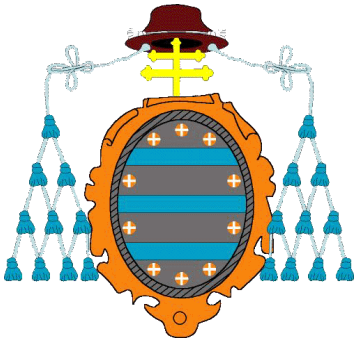
UNIVERSIDAD DE OVIEDO

Departamento de Informática



TESIS DOCTORAL

"MODELADO ESPECIFICO DE DOMINIO PARA LA CONTRUCCION DE LEARNING OBJECTS INDEPENDIENTES DE LA PLATAFORMA"

Presentado por

Carlos Enrique Montenegro Marín

Para la obtención del titulo de Doctor en Informática

Dirigido por

Doctor D. Juan Manuel Cueva Lovelle

Doctor D. Oscar Sanjuán Martínez

Oviedo, 2011

Agradecimientos

Primero que todo agradecer a las dos instituciones que hicieron posible la realización de este trabajo; La Universidad Distrital “Francisco José de Caldas” y la Universidad de Oviedo, la primera como alma mater formadora que siempre ha apoyado mis esfuerzos, y la segunda por haberme acogido en su corazón y ofrecerme su más valioso tesoro “la sabiduría en pro de este trabajo”.

Es para mí muy importante resaltar y agradecer al director del proyecto Dr. D. Juan Manuel Cueva Lovelle y al codirector Dr. D. Oscar Sanjuán Martínez. Con su vasta experiencia y conocimiento han logrado dirigir como lo haría el más experto de los capitanes de navegación, un largo y exitoso viaje, que hoy culmina.

También es necesario plasmar el agradecimiento por sus valiosos aportes a Dr. Dña. Cristina Pelayo, a Dr. D. Vicente García y a todo el grupo de investigación mda-oviedo3, de la Universidad de Oviedo. Sin sus exploraciones esto no hubiese sido posible.

Gracias al magnífico grupo de profesores adscritos al Departamento de Informática de la Universidad de Oviedo, haber podido compartir con ustedes me ha enseñado una muy valiosa lección de amistad.

Son muchos los nombres que se me escapan, pero no por ello son menos importantes, así que pido me perdonen pues siempre hay alguien que se me olvida. Bety, Víctor, Nelson, JJ, Magdalena, Lilian, Julio, Susana, Alejandro, Marcela, Alonso, Cesar, Javier, Giovanny, Juan Carlos, John y a todos los demás gracias en Colombia. Edward, Chema, Weena, Pablo, Jordan, Miguel, Manuel, Francisco, Omar, Juanjo, Hernán, Gillermo y a los que se me escapan también, muchas gracias por su ayuda en Oviedo.

Por último no puedo dejar de lado a mi familia; Mama, Papa, Abuelita, Hermano y claro Karencita mi cuñada, Tíos, Tías, Primos y claro tú también mi hermosa princesa Jenny Milena.

Gracias a Todos.

Tabla De Contenido

Lista De Figuras

Lista De Tablas

Lista De Códigos Fuente

1 Datos del proyecto de tesis

TÍTULO: Modelado especifico de dominio para la contruccion de learning objects independientes de la plataforma.

AUTOR: D. Carlos Enrique Montenegro Marín

DIRECTORES: Dr. D. Juan Manuel Cueva Lovelle

Dr D. Oscar Sanjuán Martínez

1.1 Resumen

El objetivo general del proyecto es realizar el modelado específico de dominio para la construcción de módulos de sistemas de gestión del aprendizaje (LMS) independientes de la plataforma. Para ello primero se generará una Ontología que permita modelar módulos de cursos, después se utilizará la Ontología generada para crear un metamodelo y luego un DSL basado en el, finalmente con MDE y aplicando las debidas transformaciones se logrará desde un modelo Independiente de la plataforma, un módulo desplegable en un LMS construido bajo un estándar.

Abstract

The overall project goal is to make the domain-specific modeling for the construction of modules of learning management systems (LMS) platform independent. For this, first generate an ontology for modeling modules course, then use the generated ontology to create a metamodel and then create a domain-specific language. Finally with MDE and applying the appropriate transformations are achieved from a platform independent model, a module built under a standard LMS.

1.2 Motivación

Esta propuesta enfrentara el problema de la interoperabilidad entre LMSs, a partir de la generación de una ontología para la posterior creación de un metamodelo. La generación de la ontología ya se ha tratado de hacer en proyectos como (Dzemydiene et al., 2007) en el cual se analiza la posibilidad de integrar ontologías en los LMSs para mejorar las búsquedas en ellos, aquí se pretende emplearla para modelar módulos dentro de un LMS, que sean desplegables en cualquier plataforma LMS, una propuesta similar pero desde otro enfoque se trata en (Bizoňová et al., 2007, Bizonova and Ranc, 2007, Bizonova and Ranc, 2008), las cuales facilitan una especificación integrada de las arquitecturas de diferentes plataformas LMS. Este framework independiente de la plataforma se puede utilizar para especificar y clasificar los LMSs existentes o futuras gestiones de ellos, para simplificar la migración de datos entre diferentes tipos de sistemas e-Learning, los artículos dejan abierta la problemática existente entre la multiplicidad de plataformas. En el trabajo (Grob et al., 2010) se muestra el gran problema de heterogeneidad, por la utilización de diversos lenguajes de programación sobre los cuales están hechos los LMSs y en (Moreno and Romero, 2005b) hablan sobre la falta de integración entre los LMSs existentes a pesar de sus funcionalidades similares.

Los trabajos (Perniu et al., 2006, Dzemydiene et al., 2007, Bizoňová et al., 2007, Bizonova and Ranc, 2007, Bizonova and Ranc, 2008, Grob et al., 2010, Moreno and Romero, 2005b) han tratado la problemática de la interoperabilidad entre LMSs, otras propuesta como (Muñoz-Merino et al., 2009) plantea la solución a este problema con la generación de servicios, y evidencian que esta característica la poseerán las próxima generación de LMSs, pero que los actuales no poseen, en (Bizoňová et al., 2007, Bizonova and Ranc, 2007, Bizonova and Ranc, 2008) se plantea la creación de un framework que integre la especificación de una arquitectura general para los LMSs y utilice MDA para migrar de una plataforma a otra, las pruebas se realizaron sobre las plataformas OLAT y Moodle, pero únicamente trataron la parte de administración de objetos de aprendizaje y derechos de autor, la mejor aproximación que se encontró a lo que se plantea en esta investigación, se halló en (Dodero et al., 2010) allí se presenta un lenguaje visual y de texto, especifico para la creación de diseños de aprendizaje independientes de la plataforma, algunas otras aplicaciones relacionadas con la temática son CDF(Verbert and Duval, 2004), LMML(Slavin, 1995), PALO(Rodriguez-Artacho et al., 1999), Targetam(Michael, 2002), TML/Netquest(Brickley, 1996), IMS Learning Design (IMS LD)(Consortium, 2003b) pero estas están no consideran los módulos que posee un LMS. La conclusión de esta indagación es que la solución a la problemática se ha atacado desde varios frentes, algunas desde la perspectiva de MDA pero ninguna contempla los módulos actuales que posee un LMS.

Para poder abordar esta problemática, se plantea el diseño y creación de una ontología LMS, trabajos como (Heiyanthuduwage and Karunaratne, 2006, Srimathi, 2010) en los cuales se presenta una ontología muy genérica para LMSs ó (Díaz-Antón and Pérez, 2005, Díaz-Antón and Pérez, 2006) en los que muestra una ontología para integrar e implementar un LMS en una organización ó propuestas trabajadas en (Amorim et al., 2006) que presentan una ontología un poco más completa pero enfocada en los niveles B de las especificaciones dadas IMS LD (que son las son las propiedades y las condiciones, las primeras son pares atributo-valor que parten de un estado inicial y se modifican a lo largo del proceso de ejecución de la unidad de aprendizaje y las segundas son consultas que se realizan sobre el valor de las propiedades en un momento determinado (Consortium, 2003b)); estos modelos pueden ser tomados como base pero ninguno trata los módulos que componen un LMS. Posteriormente se creará un metamodelo con la ontología obtenida, trabajos como (Boticario and Santos, 2007) lo han hecho pero nuevamente enfocados a niveles B del IMS LD, con lo cual y al igual que con las ontologías, ninguna de las propuestas suple las necesidades de este proyecto, asi que es necesaria la creación de este metamodelo para módulos que posee un LMS.

1.3 Hipótesis

Es posible aumentar el nivel de abstracción para obtener un framework independiente de la plataforma, con el cual se pueda modelar un LMS para su posterior despliegue en una plataforma especifica.

Esto se logrará mediante los principios propuestos por la Ingeniería Dirigida por Modelos (MDE) y específicamente la propuesta de la OMG que es la Arquitectura Dirigida por Modelos (MDA), lo que posibilitará minimizar el tiempo y esfuerzo necesarios para la creación de módulos para un LMS.

1.4 Objetivo

Realizar el modelado específico de dominio para la construcción de Módulos de LMSs independientes de la plataforma.

1.5 Objetivos específicos

- Seleccionar una muestra representativa de LMSs, modelar sus módulos y determinar cuáles son comunes.
- Generar una Ontología que permita modelar las características comunes entre los LMSs.
- Utilizar la Ontología generada para crear un Metamodelo de los LMSs.
- Crear un DSL Grafico acorde al metamodelo planteado.
- Con MDE y aplicando las debidas transformaciones lograr desde un modelo Independiente de la plataforma un módulo desplegable en un LMS.

1.6 Plan de trabajo

1. Recopilación de información. Generar una base documental estandarizada que soporte la elaboración del proyecto, en las siguientes áreas:

    - Estándares, ahondando en la representación de contenidos para plataformas Virtuales de aprendizaje LMS.

    - Como están construidos (implementados) los contenidos en algunos LMS de uso libre.

    - Modelamiento de las plataformas LMS, Ontología y construcción de un metamodelo LMS.

    - Construcción de un DSL Grafico basado en el metamodelo para la creación de módulos en LMS aplicando MDE.

2. Desarrollo y Elaboración de la propuesta. Una vez realizada toda la base documental se procederá al desarrollo y elaboración de la propuesta en el siguiente orden:

- Modelado y Construcción de una Ontología que represente los contenidos de los más relevantes LMSs de uso libre que estén construidos bajo algún estándar.

- Generación de un metamodelo y un DSL Grafico asistido por la ontología obtenida con anterioridad.

- Empleando MDA y diversas transformaciones, lograr que a través del DSL generado se pueda desplegar un contenido para un LMS de uso libre construido bajo algún estándar.

3. Despliegue de la propuesta: Construir los modulos para un LMS de libre uso que este cimentado bajo algún estándar, empleando el DSL anteriormente creado y sus debidas transformaciones, este contenido deberá ser desplegado sobre el LMS.

4. Pruebas del contenido: Probar y validar el funcionamiento de los módulos desplegado en el LMS con la herramienta DSL.

5. Difusión de resultados: la fase de difusión de resultados se realizara de manera continuada durante todo el proyecto; consistirá en acudir a congresos internacionales, reuniones con miembros de grupos de investigación externos y escribir artículos para revistas de investigación especialistas en el sector. Los contenidos difundidos serán sobre los avances que se vayan consiguiendo por etapas. Esta fase tendrá una especial importancia en el final del proyecto

donde puedan difundirse de manera completa los resultados globales de la investigación, estos artículos deberán referenciar trabajos realizado en el grupo que son la base de esta propuesta.

6. Finalización y defensa de la tesis doctoral: Durante toda las fases de la investigación se irán recopilando y clasificando información y resultados obtenidos, con el objetivo de completar la tesis doctoral, una vez llegados a este punto se trabajara a fondo en el documento de la tesis y la preparación de su defensa, para posteriormente su defensa en el tribunal del departamento de informática de la universidad de Oviedo.

2 Antecedentes y estado actual del tema

2.1 Introducción

MDE (Model Driven Engineering) o Ingeniería Basada en Modelos es una propuesta que plantea el uso de modelos como eje fundamental en todo el ciclo de vida de un proyecto de software, una de las aproximaciones de este planteamiento es MDA (Model Driven Architecture) que es la proposición de la OMG (Object Manage Group) la cual establece una serie de tecnologías a utilizar en la construcción de software bajo el esquema de MDE, la idea fundamental de MDA se basada en la transformación de modelos para ir de lo mas general (Requerimientos) a lo mas particular (Despliegue de la Solución) a través de las conversiones de sus modelos. (García, 2010, Garcia-Diaz et al., 2009, Zdun, 2010)

Una buena propuesta para hacer estos modelos es DSL (Domain Specific Languages) y en el trabajo "Diseño y construcciones de lenguajes específicos de dominio asistido por ontologías" (Alvarez, 2010) propone una forma de hacerlo a partir de Ontologías.

Existen estándares que definen como se deben implementar los diferentes módulos para un LMS (Learning Management System). Algunos LMS de libre distribución como MOODLE, CLAROLINE, ZAKAI, .DOTLRN, y ATUTOR, en la documentación oficial de estas herramientas se asevera que están construidas bajo algún estándar, pero al momento de realizar una migración de un curso hecho en un plataforma LMS especifica a otra, surgen problemas de compatibilidad, la posible causa de este problema es que cada una de las plataformas está construida bajo un estándar diferente o si están bajo el mismo estándar cada una hace su implementación (interpretación) particular causando problemas de compatibilidad entre plataformas, como lo evidencian en (Perniu et al., 2006) donde se pretende realizar un editor de SCO (Sharable Content Objects) bajo entorno Eclipse con el fin de poder desplegar un SCO en cualquier LMS.

Esta propuesta atacara el problema generando una Ontología para el posterior modelamiento del contenido estandarizado, La generación de la Ontología ya se ha tratado de hacer en proyectos como (Dzemydiene et al., 2007) en el cual se analiza la posibilidad de integrar Ontologías en los LMS para mejorar las búsquedas en ellos, aquí pretendemos emplearla para modelar posteriormente el contenido y que sea desplegable en cualquier LMS estandarizado, una propuesta similar pero desde otro enfoque se trata en (Bizoňová et al., 2007, Bizonova and Ranc, 2007, Bizonova and Ranc, 2008) que busca como objetivo proporcionar un marco general que permita una especificación integrada de las arquitecturas de diferentes plataformas. Este Framework independiente de la plataforma se puede utilizar para especificar y clasificar los LMS existentes o futuras gestiones de ellos para simplificar la migración de datos entre diferentes tipos de sistemas e-Learning, el articulo deja abierta la problemática existente entre la multiplicidad de plataformas. En el trabajo (Grob et al.,

2010) se muestra el gran problema de heterogeneidad por la utilización de diversos lenguajes de programación sobre los cuales están hechos los LMS y en (Moreno and Romero, 2005b) hablan la falta de integración entre los LMS existentes a pesar de sus funcionalidades similares.

Es por lo anterior que se pretende realizar el modelado específico de dominio para la construcción de módulos para plataformas LMS independientes de la plataforma.

2.2 Estudio de interoperabilidad entre los sistemas de gestión del aprendizaje LMS

En el siguiente Capitulo se realiza un estudio detallado para analizar desde el punto de vista informático la evolución que han tenido las Plataformas Virtuales de Aprendizaje en torno al tema de interoperabilidad entre ellas, seleccionando de esta manera las más representativas dentro del ámbito académico bajo licencia GNU con el ánimo de identificar como trabajan los mecanismos.

Para la selección de las plataformas virtuales de aprendizaje de estudio se tomaron como referentes los trabajos de investigación doctorales realizados por (Márquez, 2007) para la definición de una plataforma de aprendizaje universal, abierta y escalable bajo enfoques sociales, políticos, económicos y educativos, con miras a identificar desde estos puntos de vista herramientas que han evolucionado de la mano en estos contextos, al igual que desde el punto de vista metodológico el trabajo realizado por (Vélez R, 2007) para identificación de mecanismos que logren apoyar el proceso de aprendizaje de estudiantes en estos ambientes de trabajo. De la misma manera se tomo en cuenta los reportes realizados por el grupo (SIGOSSEE, 2010) SIG Open-Source Software for Education in Europe, el cuál es un proyecto financiado por el programa e-learning de la comisión Europea en consorcio con varios grupos de investigación y universidades europeas, los cuáles presentan estudios de funcionalidad de plataformas libres a nivel informático, mediante una metodología creada por este proyecto y cobijada por la ISO IEC 9126, identificando patrones de evaluación mediante elementos como: funcionalidad, usabilidad, rehúso, eficiencia, mantenibilidad y portabilidad, al igual, del análisis de plataformas que logren acoplarse a los desarrollos de estándares y especificaciones de contenidos que es un tema de suma importancia para el objetivo que se desea estudiar.

Desde el punto de vista de funcionalidades, vale la pena rescatar los trabajos realizados por (Llorente-Cejudo, 2007) y desde el punto comparativo de plataformas los realizados por (EduTools, 2010), los cuáles presentaron factores importantes que lograron junto con los anteriores referentes trabajar desde el punto de vista de arquitectura de módulos permitiendo definir las siguientes plataformas de estudio: .DOTLRN, SAKAI, ATUTOR, CLAROLINE y MOODLE ,las cuáles para efectos de trabajo colaborativo según el grupo (SIGOSSEE, 2010) representan un buen grado de aceptación y una buena valoración en la mayoría de comunidades y grupos dedicados

a trabajar temas relacionados con la gestión de contenidos y lógicamente bajo estrategias de aprendizaje virtual.

2.2.1 Realidad sobre plataformas virtuales de aprendizaje

Desde el punto de vista de regulación del desarrollo de contenidos de aprendizaje que será un tema que se abordará más adelante, las entidades y organizaciones encargadas de estandarizar los parámetros para proporcionar reuso de los mismos sobre diferentes tipos de plataformas, solamente se han dedicado a facilitar un modelo para la reutilización de recursos de aprendizaje, por lo que en temas de garantizar el uso de estos recursos y una forma de crearlos genérica e integrada (independiente de la plataforma) ha sido una actividad que todavía no ha generado una iniciativa fuerte para empezar a desarrollar alternativas que de manera consistente aseguren en cierta medida los contenidos que se generan bajo sus estándares y por ende sus especificaciones.

Para realizar un estudio sobre plataformas virtuales de aprendizaje, el análisis se ha inclinado sobre plataformas de uso libre GNU, debido a su flexibilidad en analizar variables de código y sus mismos lenguajes y especificaciones, en tanto, las plataformas LMS de uso comercial presentan una característica que aparte de ser demasiado costosas para su adquisición, no ofrecen un ambiente de trabajo representativo para su análisis desde el punto de vista de lenguajes y especificaciones propias de cada plataforma, esto debido a políticas propias de las mismas empresas encargadas de su mantenimiento y adecuación. En algunas instituciones educativas alrededor del mundo(SIGOSSEE, 2010), lo que se trata de masificar en cierta medida en este estudio es contribuir al desarrollo de mecanismos y alternativas de uso libre y apoyar la idea de muchas comunidades de usuarios a nivel académicos e investigativos bajo el mismo propósito trabajado por (Márquez, 2007) en su propuesta de investigación Doctoral para la integración de herramientas e-learning y definición de una plataforma universal, abierta y escalable, viable para toda la comunidad académica.

Para apoyar los anteriores estudios, a nivel de desarrollo de proyectos de uso libre, se realizó un estudio de proyectos asociados a desarrollos enfocados en aspectos CMS y LMS, llegando a encontrar 3011 proyectos relacionados con características CMS y 88 Proyectos relacionados con LMS, a través del portal (Geeknet, 2010) en Internet.

Basados en los anteriores estudios, a continuación se tomarán como referencia Cinco (5) plataformas de aprendizaje de uso libre GNU, DOTLRN, SAKAI, ATUTOR, CLAROLINE y MOODLE, los cuáles desde su concepción han logrado captar la atención de muchas comunidades a nivel académico e investigativo, lo que ha generado una gran aceptación en la mayoría de instituciones académicas para el uso y masificación de estrategias que logren apoyar sus procesos de formación.

2.2.2 Aspectos relevantes para el desarrollo de los sistemas de gestión del aprendizaje

Los LMSs (Learning Management System) en cierto sentido han sido la evolución de los sistemas gestores de contenidos CMSs creados específicamente para la administración y control de información de usuarios a nivel corporativo. Desde su aparición ha sido producto de integración y creación de diferentes tipos de comités y grupos que a nivel mundial han permitido apoyar esta iniciativa para tratar de estandarizar los procesos de desarrollo de plataformas que se prestan para su uso en particular de formación y que por tanto según (Boneu, 2007) a representado un factor determinante para facilitar metodologías de estudio diferentes a las tradicionales basadas en entrenamiento CBT (Computer Based Training), IBT (Internet Based Training) y WBT (Web Based Training).

Dentro de los estudios más representativos que dieron lugar a estas especificaciones parten de iniciativas que en el caso particular a nivel europeo el grupo JOIN (SIGOSSEE, 2010) mediante su iniciativa para regular elementos que definieran características de una plataforma e-learning. Se apoyo bajo el estándar ISO/IEC 9126 para la definición de métricas y parámetros mínimos para el desarrollo de este tipo de plataformas, y esfuerzos de trabajo colaborativo entre UNESCO y la FSF (UNESCO, 2005) como una iniciativa para el uso de herramientas Open Source dentro de ámbitos académicos; al igual que aportes representativos por parte de grupos en el ámbito académico e investigativo como (Edutech, 2000, Commonwealth of Learning (COL), 2003), y la (CENT Universitat Jaume I, 2004).

A continuación se resaltara en este punto los resultados dados por el grupo (SIGOSSEE, 2010) para la definición de métricas que permitan la construcción de un LMS:

1. El sistema debe ser de código abierto.
2. Debe ser accesible a través de un navegador Web estándar.
3. Las opciones de autoría así como el resto de funciones del sistema deben poder ser utilizadas sin la necesidad de comprar ningún plugin o visualizador adicional.
4. Debe existir funciones básicas para la administración de usuarios.
5. El sistema debe ofrecer una función de autenticación.
6. El sistema debe ofrecer gestión de permisos.
7. El sistema debe estar abierto a la localización.
8. El alumno debe poder interaccionar a través del navegador con el profesor, el sistema y otros alumnos. La comunicación debe poder ser electrónica.

9. Debe existir funciones básicas para la evaluación y progreso de los alumnos y funciones básicas para al menos la autoría de test y evaluaciones.

10. Debe existir funciones para la gestión de cursos.

11. Debe existir funciones para la gestión de contenidos.

2.2.3 Sistemas de gestión del aprendizaje seleccionados para modelar

Sakai

Este proyecto se originó en la Universidad de Michigan y en la Universidad de Indiana, a las que luego se unieron el MIT y Stanford University, junto otras entidades y fundaciones en el año 2004. Es un LMS, el cual desde un principio ha trabajado bajo estándares IMS, lo cual lo pone en una posición favorable con respecto a otros LMS, esto se debe a que una parte de los impulsadores de Sakai son las entidades promotoras de estos estándares. Sakai provee una estructura modular construida bajo especificaciones de código Java cuyo objetivo es integrar funcionalidades de formación y comunicación.

ATutor

Esta plataforma empezó como una iniciativa de la Universidad de Toronto en el año 2002 en su centro de estudios ATRC (Adaptive Technology Resource Centre). Según estudios realizados por el grupo (SIGOSSEE, 2010),el desarrollo de esta plataforma ha prestado especial interés a la accesibilidad, llegando a ser el único LMS que cumple las especificaciones de accesibilidad encomendadas por la W3C mediante la norma WCAG 1.0 de nivel AA+, el cual indica la capacidad de la plataforma de permitir el ingreso a personas discapacitadas.

Claroline

Fue un proyecto auspiciado por la Universidad de Louvain en Bélgica en el año 2000. Se considera un proyecto orientado para el trabajo bajo el enfoque de formación mediante estrategia e-learning y de trabajo colaborativo mediante estrategia e-working al presentarse también como un sistema gestor de contenidos CMS.

Según resultados obtenidos de parte del grupo (SIGOSSEE, 2010) muchas universidades aprecian su ambiente de aprendizaje colaborativo que permite a los docentes y a las instituciones educativas crear y administrar cursos en la Web. Las herramientas que ofrece el sistema son muchas y dan a los usuarios la posibilidad de establecer una variedad de escenarios de aprendizaje.

Moodle

Este proyecto es uno de los que ha tenido mayor acogida en la mayoría de comunidades académicas e investigativas a nivel mundial. Al ser uno de los primeros proyectos bajo iniciativo GNU mediante su creador Martin Dougiamas. Alrededor de esta corriente su acogida fue tal, que desde un principio se empezaron a adaptar metodologías desde el punto de vista pedagógico, trabaja el constructivismo social por lo que no sigue otras recomendaciones. Adopta estándares de accesibilidad y uso.

De manera estructural es una plataforma que posee todas las características válidas para el desarrollo de componentes que permiten una interoperabilidad con funcionalidades a diferentes niveles de trabajo. Sus mismas siglas lo describen como un Ambiente de Aprendizaje Dinámico Orientado a Objetos Modular.

DotLRN

DotLRN es una plataforma virtual de aprendizaje (e-learning) que facilita la colaboración y gestión de clases a través de Internet. Está basado en la plataforma de gestión de comunidades virtuales OpenACS. Es una plataforma potente, escalable y flexible que puede soportar un uso fuerte por parte de los usuarios. Es una plataforma tecnológica que está siendo utilizada por universidades principalmente y que está auspiciada por uno de los principales centros tecnológicos del mundo, el MIT (Instituto Tecnológico de Massachutets).

2.2.4 Estándares soportados por los sistemas de gestión del aprendizaje seleccionados

Para identificar de manera general los mecanismos de funcionamiento de cada plataforma, la siguiente tabla muestra un estudio producto del análisis encontrado en los portales y documentación de cada uno de los proyectos, presenta una relación sobre las especificaciones a nivel de contenidos soportados por cada una de ellas.

Tabla 1 Resumen de mecanismos de seguridad y especificaciones de contenidos usadas por plataformas analizadas.

Plataforma	Estándares E-learning
Moodle 1.9	**Especificaciones:** Norma SCORM para intercambio de contenidos. Soporta IMS para la representación de exámenes. Soporta AICC para el intercambio de cursos.

	Permite la integración con LAMS para la creación de secuencias de aprendizaje. Módulo para la carga de los paquetes SCORM (. Zip) que simplifica la carga / descomprimir proceso. Importar el contenido de los cursos en AICC SCORM 1.2 o compatible Exportar cuestionarios, contenidos en el formato IMS QTI 2.0.
ATutor	**Especificaciones:** Soporta la creación, importación y exportación de paquetes de contenido IMS 1.1.3 y contenido SCORM 1.2. Implementa especificaciones Runtime Environment (LMS-RTE3). Se pueden desempeñar (SCO) Incluye herramientas para facilitar la migración del contenido del curso entre las diferentes versiones del software.
DotLrn	**Especificaciones:** SCORM, así como IMS CP. Soporta IMS-QTI para el intercambio de cuotas.
Claroline	**Especificaciones:** Claroline puede importar paquetes SCORM 1.2.
Sakai	**Especificaciones:** Uso del código abierto Melete lección herramienta. El sistema puede exportar el contenido de los cursos utilizando el estándar IMS CP. El sistema de evaluación puede importar el

	contenido en el formato IMS QTI 1.2

2.2.5 Estándares y especificaciones de contenidos en el mercado

El objetivo general de un estándar enmarcado dentro de plataformas virtuales de aprendizaje es definir normas válidas para la creación de cursos virtuales, por lo tanto su principal característica es permitir que los cursos puedan trabajar sin problema alguno en diferentes plataformas garantizando interoperabilidad, manipulación, adaptabilidad y crecimiento gradual del contenido generado. Por ello es necesario establecer cuáles son los lineamientos que actualmente existen en el mercado para poder plantear alternativas de implementación de cada una de ellas, y al mismo tiempo generar una cultura para la creación de cursos y masificación de contenidos con normas básicas de publicación.

Antes de empezar a describir cada una de las especificaciones es necesario conocer el fundamento principal para llevar a cabo un proceso de construcción de contenidos, para ello el autor (Márquez, 2007) en su proyecto de doctorado enmarca una serie de estructuras básicas necesarias para la construcción de contenidos, una de ellas se encuentra ligada al tipo de formación que actualmente se presenta a saber: D-Learning (Distance learning), E-Learning (Electronic Learning), B-learning (Blended-learning) y recientemente a raíz de éste M-Learning (Mobile Learning) y T-learning (Televisión Learning); este último enmarcado dentro de la evolución que ha tenido la Televisión Digital mediante IPTV. La otra estructura básica se ve reflejada en determinar la creación de contenidos mínimos que cumplan con tres propósitos:

- Contenidos conceptuales (hechos, principios y conceptos).
- Contenidos procedimentales (habilidades, técnicas y estrategias).
- Contenidos actitudinales (actitudes, valores y normas).

Estos contenidos son apoyados en una variada gama de metodologías que permitan en cierto sentido manejar modelos que garantizan un enfoque de aprendizaje, y para poder llegar a este estado es importante proponer alternativas para identificar de qué manera viene representado ese contenido a nivel de su estructura modular.

Para apoyar este esquema de creación de contenidos es importante enmarcarlo dentro de las postura trabajadas por (VALIATHAN P, 2002) al orientar este tipo de esquemas para permitir encarrilarlos sobre enfoques de: Habilidades, Actitudes y Competencias, con el ánimo de tener una base ideal para la posterior construcción de Objetos Virtuales de Aprendizajes (OVA). Otro enfoque representativo es el trabajado (Vélez R, 2007) en su proyecto de investigación de tesis doctoral al presentar diferentes modelos de aprendizajes para representarlos dentro de la creación de objetos virtuales de aprendizaje y su representación sobre las especificaciones para creación de los mismos.

2.2.6 Objetos y representación de contenidos

Para identificar partes funcionales de cualquier especificación o estándar, es importante resaltar el concepto de Objetos de Aprendizaje desde el punto de vista de programación, el cual, (Gutierrez S, 2007) dentro de sus trabajos resalta la idea de que actualmente este concepto está ligado para representar contenidos reflejados mediante creación de contenidos; por tanto define que actualmente este concepto ha evolucionado para lograr trabajar con objetos de aprendizaje reutilizable RLO (Reusable Learning Object) temas trabajados en su momento bajo recomendaciones dadas por la (IEEE, 2002) y por el autor (Wiley D and Edwards K, 2002) para definición de metadatos y elementos válidos para definición de contenidos, los cuales tratan de adaptar en cierta forma al mundo del E-learning para ofrecer un paradigma de programación orientada a objetos en el uso de componentes dentro de las definiciones de la mayoría de especificaciones.

Según (Gutierrez S, 2007) "El objetivo de los RLO es tener diversas piezas de material educativo que se pueden combinar entre sí y reutilizarse en diferentes contextos. Sin embargo, la reutilización de objetos de aprendizaje plantea más problemas que la reutilización de objetos software", estos problema están asociados con la identificación del estándar a utilizar, permitiendo de esta manera ser un elemento fundamental dentro del proceso de creación de contenidos.

A pesar de la definición de estos elementos, parece haber cierto grado de insatisfacción por parte de autores que reconocen la importancia de la reutilización de contenidos, pero que en el trasfondo se orientan hacia una mala contextualización de los contenidos y temas tratados dentro de los elementos y componentes al ser parte fundamental dentro de un proceso de aprendizaje como lo afirma (Wiley D and Edwards K, 2002), y que según(Gutierrez S, 2007), "iniciativas como IMS Learning Design modifican la idea de RLO y la orientan a enfoques pedagógicos complejos con varios participantes en colaboración". Lo que permite identificar que no existe una directriz alineada que permita manejar este tipo de componentes dentro de las especificaciones presentes en el mercado.

Al ser este un parámetro que las especificaciones tratan de determinar dentro de sus modelos y modo de operación, es importante resaltar este concepto como parte fundamental de una plataforma virtual de aprendizaje, los cuales bajo este contexto y mecanismos de representación trabajados por (Vélez R, 2007), "los LMS son los que se encargan de la creación, reusabilidad, localización, desarrollo y gestión de este tipo de contenidos formativos", lo que permite identificar a estos elementos como contenidos que por lo general son almacenados en un repositorio en forma de Objetos de Aprendizaje Virtuales (OVA), cuya característica va a permitir satisfacer objetivos definidos dentro del proceso de formación de un estudiante. En definitiva, sobre las plataformas LMS se deben garantizar una interoperabilidad entre ellas con el ánimo de ofrecer la posibilidad de reutilizar recursos creados mediante los OVAs.

2.2.7 Especificaciones y estándares

Se define como estándar al conjunto de especificaciones trabajadas mediante un formato, plantilla, método o tecnología el cuál logra ser ratificado mediante una organización a nivel Internacional reconocida a nivel de estandarización. Para este caso en particular se mencionaran la ISO (International Organization for Standardization), la IEEE (Institute of Electrical and Electronics Engineers) y nacionales como la BSI (British Standards). El único organismo de estandarización en E-Learning es el LTSC (Learning Technology Estándards Committee) de la IEEE y sus estándares serán luego adoptados por la ISO.

A continuación se resume una tabla la cual fue definida por (Wiley D and Edwards K, 2002) para complementar la idea del concepto de especificaciones y estándares.

Tabla 2 Diferencias entre especificaciones y estándares e-learning según (Wiley D and Edwards K, 2002).

Especificaciones	Estándares
Capturan el consenso aproximado	Capturan la aceptación general
Evolucionan rápidamente	Evolucionan lentamente
Facilitan	Regulan
Gestionan los riesgos a corto plazo	Gestionan los riesgos a largo plazo
Experimentales	Conclusivos

Bajo las anteriores premisas, es importante aclarar en este punto, que a la fecha no existe un estándar universal definido y estandarizado a nivel global, y esto es debido a que la mayoría de las plataformas LMS definían sus propios formatos de almacenamiento y procesamiento de contenidos, lo que ocasionaba una falta de coordinación en el desarrollo de estándares, desencadenado una serie de especificaciones para tratar de regular en cierto sentido estos elementos; mas sin embargo en el ámbito académico las especificaciones definidas en IMS se les sigue conociendo en el mercado como estándar, pero desde hace algunos años este panorama ha empezado a cambiar paulatinamente bajo la convergencia de varios proyectos que se están aglutinando alrededor del proyecto SCORM, el cual se encuentra trabajando de la mano bajo el apoyo de AICC, IEEE LTSC y, lógicamente amparado bajo un fuerte apoyo sobre IMS.

El siguiente apartado se centra exclusivamente en la explicación de un conjunto de especificaciones que se definen para delinear los aspectos más relevantes dentro de la construcción de contenidos y posterior generación de objetos virtuales de aprendizaje dentro de las plataformas LMS, el cual, al mismo tiempo servirá de apoyo para determinar las características de cada uno de ellos; en especial el conjunto de

especificaciones SCORM para conocer las estrategias mediante el cual se pueda trabajar.

Los elementos que serán estudiados a nivel de estructura a saber son: El estándar LOM de la IEEE desde el punto de vista de su estructura, al igual se tendrá en cuenta dentro del conjunto de especificaciones IMS, los modelos IMS-MD (Model Data), IMS-CP (Content Packaging) e IMS-LD (Learning Data) como elementos primordiales para introducir al conjunto de especificaciones SCORM el cual tiene incluidos las normas anteriormente mencionadas, con el ánimo de identificar dentro de su estructura posibles elementos trabajados a nivel de interoperabilidad.

2.2.7.1 Estándar IEEE Learning Object Metadata LOM

LOM (Learning Object Metadata) fue creado para definir esquemas de metadatos usados para la generación de un objeto de aprendizaje. En este sentido metadatos se conoce como información adicional que se agregan a la creación de contenidos para facilitar su clasificación y reutilización. Al tratarse de un proyecto dirigido por la IEEE, se le conoce como una de las primeras organizaciones a nivel americano en el mercado en tratar de impulsar a LOM como estándar, el cual es trabajado a través de un comité conocido como LTSC (Learning Technology Standards Committe). Otra de las características de este estándar es que ha sido adoptado como una especificación de descripción de metadatos por IMS, lo que permite realizar migraciones entre estas dos versiones.

Varios autores coinciden en el hecho de que este estándar permite centrarse en la especificación de los metadatos para generar la estructura básica de la creación de los objetos de aprendizaje, en este sentido (Gutierrez S, 2007) afirma que la contribución más importante de LOM se centra en “definir el conjunto mínimo de atributos que deben adjuntarse a estos objetos de aprendizaje para manejarlos, los cuales no solamente intentan realizar el manejo relacionado con elementos administrativos como (autor, título, fecha de creación, etc) sino también aspectos pedagógicos de la utilización del objeto de aprendizaje a saber (nivel de dificultad, enfoque pedagógico, prerrequisitos para llevar a cabo la lección, etc)”. Lo cual permite almacenar dentro de su conjunto características administrativas y pedagógicas para complementar la realización de objetos de aprendizaje, y en este sentido sería un gran apoyo para el proyecto desde el punto de vista administrativo.

2.2.7.2 Características de atributos manejados por learning object metadata

Dentro de las características anteriormente mencionadas, a continuación se presenta los atributos que son manejados por este estándar para trabajar sus contenidos y los cuáles fueron productos de la IEEE mediante su comité (IEEE, 2002)

1. General. Trabaja información general sobre el objeto de aprendizaje. Ejemplo: Título, numero identificador del objeto, número de catálogo, descripción del objeto en cuanto a grado de dificultad, palabras claves, entre otras.

2. Ciclo de vida. Maneja características relacionadas con la evolución del objeto para identificar si ha habido cambios en el mismo. Ejemplo: Número de versión, estado del objeto, si es la versión de prueba o la definitiva, personas que han contribuido a la mejora del objeto, entre otras.

3. Meta-metadatos. Maneja atributos propios, metadatos que se agregan al objeto Ejemplo: Tipo de idioma, personas que han contribuido al marcado.

4. Técnicos. Estos atributos están orientados a explicar cómo debe manipularse el objeto. Ejemplo: Versión, duración, método de instalación, software necesario, entre otras.

5. Educacional. Contiene características relacionadas sobre aspectos pedagógicos Ejemplo: Nivel de interactividad, densidad semántica, nivel de dificultad, etc.

6. Derechos. Maneja información de tipo legal en relación con la propiedad intelectual y características de distribución del objeto.

7. Relacional. Estos atributos capturan cómo se relaciona este objeto con otros, para mirar grados de coherencia con otros contenidos desarrollados en otros objetos creados.

8. Anotaciones. Maneja información acerca de cómo ha sido el uso del objeto en un entorno de aprendizaje, permitiendo de esta manera un intercambio de información entre personas académicas que lo usan, sugerencias, y lógicamente la evolución del mismo.

9. Clasificación. La característica de este atributo es determinar una relación directa con el grado de Jerarquía en el cuál se encuentra ubicado el Objeto, con miras a clasificar y mejorar su nivel de utilización dependiendo de la rama en particular al cuál haya sido orientado y por tanto creado.

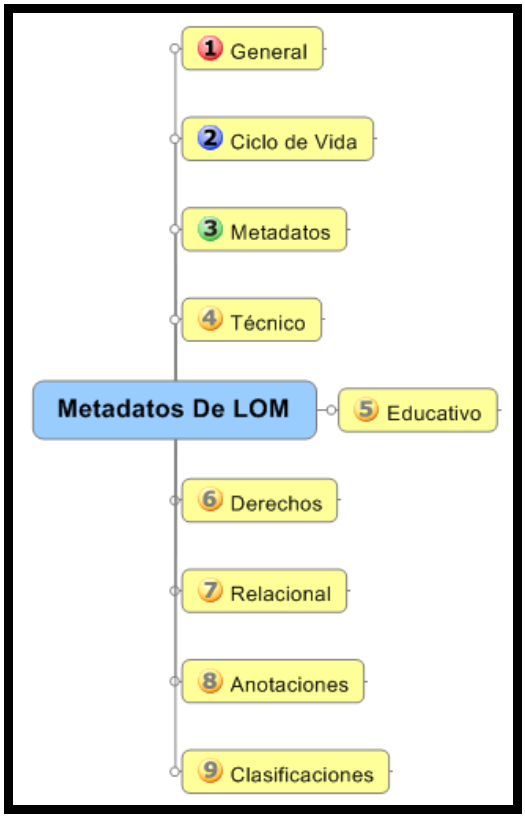


Figura 1 Metadatos trabajados por LOM.

La anterior figura representa cada una de los metadatos trabajados por LOM definidos por la IEEE mediante (IEEE, 2002), el cuál dentro de cada uno de los bloques, presentan características de formatos que se pueden trabajar directamente para agregar información adicional a cada contenido creado dentro del objeto de aprendizaje. Estas características permiten personalizar más el contenido y determinar elementos de reusabilidad del mismo para garantizar evolución de los contenidos creados.

Las descripciones definidas en cada bloque consisten en proporcionar valores para los distintos metadatos considerados. Esta asignación se podrá crear para efectos de reuso mediante lenguajes de etiquetas utilizando en este caso representación XML. Por tanto la definición de metadatos desde el punto de vista de lenguaje XML, se puede usar dentro del contexto tecnológico para ubicar fácilmente objetos de aprendizaje que puedan conducir fácilmente a la ubicación de contenidos y posterior utilización de los mismos, garantizando en este sentido una flexibilidad, rapidez, reutilización y uniformidad de contenidos expuestos a través de objetos de aprendizajes.

2.2.7.3 Conjunto de especificaciones del Instructional Management Systems IMS

IMS bajo sus siglas (Instructional Management Systems), es un consorcio trabajado por la Global Learning Consortium Inc, creado en 1997 con el ánimo de desarrollar especificaciones generadas por el mismo grupo según las necesidades de los más de 138 organizaciones que hacen parte de ella (Consortium, 2010) cuyos intereses van desde el desarrollo de contenidos como grandes editoriales: Pearson Education, McGrawHill, Angel Learning entre otras; a nivel de desarrolladores de plataformas LMS, en los que se resaltar proyectos a nivel tecnológico como Blackboard en unión con WebCT, Moodle, Sakai entre otras; a nivel académico universidades como Cambridge, Michigan, Utah, Toronto, California, entre otras; al igual empresas dentro del sector tecnológico como IBM, Oracle, Microsoft y mas; consolidando de esta manera un grupo bastante fuerte para que las decisiones que se tomen al interior sean de impacto para que otras versiones utilicen las normas que ellos desarrollan, generando de esta forma una gran aceptación en el resto de grupos que trabajan en estas soluciones a nivel internacional y consolidándolos en este sentido como un grupo sólido dentro de las normatividades creadas al interior del Consorcio.

Para su óptimo funcionamiento, este consorcio ha generado una serie de especificaciones las cuales logran abordar temáticas a nivel de estructuras de aprendizaje mediante enfoques a nivel de E-learning para creación de contenidos, al igual que desde el punto de vista tecnológico mediante las necesidades creadas por los mismos proyectos sobre plataformas LMS, lo que ha generado en el Consorcio la creación de una serie de lineamientos para la explicación de las más de 20 especificaciones conocidas a la fecha (Consortium, 2010) públicas y otras 20 más que no han sido publicadas a la fecha, las cuales son amparadas por tres documentos para orientar al usuario en la forma de usar cada una de ellas:

- Guía de Implementación: Es un documento que determina de manea introductoria en qué consiste la guía y como ha de utilizarse dentro de su especificación, siendo un documento de ayuda para guiar al usuario en el uso de la especificación.

- Modelo de Información: Describe de manera general las características de los datos, la estructura de los mismos para orientar al usuario en la forma de uso del mismo.

- Documento de Enlace: Ofrece la manera en la cual se van a representar los datos, para este caso usa un lenguaje formal en XML para proporcionar una estructura del documento que se va a generar dentro de la especificación a la cual estará relacionado.

Partiendo de estas iniciativas al tratar de utilizar de manera adecuada las especificaciones dadas por el (Consortium, 2010), se resalta nuevamente que para efectos de este trabajo solamente se abordaran tres especificaciones, a saber: A nivel

de modelamiento de datos IMS-MD (Model Data), a nivel de estructura de contenidos mediante IMS-CP (Content Packaging) y a nivel de datos de aprendizaje con IMS-LD (Learning Data).

2.2.7.4 Especificación para modelos de datos del Instructional Management Systems IMS-MD

Es una especificación que se encarga de hacer más eficiente el uso de recursos o contenidos y la búsqueda de los mismos mediante una estructura generada para los metadatos, quienes son los encargados en esta instancia de catalogar los recursos de aprendizaje, al igual que parámetros que deben ser usados, representados y organizados.

Según (Márquez, 2007), otra de las características de esta especificación es que "proporciona información para conocer el estado de los contenidos "etiquetados" y la forma en que deben ser organizados para que los alumnos puedan intercambiar información entre los diferentes servicios involucrados en un sistema de gestión de aprendizaje". Lo que garantiza la generación de un modelo para el intercambio de información dentro de un LMS.

Una evolución que tuvo esta especificación fue la adaptación del estándar LOM de la IEEE dada por la (IEEE, 2002) para la definición de parámetros y elementos que permitan la construcción de contenidos, lo que obligo a cambiar de denominación y publicar la última versión conocida como la 1.3(Global Learning Consortium, 2001) mediante la especificación IMS-LRMD (Learning Resource Meta Data). Lo que permite en este sentido representar los datos mediante lenguajes a través de XML.

El modelo conceptual manejado por esta especificación es jerárquico, pues al adoptar el estándar LOM, ha permitido trabajar etiquetas que permitan la identificación de contenidos de manera eficiente, el cuál en la siguiente figura se puede ver representado la forma de trabajo adoptado.

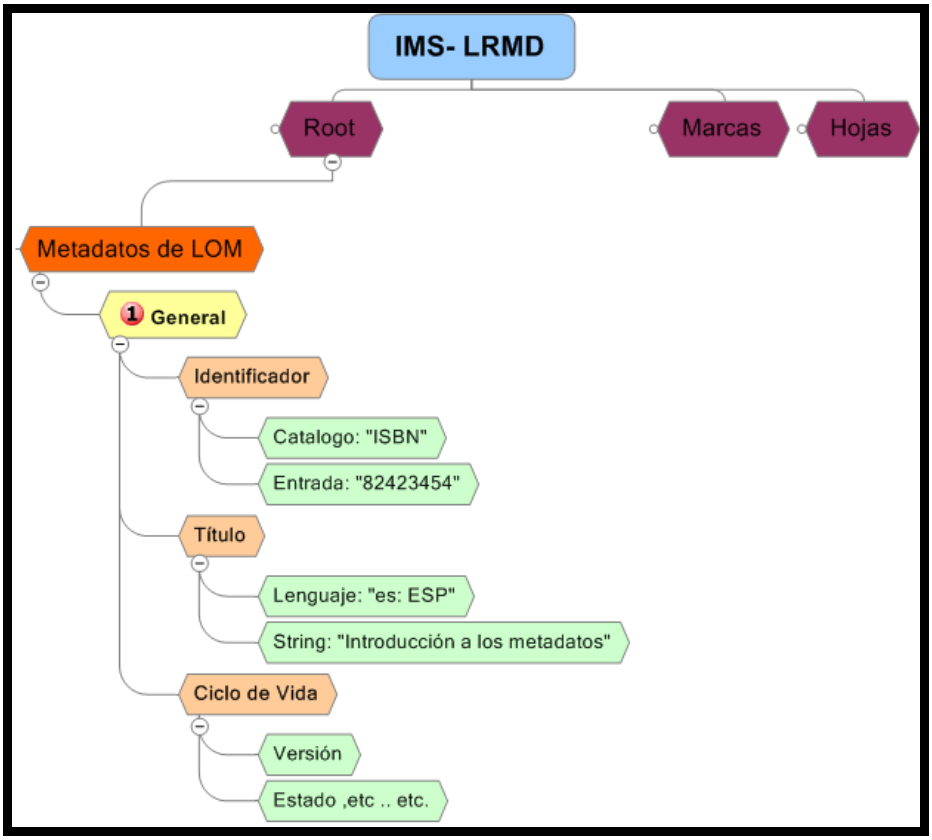


Figura 2 Árbol Root de IMS-MD apoyado sobre LOM.

En la anterior figura se puede observar que cada valor ó metadatos es usado jerárquicamente para definición de tipos de datos, valores y detalles de cada elemento encontrado dentro de un proceso de búsqueda de contenido, manejando en este sentido un esquema de metadatos y elementos en formato tabular.

2.2.7.5 Especificación para contenidos de paquetes del Instructional Management Systems IMS-CP

Es una especificación que como su nombre lo indica define la forma de empaquetar contenidos para facilitar su utilización dentro de varios LMS, garantizando en este sentido el concepto de reutilización y adaptabilidad comentado en varias ocasiones en este apartado; por lo tanto permite definir la estructura del contenido y representar la manera en la cual debe mostrarse al usuario final.

Para la elección de los elementos a empaquetar contenidos según (Márquez, 2007) “se encuentran relacionados directamente con etiquetas como: la descripción, estructura, y ubicación de los materiales, y a la definición de algunos tipos particulares de contenidos, tipos de recursos entre otros”, lo cual permite una integración de elementos de búsqueda para identificación de contenidos en un LMS, el problema radica en organizarlos para poderlos representar de manera adecuada, para lo cual esta especificación cuenta con una estructura que permite organizar el material para

que se pueda exportar, agregarse e importarse dentro de un LMS y otros parámetros que garantizan su óptimo funcionamiento.

Para realizar la distribución e intercambio de contenidos y cursos dentro de los LMS, se crea es un archivo conocido como PIF (Package Interchange File), el cual se encarga de guardar el archivo manifiesto y el resto de recursos y contenidos que se referencia dentro del manifiesto; este tipo de empaquetamiento se realiza mediante formatos como .ZIP o .JAR, lo que garantiza manejar una estructura modular para garantizar el funcionamiento del contenido a trabajar, reuniendo de esta forma una serie de características ligadas al contenido como, recursos, tareas, actividades, evaluaciones ó exámenes. Esta funcionalidad de PIF, permite realizar acciones de importación y exportación de formatos para que se puedan trabajar sobre cualquier tipo de plataforma a nivel comercial como Blackboard, o a nivel open source como Moodle, Claroline, dotLRN, entre otros LMSs.

Uno de los elementos más representativos de esta especificación se conoce como el manifiesto, el cuál es un fichero trabajado en lenguaje XML llamado imsmanifest.xml que permite identificar los parámetros más relevantes para describir la estructura de contenidos incluidos dentro del paquete representado a través de la siguiente figura:

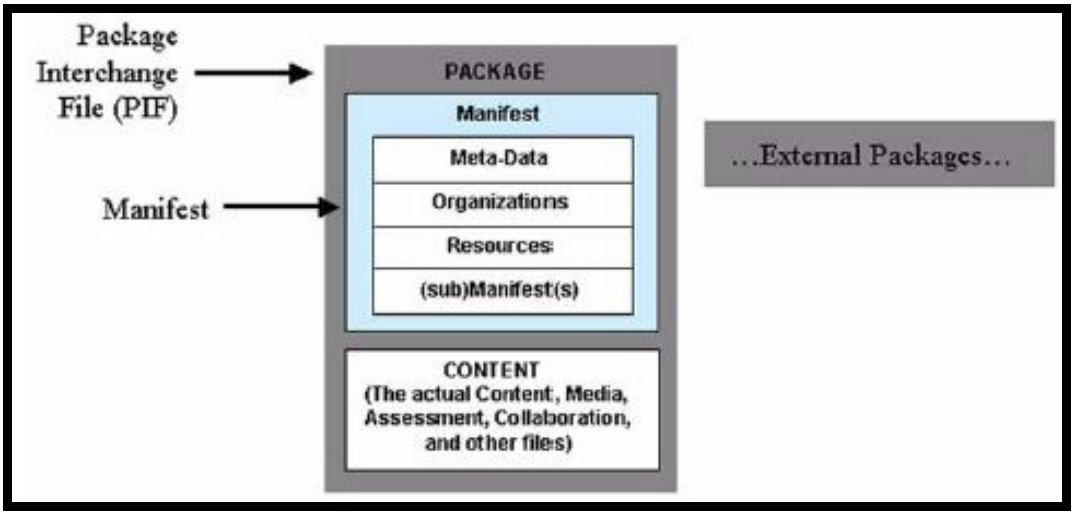


Figura 3 Estructura del Manifiesto IMS-CP (Global Learning Consortium, 2005).

Dentro de la representación del Manifiesto se describen dos elementos los cuales definen la organización y los recursos a utilizar:

1. Organización del Contenido del Paquete: Crear en orden jerárquico en forma de árbol los Recursos del paquete. Uno de los parámetros más representativos para llevar a cabo esta organización es mediante los Ítems, a los cuáles se le puede asociar un Recurso.

2. Recursos utilizados por contenidos: Esta etiqueta describe un conjunto de recursos y sus dependencias, lo que garantiza agregar elementos como presentaciones, documentos, animaciones, imágenes, etc.

Esta especificación usa lenguaje XML, lo que permite describir su estructura en cuanto a la forma de representación del contenido, y por tanto representa una de las especificaciones más importantes trabajadas por SCORM.

En la siguiente figura se detallan los elementos del manifiesto trabajado a través de lenguaje XML.

