UNIVERSIDAD DE LA REPÚBLICA
FACULTAD DE INGENIERÍA

Estudio de la Transformación de Contenido Educacional Representado Mediante LOM, hacia SCORM

Informe Final del Proyecto de Grado

Autores
Martín Cabrera, Sergio Pérez

Tutor
Prof. Dr. Regina Motz

Montevideo, Uruguay
Diciembre 2004

ÍNDICE

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

TABLA DE FIGURAS

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

1 INTRODUCCIÓN

Debido al desarrollo de la informática, la educación ha encontrado nuevas opciones que aprovechan las ventajas que brindan las computadoras y las redes de comunicación. Características como la personalización de contenido, accesibilidad remota y nuevos modos de presentación, han posibilitado que diferentes estudiantes y profesores, tomen al e-Learning como una nueva alternativa en la capacitación.

Actualmente existen cientos de plataformas de *e-Learning*, también conocidos como *Learning Management Systems* (LMS), pero estos cuentan con distintos problemas causados por la falta de utilización de estándares, la utilización de metadatos propietarios, o la imposibilidad de integración entre ellos.

A medida que ha evolucionado el *e-Learning*, distintas organizaciones, empresas de educación y empresas de software, han comenzado a trabajar en la creación de estándares y especificaciones que unifiquen criterios para integrar el contenido desarrollado en ellas. Estos estándares abordan el tratamiento de contenidos a través del uso de objetos de aprendizaje (*Learning Objects*).

Los objetos de aprendizaje constituyen nuevos elementos de un tipo de enseñanza basada en el paradigma de orientación a objetos. La idea principal detrás de ellos es que los diseñadores de contenido construyan pequeños componentes que puedan ser utilizados cualquier cantidad de veces en diferentes contextos.

Los mismos, son la base de una tecnología que posibilitará el cumplimiento de distintos objetivos claves en las organizaciones: interoperabilidad, durabilidad, accesibilidad y reusabilidad.

Para permitir la convergencia a estos estándares comunes, es necesario que los consumidores y productores de LMS, repositorios de contenido y otros sistemas relacionados con la educación a distancia, se adhieran a un estándar común. Para ello, en muchos casos es necesario implementar nuevamente las plataformas basados en el estándar elegido. Dado que esto generalmente no es una opción viable, es necesario visualizar otro tipo de alternativas.

Este proyecto propone transformar el contenido educacional utilizado o generado por este tipo de plataformas a un estándar común. Si bien, en este trabajo no analizamos todas las posibles transformaciones, se realiza un estudio sobre estándares muy utilizados hoy en día, como son: LOM y SCORM, analizando los problemas en la transformación, así como sus posibles soluciones.

La utilización de este tipo de transformaciones hacia un estándar común de contenido educacional permite que el trabajo realizado en las herramientas existentes, así como el contenido ya generado, no queden rápidamente obsoletos, protegiendo de esta manera las inversiones que han sido realizadas por las distintas organizaciones.

1.1 Objetivos Planteados

El primer objetivo planteado, es el de estudiar la viabilidad de transformar contenido existente en LOM a SCORM, analizando posibles soluciones a los distintos problemas detectados y haciendo hincapié en los puntos críticos de decisión.

Debido a la versatilidad con la que se quiso dotar a la especificación de LOM, esta determina una descripción de los objetos de aprendizaje en forma muy genérica, definiendo sus distintos atributos en forma abierta y sin dar una semántica específica en cada uno de ellos. Es así que surgen los perfiles de aplicación y el trabajo con ontologías que permiten incorporar a la definición de los elementos de datos de LOM, contenido semántico especifico a cada realidad organizacional.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

El proyecto AdaptWeb-Multicultural, que sirve de marco para este proyecto, cuenta con una especialización de LOM que agrega contenido semántico a través de una ontología. Por lo tanto, el segundo objetivo planteado es el de desarrollar una transformación específica a partir de contenido educacional desarrollado en AdaptWeb a SCORM.

Finalmente, para analizar la factibilidad de las transformaciones planteadas y en particular, la transformación específica, es que se plantea la implementación de un prototipo con el cual, a partir de contenido educacional definido en AdaptWeb, se construya contenido en SCORM. Incorporando en este, la noción de repositorio, permitiendo así la generación de nuevos contenidos a partir de objetos ya transformados.

1.2 Lectura del Informe

En este informe, inicialmente introduciremos los conceptos básicos que se manejan a lo largo del mismo, como pueden ser los estándares analizados y otros conceptos importantes. Posteriormente se detalla el análisis realizado en las transformaciones propuestas, partiendo de la Transformación General a la Específica. Tomando como base esta última transformación, se plantea la realización de un prototipo y se describe la solución propuesta.
Por último se brindan las conclusiones del trabajo, presentando el aporte realizado, así como las limitaciones y trabajos futuros.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

2 CONCEPTOS BÁSICOS

Al investigar las tecnologías de información y la comunicación aplicada a la educación a distancia, se encuentran varios problemas aún sin resolver, destacándose la falta de una metodología común que garantice los siguientes objetivos:

- **Durabilidad**: Que la tecnología desarrollada con el estándar evite la obsolescencia de los cursos.
- **Interoperabilidad**: Que se pueda intercambiar información a través de una amplia variedad de LMS (*Learning Management System*), sin importar el tipo de hardware, los sistemas operativos, o los navegadores web.
- **Accesibilidad**: Que los cursos y objetos de aprendizaje puedan ser fácilmente encontrados o publicados ya sea por los diseñadores de contenido o por los estudiantes.
- **Reusabilidad**: Que los distintos cursos y objetos de aprendizaje puedan ser fácilmente utilizados y modificados con diferentes herramientas de desarrollo y en distintas plataformas.

Los estándares son el vehículo a través del cual será posible asegurar el cumplimiento de estos objetivos en las soluciones e-*Learning*.

A continuación se presentan las principales líneas de trabajo en *e-Learning*.

Metadatos de contenido: Los metadatos son básicamente, información sobre la información. Esta permite poder clasificar, ubicar y reutilizar el contenido de aprendizaje que se va generando. Su información es utilizada ya sea por un estudiante que necesita acceder a un material especifico, como por un diseñador de contenidos que pretende reutilizar el material descrito con los metadatos en su propio curso.

Empaquetamiento de contenidos: Una vez que el contenido de aprendizaje es diseñado y construido, es necesario ponerlo a disposición de los estudiantes, repositorios de contenidos o en los mismos LMS. El propósito principal es poder intercambiar el contenido de aprendizaje, y para esto es necesario contar con especificaciones que definan como se deben organizar estos paquetes de una forma estándar para que el intercambio no se vea impedido por las barreras que pueden representar los distintos sistemas o herramientas.

Secuenciamiento de contenidos: El secuenciamiento es la forma en que se especifican los posibles caminos en que puede recorrerse el contenido de aprendizaje. Estos pueden ser definidos en tiempo de diseño y prever distintos comportamientos a lo largo de la navegación del contenido en base a distintos eventos. Los eventos pueden ser iniciados, ya sea por el estudiante, o por el mismo sistema.

Adaptabilidad de los contenidos: Al contrario que en los cursos presentados solamente en páginas HTML, como si fuera una simple trascripción de un libro, y donde el contenido que se le presenta a todos los estudiantes es el mismo, los Sistemas de Hipermedios Adaptativos (ó AHS - *Adaptive Hypermedia Systems*) adaptan el contenido y el estilo de presentación del material, basándose en el perfil del estudiante.

Perfil de los estudiantes: el mantener un perfil del estudiante, permite adaptar para cada estudiante, la forma de aprender que mejor se ajuste a sus necesidades, costumbres, herramientas, capacidades y conocimientos previos.

2.1 Especificaciones y Estándares

Distintas organizaciones, empresas de educación y empresas de software, trabajan en la creación de estándares y especificaciones. Con estos, se pretende brindar las herramientas necesarias para que se puedan crear plataformas que utilicen recursos interoperables entre distintos LMS.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

Inicialmente se emprendieron esfuerzos paralelos en pro de la estandarización produciendo distintas opciones que en algunos casos se sobreponían y que debido a su rápida expansión en el mercado agravaron la situación. Sin embargo, en la actualidad esta competencia ha sido dejada a un lado, y se aunaron esfuerzos en pos de generar estándares comunes entre las distintas organizaciones. Un ejemplo de ello es SCORM, en el cual se utilizaron los trabajos de varias organizaciones en un solo modelo.

Dentro de las principales iniciativas de estándar para *e-Learning* se puede mencionar:

AICC, Aviation Industry CBT Committee
La industria de la aviación en 1992 decidió crear un comité que desarrollase una normativa para sus proveedores de cursos a distancia. De este modo garantizaban la armonización de los requerimientos de los cursos, así como la homogeneización de los resultados obtenidos de los mismos. Este organismo fue el pionero en tratar estos temas en torno a problemas de contenidos educacionales. [AICC]

IEEE Learning Technologies Standards Committee (LTSC)
La IEEE es un grupo multinacional que crea estándares internacionales para sistemas eléctricos, electrónicos, computacionales y de comunicación. Está organizado en distintos comités, siendo uno de estos el comité para los Estándares de Tecnología de Aprendizaje o *Learning Technology Standars Committee (LTSC).* Una de sus especificaciones más difundida y utilizada se refiere a los Metadatos de Objetos de Aprendizaje o *Learning Object Metadata (LOM).* Con este modelo, LTSC intenta facilitar la reutilización e integración de contenidos. Este organismo recolectó los distintos trabajos de la AICC, intentando mejorarlos.[LOM]

LOM se explica en más detalle en la sección 2.2 (LOM – Learning Object Metadata).

IMS Global Learning Consortium Inc
Este Consorcio está formado por miembros provenientes de organizaciones educacionales, empresas públicas y privadas. Su misión es desarrollar y promover especificaciones abiertas para facilitar las actividades del aprendizaje online.
El trabajo de la IEEE fue recogido por esta corporación privada creada por algunas de las empresas más importantes del sector. Su objetivo fue la creación de un formato que pusiese en práctica las recomendaciones de la IEEE y la AICC. [IMS]

ADL SCORM
SCORM *(Shareable Content Object Reference Model)*, es un estándar generado por la ADL *(Advanced Distributed Learning)*. Este organismo fue fundado en 1997, como un programa del Departamento de Defensa de los Estados Unidos y de la Oficina de Ciencia y Tecnología de la Casa Blanca para desarrollar principios y guías de trabajo necesarias para el desarrollo y la implementación eficiente, efectiva y en gran escala, de formación educativa sobre nuevas tecnologías Web. [ADLSCORM]

Este estándar se explica en más detalle en la sección 2.3 (SCORM).

2.2 LOM – Learning Object Metadata

LOM [LOM] es un estándar que especifica los metadatos de un *Learning Object* (L.O.). Para este estándar, un L.O. es definido como una entidad (digital, o no digital) que puede ser utilizada para educación o entrenamiento.

Lo que se presenta en este estándar, es un esquema de datos conceptual. Este define la estructura de la instancia de los metadatos de un L.O., permite diversidad lingüística a nivel del mismo L.O. y de las instancias de los metadatos y a su vez, especifica cuales son los elementos de datos que componen esta instancia. El estándar está hecho con la intención de que sea referenciado por otros estándares que definan la descripción de la implementación del esquema de datos conceptual, para que de esta forma una instancia de metadatos de un L.O. pueda ser utilizada por sistemas de tecnología de aprendizaje para localizar, evaluar e intercambiar L.O.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

2.2.1 Estructura General

A continuación se brinda una breve descripción de la estructura especificada en LOMv1.0 para los metadatos que describen un L.O.

Categorías

En LOMv1.0, un L.O. está descrito únicamente por sus Elementos de Datos. Estos son agrupados en las siguientes nueve categorías:

1. **General:** agrupa la información general que describe el L.O. como un "todo"
2. **Lifecycle:** abarca información relacionada con la historia y el estado actual del L.O.
3. **Meta-metadata:** contiene información acerca de la instancia de los metadatos utilizada para describir el L.O.
4. **Technical:** agrupa los requerimientos y características técnicas del L.O.
5. **Educational:** agrupa las características educacionales y pedagógicas del L.O.
6. **Rights:** agrupa los derechos de propiedad intelectual y las condiciones de uso.
7. **Relation:** define las relaciones entre el L.O. descrito y otros L.O.
8. **Annotation:** provee comentarios del uso educacional.
9. **Classification:** describe el L.O. en relación a un sistema de clasificación particular. Esta clasificación permite proveerle extensiones a LOMv1.0.

2.2.2 Esquema Base

A continuación se presenta un diagrama neuronal que muestra la estructura completa definida en el estándar LOMv1.0. [FDLOM]

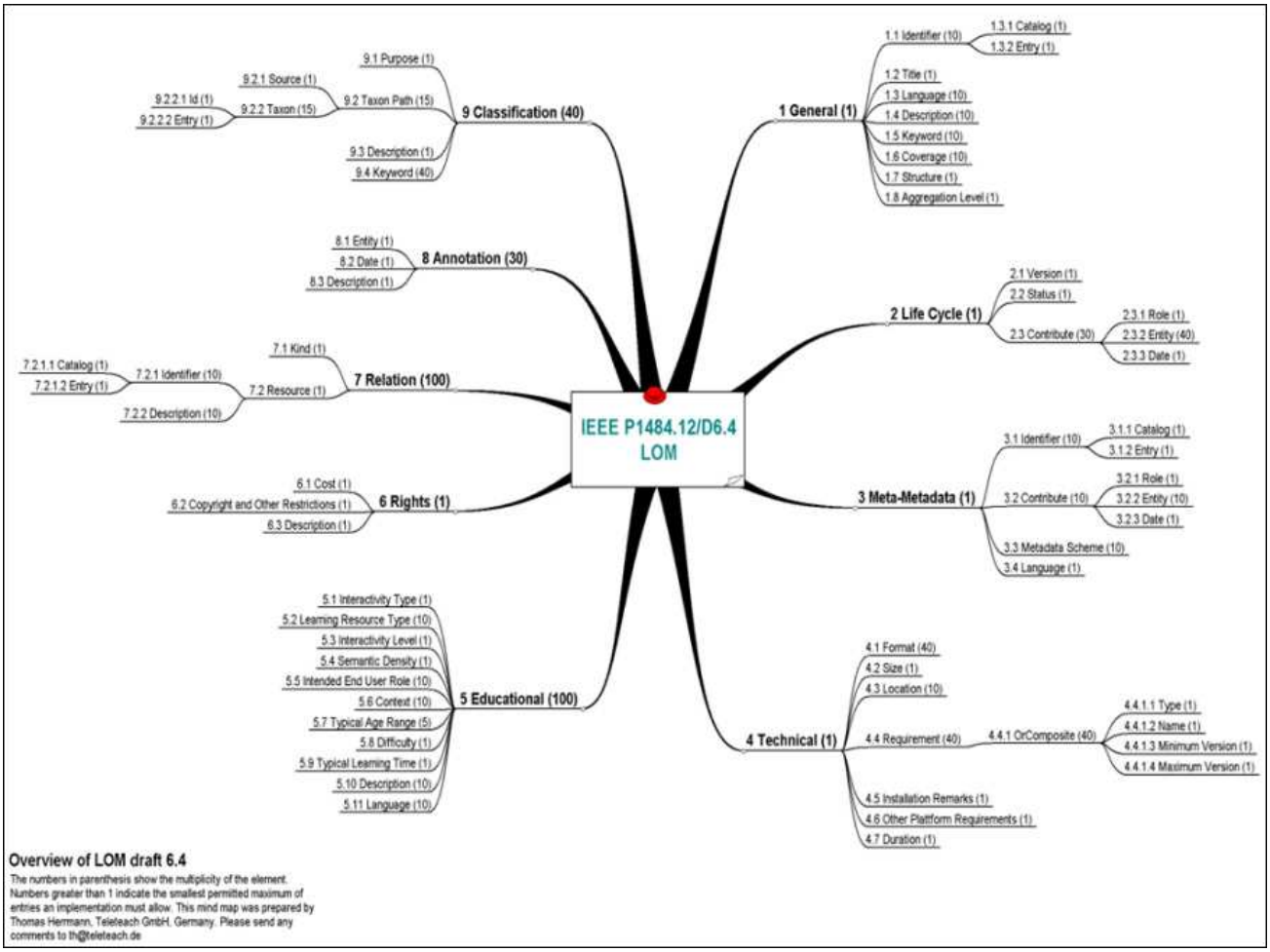


Figura 1: Estructura de LOM – (Thomas Hermann, Teleteach GmtH, Alemania)

2.2.3 Elementos de Datos

El modelo de datos de LOM es una jerarquía de elementos de datos que incluyen Elementos de Datos agregados y Elementos de Datos simples (las hojas en la jerarquía). Sólo estos nodos hoja son los que poseen valor individual definido a través de su espacio de valores asociado y su tipo de datos (los agregados no poseen valores individuales).

Para cada Elemento de Datos el esquema base de LOMv1.0 define:

- *name*: el nombre por el cual el elemento de datos es referenciado.
- *explanation*: la definición del elemento de datos
- *size*: el número de valores permitidos
- *order*: especifica si el orden de los valores es significativo (aplicable solamente para elementos de datos con lista de valores).
- *example*: un ejemplo ilustrativo.

Para los elementos de datos simples, LOMv1.0 también define:

- *value space*: el conjunto de valores permitidos para el elemento de datos (típicamente definido mediante un vocabulario o con una referencia a otro estándar).
- *Datatype*: indica si los valores son *LangString*, *DateTime*, *Duration*, *Vocabulary*, *CharacterString* o *Undefined*.

En ambos casos, se puede incluir un valor SPM (*smallest permitted maximum*, o menor máximo permitido), que indica que las aplicaciones que procesan los metadatos, si van a imponer un máximo de elementos de datos a procesar, deben imponer un número que sea mayor o igual que el número SPM especificado.

Todos los elementos de datos son opcionales

Esto implica que una instancia *LOM conforme* puede incluir valores de cualquiera de los Elementos de Datos que están presentes en la definición del estándar. De todas formas, como en el esquema de base se impone una relación de agregación, por definición, un elemento solamente puede estar presente en una instancia LOM como un componente de un elemento agregado que lo contiene.

Lista de valores

Algunos elementos de datos contienen lista de valores en lugar de un valor único. En este caso la lista puede ser de dos tipos: Ordenada o Sin Orden.

2.2.4 Vocabularios

Para algunos elementos se define un vocabulario. En el estándar se recomiendan vocabularios específicos para estos elementos, pero se pueden utilizar valores no presentes en él. Si se utilizan instancias basadas solamente en los valores recomendados, tendrán mayor nivel de interoperabilidad semántica que si se utilizan valores particulares.

