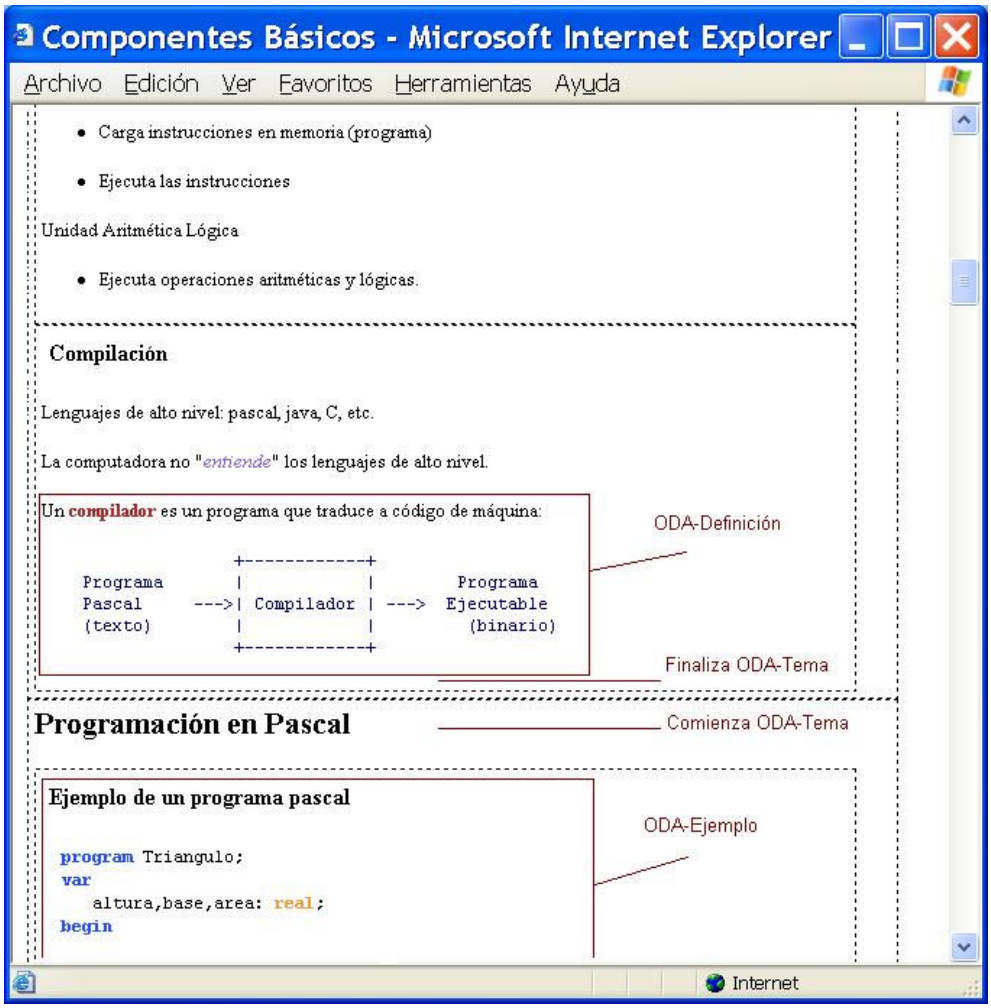


Figura 22 - Esquema del resultado del marcado en Componentes Básicos

Las Figura 23, Figura 24 y Figura 25 muestran los resultados obtenidos en el testeo del MMO en la búsqueda de Definiciones, Ejercicios y Ejemplos respectivamente. Hay tres barras -una azul, una violeta y una blanca- por cada tema al que se le aplicó el testeo. Las barras azules muestran las cantidades detectados en forma manual. Las barras violetas muestran las detectadas automáticamente por el módulo y las barras blancas muestran las cantidades de ODAs detectados correctamente en forma automática.

Para comprender mejor las gráficas, para cada tema definimos: a) la separación entre las barras azules y blancas muestran que cantidad de ODAs existentes fueron detectados en forma incorrecta y; b) la separación entre las barras blancas y la violetas muestra el porcentaje de falsos positivos encontrados.