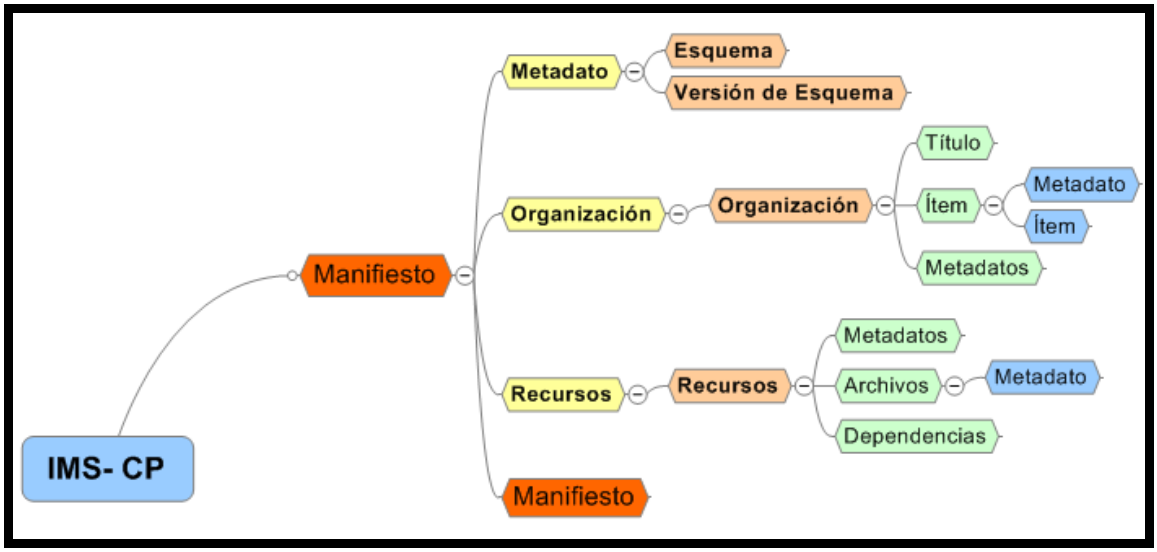


Figura 4 Elementos del archivo manifiesto (Global Learning Consortium, 2005).

De manera general este esquema representa las etiquetas que representa el empaquetamiento de contenidos, en el cual cada representación que se observa se etiqueta a través de Identificadores, versión y esquema de lenguaje XML que se está trabajando para cada caso en particular, permitiendo de esta forma caracterizar los diferentes contenidos que se van a trabajar y especificar los tipos de documentos que se trabajan.

2.2.7.6 Especificación para diseño del aprendizaje del Instructional Management Systems IMS-LD

De acuerdo a la especificación IMS Learning Design (IMS LD). El concepto central de IMS LD es que, independientemente de la aproximación pedagógica, una persona toma un rol en un proceso de aprendizaje-enseñanza, normalmente el rol de alumno o docente. En este rol la persona tiene que trabajar para conseguir ciertos resultados mediante la realización de actividades de aprendizaje o soporte más o menos estructuradas en un entorno (Consortium, 2003b). Las características particulares de los roles que una persona puede tomar, las actividades a realizar y las características particulares del entorno definen un escenario de aprendizaje específico. Este

escenario de aprendizaje puede ser representado en IMS LD, donde se denomina una Unidad de Aprendizaje o Unit of Learning (UoL). La UoL puede ser ejecutada en algún sistema compatible IMS LD.

IMS LD no ofrece ningún modelo o modelos pedagógicos concretos, sino que puede ser utilizado para definir prácticamente una serie ilimitada de escenarios educativos. A causa de esto se suele referenciar como un metamodelo pedagógico. Algunas iniciativas anteriores de aprendizaje electrónico habían reclamado su neutralidad pedagógica. IMS LD no busca la neutralidad pedagógica, sino que trata de permitir que el e-learning sea consciente de la pedagogía. IMS LD fue desarrollado en un contexto de e-learning, pero no hay ninguna razón por la que las Units of Learning no puedan ser utilizadas en combinación de contextos de aprendizaje en línea y cara a cara, o de forma completa en aprendizaje cara a cara.

IMS LD es el intento de avanzar diseñando para alumnos aislados en línea que están limitados a leer desde las pantallas, IMS LD agrupa personas, actividades, recursos, flujos en escenarios para alcanzar objetivos de aprendizaje. La cuestión principal no es crear contenido, sino crear actividades de aprendizaje estructuradas diseñadas para alcanzar objetivos de aprendizaje. Por lo tanto, IMS apostó por una especificación centrada en el proceso de aprendizaje y no tanto en los contenidos finales, intentando asegurar la interoperabilidad de los módulos o Unidades de Aprendizaje (UoL) generados con ella. Mediante IMS LD el profesor o pedagogo puede representar un escenario de aprendizaje sustentando en cualquier pedagogía, ya que es completamente neutro en este aspecto (Burgos et al., 2005). Es, por tanto, pedagógicamente flexible, lo que implica además una ejecución multiplataforma, independientemente del visualizador IMS LD utilizado.

IMS LD utiliza la metáfora del teatro para ayudar a entender las Unidades de Aprendizaje. Una serie de actores representa una obra/ejecución/play; cada uno de ellos puede asumir un número de roles en diferentes momentos de la obra, en varios actos.

En IMS LD un usuario ejecuta un rol (profesor, tutor, alumno, supervisor, oyente…) dentro del proceso de aprendizaje-enseñanza y sigue una serie de actividades de aprendizaje para completar los distintos momentos o actos de dicho proceso. Un escenario de aprendizaje completo viene definido por los recursos, la metodología, las actividades de aprendizaje y soporte, los servicios complementarios, la relación entre los roles, la agrupación de usuarios y un largo etcétera. Todo ello se define en un fichero llamado manifiesto, escrito en XML, que junto con los recursos es empaquetado en un fichero en formato ZIP, consiguiendo la Unidad de Aprendizaje (UoL). Este fichero comprimido podrá ser ejecutado, abierto o modificado en cualquier sistema compatible LD. Del mismo modo, en IMS Learning Design, un estudiante puede asumir diferentes roles en diferentes etapas del proceso de aprendizaje. Al final de cada acto, la acción se detiene, todos los estudiantes se sincronizan y todo puede volver a empezar.

Estructura de la especificación

La especificación IMS LD comprende diversos documentos:

a) Un modelo conceptual que define los conceptos básicos y las relaciones dentro de Learning Design (conceptual model)

b) Un modelo de información que describe los elementos y los atributes que pueden ser definidos en LD de una manera detallada (information model)

c) Un conjunto de ficheros XSD (XML Schemas) en los que se basa la implementación del modelo de información

d) Una guía para la óptima implementación de la especificación (Best Practices and Implementation Guide)

e) Un conjunto de ejemplos y escenarios desarrollados en XML (binding document)

IMS LD se divide a su vez en tres niveles de implementación:

Nivel A (Level A)

Constituye la base y comprende la definición de usuarios, actividades de aprendizaje, actividades de soporte, entornos, recursos, método, ejecuciones (plays), actos, roles y la coordinación entre todos ellos, es decir, la expresividad pedagógica. Además los usuarios podrán utilizar recursos externos, enlaces web y diversos servicios (foros, chats…).

Nivel B (Level B)

A la base proporcionada por el Nivel A añade propiedad y condiciones, así como servicios de monitorización y elementos globales, lo que permite al usuario definir estructuras más complejas. Las propiedades almacenan información sobre personas (preferencias, resultados, información personal...), sobre un rol o sobre el diseño de aprendizaje en sí mismo. Si las propiedades son locales, se denominan internas y se mantienen únicamente durante la ejecución de una instancia (run). Si son globales, también llamadas externas, pueden ser consultadas y utilizadas en diferentes instancias y sus datos persisten a través de varias sesiones. El estado de las propiedades y de las condiciones puede modificar el flujo de trabajo e influir en el desarrollo de la Unidad de Aprendizaje (UoL). Constituye el nivel que aporta más flexibilidad a la hora de la representación didáctica ya que permite esconder y mostrar elementos, condicionar el flujo de aprendizaje, almacenar datos del usuario y la instancia, bien a nivel local y personal, bien a nivel global y compartido. De esta manera podemos incorporar cuestionarios, evaluaciones, cálculos numéricos, seguimiento de usuarios y otros.

Nivel C (Level C)

Añade notificaciones al Nivel B. Las notificaciones se ejecutan automáticamente como respuesta a eventos que se originan en el proceso de aprendizaje. Por ejemplo, si un estudiante envía un trabajo para ser evaluado, se podría enviar automáticamente un correo electrónico al profesor para informarle.

La especificación IMS LD se ha diseñado con el objetivo de beneficiar, básicamente, a dos grupos: los profesores y especialistas en educación, y los alumnos.

2.2.7.7 Modelo de referencia para compartir objetos de contenido SCORM

SCORM, bajo sus siglas (Shareable Content Object Reference Model) fue una iniciativa desarrollada por ADL (Advanced Distributed Learning) el cuál surgió cobijado por el Departamento de Defensa de los Estados Unidos en el año de 1997.

La propuesta fundamental de este organismo es apoyarse en las anteriores iniciativas para conformar su propio conjunto de especificaciones, el cual combina muchas especificaciones de IMS, IEEE mediante LOM, AICC entre otras mencionadas anteriormente. Pero desde su última versión del año 2004, trabaja específicamente las siguientes especificaciones:

- IEEE Data Model For Content Object Communication.
- IEEE ECMA Script 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.

Su último lanzamiento hecho en el año 2004, se realizó con el ánimo de mantener independiente cada especificación trabajada por SCORM. Lo que garantiza una buena gestión a la hora de mantener las revisiones y correcciones que se introduzcan dentro de una sola especificación en conjunto, permitiendo de esta manera aplicar solamente cambios a las especificaciones afectadas.

Según sus integrantes y el grupo interdisciplinario que se encuentra ubicado en regiones norte y centro América, Europa y Oceanía (Advanced Distributed Learning, 2004), SCORM ayuda a definir las bases técnicas de un aprendizaje basado en el aprendizaje basado en la Web con el ánimo de crear un Modelo de Referencia. En su forma más simple, es un modelo que hace referencia a un conjunto de normas

técnicas, especificaciones y directrices destinadas a satisfacer las necesidades de alto nivel para el aprendizaje de contenidos y sistemas.

2.2.7.8 Principios del modelo de referencia para compartir objetos de contenido

Según (Advanced Distributed Learning, 2004),se define que deben existir tres criterios fundamentales para la adopción de SCORM como el conjunto de especificaciones válidas para trabajo con desarrollo de contenidos y objetos de aprendizaje:

- Debe ser usado para articular normas y directrices que puedan ser comprendidas y aplicadas por los desarrolladores de contenidos de aprendizaje.
- Debe ser usado para que una gran variedad de usuarios interesados en el desarrollo de contenidos de aprendizaje y plataformas de aprendizaje LMS puedan trabajarlo.
- Debe permitir un seguimiento de las características desarrolladas por los diseñadores de sistemas de instrucción para el desarrollo apropiado de contenidos.

Otra de las características adicionales de SCORM definidas por (Advanced Distributed Learning, 2004) es que debe asegurar los principios básicos de manejo de información para todo sistema de información, a saber:

- Accesibilidad: La capacidad de localizar y acceder a los componentes sobre una ubicación remota para acceder desde cualquier lugar.
- Adaptabilidad: La capacidad para adaptar los contenidos a las necesidades individuales y organizacionales.
- Asequibilidad: La capacidad para aumentar la eficiencia y la productividad al reducir el tiempo y los costos involucrados en la entrega del contenido.
- Durabilidad: La capacidad de soportar la evolución tecnológica y los cambios sin costoso para rediseñar, reconfigurar o reprogramar el contenido.
- Interoperabilidad: La capacidad de tomar componentes desarrollados en una ubicación remota sobre una plataforma y usarlos en otra ubicación con un conjunto de herramientas y plataformas diferentes.
- Reutilización: La flexibilidad de incorporar componentes de instrucción en múltiples aplicaciones y contextos.

2.2.7.9 Modelo operativo del modelo de referencia para compartir objetos de contenido

Actualmente se está trabajando la segunda versión del 2004, el cuál (ADL, 2004) lo representa mediante cambios significativos realizados de la primera versión lanzada al mercado.

2.2.7.10 Contenido del modelo de agregación (CAM)

El Modelo de agregación de contenidos según su especificación (ADL, 2004) define cómo hay que ensamblar, etiquetar y empaquetar el contenido. Como el objetivo de SCORM es orientado a objetos, se necesita de una descripción detallada sobre cómo se conectan estos objetos.

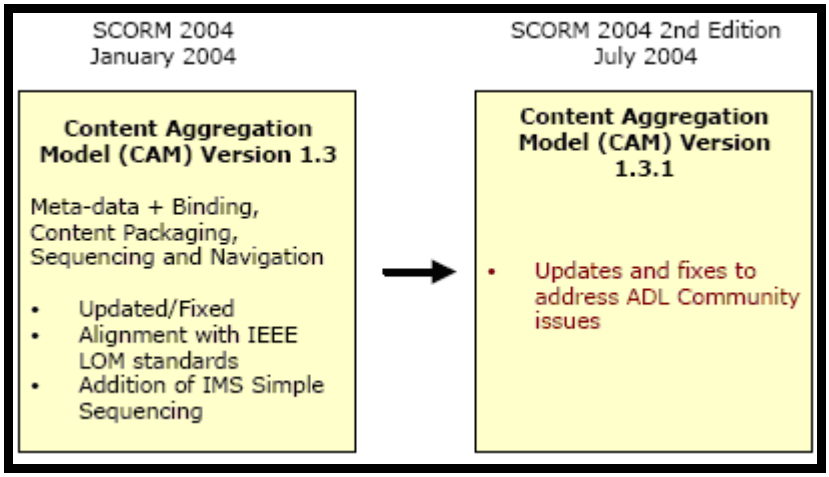


Figura 5 Modelo CAM (Advanced Distributed Learning, 2004).

Según (Gutierrez S, 2007) "Una experiencia de aprendizaje se compone, en este contexto, de activos (assets), objetos de contenido compartible (Shareable Content Objects, SCOs) y organizaciones de contenido. Un SCO está compuesto de varios activos, y varios SCOs componen una organización de contenido. Ejemplos de activos son: documentos HTML, ficheros de audio o video, etc. Los SCOs contienen un conjunto de activos y deben ser ejecutables por el LMS.

2.2.7.11 Run-Time Environment (RTE)

RTE describe el proceso de ejecución que debe realizar un LMS con un SCO, así como el proceso de comunicación entre ambos. Un estudiante sólo tiene un SCO activo en cada momento.

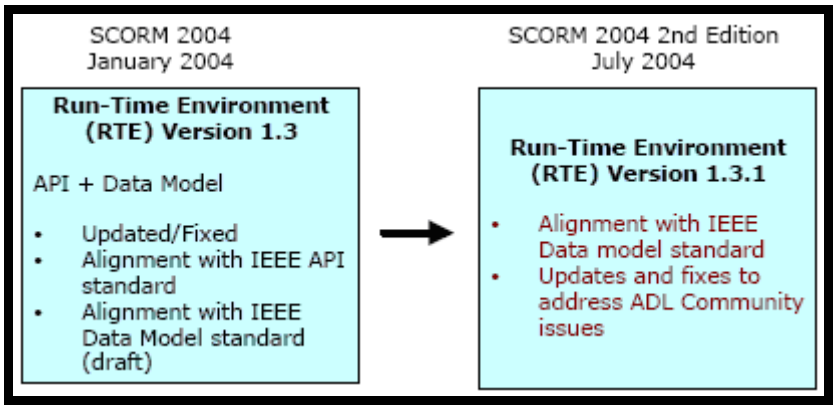


Figura 6 Run-Time Environment (RTE) (Advanced Distributed Learning, 2004).

2.2.7.12 Secuenciación y navegación (SN)

El modelo SCORM describe cómo el secuenciamiento interactúa con el resto del RTE; sin embargo, la descripción del proceso, de secuenciamiento se hace usando la especificación IMS (Simple Sequencing) descrita en (Consortium, 2003a), el cual (Gutierrez S, 2007) lo define como "un método para definir la secuencia en la que un grupo de actividades de aprendizaje se presentan al estudiante. Incorpora reglas que describen el flujo o ramificación de las actividades según la interacción entre el usuario y las actividades. Su objetivo es ser el punto de encuentro entre los diferentes sistemas de gestión de aprendizaje en términos de la secuenciación de sus contenidos."

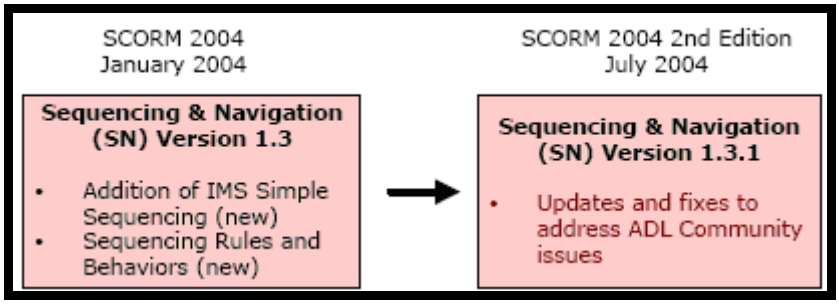


Figura 7 Secuenciación y navegación (SN) (Advanced Distributed Learning, 2004).

Por lo tanto, el modelo (Consortium, 2003a) está orientado a la relación e intencionalidad directa de los modelos pedagógicos utilizados y el uso de estrategias metodológicas de tipo instruccional, para que de manera guiada el proceso de aprendizaje se garantice al estudiante.

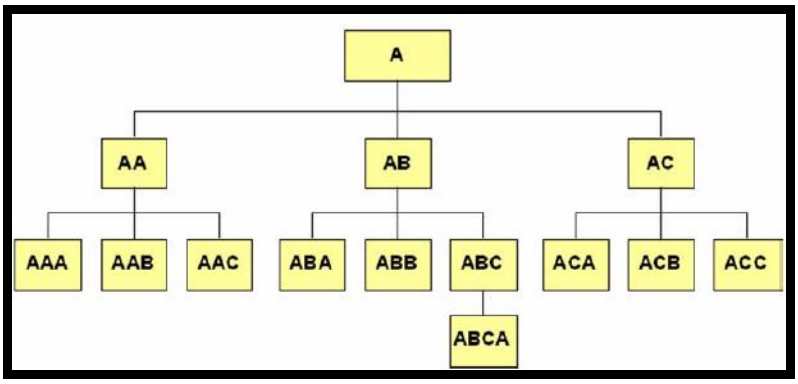


Figura 8 Modelo de Secuenciamiento (Consortium, 2003a).

Por lo tanto el Modelo de secuenciamiento garantiza un seguimiento de las actividades, (Burgos D, 2006) apoya el concepto al afirmar que esta especificación "facilita los medios necesarios para representar la información utilizada para definir diversas posibilidades de secuencias de actividades educativas, a través de la descripción de diferentes rutas de navegación por una colección de materiales didácticos. Además, define el método para representar el comportamiento de un objeto de aprendizaje para que cualquier tecnología educativa compatible sea capaz de ejecutar las secuencias de actividades educativas de forma consistente." Por lo que es una de las especificaciones más trabajadas por muchos autores desde el punto de vista pedagógico para garantizar un proceso de aprendizaje.

2.2.7.13 Modelo y repositorio de datos de SCORM sobre LMS

El trabajo realizado por SCORM bajo plataformas LMS, es servir de Repositorio Local para el empaquetamiento de Contenidos, con el ánimo de ser usado al interior de la plataforma como un administrador de servicios de contenidos dentro de un curso para que finalmente sea usado por un usuario en particular para visualización de los mismos como se representa en la siguiente figura:

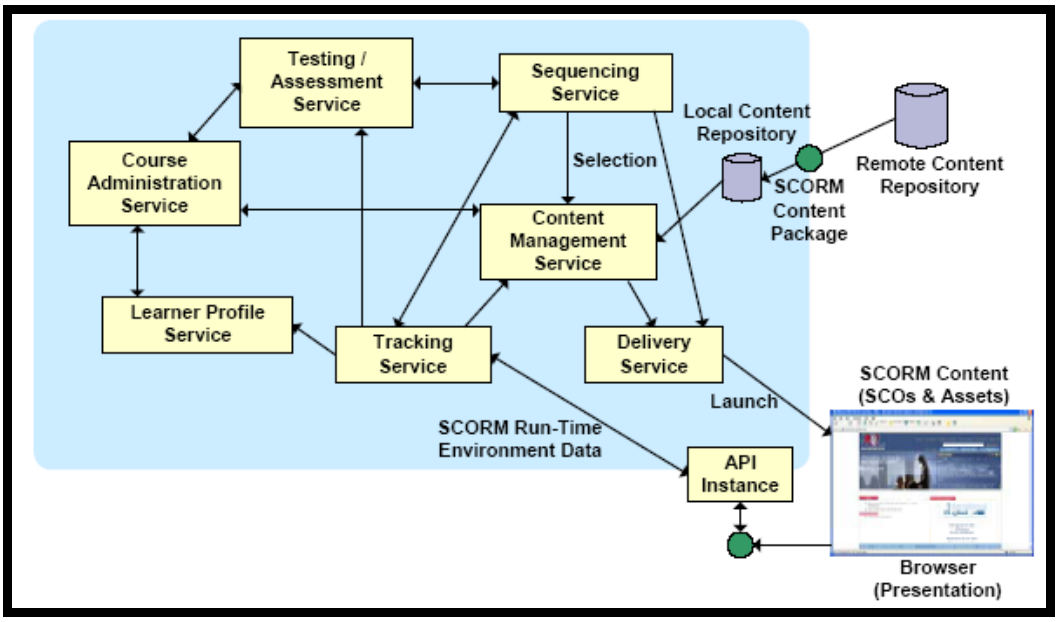


Figura 9 Modelo SCORM dentro de Plataformas (Advanced Distributed Learning, 2004).

Dentro de las especificaciones trabajadas por SCORM a nivel de LMS, básicamente se necesita una arquitectura basada en ambiente servidor, con parámetros para el manejo de contenidos para el aprendizaje correcto por parte de los usuarios, garantizando de esta manera una evolución paulatina de momentos de aprendizaje a través de la secuencia que SCORM maneja al interior de cada contenido realizado. En este sentido dentro de la especificación se definen reglas para el manejo de los contenidos y momentos para la secuencialización de contenidos y seguimiento del proceso de aprendizaje de los estudiantes.

Por otro lado es importante resaltar el uso de este lenguaje XML para tratar de entender los parámetros dados a través de las etiquetas y metadatos, los cuáles, vendrían representando un esquema de trabajo de apoyo para el uso de especificaciones que se llevan a cabo mediante SCORM, los cuáles en trabajos de investigación enmarcados a nivel de doctorado realizados por (Burgos D, 2006) los tienen en cuenta para determinar los enfoques primordiales de este lenguaje para aproximar el concepto de metadatos, junto con lenguajes a nivel d XFML (eXtensive Faceted Markup Language), OWL (Ontology Web Language), RDF y RDF Schema a nivel ontológicos.

En el próximo apartado se hablara de los que es un objeto de aprendizaje (Learning Object).

2.2.8 Objetos de aprendizaje

No es fácil encontrar una definición de lo que es un Learning Object pero Roy McGreal (McGreal, 2004) se atreve a definirlos como los recursos educativos que pueden ser empleados en cualquier tecnología de apoyo a la educación, Un Objeto de aprendizaje

permite y facilita el uso de los contenidos educativos, a través de especificaciones y normas aceptadas internacionalmente que logran la interoperabilidad y reusabilidad en las distintas aplicaciones y en diversos ambientes de aprendizaje.

Andreas Holzinger en su artículo titulado Multimedia Learning Systems based on IEEE Learning Object Metadata (LOM) (Holzinger et al., 2001) define la estructura de un LOM en tres grandes partes un elemento ubicado en la parte superior llamado raíz (root) que contiene sub-elementos adicionales llamados ramas (branches) y estos a su vez tienen otros sub-elementos llamados hojas (leaves) que ya no poseen nada mas, ha esta estructura jerárquica se le conoce como árbol (tree), gráficamente se visualiza en la figura siguiente.

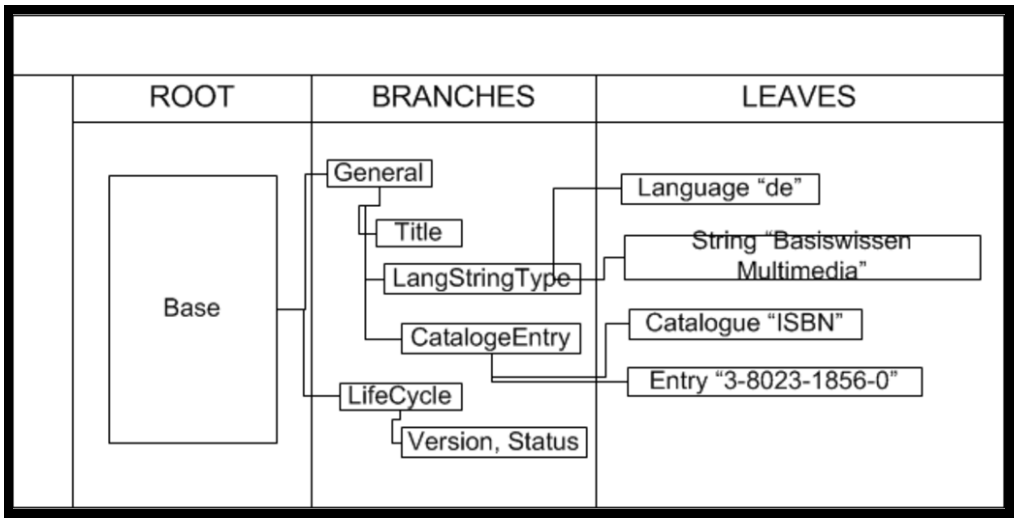


Figura 10 Modelo Jerárquico de un LOM dado por la IEEE (Holzinger et al., 2001).

2.2.9 Estructura de un sistema de gestión del aprendizaje

Según Roy McGreal (McGreal, 2004) las experiencias de aprendizaje o grupos de clases son consideradas como módulos. Un módulo suele estar compuesto por lo menos de 10 horas de aprendizaje. Cuando las lecciones son de más de 10 horas o si consisten en más de un módulo se consideran como un curso. Un grupo de cursos que conducen hacia un certificado o diploma es considerado como un programa, La figura siguiente muestra el esquema planteado.

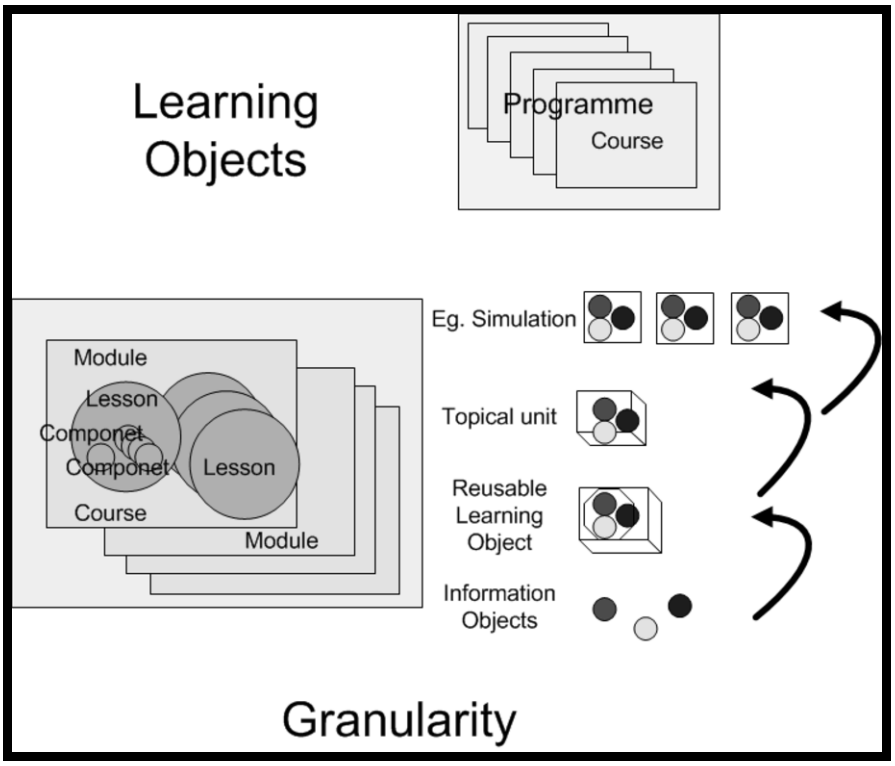


Figura 11 Granularidad de los objetos de aprendizaje (McGreal, 2004).

En el trabajo "Empaquetamiento y Visualización de Objetos de Aprendizaje SCORM en LMSs de Código Abierto" (ÁLVAREZ et al., 2006) realizan un empaquetamiento de contenidos bajo SCORM 1.2 con el programa ReLOAD y luego lo importan desde Dokeos, ATutor, ILIAS y Moodle, una de las conclusiones de este trabajo es que la utilización de editores de contenidos aún no resulta lo suficientemente fácil, debido a que los conceptos de objetos de aprendizaje y estándares no están muy difundidos, además el editor ReLOAD, está en inglés, y aunque existe la opción de cambiarlo a otros idiomas, es insuficiente puesto que modifica sólo algunos términos. Todo esto hace que la construcción, edición y empaquetamiento de contenidos digitales de aprendizaje bajo SCORM, aún siga siendo poco utilizado.

EduTools es una entidad que ofrece exámenes independientes a plataformas LMSs, comparaciones entre LMSs y servicios de consultoría para ayudar a la toma de decisiones en el e-learning respecto a que LMS seleccionar (EduTools, 2010), realizando el respectivo filtro en su portal la Tabla que a continuación se muestra es una comparación respecto a los subtemas Instructional Design Tools e Instructional Standards Compliance que pertenecen al área de Content Development Tools, allí se evidencia la existencia de herramientas de diseño que ayudan a los instructores en la creación de secuencias de aprendizaje y la adopción de estándares en los LMS previamente seleccionados, pero igualmente se ve que por mas adopción de estándares que tengan es su implementación varían generando problemas de compatibilidad entre ellos, para finalizar este capítulo en la Wiki (PBworks, 2007)

precisamente hablan del tema que compete trabajar “el problema de Integración de componentes del LMS, interoperabilidad entre múltiples LMS”.

Tabla 3 Sistema de Comparación de productos CMS (EduTools, 2010).

Product Name	Claroline 1.8.1	dotLRN/OpenACS	Moodle 1.9	Sakai Community Release 2.5	ATutor 1.6.3
Developer Name	Claroline	dotLRN	Moodle	Sakai	Adaptive Technology Resource Centre University of Toronto
Content Development Tools					
Instructional Design Tools		Instructors can organize learning objects, course tools, and content into learning sequences that are reusable.	Instructors can organize learning objects, course tools, and content into learning sequences that are reusable.	Instructors can organize learning objects, course tools, and content into learning sequences that are reusable.	Instructors can organize learning objects, course tools, and content into learning sequences that are reusable.
			Instructors can create linear learning sequences organized hierarchically by course, lesson, and topic.	Instructors can create linear learning sequences organized hierarchically by course, lesson, and topic.	Instructors can create linear learning sequences organized hierarchic

					ally by course, lesson, and topic.
			Instructors can reuse courses as templates for future lessons.	Instructors can reuse courses as templates for future lessons.	Instructors can reuse courses as templates for future lessons.
Instructional Standards Compliance	IMS Content Packaging 1.1.3	IMS Content Packaging 1.1.3	AICC	IMS Content Packaging 1.1.4	IMS Content Packaging 1.1.3
	IMS Content Packaging 1.1.4	IMS Content Packaging 1.1.4	IMS Content Packaging 1.1.3	IMS QTI 1.2.1	IMS Content Packaging 1.1.4
	IMS QTI 2.0	IMS QTI 1.2.1	IMS Content Packaging 1.1.4	IMS QTI 2.0	IMS QTI 1.2.1
	SCORM 1.2	IMS Enterprise 1.1	IMS QTI 1.2.1	SCORM 1.3	IMS Metadata 1.2.2
	SCORM 1.3	SCORM 1.2	IMS Enterprise 1.1		IMS Metadata 1.3
			SCORM 1.2		SCORM 1.2
			SCORM 1.3		

En conclusión y basados en el estudio anterior se ha detectado que el problema de interoperabilidad entre las diferentes plataformas LMS es causado básicamente por dos situaciones:

- La interpretación de los estándares por parte de las diferentes plataformas LMS, quienes realizan su propia adecuación a su arquitectura de dichos módulos, causando que los

mismos módulos sean construidos de diferentes formas y a pesar de ello cumpliendo el mismo objetivo al final.

- La diversidad de tecnologías y arquitecturas que se emplean en las diversas plataformas LMS, por ejemplo mientras moodle trabaja con una arquitectura modular y está construida con PHP, sakai utiliza la orientación a servicios como su eje fundamental y java como lenguaje de programación.

2.3 Herramientas de soporte para el modelamiento de sistemas de gestión del aprendizaje

2.3.1 Mapas mentales o de conocimiento

Los mapas mentales, desarrollados por Novak (NOVAK and GOWIN, 1984), se usan como un medio para la descripción y comunicación de conceptos dentro de la teoría de asimilación, una teoría del aprendizaje que ha tenido una enorme influencia en la educación (AUSUBEL et al., 1978). La teoría está basada en un modelo constructivista de los procesos cognitivos humanos. El mapa mental es la principal herramienta metodológica de la teoría de asimilación para determinar lo que el estudiante ya sabe. De acuerdo con Novak y Gowin, los mapas mentales han ayudado a personas de todas las edades a examinar los más variados campos de conocimiento en ambientes educativos (AUSUBEL et al., 1978)

El mapa mental es una representación gráfica de un conjunto de conceptos y sus relaciones sobre un dominio específico de conocimiento, construida de tal forma que las interrelaciones entre los conceptos son evidentes. En este esquema, los conceptos se representan como nodos rotulados y las relaciones entre conceptos como arcos rotulados conectándolos. De esta forma, los mapas mentales representan las relaciones significativas entre conceptos en forma de proposiciones o frases simplificadas: dos o más conceptos ligados por palabras para formar una unidad semántica.

Los mapas mentales son muy utilizados para el modelamiento en ingeniería del software trabajos como (Brine et al., 2007, Byrd et al., 1992, Canas et al., 2005, Canas et al., 2003, Chien et al., 2008, NOVAK and GOWIN, 1984, Polansky, 2010, Rodriguez et al., 2009, Roy and Ieee, 2008) lo han empleado de manera satisfactoria, además es un excelente insumo a la hora de construir ontologías como afirma (Byrd et al., 1992)

Para visualizar los módulos que componen a cada LMS se opto por realizarlo mediante la representación de mapas mentales o de conocimiento, ya que es una manera formal de realizar el análisis de requerimientos en la ingeniería de software como se dice en (Byrd et al., 1992). La herramienta para la construcción de los mapas

debería cumplir un requisito esencial; Debe permitir hacer mapas mentales o de conocimiento que se puedan visualizar gráficamente para su posterior análisis, la herramienta debe ser de libre distribución y multiplataforma.

Considerando el anterior requisito se encontraron dos herramientas que suplían esta necesidad, ellas fueron FreeMind (Polansky, 2010) y CMaptools (CmapTools, 2010), de la cuales vamos a hablar a continuación:

2.3.1.1 Herramienta FreeMind (FreeMind, 2010)

Según (FreeMind, 2010) FreeMind es una herramienta que permite la elaboración de mapas mentales o de conocimiento. Es útil en el análisis y recopilación de información o ideas generadas en grupos de trabajo. Es la alternativa libre a la aplicación MindManager fabricada por la empresa MindJet. Con freemind es posible generar mapas mentales y publicarlos en internet como páginas html, java o insertarlos dentro de wikis como Dokuwiki mediante la configuración de un plugin.

Es una herramienta que se ha utilizado en trabajos como (Chien et al., 2008) en el cual se presenta una representación totalmente visual basado en mapas mentales para representar los pasos de cálculo para una calculadora lambda puro sin tipo, VLM.

2.3.1.2 Herramienta CMaptools (CmapTools, 2010)

Es un aplicación desarrollado por el "Institute for Human and Machine Cognition" (IHMC), de la Universidad de West Florida (Estados Unidos), se diseñó con el objeto de apoyar la construcción de modelos de conocimiento representados en forma de "Mapas mentales". Sin embargo, también pueden elaborarse con él "Telarañas", "Mapas de Ideas" y "Diagramas Causa-Efecto", todos dentro de un entorno de trabajo intuitivo, amigable y fácil de utilizar.

CMaptools es una herramienta que se ha utilizado en proyectos como(Rodriguez et al., 2009) el cual es un caso de estudio real en Panamá que abarca más de 700 escuelas con el objetivo de compartir el uso de las TIC's (Tecnologías de la Información y las Comunicaciones) como una herramienta para la colaboración en la creación, intercambio y difusión del conocimiento, o (Roy and Ieee, 2008) que asegura que los mapas de conocimiento y el software asociado son herramientas eficaces para comprender los detalles técnicos de lo que se analiza, o (Brine et al., 2007) en el cual se utiliza CmapTools como herramienta para generación de de mapas dentro de Moodle para mejorar la lectura y escritura del inglés como lengua extranjera, o (Canas et al., 2005) en el cual se presenta el software CmapTools como un ejemplo de cómo los mapas de conocimiento son una herramienta de visualización del conocimiento que se puede combinar con la tecnología reciente para proporcionar una integración entre el conocimiento y la visualización de la información, o (Canas et al., 2003) en el cual recomiendan una selección apropiada de palabras, tanto para los conceptos y frases

de enlace, ya que esta es la clave para una representación del conocimiento exacto de la comprensión del usuario del dominio.

2.4 Conceptos sobre ontologías

Es mucha la información que se encuentra respecto a la temática de ontologías con los cual respecto a las definiciones de esta temática aremos referencia al documento "Desarrollo de Ontologías-101: Guía Para Crear Tu Primera Ontología" (Natalya and Deborah, 2005) que es una guía oficial proporcionada por la Universidad de Stanford y creadora de Protege (Stanford Center for Biomedical Informatics Research, 2010), que es la herramienta más utilizada en la creación de ontologías.

La literatura de inteligencia artificial contiene varias definiciones de ontología; muchas de ellas contradicen otras. Para los propósitos de este trabajo una ontología es una descripción explicita y formal de conceptos en un dominio de discurso (clases (a veces llamadas conceptos)), propiedades de cada concepto describiendo varias características y atributos del concepto (slots (a veces llamados roles o propiedades)), y restricciones sobre los slots (facetas (algunas veces llamados restricciones de rol)). Una ontología junto con un conjunto de individuos de clases constituye una base de conocimiento. En realidad, hay una línea muy delgada donde la ontología termina y la base de conocimiento empieza. Las clases son el centro de la mayoría de las ontologías. Las clases describen conceptos de un dominio.

2.4.1 Por qué utilizar ontologías

Las ontologías ofrecen varias características útiles para los sistemas inteligentes en el proceso de la ingeniera del conocimiento. Estas características se analizarán a continuación.

2.4.1.1 Vocabulario en una ontología

Una ontología proporciona un vocabulario para referirse a los diferentes términos de una determinada área. Las ontologías están formadas por afirmaciones lógicas que describen el significado de los términos del vocabulario y como se relacionan de tal forma que permitan extender el vocabulario. Una ontología especifica términos cuyo significado no es ambiguo, de modo que se usan semánticas independientes del lector y del contexto.

2.4.1.2 Taxonomía en una ontología

Una taxonomía es una clasificación de entidades en un determinado dominio. Una buena taxonomía separa sus entidades en grupos y subgrupos mutuamente excluyentes y no ambiguos.

Toda ontología proporciona una taxonomía en un formato apto para ser leído y procesado por una maquina. Por otro lado, una ontología es más que su

correspondiente taxonomía, es una especificación completa de un dominio. El vocabulario y taxonomía incluidos dentro de la ontología proporcionan un framework conceptual útil para el análisis y la recuperación de información en un determinado dominio.

2.4.1.3 Reutilización del intercambio del conocimiento

El principal objetivo de las ontologías no es servir como un vocabulario o una taxonomía, sino que es ser utilizadas como fuente para intercambio y reutilización de conocimiento por parte de las aplicaciones. Lo fundamental es que toda ontología proporciona una descripción de los conceptos y las relaciones que existen en un dominio, que pueden ser intercambiadas y reutilizadas por otras aplicaciones.

2.4.2 Áreas de aplicación de las ontologías

Existen multitud de posibles aplicaciones para las ontologías, pero (Ciocoiu et al., 2001) indica una serie de áreas de aplicación clave:

Colaboración: Personas diferentes pueden tener puntos de vista diferentes sobre un determinado problema al trabajar en un proyecto de equipo. Esto es realmente importante en áreas interdisciplinarias, donde conviven especialistas de diferentes áreas de conocimiento. Para estos especialistas las ontologías proporcionan un esqueleto de conocimiento unificado que puede ser utilizado como una referencia común y unificada.

Interoperabilidad: Las ontologías permiten la integración de información proveniente de sistemas diferentes. Ya que a un usuario final no le interesa saber de dónde proviene la información que maneja, sino obtener la información que necesita y sacarle el máximo rendimiento. Las aplicaciones distribuidas pueden tener que acceder a diferentes formatos y niveles de detalle. Sin embargo, si todos los sistemas reconocen las mismas ontologías, la conversión de los datos y la integración de los datos son más fáciles de realizar automáticamente.

Educación: Como las ontologías son el resultado de un amplio consenso acerca del dominio que representa, son una fuente de información fiable y objetiva para aquellos que quieren aprender sobre un dominio.

Modelo: Las ontologías pueden servir como bloques de conocimiento prefabricados que pueden ser utilizados por muchas aplicaciones. Por ejemplo una ontología sobre músicos pude ser utilizada por tanto sitios web de venta de música por internet, como por sistemas educacionales para encontrar información acerca de músicos.

2.4.3 Lenguajes de representación de ontologías

Existen numerosos lenguajes para representar ontologías, entre los principales se encuentran los siguientes:

- KIF (Genesereth and Ketchpel, 1994)
- Ontolingua (Patil et al., 1998)
- Loom (MacGregor, 1991)
- SHOE (Heflin et al., 1999)
- XOL (Karp et al., 1999)
- RDF (Lassila and Swick, 1998)
- RDF Schema (W3C, 2004)
- OIL (Fensel et al., 2001)
- DAM + OIL (Bechhofer et al., 2001)
- OWL (Motik et al., 2008)

2.4.4 El papel de las ontologías en la web semántica

Las ontologías posibilitan el acceso a una gran red de conocimiento que puede ser procesado y comprendido por los ordenadores. Una vez que el conocimiento esencial de un dominio ha sido publicado en la Web en forma de ontologías interconectadas, se crea una solida base para el desarrollo de sistemas inteligentes en ese dominio ya que reduce el problema de la adquisición del conocimiento. Como se puede ver en la siguiente figura, gracias a las relaciones entre las ontologías, una aplicación que utilice una determinada ontología tiene acceso también al resto de ontologías relacionadas.

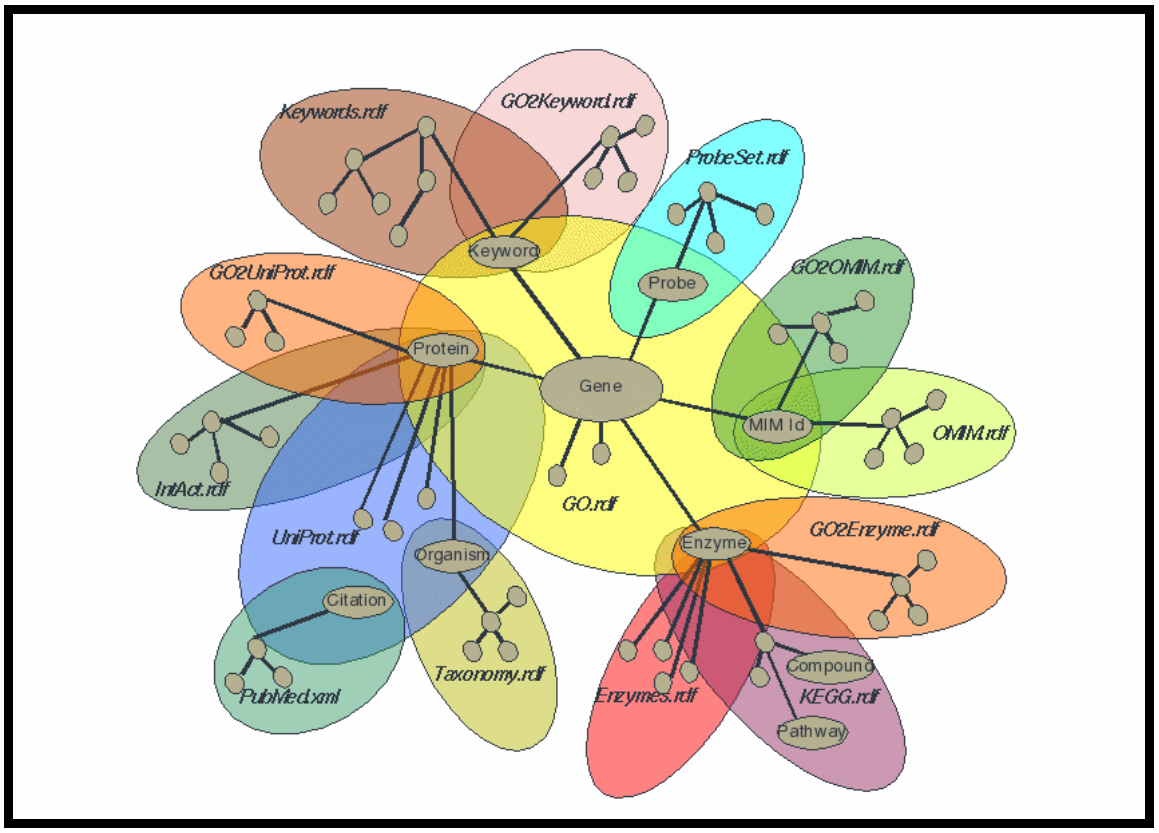


Figura 12 Ejemplo de Ontologías interconectadas biopax (Berners-Lee, 2005).

En la arquitectura de la Web semántica las ontologías juegan diversos papeles:

- Permiten que el conocimiento en la Web pueda ser procesado y compartido entre aplicaciones.
- Establecen nuevos niveles de interoperabilidad (interoperabilidad semántica) en la Web en términos de relacionar los conceptos con los datos.
- Permiten la aparición de sistemas inteligentes (agentes de búsqueda, gestores de conocimiento, etc).

Con las ontologías, la Web Semántica proporcionan un nuevo nivel de servicio ya que la Web se convierte en un sistema muy grande con diversos servicios de razonamiento. Las ontologías ofrecen una infraestructura para convertir la Web de información y datos en una Web de conocimiento (Web Semántica).

2.5 Lenguajes de la web semántica

Lo que se conoce como “Semantic Web Layer Cake” ilustra la arquitectura de la Web Semántica. En este esquema se muestra la jerarquía de los lenguajes donde cada lenguaje utiliza la sintaxis y semántica de los lenguajes de capas inferiores. Todos los

lenguajes de la Web Semántica están utilizando la sintaxis de XML, como por ejemplo RDF, RDFS o OWL, la siguiente imagen muestra lo que se acaba de de plantear.

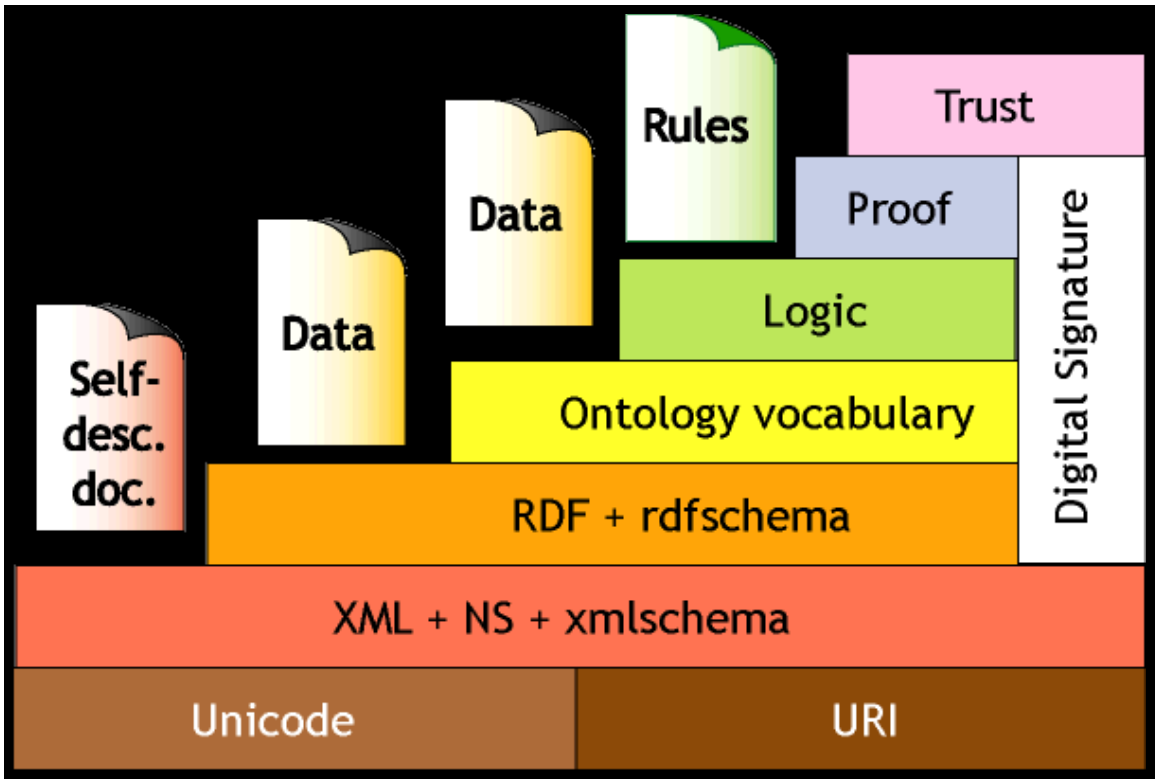


Figura 13 Capas de la Web Semántica presentada por Tim Berners-Lee "Semantic Web Layer Cake".

2.5.1 Lenguaje de etiquetado extensible (XML) y esquema de XML (XML Schema)

Los lenguajes actuales de la web semántica tienen una sintaxis basada en XML. Generalmente los documentos XML (eXtensible Markup Language) permiten la especificación de documentos que pueden ser leídos por ordenadores. XML no implica una interpretación de los datos específica, la información solo se codifica en una sintaxis no ambigua, pero su uso y la semántica no están especificados. Es decir, XML se ocupa únicamente de la estructura de un documento y no de la interpretación común de dicho documento. XML proporciona un formato de los datos destinado a componer documentos estructurados, pero no especifica un vocabulario.

El uso de XML entre aplicaciones de intercambio de datos requiere ponerse de acuerdo en el vocabulario, su uso y el significado de sus términos. Este acuerdo se puede conseguir parcialmente mediante el uso de XML Schema, el cual provee un mecanismo para especificar la estructura de un documento XML. El esquema describe las diversas etiquetas, elementos, atributos, estructura y tipos de datos de un determinado XML.

2.6 Web semántica, ontologías y la relación con los metamodelos

La parte de la informática que estudia las ontologías es la Web semántica. Existen muchas definiciones sobre la Web semántica como *"The Web more understandable by machines" (Heflin and Hendler, 2001) o "Semantic Web is about building an appropriate infrastructure for intelligent agents to run around the Web performing complex actions for their users*" (Hendler, 2001). También la definen como *"Semantic Web is about explicit declaring the knowledge embedded in main Web-based applications, integrant information in a intelligent way, providing semantic-based access to the internet, and extracting information from text"* (Asunci et al., 2002) ó *"Semantic web is about how to implement reliable, large-scale interoperation of web services, to make such services computer interpretable, to create a Web of machine-understandable and interoperable services that intelligent agents can discover and compose automatically"* (McIlraith et al., 2001). La razón por la cual existen tantas definiciones sobre la Web Semántica es por la heterogeneidad de la Web, con lo cual las definiciones se adaptan acorde a los diferentes contextos donde se empleen. En la actualidad mucha de la información en la Web es presentada mediante un lenguaje natural que las maquinas no pueden comprender, la Web semántica es necesaria para expresar la información de una manera precisa e interpretable por las maquinas.

Es mucha la información que se encuentra respecto a la temática de ontologías, como hemos dicho para las definiciones de esta temática aremos referencia a (Natalya and Deborah, 2005), que es una guía oficial proporcionada por la Universidad de Stanford y creadora de Protege (Stanford Center for Biomedical Informatics Research, 2010) que es la herramienta más utilizada en la creación de ontologías.

La literatura de inteligencia artificial contiene varias definiciones de ontología; muchas de ellas contradicen otras. Para los propósitos de este trabajo y como hemos expuesto anteriormente, una ontología es una descripción explicita y formal de conceptos en un dominio de discurso (clases (a veces llamadas conceptos)), propiedades de cada concepto describiendo varias características y atributos del concepto (slots (a veces llamados roles o propiedades)), y restricciones sobre los slots (facetas (algunas veces llamados restricciones de rol)). Una ontología junto con un conjunto de individuos de clases constituye una base de conocimiento. En realidad, hay una línea muy delgada donde la ontología termina y la base de conocimiento empieza. Las clases son el centro de la mayoría de las ontologías y describen conceptos de un dominio. Una ontología se define como la especificación formal de un vocabulario de conceptos y axiomas sobre ellos (Wouters, 2005, Chandrasekaran et al., 1999).