Para el análisis de la Transformación Genérica propuesta, se tienen en cuenta los vocabularios definidos en el estándar, ya que sería imposible plantear transformaciones basadas en vocabularios desconocidos. Los valores de estos elementos de datos, en algunos casos, podrán servir para tomar decisiones en el pasaje de LOM a SCORM, pero para definir estas posibles decisiones se tiene que trabajar con valores conocidos.

2.2.5 Conformidad

- **Una instancia LOM estrictamente conforme** debe consistir únicamente de Elementos de Datos LOM.

- **Una instancia LOM conforme** puede contener Elementos de Datos extendidos.

- Una instancia LOM que no contiene valores para ninguno de los Elementos de Datos LOM es una instancia conforme.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

En pos de maximizar la interoperabilidad semántica, los Elementos de Datos que son extendidos no deben remplazar los Elementos de Datos ya existentes en la estructura LOM.

2.2.6 Perfiles de Aplicación (Application Profiles)

Dado que la estructura de LOM fue hecha en forma muy genérica como para poder abarcar la mayor cantidad posible de realidades, generalmente cuando se va a aplicar a un contexto específico, resulta muy general en sus vocabularios y con más Elementos de Datos que los necesarios.

Una alternativa para adaptar la especificación de LOM, es la utilización de perfiles de aplicación (ó *Application Profiles*). Estos permiten incrementar la interoperabilidad semántica dentro de una comunidad, preservando a su vez, la compatibilidad hacia fuera de la comunidad.

Para definir estos perfiles de aplicación se pueden realizar distintas tareas:

- especificar Elementos de Datos como obligatorios
- restringir los espacios de valores de los Elementos de Datos
- imponer relacionamientos entre Elementos de Datos
- excluir algunos Elementos de Datos
- identificar taxonomías y clasificaciones

Un ejemplo de esto puede ser el *Application Profile* definido en el proyecto AdaptWeb en el cual se detallan claramente cada uno de los puntos antes mencionados (ver anexo: Perfil de aplicación de AdaptWeb)

Los perfiles de aplicación pueden brindar un importante valor agregado a la comunidad para la cual fueron desarrollados. La mayor especificidad de los elementos de datos, permite disminuir ambigüedad de criterios y simplifica el desarrollo de los contenidos dentro de la comunidad.

Como se ha mencionado anteriormente, en la especificación de LOM cualquiera de sus Elementos de Datos son opcionales. Pero a su vez, esta estructura se puede extender con nuevos Elementos de Datos que pueden agregarse en cualquier parte de la jerarquía de LOM. Estas extensiones a LOM, pueden brindar mayor potencial dentro de la comunidad, pero hacen que las instancias pierdan generalidad, lo que conlleva a una menor interoperabilidad hacia fuera de la comunidad.

2.3 SCORM

SCORM (*Shareable Content Object Reference Model*) [ADLSCORM], es un estándar generado por la ADL (Advanced Distributed Learning). Este organismo fue fundado en 1997, como un programa del Departamento de Defensa de los Estados Unidos y de la Oficina de Ciencia y Tecnología de la Casa Blanca para desarrollar principios y guías de trabajo necesarias para el desarrollo y la implementación eficiente, efectiva y en gran escala, de formación educativa sobre nuevas tecnologías Web.

SCORM intenta reunir "las mejores" iniciativas de especificaciones y estándares en torno al E-Learning, tanto desde el punto de vista de descripción de los Metadatos como especificaciones sobre como manejar esta información para permitirle a los sistemas interoperar, reutilizar y obtener accesibilidad sobre contenido de aprendizaje en un entorno Web. Proporciona también un *framework* y una referencia de implementación detallada que permite a los sistemas cumplir con este objetivo.

Existen varias versiones de SCORM, debido a que este está definido sobre un conjunto de especificaciones de otros organismos, y estos últimos evolucionan constantemente, produciendo actualizaciones en torno al *framework* descrito por ADL.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

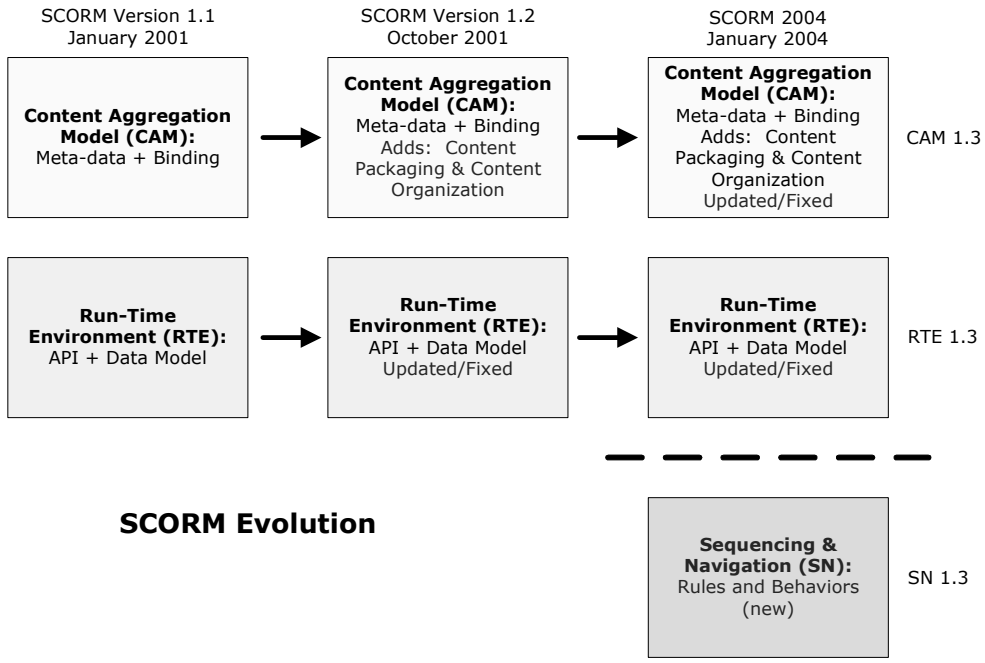


Figura 2: Evolución de SCORM (extraído de "SCORM_2004_Overview" [ADLOV])

Los estándares y especificaciones en SCORM 2004 son:

- IEEE Data Model For Content Object Communication
- IEEE ECMAScript Application Programming Interface for Content to Runtime Services Communication
- IEEE Learning Object Metadata (LOM)
- IEEE Extensible Markup Language (XML) Schema Binding for Learning Object Metadata Data Model
- IMS Content Packaging
- IMS Simple Sequencing

Existen tres módulos bien diferenciados en el *framework* que propone ADL, sobre cada uno de ellos, ADL proporciona un "libro" que lo detalla, estos son: *Content Aggregation Model* (CAM)[ADLCAM]*, Run-Time Environment* (RTE)[ADLRTE] y *Sequencing and Navigation* (SN)[ADLSN].

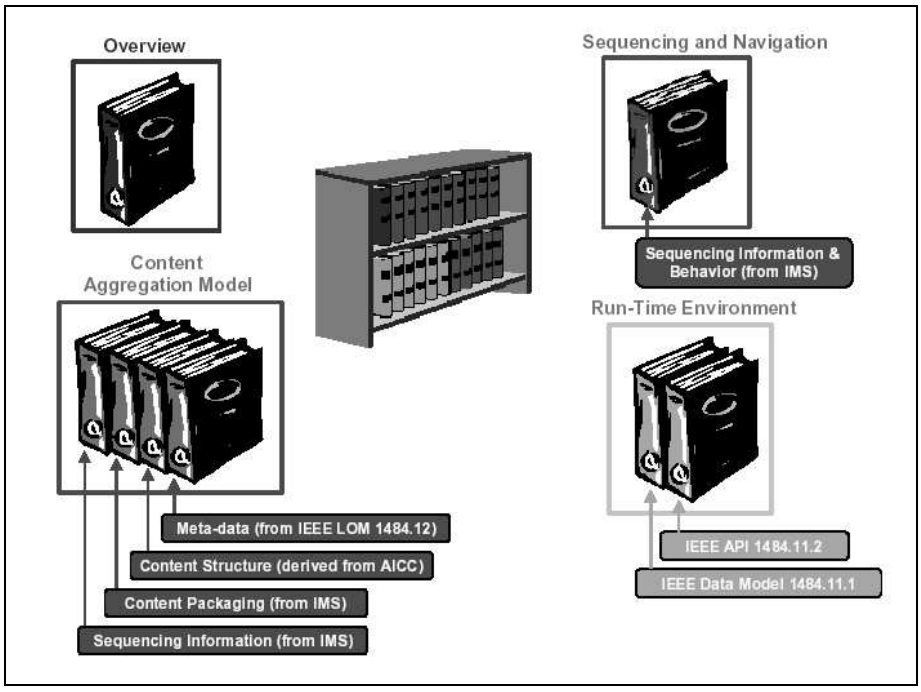


Figura 3: Libros de SCORM (extraído de "SCORM_2004_Overview [ADLOV])

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

A continuación se describe cada uno de los módulos presentados, prestando especial interés sobre la sección *Content Aggregation Model* (CAM) dado que en la misma, se especifican las estructuras de datos que deben ser utilizados al transformar contenido LOM a SCORM.

2.3.1 SCORM Content Aggregation Model (CAM)

SCORM *Content Aggregation Model* (CAM), describe los componentes utilizados para representar los contenidos educacionales. Estos especifican como se empaquetará la información para interoperarla entre los distintos sistemas, también describen como los componentes son buscados y como se define las reglas de secuenciamiento para ellos.
En el libro de SCORM CAM [ADLCAM] solo se define responsabilidades y requerimientos para construir contenido agregacional (ejemplo cursos, lecciones, módulos, etc.). Indicando como crear paquetes de contenido, especificando cada componente por medio de metadatos en formato LOM. Otro punto importante que detalla el libro son las reglas de secuenciamiento y navegación. Se mantienen varias dependencias con otros libros, en particular con SCORM RTE [ADLRTE].

A continuación se describen los cuatro grandes conceptos que define SCORM *Content Aggregation Model*, detallando cada uno de estos en secciones posteriores. Estos conceptos son:

- ***Content Model:*** Nomenclatura que define el contenido de un componente de aprendizaje.

- ***Content Packaging:*** Los componentes antes descritos, se reúnen en una única definición de datos, definiendo a esta como paquete de contenido (ó *Content Package*). Esta describe como los objetos que definen un contenido son vinculados y relacionados entre si. La forma que tiene de realizarlo es por medio de la definición de contenido organizacional descrito en un documento XML, llamado *Manifest*. Este contenido organizacional, representa un curso, lección, o cualquier otra colección de contenidos. El XML que describe el paquete de contenido es especificado por IMS, a través de su especificación de *Content Packaging*.

- **Meta-data**: Mecanismo para describir las especificaciones de las distintas instancias del modelo de contenido. Estos son: *Content Aggregations*, *Activities*, *SCOs* y *Assets*. Cada uno con un rol bien definido dentro de la definición de datos presentada por SCORM.

- ***Sequencing and Navigation***: Reglas que definen el secuenciamiento y ordenación de Actividades. Las Actividades pueden ser referenciadas o no a un recurso de aprendizaje, para poder vincularlo directamente con un elemento que el consumidor de contenido (estudiante) pueda visualizar.

Componentes del *SCORM Content Model*
SCORM Content Model describe como los componentes son construidos por medio de recursos educacionales. El modelo de contenido define los componentes a un nivel de compartimiento muy pequeño y es necesario incluir elementos como el *paquete de contenido* para hacer que estos puedan interoperar completamente entre los sistemas.

Los distintos contenidos que se pueden definir en *SCORM Content Model* son:
- *Assets*
- *Sharable Content Objects (SCOs)*
- *Content Organization.*

Assets
Esta es la forma más básica para describir un recurso de aprendizaje. Un *Asset* es una representación electrónica de un texto, imagen, sonido, multimedia, y cualquier otro objeto simple que será presentado a un cliente Web. A su vez, un *Asset* puede estar formado por un conjunto de *Assets*.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

Un *Asset* es descrito por medio de sus metadatos (*Asset Metadata*). De esta manera puede ser buscado en distintos repositorios para ser reutilizado por distintos cursos o unidades educacionales. La manera de asociar esta información (*Asset – Asset Metadata*) es por medio del *Content Package* que se detalla posteriormente.

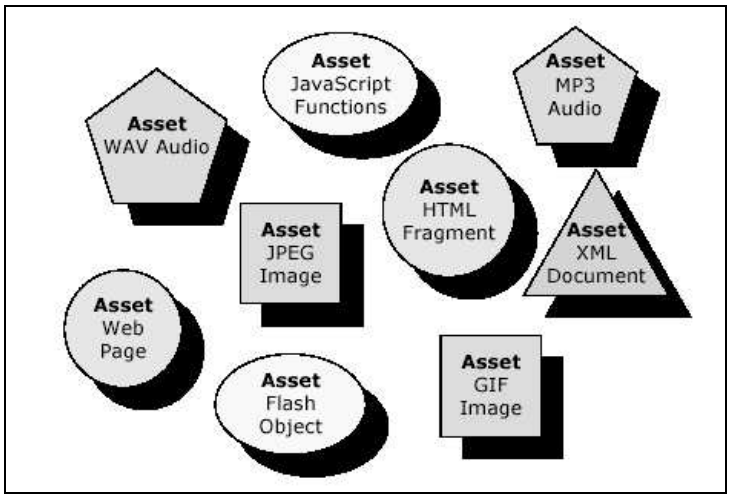


Figura 4: Ejemplos de *Assets* (extraído de "SCORM_CAM" [ADLCAM])

Sharable Content Aggregation (SCA)
Este tipo de objeto de SCORM es una colección de Assets que no poseen la capacidad de ser referenciados dentro de los LMS (como sí la tienen los SCO).

Sharable Content Object (SCO)
Un *SCO* es una colección de uno o más *Assets*, representando estos últimos a un recurso educacional simple. Este utiliza SCORM *Run-Time Environment Data Model* para comunicarse con un LMS. Un *SCO* es granularmente, el menor recurso educacional que puede comunicarse con un LMS. La única diferencia entre un *Asset* y un *SCO* es que este último puede comunicarse con un LMS.

Para mejorar la reusabilidad, un *SCO* puede ser independiente de un contexto educacional. Por ejemplo, un *SCO* puede ser reutilizado en diferentes ambientes, representando en cada uno un objeto de contenido semántico distinto. Por esto se introduce el concepto de Actividad (ver *Content Organization*) para formar contenidos con mas de un *SCO / Assets* de manera de definir un mayor nivel de granularidad en las unidades instruccionales.

Los *SCOs* intentan ser unidades pequeñas, de manera de potenciar su reuso en múltiples ambientes educacionales, extendiendo y agregando relaciones (de ser necesario) por medio del *Content Organization*.
El *SCO* se describe por medio de metadatos (*SCO Metadata*) al igual que los *Assets* y de la misma forma, esta descripción puede ser utilizada para realizar búsquedas sobre repositorios. El modo de asociar esta información es por medio del *Content Package*.

Content Organization
Un *Content Organization* es un mapa que representa las relaciones entre las distintas unidades instruccionales (Actividades). Este mapa representa un árbol de relaciones, conteniendo restricciones que se detallan a continuación. Indica como una Actividad se relaciona con otra.

A continuación se presenta la estructura clásica de un *Content Organization.*

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

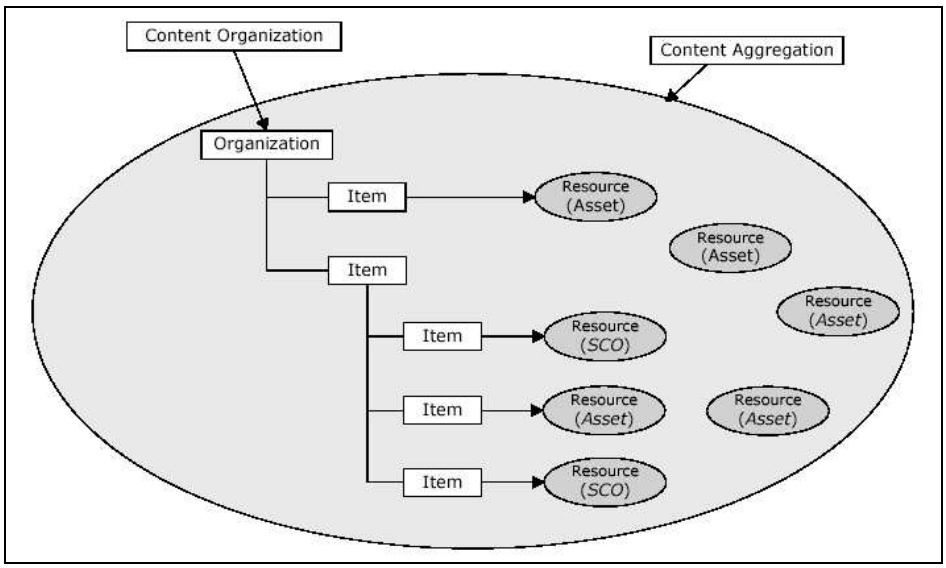


Figura 5: Content Organization y Content Aggregation (extraído de "SCORM_CAM" [ADLCAM])

Las Actividades representadas en un *Content Organization* pueden estar constituidas, tanto por un conjunto de Actividades (sub-Actividades) o directamente (si es una hoja) por un recurso. Si la Actividad es una hoja obligatoriamente debe tener una relación con un recurso y este debe ser obligatoriamente un *SCO* o un *Asset*.

Sobre los niveles que se pueden definir en el árbol organizacional, no existen límites en cuanto a la cantidad de ellos, o sea que una Actividad puede contener sub-Actividades y a su vez estas contener otras sucesivamente, sin un número que acote la cantidad de niveles definidos. Estos niveles pueden tener asociadas distintos niveles de jerarquías, por ejemplo, se puede definir un nivel como un curso, capitulo, módulo, etc.

Cuando una Actividad esta compuesta por varias Actividades, decimos que esta es un *Cluster.* Sobre la estructura de relaciones que tiene el árbol definido en el *Content Organization* vemos que se puede incorporar la idea de secuenciamiento, definiéndolo solamente para las Actividades. Un LMS tiene la obligación de interpretar esta información y aplicar el secuenciamiento correspondiente sobre cada recurso que tenga asociado las Actividades. En el "libro" de SCORM SN [ADLSN] se describe completamente el comportamiento de las Actividades y los *Clusters*

Al igual que los casos anteriores (*SCO* y *Asset*) los metadatos del *Content Organization (Content Organization Metadata)*, describen características sobre este, de manera que pueda ser buscado en los repositorios y así ser reutilizado adecuadamente. Nuevamente la asociación entre un *Content Organization* y sus metadatos se describe en el *Content Package.*

Metadatos de SCORM

Como se ha especificado anteriormente, sobre cada uno de los contenidos (*Assets, SCOs y Content Organization*) se pueden definir metadatos asociados para poder realizar búsquedas en repositorios y aplicar con mayor facilidad el reuso de componentes.
Para esto como parte del modelo, SCORM propone un Perfil de Aplicación sobre sus metadatos (ver anexo: Perfil de Aplicación de los Metadatos SCORM). Este representa relaciones y usos sobre el estándar LOM de la IEEE-LTSC para cada componente del *SCORM Content Model*. En general se provee una guía para aplicar este estándar a *Assets*, *SCOs*, *Actividades* y *Content Organizations*. Es una forma de categorizar cada uno de estos componentes, para buscarlos, facilitar el reuso y la propiedad de compartir estos elementos.