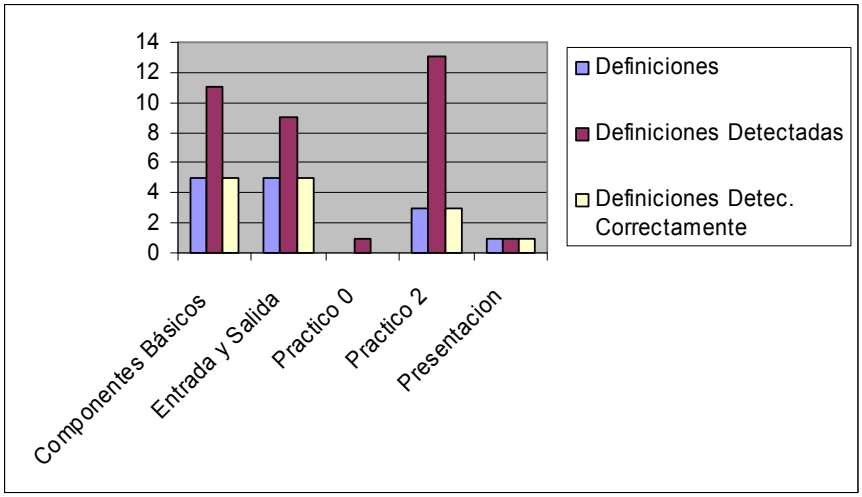


Figura 23 - Resultados de Reconocimiento de Definiciones

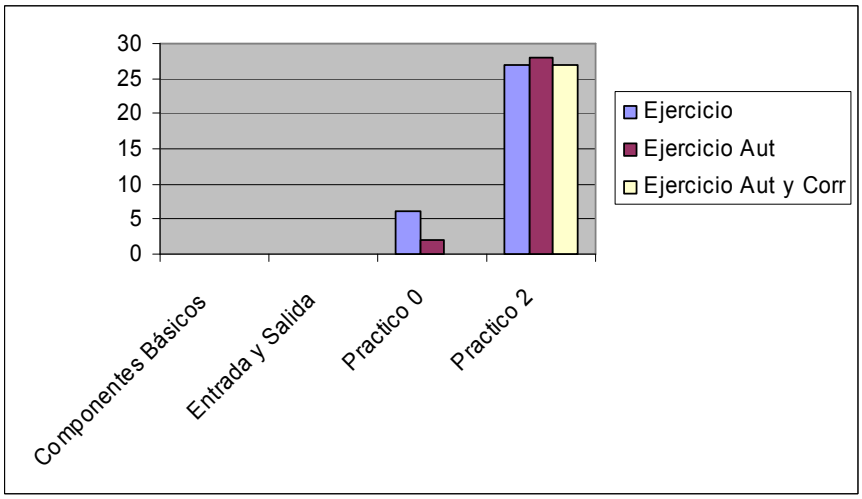


Figura 24 - Resultado de Reconocimiento de Ejercicios

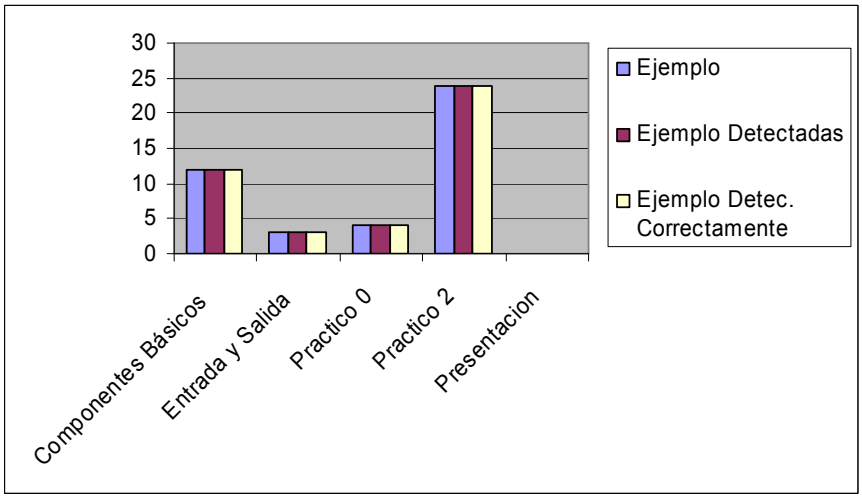


Figura 25 - Resultado del reconocimiento de ejemplos

En el caso de las definiciones encontramos que el error tipo a) es prácticamente nulo y que el error tipo b es alto.