La razón de utilizar una ontología para modelar el conocimiento de las plataformas LMS, es que las ontologías proveen todas las características de vocabulario y taxonomía que se necesitan para especificar un dominio del conocimiento.

Los modelos son una abstracción acotada de la realidad, ellos representan un punto de vista particular, necesariamente deben ser acotados y omiten detalles según el punto de vista, por ejemplo (Guizzardi, 2005) define formalmente los modelos como *"A model is an abstraction of reality according to a certain conceptualization"*. Y el metamodelo es simplemente un modelo del lenguaje de modelamiento. En (Pardillo and Cachero, 2010) muestran la existencia de dos metamodelos: UML y DI (Diagram Interchange) que cubren diferentes propósitos. En (García-Magariño et al., 2009) se plantea que el metamodelo es un mecanismo para definir la sintaxis abstracta de un lenguaje de modelamiento, que determina la validez de los modelos en esos lenguajes. La OMG (Object Management Group) explica la relación entre modelos y metamodelos como la definición de un lenguaje de modelado bajo la arquitectura de MOF (Meta Object Facility) (Group, 2007).

La afirmación que *"an ontology is a meta-model describing how to build models"* (Devedzi\ and \#263, 2002), relacionan directamente a las ontologías con los modelos, porque son la descripción de dichos modelos. Así que con esta definición se concluye que desde una ontología se puede obtener un metamodelo; para este proyecto será el metamodelo LMS.

2.7 Ingeniería dirigida por modelos (MDE)

La ingeniería dirigida por modelos surge como la respuesta de la ingeniería de software a la industrialización del desarrollo de software, MDA es la propuesta de la OMG, que centra sus esfuerzos en reconocer, que la interoperabilidad es fundamental y el desarrollo de modelos permite el desarrollo de otros modelos que luego al ser juntados proveerían la solución a todo un sistema e independiza el desarrollo de las tecnologías empleadas (Stephen et al., 2004).

Se ha visto como el nivel de abstracción de los diferentes lenguajes de programación ha incrementado durante décadas, esto se puede considerar una evolución en los lenguajes de programación, así antes se hablaba de lenguajes en Binario, luego del ensamblador, mas adelante de los lenguajes procedimentales, posteriormente de la orientación a Objetos, esta ultima apoyada en diversos artefactos como UML y ahora hablamos de construcción de esos modelos como UML, a esto se le ha llamado MDA. Y aunque el nivel de abstracción aumente, este se acerca más al dominio especifico sobre el cual se trabajara, así cualquier persona podría llegar a realizar un modelo empleando los conocimientos de su contexto y la complejidad se traslada a ver como ese modelo se convierte en una solución desplegable y funcional sobre cualquier tecnología especifica.

Una buena interpretación de lo que es un modelo, un metamodelo y un meta-metamodelo se encuentra en (García Díaz and Cueva Lovelle, 2010), allí un metamodelo son esas herramientas que permite la creación de un modelo, que es una descripción de uno o varios elementos del dominio o mundo real y finalmente el meta-

metamodelo describe a esos metamodelos planteados, generando un grado de abstracción supremamente alto en el cual coinciden todos los modelos, vea la siguiente figura.

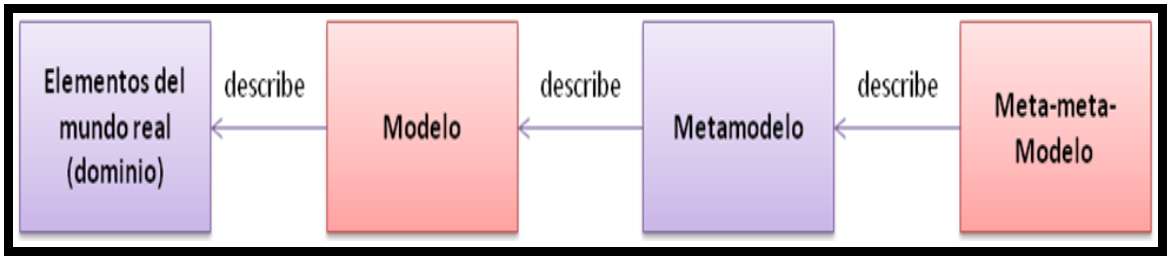


Figura 14 Modelo, metamodelo y meta-metamodelo.

Según (García Díaz and Cueva Lovelle, 2010) básicamente hay cuatro espacios de modelamiento, los niveles base M0 que son los elementos del mundo real, los niveles M1 que son las clases UML, los programas informáticos, entre otros, los niveles M2 que sería la especificación de UML, ODM, Java, C#, XML u otras, y finalmente están los niveles M3 que son los de mayor abstracción. Básicamente hay dos meta-metamodelos, planteados por un lado esta MOF y por el otro EBNF, como se muestra en la siguiente figura.

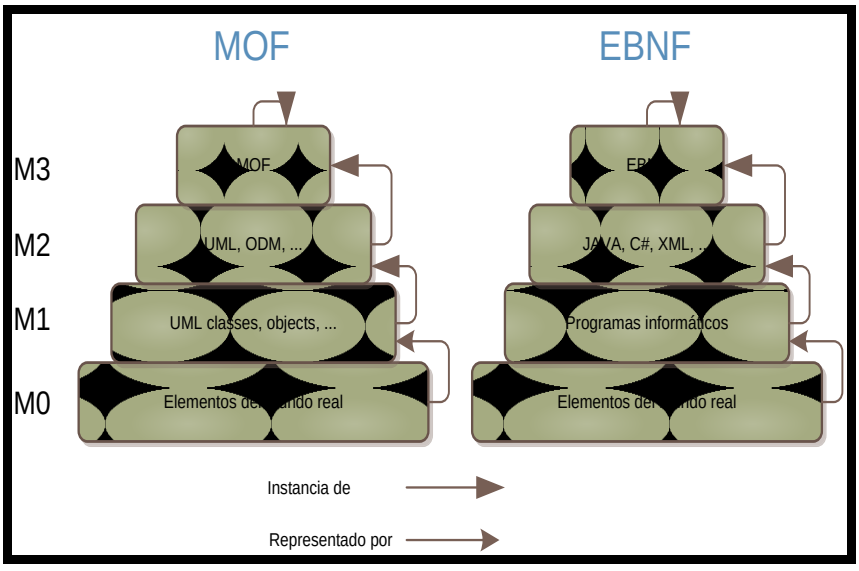


Figura 15 Espacios de Modelado.

La idea de generar estos niveles de abstracción tan altos, es proveer un mecanismo común que permita a través de transformación de un modelo a otro la interoperabilidad de los sistemas, una definición explicita de lo que es MDA, la encontramos a continuación:

"The Model Driven Architecture (MDA) is a framework for software development defined by the Object Management Group (OMG). Key to MDA is the importance of models in the software development process. Within MDA the software development

process is driven by the activity of modeling your software system." (Kleppe et al., 2003)

MDA se centra en la generación de modelos, por ello su ciclo de vida se enfoca hacia la creación de modelos y no modifica para nada el ciclo tradicional de vida de desarrollo de software, tan solo que la generación de artefactos cambia, y algunos de ellos como el paso de un PIM (Modelo Independiente de la Plataforma) a PSM (Modelo Especifico de la Plataforma) y luego a código, se realiza de manera automática y recibe el nombre de transformaciones a partir de los modelos generados, de tal forma que si hay un cambio en el sistema, se realiza en el modelo más global, y para desplegar la solución se vuelve a repetir el proceso de trasformaciones entre modelos, en la siguiente figura se muestra el ciclo de vida del desarrollo de software con MDA.

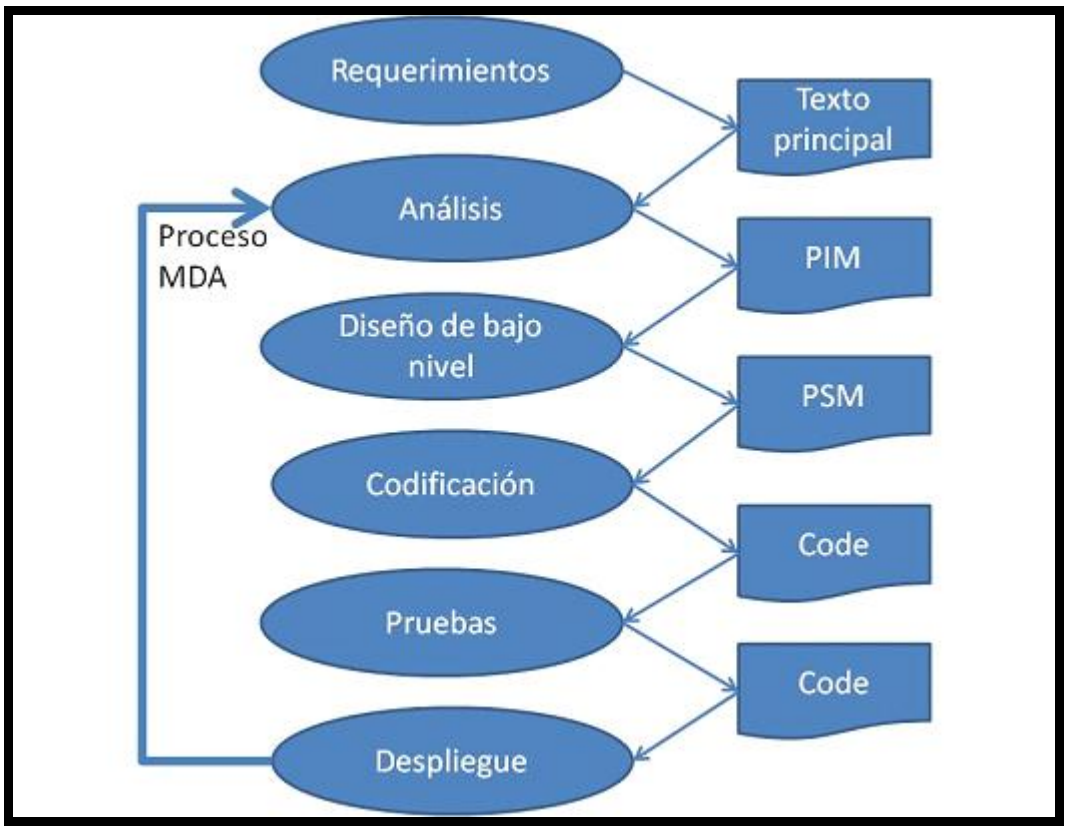


Figura 16 Ciclo de vida del desarrollo de software con MDA (Kleppe et al., 2003).

El artefacto que reúne los requerimientos del sistema se llama el CIM (Modelo Independiente de la Computación), el resultado de modelar este sistema es un PIM que se hace a través de de un DSL (Domain Specific Language) construido previamente. Este DSL genera a través de un proceso de transformación un PSM, que por ultimo y nuevamente a través de otra transformación se convierte en código desplegable o modelo especifico de implementación (ISM), este proceso se visualiza en la siguiente figura.

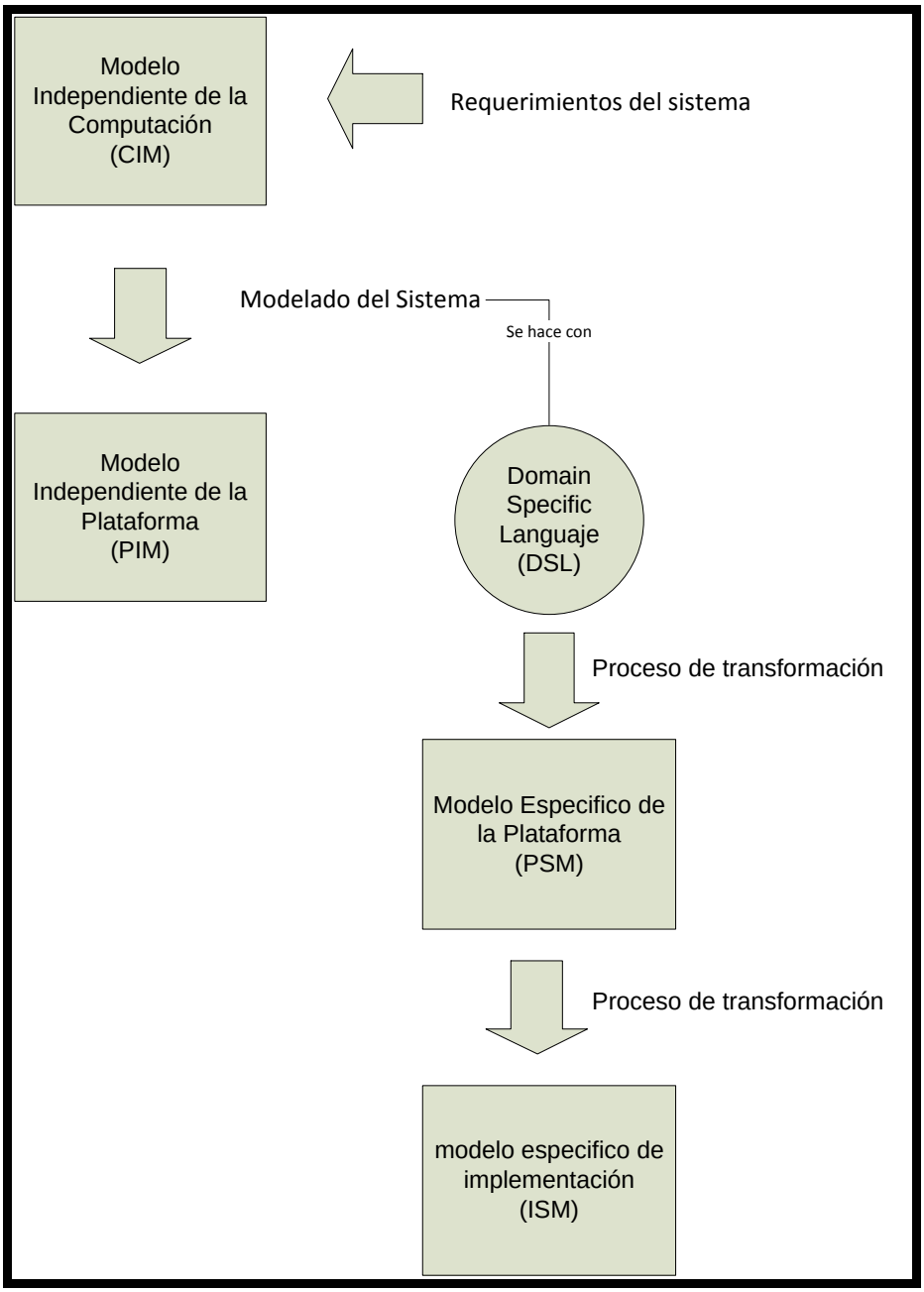


Figura 17 Proceso de despliegue de una solución completa con MDE.

2.7.1 Metamodelos

Los metamodelos (Atkinson and Kuhne, 2003) son uno de los pilares del Desarrollo de Software Dirigido por Modelos (DSDM). Sin embargo, cuando se trabaja con modelos se suelen encontrar dificultades en la definición de los metamodelos por la gran variedad de representaciones posibles para los mismos y la ausencia de manuales que guíen su definición.

2.7.1.1 Metamodelos y arquitectura de metadatos

Un modelo es una abstracción de la realidad, que la presenta de una manera simplificada. Un metamodelo es un modelo que describe un Lenguaje de Modelado (LM), con el que se describen otros modelos.

La noción de metamodelo se basa en la arquitectura de metadatos que se muestra en la figura de abajo, adoptada por el consorcio Object Management Group (OMG) en la especificación del Meta-Object Facility (MOF) (Group, 2007). Esta figura muestra la diferencia entre información, modelos, metamodelos y meta-metamodelos. MOF nombra cada una de estas capas o niveles con los nombres M0, M1, M2 y M3 respectivamente, como se puede ver en la figura de abajo. De ahora en adelante, se usarán los términos M0, M1, M2 y M3 para referirse a las capas de MOF. Es importante fijarse en que estos conceptos están en capas diferentes y poseen significados diferentes aunque usen la misma notación.

El nivel más alto es el nivel de los meta-metamodelos, M3. Algunos ejemplos de lenguajes en este nivel son MOF (Group, 2007) usado por OMG, ECore (Budinsky et al., 2009) usado por EMF, y GOPRR (Grafo, Objeto, Propiedad, Relación, y Rol) (Tolvanen et al., 1993). Las instancias de los lenguajes de M3 son los metamodelos, que definen LMs y corresponden al nivel M2. Un ejemplo bien conocido de metamodelos es la definición de UML (OMG, 1999). Luego están los modelos que representan, por ejemplo, diseños de aplicaciones, estos son instancias de los metamodelos y constituyen el nivel M1, cualquier diseño concreto de una aplicación en UML podría servir como ejemplo. Por último se encuentra el nivel M0, que sería la implementación de los diseños, por decirlo en términos de programación Orientada a Objetos, en el nivel M0 se considerarían los objetos de las clases en ejecución y en el nivel M1 estarían las definiciones de las clases de objetos.

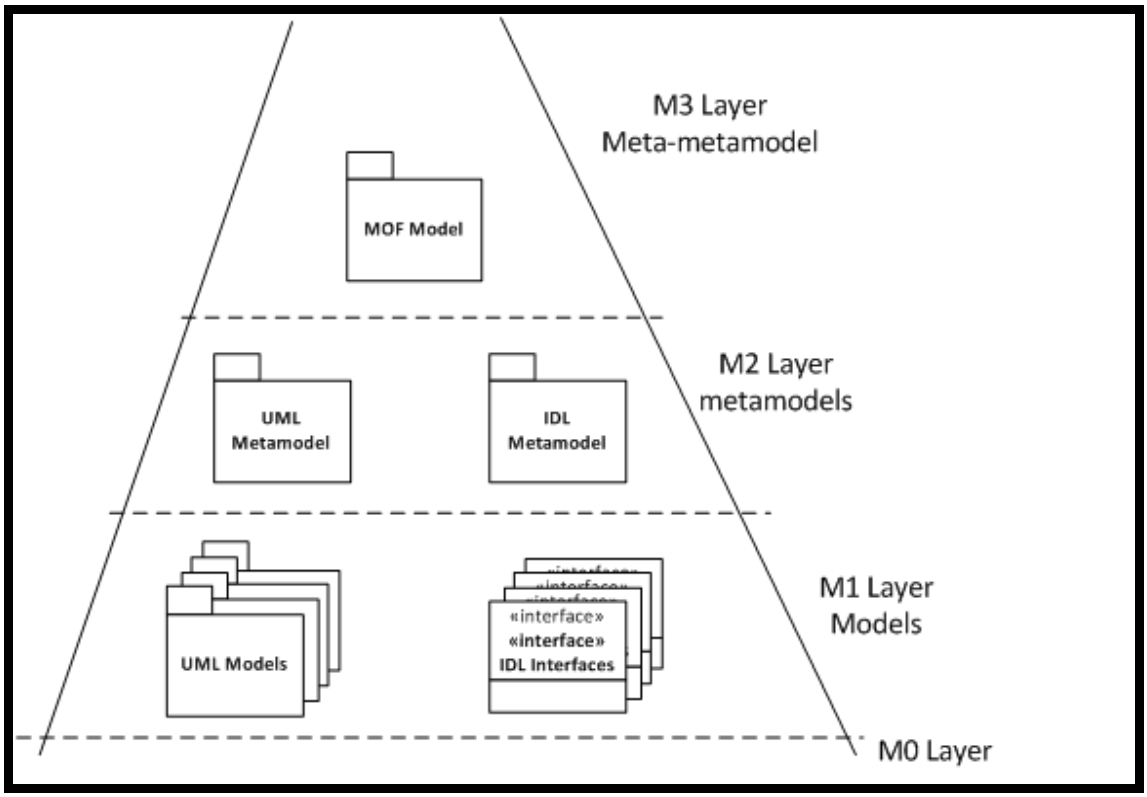


Figura 18 Arquitectura de Metadatos en MOF.

2.7.2 Especificación para el intercambio de modelos usando XML (XMI)

El estándar de la OMG XMI (Group, 2007) es una especificación basada en XML para compartir metadatos MDA. Actualmente XMI es la base para conseguir interoperabilidad entre herramientas de apoyo a MDA, una de las grandes ventajas de XMI es que sobre él se pude almacenar todos los modelos basados en MOF utilizando un mismo esquema. Sin embargo, tiene inconvenientes, como su complejo formato para ser comprendido por humanos o que muchas herramientas que trabajan con XMI no se ajustan a las especificaciones de la recomendación.

2.7.3 Espacios de modelado

Un espacio de modelado o Modeling Space (MS) es una determinada arquitectura de modelado basada en un determinado meta-metamodelo. Todos los conceptos de las capas inferiores (empezando por la M0), se definen mediante los conceptos de capas superiores hasta llegar a la última capa, la cual contiene el meta-metamodelo que es autodefinido. Si el meta-metamodelo se definiese con conceptos de otro metamodelo entonces no sería un meta-metamodelo.

La siguiente figura muestra una serie de MS conocidos. El más difundido tal vez es MOF MS, que se define mediante el meta-metamodelo MOF y a su vez define varios metamodelos como UML y ODM. Otro ejemplo de MS es el que tiene EBNF(Anastasakis et al., 2007) como meta-metamodelo par definir gramáticas libres

de contexto. Debajo de EBNF estarían los lenguajes como Java, C#, XML y por ultimo en la capa M1 estarían los programas o ficheros escritos en esos lenguajes.

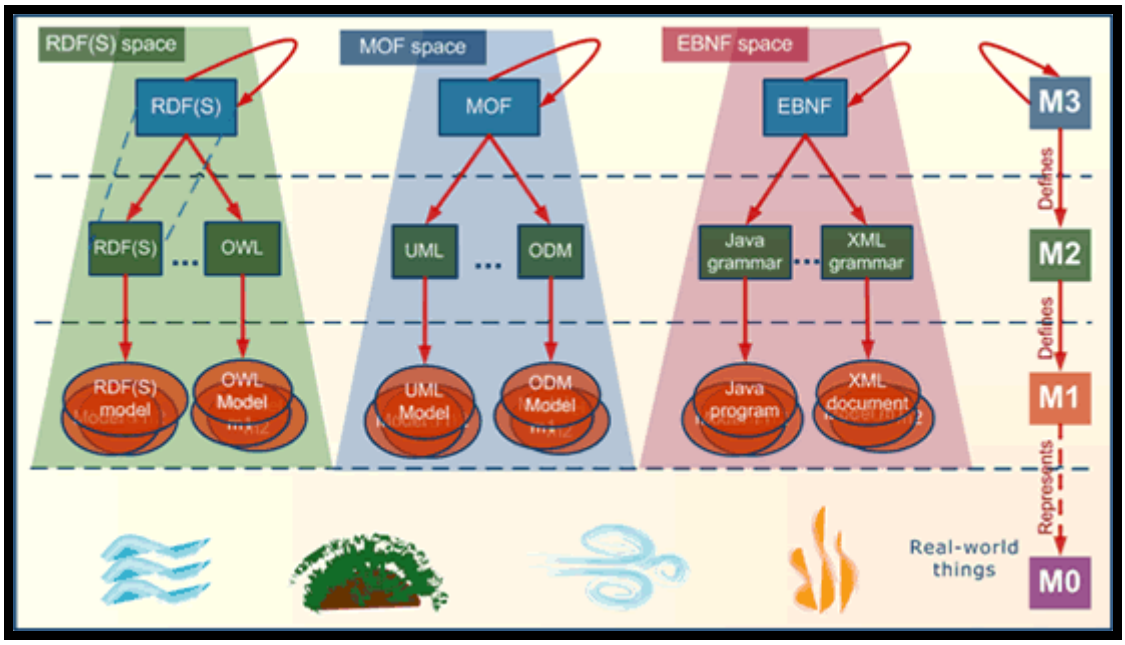


Figura 19 Espacios de modelado RDF, MOF Y EBNF.

2.7.4 Espacio técnico

Los Espacios de modelado son un concepto inspirado en los Espacios técnicos o Technical Spaces (TS) (Jouault and Bézivin, 2006). Se puede definir un TS como un contexto de trabajo que incluye varios MS relacionados. Por ejemplo un TS podría ser MDA TS, el cual incluye al MOF MS. Pero también incluye parcialmente a otros MS como EBNF MS para darle soporte a la construcción de los archivos en formato XMI.

2.7.5 Transformaciones entre modelos (M2M)

Una transformación de modelos significa convertir un modelo de entrada (o un conjunto de modelos) descritos por un determinado metamodelo, en otro modelo descrito por un segundo metamodelo, como se muestra en la siguiente grafica.

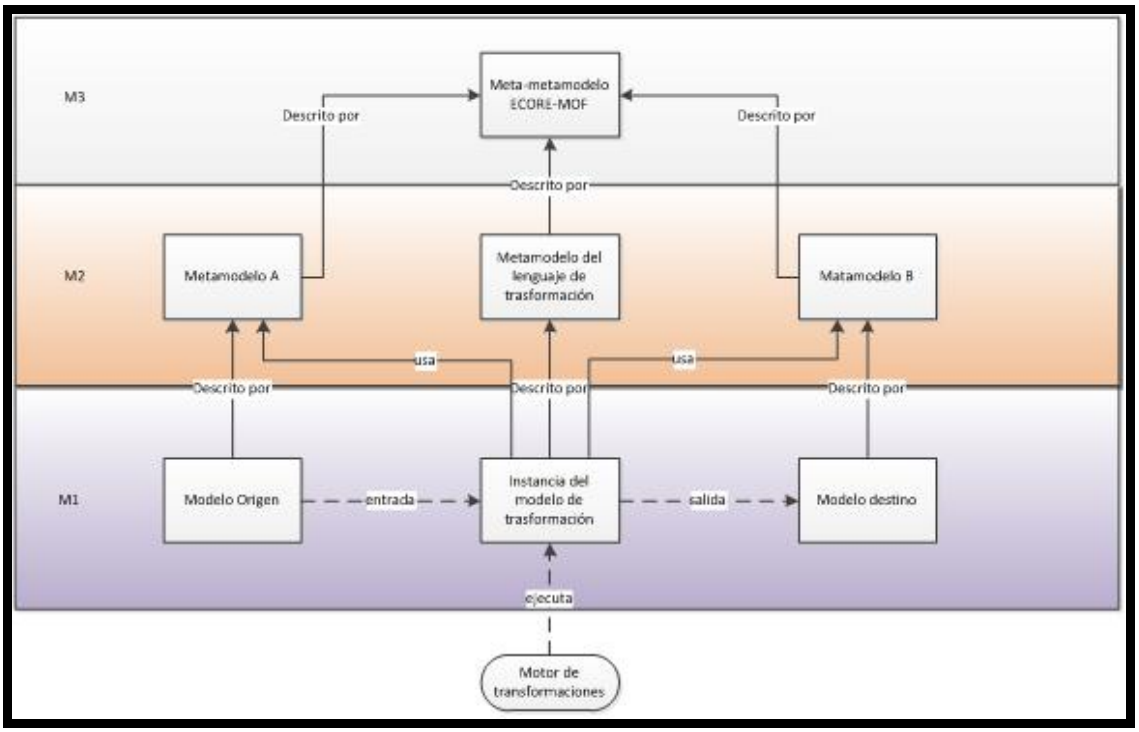


Figura 20 Transformaciones entre modelos.

Esta conversión se realiza definiendo una serie de reglas que concuerdan con los elementos del modelo de entrada y producen elementos del modelo de salida. Las transformación en sí misma es un modelo descrito por un determinado metamodelo de transformación (Jouault et al., 2008).

2.7.5.1 Clasificación de los lenguajes de transformaciones

Un lenguaje de transformación es declarativo si las definiciones de las transformaciones escritas en dicho lenguaje especifican relaciones entre los elementos del modelo origen y destino, sin importar el orden de ejecución. Las relaciones se pueden especificar en términos de funciones o funciones de inferencia. Por otro lado un lenguaje de transformaciones es imperativo si especifica una determinada secuencia de pasos para ser ejecutados en orden con la finalidad de producir un determinado resultado. Puede existir una categoría intermedia conocida lenguajes de transformaciones híbridos, los cuales tiene una mezcla de contracciones declarativas e imperativas.

Algunos lenguajes permiten especificar las definiciones de las transformaciones en un único sentido: de un modelo origen a un modelo destino. A estos lenguajes se les conoce como unidireccionales. Otros lenguajes (normalmente declarativos) permiten especificar las definiciones en los dos sentidos, se les conoce como lenguajes bidireccionales. Esta característica es importante cuando se necesita que dos modelos estén sincronizados.

Un escenario típico toma un modelo origen y produce un modelo destino como salida (transformaciones 1-1). Generalmente existen tres casos más: 1-N, N-1 y N-M, el más general es el de 1-N donde se toma un modelo de origen y se generan varios modelos (por ejemplo de un modelo se puede generar código java y ficheros XML para intercambio de datos).

Por último se pueden definir tres tipos de transformaciones (García Díaz and Cueva Lovelle, 2010):

- Refactoring transformations: Manejan la reorganización de un modelo base de acuerdo a una serie de criterios bien definidos. En este caso el modelo destino es una revisión del modelo original.
- Model-to-model transformations: Convierten la información de un modelo origen (o un conjunto de modelos) a un modelo destino (o un conjunto de modelos).
- Model-to-text tranformations: Convierten cada elemento del modelo origen en definiciones de texto.

2.7.5.2 Lenguajes de transformaciones

Básicamente, existen dos lenguajes para las transformaciones entre modelos: QVT que ha sido creado por la OMG y ATL que se enmarca dentro de EMF.

QVT

Query View Transformation (Object Management Group, 2011) es un estándar propuesto por la OMG para resolver el problema de la transformación de modelos.

Define tres abstracciones cuyas letras iníciales forman el acrónimo que le da su nombre:

- Query: O consulta es una expresión que se evalúa sobre un modelo. Los resultados de aplicar esta expresión sobre un modelo son una o varias instancias de los tipos definidos en el modelo transformado, o en el propio lenguaje de consulta.
- View: O vista es un modelo obtenido a partir de otro modelo base.
- Transformation: O transformación es una operación que obtiene un modelo (destino) a partir de un modelo (fuente).

A partir de las abstracciones anteriores se definieron dos tipos de lenguaje. Uno de consulta, que cubriría las vistas y consultas, y otro de transformaciones, que se corresponde con la abstracción del mismo nombre. Estos lenguajes deben estar definidos como metamodelos de MOF y todos los modelos manipulados por las transformaciones deben ser instancias de ese metamodelo.

QVT está compuesto de un conjunto de lenguajes estándar de dominio específico (DSL) que propone la OMG para la transformación de modelos. Las transformaciones que se definen están basadas en metamodelos MOF.

Los lenguajes que componen esta especificación son:

- Core: Es el lenguaje declarativo de menor nivel de abstracción dentro de QVT. Soporta reconocimiento/coincidencia de patrones sobre un conjunto de variables.
- Relations: Es un lenguaje declarativo de QVT. Proporciona formas de declarar restricciones que deben satisfacer los elementos de los modelos candidatos.
- Operational Maping: es un lenguaje imperativo de QVT. Proporciona extensiones OCL con efectos laterales que permiten una programación de tipo procedimental y una sintaxis concreta que es más familiar para los programadores acostumbrados a este tipo de lenguajes de programación.

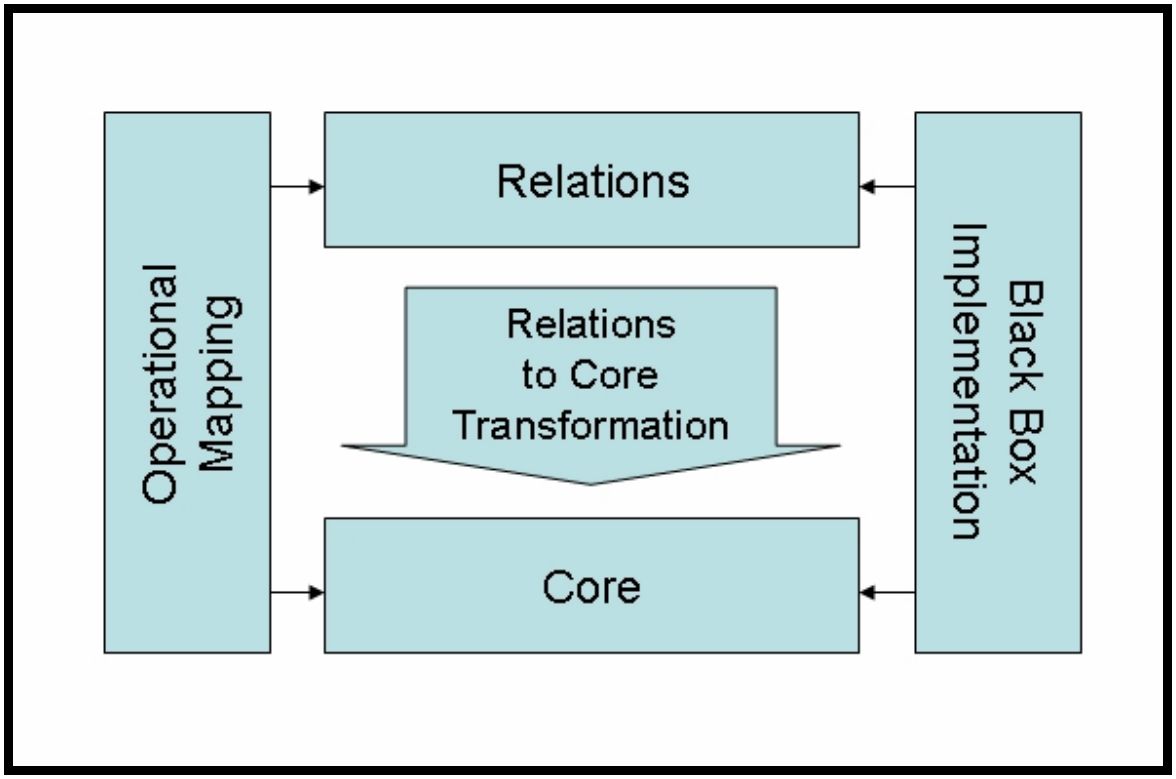


Figura 21 Arquitectura de QVT.

Cabe destacar que QVT se centra en transformaciones Model-to-model quedando las Model-to-text fuera del estándar.

ATL

El Atlas Tranformation Language (ATL) es un lenguaje para transformaciones de modelos, especificando un metamodelo y una sintaxis concreta de tipo textual. En el contexto de MDE, ATL proporciona una forma de especificar como producir un conjunto de modelos destino a partir de un conjunto de modelos origen.

ATL es un lenguaje hibrido, aunque el estilo preferido es el declarativo, por que posibilita el expresar mapeos entre los elementos del modelo destino y los del modelo origen de forma sencilla. Por otro lado, se proporcionan construcciones imperativas para poder especificar aquellos mapeos que difícilmente son expresables mediante construcciones declarativas.

Una transformación ATL se compone de reglas que definen sobre qué elementos del modelo origen se aplican y que elementos del modelo destino se producen.

2.7.6 Lenguajes de dominio específico (DSL)

Muchos de los problemas que aparecen durante el desarrollo del software surgen una y otra vez de forma repetitiva, además en muchas ocasiones estos problemas pertenecen a un dominio concreto. Para dar solución a estos problemas se puede utilizar un GPL (General Purpose Language) como es Java y C# o se puede recurrir a la utilización de un DSL (Domain Specific Language). Un DSL no es más que un lenguaje definido específicamente para resolver los problemas de un dominio concreto.

A partir del contexto de DSL se puede hablar de DSM (Modelado de Dominio Especifico) que tiene su origen en la existencia de muchos desarrollos de software similares para un mismo dominio, una parte común y una parte variable. La parte común podría desarrollarse utilizando las técnicas de desarrollo tradicionales y la parte variable podría desarrollarse utilizando un DSL. Un ejemplo seria la realización de sistemas para la realización de encuestas, que pueden compartir un mismo motor y una misma base de datos pero que tiene que adaptarse a las diferentes encuestas. Como la única parte variable es la encuesta se podría definir un DSL para definirlas.

La unificación de la parte fija y la parte variable se puede hacer mediante dos aproximaciones:

- Interpretativa: La parte es interpretada mediante un intérprete que se encuentra en la parte común. Esta aproximación proporciona una mayor flexibilidad pero acarrea inconvenientes como la perdida evidente de rendimiento.

- Generativa de código. La parte común y la variable se juntan y se compilan para generar una solución como un todo. Es una aproximación más complicada de realizar pero evita las desventajas de la aproximación interpretativa.

Los DSL intercambian generalidad por expresividad en un dominio limitado. Ofrecen notaciones y construcciones orientadas a un dominio particular, presentando ganancias sustanciales en términos de expresividad y facilidad de uso en comparación con los GPL para el dominio en cuestión. Estas ganancias se corresponden en mejoras de la productividad y reducción de los costes de mantenimiento. Algunos lenguajes de dominio específico se listan a continuación:

- EBNF: Especificación de sintaxis.
- Exel: Hojas de cálculo.
- HTML: Páginas Web de hipertexto.
- Make: Construcción de software.
- Matlab: Matemáticas Computacionales.
- SQL: Consultas a bases de datos.
- VHDL: Diseño de hardware.

2.7.6.1 Tipos de lenguajes de dominio específico (DSL)

Se pueden distinguir tres clasificaciones para los DSL:

Desde un punto de vista de la construcción del lenguaje en:

1. Internos: Utilizan un determinado lenguaje anfitrión para darle la apariencia de otro lenguaje concreto. Un ejemplo claro son lo que actualmente se conoce como Fluent Interfaces (Freeman and Pryce, 2006).
2. Externos: Tiene su propia sintaxis y es necesario un parser para poder procesarlos. Un ejemplo claro de DSL externo es SQL.

Desde el punto de vista del formato del lenguaje:

1. Textuales: La mayoría de los lenguajes informáticos son textuales y están formados por un conjunto ordenado de sentencias. Un ejemplo muy conocido de DSL textual es SQL utilizado para realizar consultas a una base de datos. Una forma de crear DSLs textuales es mediante la creación de una determinada gramática (por ejemplo utilizando EBNF) y posteriormente crear o utilizar un parser par dicha gramática, para en etapas posteriores poder interpretar el DSL o generar código.
2. Gráficos: En los últimos años están ganando gran aceptación los lenguajes gráficos, podrían citarse como ejemplo UML. La creación de un lenguaje grafico es similar a la de un lenguaje textual, la única diferencia es que en lugar

de usar texto para representar los conceptos, se utilizan conectores y figuras simples.

Desde el punto de vista del dominio del problema:

1. Horizontales: Los DSL horizontales son aquellos en los que el cliente que utilizará el lenguaje no pertenece a ningún dominio específico. Un ejemplo son los editores visuales de entornos de desarrollo que permiten generar interfaces de usuario automáticamente (por ejemplo Windows Forms de visual Studio).

2. Vertical: A diferencia de los DSL horizontales, el cliente que utilizará el lenguaje pertenece al mismo dominio que el lenguaje en sí. Como en el ejemplo anterior para un lenguaje de definición de encuestas, los usuarios finales serían los expertos en estadística encargados de definir dichas encuestas.

2.7.6.2 Partes de un lenguaje de dominio específico (DSL)

Los lenguajes de dominio específico son el elemento principal de cualquier solución de dominio específico. El lenguaje se define como un metamodelo, una notación específica y generalmente una herramienta que lo soporta para facilitar la usabilidad. La idea básica es no utilizar conceptos de lenguajes de programación de propósito general. En su lugar, se utilizan conceptos y reglas de dominio.

La sintaxis abstracta de un lenguaje especifica su estructura, es decir, las construcciones, propiedades y conectores que pueda tener dicho lenguaje. La sintaxis concreta es necesaria para especificar la notación específica con la que los usuarios del lenguaje podrán utilizarlos. Idealmente cada concepto del dominio y del lenguaje se mapearan a una representación en la notación especifica. Es importante reseñar que una misma sintaxis abstracta podría tener diferentes sintaxis concretas.

2.7.7 Ingeniería dirigida por modelos (MDE) con eclipse

2.7.7.1 Meta-metamodelo ecore de eclipse

El meta-metamodelo que se empleara será Ecore, este se encuentra en el paquete org.eclipse.emf.ecore y es la especificación más alta que existe en la pirámide de los modelos (M3) de la siguiente figura, sobre ella se construirá el metamodelo del proyecto, la especificación de Ecore se puede consultar en (others, 2006).

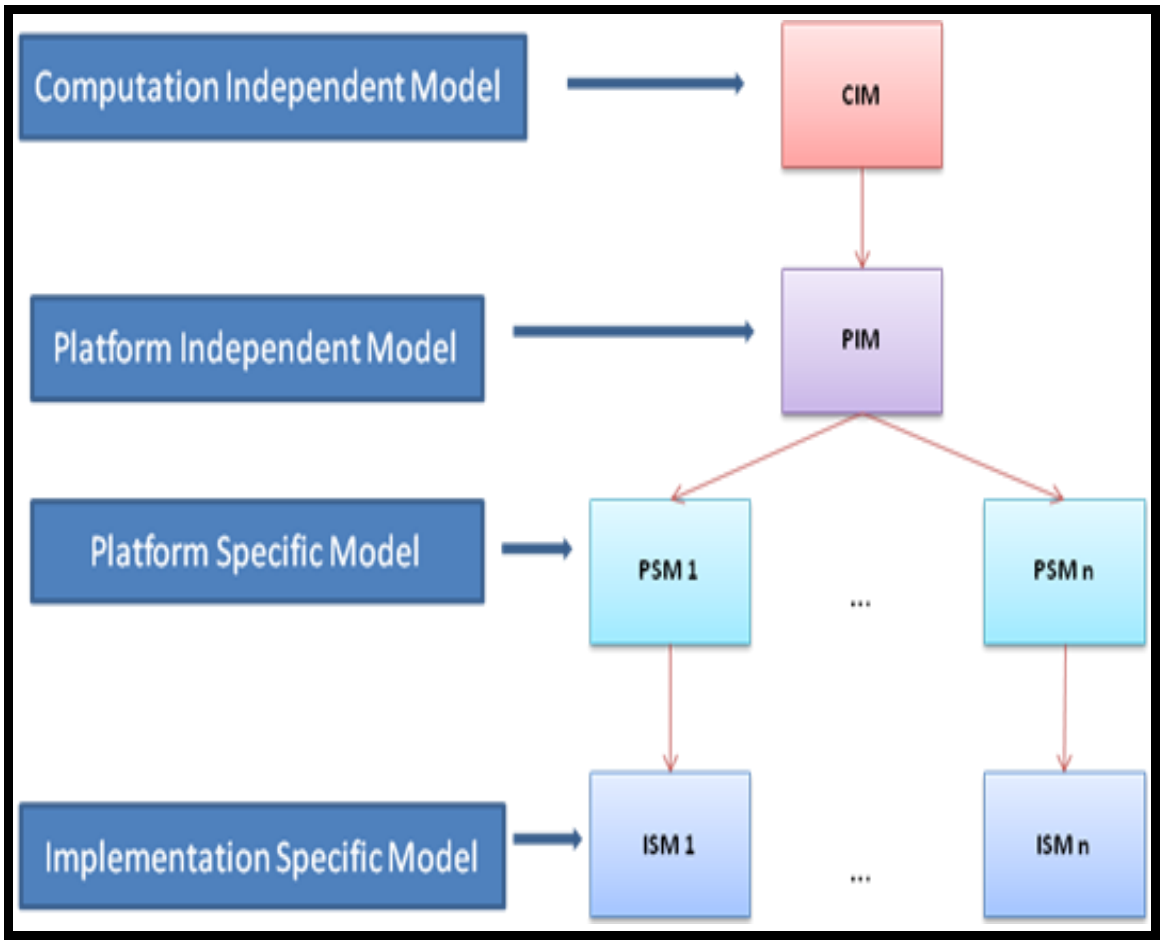


Figura 22 Modelos generados en MDA.

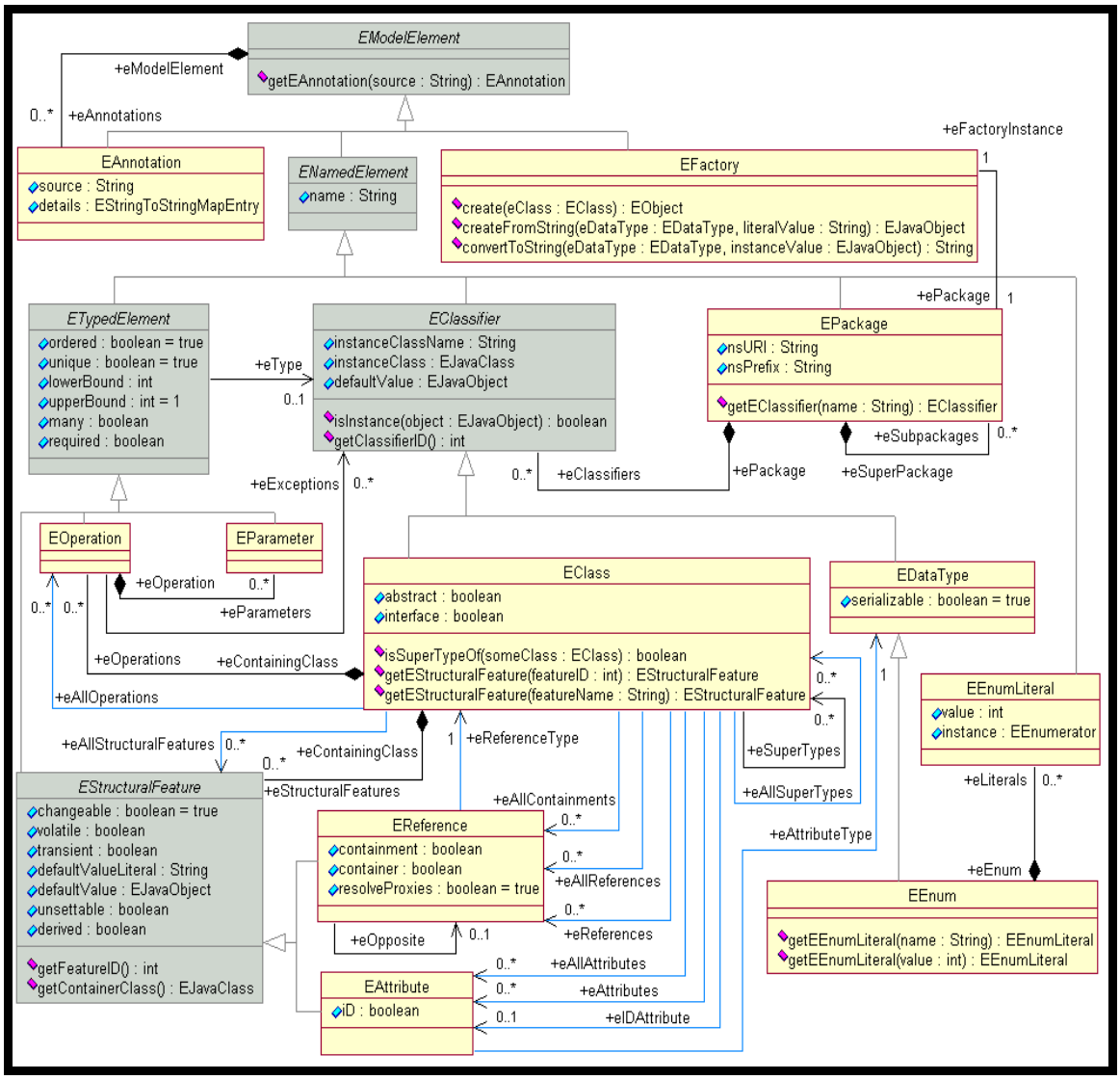


Figura 23 Modelo Ecore con sus relaciones, atributos y operaciones.

2.7.7.2 El Metamodelo

Para la utilización de Ecore en Eclipse es necesario tener instalado el plugin de EMF (Eclipse Modeling Framework), este plugin provee básicamente dos herramientas para construir un modelo basado en Ecore, una el Ecore Model que es un editor manual que funciona en un estilo de árbol de navegación para la creación del modelo basado en Ecore, la otra es el Ecore Diagram siendo un editor grafico similar a las herramientas graficas para la creación de diagramas de clases UML. Cualquiera de las dos forma que se utilice para crear el diagrama basado en Ecore, genera un fichero XMI(Group, 2007) (XML Metadata Interchange) que es una especificación para el intercambio de diagramas.

2.7.7.3 Construcción del editor para el modelo o lenguaje de dominio específico (DSL)

Como se está trabajando bajo Eclipse, para esta etapa se debe emplear GMF Tooling (Graphical Modeling Framework Tooling) que hace parte del proyecto GMP (Graphical Modeling Project)(Foundation, 2010c). El proceso para la construcción del DSL Grafico se visualiza en la siguiente figura.

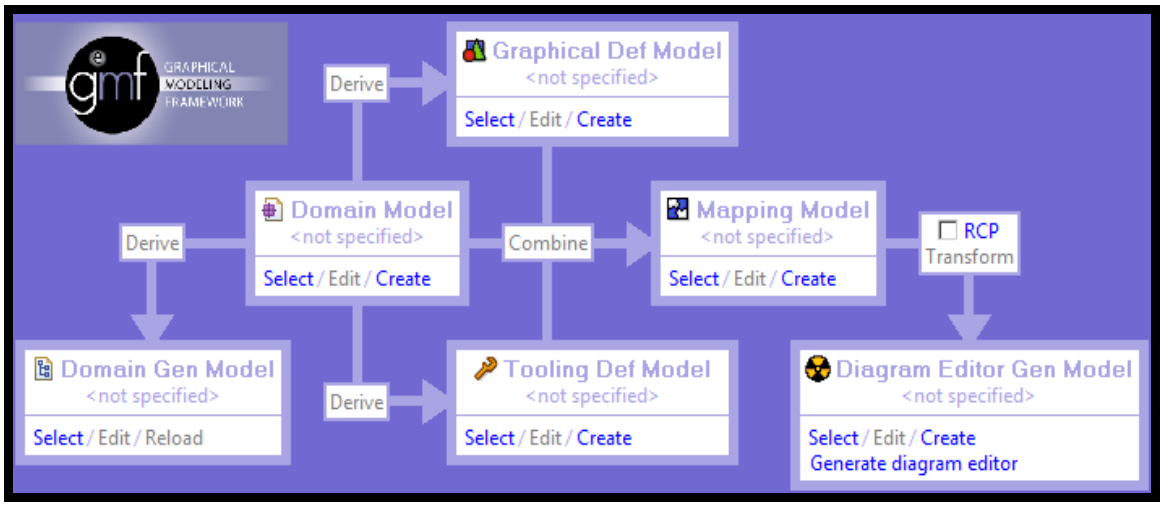


Figura 24 GMF Overview (Foundation, 2010b).

2.7.7.4 El Modelo

El diagrama que se obtenga como resultado de emplear el editor de DSL, tendrá asociado un fichero XMI(Group, 2007) que basará su sintaxis en el metamodelo creado.

2.7.7.5 Proceso de generación de código

En este último paso, existen varias tecnologías que se integran a Eclipse, como son Acceleo(Obeo, 2010), Jet(Foundation, 2010d), Xpand(Foundation, 2010f) y MOFScript(Foundation, 2010e), todas emplean el mismo principio, la creación de reglas de transformación basándose en un metamodelo. Estas reglas serán aplicadas al modelo, para generar código en el lenguaje deseado, este tipo de tecnologías reciben el nombre M2T o Model to Text. La transformación de un modelo a otro se llama de M2M o Model to Model, algunas herramientas como ATL(Foundation, 2010a) o OperationalQVT, realizan esta tarea.

Finalmente aplicando las tecnologías de transformación sobre el modelo creado se obtiene el código en el formato definido para su despliegue.

3 Desarrollo de la propuesta

3.1 Estudio de los módulos comunes para sistemas de gestión del aprendizaje

3.1.1 Clasificaciones de los sistemas de gestión del aprendizaje

Para la clasificación taxonómica de los componentes de los LMSs se utilizo un método experimental practico que consistió en trabajar con cada una de las plataformas y a través de su manejo determinar la taxonomía de sus componentes, para el trabajo de diagramación se decidió emplear un herramienta que proveyera facilidades de representación grafica a través de mapas y que fuese de libre distribución, en esta etapa básicamente se estudiaron dos; FreeMind(Polansky, 2010) (Chien et al., 2008) y CMaptools(CmapTools, 2010), finalmente se opto por CMapTools ya que ofrece más facilidad para el trabajo colaborativo Web, diversidad de formatos que posee para la exportación del mapa y una buena cantidad de proyectos como (Rodriguez et al., 2009) (Roy and Ieee, 2008) (Brine et al., 2007) (Canas et al., 2005) (Canas et al., 2003), trabajados con ella. Para el trabajo con esta herramienta se definieron los siguientes patrones:

- Las relaciones entre el LMS y los primeros elementos son de “recursos”.
- Conceptos: Los nombres de los conceptos siempre deben empezar por una letra en mayúscula, si el nombre se compone por más de una palabra la segunda, tercera y n-necima palabra debe escribirse toda en minúscula, para realizar la descripción del componente, esta se debe realizar en el mismo componente después de su nombre y entre paréntesis ejemplo: “Este es un concepto (Descripción del componente)”. Los conceptos identifican módulos funcionales del LMS a describir y todos los conceptos que sean hojas (conceptos terminales) deben corresponder a elementos de la interfaz grafica que el usuario accederá directamente, estos se clasificaron según la siguiente tabla.

Tabla 4 Conceptos hojas (terminales) en un LMS.