Los metadatos definidos para los distintos componentes son:

- ***Content Aggregation Metadata:*** describe la información de un *Content Aggregation* como un todo.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

- ***Content Organization Metadata:*** describe la información sobre un la estructura definida en un *Content Organization*.
- ***Activity Metadata:*** describe una actividad individual
- ***SCO Metadata:*** describe la información del *SCO*, independientemente del uso que se le quiera dar.
- ***Asset Metadata*:** describe la información de un *Asset*, independientemente del uso que se le quiera dar.
- ***Application of Metadata:*** este es un mecanismo para agrupar o vincular los distintos componentes definidos anteriormente. A estos metadatos se le llaman *Content Package* y definen los siguiente cinco elementos:
    - **Manifest:** Metadatos que definen el mayor nivel de abstracción (por ejemplo: *Content Aggregation Metadata*). Se puede crear por medio de LOM, agregándole varias restricciones de SCORM.
    - **Organization**: Metadatos que definen el *Content Organization* como un todo. En este caso se puede indicar el contenido semántica que se quiere dar a la organización descripta (por ejemplo: curso, lección, etc.).
    - **Item:** Metadatos que describen las actividades. Se debe asociar a esta con una definición de *Activity Metadata*
    - **Resource:** Metadatos que describen un recurso, puede ser tanto un *SCO* como un *Asset*. Esta definición debe ser vinculada con otra de tipo *SCO Metadata* o *Asset Meta-data*.
    - **File:** Metadatos que describen un *Asset* en un contexto básico. Estos metadatos deben ser vinculados con una definición de SCORM *Asset Metadata*

Content Packaging

El propósito del *Content Packaging* es proveer una forma estandarizada para publicar un contenido educacional de forma que los distintos sistemas involucrados puedan manejarlo adecuadamente.
SCORM Content Packaging es un conjunto de especificaciones y guías, que utilizan la especificación de *IMS Content Packaging.*

A continuación se describen los elementos mas importantes definidos aquí, estos son:

- ***Content Package Components:*** define los principales conceptos de un *Content Package*. Estos conceptos son utilizados para entender el *Content Package* antes de describir la especificación completa.
- ***Components of a Manifest:*** describe los componentes del *Content Package manifest*.
- *Building Content Packages* define el proceso de construcción de un paquete de contenido. Esta sección describe los componentes XML del *manifest* y los requerimientos utilizados por estos componentes.
- ***SCORM Content Package Application Profiles****:* define específicamente como crear un paquete *SCORM-conformant* que contenga *Assets, SCOs y Content Organizations.*
- ***Best Practices and Practical Guidelines:*** Define una colección de buenas prácticas y guías para construir un *Content Package*.

Content Package Components

El *IMS Content Package* contiene dos componentes:

- Un documento XML que describe el contenido estructural y los recursos del paquete, llamado *manifest* (IMS exige que el nombre de este documento sea obligatoriamente: imsmanifest.xml).
- Archivos físicos que se hacen referencia en el *Content Package*.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

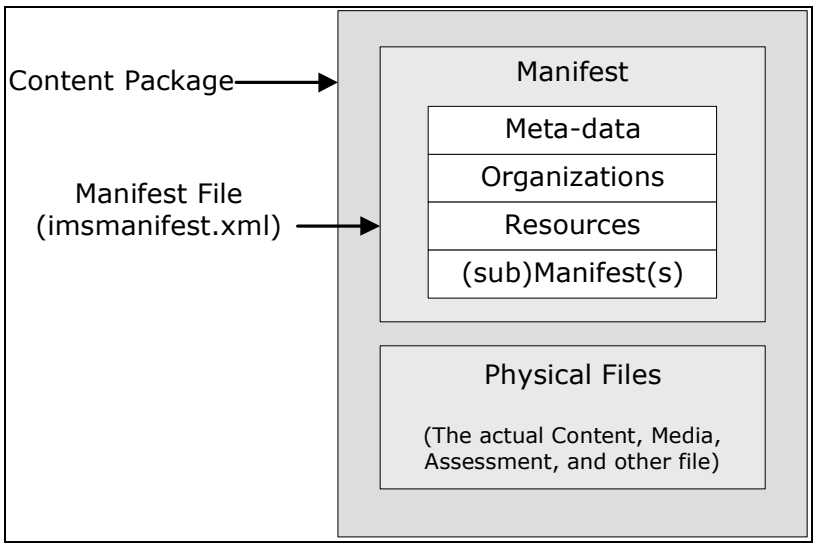


Figura 6: Estructura del *Content Package* (extraído de "SCORM_CAM" [ADLCAM])

El archivo *manifest* representa la información necesaria para describir los contenidos del paquete. Este está compuesto por cuatro secciones:

- Meta-data: describe el contenido del paquete como un todo.
- Organizations: contiene la estructura organizacional de los recursos educacionales.
- Resources: define los recursos educacionales que son utilizados en el *Content Package.*
- (sub)Manifest(s): describe las unidades anidadas, o sea que se permite anidar tantos *manifest* como se crea necesario. En las mejores prácticas y guías se hace especial hincapié en tratar de definir un solo *Manifest*, sin anidaciones.

2.3.2 The SCORM Run-Time Environment (RTE)

El SCORM RTE, describe los requerimientos de un LMS para manejar el *run-time environment*. Un ejemplo puede ser el proceso de lanzamiento de contenido, o la comunicación entre el LMS y los elementos del modelo de datos sobre la información de aprendizaje.
El RTE maneja los requerimientos de los *SCOs*, su uso de la *API* y el *SCORM Run-Time Environment Data Model*. El propósito de SCORM RTE es proveer una forma de interoperar los *SCOs* con los LMSs. Para hacer esto posible, este ofrece una manera común de: lanzar contenido, realizar la comunicación entre distintos LMS y definir elementos de datos que serán compartidos entre el LMS y el contenido en tiempo de ejecución.
Existen tres componentes descritos para cumplir con los objetivos planteados y cada uno de ellos son definidos completamente en los "libros" que provee ADL. Estos componentes de SCORM RTE son:

- Definición de lanzamiento
- API (*Application Program Interface*)
- Modelo de datos

Lanzamiento, define la relación entre los LMS y el contenido descrito en SCORM (a este contenido se le llama SCORM-conformant). De esta manera se puede publicar y desplegar la información descripta en los contenidos para que el estudiante o la persona que consuma servicios del LMS visualice sus cursos, lecciones, etc. Parte del problema agregado en la nueva versión de SCORM (SCORM 2004) es la de controlar la navegación por parte del usuario en el contenido, este problema es analizado con mas detalle en el "libro" SCORM SN [ADLSN].

SCORM API, describe un conjunto de funcionalidades que deben ser implementadas por el proveedor del LMS para que se puedan lanzar y publicar los *SCOs*.
Estas funcionalidades, establecen y manejan el comportamiento del LMS y los *SCOs*, manejando los errores (en caso de producirse) en tiempo de ejecución.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

Estas funcionalidades completan el proceso de lanzamiento poniendo de acuerdo al LMS y al *SCO* para que se puedan publicar los distintos contenidos descriptos.

Data Model provee un vocabulario que puede utilizarse para pasar información desde y hacia un LMS utilizando las funciones de la SCORM API. Por ejemplo, cuando se envía un examen a un estudiante, el SCO puede utilizar un elemento del SCORM Data Model en el cual se informa al LMS cual fue el resultado del estudiante.

2.3.3 SCORM Sequencing and Navigation (SN)

SCORM SN describe como un contenido *SCORM-Conformant* puede ser secuenciado e inicializado. Describe como un LMS debe interpretar las reglas de secuenciamiento expresadas por el desarrollador de contenido para inicializar eventos, páginas y otros requerimientos que se piden en el *run-time environment*.

El diseñador de contenido debe agregar sobre cada recurso, reglas, de forma que un LMS posteriormente las interprete para representar adecuadamente el flujo y bifurcaciones descritas. Estas reglas son agregadas al árbol de actividades, definiendo completamente la navegación y el secuenciamiento que deberá adoptar cada LMS al consumir un contenido representado en SCORM. El árbol de actividades describe las relaciones entre los recursos educacionales y es definido en el *Content Organization* dentro del *Content Package*.

2.4 Ontología

2.5 DAML

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

3 ANÁLISIS

3.1 Correspondencias entre LOM y SCORM

Al iniciar el estudio de las correspondencias entre LOM y SCORM, claramente se percibe que se están manejando conceptos distintos y que no es posible establecer una correspondencia completa entre todos sus elementos.

Estamos comparando cosas distintas:

- SCORM es un entorno de especificaciones para definir, implementar y poner en marcha contenido de aprendizaje en los LMS.

- LOM en cambio, es un estándar que define una estructura de metadatos para describir un L.O.

SCORM provee una estructura mucho más amplia que LOM; mientras que LOM define la estructura de metadatos sobre un L.O., SCORM amplía este concepto agregándole distintas funcionalidades y especificaciones de forma de interoperar distintos LMS. Algunos ejemplos son la definición de navegación, metadatos sobre la estructura de SCORM (Organización, Recursos, etc.), entre otros.

En este proyecto hemos hecho particular hincapié en la transformación desde LOM a SCORM, dado que nuestro estudio está enmarcado dentro del proyecto AdaptWeb-Multicultural y este cuenta con descripciones de sus objetos de aprendizaje basadas en LOM. Es así, que será necesario transformar estas instancias desde LOM a SCORM para poder aprovechar las funcionalidades que brinda este último.

La transformación desde SCORM a LOM, se puede realizar por razones de compatibilidad con otras instancias LOM, más que por buscar mejorar en algún aspecto. Al ser SCORM un concepto mucho más amplio que LOM, en este tipo de transformaciones se pierde funcionalidad.

3.2 Problemas planteados para establecer las correspondencias

A continuación presentamos los problemas que hemos identificado al analizar las posibles correspondencias que pueden existir o se pueden establecer entre los dos modelos.

Esta sección solo presenta los problemas identificados sin expresar una solución concreta a los mismos. De todas formas, servirá como antesala para la descripción de las posibles transformaciones en la siguiente sección.

Las transformaciones no son directas
Entre los dos modelos, existen elementos puntuales con los cuales se puede establecer una correspondencia directa, pero son escasos. En líneas generales, y especialmente en la transformación de LOM a SCORM, existen muchas decisiones a tomar para realizar la transformación.

En algunos casos, se pueden tomar decisiones automáticas basadas en ciertos criterios que especificaremos posteriormente, pero en otros casos, la falta de una única interpretación semántica puede limitar la automatización.

El resultado de la transformación no es único
Al no haber una correspondencia directa entre los modelos y existir distintos caminos para realizar la transformación (ya sea de LOM a SCORM, o viceversa), se puede llegar a tener resultados muy variados dependiendo de las decisiones tomadas en el proceso. Esto conlleva que el resultado en el nuevo modelo tras aplicar la transformación, no necesariamente cumplirá las necesidades que cubría el modelo anterior.

La transformación debe ser rigurosa y solo puede tener resultados válidos dentro de las especificaciones del nuevo modelo. Pero un problema muy importante a tener en cuenta, es que a pesar de que el resultado en

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

el nuevo modelo, cumpla con las especificaciones del mismo, puede no reflejar fielmente el material original.

Las Relaciones entre instancias LOM permiten grafos, mientras que SCORM maneja solo estructuras arbóreas (sin ciclos)
El esquema base del estándar LOMv1.0, especifica que el relacionamiento entre instancias LOM debe mantenerse a través de la categoría *Relation* que cuenta con la siguiente estructura:

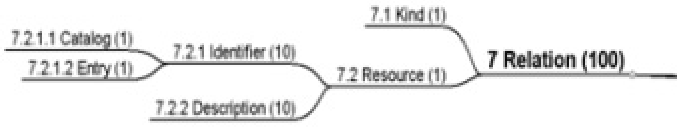


Figura 7: Categoría *Relation* de LOM [REHACER]

Como se desprende de la descripción de cada uno de los elementos de datos que componen la categoría *Relation* (ver anexo: Especificaciones sobre la Categoría Relation de LOM), un L.O. puede estar relacionado con cualquier otro L.O. sin restricción alguna. Esto implica que un conjunto de instancias de LOM puede estar determinando una estructura de grafo en sus relaciones.

Al poseer un grafo, dado que estos pueden tener ciclos, se plantea todo un desafío en la transformación de LOM a SCORM para poder lograr derivar de esa estructura, la definición de jerarquías necesaria para representar el *Content Organization* de SCORM, ya que este se representa mediante una estructura arbórea.

La Navegación y el Secuenciamiento de SCORM no son representables en LOM
En la especificación de SCORM, a un contenido *SCORM conforme* se le puede establecer una secuencia a través de un conjunto de eventos de navegación. Los flujos posibles del contenido pueden ser descritos por un conjunto de actividades predefinidas, típicamente definidas en tiempo de diseño que muchas veces se basan en el resultado de la interacción con el estudiante.
Toda esta funcionalidad, no está contemplada en la especificación de LOM, ya que este modelo simplemente se centra en especificar información acerca de un L.O. y sus relaciones con otros.
Es así, que en la transformación de SCORM a LOM, puede ser inevitable la pérdida de mucha información. En la transformación inversa, el resultado no representará la navegación y secuenciamiento.

La inevitable participación del diseñador de contenidos
En la transformación de LOM a SCORM, para poder instanciar adecuadamente un *Content Package* es necesario indicar el conjunto de L.O. que lo componen. El diseñador toma un gran papel en esta instancia, definiendo un conjunto de L.O. que representen "su" idea de paquete (lo que tras la transformación será el *Content Package*).

A nivel de definición de los datos, algunas correspondencias pueden definirse automáticamente (como se presenta en la siguiente sección), pero para que el significado y la intención de diseño se representen en forma adecuada, parece ser inevitable la intervención del diseñador de contenidos.

Cabe mencionar, que sobre este punto se hace especial hincapié en las guías y buenas prácticas que SCORM ofrece para los diseñadores de contenidos.

3.3 Transformación General

La transformación de LOM a SCORM que se analiza a continuación, consiste en:

> *Dado un conjunto de instancias de LOM, que representan a un conjunto de elementos de aprendizaje, se debe obtener un elemento definido en base a las especificaciones de SCORM, que represente lo más fielmente posible el material original.*

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

Estos elementos resultantes, definidos en base a las especificaciones de SCORM, pueden ser de tres tipos (*Asset, SCO, Content Package*) y por tanto determinan las siguientes transformaciones.

Identificamos las siguientes transformaciones desde un conjunto de instancias LOM:

- **Transformación a *Asset***
  - Algunas instancias de LOM, pueden ser representadas como Asset de SCORM. Si en el conjunto inicial de instancias de LOM hay más de una instancia, se deberá crear una agregación que incluya a todas las instancias para representar el Asset de SCORM.

- **Transformación a *SCO***
  - Algunas instancias de LOM, pueden ser representadas como *SCO* de SCORM. Si en el conjunto inicial de instancias de LOM hay más de una instancia, se deberá crear una agregación que incluya a todas las instancias para representar el *SCO* de SCORM.

- **Transformación a *Content Package***
  - Esta transformación es la más compleja. En este caso, además de transformar todas las instancias de LOM a los *Assets* y *SCOs* necesarios, el reracionamiento entre estas instancias LOM toma vital importancia para definir el árbol de organización del *Content Package* de SCORM.

A su vez se también se puede plantear una cuarta transformación, que es armar un Content Package a partir de distintos *SCOs y Assets* ya generados con las transformaciones anteriores.

- **(LOM)+ >>> Asset**
  - LOM(a) >>> Asset(a)
  - LOM(a) LOM(b) LOM* >>> Asset(a,b,…)
- **(LOM)+ >>> SCO**
  - LOM(a) >>> SCO(a)
  - LOM(a) LOM(b) LOM* >>> SCO(a,b,…)
- **(LOM)+ >>> ContentPackage**
  - LOM(a) LOM* >>> ContentPackage(a,...)
- **(Asset)+ (SCO)* | (Asset)* (SCO)+ >>> Content Package**
  - Asset(a) (Asset)* (SCO)* >>> ContentPackage(a,...)
    | (Asset)* SCO(a) SCO* >>> ContentPackage(a,...)

Figura 8 - Transformaciones identificadas, con sus distintos subcasos

Nota: La notación propuesta, toma como base la definición de expresiones regulares (por más detalles, ver anexo: Notación utilizada en las transformaciones).

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

Transformación: (LOM)+ >>> Asset
Para realizar la transformación de un conjunto de instancias LOM, directamente a un *Asset* de SCORM, se puede diferenciar dos casos:

i) LOM(a) >>> Asset(a)
Este primer caso, es el más simple de todas las transformaciones posibles que se le pueden aplicar a una instancia LOM.

Para esta transformación, se cuenta con el Perfil de Aplicación de los Metadatos, en el documento *SCORM Content Aggregation Model* (ver anexo: Perfil de Aplicación de los Metadatos SCORM), que indica cuales son los Elementos de Datos de la especificación de LOM que son opcionales o requeridos para conformar un *Asset*.

Teniendo en cuenta este perfil, la transformación consta simplemente en utilizar la misma instancia LOM exigiendo que los campos indicados como requeridos en el perfil, existan en la instancia.

ii) LOM(a) LOM(b) LOM* >>> Asset(a,b,...)
En este caso, para cada instancia de LOM se aplica el mismo criterio que en el punto anterior. Es decir, todos los elementos de datos especificados como requeridos en el Perfil de Aplicación de los Metadatos tienen que estar definidos en las instancias LOM.

El único paso adicional necesario, es que para representar el conjunto de instancias LOM como un solo *Asset* de SCORM, será necesario realizar una agregación de estas instancias LOM para poder tomar esta nueva instancia como base para el *Asset* de SCORM. Esta nueva instancia, también deberá contener los elementos de datos especificados como requeridos en el perfil y los mismos serán deducidos a partir del conjunto de instancias LOM que representa.