Para el caso de los ejemplos encontramos prácticamente nulo el error tipo a y tipo b.

Para los ejercicios es interesante notar la diferencia entre el práctico 0 y práctico 2. En el práctico 2 el error es menor al 4% y en el práctico 0 el error es muy alto. En el tema práctico 0 los ejercicios se encuentran solo numerados y en el práctico 2 cada ejercicio es identificado al comienzo por la palabra ejercicio. Esto último explica porqué para el tema Práctico 0 es tan baja la cantidad de ejercicios detectados automáticamente.

Los problemas encontrados se dan por las siguientes causas:

- Existen oraciones que no terminan en punto. Como el marcado POS junta dos oraciones, pueden quedar marcados juntos una definición y un ejercicio o reconocer la oración bajo otra regla.

- La intersección del resultado de la aplicación de las reglas definidas, no devuelven conjuntos disjuntos. Por lo tanto, en ocasiones un ODA queda clasificado en dos categorías. La Figura 26 muestra el texto donde se encontrará un caso de estos, y la Figura 27 muestra un esquema de como queda marcado el texto.

- Se reconocen como definiciones frases que no lo son, esto se debe a que la regla es muy general.

- No detecta correctamente los ejercicios cuando el texto no especifica que es un ejercicio.

- No detecta si lo que reconoció, es sensible al contexto. En el caso de la Figura 27 los dos últimos ODA-Ejemplos reconocidos dependen del Ejercicio 18 y por lo tanto no son reutilizables. En este caso, el marcado aporta a las estadísticas de la cantidad de ejemplos en el texto, pero no es correcto tomarlo como ODA.

Para mejorar el problema de detección de ejercicios necesitaríamos una herramienta que detecte el tiempo verbal de cada oración. De esa forma podríamos suponer que las oraciones imperativas dentro de un texto de estudio son ejercicios.