Caja de texto	Explorador	Dato calendario	Evento
Menú desplegable	Vista	Área de texto	Vinculo
Área de texto no editable	Botón	Botón de opción	Paleta de colores

Árbol	Botón de chequeo	Lista	Calendario

- Palabra de Enlace: Los nombre de las palabras de enlace siempre deben de ir en minúscula y sin espacios, ejemplo: “esteesunenlace”. Se definieron tres tipos de enlaces acordes a las siguientes premisas: Todos los nodos hojas deben ser de algún tipo. Los recursos y subrecursos deben estar compuestos por otros subrecursos o por nodos hoja. Los recursos solo corresponderán a las primeras instancias del LMS, de allí en adelante se llamaran subrecursos. En la siguiente tabla se clasifican los enlaces.

Tabla 5 Palabras de enlace para clasificación de LMS.

Palabra de enlace	Descripción
recurso	Los recursos solo corresponderán a las primeras instancias del LMS
subrecurso	Relaciones intermedias que no sean ni primera instancias ni nodos hoja.
tipo	Todos los nodos hojas deben ser de algún tipo

- Los eventos de Hacer (aceptar, ok, commit) y Cancelar (cancel) están en todas la acciones por lo cual no se modelaron, pero se deben considerar.

- Ya que todos los LMSs ofrecen diversidad de rolles, en todos se trabajara con el usuario profesor (teacher) o su equivalente.

3.1.2 Creación del mapa de conocimiento por cada sistema de gestión del aprendizaje

A continuación se mostraran los resultados finales de los mapas de cada LMS a fin de conocer su estructura genérica, los mapas se encuentran en el anexo I y las fuentes de cada mapa se encuentran en el anexo digital (+My Cmaps) a este documento, allí se encontrara dos tipos de mapas uno en el cual se describe al LMS y otro en el cual a cada nodo terminal se le agrega un descriptor señalando que acciones realiza ese nodo terminal. También se determinaran los módulos que maneja cada LMS basado en el conocimiento adquirido que el manejar la herramienta proveyó, finalmente se determinaran los módulos comunes que existen entre los LMS estudiados.

3.1.3 Mapa de conocimiento para ATutor

ATutor emplea como lenguaje principal para su desarrollo PHP, el montaje para su prueba se realizo con un servidor Web Apache, PHP y un motor de base de datos MySQL, el navegador utilizado para las pruebas fue Firefox3.

ATutor ofrece una página principal que contiene los módulos de Mis cursos, Navegador por cursos, Perfil, Preferencias, Networking, Buzón de entrada, Buscar y Ayuda, estos módulos a su vez conducen a otros, los mapas resultantes se muestran en el anexo I.

Los módulos obtenidos de ATutor se visualizan en la siguiente Tabla, que muestra la jerarquía en árbol de cada uno, ósea quien pertenece a cada módulo.

Tabla 6 Módulos de ATutor.

Atutor			
Mis cursos			
	curso		
		Mis examenes y encuestas	
		Directorio	
			Grupos
		Foros	
		Networking	
			Galeria fotografica
		Mapa del sitio	
		Galeria fotografica	
		Herramientas del estudiante	
			Glosario
			Exportar contenidos
			Chat
			Enlaces
			Foros
			Encuestas
			Preguntas frecuentes
			Grupos
			Lista de lecturas
			Almacen de archivos
			Pruebas y tareas
			Galeria fotografica
			Directorio
			Busqueda en repositorio
			Mapa del sitio
			Mi seguimiento
		Administacion	
			Administacion de archivos
			Anuncios
			Chat
			Contenidos
			Copias de seguridad
			Encuestas
			Email del curso
			Estadisticas
			Examenes y encustas
			Foros
			Glosario
			Grupos
			Herramientas del curso
			Herramientas del estudiante
			Inscripcion
			Lista delecturas
			Preguntas frecuentes
			Propiedades
			Pruebas y tareas
			Tareas
	Creacion de cursos		
Navegador por cursos			
Perfil			
Preferencias			
Networking			
Buzonde entrada			
Buscar			
Ayuda			

3.1.4 Mapa de conocimiento para Claroline

Claroline emplea como lenguaje principal para su desarrollo PHP, el montaje para su prueba se realizo con un servidor Web Apache, PHP y un motor de base de datos MySQL, el navegador utilizado para las pruebas fue Firefox3.

Claroline en su inicio muestra una página principal que contiene los módulos de Evaluation system, Agenda, Página de inicio del curso, Descripción del curso, Anuncios, Documentos, Ejercicios, Secuencia de aprendizaje, Trabajos, Foros, Grupos, Wiki, Debates y Usuarios, los mapas que se obtuvieron para Claroline se muestran en el anexo I.

Los módulos obtenidos de Claroline se visualizan en la siguiente Tabla, que muestra la jerarquía en árbol de cada uno, ósea quien pertenece a cada módulo.

Tabla 7 Módulos de Claroline.

Claroline
Evaluation System
Agenda
Pagina de inicio del curso
Descripcion del curso
Anuncios
Documentos
Ejercicios
Secuencia de aprendizaje
Trabajos
Foros
Grupos
Wiki
Debate
Usuarios

3.1.5 Mapa de conocimiento para Moodle

Moodle emplea como lenguaje principal para su desarrollo PHP, el montaje para su prueba se realizo con un servidor Web Apache, PHP y un motor de base de datos MySQL, el navegador utilizado para las pruebas fue Firefox3.

Moodle en su inicio muestra una página principal que contiene los módulos de Entrada, Categorías de cursos, Usuarios en línea y Administración del sitio, los mapas que se obtuvieron para Moodle se muestran en el anexo I.

Los módulos obtenidos de Moodle se visualizan en la siguiente Tabla, que muestra la jerarquía en árbol de cada uno, ósea quien pertenece cada módulo.

Tabla 8 Módulos de Moodle.

Moodle		
Entrada		
	Agregar recurso	
		Insertar etiqueta
		Componer una pagina de texto
		Componer una pagina web
		Enlazar un archivo o una web
		Desplazar paquetes de contenidos ims
	Agregar actividad	
		Base de datos
		chat
		Consulta
		Custionario
		Encuesta
		Foro
		Glosario
		Leccion
		Scorm
		Subida avanzada de archivos
		Subir un archivo
		Actividad offline
		Wiki
		Categorias de cursos
		Usuario en linea
		Administracion en linea
	Categorias de cursos	
	Usuarios en linea	
	Administacion del sistio	

3.1.6 Mapa de conocimiento para DotLRN

DotLRN o .LRN utiliza un motor de Base de datos PostgreSQL, un servidor Web AOL Server (American On Line), sobre él se monta Open ACS (Architecture Community System) que es el corazón que ofrece funcionalidades a las capas superiores y finalmente esta .LRN, todo sobre un distribución Ubuntu server, el navegador utilizado para las pruebas fue Firefox3.

La página principal de este LMS contiene los módulos de Inicio, Cursos, Comunidades y Panel de control, los mapas de conocimiento que se obtuvieron se muestran en el anexo I.

Los módulos obtenidos de .LRN se visualizan en la siguiente Tabla, que muestra la jerarquía en árbol de cada uno, ósea quien pertenece a cada módulo.

Tabla 9 Módulos de DotLRN ó .LRN.

RLN					
Inicio					
	Mi portal				
		Grupos			
			Curso		
				Foros	
				Preguntas frecuentes	
				Noticias	
				Informacion del curso	
				Contenidos de aprendizaje	
				equipo docente	
				Subgrupos	
				Planificaciones	
				Calendario	
				Tareas	
				Cuestionario	
				Chat	
				weblogger	
			Calendario		
			Documentos		
			Material clase		
				Asignaciones	
					Tareas
					Proyectos
					Examenes
				Avaluaciones	
					Administrar mis tipos de asignacion
					Solicitar notificaciones
					Tareas
					Proyectos
					Examenes
			Administracion		
				Administracion de grupos	
				Forus administrator	
				Administracion de FAQ	
				Administracion de portkets personalizados	
				Administracion de tareas	
				Administracion de evaluacion	
				Administracion de chat	
				Xowiki Portlet administration	
				Administracion de documentos	
				Administracion de calendario	
				Administracion de noticias	
				Servicio de correo	
				Administracion de cuestionarios	
				Administracion de contenidos de aprendizaje	
				Administracion de weblogger	
		Foros			
		Preguntas frecuentes			
		Noticias			
		Agenda			
		Cuestionario			
		Chat			
		weblogger			
	Mi calendario				
	Mis documentos				
		Documentos			
Cursos					
Comunidades					
Panel de control					
	Mi cuenta				
	Su foto				
	Privacidad				
	Ayuda				

3.1.7 Mapa de conocimiento para Sakai

Sakai está basado en Java lo que lo hace multiplataforma, para su funcionamiento se recomienda instalar java SE 6, como contenedor de servlets se recomienda Apache Tomcat 5.5.30, el motor de base de Datos de MySQL 5.1 y el driver de conexión a la base de datos, el navegador utilizado para las pruebas fue Firefox3.

Sakai en su inicio muestra una página principal que contiene los módulos de Home, Mycommunications, My tools con Resources, Portfolios, News, Web content, Search, y Evaluation System, y My settings con Profile2, Membership, Preferences, Account, Site setup. Los mapas que se obtuvieron para Sakai se muestran en el anexo I.

Los módulos obtenidos de Sakai se visualizan en la siguiente Tabla, que muestra la jerarquía en árbol de cada uno, ósea quien pertenece a cada módulo.

Tabla 10 Módulos de Sakai.

Sakai	
Home	
	Menssage of the day
	My workspace information
	Recent announcements
	Calendar
	Message y forums notification
My communications	
	Calendar
	Announcements
	Help
My tools	
	Resources
	Portfolios
	News
	Web content
	Search
	Evaluation system
My settings	
	Profile2
	Membership
	Preferences
	Account
	Site setup

3.1.8 Comparativa entre los modelos y definición de módulos comunes.

Tomado como punto de partida los módulos de cada uno de LMS y basándose en la experiencia que proporciono el haber utilizado cada uno de ellos en modo profesor (teacher), se realizo un proceso de comparación entre los módulos de los LMS generando finalmente la siguiente tabla, que muestra una primera aproximación de los posibles módulos compatibles entre los LMS trabajados y un nombre genérico para ellos.

Tabla 11 Comparativa Módulos LMS.

Comparativa modulos LMS

Generico	Atutor	Claroline	Moodle	RLN	Sakai
Manejador de archivos	Administacion de archivos	Documentos	Agregar recurso	Documentos	Recursos
Anuncios	Anuncios	Anuncios	Entrada		Anuncios
Ayuda	Ayuda	Ayuda	Ayuda	Ayuda	Ayuda
Chat	Chat		Chat	Chat	
Contenidos curso	Contenidos		Agregar recurso	Material clase	Recursos
			Agregar actividad	Contenidos de aprendizaje	
Gestion del curso	Administrar	Secuencia de aprendizaje	Administracion del sitio	Panel de control	Portafolios
					Preferences
Cursos	Curso	Cursos claroline	Modulo	Curso	Portafolios
Usuarios	Directorio	Usuarios	Usuarios	Equipo docente	Cuentas
Encuentas	Encuestas		Encuesta		
Sistemas de evaluacion	Examenes y encuestas	Evaluation System	Cuestionario	Evaluación	Sistema de evaluacion
				Cuestionario	
Foros	Foros	Foros	Foro	Foros	Mensajes y foros
Glosario	Glosario		Glosario		
Manejo de grupos	Grupos	Grupos		Grupos	
Redes de trabajo colaboativo	Networking			Comunidades	Profile2
Preguntas frecuentes FAQ	Preguntas frecuentes			Preguntas frecuentes	
Calendario de actividades		Agenda	Actividades	Mi Calendario	Calendario
				Agenda	
Noticias				Noticias	Noticas
Wiki		Wiki	Wiki		
Trabajos	Pruebas y tareas	Ejercicios	Leccion	Evaluaciones	Evento
		Trabajos	Subida avanzada de arch	Examenes	
			Subir un solo archivo	Proyectos	
				Tareas	
Administracion de la plataforma	Preferencias	Si	Administracion del sitio	Panel de control	Membership (Administrador de sitios)

En esta tabla se visualiza que los módulos genéricos Manejo de archivos, Ayuda, Gestión de curso, Cursos, Usuarios, Sistemas de evaluación, Foros, Trabajos y Administración de la plataforma, en una primera comparativa existen en todos los LMS. Pero algunos otros módulos pueden ser homologables como es el caso del Chat con las herramientas de Profile 2 de Sakai y comunicaciones con el usuario de Claroline, o los anuncios con el calendario o las noticias en .RLN, o que en todos los LMS existe una forma de mantener comunicaciones con los usuarios fuera de línea a manera de un correo interno.

Todos los LMS tienen un núcleo que es el manejo de cursos y las actividades que allí se traten, y de alguna manera estas actividades están asociadas a un calendario.

Todos los LMS tienen diferentes usuarios pero esencialmente deben existir tres que son el administrador de la plataforma, el profesor y el estudiante, es muy importante aclarar que el profesor no debe ser el encargado de administrar la plataforma, él solo se encarga de gestionar su curso, se aclara ya que estas tareas se suelen confundir.

Todas las plataformas trabajan orientas a internet poseen una base de datos para el almacenamiento, y un servidor web para ser desplegadas.

Por más que exista cierto grado de similitud entre los módulos de las plataformas LMS, cada una implementa y maneja de una forma particular dichos módulos por ejemplo la gestión de usuario o los foros.

Este trabajo es la base fundamental y da vía libre a la posibilidad de construcción de la Ontología que permita modelar cursos para un LMS.

3.2 Ontología para sistemas de gestión del aprendizaje

Para la creación de la ontología se siguió la metodología planteada por (Natalya and Deborah, 2005) que es la guía oficial proporcionada por protege (Stanford Center for Biomedical Informatics Research, 2010) para la creación de ontologías, en esta guía definen una ontología "como *"is a formal explicit description of concepts in a domain of discourse (classes (sometimes called concepts)), properties of each concept describing various features and attributes of the concept (slots (sometimes called roles or properties)), and restrictions on slots (facets (sometimes called role restrictions)). An ontology together with a set of individual instances of classes constitutes a knowledge base"* (Natalya and Deborah, 2005) y propone los siguientes pasos en su construcción:

1. Determinar el domino y alcance de la ontología.
2. Considerar la reutilización de ontologías existentes.
3. Enumerar términos importantes para la ontología.

4. Definir las clases y la jerarquía de clases.
5. Definir las propiedades de las clases: slots.
6. Definir las facetas de los slots.
7. Crear instancias.

A continuación se va a desarrollar la metodología propuesta adaptada a las necesidades especiales del proyecto. Respecto a los estándares de codificación, se utiliza el mismo de la recomendación de Java (Microsystems, 1997).

3.2.1 Determinación del domino y alcance de la ontología

El dominio del problema se limita a la gestión de módulos de un LMS y especialmente de las plataformas LMSs seleccionadas, con un enfoque hacia la creación de cursos. La ontología se empleara parar obtener el conocimiento necesario respecto a los módulos que componen un LMS y la creación de cursos dentro de él, hay que tener muy presente que el objetivo es determinar los módulos comunes entre LMSs. La ontología podrá responder preguntas como ¿Un foro es parte de la herramienta LMS? ó ¿Un profesor es un tipo de usuario en un LMS?, es muy importante aclarar que la ontología se creara para un contexto de cambio y evolución constante, por lo tanto esta debe ser mantenible, actualizable y ampliable, acorde a las necesidad del contexto.

3.2.2 Consideración de la reutilización de ontologías existentes

Para considerar la reutilización de ontologías existentes para LMS, se realizo una búsqueda de ontologías con el fin de trabajar sobre ellas y enriquecerlas más, la búsqueda se llevo a cabo en los siguientes navegadores recomendados por (Natalya and Deborah, 2005):

- http://swoogle.umbc.edu/: Swoogle es un motor de búsquedas de ontologias para la web semántica (UMBC, 2007).
- http://protegewiki.stanford.edu/wiki/Protege_Ontology_Library: Es una wiki que contiene ontologías, las ontologías están organizadas por agrupaciones temáticas (Stanford Center for Biomedical Informatics Research, 2010)
- http://www.ksl.stanford.edu/software/ontolingua/: Ontolingua proporciona un entorno colaborativo distribuido para gran variedad de ontologías, allí se puede crear, explorar, modificar y utilizar ontologías. Algunos de los cuales han proporcionado la descripción de sus proyectos pero otros no. (University, 2005).
- http://www.daml.org/ontologies/: El DARPA Agent Markup Language (DAML) es un programa que se inicio oficialmente en agosto del 2000. El objetivo de

DAML es desarrollar un lenguaje y sus herramientas para facilitar el entendimiento de la Web semántica. Michael Pagels es el director del programa DARPA para DAML. El programa DAML termino a principios de 2006 (Office., 2000).

- http://www.unspsc.org/: la United Nations Standard Products and Services Code (UNSPSC) proporciona un estándar abierto y global, para la clasificación eficiente de productos y servicios a nivel multisectorial. La idea es buscar código en sitios Web para localizar códigos que puedan ser usados en nuestra compañía. La UNSPSC ofrece un único sistema de clasificación global, que se puede utilizar para la empresa en: Análisis de gastos, optimización de la relación costo-beneficio, Explotación completa de las capacidades del comercio electrónico. (Code, 2010).

- http://www.dmoz.org/: El Open Directory Project es el mayor y más completo directorio de la web, editado por humanos. Esta construido y es mantenido por una gran comunidad de editores voluntarios. (Netscape, 2002).

En la búsqueda de una ontología de módulos para plataformas LMS en los anteriores navegadores no se encontró ninguna, algunas ofrecen el concepto de Learning Management System (LMS), poro no el modelo de la ontología, por esta razón ninguna de las ontologías ofrecida se puede utilizar para el fin de este proyecto.

El siguiente paso en la búsqueda de una Ontología base para esta tesis, fue buscar en bases de datos académicas como ISI Web of Knowledge, CiteSeer, Ebsco, Dialnet, Proquest, Google Schoolar y otras similares, los resultados de estas búsquedas se resumen a continuación:

- A Learner Oriented Ontology of Metadata to Improve Effectiveness of Learning Management Systems: Este artículo presenta una ontología para e-Learning Management System (LMS), que organiza y relaciona los metadatos para objetos de aprendizaje relacionados con cursos académicos y usuarios. Esta ontología se ha incorporado como parte critica de la arquitectura propuesta. Con esta ontología se espera recuperar de manera óptima los objetos de aprendizaje y personalizar los LMS. Los metadatos utilizado en este artículo están basados en estándares para metadatos. Esta ontología especifica un formato para comprensión humana y de maquina también. En su implementación se definieron varios APIs para la gestión de la ontología que fueron introducidos en varios LMSs de código abierto. El propósito en la ontología es mapear las características de usuarios con contenidos de aprendizaje para satisfacer los requerimientos en el aprendizaje. Los objetos de aprendizaje se presentan al alumno como relaciones ontológicas. Por las anteriores circunstancias se aumenta la facilidad de uso y personalización del LMS (Heiyanthuduwage and Karunaratne, 2006).

- Hacia una Ontología sobre LMS: Este artículo hace una revisión de conceptos de LMS y propone una ontología basada en las anteriores definiciones. Este articulo es parte de una investigación que tiene como objetivo clasificar y ofrecer un sistema de calidad en el proceso de implementación de LMSs en las organizaciones (Díaz-Antón and Pérez, 2005).

- Justificación y Descripción del Dominio de Conocimiento de una Ontología para la Formalización y Automatización de Escenarios Educativos: En este artículo se justifica la necesidad de construir una ontología con el objetivo de proveer un soporte tecnológico para la especificación de escenarios de aprendizaje y herramientas de desarrollo y validación de nuevos escenarios. Especialmente se justifica la necesidad para una ontología de describir una especificación del lenguaje, para tener una primera aproximación a el dominio de conocimientos en el área (Rius et al., 2007).

- Knowledge Representation of LMS using Ontology: Aunque existen muchas formas de interpretar una implementación, todos los puntos de vista jamás son concebidos por un autor. Este artículo se enfoca en la integración de lo anterior, empleando los principios de la web semántica a los servicios educativos con estándares de e-learning. Tratando que los elementos del conocimiento (objetos de aprendizaje) estén vinculados a una ontología común (Srimathi, 2010).

- TOWARDS AN ONTOLOGY OF LMS A Conceptual Framework: Este articulo presenta una investigación en curso que tiene como objetivo final el desarrollo de un método para seleccionar implementar e integrar un LMS en una organización con un enfoque de calidad sistémica, como primer paso este articulo presenta una ontología para conceptualizar los términos asociados a un LMS unificando sus relaciones (Díaz-Antón and Pérez, 2006).

Todos los trabajos anteriores plantean la creación de una ontología en sus respectivos contextos, pero no se presenta una ontología de acceso público, sobre la cual podamos empezar a trabajar y que funcione como punto de partida, Por esta razón se decidió realizar la construcción de la ontología a partir de cero, obviamente empleado todos los elementos posibles que se han presentado, pero creándola desde un inicio, Para ello se empleara como punto de partida todo el estudio realizado en el capitulo anterior “Estudio de los módulos comunes para sistemas de gestión del aprendizaje” y que se trabajo con mapas de conocimiento.

3.2.3 Enumeración de términos importantes para la ontología

La lista de términos (posibles clases) derivados del trabajo anterior se presentan a continuación, es muy relevante nombrar que muchos de los términos fueron obtenidos del trabajo de (Díaz-Antón and Pérez, 2005), el resultado final se muestra en la siguiente tabla:

Tabla 12 Términos importante para la construcción de la ontología.

Term	Term	Term
Activity calendar	Educational Design	Portfolio
Administration	Evaluation System	Productivity
Announcement	FAQ	Registry
Authetification	File manager	Serch
Calification system	Foros	Share and reuse content
Chat	Glossary	Standars E-Learning
Communications	Groups	Student
Course	Help	Survey
Course Authorization	Hosting Service	Tools
Course template	Learning Object	Wiki
Curricula Design	Management Curricula	Work Group
Customize interface	News	

3.2.4 Definición de las clases y la jerarquía de clases

Para la definición de las clases se utilizo una combinación de procesos mixtos entre procesos de arriba hacia abajo (Top-Down) en el cual se empieza por definir los conceptos generales para después establecer la composición de estos, por ejemplo un curso dispone de una materia, y el proceso de abajo hacia arriba (Buttom-Up) que comienza con la definición de las clases más especificas y se va generalizando, por ejemplo, un foro es parte de las comunicaciones. El resultado final de este paso se puede visualizar en la siguiente figura.

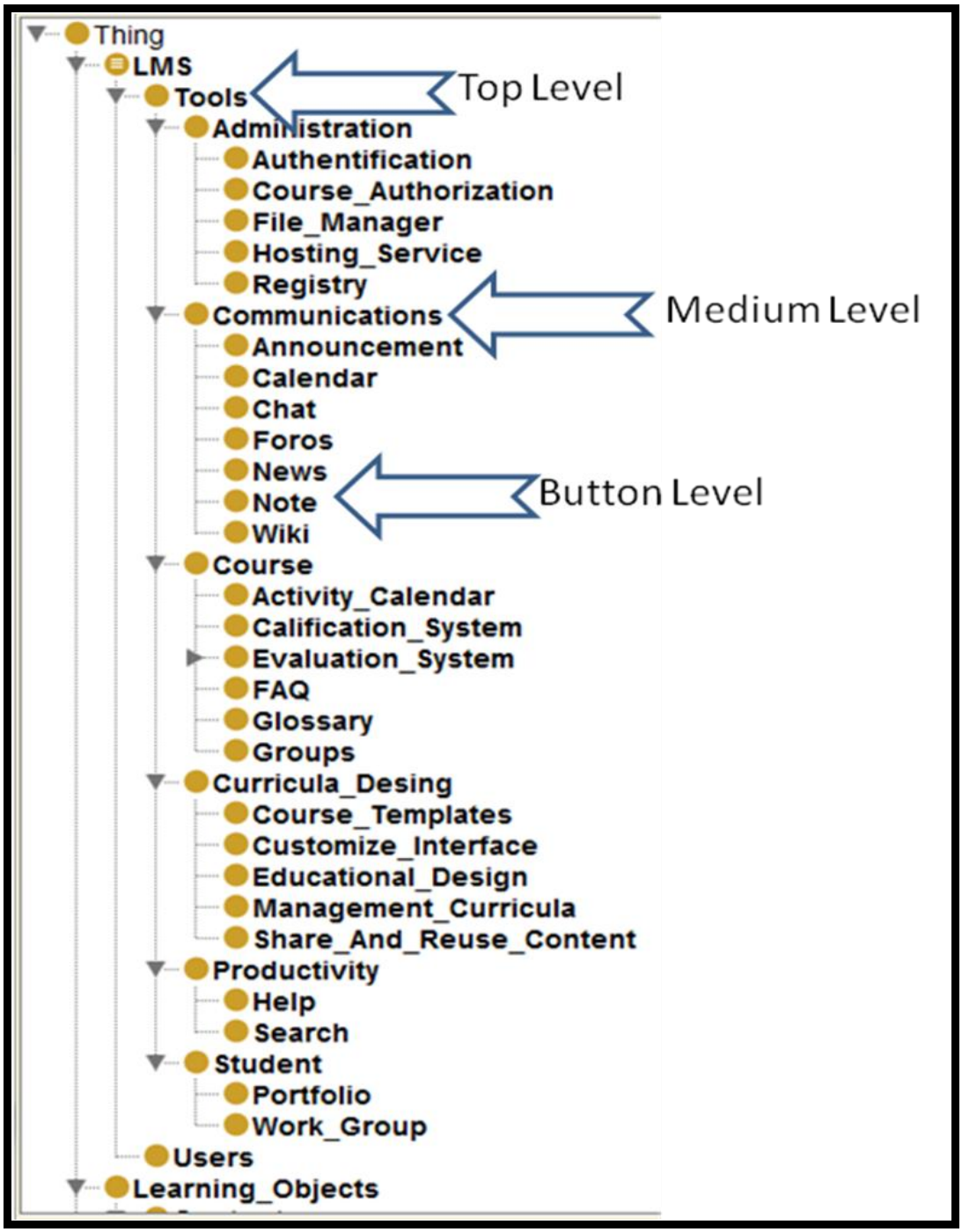


Figura 25 Vista de las clases de la Ontología en protege.

3.2.5 Definición de las propiedades de las clases: slots

Las propiedades de las clases se definen basándose en los tipos de componentes que ellas tienen, esta clasificación de los componentes fue la que se realizo en el capitulo anterior "Estudio de los módulos comunes para sistemas de gestión del aprendizaje", y son los nodos hojas de los mapas de conocimiento, por ejemplo en los mapas de

conocimiento (Vista recurso Foro con CmapTools de Moodle y Vista recurso Foros con CmapTools de Claroline), corresponden a los mapas de conocimiento para el módulo Foro de Moodle y Claroline respectivamente, como para la realización de estos mapas fue necesario navegar por los respectivos LMS, hemos tomado esa experiencia para determinar que los elementos comunes en los foros son (forumName, message, subject y optionally attachment). Estos elementos son las propiedades de las clases y son de tipo (textBox, textBox, textBox y file) respectivamente, con el anterior análisis se procede a la construcción de las clases, subclases, propiedades de las clases y slots (type) de cada propiedad. Que para el caso ejemplo de módulo Foro son las imágenes que aparecen a continuación:

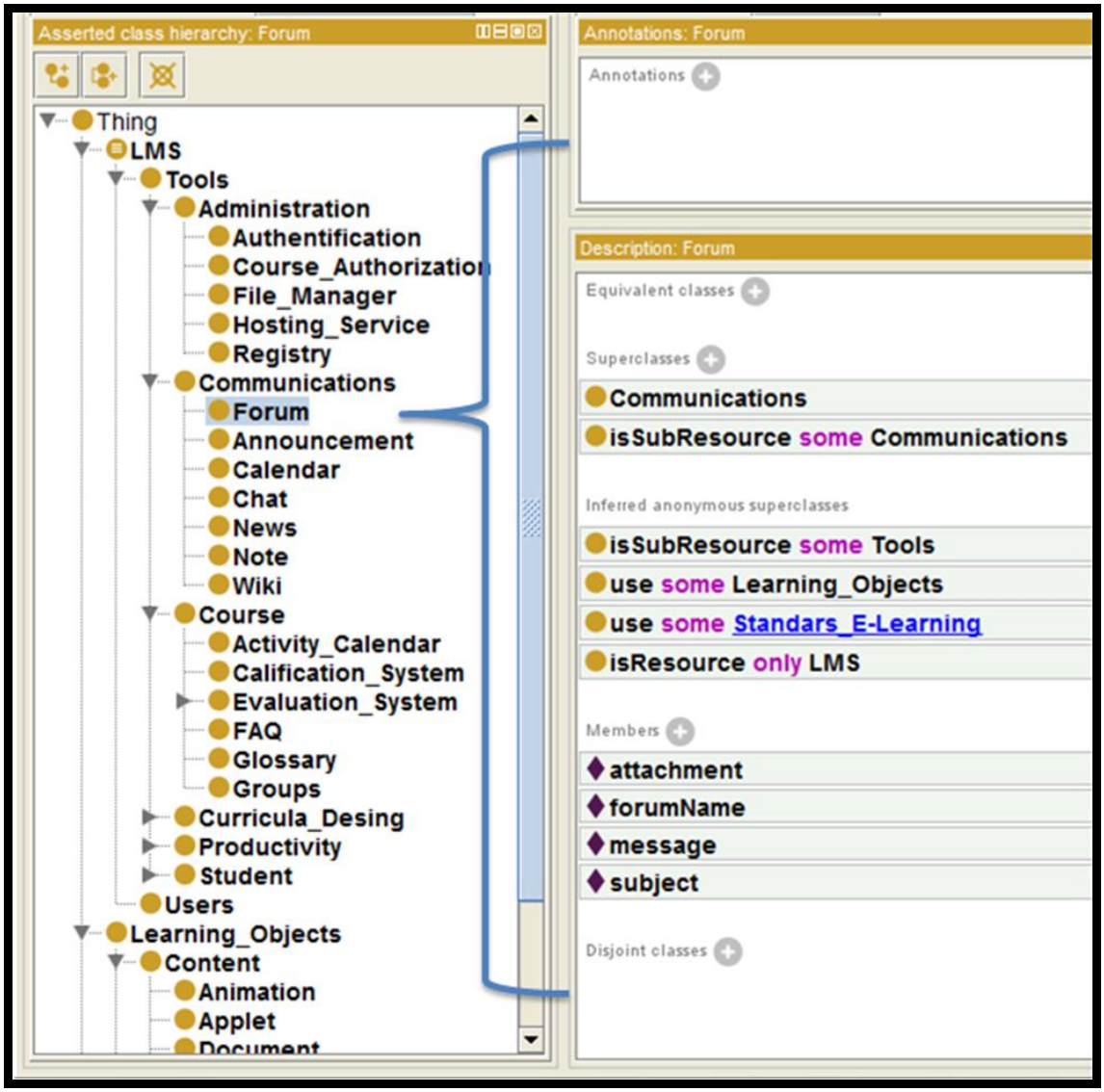


Figura 26 Miembros de la clase Foro.

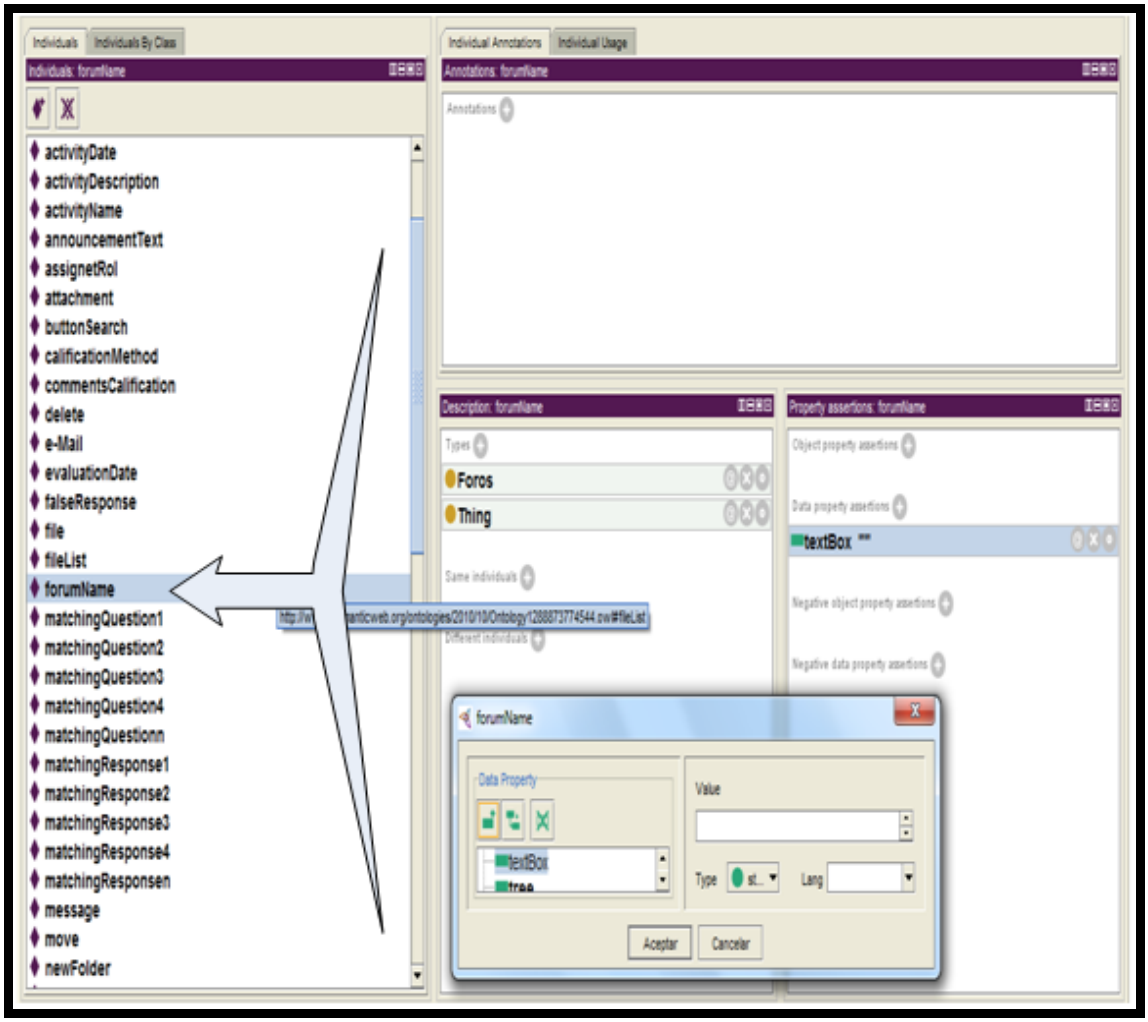


Figura 27 Slot (type) de el elemento forumName.

Más adelante se encuentra la descripción de toda la ontología, allí se puede visualizar como es la estructura de la misma en clases, subclases, propiedades de las clases y slots (type) de cada propiedad.

3.2.6 Definición de las facetas de los slots

Aquí se definen las facetas de los Slots, es decir el tipo de valor, los valores validos, el numero de valores (cardinalidad) y otras características que los slots puedan tomar, por ejemplo, la propiedad evaluationDate que pertenece a la clase Evaluation_System es de tipo calendarDate y sus valores únicamente pueden ser Activity_Calendar, y este tipo de valores pueden ser cadenas de caracteres (Strings) o números, esto se muestra en la siguiente figura. Más adelante se encuentra la descripción de toda la ontología.

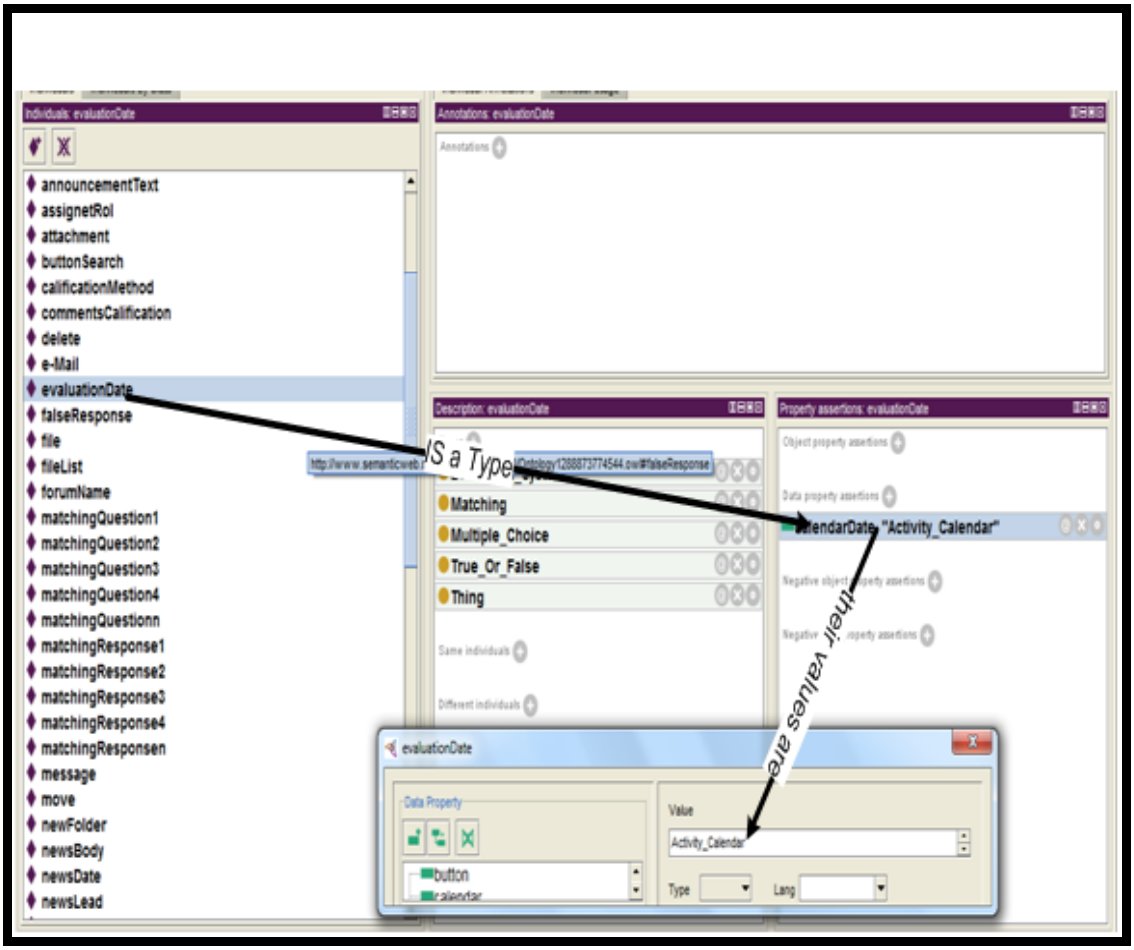


Figura 28 Propiedad evaluationDate que es de tipo calendarDate y sus valores son Activity_Calendar.

3.2.7 Creación de las instancias

Este último paso no se probara en este momento ya que corresponde al DSL. Aquí se crearan las instancias de las clases, como por ejemplo la creación de un Foro en Moodle, como se ha dicho la responsabilidad de esta parte recaerá en el DSL, allí se realizara un modelo del curso y luego se desplegara sobre un LMS, por esta razón este capítulo se tratara más adelante.

3.2.8 Generación de la ontología

La ontología generada se compone de 50 clases incluyendo la clase (Thing), que es la raíz en cualquier ontología que se construya con protege, 4 propiedades de los objetos, 16 propiedades de los datos, 69 individuos, cada uno de estos componentes con sus respectivas descripciones, las clases más generales son LMS, Learning_Objects y Standars_E-Learning, estas clases se muestra en la siguiente figura, la visualización de estas clases se hace con la vista de jerarquía (class

hierarchy) y con la vista del plugin OWLViz, ambos tipos de visualización los ofrece protege.

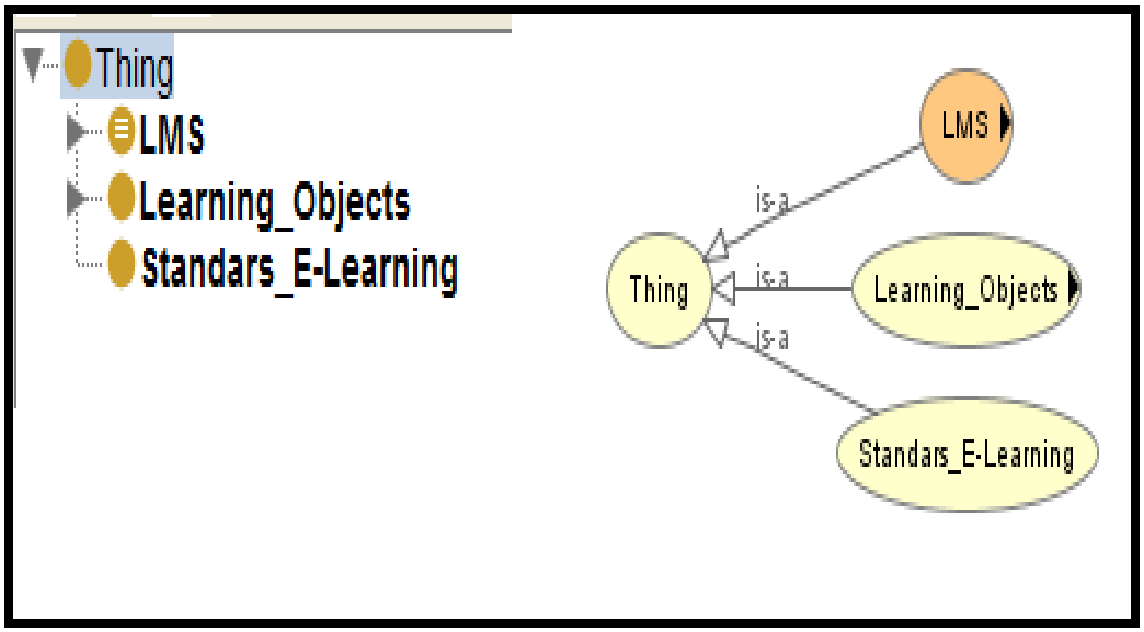


Figura 29 Vista general en protegé con class hierarchy y OWL Viz Plugin.

La clase LMS tiene Usuarios (Users) y Herramientas (Tools), los usuarios tienen assignetRol, e-Mail, password and UserName como miembros de él, y la clase herramientas tiene las subclases: Tools Administrations, Communications, Course, Curricula_Desing, Productivity, and Student, esto se muestra en la siguiente figura.

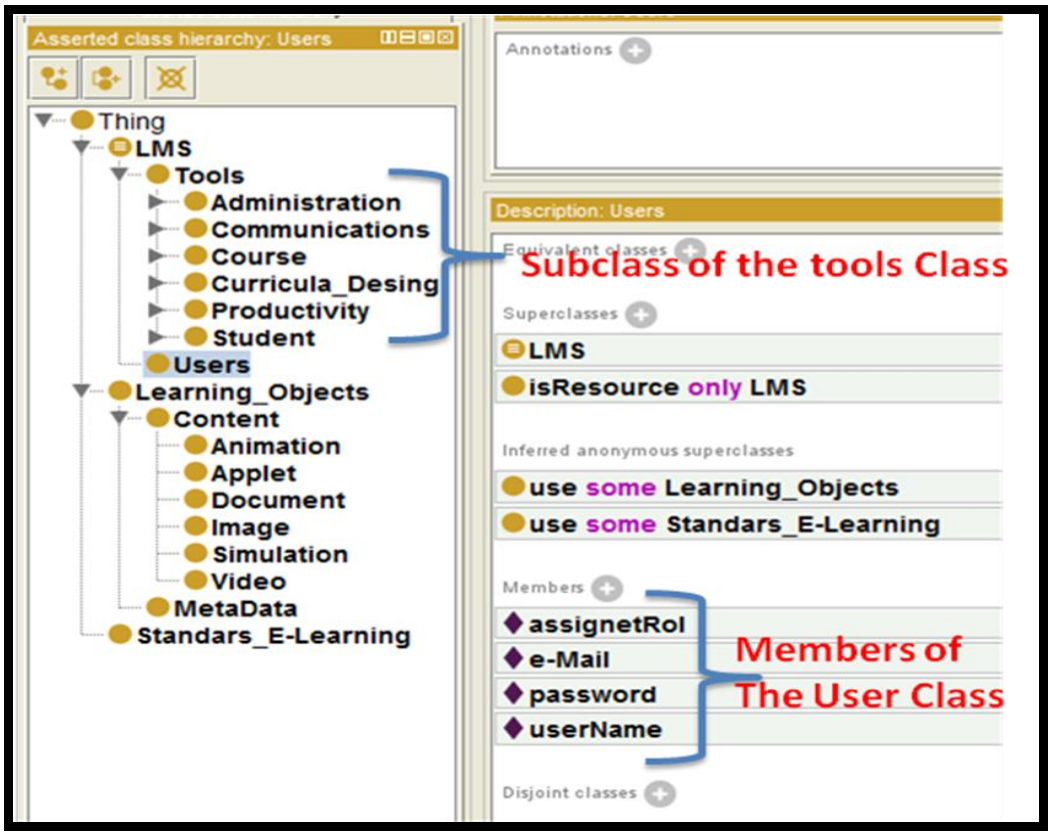


Figura 30 Subclases de la clase Tools y miembros de la clase Users.

La clase Administration es hija de la clase Tools y tiene cinco subclases; Authentication, Course_Authorization, File_Manager, Hosting_Service y Registry. La clase Authentication tiene un password que es de tipo textBox y userName que también es de tipo textBox. La clase Course_Authorization tiene un assigneRol de tipo comboBox y userList de tipo list. La clase File_Manager tiene un delete de tipo event, fileList de tipo list, move de tipo event, newFolder de tipo button, rename de tipo event y uploadfile de tipo explorator. La clase Registry tiene un registerOfUsers de tipo link. Y la clase Hosting_Service. Todo esto se muestra en la siguiente figura.

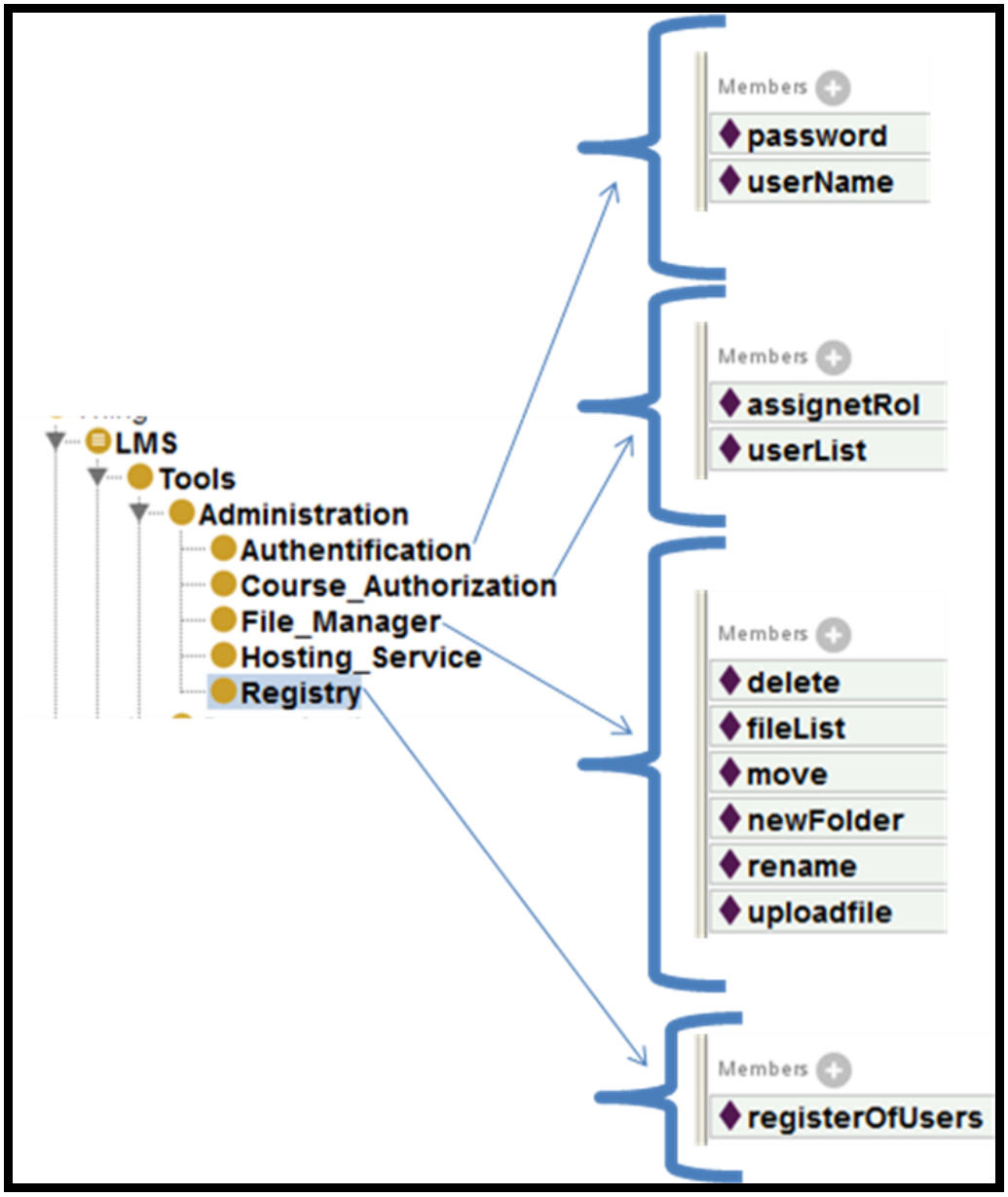


Figura 31 Subclases de la clase Administration y sus miembros.

La clase Communications es una subclase de la clase Tools y tiene siete subclases; Announcements, Calendar, Chat, Foros, News, Note y Wiki. La clase Announcements class, tiene un announcementText de tipo textArea. La clase Calendar, tiene un showCalendar de tipo calendar. La clase Chat tiene un roomName de tipo textBox, send de tipo button, sendText de tipo textBox, textWrittenByUsers de tipo nonEditableTextArea y userList de tipo list type. La clase Foro, tiene un attachment de tipo file, forumName de tipo textBox, message de tipo textBox y subject de tipo textBox. La clase News tiene un newsBody de tipo textArea, newsDate de tipo textBox, newsLead de tipo textBox y newsTitle de tipo textBox. La clase Note, tiene un message de tipo textBox, send de tipo button y userName de tipo textBox. Y la clase

Wiki, tiene un wikiEdit de tipo textArea y wikiView de tipo view. Todo esto se muestra en la siguiente figura.

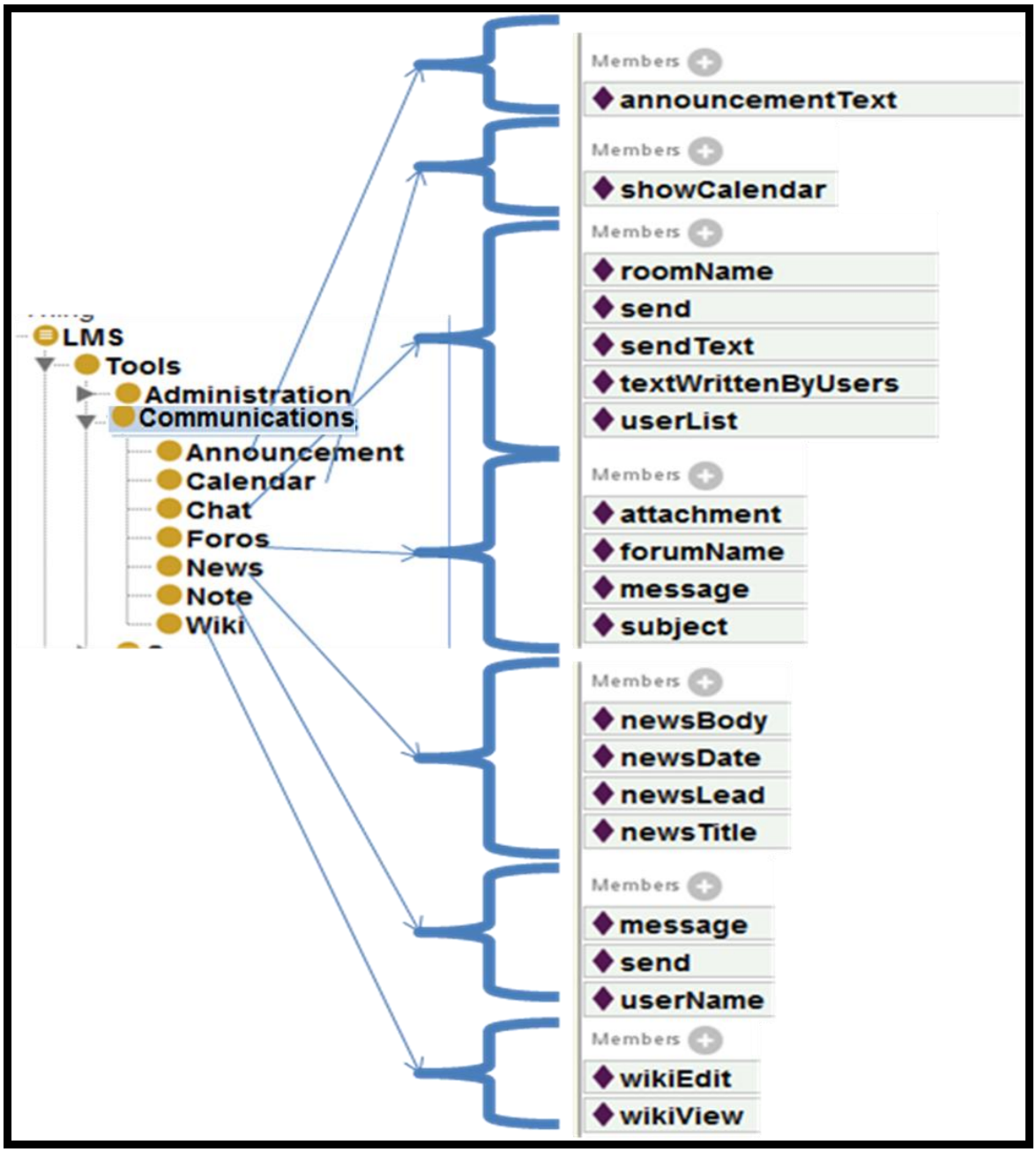


Figura 32 Subclases de la clase Communications y sus miembros.