Transformación: (LOM)+ >>> SCO
Este tipo de transformaciones también se divide en dos casos como la transformación anterior:

i) LOM(a) >>> SCO(a)
Este primer caso, sigue los mismos pasos que la transformación 'LOM(a) >>> Asset(a)', pero al *Asset* generado, se le agrega la funcionalidad necesaria para representar un *SCO*. Es decir, la capacidad de iniciar y terminar una comunicación con un LMS.

ii) LOM(a) LOM(b) LOM* >>> SCO(a,b,...)
En este caso, se aplica el mismo criterio que en 'LOM(a) LOM(b) LOM* >>> Asset(a,b,…)' y de la misma forma que en la transformación anterior, se le debe agregar la funcionalidad necesaria para que el *Asset* de agregación que agrupa a las instancias de LOM iniciales, se represente como un *SCO*.

Transformación: (LOM)+ >>> Content Package
Este es la transformación más compleja. En ella se parte de un conjunto de instancias en LOM y se debe generar un paquete de contenido en SCORM.

Como se ha mencionado anteriormente, el paquete de contenidos de SCORM posee una jerarquía de organización del tipo arbórea sin ciclos, mientras que el relacionamiento de las instancias LOM, puede llegar a estar determinado por un grafo.

A continuación proponemos los pasos a seguir para la transformación de las relaciones que existen entre el conjunto inicial de instancias LOM y la determinación de la mejor toma de decisiones para construir a partir de ese grafo inicial, el árbol de organización necesario para construir el *Content Package*.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

3.3.1 Metodología para la Transformación: (LOM+) >>> Content Package

Los pasos que proponemos son los siguientes:

a) Analizar las relaciones entre todas las instancias de LOM
   i) Crear nuevas instancias que representen Relaciones de Agregación
   ii) Construir el Grafo de Organización
b) Definir la jerarquía de las instancias para pasar del Grafo al Árbol
c) Pasar las instancias de LOM a sus correspondientes en *Asset/SCO*
d) Construir la instancia de *Content Package*

a) Analizar las relaciones entre todas las instancias de LOM

Las relaciones entre las distintas instancias son especificadas en LOM por los siguientes tipos dentro del Elemento de Datos *Relation.Kind* :

- *Ispartof* : es parte de
- *haspart* : tiene parte
- *isversionof* : es versión de
- *hasversion* : tiene versión
- *isformatof* : es formato de
- *hasformat* : tiene formato
- *references* : referencia
- *isreferencedby* : es referenciado por
- *isbasedon* : está basado en
- *isbasisfor* : es base para
- *requires* : requiere
- *isrequiredby* : es requerido por

Si se desea una mayor descripción de lo que define cada tipo de relación, ver anexo: Tipos de Relación utilizados en LOM.

Como se puede apreciar, en estos tipos de relacionamientos existen dos agrupaciones bien diferenciadas. Por un lado, algunos tipos de relacionamiento permiten agrupar las instancias que relacionan (como ser: *isversionof*), mientras que los otros tipos de relacionamiento permiten organizar jerárquicamente las instancias que relacionan (como ser: *Ispartof*).

Es así que al analizar las relaciones de las distintas instancias especificadas en LOM que componen el grafo, agrupamos a estas en dos categorías que se presentan a continuación:

- *Relaciones de Agregación*
- *Relaciones de Organización*.

Relaciones de Agregación
Definimos como Relaciones de Agregación a aquellas que posiblemente implican que el elemento descrito y el elemento referenciado representan al mismo recurso.

- *isversionof* : es versión de
- *hasversion* : tiene versión
- *isformatof* : es formato de
- *hasformat* : tiene formato
- *isbasedon* : está basado en
- *isbasisfor* : es base para

En el caso de *isformatof* y *hasformat*, los recursos relacionados contienen el mismo contenido intelectual, pero con distinto formato.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

En el caso de *isversionof* y *hasversion*, aunque el contenido puede ser sustancialmente distinto, los dos recursos son distintas versiones de lo mismo.

Por último, según lo especificado en el anexo antes mencionado, *isbasedon* e *isbasisfor* (relaciones que indican que un elemento es base sobre otro, el primero es utilizado sobre para el sub-elemento y el segundo es utilizado sobre el elemento base) se utilizan para cubrir casos no cubiertos por las cuatro anteriores y lo que permiten, es representar recursos derivados (como ser correcciones, revisiones, cambios, etc.).

Por medio de estas Relaciones de Agregación, podemos agrupar distintos nodos del grafo de instancias LOM inicial, permitiendo representar cada grupo en una única nueva instancia.

Relaciones de Organización

Definimos como Relaciones de Organización todas aquellas relaciones que implican que entre el elemento descrito y el elemento referenciado existe un vínculo que permite establecer algún criterio de organización entre los dos.

- *ispartof* : es parte de
- *haspart* : tiene parte
- *references* : referencia
- *isreferencedby* : es referenciado por
- *requires* : requiere
- *isrequiredby* : es requerido por

i) Crear nuevas instancias que representen relaciones de agregación

Tomando como entrada el conjunto de Relaciones de Agregación identificadas en el paso anterior, se deben crear nuevas instancias LOM de forma tal que representen a cada uno de los subconjuntos de instancias LOM con relaciones de Agregación.

Para esto, a partir de cada uno de estos subconjuntos que se encuentran agrupados en relaciones de agregación, se genera una nueva instancia LOM que lo represente. A continuación la figura muestra los distintos objetos LOM **únicamente** con sus relaciones de agregación y la transformación que generan las nuevas instancias.

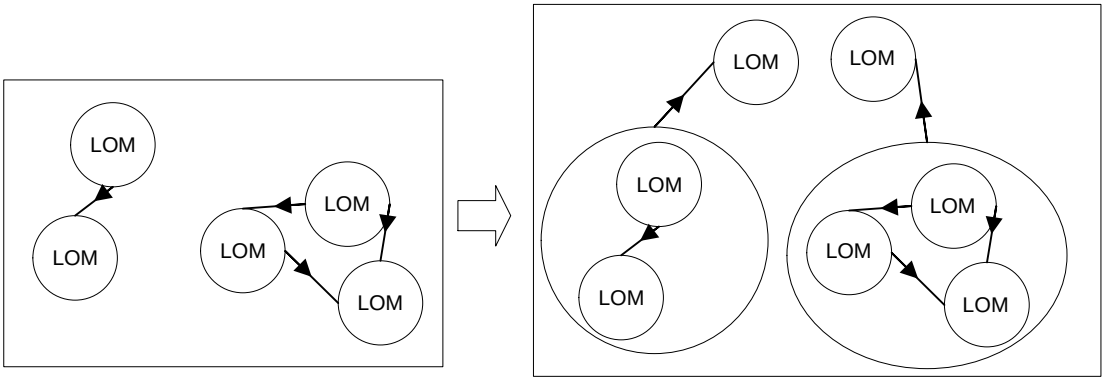


Figura 9: Ejemplo del uso de las Relaciones de Agregación

Estas nuevas instancias que representan relaciones de agregación, serán utilizadas para formar el grafo de organización en el paso siguiente, sustituyendo, cada una, al subconjunto de instancias LOM que le dio origen.

ii) Construir el Grafo de Organización

Una vez que son descartadas las relaciones de agregación por medio del punto anterior, se construye un nuevo grafo tomando como aristas las Relaciones de Organización y como nodos: las nuevas instancias que representan relaciones de agregación y las instancias que no formaron parte de ningún subconjunto de agregación.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

b) Definir la jerarquía de las instancias para pasar del Grafo al Árbol

Una vez representado el Grafo de Organización, debemos realizar su transformación a un árbol de jerarquía de SCORM.

A continuación presentamos tres alternativas:

i) Exigir que el diseñador de contenido, especifique la estructura del árbol dentro del grafo de instancias LOM.

Las posibles decisiones que puede tomar son:

- Descartar aristas (relaciones) del grafo
- Descartar nodos (instancias en LOM) del grafo
- Seleccionar subconjuntos de nodos para agruparlos (Este caso es similar a la detección de subconjuntos por medio de relaciones de agregación anteriormente mencionada).

Ventajas

- La solución refleja la selección del diseñador
- Es fácil de implementar
- Se evita la complejidad del punto mas crítico en la automatización

Desventajas

- La definición del árbol queda a completo juicio del diseñador
- Es necesaria la interacción con el diseñador, por lo tanto no se puede automatizar
- Si el grafo es muy complejo, es muy probable que las decisiones del diseñador no sean correctas
- La experiencia del diseñador de contenido pasa a ser un aspecto crítico

ii) Utilizar el grafo de instancias LOM y generar un árbol con hojas repetidas

Una conocida forma de convertir un grafo en un árbol, es la generación de hojas repetidas, que permiten representar las múltiples aristas de entrada a un nodo. En este caso, como las hojas representan *Assets* ó *SCOs* del árbol de organización, se genera contenido redundante y posiblemente se pierda coherencia si no se fuerzan posteriormente los secuenciamientos dentro del árbol.

Ventajas:

- Fácil de implementar.

Desventajas:

- Repetición de información.
- Requiere personalizar algún elemento de SCORM para diferenciar los nodos duplicados.
- Al generar el árbol, se definirá secuencias entre los distintos elementos. Para determinar el correcto orden es necesario agregar criterios que permitan generar el contenido en forma coherente.

iii) Asociar distintos grafos de instancias LOM en *Asset/SCO* para poder formar un árbol

Este caso deberá analizar el grafo de relaciones, identificando subgrafos de manera de representar a estos por medio de la relación de agregación. Sobre este subgrafo se generará un nuevo elemento que represente las relaciones entre estos nodos (instancias de LOM) de forma similar al proceso utilizado en el paso a) i).

La detección de subgrafos se realizará mediante diferentes estrategias que permitan identificarlos.

Ventajas:

- Se minimiza con respecto a los métodos anteriores:
  - la pérdida de contenido semántico.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

- la intervención del diseñador de contenidos.

Desventajas:.

- La toma de decisiones puede ser incorrecta, debido a la complejidad de las estructuras.

Dadas las ventajas y desventajas expuestas para resolver este paso, concluimos que el mejor método a utilizar es el descrito en el punto c, ya que este minimiza la pérdida de contenido semántico, a pesar de que agregue complejidad a su implementación.

c) Pasar las instancias de LOM a sus correspondientes en Asset/SCO

Para cada instancia de LOM del árbol obtenido, se tiene que tomar la decisión de representarla como un *Asset* ó un *SCO* dentro del árbol de organización.

A continuación presentamos los Elementos de Datos que determinamos, tras haber analizado toda la estructura de LOM, que podrían ser útiles en esta toma de decisión.

Cabe mencionar, que estos Elementos de Datos no necesariamente pertenecerán a las distintas instancias LOM a transformar, ya que como se mencionara anteriormente (ver en LOM: Elementos de Datos), ningún Elemento de Datos de la estructura de LOMv1.0 es requerida.

- ***LOM//1.7:General.Structure***

  Especifica el tipo de estructura del L.O. y cuenta con los siguientes valores:

  - *atomic* : un objeto que es indivisible (en este contexto).
  - *collection* : un conjunto de objetos que no tienen una relación especificada entre ellos.
  - *networked* : un conjunto de objetos con relaciones que están sin especificar.
  - *hierarchical* : un conjunto de objetos cuya relación puede ser representada por un árbol.
  - *linear* : un conjunto de objetos que están completamente ordenados (Ej.: objetos conectados por relaciones de "previo", "siguiente").

- ***LOM//1.8:General.AggregationLevel***

  Especifica la granularidad funcional del L.O. y cuenta con los siguientes valores:

  - *1* : el menor nivel de agregación (Ej.: archivos multimedia en bruto, o fragmentos)
  - *2* : una colección de L.O. nivel 1 (Ej.: Lección)
  - *3* : una colección de L.O. nivel 2 (Ej.: Curso)
  - *4* : el mayor nivel de agregación (Ej.: Cursos pertenecientes a un certificado)

- ***LOM//5.2:Educational.LearningResourceType***

  Especifica el tipo de L.O. (especificando el tipo más dominante primero).

  - Los valores posibles son: *exercise, simulation, questionnaire, diagram, figure, graph, index, slide, table, narrative text, exam, experiment, problem statement, self assessment* y *lecture*

- ***LOM//5.4:Educational.SemanticDensity***

  Indica el grado de densidad semántica de un LO.

  - Los valores posibles son: *very low, low, medium, high, very high*

  Su clasificación está relacionada con *5.1:Educational.InteractivityType* pero parece ser un elemento de datos muy poco especifico para poder inferir algún tipo de decisión (CanCore ***[REF!]*** también recomienda no utilizarlo).

A su vez, también se planteó la utilización de la categoría ***Classification***, pero según lo especificado en LOMv1.0 no parece ser útil en un estudio genérico de estas transformaciones. Sí podría ser tenido en cuenta, en casos específicos donde el conjunto de instancias LOM utilizara una clasificación propia mediante la cual se pudiera definir algún criterio para identificar si cada instancia es un *SCO* o un *Asset*.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

Metodología para la toma de decisión
El Elemento de Datos más representativo de los identificados anteriormente, es ***1.7:General.Structure***. Si este elemento cuenta con un valor en la instancia y el mismo es "*atomic"*, entonces queda determinado que esa instancia se debe representar como un *Asset*. Para los otros valores, no queda totalmente definido que sea un *SCO*, ya que la instancia podría consistir de un conjunto de *Assets*.

El segundo elemento a tener en cuenta, es el ***1.8:General.AggregationLevel***. En este caso, si este es mayor que "*1"*, la instancia está representando una estructura compleja como se desprende de la descripción del elemento. Es por esto que la instancia se representaría como *SCO*.

Un tercer elemento que puede servir para inferir el significado de la instancia, es el ***5.2:Educational.LearningResourceType***, pero cabe mencionar que se puede estar poniendo en juego la fidelidad de la transformación con una decisión automática basada en este elemento de datos, ya que el mismo puede ser utilizado con distintos criterios.
Si el valor de este elemento se encuentra en el siguiente conjunto {*diagram, figure, slide, narrative text* }, podríamos inferir que debemos representar la instancia como un *Asset*.

En caso de que no se haya podido inferir una decisión por medio de los elementos de datos antes mencionados, deberemos recurrir a la intervención del diseñador de contenidos.

d) Construir la instancia de Content Package
Este paso es equivalente a la transformación '(Asset)*(SCO)* >>> Content Package' analizada a continuación en forma independiente.

3.3.2 (Asset)+(SCO) | (Asset)*(SCO)+ >>> Content Package*

En este caso contamos con todos los elementos necesarios para construir el *Content Package* de SCORM. Es así que este caso es meramente un problema de construcción en el cual no es necesario ningún análisis para toma de decisiones.

Para esta construcción se deben aplicar las metodologías descritas en las guías de SCORM para representar el contenido por medio del *Content Package* explicado y definido anteriormente (ver: SCORM).

3.4 Transformación Específica

A continuación se presenta la definición de Objetos de Aprendizaje para el contexto educativo del proyecto AdaptWeb (propuesto en el trabajo "Ontology-based Metadata for e-learning content" de Lydia Silva - 2004 [LSilva]). Posteriormente se presentará la transformación específica, entre instancias creadas utilizando este modelo, y su representación correspondiente en formato SCORM.

En su trabajo, Lydia Silva realiza un análisis de los sistemas educativos actuales, intentando adaptar el contenido y estilo de aprendizaje existente a un contexto que se centre en las características de un alumno particular, posibilitando así, la adaptación de contenido a los estudiantes. Otro tema que analiza, es la capacidad de interoperar contenido en la Web reutilizando distintos objetos de aprendizaje. Para poder modelar el contenido educativo, utilizó ontologías de forma de poder refinar el vocabulario existente en la Web posibilitando así el cumplimiento de los objetivos antes mencionados. Las ontologías permiten representar un lenguaje formal de metadatos relativos de los objetos de aprendizaje simples, de forma de incluir reglas para definir posibles formas de agrupamiento más complejo.

Como base para la definición de un lenguaje formal que especifique los objetos de aprendizaje, se construyó un Perfil de Aplicación sobre el estándar LOM. Con este Perfil de Aplicación se agrega contenido semántico en sus definiciones y se especifican con mayor precisión distintos elementos de LOM para posibilitar un mayor enriquecimiento del L.O. De esta forma un patrón general como puede ser LOM, se puede adaptar a un ambiente de trabajo como AdaptWeb, definiendo "su" semántica y agregando elementos que en este ambiente son relevantes en la definición de contenido.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

3.4.1 Modelo de Datos AdaptWeb

En esta sección se presenta el modelo de datos creado para lograr interoperabilidad y adaptabilidad en el proyecto AdaptWeb. Este análisis es extraído del trabajo de tesis de maestría antes mencionado [LSilva].
La propuesta implica diseñar e implementar un repositorio de metadatos, describiendo el contenido educacional y el perfil del estudiante.

En primera instancia se define un *application profile* del estándar LOM, adecuándolo al contexto educacional del proyecto AdaptWeb. Las ontologías Web se proponen para mantener un dominio de conocimiento y un perfil sobre cualquier estudiante relacionado al sistema. A su vez, la adaptabilidad del contenido educacional se puede lograr construyendo L.O. más complejos, creándolos a la medida de cada perfil de estudiante y basándolos en el contenido de la ontología de conocimiento.

Los pasos propuestos para aplicar tecnología Web a contenido educacional, en pro de obtener interoperabilidad y adaptabilidad son:

1- Conceptualizar el modelo de dominio y el perfil del estudiante. El modelo se puede representar en distintos lenguajes semánticos como puede ser el modelo Entidad Relación, UML o cualquier otro lenguaje de ontología para la Web como DAML+OIL.

2- Crear un *application Profile* sobre el estándar LOM teniendo en cuenta las particularidades del contenido educacional del ambiente referenciado.

3- Si el modelo creado en el paso 1 es una ontología, entonces, se debe adecuar el vocabulario sobre la definición del *application profile* que se realizó. De otra manera, se debe construir una ontología para el dominio de aprendizaje y otra para el perfil del estudiante basado en el modelo conceptual existente y el vocabulario del *application profile.*

4- Poblar la ontología de dominio con descripciones individuales de metadatos de contenido hipermedia. Se debe editar la ontología en caso de ser necesario para enriquecerla.

5- Poblar la ontología del estudiante con el conocimiento que se tiene de los estudiantes en el sistema.

6- Crear agentes automáticos que definan la navegación más adecuada para cada estudiante en tiempo de ejecución a partir de las ontologías de dominio y estudiante.