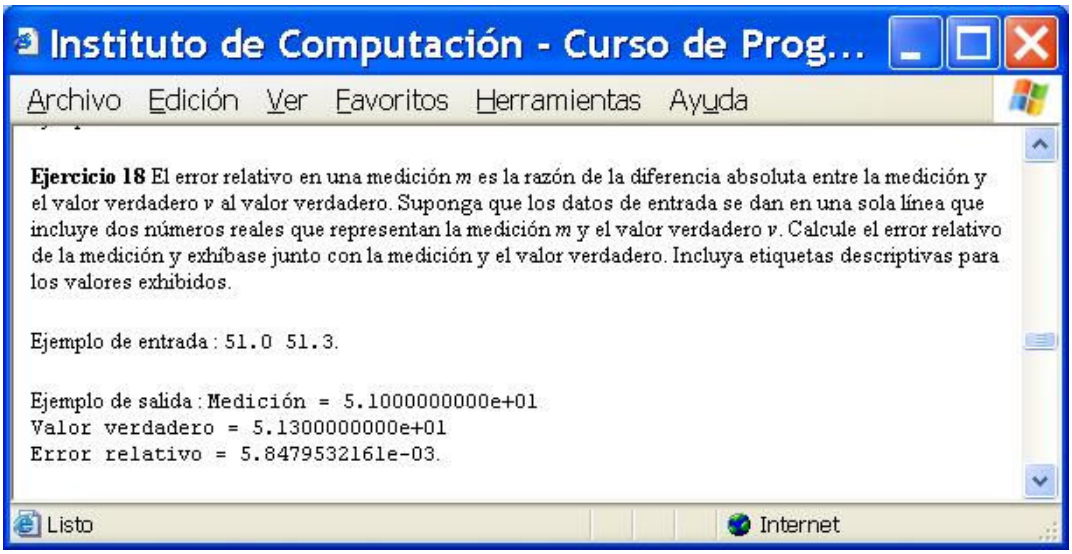


Figura 26 - Página correspondiente al Práctico 2

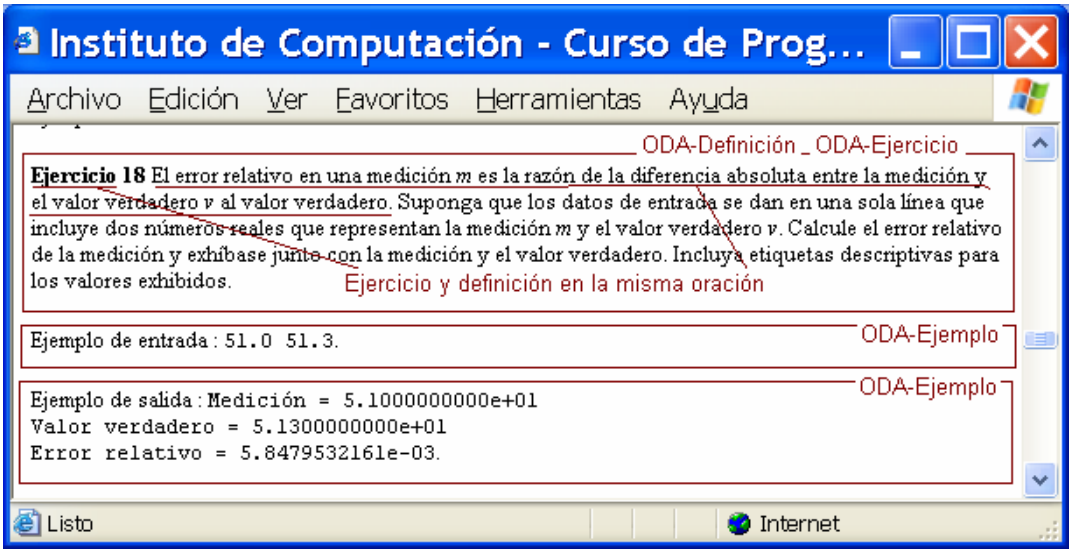


Figura 27 - Esquema de la aplicación del marcado al Practico 2

4.3. Métricas

Con el fin de medir el beneficio del sistema ODA-Web, se compara cuanto lleva hacer el trabajo en forma manual y cuanto lleva hacerlo automáticamente con el asistente.

Para esto se le pidió a una persona con experiencia docente, que realizara en un texto de curso, un marcado de temas, sub-temas, ejercicios, definiciones, ejemplos, enunciados y bibliografías.

Los contenidos usados para las pruebas fueron los capítulos: Componentes Básicos, Entrada y Salida, Practico 0, Practico 2 y Presentación del curso de Programación 1.

El marcado de los temas en forma manual demoró 2 horas y el marcado automático demora 3 minutos.

5. Conclusiones y Trabajos Futuros

El proceso de extracción de Objetos de Aprendizaje es semi-automático, pero logra ahorrar camino para la generación de los paquetes SCORM, especialmente en la generación de metadatos del curso. Con esto se cumplió el principal objetivo de este proyecto. Queda por mejorar la exactitud de las reglas contextuales que definen cómo se reconocen los ODAs, por ejemplo para definiciones.

El proceso de marcado utiliza: i) un índice con los títulos de los temas a reconocer y ii) un conjunto de reglas gramaticales para encontrar determinadas estructuras que indican la existencia de ODAs.

El autor de un curso a distancia necesita conocer con exactitud los valores que serán agregados como metadatos. Este enfoque presenta una guía para generar parte de esos metadatos automáticamente y editar y completar lo que falte.

El trabajo futuro más inmediato es poner en producción los Repositorios de Paquetes SCORM creados por los proyectos de Ingeniería de Software y que puedan ser probados los módulos de captura de características culturales [18].

Dado que las reglas son fácilmente modificables la extracción de información se puede generalizar para otros tipos de documentos o incluso mejorarlas para esta herramienta. Por ejemplo se podría usar en preparaciones para conferencias para obtener todos los Resumenes (*Abstract*) de los *papers* presentados. O podría usarse para chequeo de calidad de documentos, por ejemplo podríamos verificar si todos los documentos tienen resumen y conclusión. Para esto solamente se necesita cambiar los archivos de definición de reglas.

En base a los metadatos que el Asistente ODA agrega se posibilita la evaluación semi-automática de características pedagógicas. En trabajos futuros se pretende extender el sistema ODA-Web para implementar un asistente pedagógico.