La clase Course es una subclase de Tools y posee seis subclases; Activity_Calendar, Calification_System, Evaluation_System, FAQ, Glossary y Groups. La clase Activity_Calendar tiene un activityDay que es de tipo calendarDate, activityDescription de tipo textArea, activityName de tipo textBox and attachment de tipo file. La clase calification_System, tiene un activityName de tipo textBox, commentsCalification de

tipo textArea y valueCalification de tipo textBox. La clase Evaluations_System, tiene un calificationMethod de tipo textBox, evaluationDate de tipo calendarDate, statementQuestion de tipo textArea y tiene a su vez tres subclases, la clase Matching que posee, matchingQuestion1, 2, 3, 4 hasta n de tipo textBox y matchingResponse1, 2, 3, 4 hasta n de tipo textBox. La clase Multiple_Choise que posee, MultipleResponse1, 2, 3, 4 hasta n de tipo textBox y percentageHitResponse1, 2, 3 ,4 hasta n de tipo textbox y finalmente la clase True_Or_False que posee, falseResponse de tipo textArea y trueResponse de tipo textArea. Otras subclases de Course, son la clase FAQ, que tiene un viewFAQ de tipo textArea, la clase Glossary que tiene un viewGlossary de tipo textArea y la clase Groups que tiene un userList de tipo list. Todo esto se muestra en la siguiente figura.

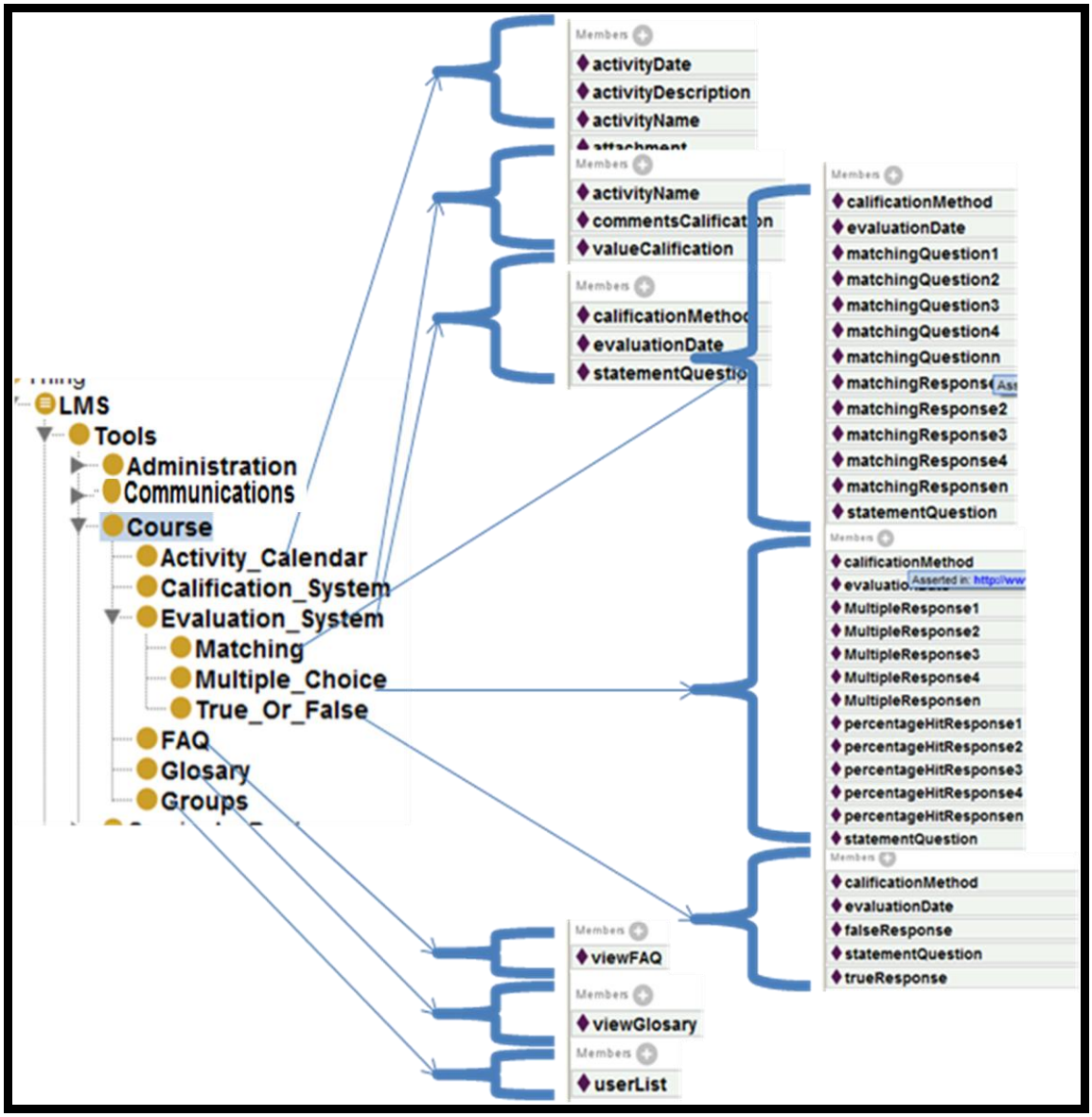


Figura 33 Subclases de la clase Course con más subclases y sus respectivos miembros.

La clase Curricula_Design es hija de la clase Tools y tiene cinco subclases; Course_Templates, Customize_Interface, Educational_Design, Management_Curricula, y Share_and_Reuse_Content. La clase Course_Templates tiene un viewCourseTemplate de tipo view. La clase Customize_Interface tiene un viewcustomizeInterface de tipo view. La clase Educational_Design tiene un viewEducationalDesign de tipo view. La clase Management_Curricula tiene un viewmanagementCurricula de tipo view y la clase Share_And_Reuse_Content tiene un fileList de tipo checkButton. Todo lo anterior se muestra en la siguiente figura:

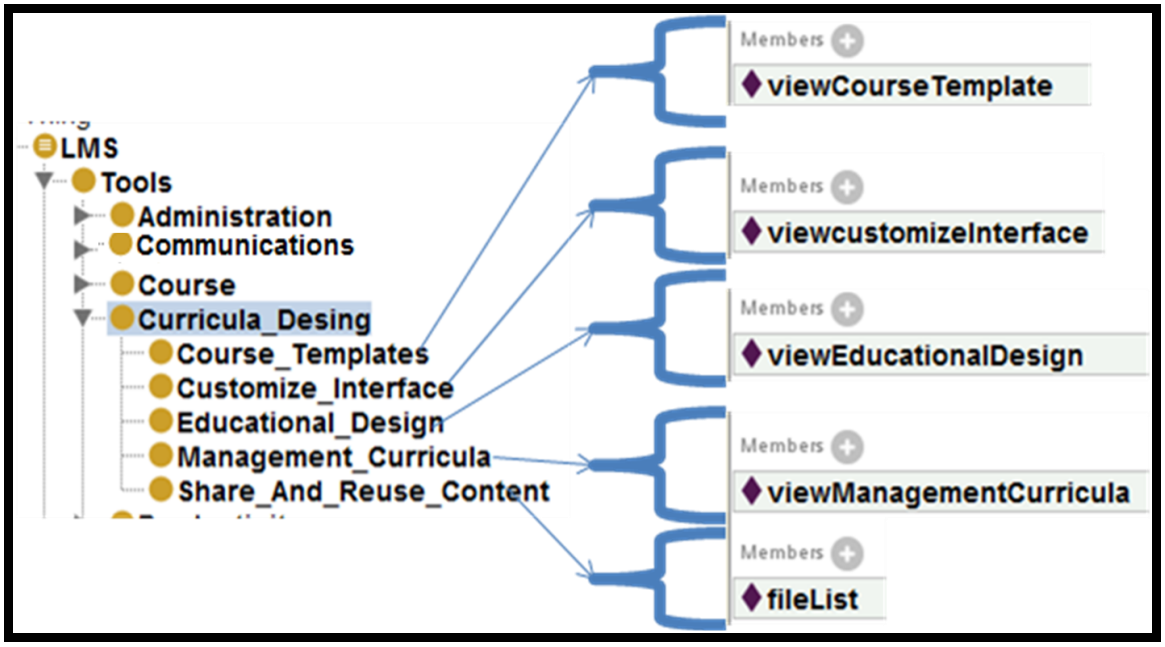


Figura 34 Subclases de la clase Curricula_Design y sus miembros.

La Clase Productivity es una subclase de Tools y tiene dos subclases; Help y Search. La clase Help tiene un viewHelp de tipo view. La clase Search tiene un buttonSearch de tipo button y tiene un textSearch de tipo textBox. Esto se muestra en la siguiente figura.

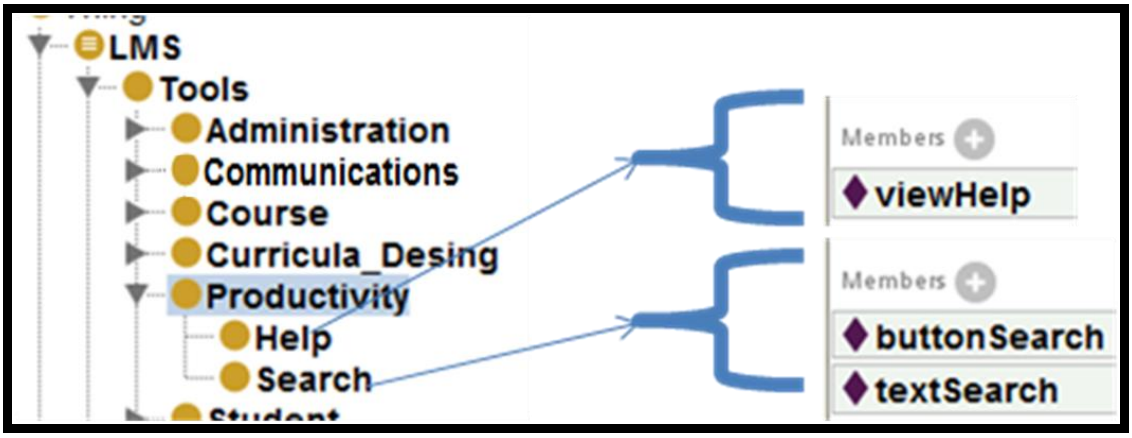


Figura 35 Subclases de la clase Productivity y sus miembros.

La clase Student es hija de Tools y tiene dos subclases; Portfolio y Work_Group. La clase Portfolio tiene un viewPorfolio de tipo view y la clase Work_Group tiene un viewworkGroup de tipo view. Esto se muestra en la siguiente figura.

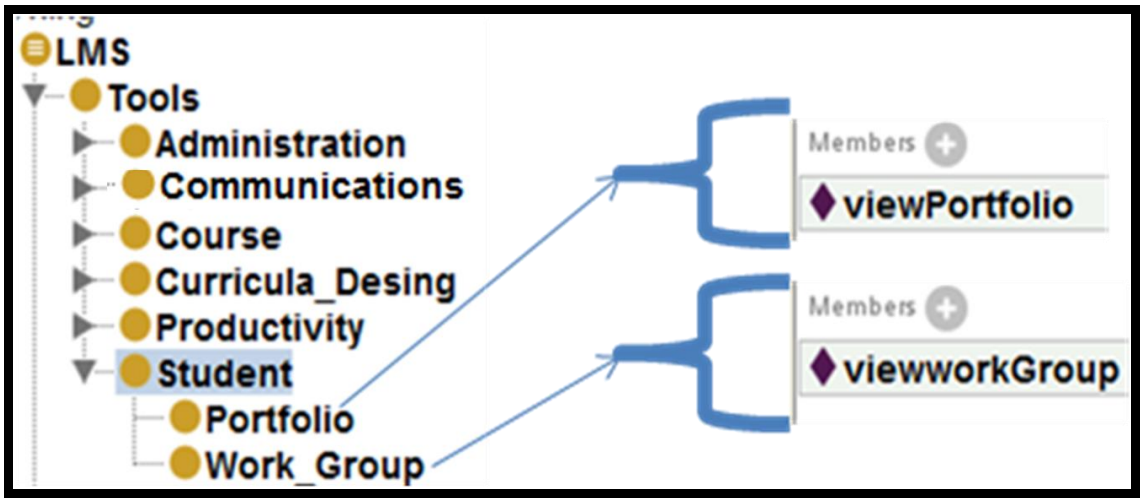


Figura 36 Subclases de la clase Studenty sus miembros.

Por último están las clases Standars_E-Learning y Learning_Objects, de esta última las subclases que derivan son Metadata y Content. La clase Content tiene las subclases Animation, Applet, Document, Image, Simulation y Video. Como se muestra en la siguiente figura.

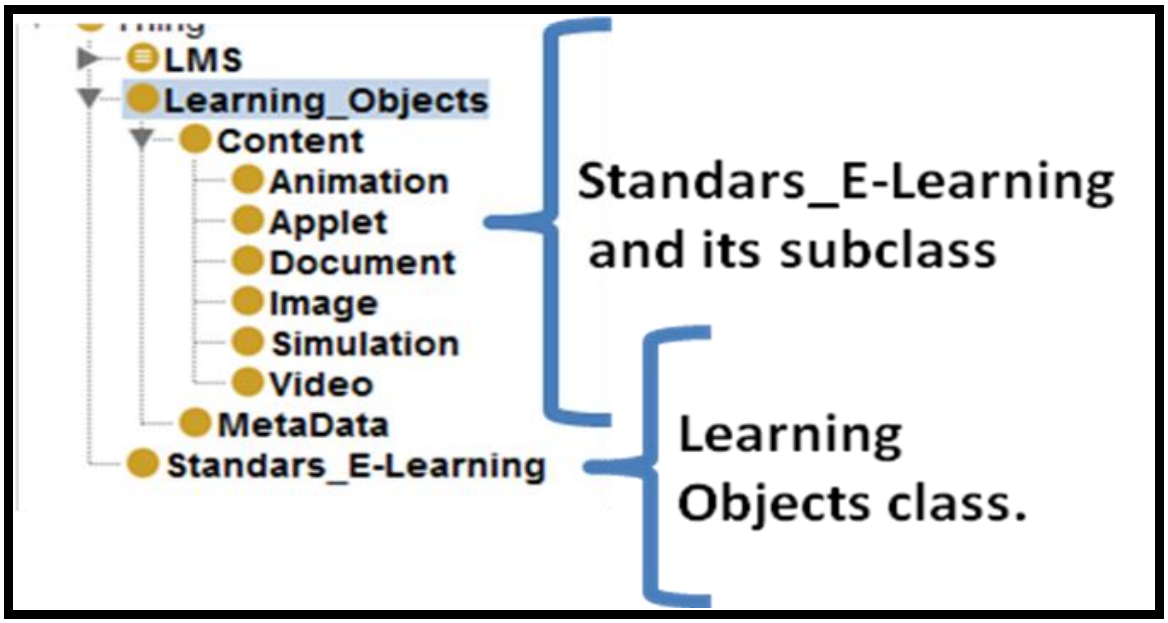


Figura 37 Clases Standars_E-Learning y Learning_Objects y sus subclases.

La figura que se muestra a continuación representa el modelo completo de la ontología creada, y muestra únicamente elementos similares entre los LMS analizados con anterioridad, los módulos comunes entre las plataformas analizadas son: File_Manager, Registry, Announcements, Calendar, Chat, Foros, News, Note, Wiki, Activity_Calendar, Calification_System, Evaluation_System, FAQ, Glossary, Groups, Help, Search, portfolio, Work_Group, Users, y Learning_Objects.

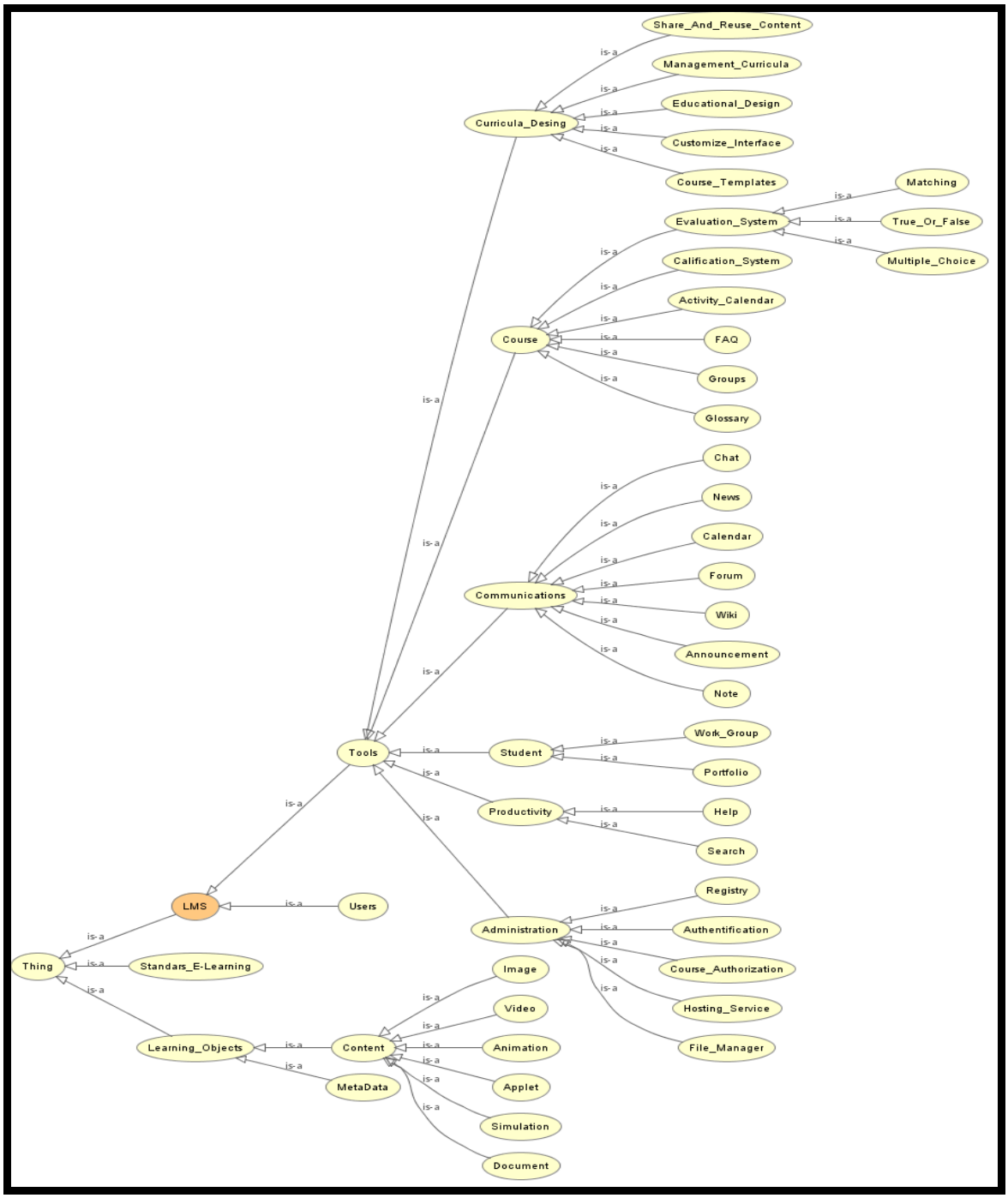


Figura 38 Ontología LMS.

Podemos concluir que las plataformas LMS usan diferentes estándares, pero los módulos principales son similares en todas las plataformas, el gran problema de compatibilidad se presenta en la implementación tecnológica que se hace en cada una de las plataformas especificas, por este motivo la idea de establecer una manera para

poder crear modules de un curso independiente de la plataforma toma más fuerza y una solución es aplicando Model Driver Engineering (MDE) para ello.

Una buena recomendación a la hora de construir una ontología, es conocer el dominio del contexto y una buena manera de hacerlo, es representar este conocimiento con mapas mentales, obviamente apoyándose para la diagramación en herramientas de software, La ontología completa está disponible en los anexos digitales a esta memoria (+LMS Ontology).

3.3 Metamodelo para sistemas de gestión del aprendizaje

Antes de explicar el metamodelo es necesario mencionar los trabajos que contribuyeron a esta propuesta, como el de (Bizoňová et al., 2007) en donde el objetivo es *"provide a generalized architectural framework enabling an integrated specification of platform architectures"*, este trabajo presenta un planteamiento original para el problema de la integración de las plataformas LMSs de diferentes arquitecturas. En particular se propone una estrategia de integración de LMSs dentro de un modelo general de LMSs, como ejemplo de esta estrategia de integración se trataran dos funcionalidades: derechos de acceso (access righs) y gestión de objetos de aprendizaje (learning object management). Aquí se realizo un Modelo Independiente de la Plataforma (PIM), de dos sistemas LMS (Moodle y OLAT) y se muestra como mapear esto a un PIM general que sea genérico. El framework debe ser enriquecido examinando más sistemas LMSs. (MORENO and ROMERO, 2005a) Proponen un framework para plataformas LMS que permite separar la especificación del sistema en diferentes perspectivas complementarias. Por otra parte, este framework aísla el proceso final de desarrollo (tecnología) del modelado del sistema, esto lo hace usando los conceptos de MDA que proporcionan la captura de la funcionalidad del sistema y organiza la especificación de los complejos sistemas e-learning. En (Henderson-Sellers, 2011) se dice que una ontología fundamental puede utilizar el mismo nivel de abstracción de un metamodelo y una ontología de dominio proporciona el mismo nivel de abstracción que un modelo (en el diseño) , esto se puede hacer con la debida asignación en la semántica entre ontologías y modelos. Por otro lado (García-Díaz et al., 2009) presentan un framework para el desarrollo de todo tipo de aplicaciones basado en la opiniones de reconocidos expertos de MDE y lecciones aprendidas de otros autores.

El metamodelo creado está basado en la ontología propuesta en la sección anterior. La ontología propuesta se puede observar en la figura anterior. Esta ontología muestra principalmente los nodos: LMS, Standars_E-Learning y Learning_Objects. El nodo LMS tiene Users y Tools, y el nodo Tools tiene Curricula_Desing, Course, Communications, Student, Productivity, y Administration. En el proceso de generación del metamodelo, el conocimiento pude transformarse para responder a las necesidades del sistema, con lo que el metamodelo no es una representación exacta de la ontología para el mapping entre la ontología y el metamodelo, se ha utilizado la

propuesta de (Henderson-Sellers, 2011), donde se postula una relación entre los elementos de la Ontología y los correspondientes al metamodelo. Este proceso de mapping se puede ver en la siguiente figura.

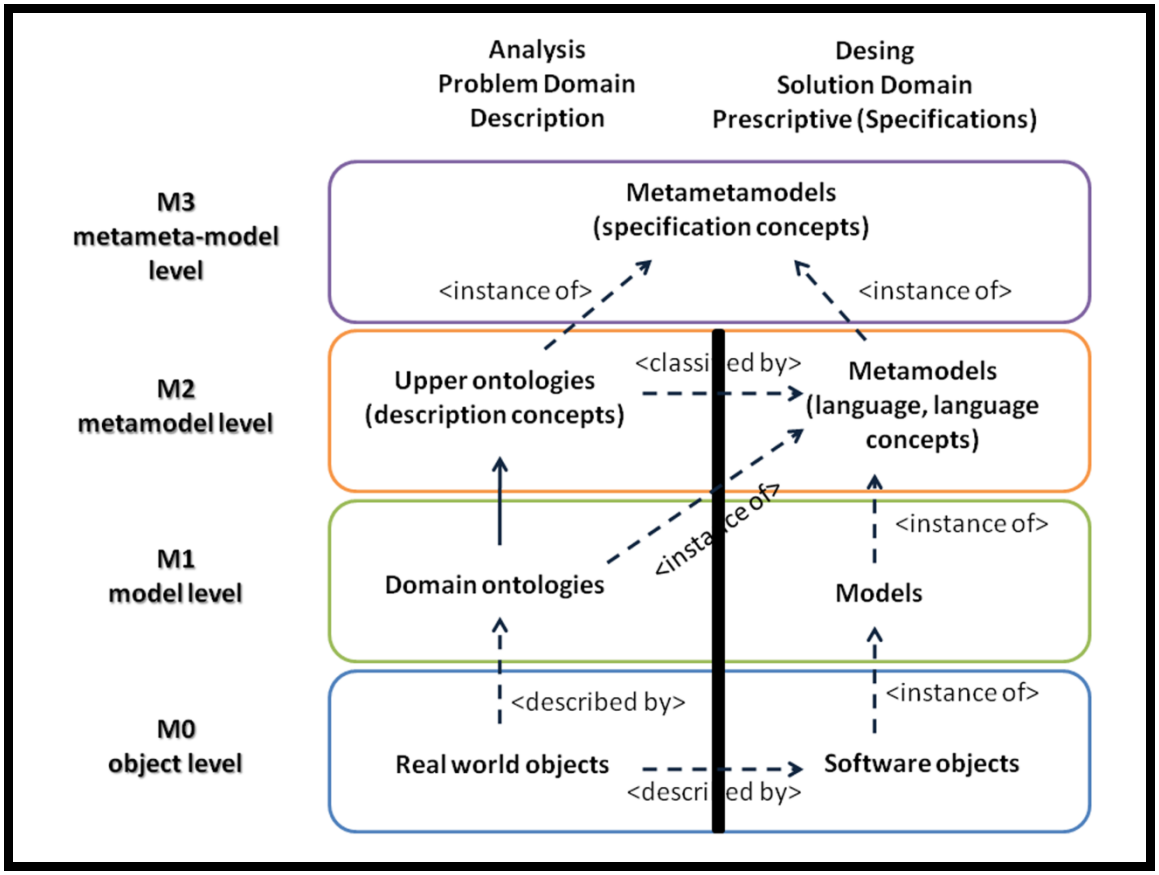


Figura 39 Propuesta de integración de una ontología a un metamodelo, en la arquitectura de cuatro niveles propuesta por la OMG para MDA (Henderson-Sellers, 2011).

El metamodelo LMS que plantearemos corresponderá únicamente al nodo LMS de la Ontología tratada, los demás nodos no son requeridos para nuestro caso. El metamodelo está construido bajo ecore y se ha empleado el editor que eclipse modeling ofrece para ello (un manual sobre la instalación y uso de los plugins para eclipse modeling esta en anexo II). El LMS puede tener cero o muchos Users, mientras que los Users pueden tender un rol (assignetRol), correo electrónico (eMail), contraseña (password) y nombre de usuario (userName). Las siguientes figuras muestran la correspondencia de la EClass User entre el modelo grafico que ofrece el plugin de eclipse para el diseño de metamodelos bajo Ecore y el explorar de Ecore. La otra figura muestra la misma correspondencia anterior pero entre el modelo grafico y el XMI que genera el plugin. El plugin de Eclipse ofrece todas estas visualizaciones.

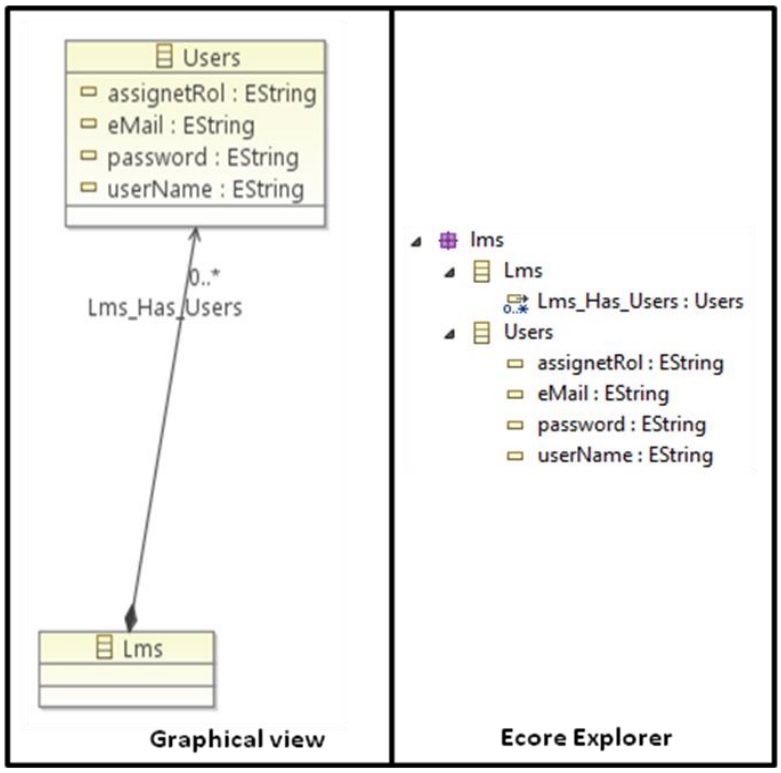


Figura 40 Representación y correspondencia de la EClass User entre la vista grafica y el explorar de ecore, que ofrece el plugin de modelado en eclipse.

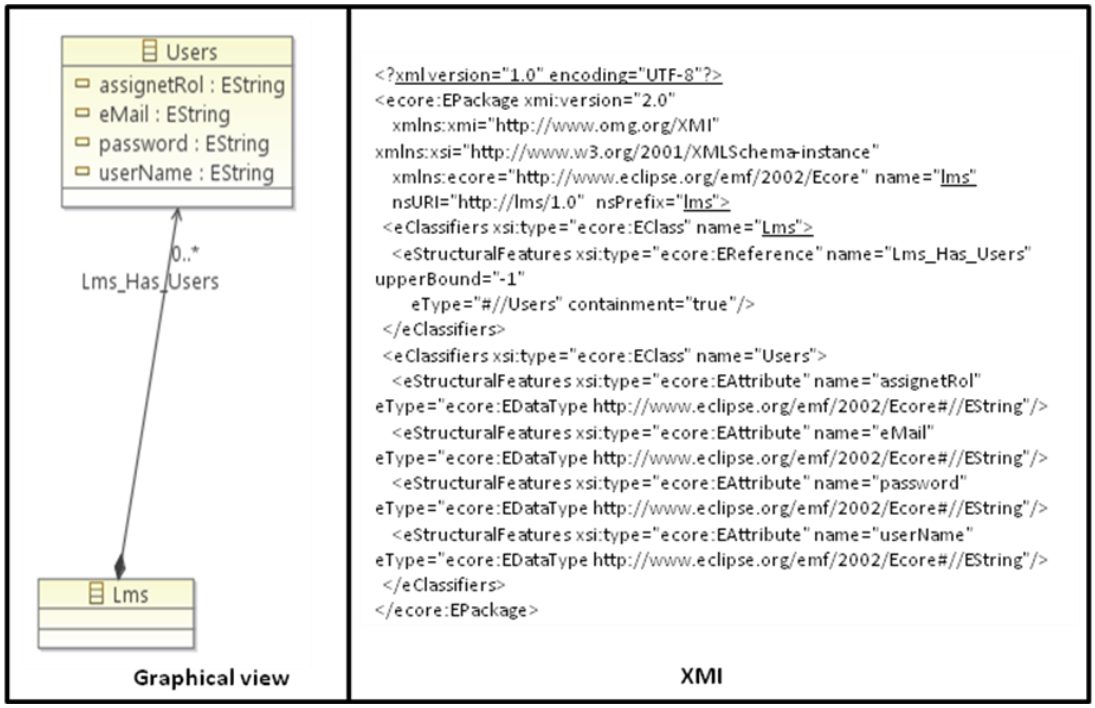


Figura 41 Correspondencia de la EClass User entre la vista grafica y el código XMI, que ofrece el plugin de modelado en eclipse.

El LMS tiene tools para: Course, Communications, and Administration, Los tres elementos tiene en común el atributo nombre (nameCourse). Por esta razón existe una EClass Tools, y de esta heredan las EClass Course, Communications, y Administration. Esto se muestra en la siguiente figura.

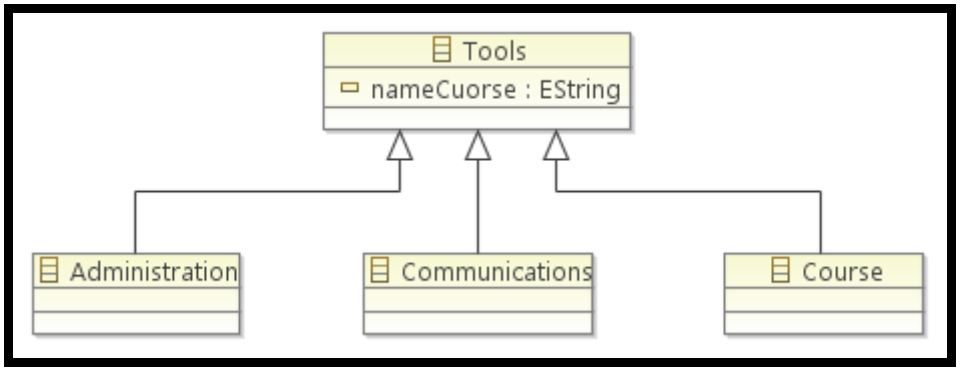


Figura 42 Composición de las EClass Tools, Course, Communications y Administration.

La EClass Course tiene has Activity_Calendar, Faq, Glossary, Calification_System, Groups y Questions. La EClass Questions tiene Answers. Cada EClass tiene sus respectivos atributos, estos se muestran en la siguiente figura.

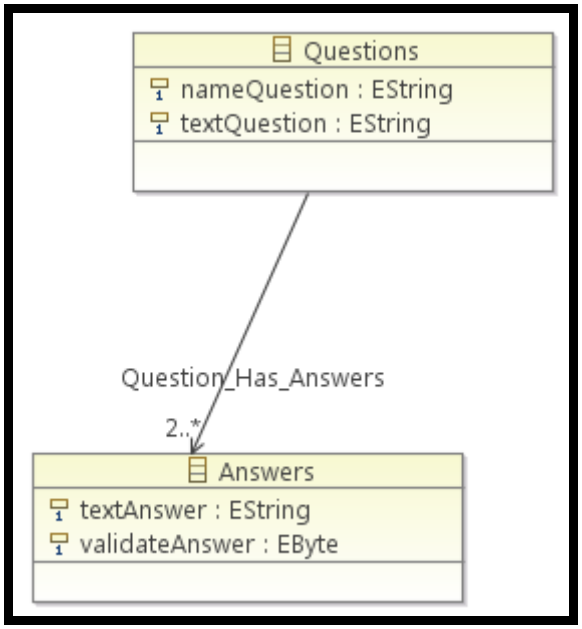


Figura 43 Composición de las EClass Questions y Answers.

La siguiente figura muestra las relaciones y atributos de las otras EClass, los atributos de la EClass Groups son nameGroup y Description. Los atributos de la EClass Calification_System son activity_name, commentsCalification y valueCalification. Los atributos de la EClass Glosary son nameGlossary y BodyGlossary. Los atributos de la EClass Faq son titleFAQ y bodyFAQ. Los atributos de la EClass Activity_Calendar son

activityDate, activityDescription, activityName y attachment. Todas las EClass están contenidas en la EClass Course. La EClass Group tiene conexión con la EClass Users, por que un grupo tiene usuario.

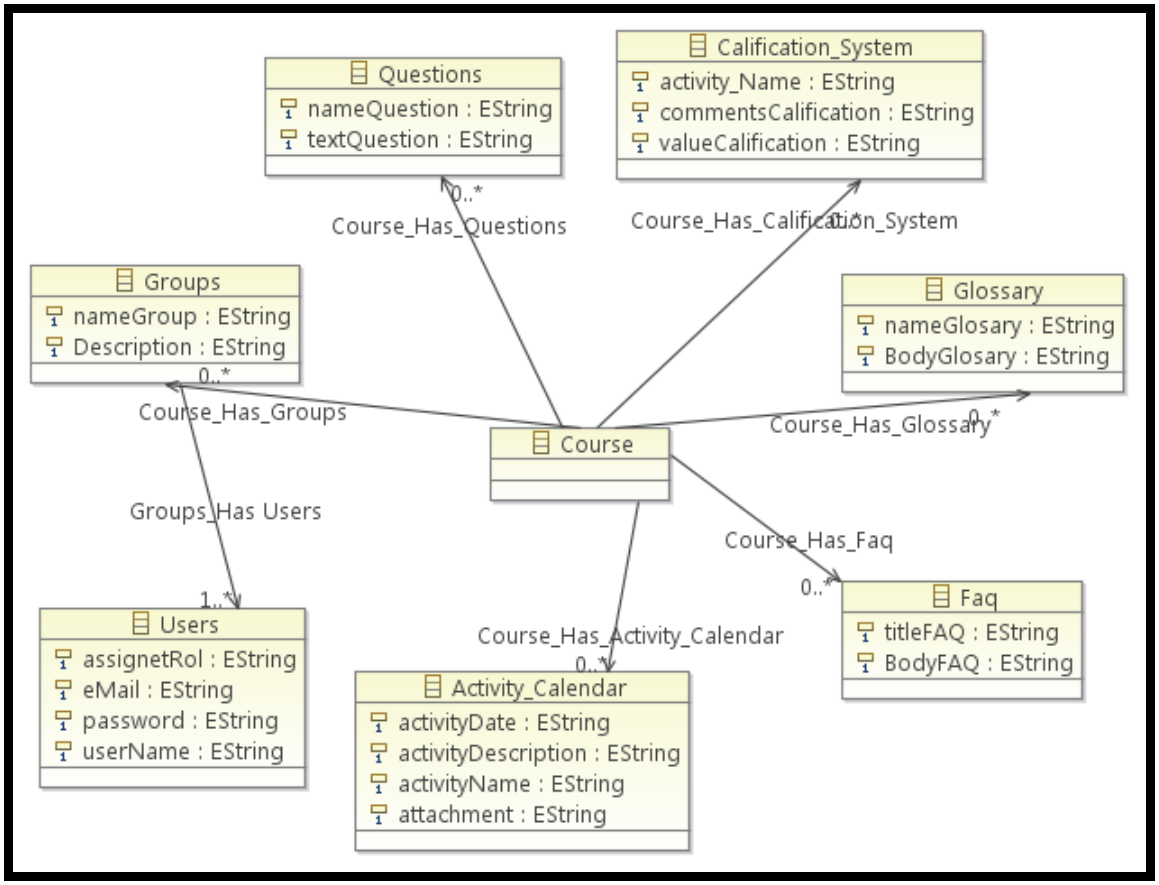


Figura 44 Composición de la EClass Course.

La EClass Communications tiene Note, News, Wiki, Chat, Announcement y Forum. La EClass Note tiene message y sendTo. La EClass News tiene newsBody y NewsTitle. La EClass Wiki tiene WikiName y WikiBody. La EClass Chat tiene roomName y sendText. La EClass Announcements tiene AnnouncementsText y la EClass Forum tiene forumName y message. Todas las Eclass están contenidas en la EClass Communications. Esto se muestra en la siguiente figura.

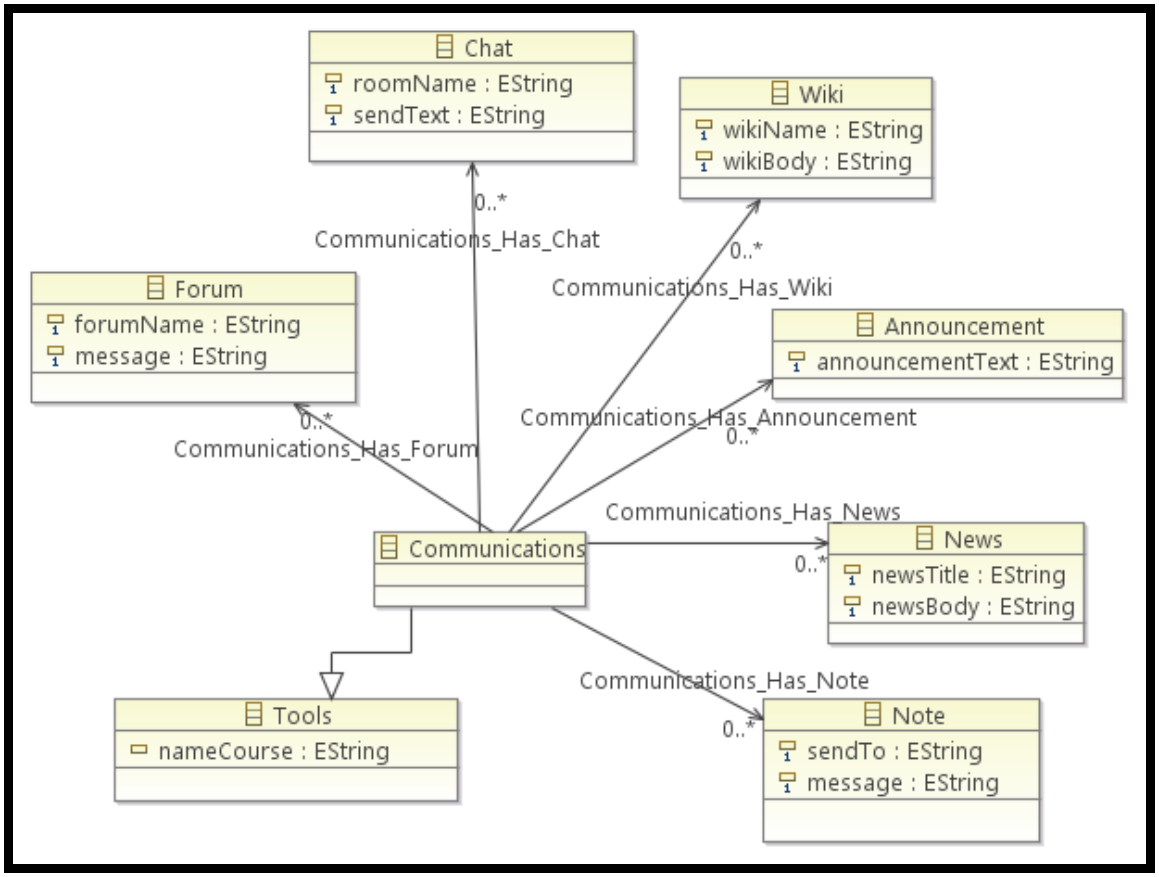


Figura 45 Composición de la EClass Communications.

La EClass Administration tiene File_Manager, Course_Authorization, y Authentication; la EClass File_Manager tiene newFolder y upLoadFile. La EClass Course_Authorization tiene assignedRol y la EClass Authentication tiene username y password. Todas estas EClass están contenidas en la EClass Administration. La EClass Course_Authorization tiene una conexión con la EClass Users, porque un rol es asignado a un usuario. Esto se muestra en la siguiente figura.

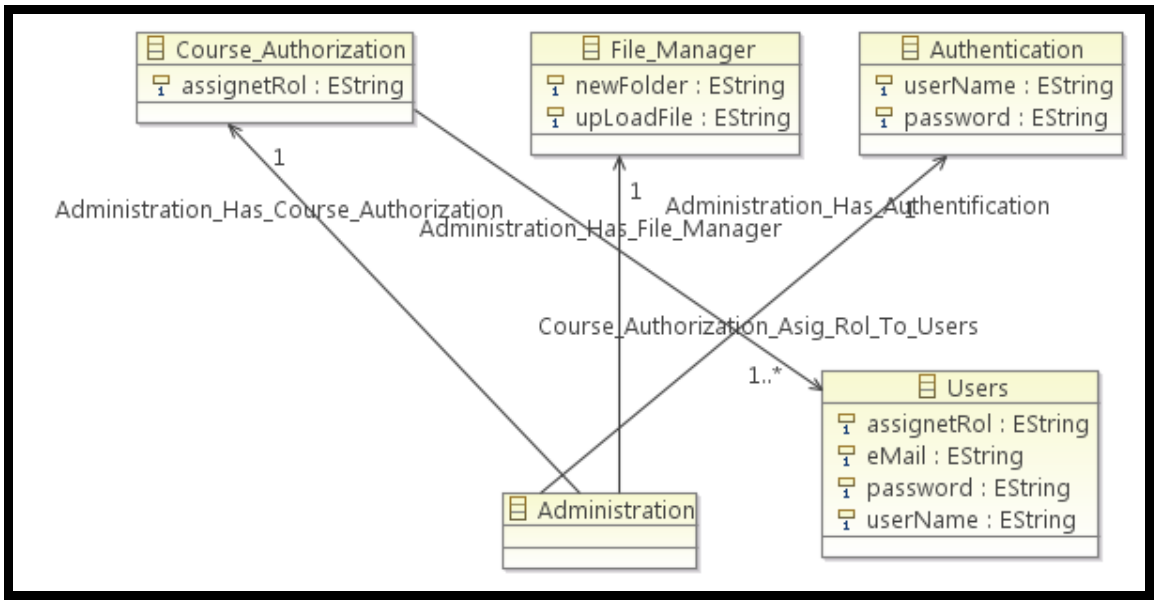


Figura 46 Composición de la EClass Administration.

La EClass LMSModel es un contenedor, que almacena todos los elementos de la sintaxis abstracta. Una EClass que almacena todos los elementos del modelo es necesaria en la definición de todo metamodelo. Esta es similar a un Canvas en programación. Esto se muestra en la siguiente figura. Allí la EClass LMSModel contiene: Users, Authentication, Course_Authorization, File_Manager, Announcement, Chat, Forum, News, Note, Wiki, Activity_Calendar, Calification_System, Faq, Glossary, Groups, Questions, Answers, Course, Communications y Administration.

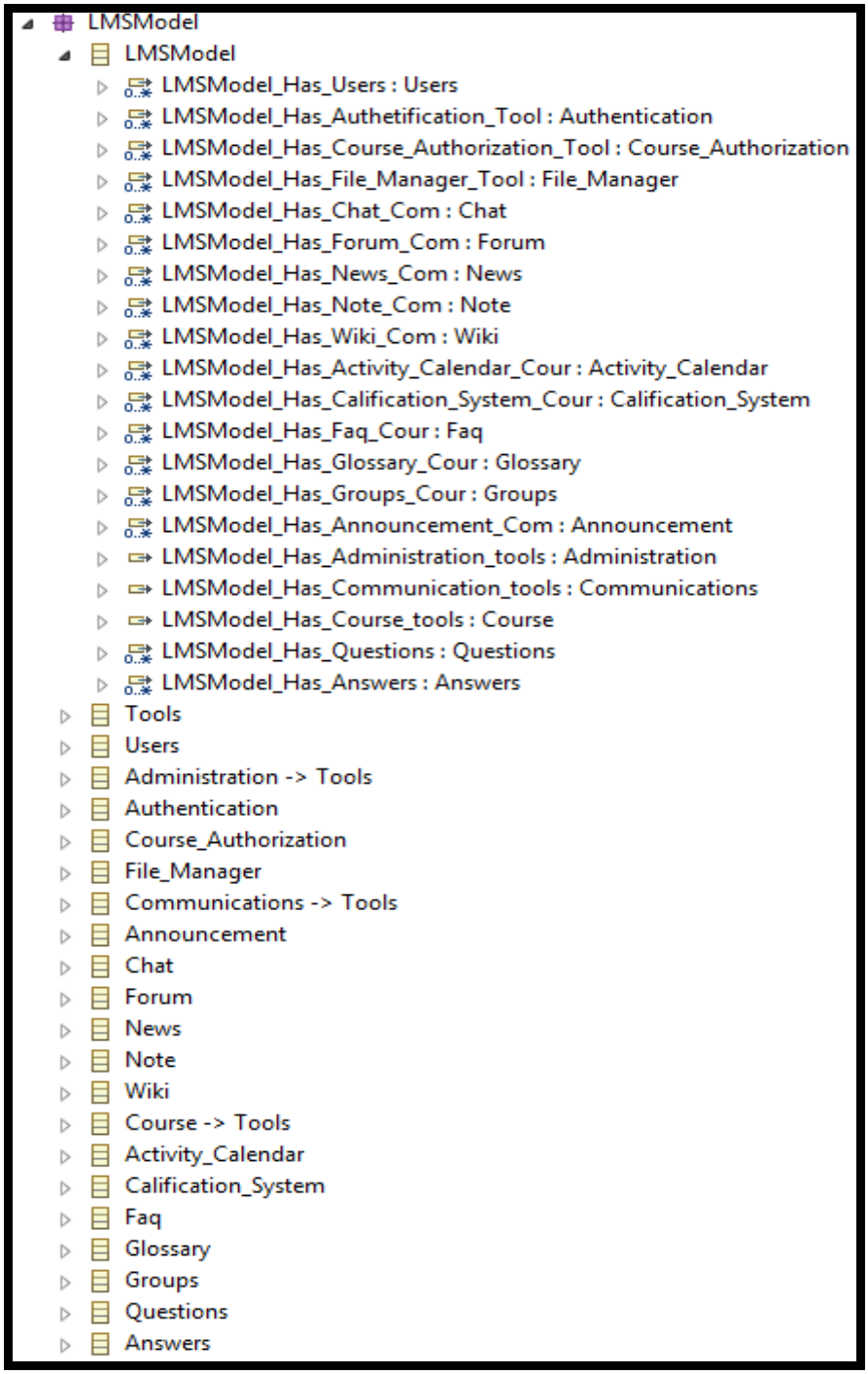


Figura 47 Metamodelo LMS.

Esta tesis tiene como objetivo modelar módulos de cursos para LMSs, y se ha definido que los módulos básicos para un LMS son los que están contenidos en la EClass Communications, por esta razón a continuación describiremos como se hizo la construcción de la herramienta DSL para los módulos de Communications del metamodelo anterior.

3.4 KiwiDSM v1.0: Herramienta DSL grafica para la construcción de módulos de un LMS.

Para la construcción del DSL se han empleado básicamente dos tecnologías Eclipse Modeling Framework (EMF) y Graphical Modeling Framework (GMF) (Foundation, 2010b), EMF se empleo apara la definición del metamodelo y de él ya se ha hablado en la sección anterior. GMF es una solución que mejora la relación EMF/GEF (Eclipse Graphical Editing Framework) con la utilización de metamodelos que permiten la definición de la sintaxis abstracta y concreta, también define facilidades para el mapping entre las dos y una forma sencilla de realizar una interfaz personalizada para él, en el anexo II se encuentra un manual para la instalación del ambiente de desarrollo.

La construcción del DSL se realizó bajo la plataforma eclipse y este debe tener instalados los plugins de EMF, GEF y GMF. Una vez abierto eclipse lo primero que se debe hacer es crear un nuevo proyecto GMF (New -> Project -> Folder Graphical Modeling Framework -> New GMF Project), al crear el proyecto se visualiza el dashboard, este muestra el flujo para la creación del DSL, la siguiente figura muestra el dashboard que se genera.

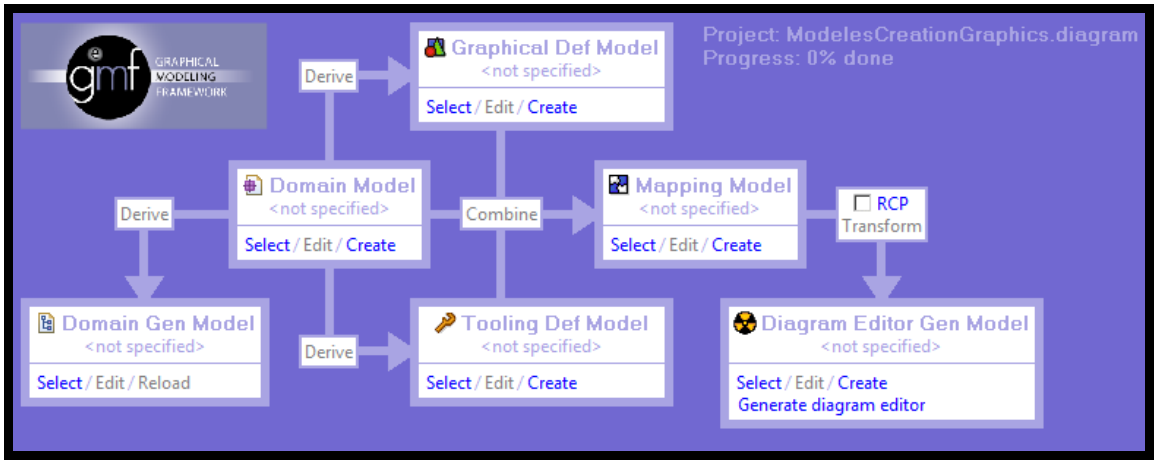


Figura 48 GMF DashBoard.