Modelo de Dominio

Los metadatos relacionados al modelo de dominio, están determinados por toda aquella información que un sistema puede necesitar en tiempo de ejecución sobre un L.O. determinado, así como por los datos sobre las personas que realizaron el L.O. El modelo conceptual (especificado en lenguaje E.R.) se muestra en la siguiente figura.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

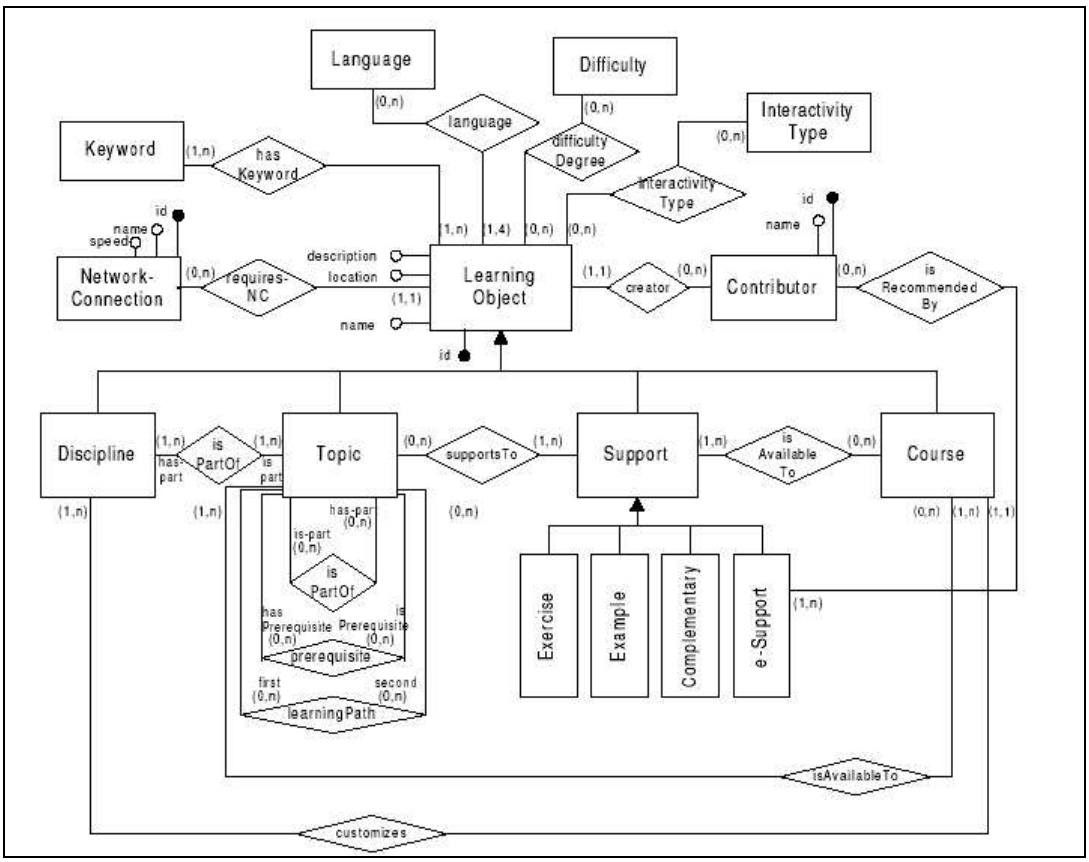


Figura 10: Modelo de Dominio AdaptWeb (extraído de [LSilva])

El elemento más general que modela a cualquier L.O. en el sistema es la entidad *Learning Object.* La información no solo se describe por atributos, sino que parte de ella se representa por medio del modelado de las estructuras. A continuación se presentan las entidades y relaciones más importantes.

Entidades: *Discipline, Topic, Support y Course* son especializaciones de la entidad *Learning Object*.

Un ***Topic*** es la explicación concreta de un concepto o idea. Un *Topic* puede ser complementado por algún ejemplo, ejercicio o material complementario, representados cada uno por las entidades: *Example*, *Exercise*, *Complementary* y *e-Support,* del tipo ***Support***. Adicionalmente, un *Topic* puede tener *sub-topics*, los cuales contienen mayor especificación que el primer *Topic*. Esta especificación se realiza por la relación *isPartOf.* Esta relación es modelada por los roles *isPart* y *hasPart.*

El orden en que son presentados los tópicos a los estudiantes, está dado por las relaciones *learningPath.* De igual manera estas relaciones están indicadas por el orden natural que tienen dos L.O. por los roles *first* y *second.* Por último esta entidad puede tener la relación *prerequisite,* la cual indica que un estudiante que quiera consumir este L.O. debe poseer los tópicos que tiene como prerrequisito al tópico indicado. Esta relación es modelada por los roles *isPrerequisite* y *hasPrerequisite.*

La entidad ***Course*** representa una disciplina adaptada específicamente para un grupo de estudiantes con un perfil determinado. El contenido educacional de un curso esta compuesto por todos los tópicos que se crean adecuados para la audiencia del curso. Por ejemplo distintos conjuntos de ejercicios o ejemplos pueden ser material de diferentes cursos.

La entidad *e-support* contiene metadatos sobre un recurso Web que no pertenece al sistema, pero que se considera adecuado para complementar la explicación de algún tópico. Cada elemento de un *e-support* debe haber sido recomendado por un contribuyente, especificado en el modelo, mediante la asociación a través de la relación *isRecommendedBy* con la entidad *Contributor.*

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

Modelo del Estudiante
El modelo del estudiante intenta caracterizar los elementos de un perfil de estudiante con objetivos específicos relacionados a su aprendizaje (ver anexo: Modelo del Estudiante de AdaptWeb).
Dado que este proyecto se centra en la transformación del contenido educacional y no así de la información del perfil del estudiante, esta información no fue tenida en cuenta para la transformación específica.

Definición del Perfil de Aplicación de LOM
El Perfil de Aplicación sobre el estándar LOM, es definido para representar los metadatos necesarios dentro del contexto del proyecto AdaptWeb.

El Perfil de Aplicación esta modelado en tres etapas:
1- Inclusión de algunas propiedades definidas en el estándar LOM para representar elementos del modelo conceptual antes definido.
2- Definición de nuevas propiedades, basadas en refinamientos específicos de algunas propiedades existentes en LOM, para representar otros elementos del modelo conceptual.
3- Planteo de restricciones sobre los valores de las propiedades con la finalidad de utilizar únicamente valores que sean pertinentes al contexto del ambiente definido en el proyecto AdaptWeb.

En el anexo Perfil de aplicación de AdaptWeb se describen las tres etapas que definen el perfil de aplicación.

3.4.2 Versión de SCORM a utilizar
Si bien la versión 1.3 de SCORM es la última e incorpora varios cambios respecto a la versión 1.2, analizamos algunos puntos relevantes en el proyecto para elegir cual de estas versiones se comporta mejor.

SCORM 1.3 es la última versión, al momento de realizar este proyecto, con fecha de última revisión en julio de 2004. Tiene grandes cambios en la parte de secuenciamiento y agrega algunos aspectos de navegación por medio de reglas. Pero como desventaja encontramos que para validar contenido generado en esta versión solamente se encuentra una herramienta en la Web provista por ADL. En esta herramienta, simplemente se puede validar el contenido haciendo un breve reporte de error que en algunos casos no especifica correctamente el error encontrado. Por otra parte, debido a lo reciente que es esta última versión, ningún LMS actualmente cumple con la especificación. Limitándonos así, a que el contenido desarrollado en esta versión no pueda ser utilizado en un servidor LMS.

Por otra parte, la versión 1.2 de SCORM, tiene como limitante que la navegación y el secuenciamiento son pobres, representados por medio del árbol de *Content Organization* y restricciones de prerrequisito. Permitiendo únicamente una navegación secuencial limitada por las reglas impuestas.
Pero a pesar de ello, esta versión cuenta con innumerables LMS y editores que visualizan adecuadamente el contenido generado.

Analizando la Ontología de AdaptWeb, vemos que en esta no existen reglas de secuenciamiento complicadas, sino que únicamente se limita la navegación por medio de la misma regla que contiene la versión 1.2 de SCORM (prerrequisito). Aparte contiene otro tipo de relaciones entre los distintos objetos que podrán ser representadas correctamente en el árbol del *Content Organization*.

Por lo analizado en los puntos anteriores, es que en la transformación específica decidimos utilizar la versión 1.2 de SCORM. De esta manera, estaríamos en condiciones de desarrollar una herramienta que realice la transformación entre estas dos especificaciones, permitiéndonos, validar correctamente el contenido, editarlo y publicarlo en distintos LMS.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

3.4.3 Metodología para la Transformación Específica

A continuación se presenta una metodología que sienta las bases para el desarrollo de una aplicación capaz de procesar un archivo DAML conteniendo la ontología AdaptWeb (resultante de la generación automática de metadatos propuesta por Lydia Silva [LSilva]) para obtener así un *Content Package* de la especificación SCORM v1.2 que represente al conjunto de L.O. presentes en el DAML.

Metodología:

a) Determinar el conjunto de Instancias LOM

- Determinando los conjuntos de *Discipline, Topic, Support* (separados por: *Exercise, Example, Complementary, e-Support*) y *Course*

b) Construir Assets/SCO a partir de las instancias LOM, generando los Resources correspondientes

- Crear Assets a partir de *Support*
- Crear Assets (SCA) a partir de *Topic* y *Support* unidos por *supportsTo*
- Construir SCO a partir de *Topic*

c) Generación de los Content Package en SCORM

Se genera un Content Package por objeto *Discipline*

i) *Para cada Discipline con relación customize a un objeto Course, se genera un árbol de organización*

- Se construye el primer nivel del árbol de organización
- Se realizan deducciones a partir de la relación *isPartOf*
- Se ordenan los objetos *Topic* hermanos, basándose en la relación *LearningPath*
- Se realizan deducciones a partir de la relación *supportsTo*
- Se termina de armar el árbol de organización

ii) Construir la instancia Content Package de SCORM para el objeto *Discipline*

- Se construye el manifiesto (utilizando los árboles de organización generados en 3.1 para el objeto Discipline y los recursos necesarios del conjunto de recursos generados en el paso 2).
- Se construye el paquete.

a) Determinar el conjunto de Instancias LOM

El primer paso que se debe realizar en la transformación, es determinar el conjunto de instancias LOM con las que cuenta el DAML que se va a procesar.

Según el modelo de dominio definido previamente, las instancias de los distintos L.O. pueden representar cualquiera de las siguientes entidades: *Discipline, Topic, Support (los cuales pueden ser de los tipos: Exercise, Example, Complementary, y e-Support) y Course.*

Las mismas pueden ser identificadas dentro del DAML, por la etiqueta <daml:Class> que se especifica como el <rdf:type>, dentro de la etiqueta <rdf:Description> que describe la instancia del L.O. A modo de ejemplo, a continuación se presenta un resumen de la descripción de una instancia tipo *Discipline*:

```
<rdf:Description rdf:about="# Computacao Algebrica e Numerica">
        <rdf:type>
                <daml:Class rdf:about="#Discipline"></daml:Class>
        </rdf:type>
</rdf:Description>
```

A continuación especificaremos como determinar cada una de las entidades posibles de un L.O. por medio de la sintaxis antes descrita.

i) Determinar conjunto de instancias LOM tipo Discipline

Las instancias LOM de tipo *Discipline*, se determinan buscando todo aquel L.O. que tenga una instancia *type* de valor *#Discipline*. A continuación se especifica su descripción en DAML.

```
<rdf:type>
        <daml:Class rdf:about="#Discipline"></daml:Class>
</rdf:type>
```


Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

ii) Determinar conjunto de instancias LOM tipo Topic

Las instancias LOM de tipo *Topic*, están definidas por:

```
<rdf:type>
        <daml:Class rdf:about="#Topic"></daml:Class>
</rdf:type>
```

iii) Determinar conjunto de instancias LOM tipo Support

Las instancias LOM de tipo *Support*, pueden ser de los siguientes cuatro tipos.

Exercise

Las instancias LOM de tipo *Exercise*, están definidas por:

```
<rdf:type>
        <daml:Class rdf:about=" #Exercise"></daml:Class>
</rdf:type>
```

Example

Las instancias LOM de tipo *Example*, estan definidas por:

```
<rdf:type>
        <daml:Class rdf:about="#Example"></daml:Class>
</rdf:type>
```

Complementary

Las instancias LOM de tipo *Complementary*, están definidas por:

```
<rdf:type>
        <daml:Class rdf:about="#Complementary"></daml:Class>
</rdf:type>
```

e-Support

Las instancias LOM de tipo *e-Support*, están definidas por:

```
<rdf:type>
        <daml:Class rdf:about="#e-Support"></daml:Class>
</rdf:type>
```

iv) Determinar conjunto de instancias LOM tipo Course

Las instancias LOM de tipo *Course*, están definidas por:

```
<rdf:type>
        <daml:Class rdf:about="#Course "></daml:Class>
</rdf:type>
```

b) Construir Assets/SCO a partir de las instancias LOM

Tras conocer todas las instancias LOM definidas en el DAML, se debe transformar cada una de ellas, a las correspondientes unidades del *Content Package* de SCORM que se quiere construir.

SCORM maneja dos tipos de elementos fundamentales para conformar el *Content Package*: los Asset (que son la unidad mínima de contenido representable y que pueden estar agrupados en los llamados SCA) y los SCO (que son un conjunto de Asset, con la particularidad de que permiten su manipulación mediante LMS).

Al transformar cada una de las instancias LOM en sus correspondientes unidades del *Content Package* de SCORM, se debe generar por cada una de ellas, un *Resource* que posteriormente se utilizará en el armado de los *Content Package* necesarios.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

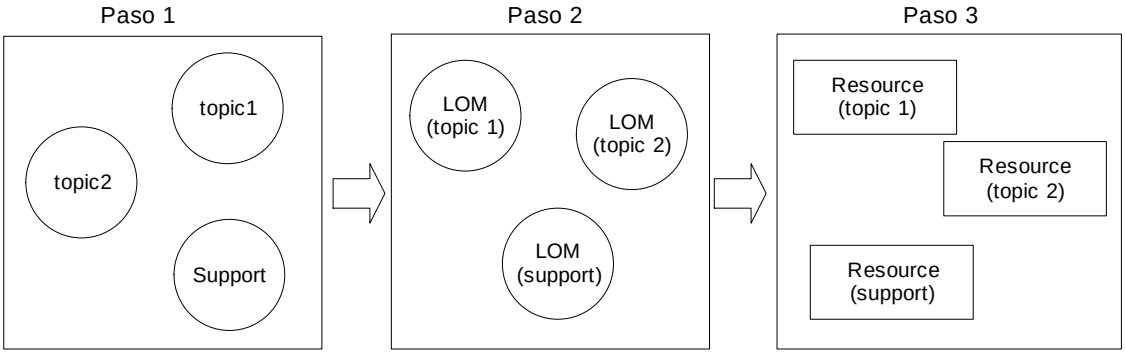


Figura 11: Pasos en la transformación, desde los elementos AdaptWeb, hasta los Resources de SCORM

Los *Resources* tienen la estructura definitiva para ser incluida en él, o los *Content Packages* finales. Los metadatos identificados en el segundo paso generarán un archivo xml con todos los metadatos propios del objeto (en el ejemplo que se muestra a continuación, este archivo se llama "sco.xml"). Esta será la referenciada en el tag *<adlcp:location>* dentro de los metadatos del *Resource*.

A continuación se muestra un ejemplo de un tipo *Resource* donde los metadatos se encuentran en el archivo "sco.xml" y referencia directamente a un archivo "SCORMexample.html"

```
<resources>
        <resource identifier="RESOURCE1" type="webcontent" adlcp:scormtype="sco" href="SCORMexample.html">
                <metadata>
                        <schema>ADL SCORM</schema>
                        <schemaversion>1.2</schemaversion>
                        <adlcp:location>sco.xml</adlcp:location>
                </metadata>
                <file href="SCORMexample.html"/>
        </resource>
        ..
</resources>
```

i) Construcción de Assets

Dentro de la especificación SCORM, los *Assets* son la forma más básica para describir un recurso de aprendizaje. Teniendo la condición de que estos deben representar la información de un recurso específico (texto, imagen, sonido, multimedia, o cualquier otro objeto simple que será presentado a un cliente Web), en el modelo de dominio definido previamente, las únicas entidades que validan esta definición son todas aquellas de tipo *Support* y algunas de las instancias *Topic.*

A partir de Instancias LOM del tipo Support

Las instancias LOM del tipo *Support* cumplen con las características adecuadas para ser representadas por medio de *Assets* de SCORM y por tanto se convierten directamente.

A partir de Instancias LOM del tipo Topic

Para determinar las instancias LOM del tipo *Topic* que pueden llegar a ser representadas por medio de *Assets*, enunciamos:

*Dado un árbol determinado por las relaciones **isPartOf** (aristas) entre elementos del tipo **Topic** (nodos), las hojas del mismo, son candidatas a ser representadas por medio de **Assets** de SCORM.*

De esta manera, en estos casos se tiene solamente las instancias que no tienen relaciones con otros tópicos, o sea, todas aquellas instancias que por si solas tengan la información suficiente para representarse. A modo de ejemplo, si tenemos los siguientes relacionamientos: *A has-part B, B has-part C, B has-part D* y *A has-part E*, entonces solo se deberá tener en cuenta las instancias D y E, por ser estas las hojas del árbol.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

Tras determinar cuales son las instancias LOM de tipo *Topic* mas atómicas o pequeñas, una forma de confirmar que se trata de un elemento simple como para ser representado por un *Asset*, es asegurarse que el mismo **no** tenga relacionamientos del tipo *supportsTo* con otras instancias LOM de tipo *Support*. De tenerlas, entrarían dentro del conjunto de instancias que deben ser tenidas en cuenta para construir un SCA.

Para la generación de los metadatos específicos que definen los elementos de SCORM (ya sea un Asset, un SCO o un Content Aggregation), realizamos un estudio de las correspondencias entre la especificación del Application Profile de SCORM con la información brindada por el Modelo de Dominio AdaptWeb. (ver anexo: Correspondencias entre Metadatos SCORM y AdaptWeb)

ii) Construcción de colección de Assets (SCAs)

Los SCA, son conjuntos de *Assets* y por tanto deben ser construidos en base a agregaciones de instancias LOM que se pueden deducir de los relacionamientos entre elementos simples.

A partir de instancias LOM de los tipos Topic y Support unidos por supportsTo

Como vimos en la construcción de Assets a partir de Instancias LOM del tipo *Topic*, cuando analizamos los árboles de relacionamientos entre las instancias LOM del tipo *Topic*, para encontrar posibles *Assets* tomamos en cuenta solo las instancias hojas de esos árboles. Si estos poseen alguna relación del tipo *supportsTo* con alguna instancia LOM del tipo *Support*, entonces estamos ante un conjunto de instancias que pueden ser representadas por un SCA.

En este punto, el problema que enfrentamos es que no tenemos forma de saber si este *Topic* que posee elementos *Support* debe ser representado por medio de un SCA o un SCO, ya que siempre va a depender del nivel de granularidad semántica con el que cuente el árbol de tópicos, e incluso puede depender de la intención con la cual se haya creado ese tópico especifico.
El nivel de granularidad semántica del árbol, se podría inferir por la profundidad en la que se encuentra la hoja, pero de todas formas se estarían tomando suposiciones.