En [30] y [31] se habla de la importancia de generar ODAs tomando en cuenta objetivos pedagógicos. Consideramos importante analizar la re-utilización del sistema ODA-Web para la creación de un Asistente Pedagógico.

Asistente Pedagógico

A continuación presentamos algunas de las funcionalidades básicas para el asistente pedagógico, así como algunos conceptos relevantes para entender las funcionalidades.

Un asistente pedagógico consiste en una herramienta, capaz de brindar al docente, lineamientos pedagógicos respecto al armado de un curso.

El problema del Asistente Pedagógico es un problema muy vasto y poco explorado, consideramos que habría que comenzar realizando un relevamiento de trabajos relacionados.

Usando el sistema ODA-Web, el asistente obtendría - en una primera etapa - información estadística del tipo de curso que se está armando. A partir de esto se podría dar una evaluación del estado pedagógico del curso. Es necesario definir como dar dicha evaluación.

Vimos que la fórmula de evaluación pedagógica del curso, podría estar basada en los siguientes criterios y pasamos a explicarlos:

a) Grado ofrecido de participación del estudiante en el curso
b) Niveles de formación incentivados en el curso

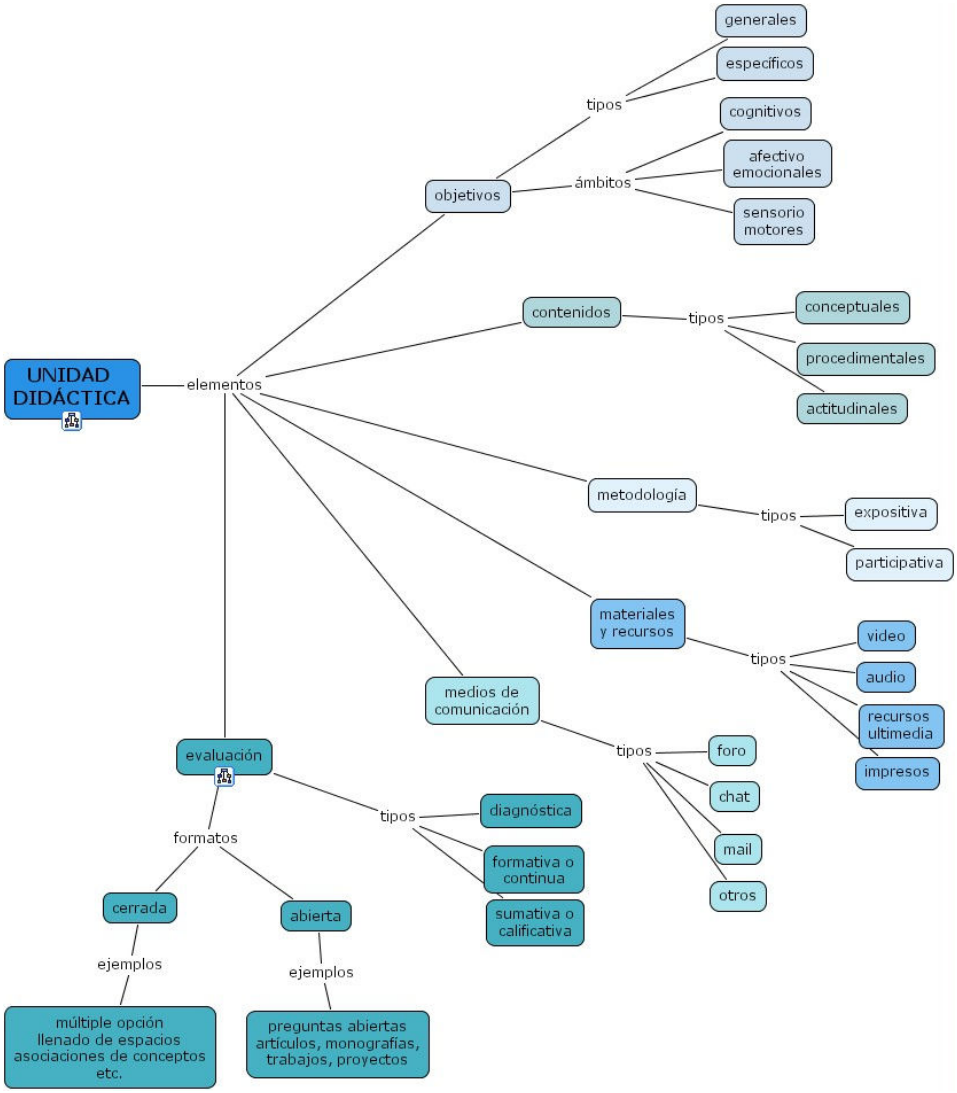


Figura 28 - Mapa Conceptual de una Unidad Didáctica

Para medir cualquiera de estos criterios es necesario usar los metadatos de lo ODAs obtenidos con el sistema ODA-Web (cantidad de definiciones, ejercicios, ejemplos),

combinados con algunos otros, como nivel de participación por actividad, nivel formativo, tipo de evaluación, etc.

Los nuevos metadatos surgen del mapa conceptual de la Figura 28. Como se explicó en la sección 2.1 -al hablar del concepto de Objeto de Aprendizaje- una Unidad Didáctica está formada por elementos de evaluación, medios de comunicación, materiales y recursos, contenido, objetivo y metodología.