Según el dashboard lo primero que se debe crear es el Domain Model, este corresponde al LMS metamodel descrito en la sección anterior. El siguiente paso es crear el Domain Gen Model, que es un modelo que permite transformar automáticamente el modelo ecore a código fuente. El código se genera aplicando patrones de transformación. El resultado es un conjunto de clases java, que serán utilizadas más adelante en la herramienta DSL. Es muy importante en las propiedades del Domain Gen Model, llenar el campo Base Package con el nombre del paquete en minúscula, como lo muestra la siguiente figura.

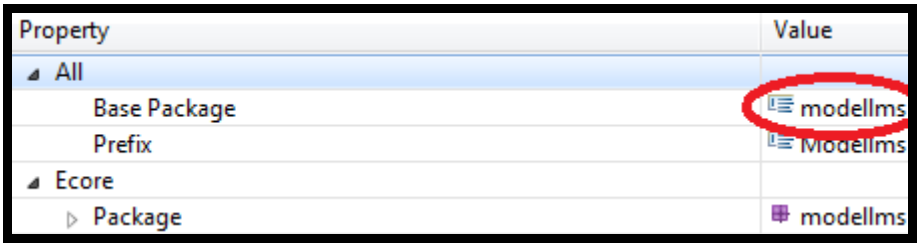


Figura 49 Propiedades del Domain Gen Model.

Luego que el Domain Gen Model este creado, se crea el Graphical Def Model, este es usado para definir las figuras, nodos, conexiones, etc. Que serán mostradas en nuestro diagrama. Para la creación del Graphical Def Model la hoja de ayuda lanzara un asistente para la creación del Simple Graphical Definition Model. Seleccione la carpeta que contiene el modelo en su proyecto, el nombre del fichero será "modellms.gmfgraph", seleccione el modelo ecore o el metamodelo LMS "modellms.ecore", y en la siguiente pagina del asistente seleccione la EClass a Mapear, esta es la clase que contiene todos los elementos para nuestro caso "LMSModel".

En la última página del asistente, seleccione la mínima cantidad de opciones elementos (element), conexiones (link) y etiquetas (label) para la herramienta DSL. Por el momento, solo estamos interesados en obtener un conjunto mínimo de elementos para validar el modelo, en este caso las herramientas de communications (Figura 45), la siguiente figura muestra los elementos, conexiones y etiquetas seleccionadas.

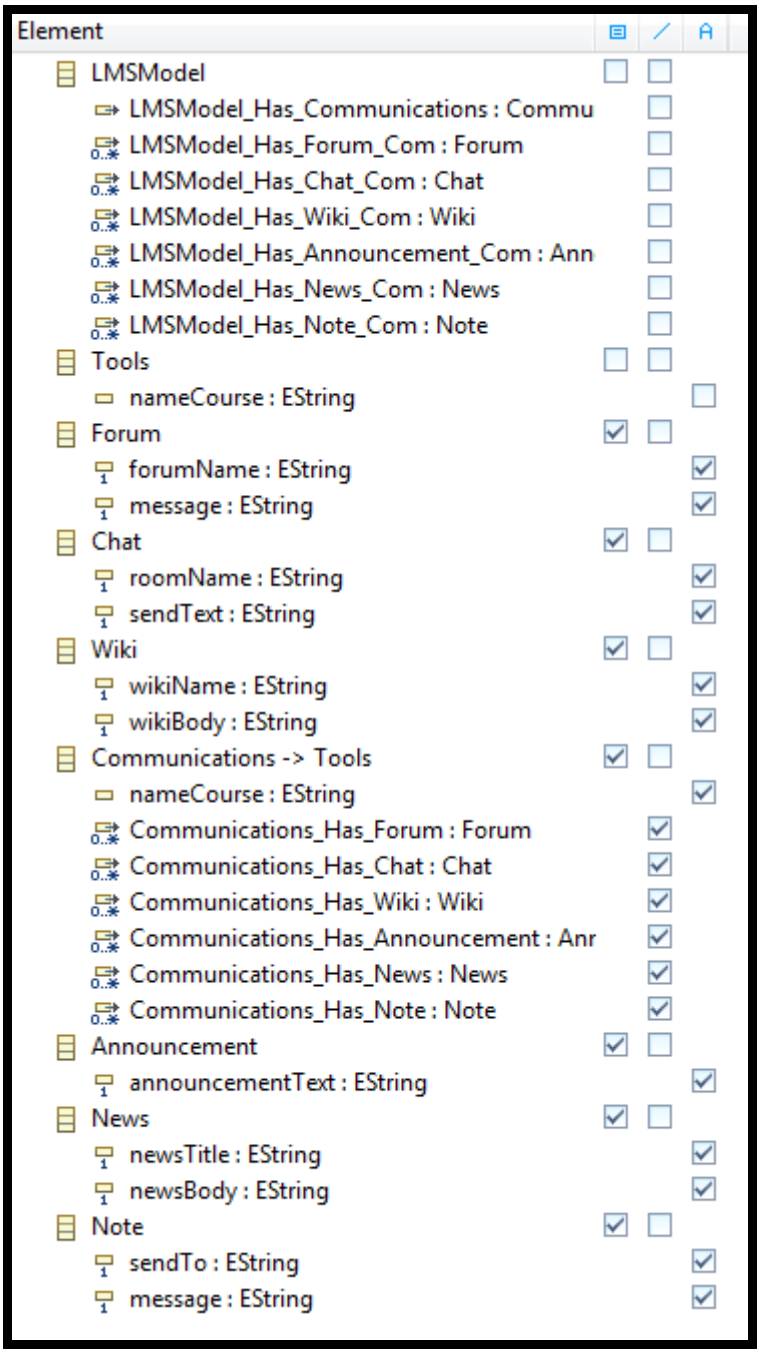


Figura 50 Graphical Definition Model: elementos, conexiones y etiquetas seleccionadas.

El resultado es un fichero con la siguiente estructura; un Canvas (lienzo) en la raíz con una galería de figuras base que contiene elementos de Rectángulos, Etiquetas y Conexiones de Polilineas. Estas son usadas por el correspondiente elemento Nodo, Etiqueta del diagrama y Conexión para representar los temas del domain model. Estos elementos pueden ser configurados, la siguiente figura muestra algunos elementos de "modellms.gmfgraph".

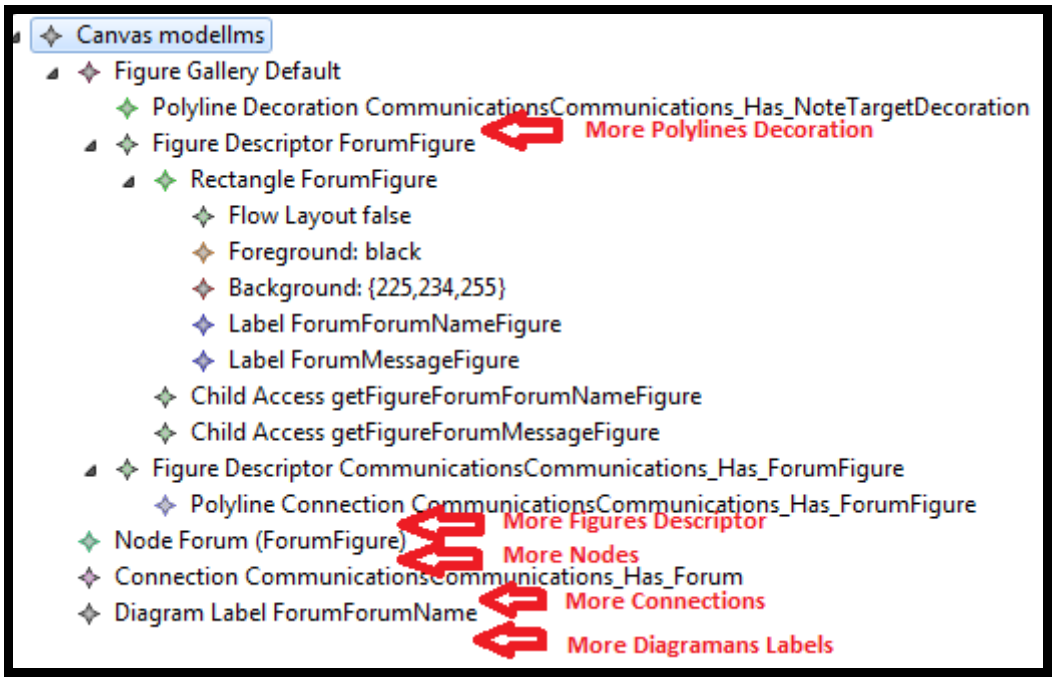


Figura 51 Algunos elementos de Graphical Definition Model.

El siguiente paso en el dashboard es el Tooling Def Model, este es usado para especificar la paleta (Pallete) de herramientas de creación, acciones, etc. Para los elementos gráficos. La hoja de ayuda nos guiara en un proceso muy similar para comenzar con la creación de un Simple Tooling Definition Model. De hecho los dos pasos son muy similares, ya que el dominio del modelo “modellms.ecore” será cargado y se escogerán las posibles herramientas que se necesitan, esto se muestra en la siguiente figura.

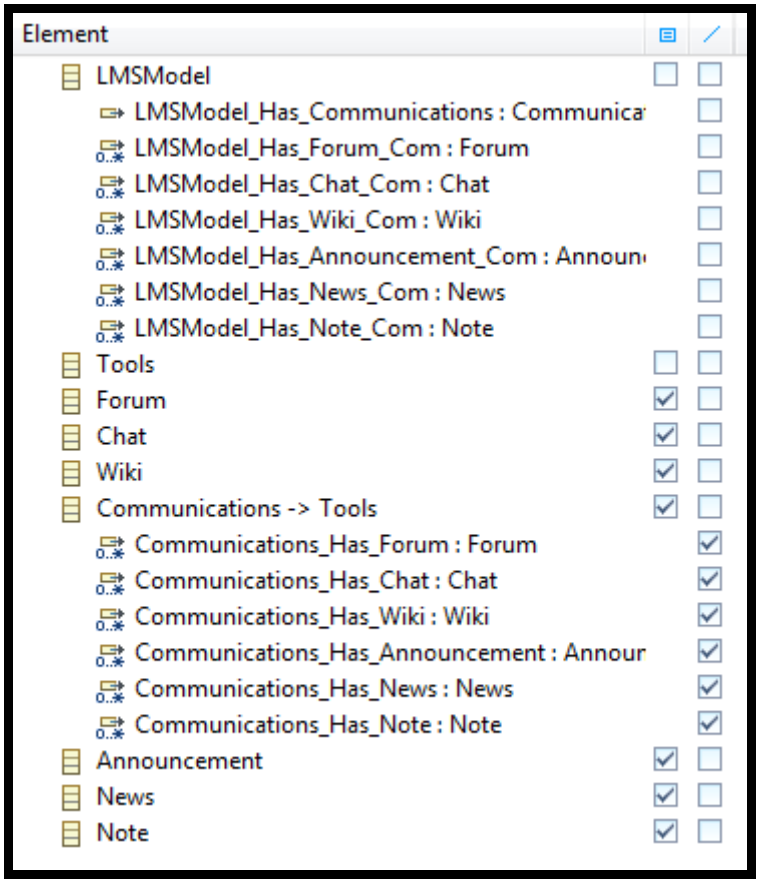


Figura 52 Elementos de Tooling Definition Model: elementos y conexiones seleccionadas.

Mirando el modelo provisto por nosotros, vemos que hay un elemento en el nivel superior 'Tool Registry", en el que encontramos una paleta (Palette). La "Palette" contiene un "Tools Group" con elementos de tipo "Creation Tool" para los nodos tema y conexiones para elementos de subtemas que fueron identificados por el asistente. Más adelante reorganizaremos y modificaremos estos elementos, por el momento lo dejaremos por defecto, realizaremos la definición del mapping, usted puede ir navegando por el modelo e inspeccionado las propiedades para familiarizarse un poco con ellas. Y el resultado es un fichero con el modelo proporcionado por nosotros, vemos que hay un nivel superior con un elemento "ToolRegistry" sobre el que encontramos una "Palette". La "palette" contiene un "Tool Group" con elementos "Creation Tool" que fueron identificados por el asistente. Estos elementos pueden ser reorganizados y se muestran en la siguiente figura.

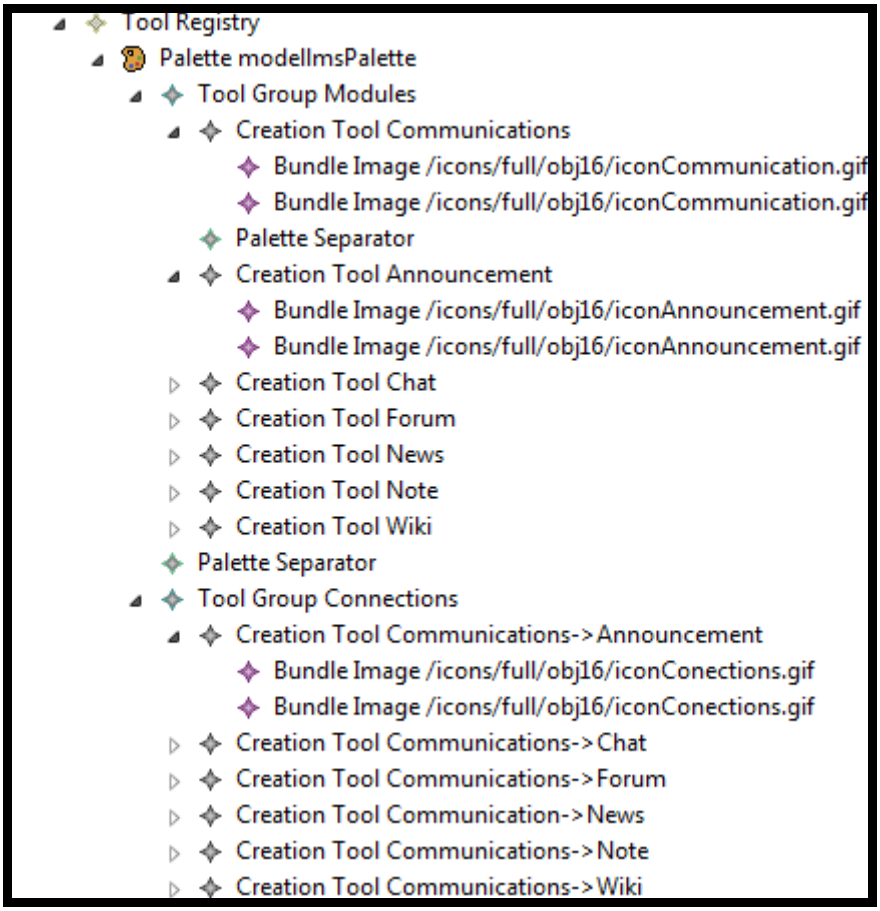


Figura 53 Elementos de Tooling Definition Model.

El “mapping model” es el siguiente paso en el dashboard, el “mapping model” combina los tres modelos: el “Domain Model”, el “Graphical Def Model”, y el “Tooling Def Model”. Continuando con nuestra hoja de ayuda se siguen las instrucciones para poner en marcha el asistente de “Guide Mapping Model Creation”. Seleccione la carpeta que contiene al modelo, el nombre del archivo para nosotros será “modellms.gmfmap” y seleccione los modelos "modellms.ecore", "modellms.gmfgraph" y "modellms.gmftool". En la siguiente página seleccione “LMSModel” como el elemento raíz, luego en aceptar el “modellms Palette” del “tooling model” seleccionado, después aceptar el “modellms diagram canvas” seleccionado y hacer clic en finalizar, el resultado se muestra en la siguiente figura.

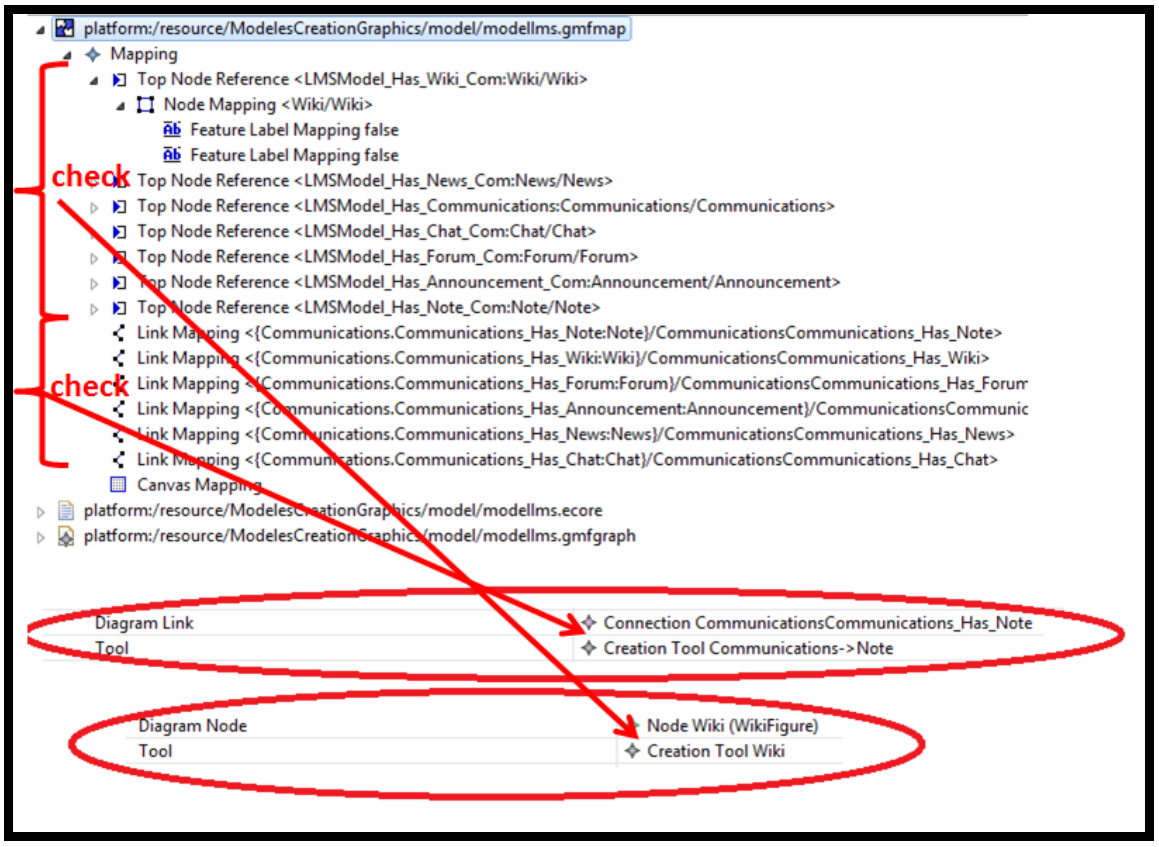


Figura 54 Mapping del modelo y propiedades.

Nota: Los elementos del mapping tienen problemas (Foundation, 2007). Una de las cosas que tenemos que hacer de forma manual es el “Labels Mapping” de la “graphical definition”. El resultado debe ser similar al de la figura anterior. Mientras en la vista de propiedades de cada elemento, se debe verificar el correcto “mapping” del “Diagram Node” con “Tool” y “Diagram Link” con “Tool”.

El último paso es la creación del “Diagram Editor Gen Model”. El modelo generador “modellms.gmfgen” establece las propiedades para la generación de código, similar al EMF genmodel. Para conseguir esto, haga clic derecho en el fichero del mapping “modellms.gmfmap” y seleccione “Create generator model...”. Cuando se le solicite, mantenga el nombre predeterminado “modellms.gmfgen”, haga clic derecho en el fichero “modellms.gmfgen” y seleccione "Generate diagram code" para continuar. Si todo va bien, usted verá un cuadro de dialogo con el mensaje “Code generation completed successfully”. Cierre el cuadro de dialogo, y observe el nuevo plugin “ModelesCreationGraphics.diagram” en su workspace.

Ahora que hemos generado el plugin necesario para nuestro diagrama, que está en ejecución en un nuevo workspace y se puede probar el diagrama. La configuración por defecto es de una nueva Eclipse Application en tiempo de ejecución y debería funcionar bien, mientras que existe una opción de ejecutar una configuración mínima

que incluye solo los plugins generados en la ejecución y sus dependencias de una configuración de org.eclipse.platform.ide.

Por último, crear un proyecto vacio (File->New->Java Project). En la sección “Examples” aparecerá una nueva opción “Modellms Diagram” y creara su nuevo diagrama. Esta es la herramienta DSL, para modelar las plataformas LMS, esto se muestra en la siguiente figura.

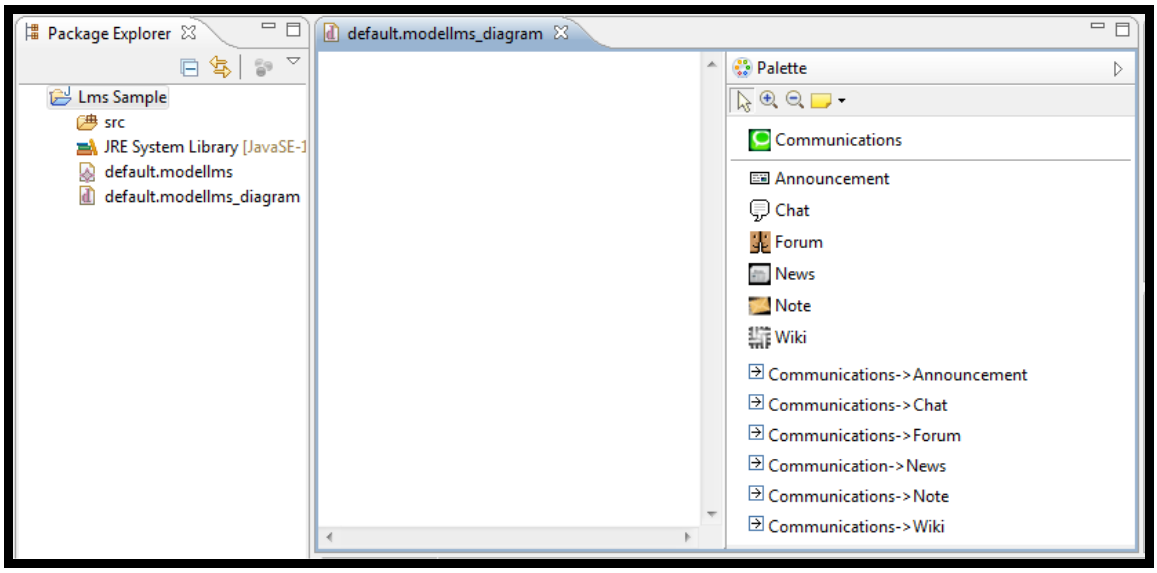


Figura 55 Herramienta DSL para modelar módulos en plataformas LMS.

El funcionamiento de la herramienta es muy sencillo, para crear los módulos basta con arrastrar los nodos de la “Pallete” al área de trabajo, rellenar los campos y conectarlos respetando las siguientes reglas:

- Un cursos solo puede tener un módulo de “Communications”.
- El módulo de “Communications” tiene cero o muchos: “Announcements”, “Chat”, “Forum”, “News”, “Note”, y “Wiki”.

La herramienta valida los siguientes casos:

- Solo permite un módulo de “Communications”.
- Los enlaces solo pueden corresponder entre el nodo “Communications” y su respectiva herramienta, por ejemplo “Communications -> Chat “, solo sirve para conectar el nodo “Communications” con el nodo “Chat”, la herramienta no permite utilizarlo en otro caso.
- Cuando se genere el código a desplegar, aquellos nodos que estén sueltos (sin conexión) no serán tenidos en cuenta para la creación del curso.

Es muy importante considerar que los elementos modelados con esta herramienta se desplegaran en un solo tema del curso, con lo cual es necesario considerar los nombres de los campos en cada elemento (Nodo o Módulo) como genéricos, para no tener que cambiar el modelo cada vez que se despliegue sobre el curso.

En la siguiente figura se puede visualizar un ejemplo, esta figura también muestra la relación de cada nodo y conexión con el XMI que genera la herramienta.

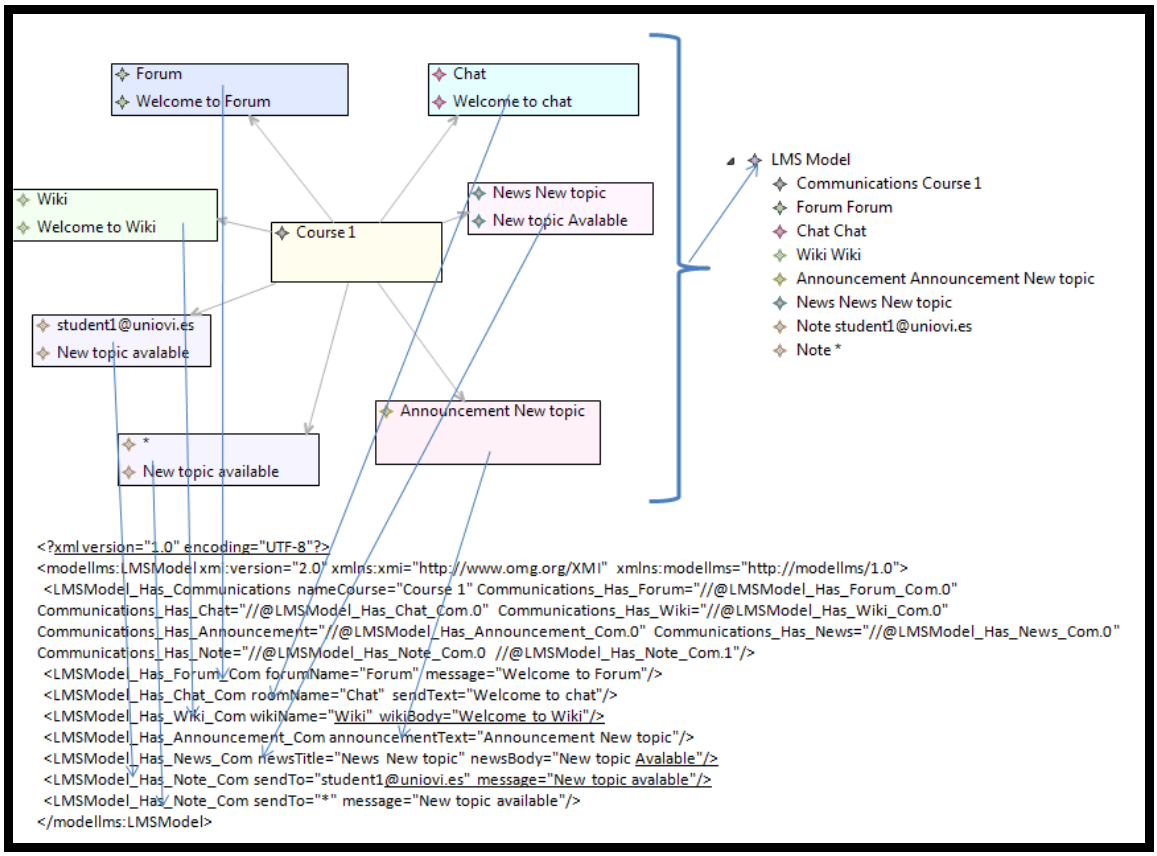


Figura 56 Ejemplo de un modelo en nuestra herramienta DSL y su correspondencia con XMI.

Observaciones a tener en cuenta en el modelado:

- Si se desea enviar una “Note” a todos los inscritos en un curso, en el campo “Send To” debe ir un *.
- Si se desea enviar una “Note” a un solo inscrito en un curso, en el campo “Send To” debe ir el correo completo con el que la persona se inscribió en la plataforma virtual. Si este correo no corresponde con el de la persona el mensaje no será enviado.

- Recuerde diligenciar todos los campos en los nodos, pues aquellos campos en los cuales el usuario no ponga información serán tomados en blanco y la herramienta no procesara esos nodos.

- Todos los campos aparecen con información por defecto pero esta información solo es de carácter orientativo, No implica información en los nodos.

En resumen, en esta sección se hablo de cómo crear un herramienta DSL grafica (M1), basada en un metamodelo generando (M2) y que a su vez se basa sobre el meta-meta modelo Ecore (M3), el siguiente paso es convertir este nuevo modelo que se obtuvo con la herramienta DSL y hacerle una transformación a código (M0), esta idea se explicara más adelante. Una completa guía sobre cómo crear un DSL en eclipse se encuentra en el anexo II, y todos los fuentes del DSL creado están en el anexo digital (+KiwiDSM) de este informe.

3.5 KiwiDSM v2.0: Mejora a la herramienta DSL grafica para la construcción de módulos de un LMS.

En esta sección se abordara la temática relacionada con las mejoras significativas realizadas a la herramienta KiwiDSM v1.0, estas mejoras tienen su fundamento en las observaciones realizadas por los testers a la primera versión de la herramienta.

Básicamente las observaciones se centran en:

- En la paleta de herramientas existen muchas conexiones que generan confusión a la hora de saber cual seleccionar de manera correcta.

- Las imágenes que aparecen en la barra de herramientas no son las mismas que en el área de trabajo, esto genera confusiones para determinar de qué es cada modulo.

- El nodo Communications en el área de trabajo tiene la misma forma que los otros nodos que se conectan a él, generando confusiones en su identificación.

- Los colores de todos los nodos son muy similares en el área de trabajo y hacen que el usuario se confunda.

- Las conexiones (flechas) del área de trabajo no se identifican visualmente.

A continuación se identificaran y priorizaran las observaciones a fin de trabajarlas como requisitos e ir solucionándolos gradualmente.

Tabla 13 Requisitos a mejorar en KiwiDSM v1.0.

Requisito	ID

En la paleta de herramientas existen muchas conexiones que generan confusión a la hora de saber cual seleccionar de manera correcta	R1
El nodo Communications en el área de trabajo tiene la misma forma que los otros nodos que se conectan a él, generando confusiones en su identificación	R2
Los colores de todos los nodos son muy similares en el área de trabajo y hacen que el usuario se confunda	R3
Las imágenes que aparecen en la barra de herramientas no son las mismas que en el área de trabajo, esto genera confusiones para determinar de qué es cada modulo	R4
Las conexiones (flechas) del área de trabajo no se identifican visualmente	R5

Para la solución a R1 se ha utilizado una adaptación a los metamodelos del patrón de diseño Abstract Factory (Fábrica Abstracta), es de recordar que el propósito de este patrón es "proporcionar una interfaz para crear familias de objetos relacionados o que dependan entre sí, sin especificar sus clases concretas"(GAMMA et al., 2003), a continuación se muestra el patrón original.

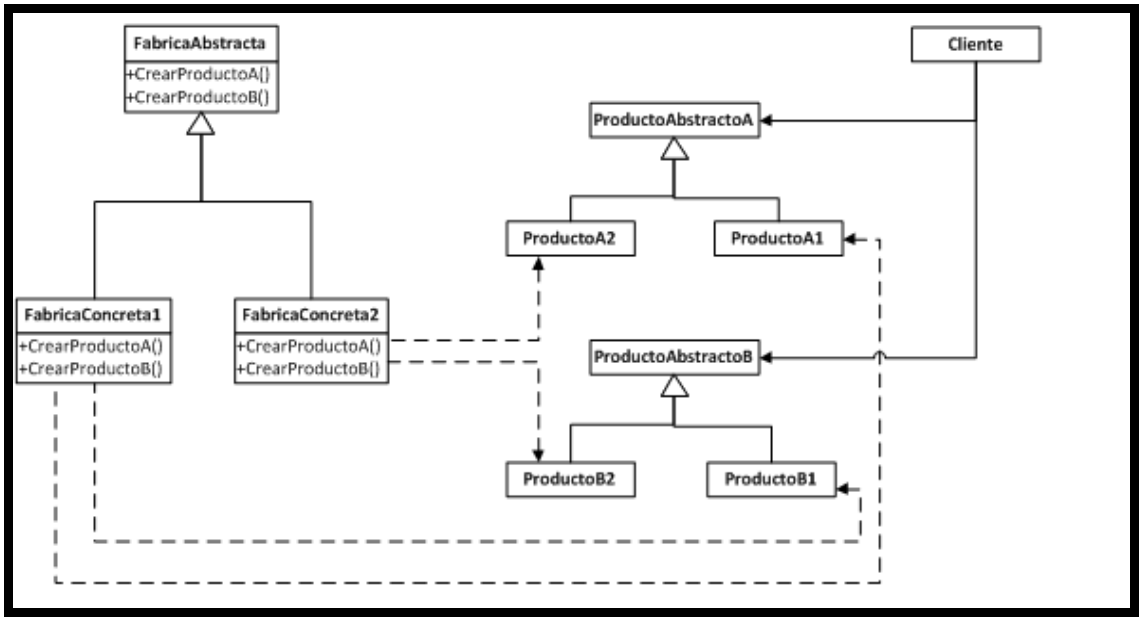


Figura 57 Patrón de diseño Fábrica Abstracta.

La adaptación lo que busca es realizar una generalización de las conexiones, ya que en el modelo original cada conexión del metamodelo tiene una representación en la paleta de herramientas y en el área de trabajo, lo que hace confundir al usuario, esto se visualiza en la siguiente imagen, allí existe una conexión de la EClass

"Communications" a cada uno de los módulos *"Annoncement, Chat, Forum, News, Note y Wiki"* tanto en la paleta de herramientas como en el área de trabajo.

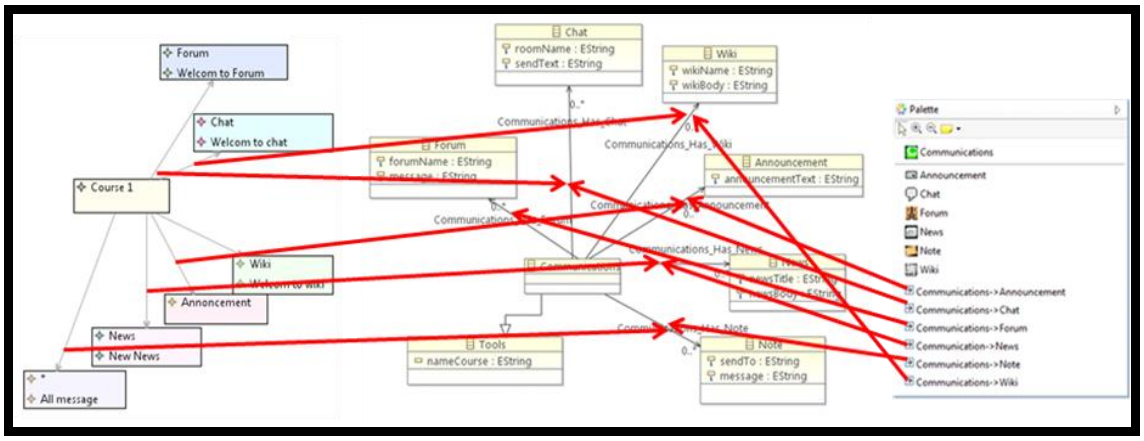


Figura 58 Correspondencia entre el metamodelo de Kiwi v1.0, la paleta de herramientas y el área de trabajo.

El nuevo planteamiento basado en el patrón fabrica abstracta, busca hacer una generalización de los módulos de tal forma que solo exista una conexión entre la EClass "Communications" y los módulos, esta será utilizada tanto en la paleta de herramientas como en el área de trabajo, como se muestra a continuación:

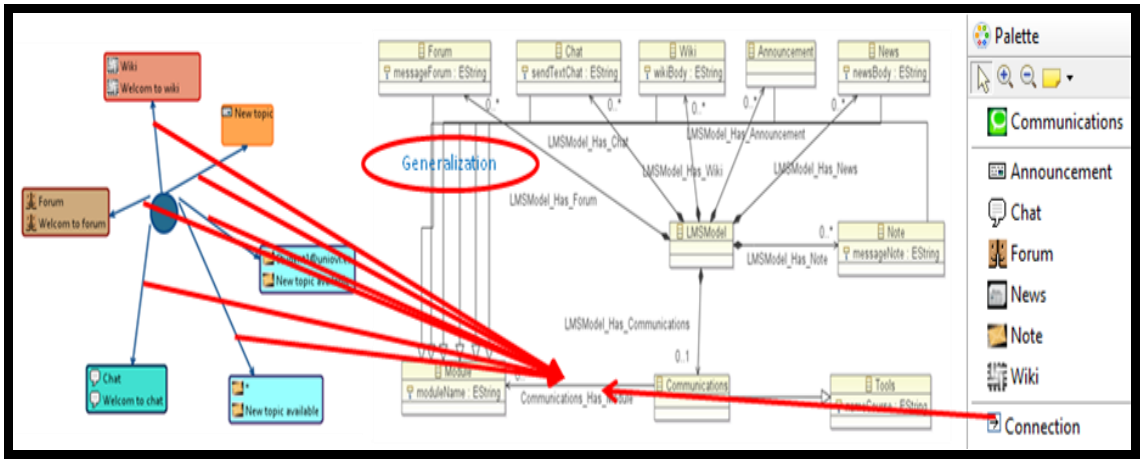


Figura 59 Correspondencia entre el metamodelo de Kiwi v2.0, la paleta de herramientas y el área de trabajo.

De esta forma se ha solucionado el primer requerimiento R1, la figura anterior muestra como en la paleta de herramientas ahora aparece solo un tipo de conexión genérica que será empleada en cualquier caso.

Para la solución al requerimiento R2 se ha modificado el *"Figure Descriptor"* en el *"Graphical Def Model"* del CommunicationsFigure por una *"Ellipse"* como se muestra en la siguiente figura.

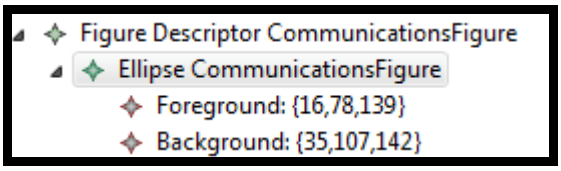


Figura 60 Cambio del figure descriptor de ComunicationsFigure por una Ellipse.

Para solucionar el requerimiento R3 se ha modificado en el *"Figure Descriptor"* los valores de los hijos *"Foreground"* y *"Background"* de cada *"Rounded Rectangle"* que pertenecen al *"Figure Descriptor"* y también se modificaron las propiedades por cada modulo Forum, Chat, Wiki, Announcement, News y Note. Quedando el *"Figure Descriptor"* como se muestra en la siguiente figura.

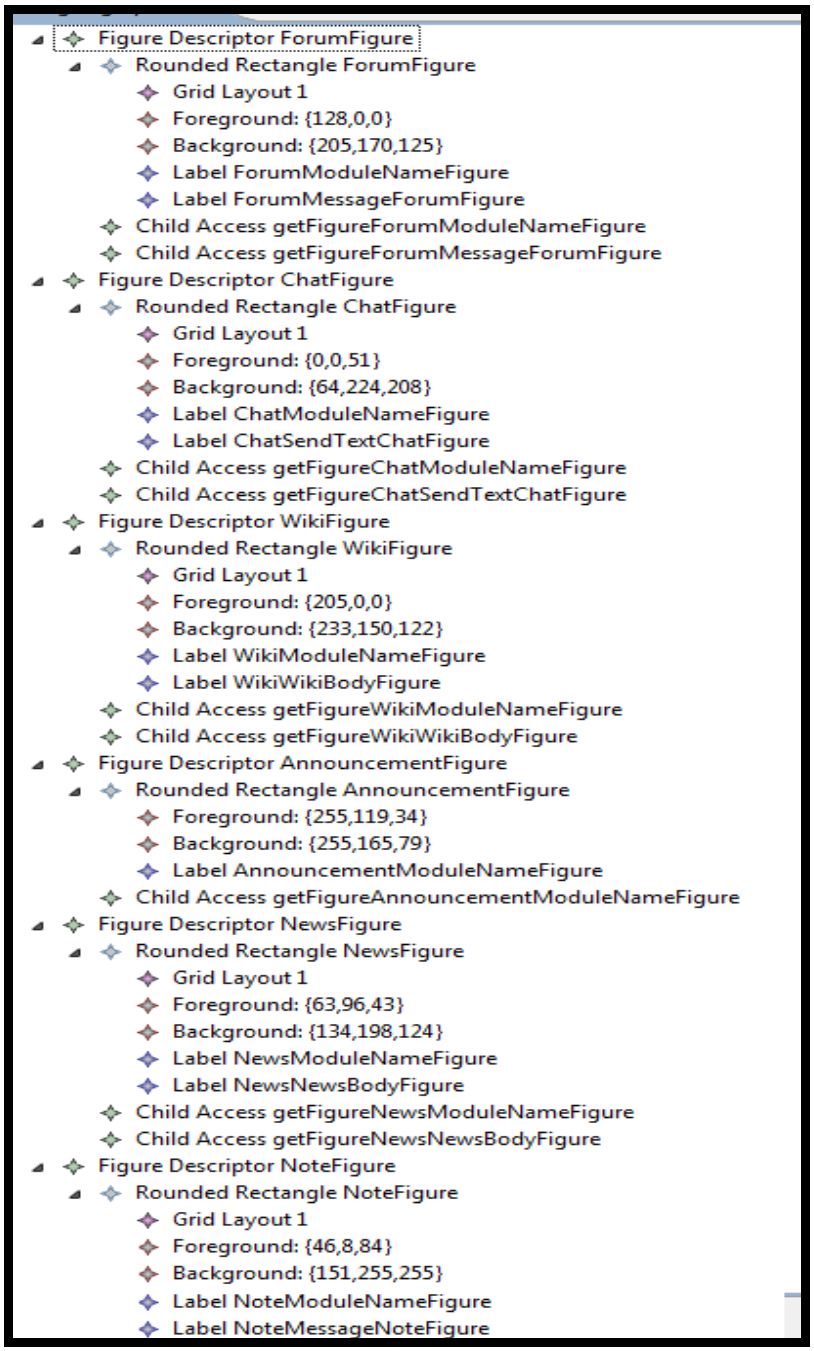


Figura 61 Cambio del figure descriptor de para los módulos Forum, Chat, Wiki, Announcement, News y Note.

Con las aplicaciones de estos cambios, el despliegue de los módulos en el área de trabajo quedaría como se muestra a continuación, mejorando sustancialmente la presentación de la aplicación.

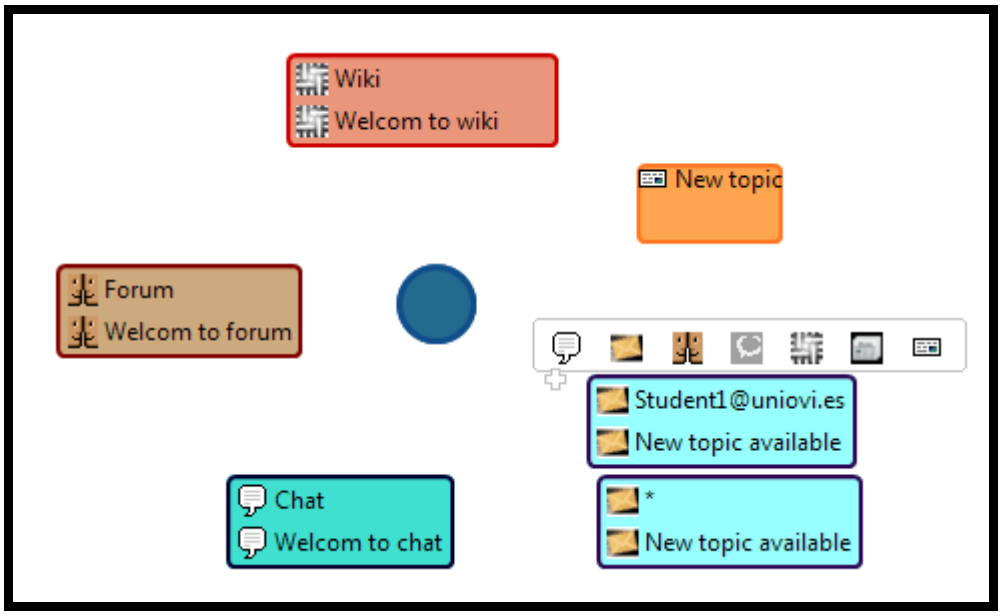


Figura 62 Cambio del despliegue de módulos Forum, Chat, Wiki, Announcement, News y Note en el área de trabajo

Para dar solución al requerimiento R4, se han reemplazado las imágenes en el directorio *".edit/icons/full/obj16/"* por las que se consideren necesarias, conservando el mismo nombre y extensión original, para ello es necesario que las imágenes de cada *"Creation Tools"* sean de tipo *"Default Image*", la apariencia final en el área de trabajo es como se muestra en la imagen anterior y tanto el árbol de directorios como el *"Pallet"* en el *"Tooling Def Model"* quedarían como se muestra a continuación.

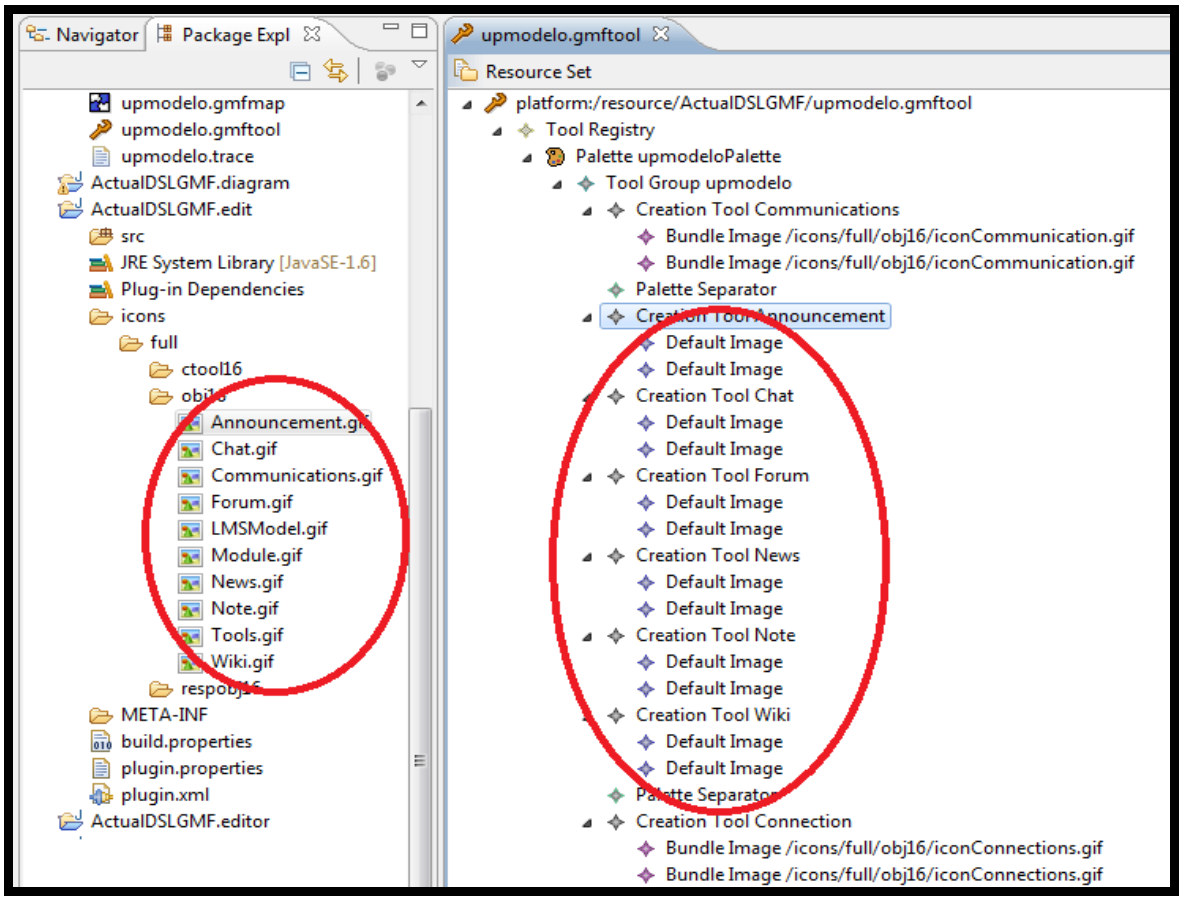


Figura 63 Modificación en el árbol de directorios y en "Pallet" del "Tooling Def Model" para que las imágenes de la barra de herramientas sean las mismas que en el área de trabajo.

Finalmente para suplir el requerimiento R5, se ha modificado en el *"Figure Descriptor"* los valores del hijo *"Foreground"* en el *"Polyline Connection"* que pertenecen al *"Figure Descriptor"* de la conexión *"Communications_Has_Module_Figure"* y también se modifico su propiedad *"Line Width"* como se muestra en la siguiente figura.

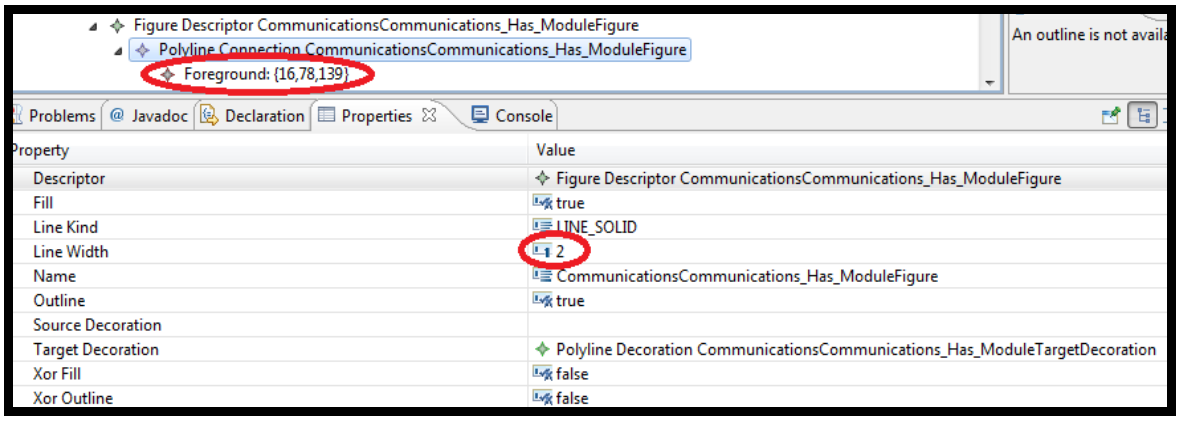


Figura 64 Cambio del figure descriptor para la conexión de los módulos.

Por último la apariencia final de la herramienta KiwiDSM v2.0 queda como se muestra a continuación.

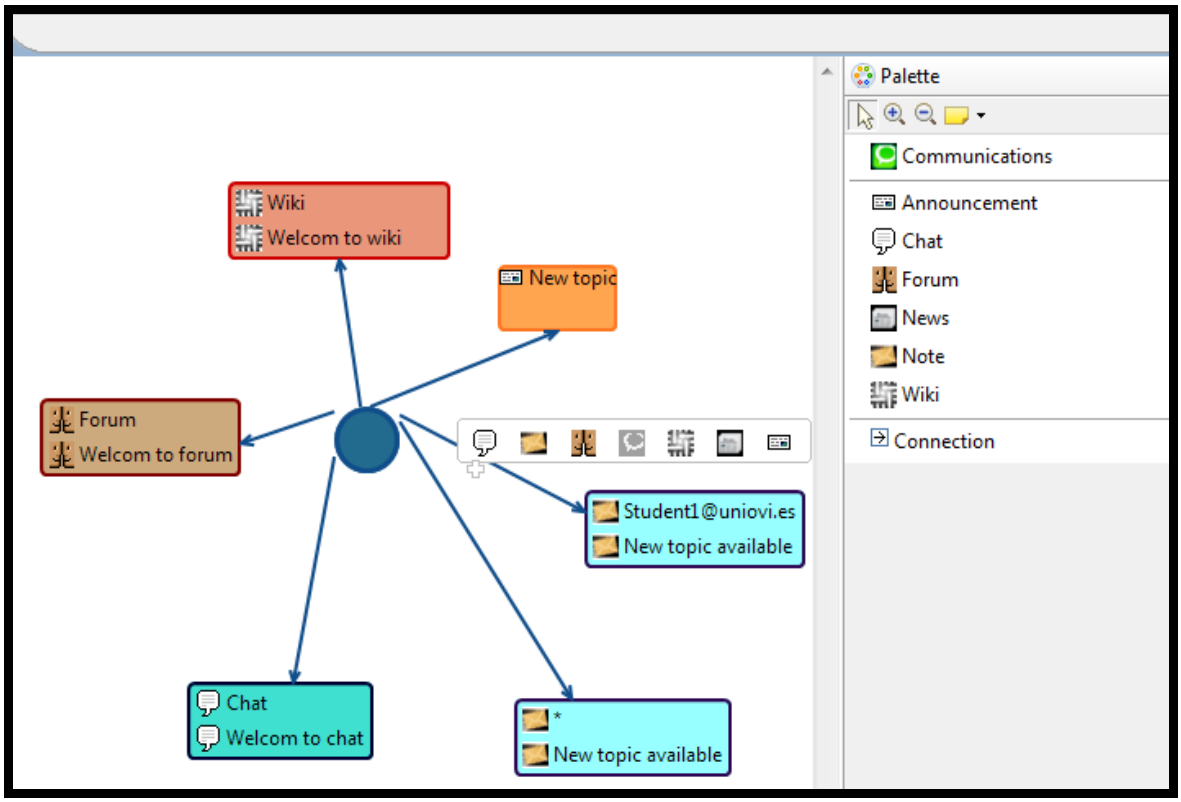


Figura 65 Apariencia final de la herramienta KiwiDSM v2.0.