Para la toma de decisión proponemos tres alternativas:

1- que el sistema infiera la transformación en base al análisis de profundidad de la hoja en el árbol
2- que la inferencia de la transformación en base al análisis de profundidad de la hoja en el árbol, sea configurable en el sistema. De esta forma quien configure el sistema, le estará aportando la granularidad semántica que en la opción uno es supuesta por el sistema.
3- Que el sistema sugiera la transformación inferida en base al análisis de profundidad de la hoja, pero que sea el diseñador de contenidos quien tenga la última palabra.

Luego de identificado el tipo de elemento SCORM a utilizar, se genera el elemento utilizando la correspondencia entre los metadatos antes mencionada (ver anexo: Correspondencias entre Metadatos SCORM y AdaptWeb).

iii) Construcción de SCOs

Dentro de SCORM, los SCOs son los objetos de aprendizaje que cuentan con la cualidad de poder ser manejados por un LMS. Para esta transformación, los SCOs serán formados a partir de algunos de los objetos del tipo *Topic* con los que cuenta el DAML, ya que como vimos en las secciones anteriores, otros son transformados a Assets y SCAs.

Las instancias LOM del tipo *Topic* que serán tenidas en cuenta, son todas las instancias que no son hojas en los árboles de relacionamiento *isPartOf* presentados en el punto "Construcción de *Assets* a partir de Instancias LOM del tipo *Topic*".

La generación del SCO, se realizará utilizando la correspondencia entre los metadatos antes mencionada (ver anexo: Correspondencias entre Metadatos SCORM y AdaptWeb).

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

c) Generación de los Content Package en SCORM

Una vez que se cuenta con las instancias de los L.O. especificadas en formato SCORM, poseemos los elementos necesarios, para generar los *Content Package* que representarán cada uno de los *Discipline.* A continuación definiremos como se generará/n el/los *Content Package* en SCORM a partir del ontología de AdaptWeb.

Antes de comenzar con el análisis de los problemas y sus soluciones, se definirá el objeto *Course. Course* es un concepto abstracto el cual es utilizado para indicar el perfil de aplicación de los otros objetos (*Discipline, Topic y Support*). Por ejemplo, para un objeto *Discipline,* "Métodos Numéricos", se puede definir sobre él mismo y sobre sus objetos pertenecientes (*Topic y Support*) restricciones para aplicarlos sobre un determinado ambiente *Course.* Esta restricción es determinada por la relación *isAvailableTo,* que permite definir para cada objeto, que ambiente de *Course* debe ser aplicado. A su vez se pueden definir varios objetos *Discipline*, y sobre cada uno de ellos se plantea nuevamente el problema anteriormente mencionado.

Como se puede analizar en la sección de SCORM, el *Content Package* representará al objeto de aprendizaje, indicando el árbol de organización, recursos, submanifiestos y los metadatos de cada uno de los elementos contenidos en él. Es en este punto en el cual se plantean varias opciones:

Alternativa 1. Generar un *Content Package* que represente todo el contenido

Para transformar toda la información contenida en la ontología, se deberían generar tantos árboles de organización como objetos *Discipline* existan, y a su vez definir para cada uno, el posible perfil de *Course* (por lo mencionado anteriormente). Si solo tuviéramos un árbol de organización por objeto *Discipline*, tendríamos que definir para cada objeto perteneciente a la *Discipline* (como lo hace la ontología)*,* sobre que *Course* este es accesible.

Este tipo de información no es representable utilizando la versión 1.2 de SCORM, por lo que la alternativa sería definir sobre cada *Discipline,* tantos árboles de organización como posibles *Course* posea.

Ventajas y desventajas

- Al representar todo en un solo *Content Package* los recursos serían siempre los mismos, permitiendo que varios ítems del árbol de organización tuvieran un mismo recurso asociado. Esto nos garantizaría que la información no se duplicará.
- La función de transformación podría ser total, pero se deberían realizar ciertas transformaciones que generarían contenido erróneo para algunos LMS.

Alternativa 2. Generar un *Content Package* por cada *Discipline* y *Course*

En este caso, se generarían tantos *Content Package* como combinaciones de *Discipline* y *Course* existan. Por ejemplo, si tenemos dos *Course* y tres *Discipline* tendríamos que construir seis *Content Package*.

Ventajas y desventajas

- La función de transformación sería total.
- Permitiría distribuir cada *Discipline* adaptada a un *Course* de manera totalmente independiente.
- Podrían repetirse los árboles de organización.
- Se repetirían los archivos físicos (recursos), dado que para cada *Content Package* se deben agregar los archivos físicos que referencia.

Alternativa 3. Generar un *Content Package* por *Discipline*

Con esta alternativa, se crearía un *Content Package* por *Discipline*, generando dentro de cada uno varios árboles de organización, uno para cada ambiente *Course* en donde puede ser utilizado.

Ventajas y desventajas

- La función de transformación sería total.
- Podrían repetirse los árboles de organización, pero los archivos físicos serían siempre los mismos.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

Para este trabajo, elegimos la última opción para generar los *Content Package* ya que analizando las ventajas y desventajas entre ellas, podemos ver que con esta opción no vamos a tener el problema de duplicación de recursos que posee la opción 2, ni la complejidad que brinda la opción 1 pudiendo generar conflictos con los LMS.

A continuación especificaremos la metodología indicando las relaciones que tendremos en cuenta para construir el *Content Package* para cada objeto *Discipline* hallado en la ontología de AdaptWeb.

i) Para cada Discipline con relación customize a un objeto Course, se genera un árbol de organización

Para cada relación *customize* entre el objeto *Discipline* elegido y cada uno de los objetos *Course,* debemos crear un árbol de organización en el cual se **determinen la jerarquía de los distintos elementos,** en base a las relaciones: *isPartOf* , supportsTo y *learningPath* de la ontología

Es así que se deben aplicar estos pasos para cada objeto *Course* que tenga una relación con el objeto *Discipline* actual.

En los siguientes pasos daremos por sobre entendido que se está trabajando con un objeto *Course* y un objeto *Discipline* específicos.

Teniendo en cuenta solamente las instancias del tipo *Topic*, o del tipo *Support* con una relación *isAvailableTo* al objeto *Course* actual y las relaciones *isPartOf* de esos objetos *Topic* con otras instancias del tipo *Topic* y con el objeto *Discipline* actual, construiremos el árbol de organización.

Se construye el primer nivel del árbol de organización

La raíz del árbol de organización va a estar determinada por un Item *cluster* llamado <Discipline>_<Course> (donde <Discipline> es el nombre del objeto Discipline y <Course> el nombre del objeto Course).

Se realizan deducciones a partir de la relación isPartOf

La metodología para encontrar los objetos relacionados al *Discipline y Course* asociados, debe ser: inicialmente buscar cualquier objeto *Topic* que tenga una relación *isPartOf* con la *Discipline* y luego para cada uno de ellos, buscar entre las demás instancias los objetos con esta relación de manera de construir un grafo con todos los elementos asociados a la *Discipline*. Como restricción importante, se debe tener en cuenta que estos objetos tengan una relación *isAvailableTo* con el objeto *Course* asociado.

Dado que en el DAML origen, la relación *isPartOf* para cada instancia, es especificada explícitamente para todos sus predecesores, tenemos que filtrar las relaciones que duplican información.

En la siguiente figura se muestra el problema y su solución:

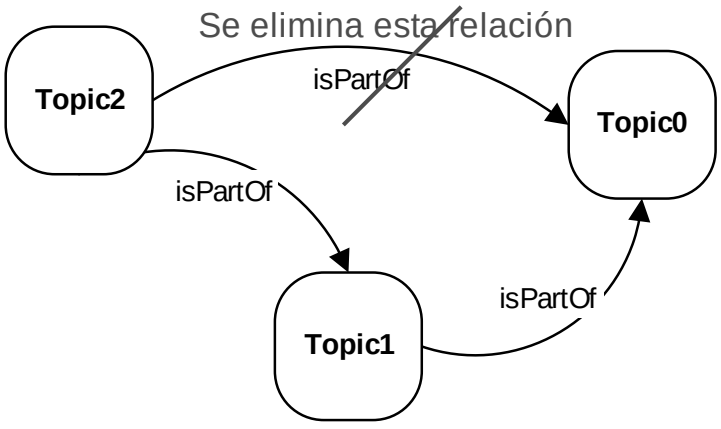


Figura 12: Eliminación de información duplicada

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

De esta forma podremos generar un árbol con las instancias de tipo *Topic* relacionadas al objeto *Discipline* y *Course* actual.

Se ordenan los objetos *Topic* hermanos, basándose en la relación *LearningPath*
La relación *LearningPath* es aplicable entre dos objetos Topic en un mismo nivel de jerarquía, es decir entre hermanos. Esta representa el orden de aparición de los dos objetos al ser accedidos desde un LMS.

En base a este tipo de información existente en la ontología, definiremos para cada conjunto de hermanos del árbol de objetos *Topic*, el orden de aparición entre los mismos. Este orden dentro del árbol de organización de SCORM, queda determinado simplemente por el orden de aparición en la estructura XML del Organization.

Se realizan deducciones a partir de la relación *supportsTo*
Una vez obtenidos todos los objetos *Topic* por medio de la relación *isPartOf* es necesario encontrar los objetos *Support* que también deben ser agregados en el árbol de organización. Para esto se debe buscar sobre todos los objetos *Topic* encontrados en el paso anterior, cualquier objeto *Support* que tenga una relación *supportsTo* y este tenga una relación *isAvailableTo* con el *Course* actual.

Esta información debe ser representada agregando aristas y nodos sobre le árbol ya construido en el paso anterior. Es claro que este tipo de relacionamiento, por como se representa en la ontología de AdaptWeb, no podrá generar ciclos sobre el árbol ya que un objeto *Support* representa un ejercicio, ejemplo, complemento o *e-support* sobre el *Topic* relacionado.

Se termina de armar el árbol de organización
En este punto contamos con la representación arbórea de los objetos *Topic y Support* relacionados con el *Discipline y Course* actual.

Para generar el resto del árbol de organización, por cada objeto del tipo *Topic o Support*, generaremos un nodo Item en el árbol de organización, que hará referencia al Recurso que se ha generado anteriormente para ese objeto (más específicamente, en el paso 2 de la transformación específica).

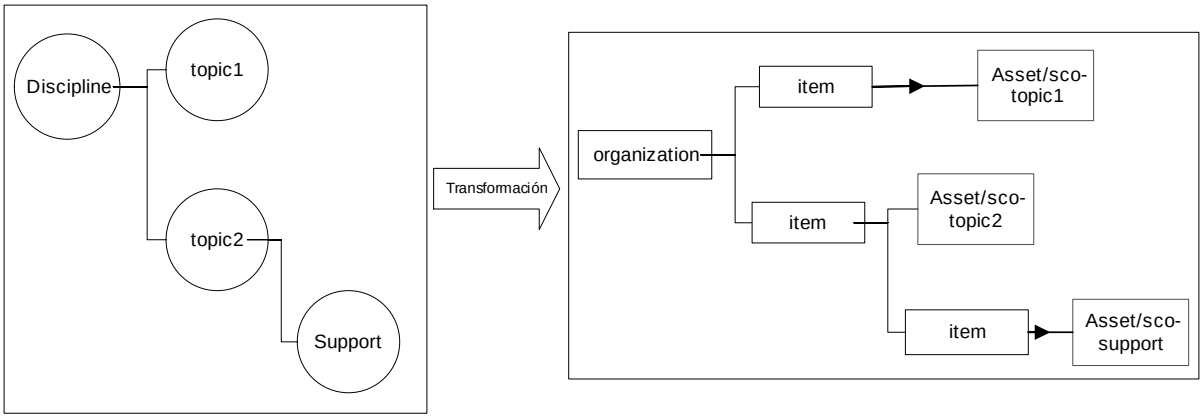


Figura 13: Creación del árbol de organización

Se agrega la información de *Prerequisite* en los *Content Organization* generados
Por medio de las relaciones *Prerequisite* existentes en el Modelo de Dominio de AdaptWeb, se incluye esta información en la descripción de los ítems del árbol de organización.

ii) Construir la instancia *Content Package* de SCORM para el objeto *Discipline*.

Construcción del Manifiesto

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

El manifiesto es un documento XML que contiene un inventario estructurado del contenido del paquete, a la vez que presenta información sobre su organización.
Los componentes relevantes de este documento son:

- Metadata (metadatos propios del archivo XML)
- Organizations (conjunto de árboles de organización)
- SubManifest
- Resources (Recursos del paquete junto con los metadatos LOM que lo define)

Sobre los pasos anteriores generamos resultados que simplemente se copiarán para generar adecuadamente las secciones antes mencionadas. En el paso numero 2, construimos completamente los *Resources* y en la sección 3.1, construimos los árboles de organización que incluiremos.
La sección de *SubManifest* no será utilizada ya que la decisión de construir un *Content Package* por *Discipline* con múltiples árboles de organización por *Course*, ya nos permite representar todo el contenido sin necesidad de aplicar este tipo de funcionalidad.

Construcción del Paquete
Como se presentó en la sección sobre SCORM, el *Content Package* es un archivo ".zip" el cual contiene el manifiesto (antes construido) y todos los archivos físicos necesarios.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

4 SOLUCION PROPUESTA

4.1 Descripción del prototipo

Este prototipo implementará la transformación específica detallada anteriormente, con la cual se genera contenido SCORM (*Asset*, *SCO*, *Content Organization* y *Content Package*) por medio de documentos DAML-AdaptWeb.
Ofrecerá un entorno de trabajo Web, en el cual, cualquier usuario autorizado, podrá generar contenido SCORM en el sistema para visualizarlo y persistirlo. De esta manera, se podrán reutilizar estos objetos en distintos *Content Organizations* y *Content Package* de forma de generar nuevos contenidos educacionales con características diferentes: objetivos, duración, complejidad, etc.

4.2 Diseño de la Arquitectura

La arquitectura propuesta para la implementación del prototipo, ha sido basada en el patrón *Model View Controller* (MVC) con algunas características particulares:

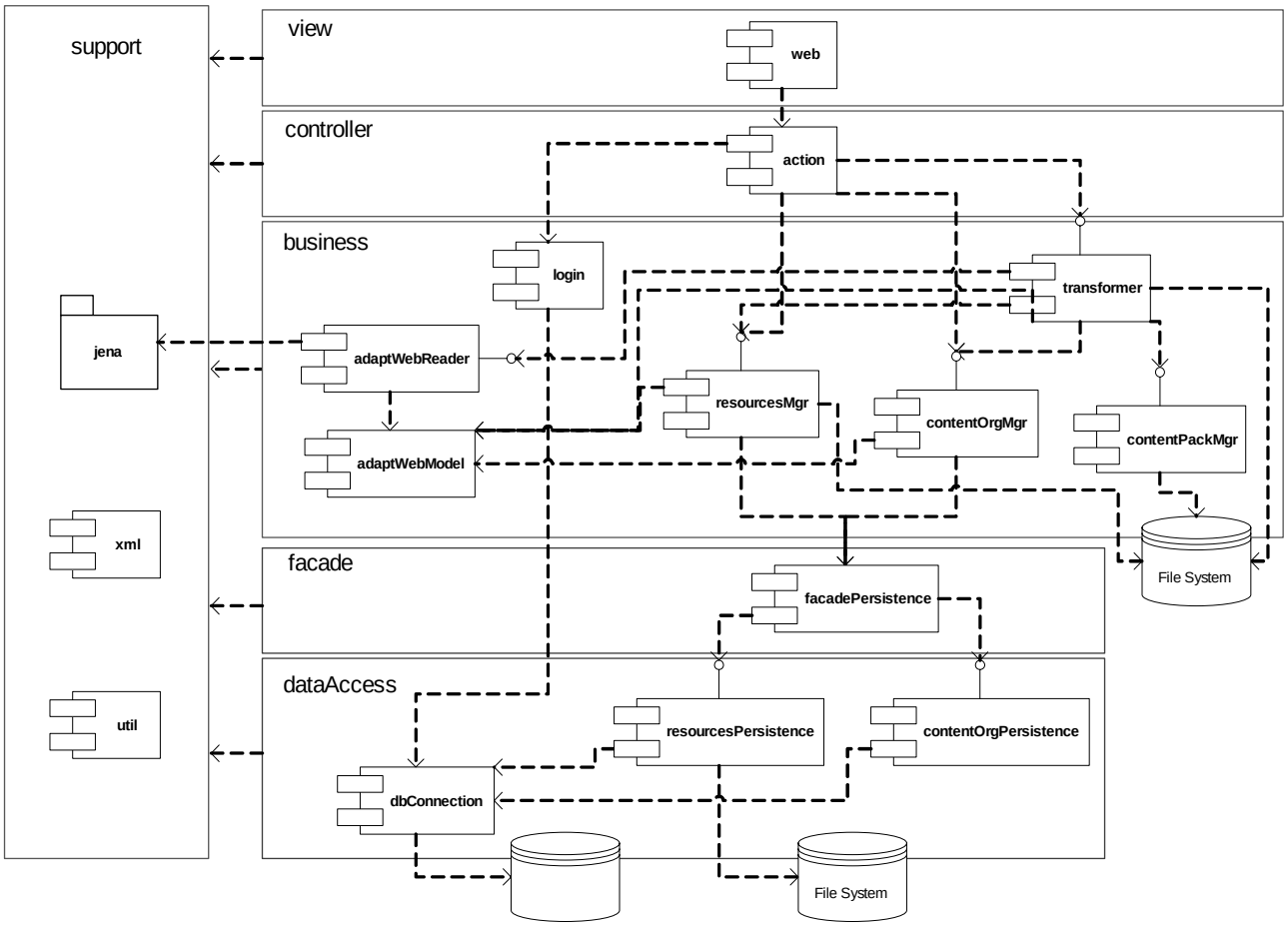
- En la misma se ha pretendido separar lo que es la transformación específica, de los componentes que manejan la información en formato SCORM. Con esta separación, se logró que la capa *dataAccess* pueda considerarse un potencial repositorio de elementos SCORM totalmente independiente de lo que son las capas *business*, *view y controller*, que sí constituyen el generador de paquetes de contenido educacional.

- Los componentes principales fueron diseñados utilizando interfaces con el propósito de poder cambiar fácilmente la implementación de distintas funcionalidades. Por ejemplo, si en el futuro se necesitara interpretar un documento con contenido educacional AdaptWeb basado en otro formato que no sea el DAML-AdaptWeb, bastaría con agregar una nueva implementación de la interface AdaptWebReader.

- La conexión a la base de datos a través de un componente especifico, permitiría poder cambiar el DBMS sin que ello impacte en la aplicación.