Según la Figura 28, la metodología puede ser expositiva o participativa. Esto no significa que una unidad didáctica tenga que caer en una u otra categoría y que estas sean excluyentes. En la realidad una metodología tiene una tendencia más expositiva o más participativa. Podríamos decir que esta clasificación es cualitativa.

Esta característica podría medirse y un Asistente Pedagógico podría llegar a comparar cursos o determinar que para un curso se está asumiendo una metodología con más participación que exposición. El dato de nivel de participación del estudiante por actividad, multiplicándolo por la cantidad de veces que aparece esa actividad en el curso (obtenido del sistema ODA-Web), nos daría una idea del grado de participación o exposición del curso.

También con los datos obtenidos del sistema ODA-Web podríamos determinar si los contendidos son más conceptuales, procedimentales o actitudinales. Es razonable pensar que si un curso o tema tiene solo definiciones es más conceptual que procedimental.

Posiblemente, para realizar estas determinaciones, lo mejor sea comenzar acotando el problema a un tipo de carrera. No tendrán los mismos lineamientos pedagógicos un curso de Filosofía que de Ingeniería en Computación, posiblemente el primero tenga mucho más material expositivo y poco material práctico y el segundo más material participativo.

La Figura 29 muestra que las actividades pueden ser clasificadas: Por modo de organización, por función, por proceso, por tipo o por tipo de tutoría.

La primera clasificación que podría dar el asistente pedagógico, es “Por Tipo”, ya que el asistente ODA-Web reconoce tipos de actividades.