3.6 Prototipo de transformaciones con MOFScript

En esta sección se abordara el problema de transformar un modelos en el nivel (M1) a código para su despliegue sobre la plataforma especifica, a este tipo de transformaciones se les llama Transformación de "Model to text" o "M2T", y para nuestro caso se empleara una tecnología llamada MOFScript (Foundation, 2010e), esta es un plugin de eclipse que convertirte (parser) modelo a texto con un lenguaje basado en reglas.

MOFScript es un lenguaje basado en reglas, presentado por la OMG, para realizar transformaciones de modelo a texto (M2T). Se puede instalar como un plugin de eclipse y tanto su sintaxis como su uso resultan bastante sencillos. Un excelente manual para el uso de MOFScript es (Oldevik, 2009) y para la instalación y uso de MOFScript sobre eclipse se pude consultar el anexo II.

A continuación mostraremos los pasos a seguir para generar código a partir del modelo LMS creado con nuestra herramienta DSL. Para no mezclar la generación de módulos de un LMS con otro se han creado plantillas de transformaciones MOFScript para cada plataforma LMS, que definen por separado las transformaciones necesarias

para pasar de Modelo a Texto, es decir, los ficheros “modellmsTransformationMoodle.m2t”, “modellmsTransformationClaroline.m2t” y ““modellmsTransformationATutor.m2t” considerados dentro del proyecto ModelesCreationGraphics (así se ha llamado el proyecto en eclipse pero usted puede cambiar este nombre) en eclipse realizan las transformaciones para Moodle, Claroline y ATutor respectivamente. Antes de explicar el contenido de cada uno de ellos y para poder comprenderlos es necesario explicar cómo se desarrolla un módulo para Moodle, Claroline y ATutor, que fueron las plataformas LMS sobre las cuales hemos probado el trabajo realizado y por qué se seleccionaron estas plataformas para las pruebas.

3.6.1 Selección de las plataformas LMS sobre las cuales se desplegara el modelo

Debido a que muchos de los trabajos realizados en esta investigación tienen su fundamento en cinco plataformas LMS Moodle, Claroline, ATutor, DotLRN y Sakai, en esta etapa del proyecto se definirán sobre cuales se realizara el despliegue final de los módulos creados, en otras palabras se seleccionaran las plataformas sobre las cuales los módulos modelados con la herramienta DSL serán creados. Para esta selección se debe tomar una muestra representativa y considerando que son cinco las plataformas iníciales una buena muestra serian tres de ellas, ya que sobrepasan la mitad de la muestra inicial.

Las tres plataformas seleccionadas fueron Moodle, Claroline y ATutor. Moodle se selecciono pues en el estudio realizado por (Ávarez, 2010) muestran que para el año 2009 esta plataforma era la más utilizada en instituciones universitarias en España, con más del 45% de todo el mercado, por otro lado Claroline y ATutor fueron seleccionar por manejar la misma arquitectura tecnología que Moodle, es decir un servidor de aplicaciones Apache, una base de datos MySQL y PHP, como principal lenguaje de programación.

3.6.2 Creación de módulos en Moodle

Antes de desarrollar un módulo para actividades es necesario conocer el esquema que utiliza Moodle, una buena guía para el desarrollo de módulos para Moodle se puede encontrar en su Web (Moodle, 2011) vinculo Develoment, nosotros además de estos recursos hemos utilizado los trabajos de (Ivorra, 2009) y (Gonzàlez, 2009) para conocer cómo desarrollar módulos para Moodle. En Moodle los módulos se almacenan en la carpeta Moodle/mod, cada uno en un directorio, la estructura general de ficheros y directorios debe ser así:

- mod_form.php: Formulario para crear o modificar una instancia de la actividad.
- version.php: para definir meta información, como por ejemplo la versión del módulo.

- lang/: directorio para almacenar los archivos de idioma del módulo. El módulo debe tener archivos de idioma que contenga las cadenas para ese módulo. Deberán ser al menos en inglés y traducidos a los idiomas de los usuarios finales que utilicen la actividad.
- db/: Directorio donde se almacenarán los ficheros con las tablas de las bases de datos necesarias para la actividad.
    - access.php: Fichero de permisos del módulo. Los permisos no son obligatorios pero sí muy recomendables para garantizar qué usuarios pueden acceder a las distintas partes del módulo.
    - install.xml: Fichero que describe la estructura de las tablas del módulo.
    - upgrade.php: código de actualización, aquí es donde se deben de hacer las alteraciones de las tablas, si las hay, entre versiones.
- index.php: Este fichero sirve para mostrar todas las instancias de una actividad en un curso, es decir, una lista con todas las instancias del mismo módulo.
- view.php: Esta es la página que muestra una instancia de la actividad.
- lib.php: librería de funciones del módulo. En este fichero se implementarán todas las funciones y procedimientos del módulo. Si el módulo se llama ejemplo, entonces las funciones mínimas y obligatorias que ha de tener la actividad tienen que ser de la forma:
    - ejemplo_install(): Acciones a realizar al instalar el módulo.
    - ejemplo_add_instance(): código para añadir una nueva instancia.
    - ejemplo_update_instance(): función para actualizar una instancia existente.
    - ejemplo_delete_instance(): código para borrar una instancia.
    - ejemplo_user_outline(): da un resumen concreto de la actividad de un usuario.
    - ejemplo_user_complete(): devuelve un informe más detallado de la contribución de un usuario.
    - ejemplo_get_view_actions(): Clasifica las acciones para el log. Se usa en el informe de participación.
    - Todas las funciones, procedimientos y constantes, creados en lib.php,
- settings.php (opcional): Formulario con las opciones generales del módulo.

Los permisos permiten establecer las diferentes vistas que tendrán los distintos tipos de usuarios para un módulo. Los tipos de usuarios son los roles en Moodle. Ejemplos de roles son administrador, profesor, estudiante, etc. Asignando permisos a la actividad se puede restringir el acceso a cierta información, configurar distintos tipos de vistas para los diferentes roles, establecer capacidades para los administradores, etc. Para establecer permisos en la actividad, se creará un fichero denominado access.php, en la carpeta db. Para nuestro módulo que se denomina plantilladsl, queremos establecer que usuario (rol) tendrá acceso a la vista del mismo, y que los profesores puedan acceder en modo edición a él, nuestro fichero quedaría de la siguiente manera.

```
<?php
defined('MOODLE_INTERNAL') || die();
$capabilities = array(
    'mod/newmodule:view' => array(
        'captype' => 'read',
        'contextlevel' => CONTEXT_MODULE,
        'legacy' => array(
            'guest' => CAP_ALLOW,
            'student' => CAP_ALLOW,
            'teacher' => CAP_ALLOW,
            'editingteacher' => CAP_ALLOW,
            'admin' => CAP_ALLOW
        )
    ),
    'mod/newmodule:submit' => array(
        'riskbitmask' => RISK_SPAM,
        'captype' => 'write',
        'contextlevel' => CONTEXT_MODULE,
        'legacy' => array(
            'guest' => CAP_ALLOW,
            'student' => CAP_ALLOW,
            'teacher' => CAP_ALLOW,
            'editingteacher' => CAP_ALLOW,
            'admin' => CAP_ALLOW
        )
    ),
);
```

Código Fuente 1 Fichero de permisos para Moodle access.php.

CONTEXT_MODULE establece el contexto en el que se aplicará el permiso, en este caso, al módulo, y CAP_ALLOW un tipo de habilidad que se asigna a los roles.

3.6.2.1 El fichero mod_form.php

Como ya se ha comentado anteriormente éste es uno de los ficheros obligatorios de todo módulo y mediante el cual se configurará cada instancia del módulo que se añade a un curso. Moodle proporciona un mecanismo estándar para este paso, y para ello está definida la clase Moodleform_mod. La clase para nuestro caso se llama

mod_plantilladsl_mod_form. Se hereda de esta clase y se crea el método definition () mediante el cual definiremos el formulario que presentará las distintas opciones a configurar de la instancia. Aquí será en donde se llamaran a las funciones que están en lib.php, Es prudente aclarar que el fichero mod_form.php será creado automáticamente por la plantilla MOFScript acorde al modelo que se cree. Un ejemplo de cómo quedaría este fichero se puede ver a continuación:

```
<?php
require_once($CFG->dirroot.'/course/Moodleform_mod.php');
require_once($CFG->dirroot.'/course/lib.php');
require_once('../config.php');
class mod_plantilladsl_mod_form extends Moodleform_mod {
      def_forum_dsl ('Foro  Todo','Welcom Foro Todo');
      def_chat_dsl ('Chat Todo','Welcome Chat Todo');
      def_wiki_dsl ('Wiki Todo','Welcom wiki todo');
    def_Announcement_dsl('Todo Prueba de anuncio Todo');
    def_news_dsl('Todo Ultima noticia de Todo ', 'Todo Ultima noticia
de Todo ');
    def_note_dsl('*', 'Todo Mensaje para estudiates de plantilla
Todo');
            }
      }
```

Código Fuente 2 Fichero mod_form.php para Moodle.

3.6.2.2 El fichero install.xml

Este fichero define las tablas que necesita el módulo. Desde que se implantó XMLDB en Moodle no es necesario tener un fichero distinto para cada base de datos soportada, la plataforma provee mecanismos para la creación de las tablas, para el acceso y para la modificación de los datos. La primera tabla tiene el mismo nombre del módulo y es obligatoria, así como los campos id, course y name. El resto de campos que proporciona la plantilla son recomendables también. Para este módulo se han mantenido los campos que la plantilla proveía por defecto, el código de este fichero se muestra a continuación:

```
<?xml version="1.0" encoding="UTF-8" ?>
<XMLDB PATH="mod/plantilladsl/db" VERSION="201101231" COMMENT="XMLDB
file for Moodle mod/plantilladsl"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="../../../lib/xmldb/xmldb.xsd">
  <TABLES>
    <TABLE NAME="plantilladsl" COMMENT="Tabla para convertir a codigo
de Moodle el DSL">
      <FIELDS>
        <FIELD NAME="id" TYPE="int" LENGTH="10" NOTNULL="true"
UNSIGNED="true" SEQUENCE="true" NEXT="course"/>
        <FIELD NAME="course" TYPE="int" LENGTH="10" NOTNULL="true"
UNSIGNED="true" SEQUENCE="false" COMMENT="Course plantilladsl activity
belongs to" PREVIOUS="id" NEXT="name"/>
        <FIELD NAME="name" TYPE="char" LENGTH="255" NOTNULL="true"
SEQUENCE="false" COMMENT="name field for Moodle instances"
```

```
PREVIOUS="course" NEXT="intro"/>
        <FIELD NAME="intro" TYPE="text" LENGTH="medium"
NOTNULL="false" SEQUENCE="false" COMMENT="General introduction of the
plantilladsl activity" PREVIOUS="name" NEXT="introformat"/>
        <FIELD NAME="introformat" TYPE="int" LENGTH="4" NOTNULL="true"
UNSIGNED="true" DEFAULT="0" SEQUENCE="false" COMMENT="Format of the
intro field (MOODLE, HTML, MARKDOWN...)" PREVIOUS="intro"
NEXT="timecreated"/>
        <FIELD NAME="timecreated" TYPE="int" LENGTH="10"
NOTNULL="true" UNSIGNED="true" SEQUENCE="false" PREVIOUS="introformat"
NEXT="timemodified"/>
        <FIELD NAME="timemodified" TYPE="int" LENGTH="10"
NOTNULL="true" UNSIGNED="true" DEFAULT="0" SEQUENCE="false"
PREVIOUS="timecreated"/>
      </FIELDS>
      <KEYS>
        <KEY NAME="primary" TYPE="primary" FIELDS="id"/>
      </KEYS>
      <INDEXES>
        <INDEX NAME="course" UNIQUE="false" FIELDS="course"/>
      </INDEXES>
    </TABLE>
  </TABLES>
</XMLDB>
```

Código Fuente 3 Fichero install.xml para Moodle.

3.6.2.3 El fichero upgrade.php

En este fichero se encuentran las acciones para actualizar a una versión posterior de un módulo. Para ello en primer lugar se comprueba la versión del módulo para saber las acciones a realizar. Para posteriormente mediante XMLDB realizar la modificaciones a las tablas que las que se precisen. Como ya se ha comentado anteriormente XMLDB permite abstraernos del gestor de bases de datos utilizado, empleando un mecanismo común para todos, para nuestro caso no hay actualizaciones más sin embargo el código de este fichero seria:

```
<?php
function xmldb_plantilladsl_upgrade($oldversion) {
    global $DB;
    $dbman = $DB->get_manager(); // loads ddl manager and xmldb
classes
    if ($oldversion < 2007040101) {
        $table = new xmldb_table('plantilladsl');
        $field = new xmldb_field('timecreated', XMLDB_TYPE_INTEGER,
'10', XMLDB_UNSIGNED, XMLDB_NOTNULL, null, '0',
            'introformat');
        if (!$dbman->field_exists($table, $field)) {
            $dbman->add_field($table, $field);
        }
        $table = new xmldb_table('plantilladsl');
        $field = new xmldb_field('timemodified', XMLDB_TYPE_INTEGER,
```

```
'10', XMLDB_UNSIGNED, XMLDB_NOTNULL, null, '0',
                'timecreated');
            if (!$dbman->field_exists($table, $field)) {
                $dbman->add_field($table, $field);
            }
            $table = new xmldb_table('plantilladsl');
            $index = new xmldb_index('courseindex', XMLDB_INDEX_NOTUNIQUE,
array('course'));
            if (!$dbman->index_exists($table, $index)) {
                $dbman->add_index($table, $index);
            }
        }
        return true;
}
```

Código Fuente 4 Fichero upgrade.php para Moodle.

3.6.2.4 El fichero lib.php

En este fichero tenemos las funciones básicas del módulo como se comentaba anteriormente, y las clases que se crean oportunas, así como las constantes del módulo. Lo más importante que podemos aprender de esta librería es a acceder a la base de datos mediante los mecanismos que proporciona la plataforma.

- get_records(tabla,[campo],[valor],...): Esta función nos devolverá todos los registros de la tabla en forma de objetos, pudiéndolos filtrar por campo y valor. En caso de no encontrar coincidencias devuelve false.

- insert_record: Esta función nos permite insertar registros en una tabla. Tiene dos campos obligatorios, la tabla sobre la que actuar y el objeto a insertar. De dicho objeto se insertarán las propiedades que tengan el mismo nombre que los campos de la tabla. Retorna el id del nuevo registro insertado.

- update_record: Función para actualizar un registro de la base de datos. El objeto ha de tener un campo id por el cual buscar en la tabla.

- delete_records: Función para borrar registros de una tabla. Podemos especificar hasta tres pares (campo, valor) para filtrar los registros que deseamos borrar.

Dado que tanto para obtener e insertar datos en las tablas se utilizan objetos, en este fichero se crean los objetos para poder usarlos cómodamente a lo largo de todo el módulo.

Este es el otro fichero que generarnos con nuestras plantillas MOFScript a partir del módulo que se construya un ejemplo de este fichero se muestra a continuación:

```
<?php
require_once($CFG->libdir.'/filelib.php');
```

```
defined('MOODLE_INTERNAL') || die();
function plantilladsl_add_instance($plantilladsl) {
    global $DB;
    $plantilladsl->timecreated = time();
    return $DB->insert_record('plantilladsl', $plantilladsl);
}
function plantilladsl_update_instance($plantilladsl) {
    global $DB;

    $plantilladsl->timemodified = time();
    $plantilladsl->id = $plantilladsl->instance;
    # You may have to add extra stuff in here #
    return $DB->update_record('plantilladsl', $plantilladsl);
}
function plantilladsl_delete_instance($id) {
    global $DB;
    if (! $plantilladsl = $DB->get_record('plantilladsl', array('id'
=> $id))) {
        return false;
    }
    $DB->delete_records('plantilladsl', array('id' => $plantilladsl-
>id));
    return true;
}
function plantilladsl_user_outline($course, $user, $mod,
$plantilladsl) {
    $return = new stdClass;
    $return->time = 0;
    $return->info = '';
    return $return;
}
function plantilladsl_user_complete($course, $user, $mod,
$plantilladsl) {
    return true;
}
function plantilladsl_print_recent_activity($course, $viewfullnames,
$timestart) {
    return false;  //  True if anything was printed, otherwise false
}

function plantilladsl_cron () {
    return true;
}
function plantilladsl_get_participants($plantilladslid) {
    return false;
}
function plantilladsl_scale_used($plantilladslid, $scaleid) {
    global $DB;
    $return = false;
    return $return;
}
function plantilladsl_scale_used_anywhere($scaleid) {
    global $DB;
    if ($scaleid and $DB->record_exists('plantilladsl', 'grade', -
$scaleid)) {
```

```
        return true;
    } else {
        return false;
    }
}
function plantilladsl_uninstall() {
    return true;
}
function def_forum_dsl($foronombre, $forointroduccion){
           global $USER, $CFG, $COURSE;
        $section = required_param('section', PARAM_INT);
        $course = required_param('course', PARAM_INT);

    $forum->timemodified = time();
    $forum->course   = $course;
    $forum->name     = $foronombre;
    $forum->intro    = $forointroduccion;
    if (empty($forum->assessed)) {
        $forum->assessed = 0;
    }
    if (empty($forum->ratingtime) or empty($forum->assessed)) {
        $forum->assesstimestart  = 0;
        $forum->assesstimefinish = 0;
    }
    if (!$forum->id = insert_record('forum', $forum)) {
        return false;
        }
    if (record_exists("forum_subscriptions", "userid", $USER->id,
"forum", $forum->id)) {
        return true;
    }
    $sub = new object();
    $sub->userid  = $USER->id;
    $sub->forum = $forum->id;
     insert_record("forum_subscriptions", $sub);
    $sectioncourse->course = $course;
    $sectioncourse->module = 6;
    $sectioncourse->instance = $forum->id;
    $sectioncourse->added = time();
    $sectioncourse->section = $section;
    $sectioncourse->is_lams = 1;
    $sectioncourse->id = insert_record('course_modules',
$sectioncourse);
    $insertsection = get_record("course_sections", "section",
$section, "course", $course);
    $insertseccion->sequence =  $insertsection-
>sequence.','.$sectioncourse->id;
    $sequence='"'.$insertseccion->sequence.'"';
    global $db;
    $query = "UPDATE mdl_course_sections SET sequence=".$sequence."
WHERE id=".$insertsection->id;
     $db->Execute($query);
    }
function def_chat_dsl($roomname, $sendtext){
           global $USER, $CFG, $COURSE;
```

```
        $section = required_param('section', PARAM_INT);
        $course = required_param('course', PARAM_INT);
    $chat->timemodified = time();
    $chat->course    = $course;
    $chat->name      = $roomname;
    $chat->intro     = $sendtext;
    if (!$chat->id = insert_record('chat', $chat)) {
        return false;
        }
    $sectioncourse->course = $course;
    $sectioncourse->module = 2;
    $sectioncourse->instance = $chat->id;
    $sectioncourse->added = time();
    $sectioncourse->section = $section;
    $sectioncourse->is_lams = 1;

    $sectioncourse->id = insert_record('course_modules',
$sectioncourse);
    $insertsection = get_record("course_sections", "section",
$section, "course", $course);
    $insertseccion->sequence =  $insertsection-
>sequence.','.$sectioncourse->id;
    $sequence='"'.$insertseccion->sequence.'"';
    global $db;
    $query = "UPDATE mdl_course_sections SET sequence=".$sequence."
WHERE id=".$insertsection->id;
     $db->Execute($query);
    }
function def_wiki_dsl($wikiname, $wikitext){
           global $USER, $CFG, $COURSE;
        $section = required_param('section', PARAM_INT);
        $course = required_param('course', PARAM_INT);
    $wiki->timemodified = time();
    $wiki->course    = $course;
    $wiki->name      = $wikiname;
    $wiki->summary   = $wikitext;
    $wiki->pagename = $wikiname;
    if (!$wiki->id = insert_record('wiki', $wiki)) {
        return false;
        }
    $sectioncourse->course = $course;
    $sectioncourse->module = 17;
    $sectioncourse->instance = $wiki->id;
    $sectioncourse->added = time();
    $sectioncourse->section = $section;
    $sectioncourse->is_lams = 1;
    $sectioncourse->id = insert_record('course_modules',
$sectioncourse);
    $insertsection = get_record("course_sections", "section",
$section, "course", $course);
    $insertseccion->sequence =  $insertsection-
>sequence.','.$sectioncourse->id;
    $sequence='"'.$insertseccion->sequence.'"';
    global $db;
    $query = "UPDATE mdl_course_sections SET sequence=".$sequence."
```

```
WHERE id=".$insertsection->id;
     $db->Execute($query);
    }
        $section = required_param('section', PARAM_INT);    // Course
section ID
        $course = required_param('course', PARAM_INT);       // Course
Module ID
    $label->timemodified = time();
    $label->course       = $course;
    $label->name         = $announcementname;
    $label->content      = $announcementname;
    if (!$label->id = insert_record('label', $label)) {
        return false;
        }
    $sectioncourse->course = $course;
    $sectioncourse->module = 10;
    $sectioncourse->instance = $label->id;
    $sectioncourse->added = time();
    $sectioncourse->section = $section;
    $sectioncourse->is_lams = 1;
    $sectioncourse->id = insert_record('course_modules',
$sectioncourse);
    $insertsection = get_record("course_sections", "section",
$section, "course", $course);
    $insertseccion->sequence =  $insertsection-
>sequence.','.$sectioncourse->id;
    $sequence='"'.$insertseccion->sequence.'"';
    global $db;
    $query = "UPDATE mdl_course_sections SET sequence=".$sequence."
WHERE id=".$insertsection->id;
     $db->Execute($query);
    }
function def_news_dsl($newstitle, $newsbody){
    global $USER, $CFG, $COURSE;
    $course = required_param('course', PARAM_INT);
    $type   = 'news';
    $timenow = time();
   $forum = get_record('forum', 'course',$course, 'type', 'news');
    $post = new object();
    $post->discussion  = 0;
    $post->parent      = 0;
    $post->userid      = $USER->id;
    $post->created     = $timenow;
    $post->modified    = $timenow;
    $post->mailed      = 0;
    $post->subject     = $newstitle;
    $post->message     = $newsbody;
    $post->attachment  = "";
    $post->forum       = $forum->id;
    $post->course      = $forum->course;
    $post->format      = 1;
    $post->mailnow     = 0;

   $post->id = insert_record("forum_posts", $post);
    $discussion->firstpost      = $post->id;
```

```
    $discussion->timemodified   = $timenow;
    $discussion->usermodified   = $post->userid;
    $discussion->userid         = $USER->id;
    $discussion->course         = $post->course;
    $discussion->forum          = $post->forum;
    $discussion->name           = $post->subject;
    $post->discussion = insert_record("forum_discussions",
$discussion);
    set_field("forum_posts", "discussion", $post->discussion, "id",
$post->id);
}
function def_note_dsl($sendto, $messagebody){
    global $USER, $CFG, $COURSE;
    $timenow = time();
if ($sendto='*'){
  $course    = required_param('course', PARAM_INT);
  $context   = get_record('context', 'contextlevel', 50, 'instanceid',
$course);
  $usersid   = get_records ('role_assignments', 'contextid', $context-
>id);
  foreach ($usersid as $users){
  $emails    = get_record ('user', 'id', $users->userid);
    $message = new object();
    $message->useridfrom    = $USER->id;
    $message->useridto      = $emails->id;
    $message->message       = $messagebody;
    $message->timecreated   = $timenow;
    $message->messagetype   = 'direct';
   insert_record("message", $message);
  }
 }
else if ($userto){
    $userto = get_record('user', 'email',$sendto);
    $message = new object();
    $message->useridfrom    = $USER->id;
    $message->useridto      = $userto->id;
    $message->message       = $messagebody;
    $message->timecreated   = $timenow;
    $message->messagetype   = 'direct';
  insert_record("message", $message);
   }
   else{
   }
}
```

Código Fuente 5 Fichero lib.php para Moodle.

3.6.3 Creación de módulos en Claroline

Al igual que en la sección anterior primero se explicara como es el esquema que utiliza Claroline para la creación de sus módulos, para ellos nos hemos basado en la información que se proporciona en su web para desarrolladores (Consorcio Claroline, 2008). Claroline almacena los modules propios en Claroline/Claroline, y los módulos

adicionales que son desarrollados como el nuestro en Claroline/module, cada uno en un directorio, la estructura general de archivos y directorios debe ser así:

- entry.php: Es el fichero en donde está el punto de entrada del módulo, en otras palabras lo que este allí es lo primero que se ejecutara cuando se invoque al módulo.
- manifest.xml: Este fichero posee la descripción del módulo.
- lib/: Directorio para almacenar las librerías que se utilizaran.
    - plantilladsl.lib.php: este fichero contiene las librerías que se utilizaran en el módulo.

3.6.3.1 El fichero entry.php

Como ya se ha comentado anteriormente éste es uno de los ficheros obligatorios de todo módulo y mediante el cual se configurará cada instancia del módulo que se añade a un curso. Aquí será en donde se llamaran a las funciones que están en lib/platilladsl.lib.php y en otras rutas, siempre es necesario ponerle un identificar al módulo para ello se utiliza la variable $tlabelReq = 'nombre del Módulo'; y es allí donde se pone toda la lógica de la aplicación. En nuestro caso este será creado automáticamente por la plantilla MOFScript acorde al modelo que se cree. Un ejemplo de cómo quedaría este fichero se puede ver a continuación:

```
<?php
$tlabelReq = 'UOPDSL';
require dirname(__FILE__) .
'/../../Claroline/inc/claro_init_global.inc.php';
if ( ! claro_is_in_a_course()  || ! claro_is_course_allowed() )
claro_disp_auth_form(true);
$context = claro_get_current_context(CLARO_CONTEXT_COURSE);
require_once dirname(__FILE__) . '/lib/plantilladsl.lib.php';
include(get_path('incRepositorySys') . '/claro_init_header.inc.php');
echo '<table border="0" cellspacing="10" cellpadding="10"
width="100%">' . "
"
.    '<tr>' . "
"
.    '<td valign="top" style="border-right: gray solid 1px;"
width="220">' . "
"
.    claro_html_menu_vertical_br($toolLinkList,
array('id'=>'commonToolList'))
.    '<br />'
;
                  $forum_name= 'Forum 1';
                        $forum_desc= 'Welcom to Foro 1';
                        $insert_id = create_forum($forum_name,
$forum_desc);
```

```
                    $message = 'Welcome to Chat 1';
                          add_chat_line($message);
                    $wikiName = 'Wiki 1';
                          $wikiDescription = 'Welcome to Wiki 1';
                          create_wiki($wikiName, $wikiDescription);
                    $title= 'Anuncio 1';
                          $content= 'Anuncio 1';
                          $insert_id =
announcement_add_item($title,$content);
                    $news_name ='News 1';
                          $news_body = 'Welcon to News 1';
                          create_news($news_name, $news_body);
                    $subject    ='Message from Claroline platform';
                          $message    ='Envio de nota 1';
                          $to                 ='*';
                          $toName             ='*';
                          $from       ='pruebasdsl@yahoo.es';
                          $fromName   ='Claroline platform';
                          create_note_all($subject, $message, $to,
$toName, $from, $fromName);
echo'<hr />Loaded Modules in Claroline';
?>
```

Código Fuente 6 Fichero entry.php para Claroline.

3.6.3.2 El fichero manisfest.xml

Este fichero contiene el manifiesto de nuestro módulo, allí se definen los datos necesarios para la instalación y ejecución del módulo, se debe proporcionar información sobre el módulo como label, nombre, tipo, descripción, versión y del autor como correo, email, o pagina web. Para nuestro caso este fichero quedaría así:

```
<module>
  <label>UOPDSL</label>
  <name>Plantilla DSL para Claroline</name>
  <type>tool</type>
  <description>Plantilla DSL para Claroline</description>
  <version>0.1</version>
  <author>
    <name>Carlos Montenegro</name>
    <email>cemontenegrom@udistrital.edu.co</email>
    <web>http://www.udistrital.edu.co</web>
  </author>
</module>
```

Código Fuente 7 Fichero manisfest.xml para Claroline.

3.6.3.3 El Fichero plantilladsl.lib.php

En este fichero tenemos las funciones básicas del módulo como se comentó anteriormente, y las clases que se crean oportunas, así como las constantes del

módulo. Lo más importante que podemos aprender de esta librería es a acceder a la base de datos mediante los mecanismos que proporciona la plataforma.

- claro_get_course_db_name_glued ($ cid = NULL)): Regresa el nombre que tiene el curso actual.
- claro_sql_get_course_tbl ($ dbNameGlued = null): Dado el parámetro retornado por claro_get_course_db_name_glued, y regresa la lista de las tablas asociadas al curso actual.
- claro_sql_query ($ *dbHandler* = '#'): Es un contenedor de querys a la base de datos, recibe como parámetro de enterada el string con el query.
- claro_sql_query_insert_id ($ *sqlQuery*, $ *dbHandler* = '#'): recibe como parámetro de entrada el query a la base de datos que está almacenado en claro_sql_query( ), inserta el valor en la base de datos y regresa el id de la fila en la cual se insertaron los datos.

Dado que tanto para obtener e insertar datos en las tablas se utilizan objetos, en este fichero se crean los objetos para poder usarlos cómodamente a lo largo de todo el módulo.

Este es el otro fichero que generamos con nuestras plantillas MOFScript a partir del módulo que se construya un ejemplo de este fichero se muestra a continuación:

```
<?php // $Id: announcement.lib.php 9707 2007-12-12 13:46:31Z mlaurent
$
if ( count( get_included_files() ) == 1 ) die( '---' );
function create_forum($forum_name, $forum_desc, $forum_post_allowed=2,
$cat_id = 2, $group_id = null, $course_id=NULL)
{
    $tbl_cdb_names =
claro_sql_get_course_tbl(claro_get_course_db_name_glued($course_id));
    $tbl_forum_forums = $tbl_cdb_names['bb_forums'];
    $sql = "SELECT MAX(`forum_order`)
            FROM `" . $tbl_forum_forums . "`
            WHERE cat_id = " . (int) $cat_id;
    $result = claro_sql_query($sql);
    list($orderMax) = mysql_fetch_row($result);
    $order = $orderMax + 1;
    $sql = "INSERT INTO `" . $tbl_forum_forums . "`
            SET forum_name      = '" . addslashes($forum_name) . "',
                group_id        = " . (is_null($group_id) ? "NULL" :
(int) $group_id) . ",
                forum_desc      = '" . addslashes($forum_desc) . "',
                forum_access    = " . ($forum_post_allowed ? 2 : 0) .
",
                forum_moderator = 1,
                cat_id          = " . (int) $cat_id . ",
                forum_type      = 0,
                forum_order     = " . (int) $order ;
```

```
    return claro_sql_query_insert_id($sql);
}
function announcement_add_item($title='',$content='',
$visibility='SHOW', $time=NULL, $course_id=NULL)
{
    $tbl=
claro_sql_get_course_tbl(claro_get_course_db_name_glued($course_id));
    if(is_null($time))
    {
        $sqlTime = " temps = NOW(), ";
    }
    else
    {
        $sqlTime = " temps = from_unixtime('". (int)$time ."'), ";
    }
    $sql = "SELECT (MAX(ordre) + 1) AS nextRank
            FROM  `" . $tbl['announcement'] . "`";
    $nextRank = claro_sql_query_get_single_value($sql);
    $sql = "INSERT INTO `" . $tbl['announcement'] . "`
            SET title ='" . claro_sql_escape(trim($title)) . "',
                contenu = '" . claro_sql_escape(trim($content)) . "',
                visibility = '" . ($visibility=='HIDE'?'HIDE':'SHOW')
. "',
             ". $sqlTime ."
            ordre ='" . (int) $nextRank . "'";
    return claro_sql_query_insert_id($sql);
}
function add_chat_line($message)
{
$coursePath  = get_path('coursesRepositorySys') .
claro_get_course_path();
$courseId    = claro_get_current_course_id();
$groupId     = claro_get_current_group_id();
$_user       = claro_get_current_user_data();
$_course     = claro_get_current_course_data();
$_group      = claro_get_current_group_data();
$is_allowedToManage = claro_is_course_manager();
$is_allowedToStore  = claro_is_course_manager();
$is_allowedToReset  = claro_is_course_manager();
    $nick = $_user['firstName'] . ' ' . $_user['lastName'] ;
$curChatRep = $coursePath.'/chat/';
        $activeChatFile =
$curChatRep.$courseId.''.$groupId.'.chat.html';
$dateNow =
claro_html_localised_date(get_locale('dateTimeFormatLong'));
$timeNow = claro_html_localised_date('[%d/%m/%y %H:%M]');
          $fchat = fopen($activeChatFile,'a');
          $chatLine =
ereg_replace("(http://)(([[:punct:]]|[[:alnum:]])*)","<a href=\"\\0\"
target=\"_blank\">\\2</a>",$message);
          fwrite($fchat, '<small>' . $timeNow . ' &lt;<b>' . $nick .
'</b>&gt; ' . $chatLine . '</small><br />' . "\n");
          fclose($fchat);
}
function create_wiki($wikiName, $wikiDescription, $gid = false )
```

```
{
      require_once
"../../Claroline/wiki/lib/class.clarodbconnection.php";
    require_once
"../../Claroline/wiki/lib/class.wikiaccesscontrol.php";
    require_once  "../../Claroline/wiki/lib/class.wikistore.php";
    require_once  "../../Claroline/wiki/lib/class.wikipage.php";
    require_once  "../../Claroline/wiki/lib/class.wiki.php";
    require_once  "../../Claroline/wiki/lib/lib.wikisql.php";
        $creatorId = claro_get_current_user_id();
        $tblList = claro_sql_get_course_tbl();
        $config = array();
        $config["tbl_wiki_properties"] = $tblList[ "wiki_properties"
];
        $config["tbl_wiki_pages"] = $tblList[ "wiki_pages" ];
        $config["tbl_wiki_pages_content"] = $tblList[
"wiki_pages_content" ];
        $config["tbl_wiki_acls"] = $tblList[ "wiki_acls" ];
        $con = new ClarolineDatabaseConnection();
        $acl = array();
        if ( $gid )
        {
            $acl = WikiAccessControl::defaultGroupWikiACL();
        }
        else
        {
            $acl = WikiAccessControl::defaultCourseWikiACL();
        }
        $wiki = new Wiki( $con, $config );
        $wiki->setTitle( $wikiName );
        $wiki->setDescription( $wikiDescription );
        $wiki->setACL( $acl );
        $wiki->setGroupId( $gid );
        $wikiId = $wiki->save();
        $wikiTitle = $wiki->getTitle();
        $mainPageContent = sprintf( "This is the main page of the Wiki
%s. Click on edit to modify the content.", $wikiTitle );
        $wikiPage = new WikiPage( $con, $config, $wikiId );
        $wikiPage->create( $creatorId
            , '__MainPage__'
            , $mainPageContent
            , date( "Y-m-d H:i:s" )
            , true );
        echo $con->getError();
}
function create_news($news_name, $news_body)
{
    $tbl_cdb_names =
claro_sql_get_course_tbl(claro_get_course_db_name_glued($course_id));
    $tbl_tool_intro = $tbl_cdb_names['tool_intro'];
       // add new News in DB
    $sql = "INSERT INTO `" . $tbl_tool_intro . "`
            SET title          = '".$news_name."',
                content        = '".$news_body."',
                rank             = 1,
```

```
                tool_id          = 0";
    return claro_sql_query_insert_id($sql);
}
/**
 * Send e-mail using Main settings
 */
function create_note($subject, $message, $to, $toName, $from,
$fromName)
{
      require_once  "../../Claroline/inc/lib/sendmail.lib.php";
    $mail = new ClaroPHPMailer();
    if (empty($from))
    {
        $from = get_conf('administrator_email');
        if (empty($fromName))
        {
            $fromName = get_conf('administrator_name');
        }
    }
    $mail->Subject  = $subject;
    $mail->Body     = $message;
    $mail->From     = $from;
    $mail->FromName = $fromName;
    $mail->Sender   = $from;
    $mail->AddAddress($to,$toName);
    if ( $mail->Send() )
    {
        return true;
    }
    else
    {
        return claro_failure::set_failure($mail->getError());
    }
}
function create_note_all($subject, $message, $to, $toName, $from,
$fromName)
{
$courseId    = claro_get_current_course_id();
      $sql= "select email
                  from cl_user
                  inner join cl_cours_user
                  on cl_user.user_id = cl_cours_user.user_id
                  where cl_cours_user.code_cours = '".$courseId."'";
      $listMail = claro_sql_query_fetch_all($sql);
      foreach ($listMail as $namemail){
            foreach ($namemail as $mail){
                  create_note ($subject, $message, $mail, $mail,
$from, $fromName);
            }
      }
}
```

Código Fuente 8 Fichero plantilladsl.lib.php para Claroline.

3.6.4 Creación de módulos en Atutor

Al igual que en la sección anterior primero se explicara como es el esquema que utiliza ATutor para la creación de sus módulos, para ellos nos hemos basado en la información que se proporciona en su web para desarrolladores (Atutor, 2010). ATutor almacena los modules propios /mods que a su vez se encuentran otras dos carpetas /_core y _/standard, los módulos adicionales son almacenados en /mods, cada uno en un directorio, la estructura general de archivos y directorios debe ser así:

- *module.php:* Es el fichero en donde está el punto de entrada del módulo, en otras palabras lo que este allí es lo primero que se ejecutara cuando se invoque al módulo. Este fichero es requerido.
- *module.xml:* Este fichero posee la descripción del módulo, este fichero es requerido.
- *module_install.php:* Este fichero es requerido cuando se instala el módulo. Este fichero es requerido.
- *module_uninstall.php:* Este fichero es necesario para remover el módulo del sistema. Este fichero es requerido.
- *module.sql:* Este fichero es para agregar tablas, modificar datos. También utilizado para insertar el nombre del módulo a la tabla “language_text”. Este fichero es opcional.
- *lib/:* Directorio para almacenar las librerías que se utilizaran.
    - dsllib.lib.php: este fichero contiene las librerías que se utilizaran en el módulo.

3.6.4.1 El fichero module.php

Como ya se ha comentado anteriormente éste es uno de los ficheros obligatorios de todo módulo y mediante el cual se configurará cada instancia del módulo que se añade a un curso. Es usado típicamente para asignar permisos, se identifica si el archivo es un módulo o no, agregar iconos, tabs de navegación, menús, entre otros. Un ejemplo de cómo quedaría este fichero se puede ver a continuación:

```
<?php define('AT_INCLUDE_PATH', '../../../include/');
//require(AT_INCLUDE_PATH.'lib/mysql_connect.inc.php'); if
(!defined('AT_INCLUDE_PATH')) { exit; } if (!isset($this) ||
(isset($this) && (strtolower(get_class($this)) != 'module'))) {
exit(__FILE__ . ' is not a Module'); }
define('AT_PRIV_PLANTILLADSL',       $this->getPrivilege());
define('AT_ADMIN_PRIV_PLANTILLADSL', $this->getAdminPrivilege());
$this->_pages['mods/plantilladsl/index.php']['title_var'] =
'plantilla_dsl';
$this->_pages['mods/plantilladsl/index.php']['parent']   =
'tools/index.php';
?>
```

Código Fuente 9 Fichero module.php para ATutor.

3.6.4.2 El fichero módulo.xml

Este fichero contiene el manifiesto de nuestro módulo, allí se definen los datos necesarios para la instalación y ejecución del módulo, se debe proporcionar información sobre el módulo como label, nombre, tipo, descripción, versión y del autor como correo, email, o pagina web. Para nuestro caso este fichero quedaría así:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<module version="0.1">
    <name lang="en">platilladsl</name>
    <description lang="en">descripcion de plantilla dsl</description>
    <maintainers>
        <maintainer>
            <name>ATutor Team</name>
            <email>info@ATutor.ca</email>
        </maintainer>
    </maintainers>
    <url>http://ATutor.ca</url>
    <license>GPL</license>
      <release>
        <version>0.1</version>
            <privileges>
                  <instructor_privilege>create</instructor_privilege>
                  <admin_privilege></admin_privilege>
            </privileges>
        <date>2011-02-24</date>
        <state>stable</state>
        <notes>This is a standard module.</notes>
    </release>
</module>
```

Código Fuente 10 Fichero módulo.xml para ATutor.

3.6.4.3 El fichero module_install.php

Este fichero es usualmente usado para correr los archivos necesarios para la instalación del módulo, tales como el fichero sql o la instalación del lenguaje del módulo. Para nuestro caso este fichero quedaría así:

```
<?php
if (!defined('AT_INCLUDE_PATH')) { exit; }
$_course_privilege = TRUE; // possible values: FALSE | AT_PRIV_ADMIN
| TRUE
$_admin_privilege  = TRUE; // possible values: FALSE | TRUE
if (!$msg->containsErrors() && file_exists(dirname(__FILE__) .
'/module.sql')) {
      // deal with the SQL file:
      require(AT_INCLUDE_PATH . 'classes/sqlutility.class.php');
      $sqlUtility =& new SqlUtility();
      /*
       * the SQL file could be stored anywhere, and named anything,
"module.sql" is simply
       * a convention we're using.
```

```
        */
      $sqlUtility->queryFromFile(dirname(__FILE__) . '/module.sql',
TABLE_PREFIX);
}
?>
```

Código Fuente 11 Fichero module_install.php para ATutor.

3.6.4.4 El fichero module_uninstall.php

Este fichero se ejecuta cuando se desea borrar o desinstalar un módulo de la plataforma. Para nuestro caso este fichero quedaría así:

```
<?php
if (!defined('AT_INCLUDE_PATH')) { exit; }
if (!$msg->containsErrors() && file_exists(dirname(__FILE__) .
'/module.sql')) {
      // deal with the SQL file:
      require(AT_INCLUDE_PATH . 'classes/sqlutility.class.php');
      $sqlUtility = new SqlUtility();
      /*
       * the SQL file could be stored anywhere, and named anything,
"module.sql" is simply
       * a convention we're using.
       */
      $sqlUtility->revertQueryFromFile(dirname(__FILE__) .
'/module.sql', TABLE_PREFIX);
}
?>
```

Código Fuente 12 Fichero module_unistall.php para ATutor.

3.6.4.5 El fichero module.sql

Este fichero crea tablas para el módulo, inserta lenguajes en la tabla “language_text”. Este fichero puede contener cualquier sentencia sql que se desea afecte la base de datos de la plataforma para el funcionamiento del módulo en instalación. Para nuestro caso este fichero quedaría así:

```
# sql file for plantilladsl module
INSERT INTO `language_text` VALUES ('en',
'_module','plantilladsl','Plantilladsl',NOW(),'');
```

Código Fuente 13 Fichero module.sql para ATutor.

3.6.4.6 El fichero dsllib.lib.php

En este fichero tenemos las funciones básicas del módulo como se comentó anteriormente. Desde este archivo se puede tener acceso a la sesión, al curso, al miembro que se logea de forma transparente ya que las variables son declaradas

globales en la plataforma ATutor. Es por ello que hacemos uso de ellas para generar los componentes del módulo solicitados de acuerdo a quién y qué solicite el instructor.

Este fichero es generado con nuestras plantillas MOFScript a partir del módulo que se construya, un ejemplo de este fichero se muestra a continuación:

```
<?php if (!defined('AT_INCLUDE_PATH')) { exit; }
require(AT_INCLUDE_PATH.'../mods/plantilladsl/lib/class.phpmailer.ph
p');
require(AT_INCLUDE_PATH.'../mods/plantilladsl/lib/class.smtp.php');
//Funcion para crear un new. function add_new_dsl($describenew){
      global $db;       $sql = "INSERT INTO
".TABLE_PREFIX."assignments VALUES (NULL, $_SESSION[course_id],
'$describenew', '0', NOW(), '0000-00-00 00:00:00', '0')";
      mysql_query($sql,$db);  return; } ///Funcion para crear un
anuncio. function add_announcement_dsl($titulo,$Announdescripcion){
      global $db;       $sql = "INSERT INTO ".TABLE_PREFIX."news
VALUES (NULL,$_SESSION[course_id], $_SESSION[member_id], NOW(),1,
'$titulo', '$Announdescripcion')";       mysql_query($sql, $db); }
//Funcion para crear un foro. function
add_forum_dsl($foronombre,$forointroduccion){  global $db;
      $sql  = "INSERT INTO ".TABLE_PREFIX."forums VALUES
(NULL,'$foronombre','$forointroduccion', 0, 0, NOW(),0)";
      mysql_query($sql,$db);  $sql  = "INSERT INTO
".TABLE_PREFIX."forums_courses VALUES (LAST_INSERT_ID(),
$_SESSION[course_id])";       mysql_query($sql,$db);  return; }
//Funcion para crear una note. function
add_note_dsl($sendto,$messagebody){       /*global $db;       $sql
      = "SELECT email FROM ".TABLE_PREFIX."course_enrollment C INNER
JOIN ".TABLE_PREFIX."members M USING (member_id) WHERE
C.course_id=1";   $sendto = mysql_query($sql,$db);*/  $mail = new
PHPMailer ();     $mail -> From = "ingcarlosmontenegro@gmail.com";
      $mail -> FromName = "DSL";    $mail -> AddAddress ("$sendto");
      $mail -> Subject = "PruebaDSL";     $mail -> Body =
"<h3>$messagebody</h3>";      $mail -> IsHTML (true);       $mail-
>IsSMTP();  $mail->Host = 'ssl://smtp.gmail.com';     $mail->Port =
465;  $mail->SMTPAuth = true;       $mail->Username =
'ingcarlosmontenegro@gmail.com';    $mail->Password = 'montenegro';
      if(!$mail->Send()) {          echo 'Error: ' . $mail-
>ErrorInfo;       }     else {            echo 'Mail enviado!';   }
} //Funcion para enviar Note. function
add_new_notemain($sendto,$bodynote){      global $db;
      if($sendto=='*'){          $sql     = "SELECT email FROM
".TABLE_PREFIX."course_enrollment C INNER JOIN
".TABLE_PREFIX."members M USING (member_id) WHERE
C.course_id=$_SESSION[course_id]";           $sendto =
mysql_query($sql,$db);                  $row    =
mysql_fetch_array($sendto);             foreach($row as $usernote){
            add_note_dsl($row['email'],$bodynote);      }  }else{
          $sql = 'SELECT member_id FROM '.TABLE_PREFIX.'members
WHERE email="'.$sendto.'"';         $enviara = mysql_query($sql,$db);
           if ($enviara){           $row    =
mysql_fetch_array($enviara);                    $enviar =
$row['member_id'];                      $sql = "INSERT INTO
".TABLE_PREFIX."messages VALUES (NULL, $_SESSION[course_id],
```

```
$_SESSION[member_id], $enviar, NOW(), 1, 0, 'DSLnote',
'$bodynote')";          mysql_query($sql,$db);              // sent
message box:            $sql = "INSERT INTO
".TABLE_PREFIX."messages_sent VALUES (NULL, $_SESSION[course_id],
$_SESSION[member_id],$enviar, NOW(), 'DSLnote', '$bodynote')";
      mysql_query($sql,$db);              }                 else{
                        echo "Usuario No existe";           }     } }

?>
```

Código Fuente 14 Fichero dsllib.lib.php para ATutor.

Con esto hemos explicado cómo crear módulos en cada una de las plataformas a utilizar, en el siguiente apartado se mostrara como generar estos módulos a partir del modelo obtenido con el DSL.

3.6.5 Transformación de modelos a módulos para Moodle, Claroline y ATutor desde un modelo en KiwiDSM v1.0

Para transformar el modelo que se obtiene con KiwiDSM v1.0 hay que tener en cuenta que este modelo estará basado en el metamodelo LMS de la figura 45, MOFScript tiene su propia sintaxis que pude ser consultada en (Oldevik, 2009). Para el funcionamiento de MOFScript se puede consultar el anexo II. Aquí nos centraremos en explicar únicamente cómo funcionan las plantillas MOFScript para Moodle, Claroline y ATutor.

Lo primero que se debe hacer es definir el modelo de entrada a las plantillas MOFScript, para ello se debe declarar la transformación con **"texttransformation"**, darle un nombre a la trasformación y enviarle como parámetro de entrada el nombre del metamodelo con extensión .ecore, para nuestro caso "modellms", también se debe dar un nombre al modelo enviado, para nuestro caso se llamo "mlms". Esto se puede visualizar el siguiente fragmento de código.

```
texttransformation modellmsTransformationMoodle (in mlms:"modellms") {
```

Código Fuente 15 Declaración de una transformación en MOFScript para Moodle.

```
texttransformation modellmsTransformationClaroline (in
mlms:"modellms") {
```

Código Fuente 16 Declaración de una transformación en MOFScript para Claroline.

```
texttransformation modellmsTransformationATutor (in mlms:"modellms") {
```

Código Fuente 17 Declaración de una transformación en MOFScript para ATutor.

Ahora dentro del bloque que conforma la nueva transformación se declaran las variables a utilizar, en nuestro caso estas variables servirán para contar cuantas

trasformación se realizaron correctamente, cuantas no se realizaron por falta de alguna conexión y cuantas no se realizaron por falta de información en los campos, esto por cada módulo modelado con la herramienta DSL, el código correspondiente se muestra a continuación:

```
var numForumY:integer = 0;

var numForumN:integer = 0;

var numForumSN:integer = 0;

var numAnnouncementY:integer = 0;

var numAnnouncementN:integer = 0;

var numAnnouncementSN:integer = 0;

var numChatY:integer = 0;

var numChatN:integer = 0;

var numChatSN:integer = 0;

var numWikiY:integer = 0;

var numWikiN:integer = 0;

var numWikiSN:integer = 0;

var numNewsY:integer = 0;

var numNewsSN:integer = 0;

var numNewsN:integer = 0;

var numNoteY:integer = 0;

var numNoteN:integer = 0;

var numNoteSN:integer = 0;

var quote:String = "'"
```

Código Fuente 18 Declaración de variables en MOFScript.

La declaración de la variable quote “`var quote:String = "'"`”, se realizo ya que las cadenas caracteres en MOFScript y PHP se representan entre “`'`” con lo cual se genera una confusión que causa errores en el despliegue de la solución final, para corregir este error aquellas variables que en MOFScript sean “`quote`” corresponderán en PHP al carácter “`'`”.

La funciona principal o punto de partida para las transformaciones en MOFScript están contendidas en la función "**main ()**" que se declara como se muestra a continuación:

```
mlms.LMSModel::main () {
```

Código Fuente 19 Función principal "main" en MOFScript.

Para crear un nuevo fichero se utiliza la instrucción "**file**" como se ve a continuación:

```
    file ('mod_form.php');
```

Código Fuente 20 Creación de fichero mod_form.php en MOFScript para Moodle.

```
    file ('entry.php');
```

Código Fuente 21 Creación de fichero mod_form.php en MOFScript para Claroline.

O también se puede crear un fichero con nombre como se muestra a continuación y dentro de una carpeta:

```
file f1('/plantilladsl/index.php');
```

Código Fuente 22 Creación de fichero /plantilladsl/mod_form.php en MOFScript para ATutor.

Y para escribir en ese fichero las cadenas deben ir entre el carácter "'". Para la declaración de funciones, primero se debe anteponer la palabra "**module**" seguida de el operados "**::**" y luego el código correspondiente, a continuación se muestra la función encabezado para las plantillas de transformación de Moodle.

```
      module::encabezado()

{'<?php

require_once($CFG-
>dirroot.'+quote+'/course/Moodleform_mod.php'+quote+');

require_once($CFG->dirroot.'+quote+'/course/lib.php'+quote+');

require_once('+quote+'../config.php'+quote+');

class mod_plantilladsl_mod_form extends Moodleform_mod {

    function definition() {'

      }
```

Código Fuente 23 Función encabezado en MOFScript para Moodle.

Ahora se muestra la función encabezado para las plantillas de transformación de Claroline.

```
      module::encabezado()

      {

'<?php

$tlabelReq = '+quote+'UOPDSL'+quote+';

require dirname(__FILE__) .
'+quote+'/../../Claroline/inc/claro_init_global.inc.php'+quote+';

if ( ! claro_is_in_a_course()  || ! claro_is_course_allowed() )
claro_disp_auth_form(true);

$context = claro_get_current_context(CLARO_CONTEXT_COURSE);



require_once dirname(__FILE__) .
'+quote+'/lib/plantilladsl.lib.php'+quote+';

include(get_path('+quote+'incRepositorySys'+quote+') .
'+quote+'/claro_init_header.inc.php'+quote+');

echo '+quote+'<table border="0" cellspacing="10" cellpadding="10"
width="100%">'+quote+' . "\n"

.    '+quote+'<tr>'+quote+' . "\n"

.    '+quote+'<td valign="top" style="border-right: gray solid 1px;"
width="220">'+quote+' . "\n"

.    claro_html_menu_vertical_br($toolLinkList,
array('+quote+'id'+quote+'=>'+quote+'commonToolList'+quote+'))

.    '+quote+'<br />'+quote+'

;'

}
```

Código Fuente 24 Función encabezado en MOFScript para Claroline.

Finalmente se muestran la función encabezado para las plantillas de transformación de ATutor.