A continuación se presenta la arquitectura global de los componentes del prototipo, indicando con recuadros la separación en capas antes mencionada. Por más detalles sobre el diseño del prototipo, referirse al anexo: Documento de Arquitectura (como indicamos la ubicación al anexo).

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay



4.3 Consideraciones de los componentes

En esta sección se detalla algunos de los componentes fundamentales, así como aspectos importantes de la solución implementada.

4.3.1 AdaptWebReader

Este componente es el encargado de interpretar la estructura del archivo DAML-AdaptWeb y generar el modelo de datos (AdaptWebModel) con el cual el sistema podrá generar los objetos necesarios en SCORM.
Una característica importante es la utilización de la biblioteca JENA. Esta permite interpretar fácilmente archivos DAML, DAML+OIL, OWL, entre otros. Claramente se puede ver que si el lenguaje de definición de las instancias provenientes de AdaptWeb, es modificado, el efecto en el prototipo es mínimo, limitándose a re-escribir este componente pero utilizando gran parte de la implementación provista.

4.3.2 Transfomer

Este componente concentra todo el conocimiento y la lógica a utilizar en la transformación específica. Es el encargado de generar los objetos necesarios para representar el contenido educacional en SCORM. Para esto conoce toda la definición del Modelo de Dominio de AdaptWeb y basándose en este conocimiento implementa la transformación específica desarrollada en este trabajo.
Es por esto que cualquier cambio en la definición del Modelo de Dominio, reflejará un cambio importante en la implementación de este componente. Cabe recordar que la especialización de esta transformación y cada una de las tomas de decisión están fuertemente ligadas a la definición del Modelo de Dominio.

4.3.3 Repositorio

Este módulo, es una primera aproximación a un repositorio de contenido educacional SCORM. Aprovechando la necesidad de una capa de persistencia para almacenar el contenido generado en la transformación, consideramos que era importante separarlo a nivel arquitectónico de manera de implementar un prototipo básico de repositorio. Esta separación, permite, por un lado, abrir la puerta a trabajos futuros en materia de

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

repositorios de contenido SCORM, y a su vez, brinda la modularización suficiente para poder utilizar cualquier otro repositorio de SCORM existente, con tan solo cambiar el componente "Facade".

4.3.4 Adaptabilidad del sistema

A continuación se presentan distintos puntos en los que se le permite al usuario determinar algunas características de la transformación.

Criterio de transformación a Resources

Como fue descrito en el desarrollo de la transformación específica, tras identificar todas las instancias LOM existentes en el documento DAML-AdaptWeb, es necesario determinar adecuadamente la correspondencia de cada una de estas instancias a un tipo de *Resource* de SCORM (SCO/Asset). En la toma de esta decisión planteamos las siguientes opciones que quedan a consideración del usuario:

- *Opción por defecto*: Al construir los distintos árboles de tópicos existentes dentro de un *Content Organization*, todos los *Supports* son transformados a *Asset*, mientras que en el caso de los *Topic*, son *Asset*, únicamente las hojas de estos árboles que no posean relaciones con algún objeto *Support*.

- *Opciones configurables*: Si el usuario decide, no utilizar la opción por defecto puede configurar dos variables en la toma de decisión.

    - *Profundidad*: determina a partir de que nivel de profundidad, dentro de cada árbol de tópicos, estos serán transformados a *Asset*.

    - *Forzar criterio de profundidad*: fuerza a que solamente se tenga en cuenta la profundidad para determinar el tipo de transformación, sin tomar en cuenta el hecho de que tenga, o no, relaciones con elementos *Support*, ó sub-tópicos.

Elementos de Datos configurables

Un punto importante discutido en secciones previas y posteriormente en el anexo Correspondencias entre Metadatos SCORM y AdaptWeb, es el de manejar adecuadamente las correspondencias de metadatos entre SCORM y AdaptWeb. Sobre esta correspondencia se visualizaron problemas al determinar el valor en algunos de los atributos obligatorios de los objetos en SCORM. Para este punto se elaboró una página de configuración accesible luego de seleccionar los objetos de aprendizaje y antes de construir los objetos SCORM. En esta, el usuario puede determinar el valor correspondiente para cada uno de estos atributos, estos son: *catalog, cost y copyRightandOtherRestriction.*

4.4 Ambiente de Trabajo

Para el ambiente de trabajo se propone utilizar JAVA como lenguaje de programación en una arquitectura J2EE.

¿Por que utilizamos JAVA?

Si bien Java es simplemente un lenguaje de programación, existen en torno a él múltiples herramientas que facilitan y posibilitan el desarrollo de una aplicación, en particular una aplicación Web. Sobre muchas de estas herramientas se dispone el código (llamadas open-source), otras tantas son gratuitas y con variedad de licencias que posibilitan la utilización de las mismas para el desarrollo.

Algunos de estos tipos son: IDE (Interfaz Developer Editor), sistemas de Debug, servidores de aplicación (compatibles con la arquitectura J2EE), sistema de log, manejador de archivos XML, manejador de archivos DAML, entre tantos.

¿Por que utilizamos J2EE?

La estructura planteada por la arquitectura J2EE provee diferentes capas (cliente, presentación, lógica de la aplicación, sistemas integrados y persistencia) las cuales (si bien todas no se utilizarán) se adecuan completamente a los requerimientos planteados.

A su vez como se comentó en el punto anterior, se disponen sobre esta arquitectura, múltiples servidores gratuitos para realizar la publicación de estas aplicaciones.

4.5 Testeo y validación

Una vez creado el contenido en SCORM a partir de los documentos AdaptWeb-DAML, es necesario validar y testear dicho contenido. Básicamente utilizaremos dos tipos de herramientas para validar:

- **Editores de contenido SCORM**
  De forma de validar cualquier contenido de SCORM, tanto Asset, Sco, *Content Organization* y *Content Package*.
- **LMS**
  De manera de poder publicar los *Content Package* en algún servidor que brinde a los distintos tipos de usuarios acceso al contenido generado.

Sobre estos dos tipos de herramientas se realizaron evaluaciones y estudios comparativos. Las herramientas evaluadas fueron varias, teniendo la restricción de que la elección debía ser sobre una herramienta gratuita, algunas de las herramientas evaluadas fueron:

- LMS
  - Claroline ([Claroline)]
  - Moodle ([Moodle)]
  - DotLRN ([DotLRN)]

- Editores SCORM
  - Reload Editor ([ReloadEditor)]
  - Test Suite SCORM ([TestSuiteSCORM)]

Optando en cada caso por:

- **Moodle,** ya que es el único de ellos que implementa correctamente la versión 1.2 de SCORM
- **Reload Editor,** ya que no solo valida el contenido sino que deja manipularlo libremente, posibilitando la opción de previsualización del contenido generado.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

5 CONCLUSIONES

El trabajo presentado realiza un aporte en la investigación sobre la capacidad de transformar contenido educacional utilizado o generado por distintas plataformas de *e-Learning* a un estándar común. Si bien, en este trabajo no analizamos todas las posibles transformaciones, se intenta realizar un estudio sobre estándares muy utilizados hoy en día, como son, LOM y SCORM.

El análisis comienza con el estudio de la viabilidad de una transformación genérica entre estos modelos, investigando las posibles soluciones a los problemas encontrados. Para luego, mediante la especificación semántica dada por el Modelo de Dominio realizado por AdaptWeb, profundizar el estudio en una transformación específica. El desarrollo de esta transformación específica, nos brinda la posibilidad de validar su funcionalidad real mediante un prototipo, proporcionando además, un marco de trabajo, sobre el cual el usuario podrá reutilizar contenido desarrollado en AdaptWeb mediante SCORM.

5.1 Aportes

Transformación Genérica
En esta primera investigación se analizaron los problemas de correspondencias existentes entre LOM y SCORM. A su vez, se plantearon las distintas transformaciones que se pueden realizar entre los distintos tipos de contenido generables, presentando para cada una de ellas una metodología para llevarlas a cabo.

Transformación Específica
En una segunda instancia, se desarrolló una transformación completa de contenido educacional representado mediante el Modelo de Dominio AdaptWeb.

Prototipo de la Transformación Específica
Se diseñó e implementó un prototipo basado en la transformación específica antes mencionada. Las características fundamentales son:

- *Posibilidad real de transformar el contenido educacional AdaptWeb existente a SCORM*
  A partir de la definición de la ontología (archivo DAML) y los archivos utilizados como referencia en esta, se generan *Content Packages* que cumplan con la especificación SCORM.

- *Posibilitar al usuario configurar algunos criterios críticos de decisión*
  En los pasos críticos definidos en la transformación específica, se posibilita al usuario la selección de algún criterio para la generación de contenido en forma mas apropiada a las características del Diseñador de Contenido.

- *Primer aproximación a un repositorio de contenido SCORM*
  Los componentes SCORM (*Resources y Content Organizations*) generados en los pasos intermedios son persistidos. Posibilitando así generar consultas sobre los objetos creados o reutilizarlos en nuevos componentes.

- *Independencia de la transformación con respecto al repositorio*
  Su arquitectura posibilita modificar la capa de persistencia, de manera que el sistema no se vea afectado. Esto es ideal cuando se quiere utilizar otro tipo de repositorio, ya sea, desarrollado por terceros o bien, la elaboración de un nuevo repositorio con manejo más apropiado sobre los componentes.

- *Generación de nuevos contenidos SCORM a partir de datos existentes en el repositorio*
  Se agregaron funcionalidades adicionales para manipular el contenido generado en el repositorio, como modo de presentar el potencial de reutilización de contenido.

- *Comportamiento de la aplicación ante cambios.*
  Debido a la arquitectura en capas y al diseño por interfaces, modificar algún aspecto (que no repercuta en la arquitectura en si) tendrá su correspondiente en modificar algún componente del

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

sistema. Ejemplo de este comportamiento puede ser: cambiar el lenguaje de definición de objetos AdaptWeb (de DAML a OWL), modificar el repositorio, modificar la base de datos, etc.

- *Reutilización de componentes*
  Si bien este prototipo es un trabajo inicial en el área de transformación de especificaciones y estándares en el e-Learning, existen componentes que tienen una madurez tal, que pueden ser reutilizados en trabajos futuros.

5.2 Limitaciones y Trabajos Futuros

Al iniciar la investigación se plantearon muchas dudas e interrogantes, ya que no hay muchas personas especializadas en este tipo de transformación. Pero tras un largo análisis, diversas reuniones con nuestra tutora y distintas presentaciones realizadas (dos de ellas con otros investigadores del proyecto AdaptWeb y la última, una ponencia en el II Congreso de Enseñanza en Facultad de Ingeniería) pudimos ir validando distintos conceptos e ideas, que nos permiten creer haber alcanzado los objetivos planteados.

De todas formas, somos concientes que el trabajo presenta limitaciones y que es tan solo un primer paso para futuros trabajos en el área. A continuación brindamos detalles al respecto:

- **Criterios de Generación de *Resources***
  Si bien el prototipo brinda al usuario la posibilidad de elegir algunos criterios de generación, se podría ahondar más en esta área, de manera de que el diseñador de Contenido tenga un rol mas preponderante en la generación. Una posibilidad, es que para cada elemento a generar, se otorgue al diseñador de contenido, la facilidad de elegir que quiere construir.

- **Agregación de instancias LOM**
  En el prototipo desarrollado, no se permite la agregación de instancias LOM. Esto quiere decir que no se permite agrupar distintos objetos de aprendizaje, de manera de dar lugar a nuevos componentes más complejos. Si bien parece muy útil en la generación de contenido, se tendría que estudiar la viabilidad y utilidad de emplear este tipo de metodologías.

- **Repositorio de Contenido SCORM**
  El repositorio implementado, es una primera aproximación a este tipo de plataformas. Un trabajo importante para continuar en el área y colaborar aún más, en la reutilización, interoperabilidad, durabilidad y accesibilidad, es la implementación de un componente con mayores funcionalidades. El repositorio debería brindar al usuario, otro tipo de herramientas para la generación de contenido.

- **Interoperabilidad y accesibilidad al repositorio**
  Un área muy importante que no se desarrolló en la implementación del prototipo, es la de brindar un repositorio que permita interoperar con otros ya existentes, compartiendo el contenido.

- **Generación de nuevo contenido a partir de los datos persistidos**
  El prototipo brinda una serie de funcionalidades básicas, para la generación de nuevos contenidos a partir de los datos persistidos. Un trabajo futuro a desarrollar, puede ser la elaboración de una herramienta más compleja que brinde al usuario la capacidad de reutilizar los componentes con mayor facilidad. Un ejemplo, puede ser un editor de contenido (similar al Reload Editor) sobre contenido ya persistido en el repositorio.

6 ANEXOS

6.1 Glosario

AICC: (Aviation Industry CBT Comittee). Comité de formación por computador de la Industria de la Aviación. Asociación Internacional de profesionales relacionados con la formación basada en tecnologías que desarrolla líneas de acción de formación para la industria de la aviación.

ADL: (Advanced Distributed Learning Network). Iniciativa del Departamento de defensa estadounidense para conseguir interoperabilidad entre computadores y software de aprendizaje basado en Internet, a través del desarrollo de un marco técnico común que almacena el contenido en forma de objetos de aprendizaje reutilizables**.**

E-learning: Aquella actividad que utiliza de manera integrada y pertinente computadores y redes de comunicación, en la formación de un ambiente propicio para la construcción de la experiencia de aprendizaje2. Incluye la entrega de contenidos vía Internet, Extranet, Intranet, (LAN/WAN), audio y vídeo, emisión satelital, televisión interactiva y CD-ROM.

IEEE: (Institute of Electrical and Electronics Engineers). Instituto de Ingenieros Eléctricos y Electrónicos (USA).

IMS: (Instructional Management System). Sistema de gestión Instruccional, Consorcio de aprendizaje global. Coalición de organizaciones gubernamentales dedicadas a definir y distribuir especificaciones de interoperabilidad de arquitectura abierta para productos de e-learning**.**

LMS: (Learning Management System). Software que automatiza la administración de acciones de formación. Un LMS registra usuarios, organiza los diferentes cursos en un catálogo, almacena datos sobre los usuarios, también provee informes para la gestión. Un LMS es diseñado generalmente para ser utilizado por diferentes editores y proveedores. Generalmente no incluye posibilidades de autoría (creación de cursos propios), en su lugar, se centra en gestionar cursos creados por gran variedad de fuentes diferentes. Generalmente también se le conoce como plataforma.

LTSC: (Learning Technologies Standards Committee). Comité de la IEEE que tiene por objetivo desarrollar estándares técnicos, prácticas recomendadas y guías para la implementación informática de sistemas de formación a distancia.

LO: (Learning Object). Objetos de aprendizaje: Unidad reusable de información independiente de los medios. Bloque modular de contenido para e-learning.

Metadata: Información sobre el contenido, que permite almacenarla y recuperarla desde una base de datos.

SCORM: (Shareable Courseware Object Reference Model). Resultado de la iniciativa de Aprendizaje avanzado distribuido (ADL) del Departamento de Defensa Estadounidense. Los elementos de la plataforma de SCORM pueden ser combinados fácilmente con otros elementos compatibles para producir reposiciones altamente modulares de materiales de formación.

SCO: (Sharable Content Object). Objeto de aprendizaje compartible. Bloque modular de contenido para e-learning.

XML: (Extensible Markup Language): Lenguaje de codificación de última generación, que permite a los diseñadores Web programar sus propios comandos de marcación. Estos comandos podrán ser usados posteriormente como si fueran comandos HTML estándares .

6.2 Referencias

[LOM] – IEEE LTSC LOM (2001). Draft Standard for Learning Object Metadata, May, Final version 1.2. Disponible on-line: http://ltsc.ieee.org/wg12/index.html - Consultado en Diciembre-2004

[ADL] - ADL, Advanced Distributed Learning Initiative, US Department of Defense initiative, http://www.adlnet.org – Consultado en diciembre-2004

[ADLSCORM] - SCORM Sharable Content Object Reference Model. Ver [ADL]

[IMS] - IMS, (2002). Instructional Management System Global Learning Consortium, Disponible on-line: http://www.imsglobal.org/

[AICC] - AICC, Aviation Industry CBT Committee, http://www.aicc.org/ - Consultado en diciembre-2004

[IMSCP] - IMS CP (2001). Content Packaging Specification. Version 1.1.2. Final Release, August 2001. Disponible on-line: http://www.imsproject.org/packaging/index.html.

[Ko01] Koper, E.R.J. (2001): Modelling Units of Study from a Pedagogical Perspective: the Pedagogical Meta-model behind EML. Open University of Netherlands, input paper for IMS Learning Design group. Disponible on-line: http://eml.ou.nl/introduction/articles.htm

[XML] - eXtensible Markup Language (XML). (http://www.w3.org/XML/) Consultado en diciembre-2004.

[DAML] - DARP Agent Markup Language (DAML). (http://www.daml.org) Consultado en diciembre-2004.

[RDF] - Resource Description Framework (RDF). (http://www.w3.org/RDF/) Consultado en diciembre-2004.

[ARIADNE] - ARIADNE, 2001. ARIADNE Foundation for the European Knowledge Pool.
Disponible on-line: http://www.ariadne-eu.org/ Consultado en Diciembre-2004

[CANCORE] - CanCore, 2002. CanCore Application Profile.
Dispnible on-line:http://www.cancore.ca. Consultado en Diciembre-2004

[LSilva] - Tesis de Maestría de Lydia Silva - Año 2004
"Ontology-based Metadata for e-learning Content"

6.3 Perfil de Aplicación de los Metadatos SCORM

La tabla que se presenta a continuación proviene de uno de los documentos que definen SCORM: *SCORM Content Aggregation Model v1.2* [ADLCAMv1.2]. La misma toma como base los Elementos de Datos especificados en LOMv1.0 y plantea si son obligatorios (representado con "M" por *Mandatory* en inglés) ú opcionales (representado con "O") al definir en SCORM un *Content Aggregation*, un *SCO* o un *Asset* conformes a la versión 1.2 de SCORM.

En el caso de los identificadores, también aparece una R de Reservado, dado que al momento de definirse el SCORM v1.2 no existía un criterio único. En el SCORM 2004 estos campos dejaron de ser reservados, por lo que en la siguiente tabla a la derecha de la R, se incluye la nueva modalidad entre paréntesis.

Se presenta el perfil de aplicación de SCORM v1.2, en lugar del SCORM 2004 dado que la transformación especifica presentada en el proyecto transforma a SCORM v1.2.