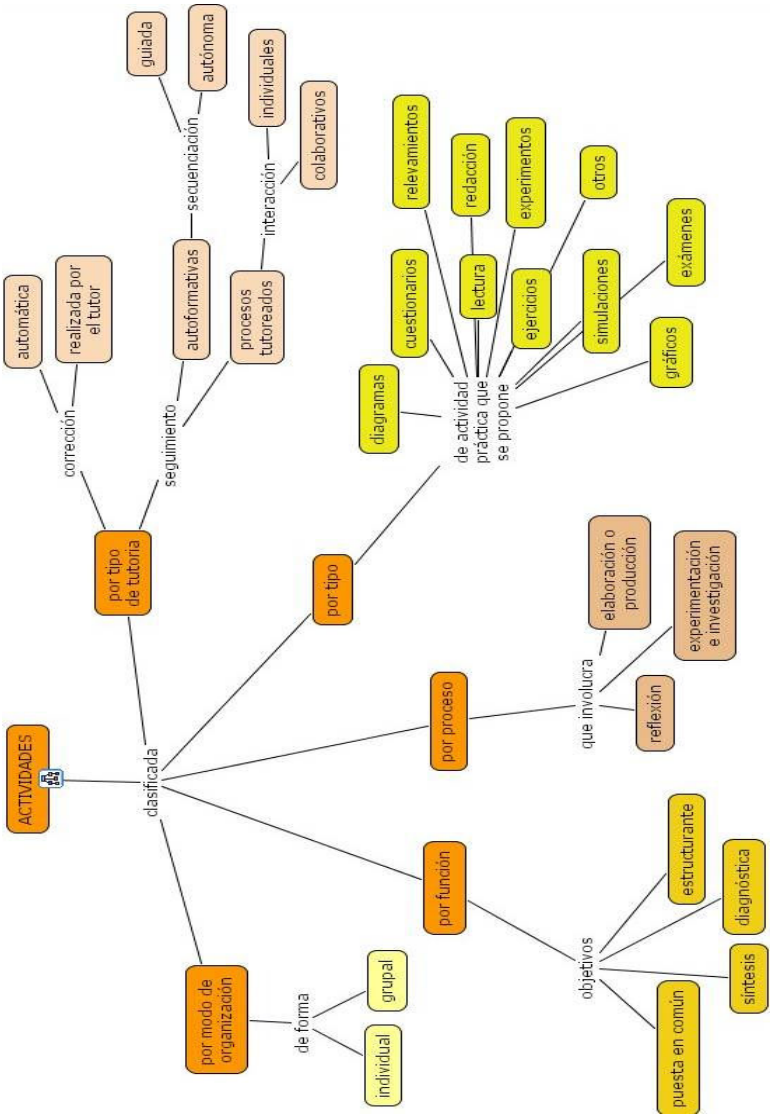


Figura 29 - Mapa conceptual de las actividades

Referencias

[1] http://www.fing.edu.uy/inco/grupos/csi/esp/Proyectos/ODA/index.html Proyecto ODA- Ultima visita: 2006-11-10

[2] "Recycling Course Web Pages for the Semantic Web". R.Motz, A. Rodríguez, R.Sosa. 4to Latin American Web Congress (LA-WEB06), Puebla, México

[3] "Adquiriendo Metadatos para Objetos de Aprendizaje", R.Motz, R.Sosa, A. Rodríguez, Primer conferencia Latinoamericana de Objetos de Aprendizaje, 23-27 octubre 2006, Guayaquil, Ecuador

[4] ATutor - www.atutor.ca - Última visita: 2006-11-10

[5] RELOAD Editor – http://www.reload.ac.uk/ - Última visita: 2006-11-10

[6] Ontología de Curso - www.fing.edu.uy/inco/grupos/csi/esp/Proyectos/OntologiasOwl/Actividad.owl Última visita: 2006-11-10

[7] LOM – Learning Object Metadata - http://ltsc.ieee.org/wg12/ Última visita: 2006-11-10

[8] UK – LOM - http://zope.cetis.ac.uk/profiles/uklomcore Última visita: 2006-11-10

[9] Proyecto EduCa - http://www.fing.edu.uy/inco/grupos/csi/esp/Proyectos/Educa.htm - última visita: 2006-11-10

[10] ARIADNE - http://www.ariadne-eu.org/ - Última visita 2006-11-10

[11] "Recuperaçao de metadados de objetos de aprendizagem no AdaptWeb." Mariusa Warpechowski. Tesis de Maestría. Universidade Federal de Rio Grande do Sul. Agosto 2005.

[12] "Towards an Adaptive Cultural e-Learning System." Jacqueline Guzmán and Regina Motz. 3er. Latin American Web Congress (LA-WEB´05), Buenos Aires

[13] SCORM – Shareable Content Object Reference Model http://www.adlnet.gov/scorm/index.cfm - Última visita: 2006-11-10

[14] ADL – Advanced Distribute Learning – http://www-adlnet.gov/ - Última visita: 10-11-2006

[15] AdaptWeb http://adaptweb.sourceforge.net/ - Última visita: 2006-11-10

[16] "Extracción e Integración de Información en una Arquitectura de Web Warehourse", Giaudrone V., Guerra M., Vaccaro M. - Tesis de grado, Facultad de Ingeniería, Universidad de la República. Diciembre 2004

[17] "Evolución de Sistemas de Web Warehousing guiado por Parámetros de Calidad", Balestra M., Vila D. - Tesis de grado, Facultad de Ingeniería, Universidad de la República. Diciembre 2005

[18] "C3bO: Captura de Características Culturales Basados en Ontologías" .B.Rabín, A.Ron, C.Vanrell. Tesis de grado, Facultad de Ingeniería, Universidad de la República. Agosto 2006.

[19] "A comparative study of metadata in SCORM and Dublin Core" J. Rouyet, V.Martín. - Primer Simposio Pluridisciplinar sobre Diseño, Evaluación y Descripción de Contenidos Educativos Reutilizables, 22-24 de octubre 2004, Guadalajara.

[20] Congreso de Sevilla – Centro virtual Cervantes (CVC) – Procesamiento de lenguaje natural - http://cvc.cervantes.es/obref/congresos/sevilla/tecnologias/ - última visita: 2006-11-10

[21] Standford University - NLP http://www-nlp.stanford.edu/links/statnlp.html Última visita: 2006-11-10

[22] Qtag POS Tagger - U.Birmigham http://www.english.bham.ac.uk/staff/omason/software/qtag.html - Última visita: 2006-11-10

[23] Jess Rules - http://www.jessrules.com/ - Última visita: 2006-11-10

[24] Tutorial XML – http://www.w3schools.com/XML última visita: 2006-10-20

[25] Tutorial XPath - http://www.w3schools.com/XPath última visita: 2006-10-20

[26] Gruber, T. R. "A Translation Approach to Portable Ontology Specifications". Knowledge Acquisition (Journal), 5(2):199220, 1993.

[27] Tidy – http://tidy.sourceforge.net/ - última visita: 2006-11-11

[28] Gruber, T. R. "Toward principles for the design of ontologies used for knowledge sharing." Padua Workshop on Formal Ontology, Italy, 1993.

[29] "Managing Source Schema Evolution in Web Warehouses", A. Marotta, R. Motz y R.Ruggia. Journal of the Brazilian Computing Society. 2001.

[30] "Desarrollo de Objetos de Aprendizaje en la Universidad Técnica particular de LOJA (UTPL)", I. Jara L, P. Sarango L. Primer conferencia Latinoamericana de Objetos de Aprendizaje, 23-27 octubre 2006, Guayaquil, Ecuador

[31] "Metodología para elaborar Objetos de Aprendizaje e integrarlos a un Sistema de Gestión de Aprendizaje", Osorio Urrutia Beatriz, Muñoz Arteaga Jaime, Álvarez Rodríguez Francisco y Carlos Arévalo Mercado. Primer conferencia Latinoamericana de Objetos de Aprendizaje, 23-27 octubre 2006, Guayaquil, Ecuador

[32] "Formato para la Determinación de la Calidad en los Objetos de Aprendizaje" R. Ruiz González, J. Muñoz Arteaga, F. Álvarez Rodríguez. Primer conferencia Latinoamericana de Objetos de Aprendizaje, 23-27 octubre 2006, Guayaquil, Ecuador

[33] HTML Parser - http://htmlparser.sourgeforce.net.- última visita: 2006-11-21

[34] "Integration Of Learning Object Repositories With Learning Management Systems", G. Parra, X. Ochoa. Primer conferencia Latinoamericana de Objetos de Aprendizaje, 23-27 octubre 2006, Guayaquil, Ecuador

[35] APROA – Aprendiendo con Objetos de Aprendizaje. – http://www.aproa.cl – Fecha última visita: 2006-11-28

[36] BALIE – http://balie.sourceforge.org – Fecha última visita: 2006-11-10

[37] Moodle – http://moodle.org – Última visita: 2006-11-10

[38] IEEE - Learning Technology Standards Committee - http://ieeeltsc.org/ – Última visita: 2006-11-10

[39] "Connecting learning objects to instructional design theory: A definition, a metaphor, and a taxonomy." D. A. Wiley (Ed.), *The Instructional Use of Learning Objects: Version Online* . http://reusability.org/read/chapters/wiley.doc - Última visita: 2006-11-10

[40] UCEL – Universities' Collaboration in eLearning - http://www.ucel.ac.uk/ - Última visita: 2006-11-10

[41] Xerces - http://xerces.apache.org/ - Última visita: 20065-11-10

[42] D.Vallespir "Actividades para Mejorar la Calidad" – tesis de maestría a publicarse. http://www.fing.edu.uy/~dvallesp/Tesis/webActividades/index2.htm - Última visita: 2006-11-10