Name	Content Aggregation	SCO	Asset
1.0 general	M	M	M
1.1 identifier	R(O)	R(M)	R(M)
1.2 title	M	M	M
1.3 catalogentry	M	M	O
1.3.1 catalog	M	M	O
1.3.2 entry	M	M	O
1.4 language	O	O	O
1.5 description	M	M	M
1.6 keyword	M	M	O
1.7 coverage	O	O	O
1.8 structure	O	O	O
1.9 aggregationlevel	O	O	O
2.0 lifecycle	M	M	O
2.1 version	M	M	O
2.2 status	M	M	O
2.3 contribute	O	O	O
2.3.1 role	O	O	O
2.3.2 centity	O	O	O
2.3.3 date	O	O	O
3.0 metametadata	M	M	M
3.1 identifier	R(O)	R(M)	R(M)
3.2 catalogentry	O	O	O
3.2.1 catalog	O	O	O
3.2.2 entry	O	O	O
3.3 contribute	O	O	O
3.3.1 role	O	O	O
3.3.2 centity	O	O	O
3.3.3 date	O	O	O
3.4 metadatascheme	M	M	M
3.5 language	O	O	O
4.0 technical	M	M	M
4.1 format	M	M	M
4.2 size	O	O	O

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

4.3 location	M	M	M
4.4 requirement	O	O	O
4.4.1 type	O	O	O
4.4.2 name	O	O	O
4.4.3 minimumversion	O	O	O
4.4.4 maximumversion	O	O	O
4.5 installationremarks	O	O	O
4.6 otherplatformrequirements	O	O	O
4.7 duration	O	O	O
5.0 educational	O	O	O
5.1 interactivitytype	O	O	O
5.2 learningresourcetype	O	O	O
5.3 interactivitylevel	O	O	O
5.4 semanticdensity	O	O	O
5.5 intendedenduserrole	O	O	O
5.6 context	O	O	O
5.7 typicalagerange	O	O	O
5.8 difficulty	O	O	O
5.9 typicallearningtime	O	O	O
5.10 description	O	O	O
5.11 language	O	O	O
6.0 rights	M	M	M
6.1 cost	M	M	M
6.2 copyrightsandotherrestrictions	M	M	M
6.3 description	O	O	O
7.0 relation	O	O	O
7.1 kind	O	O	O
7.2 resource	O	O	O
7.2.1 identifier	R(O)	R(O)	R(O)
7.2.2 description	O	O	O
7.2.3 catalogentry	O	O	O
7.2.3.1 catalog	O	O	O
7.2.3.2 entry	O	O	O
8.0 annotation	O	O	O

8.1 person	O	O	O
8.2 date	O	O	O
8.3 description	O	O	O
9.0 classification	M	M	O
9.1 purpose	M	M	O
9.2 taxonpath	O	O	O
9.2.1 source	O	O	O
9.2.2 taxon	O	O	O
9.2.2.1 id	O	O	O
9.2.2.2 entry	O	O	O
9.3 description	M	M	O
9.4 keyword	M	M	O

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

6.4 Especificaciones sobre la Categoría *Relation* de LOM

El siguiente diagrama, muestra el árbol que representa a la categoría *Relation* de la especificación LOMv1.0.

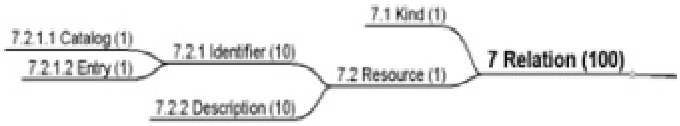


Figura 14: Categoría *Relation* de LOM [REHACER]

En la siguiente tabla, se presenta la trascripción de la definición del *LOMv1.0 Base Schema*, incluido en la sexta sección del *Final Draft Standard - IEEE 1484.12.1-2002* ***[REF!]***

Nro.	Nombre	Explicación	Tamaño	Orden	Espacio de Valores	Tipo de Datos
7	**Relation**	Define la relación entre el L.O. y otros L.O. si la hay	SPM: 100	Sin orden		
7.1	**Kind**	Naturaleza de la relación entre este LO y el LO objetivo identificado en 7.2:Relation.Resource	1	Sin orden	Basado en DublinCore ispartof : es parte de haspart : tiene parte isversionof: es version de hasversion : tiene version isformatof: es formato de hasformat: tiene formato references: referencia isreferencedby: es referenciado por isbasedon: está basado en isbasisfor: es base para requires: requiere isrequiredby: es requerido por	Vocabulario (State)
7.2	**Resource**	El L.O. objetivo que está referenciado por la relación.	1	Sin Orden		
7.2.1	**Identifier**	Una etiqueta única global que identifica el L.O. objetivo.	SPM: 10	Sin Orden		
7.2.1.1	**Catalog**	El nombre o designación de la identificación o el esquema de catalogación para esta entrada. Un namespace del esquema	1	Sin Orden	Repertorio de ISO/IEC 10646-1:2000	CharacterString (SPM:1000char)
7.2.1.2	**Entry**	El valor del identificador dentro de la identificación o el esquema de catalogación que designa o identifica el L.O. objetivo. Un string especifico con el namespace.	1	Sin Orden	Repertorio de ISO/IEC 10646-1:2000	CharacterString (SPM:1000char)
7.2.2	**Description**	Descripción del L.O. objetivo.	SPM: 10	Sin Orden		CharacterString (SPM:1000char)

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

6.5 Tipos de Relación utilizados en LOM

A continuación se brinda una descripción de cada uno de los tipos posibles que se especifican en el elemento de datos *7.1:Relation.Kind* de LOMv1.0.
Las descripciones fueron extraídas de *CanCore Guidelines: Relation Category* ***[REF!]***:

isversionof
El recurso descrito es una versión, edición o adaptación del recurso referenciado. Cambios en la versión implican cambios sustantivos en el contenido más que diferencias en el formato.

hasversion
El recurso descrito tiene una versión, edición o adaptación nombrada por el recurso referenciado.

isrequiredby
El recurso descrito es requerido por el recurso referenciado, ya sea física o lógicamente.

requires
El recurso descrito requiere el recurso referenciado para soportar su función, entrega o coherencia de contenido.

ispartof
El recurso descrito es una parte física o lógica del recurso referenciado.

haspart
El recurso descrito incluye al objeto referenciado ya sea física o lógicamente.

isreferencedby
El recurso descrito es referenciado, citado, o señalado de otra forma por el recurso referenciado.

references
El recurso descrito referencia, cita o señala de otra forma al recurso referenciado.

isformatof
El recurso descrito es el mismo contenido intelectual del recurso referenciado, pero presentado en otro formato.

hasformat
El recurso descrito pre-existe al recurso referenciado, que es esencialmente el mismo contenido intelectual presentado en otro formato.

isbasedon
El recurso descrito es derivado, completamente o en parte, del recurso referenciado. (Use este término para cualquier tipo de revisiones, correcciones, cambios, etc., que no sean adecuadamente cubiertos por *isversionof* o *isformatof*)

isbasisfor
El recurso referenciado es derivado, completamente o en parte, del recurso descrito. (Use este término para cualquier tipo de revisiones, correcciones, cambios, etc., que no sean adecuadamente cubiertos por *hasversion* o *hasformat*)

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

6.6 Notación utilizada en las transformaciones

Esta notación comprende una simple definición de elementos, sumado a una notación basada en la notación de expresiones regulares (de **¿??????**), para representar los distintos conjuntos de estos elementos.

Definición de los elementos	
LOM	Instancia de LOM (dejando sin especificar contenido de aprendizaje)
LOM(a)	Instancia de LOM cuyo contenido de aprendizaje es a
Asset	Asset del modelo SCORM (dejando sin especificar contenido de aprendizaje)
Asset(a)	Asset del modelo SCORM cuyo contenido de aprendizaje es a
Asset(a,b,…)	Asset del modelo SCORM cuyo contenido de aprendizaje esta compuesto por a,b, etc.
SCO	SCO del modelo SCORM (dejando sin especificar contenido de aprendizaje)
SCO(a)	SCO del modelo SCORM cuyo contenido de aprendizaje es a
SCO(a,b,…)	SCO del modelo SCORM cuyo contenido de aprendizaje esta compuesto por a,b, etc.
ContentPackage	Content Package del modelo SCORM (dejando sin especificar contenido de aprendizaje)
ContentPackage(a,b,…)	Content Package del modelo SCORM cuyo contenido de aprendizaje esta compuesto por a,b, etc.

Definición de conjuntos de los elementos (e) anteriormente definidos	
e	Un solo elemento representado por e
(e)+	Uno, o más, elementos del tipo e
(e)*	Cero, o más, elementos del tipo e
(e1)(e2)	El conjunto de elementos compuesto por e1 y e2 (donde e1 y e2 pueden representar cualquiera de las tres definiciones anteriores).
e1 e2	Ídem
e1 or e2	El conjunto de elementos compuesto por e1 o e2

Transformación	
A >>> B	La transformación del conjunto de origen A, al conjunto destino B

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

6.7 Perfil de aplicación de AdaptWeb

Las siguientes tablas extraídas del trabajo de Lydia Silva, describen las etapas que determinan el Perfil de Aplicación utilizado en el ambiente de AdaptWeb.

LOM Property	RDF Binding	Description	Conceptual Model Element
1.1- Identifier	lom-tech:location	A globally unique label that identifies this learning object	*location* attribute of *Learning Object* entity
1.1.1 Catalog	lom-gen:catalog	The cataloging scheme for the identifier	Fixed value "URI"
1.3- Language	dc:language using dcterms: RFC1766	Language used to communicate to the intended user.	*language* relation
1.4- Description	dc:description	Description of the content of the Learning Object.	*description* attribute of *LearningObject* entity
1.5- keyword	dc:subject	A keyword describing a topic in the Learning Object.	*hasKeyword* relation
2.3- Contribute	dc:creator and lom:Entity using Vcard:FN	Entity (people, organization) that contributed to the state of the Learning Object.	*creator* relation and *name* attribute of *Contributor* entity
3.3- Metadata Schema	lom-meta: metadataScheme	The specification used to create the metadata instance.	Fixed value "LOMv1.0"
4.4-Requirement	lom-tech: requirement	Technical capabilities necessary to use the Learning Object.	requires-NC relation
5.1- Interactivity Type	lom-edu: interactivityType	Predominant mode of learning of the Learning Object. Ex: expositive, interactive	interactivityType relation
5.2- Learning Resource Type	lom-edu: learningResource Type	Specific kind of Learning Object. Ex: Exercise, Example.	*learningType* attribute of *LearningObject* entity
5.8- Difficulty	lom-edu:difficulty	How hard it is to work with the Learning Object. Ex: Easy, Difficult.	difficultyDegree relation

Figura 15: Propiedades de LOM utilizadas en el perfil (extraído de [LSilva])

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

6.8 Modelo del Estudiante de AdaptWeb

El modelo del estudiante intenta caracterizar los elementos de un perfil de estudiante con objetivos específicos. La figura que se presenta a continuación describe el modelo conceptual del perfil del estudiante, el lenguaje utilizado es el de Entidad Relación (ER).

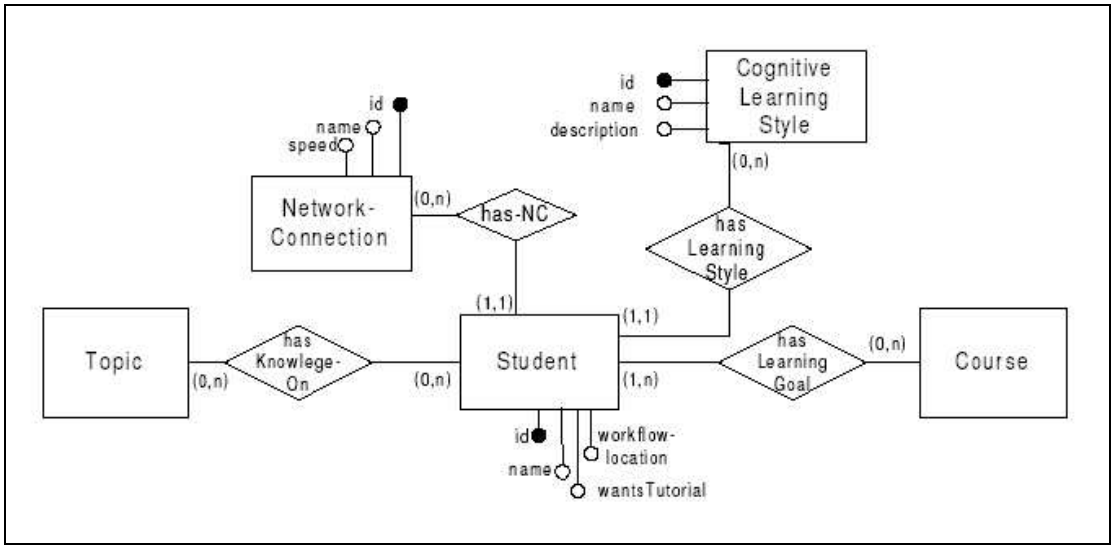


Figura 16: Modelo de Estudiante AdaptWeb (extraído de [LSilva])

El modelo del perfil del estudiante muestra a una entidad central *Student* con atributos básicos sobre el estudiante y relaciones adicionales que contribuyen a la caracterización del mismo.

El atributo *wantsTutorial* es utilizado para indicar si el estudiante prefiere un modo asistido para la realización de su curso.

La intención del estudiante por realizar un curso es representada por medio de la instancia de la relación *hasLearningGoal,* conectando al estudiante con el curso solicitado.

El conocimiento de un estudiante sobre un tópico del dominio, es modelado por la relación *hasKnowledgeOn* sobre una entidad curso*.*

Cada curso debe tener una instancia de esta relación si el estudiante tiene conocimiento sobre este tópico.

El tipo de conexión de red sobre un estudiante está declarado por la relación *hasNetworkConnection,* entre el estudiante y cualquier elemento de entidad *NetworkConnection.*

El atributo *workflowLocation* indica la URL en la cual el sistema define las preferencias e información sobre el estudiante.

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

6.9 Correspondencias entre Metadatos SCORM y AdaptWeb

Para la generación de los metadatos específicos que definen los elementos de SCORM (ya sea un Asset, un SCO o un Content Aggregation), realizamos un estudio de las correspondencias entre la especificación del Application Profile de SCORM con la información brindada por el Modelo de Dominio AdaptWeb.

A continuación se presenta una tabla que presenta sobre la izquierda el Application Profile de SCORM v1.2, con sus elementos opcionales y obligatorios , y sobre la derecha los elementos del Modelo de Dominio AdaptWeb que identificamos como sus correspondientes.

En algunos casos, surgen Elementos de Datos obligatorios en el Application Profile de SCORM v1.2, que no se corresponden directamente con un elemento del Modelo de Dominio de AdaptWeb. Estos casos se resaltan en rojo y especificando la decisión que tomamos al respecto.
Nomenclatura utilizada:

- LearningObject.attribute – indica el valor que tiene el atributo "attribute" del LearningObject
- Configurable – Este valor debe ser configurable y debe ser provisto por la aplicación que realice la transformación.
- "Texto" – palabra correspondiente al vocabulario definido en SCORM, por ejemplo: "Author"

SCORM v1.2					Learning Object (AdaptWeb)
Name	C. Aggregation	SCO	Asset		
1.0 general	M	M	M		
1.1 identifier	R(O)	R(M)	R(M)		LearningObject.id;
1.2 title	M	M	M		LearningObject.name (si el name no fue definido se utiliza id)
1.3 catalogentry	M	M	O		
1.3.1 catalog	M	M	O		Configurable
1.3.2 entry	M	M	O		Basado en los identificadores existentes
1.4 language	O	O	O		
1.5 description	M	M	M		LearningObject.description
1.6 keyword	M	M	O		LearningObject.keyWords
1.7 coverage	O	O	O		
1.8 structure	O	O	O		
1.9 aggregationlevel	O	O	O		
2.0 lifecycle	M	M	O		
2.1 version	M	M	O		1.0
2.2 status	M	M	O		"Unavailable" (no existe información para establecer otra correspondencia)
2.3 contribute	O	O	O		
2.3.1 role	O	O	O		"Author"
2.3.2 centity	O	O	O		contributor.id, contributor.name
2.3.3 date	O	O	O		
3.0 metametadata	M	M	M		
3.1 identifier	R(O)	R(M)	R(M)		LearningObject.id;
3.2 catalogentry	O	O	O		
3.2.1 catalog	O	O	O		
3.2.2 entry	O	O	O		
3.3 contribute	O	O	O		
3.3.1 role	O	O	O		"Creator"
3.3.2 centity	O	O	O		Identificador de la aplicación que realiza la transformación
3.3.3 date	O	O	O		
3.4 metadatascheme	M	M	M		LOMv1.0
3.5 language	O	O	O		

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay

4.0 technical	M	M	M		
4.1 format	M	M	M		(basado en la extención del archivo)
4.2 size	O	O	O		
4.3 location	M	M	M		LearningObject.location;
4.4 requirement	O	O	O		
4.4.1 type	O	O	O		
4.4.2 name	O	O	O		
4.4.3 minimumversion	O	O	O		
4.4.4 maximumversion	O	O	O		
4.5 installationremarks	O	O	O		
4.6 otherplatformrequirements	O	O	O		LearningObject.networkConnection(id,name,speed);
4.7 duration	O	O	O		
5.0 educational	O	O	O		
5.1 interactivitytype	O	O	O		LearningObject.interactivityType
5.2 learningresourcetype	O	O	O		LearningObject.learningResourceType
5.3 interactivitylevel	O	O	O		
5.4 semanticdensity	O	O	O		
5.5 intendedenduserrole	O	O	O		
5.6 context	O	O	O		
5.7 typicalagerange	O	O	O		
5.8 difficulty	O	O	O		LearningObject.dificulty
5.9 typicallearningtime	O	O	O		
5.10 description	O	O	O		
5.11 language	O	O	O		LearningObject.language
6.0 rights	M	M	M		
6.1 cost	M	M	M		Configurable
6.2 copyrightsandotherrestrictions	M	M	M		Configurable
6.3 description	O	O	O		
7.0 relation	O	O	O		
7.1 kind	O	O	O		
7.2 resource	O	O	O		
7.2.1 identifier	R (O)	R (O)	R (O)		
7.2.2 description	O	O	O		
7.2.3 catalogentry	O	O	O		
7.2.3.1 catalog	O	O	O		
7.2.3.2 entry	O	O	O		
8.0 annotation	O	O	O		
8.1 person	O	O	O		
8.2 date	O	O	O		
8.3 description	O	O	O		
9.0 classification	M	M	O		
9.1 purpose	M	M	O		"Educational Objective"
9.2 taxonpath	O	O	O		
9.2.1 source	O	O	O		
9.2.2 taxon	O	O	O		
9.2.2.1 id	O	O	O		
9.2.2.2 entry	O	O	O		
9.3 description	M	M	O		LearningObject.description
9.4 keyword	M	M	O		LearningObject.keyWords

Autores: Martín Cabrera, Sergio Pérez / Tutor: Dr. Regina Motz
Facultad de Ingeniería – Universidad de la República – Montevideo, Uruguay