```
            module::encabezado()
```

```
            {

              '<?php


                  define('+quote+'AT_INCLUDE_PATH'+quote+',
'+quote+'../../include/'+quote+');

                  require
(AT_INCLUDE_PATH.'+quote+'vitals.inc.php'+quote+');


      require(AT_INCLUDE_PATH.'+quote+'lib/mysql_connect.inc.php'+quot
e+');



                  authenticate(AT_PRIV_PLANTILLADSL);


      $_pages['+quote+'mods/plantilladsl/index.php'+quote+']['+quote+'
title'+quote+']  = '+quote+'plantilla_dsl'+quote+';



      require(AT_INCLUDE_PATH.'+quote+'../mods/plantilladsl/lib/dsllib
.inc.php'+quote+');

                '

            }
```

Código Fuente 25 Función encabezado en MOFScript para ATutor

Para invocar a las funciones sencillamente se llaman en donde sean necesarias, como en el metamodelo todas las EClass están contenidas en una EClass llamada LMSModel esta relación se representa con una conexión, a continuación se valida si en el modelo existe un módulo (EClass) Communications que se relaciona con LMSModel a través de la relación LMSModel_Has_Communications, esto se muestra a continuación:

```
      encabezado();

      if (self.LMSModel_Has_Communications != null){
```

Código Fuente 26 Llamado a la función encabezado y condición de validación; si en el modelo existe alguna EClass Communications.

A continuación se muestra como contar cuantos módulos tiene un modelo, independiente si están conectados o no.

```
            numForumN = self.LMSModel_Has_Forum_Com.size();

            numAnnouncementN =
self.LMSModel_Has_Announcement_Com.size();

            numChatN = self.LMSModel_Has_Chat_Com.size();

            numWikiN = self.LMSModel_Has_Wiki_Com.size();

            numNewsN = self.LMSModel_Has_News_Com.size();

            numNoteN = self.LMSModel_Has_Note_Com.size();
```

Código Fuente 27 Contador de módulos dentro de un modelo.

Luego se recorren todos los módulos Forum que están dentro del modelo y conectados al módulo Communications, y se valida si en estos módulos los campos están rellenados, si no están rellenados incrementa en 1 un contador de los módulos Forum que están conectados pero no tienen información, si están rellenados se imprime una nueva línea en el fichero acompañado de dos espacios de tabulador, la invocación a la función correspondiente en Moodle, Claroline o Atutor con parámetros de entrada los nombres ingresados en el modelo y finalmente se incrementa un contador de los módulos creados correctamente. A continuación se muestra el código correspondiente para Moodle.

```
      self.LMSModel_Has_Communications.Communications_Has_Forum-
>forEach(forum:mlms.Forum)

            {

                        if (forum.forumName = null or forum.message =
null){

                              numForumSN = numForumSN+1;

                        }

                        else{

                        newline(1);

                        tab(2);

                        'def_forum_dsl
('+quote+forum.forumName+quote+','+quote+forum.message+quote+');';

                              numForumY = numForumY+1;
```

```
                        }

                }
```

Código Fuente 28 Recorrido para los módulos Forum en MOScript para Moodle.

Ahora se mostrara el recorrido para los módulos en MOFScript para Claroline.

```
        self.LMSModel_Has_Communications.Communications_Has_Forum-
>forEach(forum:mlms.Forum)

                {

                        if (forum.forumName = null or forum.message =
null){

                                numForumSN = numForumSN+1;

                        }

                        else{

                        newline(2);

                        tab(3);

                                '$forum_name=
'+quote+forum.forumName+quote+';

                                $forum_desc=
'+quote+forum.message+quote+';

                                $insert_id = create_forum($forum_name,
$forum_desc);';

                                numForumY = numForumY+1;

                        }



                }
```

Código Fuente 29 Recorrido para los módulos Forum en MOScript para Claroline.

Finalmente se muestra el recorrido para los módulos en MOFScript para ATutor.

```
                Self.LMSModel_Has_Communications.Communications_Has_Forum-
>forEach(forum:mlms.Forum)

                        {
```

```
                        //Cuenta los módulos que tienen conexion, pero
algun campo esta sin llenar

                                if (forum.forumName = null or
forum.message = null){

                                        numForumSN = numForumSN+1;

                                }

                                else{

                                newline(2);

                                tab(3);

                                '$forum_name=
'+quote+forum.forumName+quote+';

                                        $forum_desc=
'+quote+forum.message+quote+';


      add_forum_dsl($forum_name,$forum_desc);'


                                        //Contador de módulos creados

                                        numForumY = numForumY+1;

                                }


                    }
```

Código Fuente 30 Recorrido para los módulos Forum en MOScript para ATutor.

Después se recorren todos los módulos Chat que están dentro del modelo y conectados al módulo Communications, y se valida si en estos módulos los campos están rellenados, si no están rellenados incrementa en 1 un contador de los módulos Chat que están conectados pero no tienen información, si están rellenados se imprime una nueva línea en el fichero acompañado de dos espacios de tabulador, la invocación a la función correspondiente en Moodle, Claroline o Atutor con parámetros de entrada los nombres ingresados en el modelo y finalmente se incrementa un contador de los módulos creados correctamente. A continuación se muestran el código correspondiente para Moodle.

```
        self.LMSModel_Has_Communications.Communications_Has_Chat-
```

```
>forEach(chat:mlms.Chat)

            {

                        if (chat.roomName = null or chat.sendText =
null){

                              numChatSN = numChatSN+1;

                        }

                        else{

                        newline(1);

                        tab(2);

                        'def_chat_dsl
('+quote+chat.roomName+quote+','+quote+chat.sendText+quote+');';

                              numChatY = numChatY+1;

                        }

            }
```

Código Fuente 31 Recorrido para los módulos Chat en MOScript para Moodle.

El siguiente fragmento de código muestra el recorrido para los módulos Chat en MOFScript para Claroline.

```
      self.LMSModel_Has_Communications.Communications_Has_Chat-
>forEach(chat:mlms.Chat)

            {

                        if (chat.roomName = null or chat.sendText =
null){

                              numChatSN = numChatSN+1;

                        }

                        else{

                          newline(2);

                          tab(3);

                          '$message = '+quote+chat.sendText+quote+';

                                    add_chat_line($message);';
```

```
                                        numChatY = numChatY+1;

                                }

                        }
```

Código Fuente 32 Recorrido para los módulos Chat en MOScript para Claroline.

Por último el siguiente fragmento de código muestra el recorrido para los módulos Chat en MOFScript para ATutor.

```
                self.LMSModel_Has_Communications.Communications_Has_Chat-
>forEach(chat:mlms.Chat)

                        {

                                //Cuenta los módulos que tienen conexion, pero
algun campo esta sin llenar

                                        if (chat.roomName = null or chat.sendText
= null){

                                                numChatSN = numChatSN+1;

                                        }

                                        else{


                                          //Contador de módulos creados

                                                numChatY = numChatY+1;

                                                numChatmsn = numChatY+1;

                                                file
fchat('/dslchat/'+numChatmsn+'.message');


fchat.println(quote+chat.sendText+quote);

                                        }

                        }
```

Código Fuente 33 Recorrido para los módulos Chat en MOScript para ATutor.

A continuación se recorren todos los módulos Wiki que están dentro del modelo y conectados al módulo Communications, y se valida si en estos módulos los campos están rellenados, si no están rellenados incrementa en 1 un contador de los módulos

Wiki que están conectados pero no tienen información, si están rellenados se imprime una nueva línea en el fichero acompañado de dos espacios de tabulador, la invocación a la función correspondiente en Moodle, Claroline o Atutor con parámetros de entrada los nombres ingresados en el modelo y finalmente se incrementa un contador de los módulos creados correctamente. A continuación se muestra el código correspondiente para Moodle.

```
        self.LMSModel_Has_Communications.Communications_Has_Wiki-
>forEach(wiki:mlms.Wiki)

                {

                                if (wiki.wikiName = null or wiki.wikiBody =
null){

                                        numWikiSN = numWikiSN+1;

                                }

                                else{

                                newline(1);

                                tab(2);

                                'def_wiki_dsl
('+quote+wiki.wikiName+quote+','+quote+wiki.wikiBody+quote+');';

                                        numWikiY = numWikiY+1;

                                }

                }
```

Código Fuente 34 Recorrido para los módulos Wiki en MOScript para Moodle.

El siguiente fragmento de código muestra el recorrido para los módulos Wiki en MOFScript para Claroline.

```
        self.LMSModel_Has_Communications.Communications_Has_Wiki-
>forEach(wiki:mlms.Wiki)

                {

                                if (wiki.wikiName = null or wiki.wikiBody =
null){

                                        numWikiSN = numWikiSN+1;

                                }
```

```
                        else{
                        newline(2);
                        tab(3);
                        '$wikiName = '+quote+wiki.wikiName+quote+';
                              $wikiDescription =
'+quote+wiki.wikiBody+quote+';
                              create_wiki($wikiName,
$wikiDescription);';
                              numWikiY = numWikiY+1;
                        }
            }
```

Código Fuente 35 Recorrido para los módulos Wiki en MOScript para Claroline.

Por último el siguiente fragmento de código muestra el recorrido para los módulos Wiki en MOFScript para ATutor.

```
            self.LMSModel_Has_Communications.Communications_Has_Wiki-
>forEach(wiki:mlms.Wiki)
                  {
                    //Cuenta los módulos que tienen conexion, pero
algun campo esta sin llenar
                        if (wiki.wikiName = null or
wiki.wikiBody = null){
                              numWikiSN = numWikiSN+1;
                        }
                        else{
                        //Contador de módulos creados
                              numWikiY = numWikiY+1;
                              numWikitext = numWikiY+1;
                              file
fwiki('/dslwiki/'+'FrontPage.'+numWikitext);
```

```
                                        fwiki.println("id: FrontPage");

                                        fwiki.println("version: 3");

                                        fwiki.println("flags: 1");

                                        fwiki.println("created:
1298576630");

                                        fwiki.println("meta:
a:1:{s:10:"+quote+"user-agent"+quote+";s:90:"+quote+"Mozilla/5.0
(Windows; U; Windows NT 6.1; es-ES; rv:1.9.2.13) Gecko/20101203
Firefox/3.6.13"+quote+";}");

                                        fwiki.println("hits: 0");

                                        fwiki.println("refs:
\n\nWikiHowto\nCreatePages\nEditThisPage\nWikiMarkup\nSearchPages\nNew
estPages\nMostVisitedPages\nMostOftenChangedPages\nUpdatedPages\nForma
ttingOptions\nWordIndex\nImageGallery\nOrphanedPages\nTextUpload\nWant
edPages\nWikiDump\nWikiNews\nHitCounter\nFileUpload\nFileDownload\nEwi
kiLog\nPowerSearch\nScanDisk\nRecentChanges\n\n\n");

                                        fwiki.println("lastmodified:
1298585394");

                                        fwiki.println("author:
cachewww3.si.uniovi.es 156.35.14.9:56328");

                                        fwiki.println("");


      fwiki.println(quote+wiki.wikiName+quote);


      fwiki.println(quote+wiki.wikiBody+quote);

                                  }


                  }
```

Código Fuente 36 Recorrido para los módulos Wiki en MOFScript para ATutor.

A continuación se recorren todos los módulos Announcement que están dentro del modelo y conectados al módulo Communications, y se valida si en estos módulos los campos están rellenados, si no están rellenados incrementa en 1 un contador de los módulos Announcement que están conectados pero no tienen información, si están rellenados se imprime una nueva línea en el fichero acompañado de dos espacios de tabulador, la invocación a la función correspondiente en Moodle, Claroline o Atutor con

parámetros de entrada los nombres ingresados en el modelo y finalmente se incrementa un contador de los módulos creados correctamente. A continuación se muestra el código correspondiente para Moodle.

```
self.LMSModel_Has_Communications.Communications_Has_Announcement-
>forEach(anno:mlms.Announcement)

            {

                        if (anno.announcementText = null){

                              numAnnouncementSN = numAnnouncementSN+1;

                        }

                        else{

                        newline(1);

                        tab(2);

                        'def_Announcement_dsl
('+quote+anno.announcementText+quote+');';

                              numAnnouncementY = numAnnouncementY+1;

                        }

            }
```

Código Fuente 37 Recorrido para los módulos Announcement en MOFScript para Moodle.

El siguiente fragmento de código muestra el recorrido para los módulos Announcement en MOFScript para Claroline.

```
self.LMSModel_Has_Communications.Communications_Has_Announcement-
>forEach(anno:mlms.Announcement)

            {

                        if (anno.announcementText = null){

                              numAnnouncementSN = numAnnouncementSN+1;

                        }

                        else{

                        newline(2);

                        tab(3);
```

```
                        '$title=
'+quote+anno.announcementText+quote+';

                                $content=
'+quote+anno.announcementText+quote+';

                                $insert_id =
announcement_add_item($title,$content);';

                                numAnnouncementY = numAnnouncementY+1;

                        }

                }
```

Código Fuente 38 Recorrido para los módulos Announcement en MOFScript para Claroline.

Y finalmente se presenta el código que recorre los módulos Announcement en MOFScript para Atutor.

```
self.LMSModel_Has_Communications.Communications_Has_Announcement-
>forEach(anno:mlms.Announcement)

                {

                  //Cuenta los módulos que tienen conexion, pero
algun campo esta sin llenar

                                if (anno.announcementText = null){

                                        numAnnouncementSN =
numAnnouncementSN+1;

                                }

                                else{

                                newline(2);

                                tab(3);

                                '$title=
'+quote+anno.announcementText+quote+';

                                        $content=
'+quote+anno.announcementText+quote+';


add_announcement_dsl($title,$content);';
```

```
                                        //Contador de módulos creados
                                        numAnnouncementY =
numAnnouncementY+1;
                                    }
                                }
```

Código Fuente 39 Recorrido para los módulos Announcement en MOFScript para ATutor.

Después se recorren todos los módulos News que están dentro del modelo y conectados al módulo Communications, y se valida si en estos módulos los campos están rellenados, si no están rellenados incrementa en 1 un contador de los módulos News que están conectados pero no tienen información, si están rellenados se imprime una nueva línea en el fichero acompañado de dos espacios de tabulador, la invocación a la función correspondiente en Moodle, Claroline o Atutor con parámetros de entrada los nombres ingresados en el modelo y finalmente se incrementa un contador de los módulos creados correctamente. A continuación se muestra el código correspondiente para Moodle.

```
        self.LMSModel_Has_Communications.Communications_Has_News-
>forEach(news:mlms.News)
            {
                        if (news.newsTitle = null or news.newsBody =
null){
                            numNewsSN = numNewsSN+1;
                        }
                        else{
                        newline(1);
                        tab(2);
                        'def_news_dsl
('+quote+news.newsTitle+quote+','+quote+news.newsBody+quote+');';
                            numNewsY = numNewsY+1;
                        }
            }
```

Código Fuente 40 Recorrido para los módulos News en MOFScript para Moodle.

El siguiente fragmento de código muestra el recorrido para los módulos News en MOFScript para Claroline.

```
        self.LMSModel_Has_Communications.Communications_Has_News-
>forEach(news:mlms.News)

            {

                        if (news.newsTitle = null or news.newsBody =
null){

                              numNewsSN = numNewsSN+1;

                        }

                        else{

                        newline(2);

                        tab(3);

                        '$news_name ='+quote+news.newsTitle+quote+';

                              $news_body =
'+quote+news.newsBody+quote+';

                              create_news($news_name, $news_body);';

                              numNewsY = numNewsY+1;

                        }

                  }
```

Código Fuente 41 Recorrido para los módulos News en MOFScript para Claroline.

Y finalmente se presenta el código que recorre los módulos News en MOFScript para Atutor.

```
            self.LMSModel_Has_Communications.Communications_Has_News-
>forEach(news:mlms.News)

                  {

                    //Cuenta los módulos que tienen conexion, pero
algun campo esta sin llenar

                              if (news.newsTitle = null or
news.newsBody = null){

                                    numNewsSN = numNewsSN+1;
```

```
                                    }
                                    else{
                                    newline(2);
                                    tab(3);
                                    '$news_name
='+quote+news.newsTitle+quote+';
                                          $news_body =
'+quote+news.newsBody+quote+';
                                          add_new_dsl($news_body);';
                                          //Contador de módulos creados
                                          numNewsY = numNewsY+1;
                                    }
                              }
```

Código Fuente 42 Recorrido para los módulos News en MOFScript para ATutor.

El último recorrido se hace para todos los módulos Note que están dentro del modelo y conectados al módulo Communications, y se valida si en estos módulos los campos están rellenados, si no están rellenados incrementa en 1 un contador de los módulos Note que están conectados pero no tienen información, si están rellenados se imprime una nueva línea en el fichero acompañado de dos espacios de tabulador, la invocación a la función correspondiente en Moodle, Claroline o Atutor con parámetros de entrada los nombres ingresados en el modelo y finalmente se incrementa un contador de los módulos creados correctamente. En este módulo al ingresar un "*" se enviara una nota a todos los usuarios del curso y si se ingresa el correo con el cual el usuario esta registrado en el LMS se enviara una nota solo a ese usuario, esta validación para el caso de Moodle se realiza en el fichero lib.php y para el caso de Atutor en el fichero dsllib.inc.php, pero para demostrar el poder que tiene MOFScript, en Claroline se ha realizado en él. A continuación se muestra el código correspondiente para Moodle.

```
      self.LMSModel_Has_Communications.Communications_Has_Note-
>forEach(note:mlms.Note)
            {
                        if (note.sendTo = null or note.message =
null){
                              numNoteSN = numNoteSN+1;
```

```
                        }
                        else{
                        newline(1);
                        tab(2);
                        'def_note_dsl
('+quote+note.sendTo+quote+','+quote+note.message+quote+');';
                        numNoteY = numNoteY+1;
                        }
            }
      }
```

Código Fuente 43 Recorrido para los módulos Note en MOFScript para Moodle.

El siguiente fragmento de código muestra el recorrido para los módulos Note en MOFScript para Claroline.

```
      self.LMSModel_Has_Communications.Communications_Has_Note-
>forEach(note:mlms.Note)
            {
                        if (note.sendTo = null or note.message =
null){
                              numNoteSN = numNoteSN+1;
                        }
                        else{
                         newline(2);
                         tab(3);
                         if (note.sendTo="*"){
                                    '$subject   ='+quote+'Message from
Claroline platform'+quote+';
                                    $message
     ='+quote+note.message+quote+';
                                    $to
```

```
      ='+quote+note.sendTo+quote+';

                                    $toName
      ='+quote+note.sendTo+quote+';

                                    $from
      ='+quote+'pruebasdsl@yahoo.es'+quote+';

                                    $fromName   ='+quote+'Claroline
platform'+quote+';

                                    create_note_all($subject,
$message, $to, $toName, $from, $fromName);';


                           }

                           else{

                           newline(2);

                           tab(3);

                                'Subject   ='+quote+'Message from
Claroline platform'+quote+';

                                    $message
      ='+quote+note.message+quote+';

                                    $to
      ='+quote+note.sendTo+quote+';

                                    $toName
      ='+quote+note.sendTo+quote+';

                                    $from
      ='+quote+'pruebasdsl@yahoo.es'+quote+';

                                    $fromName   ='+quote+'Claroline
platform'+quote+';

                                    create_note($subject, $message,
$to, $toName, $from, $fromName);';

                           }

                           numNoteY = numNoteY+1;

                          }

             }
```

```
        }
```

Código Fuente 44 Recorrido para los módulos Note en MOFScript para Claroline.

Y finalmente se presenta el código que recorre los módulos News en MOFScript para Atutor.

```
            self.LMSModel_Has_Communications.Communications_Has_Note-
>forEach(note:mlms.Note)

                  {

                    //Cuenta los módulos que tienen conexion, pero
algun campo esta sin llenar

                              if (note.sendTo = null or note.message =
null){

                                    numNoteSN = numNoteSN+1;

                              }

                              else{

                               newline(2);

                               tab(3);

                               if (note.sendTo="*"){

                                          '$message
      ='+quote+note.message+quote+';


add_new_notemain("*",$message);';

                               }

                               else{

                               newline(2);

                               tab(3);

                                    '$to
      ='+quote+note.sendTo+quote+';

                                     $message
      ='+quote+note.message+quote+';

                                          add_new_notemain($to,
```

```
$message);';
                                    }
                                    //Contador de módulos creados
                                    numNoteY = numNoteY+1;
                                   }
                    }
```

Código Fuente 45 Recorrido para los módulos Note en MOFScript para ATutor.

Por último se invoca a la funciones final() y createlibphp(). Y se imprime en la consola el resumen de los módulos creados, los no creados discriminados por falta de conexión, y falta de información, como se ve a continuación:

```
    final();

      createlibphp();



      stdout.println("\n---Modules Information---\n");

      stdout.println("Number of Forums created="+numForumY);

      stdout.println("Number of Forums without connection="+numForumN-
numForumY);

      stdout.println("Number of Forums connected without
information="+numForumSN);



      stdout.println("\nNumber of Announcement
created="+numAnnouncementY);

      stdout.println("Number of Announcement without
connection="+numAnnouncementN-numAnnouncementY);

      stdout.println("Number of Announcement connected without
connection="+numAnnouncementSN);



      stdout.println("\nNumber of Chats created="+numChatY);

      stdout.println("Number of Chats without connection="+numChatN-
numChatY);
```

```
	stdout.println("Number of Chats connected without
connection="+numChatSN);



	stdout.println("\nNumber of Wikis created="+numWikiY);

	stdout.println("Number of Wikis without connection="+numWikiN-
numWikiY);

	stdout.println("Number of Wikis connected without
connection="+numWikiSN);



	stdout.println("\nNumber of News created="+numNewsY);

	stdout.println("Number of News without connection="+numNewsN-
numNewsY);

	stdout.println("Number of News connected without
connection="+numNewsSN);



	stdout.println("\nNumber of Notes created="+numNoteY);

	stdout.println("Number of Notes without connection="+numNoteN-
numNoteY);

	stdout.println("Number of Notes connected without
connection="+numNoteSN);



	stdout.println("\nTotal Modules
created="+numForumY+numAnnouncementY+numChatY+numWikiY+numNewsY+numNo
teY);



	}
```

Código Fuente 46 Llamado a las funciones final() y createlibphp() e impresión del resumen de módulos.

Adicionalmente para ATutor se llaman a las siguientes funciones también, esto antes de imprimir el resumen.

```
		createsql();
```

```
modulephp();

moduleuninstall();

moduleinstall();

modulexml();
```

Código Fuente 47 Funciones adicionales que se llaman en ATutor.

Finalmente se muestra el contenido de las funciones final() y createlibphp(), la primera función se encarga de imprimir el final del fichero creado y la segunda crea el fichero de las librerías para cada plataforma (Moodle, Claroline y ATutor) acorde a lo explicado en la sección anterior, sobre la creación de módulos para Moodle, Claroline y ATutor respectivamente. Para el caso de ATutor también se crean otras funciones. Estas funciones se muestran a continuación:

Función final para moodle:

```
module::final()

{

'

}

}'

}
```

Código Fuente 48 Función final en MOFScript para Moodle.

Función final para claroline:

```
module::final()

{

'

echo'+quote+'<hr />Loaded Modules in Claroline'+quote+';

?>'

}
```

Código Fuente 49 Función final en MOFScript para Claroline.

Función final para atutor:

```
module::final()

{

'


header('+quote+'Location:
'+quote+'.AT_BASE_HREF.'+quote+'tools/index.php'+quote+');


require(AT_INCLUDE_PATH.'+quote+'footer.inc.php'+quote+');

?>'

}
```

Código Fuente 50 Función final en MOFScript para ATutor.

A continuación se muestra el contenido de la función createlibphp para moodle:

```
module::createlibphp(){


 file ('lib.php');

'<?php

defined('+quote+'MOODLE_INTERNAL'+quote+') || die();

function plantilladsl_add_instance($plantilladsl) {

    global $DB;

    $plantilladsl->timecreated = time();

    # You may have to add extra stuff in here #

    return $DB->insert_record('+quote+'plantilladsl'+quote+',
$plantilladsl);

}

function plantilladsl_update_instance($plantilladsl) {

    global $DB;

    $plantilladsl->timemodified = time();
```

```
    $plantilladsl->id = $plantilladsl->instance;

    return $DB->update_record('+quote+'plantilladsl'+quote+',
$plantilladsl);

}

function plantilladsl_delete_instance($id) {

    global $DB;

    if (! $plantilladsl = $DB-
>get_record('+quote+'plantilladsl'+quote+', array('+quote+'id'+quote+'
=> $id))) {

        return false;

    }

    $DB->delete_records('+quote+'plantilladsl'+quote+',
array('+quote+'id'+quote+' => $plantilladsl->id));

    return true;

}

function plantilladsl_user_outline($course, $user, $mod,
$plantilladsl) {

    $return = new stdClass;

    $return->time = 0;

    $return->info = '+quote+''+quote+';

    return $return;

}

function plantilladsl_user_complete($course, $user, $mod,
$plantilladsl) {

    return true;

}

function plantilladsl_print_recent_activity($course, $viewfullnames,
$timestart) {

    return false;  //  True if anything was printed, otherwise false

}
```

```
function plantilladsl_cron () {

    return true;

}

function plantilladsl_get_participants($plantilladslid) {

    return false;

}

function plantilladsl_scale_used($plantilladslid, $scaleid) {

    global $DB;

    $return = false;

    return $return;

}

function plantilladsl_scale_used_anywhere($scaleid) {

    global $DB;

    if ($scaleid and $DB-
>record_exists('+quote+'plantilladsl'+quote+',
'+quote+'grade'+quote+', -$scaleid)) {

        return true;

    } else {

        return false;

    }

}

function plantilladsl_uninstall() {

    return true;

}

function def_forum_dsl($foronombre, $forointroduccion){

           global $USER, $CFG, $COURSE;

       $section = required_param('+quote+'section'+quote+',
PARAM_INT);
```

```
        $course = required_param('+quote+'course'+quote+', PARAM_INT);
    $forum->timemodified = time();
    $forum->course   = $course;
    $forum->name     = $foronombre;
    $forum->intro    = $forointroduccion;
    if (empty($forum->assessed)) {
        $forum->assessed = 0;
    }
    if (empty($forum->ratingtime) or empty($forum->assessed)) {
        $forum->assesstimestart  = 0;
        $forum->assesstimefinish = 0;
    }
    if (!$forum->id = insert_record('+quote+'forum'+quote+', $forum))
{
        return false;
        }
    if (record_exists("forum_subscriptions", "userid", $USER->id,
"forum", $forum->id)) {
        return true;
    }
    $sub = new object();
    $sub->userid  = $USER->id;
    $sub->forum = $forum->id;
     insert_record("forum_subscriptions", $sub);
    $sectioncourse->course = $course;
    $sectioncourse->module = 6;
    $sectioncourse->instance = $forum->id;
    $sectioncourse->added = time();
```

```
    $sectioncourse->section = $section;

    $sectioncourse->is_lams = 1;

    $sectioncourse->id =
insert_record('+quote+'course_modules'+quote+', $sectioncourse);

    $insertsection = get_record("course_sections", "section",
$section, "course", $course);

    $insertseccion->sequence =  $insertsection-
>sequence.'+quote+','+quote+'.$sectioncourse->id;

    $sequence='+quote+'"'+quote+'.$insertseccion-
>sequence.'+quote+'"'+quote+';

    global $db;

    $query = "UPDATE mdl_course_sections SET sequence=".$sequence."
WHERE id=".$insertsection->id;

     $db->Execute($query);

    }

function def_chat_dsl($roomname, $sendtext){

           global $USER, $CFG, $COURSE;

        $section = required_param('+quote+'section'+quote+',
PARAM_INT);

        $course = required_param('+quote+'course'+quote+', PARAM_INT);
// Course Module ID

    $chat->timemodified = time();

    $chat->course    = $course;

    $chat->name      = $roomname;

    $chat->intro     = $sendtext;

    if (!$chat->id = insert_record('+quote+'chat'+quote+', $chat)) {

        return false;

        }

    $sectioncourse->course = $course;

    $sectioncourse->module = 2;
```

```
    $sectioncourse->instance = $chat->id;

    $sectioncourse->added = time();

    $sectioncourse->section = $section;

    $sectioncourse->is_lams = 1;

    $sectioncourse->id =
insert_record('+quote+'course_modules'+quote+', $sectioncourse);

    $insertsection = get_record("course_sections", "section",
$section, "course", $course);

    $insertseccion->sequence =  $insertsection-
>sequence.'+quote+','+quote+'.$sectioncourse->id;

    $sequence='+quote+'"'+quote+'.$insertseccion-
>sequence.'+quote+'"'+quote+';

    global $db;

    $query = "UPDATE mdl_course_sections SET sequence=".$sequence."
WHERE id=".$insertsection->id;

     $db->Execute($query);

    }

function def_wiki_dsl($wikiname, $wikitext){

           global $USER, $CFG, $COURSE;

        $section = required_param('+quote+'section'+quote+',
PARAM_INT);

        $course = required_param('+quote+'course'+quote+', PARAM_INT);
// Course Module ID

    $wiki->timemodified = time();

    $wiki->course    = $course;

    $wiki->name      = $wikiname;

    $wiki->summary   = $wikitext;

    $wiki->pagename = $wikiname;

    if (!$wiki->id = insert_record('+quote+'wiki'+quote+', $wiki)) {

        return false;
```

```
        }

    $sectioncourse->course = $course;

    $sectioncourse->module = 17;

    $sectioncourse->instance = $wiki->id;

    $sectioncourse->added = time();

    $sectioncourse->section = $section;

    $sectioncourse->is_lams = 1;

    $sectioncourse->id =
insert_record('+quote+'course_modules'+quote+', $sectioncourse);

    $insertsection = get_record("course_sections", "section",
$section, "course", $course);

    $insertseccion->sequence =  $insertsection-
>sequence.'+quote+','+quote+'.$sectioncourse->id;

    $sequence='+quote+'"'+quote+'.$insertseccion-
>sequence.'+quote+'"'+quote+';

    global $db;

    $query = "UPDATE mdl_course_sections SET sequence=".$sequence."
WHERE id=".$insertsection->id;

     $db->Execute($query);

    }

function def_Announcement_dsl($announcementname){

           global $USER, $CFG, $COURSE;

        $section = required_param('+quote+'section'+quote+',
PARAM_INT);    // Course section ID

        $course = required_param('+quote+'course'+quote+', PARAM_INT);
// Course Module ID

    $label->timemodified = time();

    $label->course       = $course;

    $label->name         = $announcementname;

    $label->content      = $announcementname;
```

```
    if (!$label->id = insert_record('+quote+'label'+quote+', $label))
{

        return false;

        }

    $sectioncourse->course = $course;

    $sectioncourse->module = 10;

    $sectioncourse->instance = $label->id;

    $sectioncourse->added = time();

    $sectioncourse->section = $section;

    $sectioncourse->is_lams = 1;

    $sectioncourse->id =
insert_record('+quote+'course_modules'+quote+', $sectioncourse);

    $insertsection = get_record("course_sections", "section",
$section, "course", $course);

    $insertseccion->sequence =  $insertsection-
>sequence.'+quote+','+quote+'.$sectioncourse->id;

    $sequence='+quote+'"'+quote+'.$insertseccion-
>sequence.'+quote+'"'+quote+';

    global $db;

    $query = "UPDATE mdl_course_sections SET sequence=".$sequence."
WHERE id=".$insertsection->id;

     $db->Execute($query);

    }

function def_news_dsl($newstitle, $newsbody){

    global $USER, $CFG, $COURSE;

    $course = required_param('+quote+'course'+quote+', PARAM_INT);

    $type   = '+quote+'news'+quote+';

    $timenow = time();

   $forum = get_record('+quote+'forum'+quote+',
'+quote+'course'+quote+',$course, '+quote+'type'+quote+',
```

```
'+quote+'news'+quote+');

    $post = new object();

    $post->discussion  = 0;

    $post->parent      = 0;

    $post->userid      = $USER->id;

    $post->created     = $timenow;

    $post->modified    = $timenow;

    $post->mailed      = 0;

    $post->subject     = $newstitle;

    $post->message     = $newsbody;

    $post->attachment  = "";

    $post->forum       = $forum->id;     // speedup

    $post->course      = $forum->course; // speedup

    $post->format      = 1;

    $post->mailnow     = 0;

   $post->id = insert_record("forum_posts", $post);

    $discussion->firstpost      = $post->id;

    $discussion->timemodified   = $timenow;

    $discussion->usermodified   = $post->userid;

    $discussion->userid         = $USER->id;

    $discussion->course         = $post->course;

    $discussion->forum          = $post->forum;

    $discussion->name           = $post->subject;

    $post->discussion = insert_record("forum_discussions",
$discussion);

    set_field("forum_posts", "discussion", $post->discussion, "id",
$post->id);

}
```

```
function def_note_dsl($sendto, $messagebody){

    global $USER, $CFG, $COURSE;

    $timenow = time();

if ($sendto='+quote+'*'+quote+'){

  $course    = required_param('+quote+'course'+quote+', PARAM_INT);
// Course Module ID

  $context  = get_record('+quote+'context'+quote+',
'+quote+'contextlevel'+quote+', 50, '+quote+'instanceid'+quote+',
$course);

  $usersid  = get_records ('+quote+'role_assignments'+quote+',
'+quote+'contextid'+quote+', $context->id);


  foreach ($usersid as $users){

  $emails    = get_record ('+quote+'user'+quote+',
'+quote+'id'+quote+', $users->userid);

    $message = new object();

    $message->useridfrom    = $USER->id;

    $message->useridto      = $emails->id;

    $message->message       = $messagebody;

    $message->timecreated   = $timenow;

    $message->messagetype   = '+quote+'direct'+quote+';

   insert_record("message", $message);

  }

 }

else if ($userto){

    $userto = get_record('+quote+'user'+quote+',
'+quote+'email'+quote+',$sendto);

    $message = new object();

    $message->useridfrom    = $USER->id;
```

```
    $message->useridto      = $userto->id;
    $message->message       = $messagebody;
    $message->timecreated   = $timenow;
    $message->messagetype   = '+quote+'direct'+quote+';
   insert_record("message", $message);
   }
   else{
   }
}
    '
      }
```

Código Fuente 51 Función createlibphp en MOFScript para Moodle.

Ahora se muestra el contenido de la función createlibphp para Claroline:

```
module::createlibphp(){


 file ('plantilladsl.lib.php');
'<?php
if ( count( get_included_files() ) == 1 ) die( '+quote+'---'+quote+'
);
function create_forum($forum_name, $forum_desc, $forum_post_allowed=2,
$cat_id = 2, $group_id = NULL, $course_id=NULL)
{
    $tbl_cdb_names =
claro_sql_get_course_tbl(claro_get_course_db_name_glued($course_id));
    $tbl_forum_forums = $tbl_cdb_names['+quote+'bb_forums'+quote+'];


    // find order in the category we have to give to the newly created
forum
```

```
    $sql = "SELECT MAX(`forum_order`)
            FROM `" . $tbl_forum_forums . "`
            WHERE cat_id = " . (int) $cat_id;

    $result = claro_sql_query($sql);

    list($orderMax) = mysql_fetch_row($result);
    $order = $orderMax + 1;

    // add new forum in DB

    $sql = "INSERT INTO `" . $tbl_forum_forums . "`
            SET forum_name      = '+quote+'" . addslashes($forum_name)
. "'+quote+',
                group_id        = " . (is_null($group_id) ? "NULL" :
(int) $group_id) . ",
                forum_desc      = '+quote+'" . addslashes($forum_desc)
. "'+quote+',
                forum_access    = " . ($forum_post_allowed ? 2 : 0) .
",
                forum_moderator = 1,
                cat_id          = " . (int) $cat_id . ",
                forum_type      = 0,
                forum_order     = " . (int) $order ;

    return claro_sql_query_insert_id($sql);
}
function
```

```
announcement_add_item($title='+quote+''+quote+',$content='+quote+''+qu
ote+', $visibility='+quote+'SHOW'+quote+', $time=NULL,
$course_id=NULL)

{

    $tbl=
claro_sql_get_course_tbl(claro_get_course_db_name_glued($course_id));


    if(is_null($time))

    {

        $sqlTime = " temps = NOW(), ";

    }

    else

    {

        $sqlTime = " temps = from_unixtime('+quote+'". (int)$time
."'+quote+'), ";

    }


    // DETERMINE THE ORDER OF THE NEW ANNOUNCEMENT

    $sql = "SELECT (MAX(ordre) + 1) AS nextRank

            FROM  `" . $tbl['+quote+'announcement'+quote+'] . "`";


    $nextRank = claro_sql_query_get_single_value($sql);

    // INSERT ANNOUNCEMENT


    $sql = "INSERT INTO `" . $tbl['+quote+'announcement'+quote+'] . "`

            SET title ='+quote+'" . claro_sql_escape(trim($title)) .
"'+quote+',

                contenu = '+quote+'" .
claro_sql_escape(trim($content)) . "'+quote+',

                visibility = '+quote+'" .
```

```
($visibility=='+quote+'HIDE'+quote+'?'+quote+'HIDE'+quote+':'+quote+'S
HOW'+quote+') . "'+quote+',

             ". $sqlTime ."

            ordre ='+quote+'" . (int) $nextRank . "'+quote+'";

    return claro_sql_query_insert_id($sql);

}

$coursePath  = get_path('+quote+'coursesRepositorySys'+quote+') .
claro_get_course_path();

$courseId    = claro_get_current_course_id();

$groupId     = claro_get_current_group_id();

$_user       = claro_get_current_user_data();

$_course     = claro_get_current_course_data();

$_group      = claro_get_current_group_data();


$is_allowedToManage = claro_is_course_manager();

$is_allowedToStore  = claro_is_course_manager();

$is_allowedToReset  = claro_is_course_manager();


//nick of the User

    $nick = $_user['+quote+'firstName'+quote+'] . '+quote+' '+quote+'
. $_user['+quote+'lastName'+quote+'] ;

$curChatRep = $coursePath.'+quote+'/chat/'+quote+';


        $activeChatFile =
$curChatRep.$courseId.'+quote+''+quote+'.$groupId.'+quote+'.chat.html'
+quote+';


$dateNow =
claro_html_localised_date(get_locale('+quote+'dateTimeFormatLong'+quot
e+'));
```

```
$timeNow = claro_html_localised_date('+quote+'[%d/%m/%y
%H:%M]'+quote+');

/*--------------------------------------------------------------------
--------

    '+quote+'ADD NEW LINE'+quote+' COMMAND

--------------------------------------------------------------------
------*/

          $fchat = fopen($activeChatFile,'+quote+'a'+quote+');

         // $chatLine = htmlspecialchars(
$_REQUEST['+quote+'chatLine'+quote+'] );

          // replace url with real html link



          // En la plantilla de MOFScript por cada barra (slash) que
deseemos imprimir se debe agregar otro para que lo imprima bien

          $chatLine =
ereg_replace("(http://)(([[:punct:]]|[[:alnum:]])*)","<a
href=\\"\\\\0\\" target=\\"_blank\\">\\\\2</a>",$message);



          fwrite($fchat, '+quote+'<small>'+quote+' . $timeNow .
'+quote+' &lt;<b>'+quote+' . $nick . '+quote+'</b>&gt; '+quote+' .
$chatLine . '+quote+'</small><br />'+quote+' . "\n");



          fclose($fchat);

}

function create_wiki($wikiName, $wikiDescription, $gid = false )

{



      require_once
"../../Claroline/wiki/lib/class.clarodbconnection.php";

    require_once
"../../Claroline/wiki/lib/class.wikiaccesscontrol.php";

    require_once  "../../Claroline/wiki/lib/class.wikistore.php";
```

```
    require_once  "../../Claroline/wiki/lib/class.wikipage.php";
    require_once  "../../Claroline/wiki/lib/class.wiki.php";
    require_once  "../../Claroline/wiki/lib/lib.wikisql.php";

        $creatorId = claro_get_current_user_id();

        $tblList = claro_sql_get_course_tbl();

        $config = array();
        $config["tbl_wiki_properties"] = $tblList[ "wiki_properties"
];
        $config["tbl_wiki_pages"] = $tblList[ "wiki_pages" ];
        $config["tbl_wiki_pages_content"] = $tblList[
"wiki_pages_content" ];
        $config["tbl_wiki_acls"] = $tblList[ "wiki_acls" ];

        $con = new ClarolineDatabaseConnection();

        $acl = array();

        if ( $gid )
        {
            $acl = WikiAccessControl::defaultGroupWikiACL();
        }
        else
        {
            $acl = WikiAccessControl::defaultCourseWikiACL();
        }
```

```
        $wiki = new Wiki( $con, $config );
        $wiki->setTitle( $wikiName );
        $wiki->setDescription( $wikiDescription );
        $wiki->setACL( $acl );
        $wiki->setGroupId( $gid );
        $wikiId = $wiki->save();
        $wikiTitle = $wiki->getTitle();

        $mainPageContent = sprintf( "This is the main page of the Wiki
%s. Click on edit to modify the content.", $wikiTitle );

        $wikiPage = new WikiPage( $con, $config, $wikiId );
        $wikiPage->create( $creatorId
            , '+quote+'__MainPage__'+quote+'
            , $mainPageContent
            , date( "Y-m-d H:i:s" )
            , true );

        echo $con->getError();
}
function create_news($news_name, $news_body)
{
    $tbl_cdb_names =
claro_sql_get_course_tbl(claro_get_course_db_name_glued($course_id));
    $tbl_tool_intro = $tbl_cdb_names['+quote+'tool_intro'+quote+'];

       // add new News in DB
```

```
    $sql = "INSERT INTO `" . $tbl_tool_intro . "`
            SET title          = '+quote+'".$news_name."'+quote+',
                content        = '+quote+'".$news_body."'+quote+',
                rank             = 1,
                tool_id          = 0";

    return claro_sql_query_insert_id($sql);
}
function create_note($subject, $message, $to, $toName, $from,
$fromName)
{
//import lib of the sendmail
      require_once  "../../Claroline/inc/lib/sendmail.lib.php";

    $mail = new ClaroPHPMailer();

    if (empty($from))
    {
        $from = get_conf('+quote+'administrator_email'+quote+');
        if (empty($fromName))
        {
            $fromName =
get_conf('+quote+'administrator_name'+quote+');
        }
    }

    $mail->Subject  = $subject;
```

```
    $mail->Body     = $message;
    $mail->From     = $from;
    $mail->FromName = $fromName;
    $mail->Sender   = $from;

    $mail->AddAddress($to,$toName);

    if ( $mail->Send() )
    {
        return true;
    }
    else
    {
        return claro_failure::set_failure($mail->getError());
    }
}
function create_note_all($subject, $message, $to, $toName, $from,
$fromName)
{
//Selection id of the course
$courseId    = claro_get_current_course_id();

//SQL Query
      $sql= "select email
                  from cl_user
                  inner join cl_cours_user
                  on cl_user.user_id = cl_cours_user.user_id
                  where cl_cours_user.code_cours =
```

```
'+quote+'".$courseId."'+quote+'";


// Sen a E-Mail users

      $listMail = claro_sql_query_fetch_all($sql);

      foreach ($listMail as $namemail){

            foreach ($namemail as $mail){

            // Send a e-mail

                  create_note ($subject, $message, $mail, $mail,
$from, $fromName);

            }

      }

}

    '

      }
```

Código Fuente 52 Función createlibphp en MOFScript para Claroline.

Y este es el contenido de la función createlibphp, createsql, modulephp, moduleuninstall, moduleinstall y modulexml para atutor.

```
module::createlibphp(){


       file ('/plantilladsl/lib/dsllib.lib.php');

      '<?php

      if (!defined('+quote+'AT_INCLUDE_PATH'+quote+')) { exit; }

      require(AT_INCLUDE_PATH.'+quote+'../mods/plantilladsl/lib/class.
phpmailer.php'+quote+');

      require(AT_INCLUDE_PATH.'+quote+'../mods/plantilladsl/lib/class.
smtp.php'+quote+');


      //Funcion para crear un new.
```

```
	function add_new_dsl($describenew){

		global $db;

		$sql = "INSERT INTO ".TABLE_PREFIX."assignments VALUES
(NULL,$_SESSION[course_id], '+quote+'$describenew'+quote+',
'+quote+'0'+quote+',NOW(), '+quote+'0000-00-00 00:00:00'+quote+',
'+quote+'0'+quote+')";


		mysql_query($sql,$db);

		return;

	}


	///Funcion para crear un anuncio.

	function add_announcement_dsl($titulo,$Announdescripcion){

		global $db;

		$sql = "INSERT INTO ".TABLE_PREFIX."news VALUES
(NULL,$_SESSION[course_id], $_SESSION[member_id], NOW(),1,
'+quote+'$titulo'+quote+', '+quote+'$Announdescripcion'+quote+')";

		mysql_query($sql, $db);

	}


	//Funcion para crear un foro.

	function add_forum_dsl($foronombre,$forointroduccion){

		global $db;

		$sql  = "INSERT INTO ".TABLE_PREFIX."forums VALUES
(NULL,'+quote+'$foronombre'+quote+','+quote+'$forointroduccion'+quote
+', 0, 0, NOW(),0)";

		mysql_query($sql,$db);


		$sql  = "INSERT INTO ".TABLE_PREFIX."forums_courses VALUES
(LAST_INSERT_ID(), $_SESSION[course_id])";
```

```
        mysql_query($sql,$db);


        return;
}
//Funcion para enviar un note.
function add_note_dsl($sendto,$messagebody){
        $mail = new PHPMailer ();
        $mail -> From = "comarmeza@gmail.com";
        $mail -> FromName = "DSL";
        $mail -> AddAddress ("$sendto");
        $mail -> Subject = "PruebaDSL";
        $mail -> Body = "<h3>$messagebody</h3>";
        $mail -> IsHTML (true);
        $mail->IsSMTP();


        $mail->Host = '+quote+'ssl://smtp.gmail.com'+quote+';
        $mail->Port = 465;
        $mail->SMTPAuth = true;
        $mail->Username = '+quote+'comarmeza@gmail.com'+quote+';
        $mail->Password = '+quote+'1400182comarmeza'+quote+';


        if(!$mail->Send()) {
                echo '+quote+'Error: '+quote+' . $mail->ErrorInfo;
        }
        else {
                echo '+quote+'Mail enviado!'+quote+';
        }
```

```
	}


	//Funcion para enviar uno o varios note.

	function add_new_notemain($sendto,$bodynote){

		global $db;

		if($sendto=='+quote+'*'+quote+'){

			$sql	= "SELECT email FROM
".TABLE_PREFIX."course_enrollment C INNER JOIN
".TABLE_PREFIX."members M USING (member_id) WHERE
C.course_id=$_SESSION[course_id]";

		 $sendto = mysql_query($sql,$db);

			$row	= mysql_fetch_array($sendto);


			foreach($row as $usernote){


add_note_dsl($row['+quote+'email'+quote+'],$bodynote);

			}

		}else{

			 $sql = '+quote+'SELECT member_id FROM
'+quote+'.TABLE_PREFIX.'+quote+'members WHERE
email="'+quote+'.$sendto.'+quote+'"'+quote+';

			$enviara = mysql_query($sql,$db);

			if ($enviara){

			$row	= mysql_fetch_array($enviara);

				$enviar = $row['+quote+'member_id'+quote+'];


			$sql = "INSERT INTO ".TABLE_PREFIX."messages VALUES
(NULL, $_SESSION[course_id], $_SESSION[member_id], $enviar, NOW(), 1,
0, '+quote+'DSLnote'+quote+', '+quote+'$bodynote'+quote+')";

			mysql_query($sql,$db);
```

```
                $sql = "INSERT INTO ".TABLE_PREFIX."messages_sent
VALUES (NULL, $_SESSION[course_id], $_SESSION[member_id],$enviar,
NOW(), '+quote+'DSLnote'+quote+', '+quote+'$bodynote'+quote+')";

                mysql_query($sql,$db);

                }

                      else{

                            echo "Usuario No existe";

                }

          }

      }

      ?>

       '

      }

      module::createsql(){

            file modulesql("/plantilladsl/module.sql");

            modulesql.println("# sql file for plantilladsl module");

            modulesql.println("");

            modulesql.println("INSERT INTO `language_text` VALUES
('en', '_module','plantilladsl','Plantilladsl',NOW(),'');");

      }


      module::modulephp(){

      file ("/plantilladsl/module.php");

      '

      <?php

      define('+quote+'AT_INCLUDE_PATH'+quote+',
'+quote+'../../../include/'+quote+');
```

```
      if (!defined('+quote+'AT_INCLUDE_PATH'+quote+')) { exit; }

      if (!isset($this) || (isset($this) &&
(strtolower(get_class($this)) != '+quote+'module'+quote+'))) {
exit(__FILE__ . '+quote+' is not a Module'+quote+'); }


      define('+quote+'AT_PRIV_PLANTILLADSL'+quote+',       $this-
>getPrivilege());

      define('+quote+'AT_ADMIN_PRIV_PLANTILLADSL'+quote+', $this-
>getAdminPrivilege());


      $this-
>_pages['+quote+'mods/plantilladsl/index.php'+quote+']['+quote+'title
_var'+quote+'] = '+quote+'plantilla_dsl'+quote+';

      $this-
>_pages['+quote+'mods/plantilladsl/index.php'+quote+']['+quote+'paren
t'+quote+']    = '+quote+'tools/index.php'+quote+';


      ?>

      '

      }


      module::moduleinstall(){

        file ("/plantilladsl/module_install.php");

      '

      <?php

      if (!defined('+quote+'AT_INCLUDE_PATH'+quote+')) { exit; }


      $_course_privilege = TRUE; // possible values: FALSE |
AT_PRIV_ADMIN | TRUE

      $_admin_privilege  = TRUE; // possible values: FALSE | TRUE

            if (!$msg->containsErrors() &&
```

```
file_exists(dirname(__FILE__) . '+quote+'/module.sql'+quote+')) {

            // deal with the SQL file:

            require(AT_INCLUDE_PATH .
'+quote+'classes/sqlutility.class.php'+quote+');

            $sqlUtility =& new SqlUtility();

                  $sqlUtility->queryFromFile(dirname(__FILE__) .
'+quote+'/module.sql'+quote+', TABLE_PREFIX);

      }


      ?>


      '

      }


      module::moduleuninstall(){

      file ("/plantilladsl/module_uninstall.php");

      '

      <?php

      if (!defined('+quote+'AT_INCLUDE_PATH'+quote+')) { exit; }

      if (!$msg->containsErrors() && file_exists(dirname(__FILE__) .
'+quote+'/module.sql'+quote+')) {

            // deal with the SQL file:

            require(AT_INCLUDE_PATH .
'+quote+'classes/sqlutility.class.php'+quote+');

            $sqlUtility = new SqlUtility();


            $sqlUtility->revertQueryFromFile(dirname(__FILE__) .
'+quote+'/module.sql'+quote+', TABLE_PREFIX);

      }
```

```
?>
'
}

module::modulexml(){
file ("/plantilladsl/module.xml");
'
<?xml version="1.0" encoding="ISO-8859-1"?>
<module version="0.1">
    <name lang="en">platilladsl</name>
    <description lang="en">descripcion de plantilla
dsl</description>
    <maintainers>
        <maintainer>
            <name>ATutor Team</name>
            <email>info@ATutor.ca</email>
        </maintainer>
    </maintainers>
    <url>http://ATutor.ca</url>
    <license>GPL</license>
      <release>
        <version>0.1</version>
            <privileges>

<instructor_privilege>create</instructor_privilege>
                    <admin_privilege></admin_privilege>
            </privileges>
```

```
            <date>2011-02-24</date>
            <state>stable</state>
            <notes>This is a standard module.</notes>
        </release>
    </module>
    '
    }
```

Código Fuente 53 Funciones createlibphp, createsql, modulephp, moduleuninstall, moduleinstall, modulexml en MOFScript para ATutor.

3.6.6 Transformación de modelos a módulos para Moodle, Claroline y ATutor desde un modelo en KiwiDSM v2.0

Para este apartado debemos partir de la mejora realizada al metamodelo LMS que se muestra en la figura 59, Aquí nos centraremos en explicar únicamente cómo funcionan los cambio acordes al nuevo metamodelo en MOFScript, estos cambios aplican para cualquiera de las tres transformaciones (moodle, atutor ó claroline).

Como ahora solo existe una conexión desde el modulo “Communications”, que es llamada “Communications_Has_Module” a la superclase “Module”, es necesario recorrer todos los nodos conectados entre “Communications” y “Module”, esto se muestra a continuación.

```
    self.LMSModel_Has_Communications.Communications_Has_Module-
>forEach(modul:mlms.Module){
```

Código Fuente 54 Instrucción para recorrer todos los módulos conectados al nodo Communications en MOFScript.

Debido a que la instrucción anterior recorre todos los módulos conectados a “Communications”, ahora es necesario validar que cada uno de los módulos que se están recorriendo correspondan a un Foro, Chat, Wiki, Anuncio, Noticia ó Nota, para ellos se ha utilizado la función “oclGetType()”, que retorna el nombre del nodo sobre el cual se está en ese momento, con esta función se compara si el nodo corresponde por ejemplo a un “Forum”, un ejemplo como se muestra a continuación.

```
if (modul.oclGetType()='Forum'){
```

Código Fuente 55 Instrucción para validar si se está sobre el modulo Forum en MOFScript.

De ser verdad la condición anterior se ingresa y se crea una nueva variable de tipo "Forum" a la cual se le asigna el valor del modulo sobre el cual se está en ese momento y que acorde a la validación anterior debe ser del tipo correcto, un ejemplo para "Forum" se muestra a continuación.

```
        var forum:mlms.Forum;

        forum = modul;
```

Código Fuente 56 Creación de una variable de tipo Forum y asignación de una variable de tipo module a ella en MOFScript.

Las demás instrucciones se mantienen como ya se habían explicado en la sección anterior con la primera versión de KiwiDSM v1.0. A continuación se muestra como quedarían las validaciones para los demás módulos.

Validación para módulos chat:

```
if (modul.oclGetType()='Chat'){

        var chat:mlms.Chat;

        chat = modul;
```

Código Fuente 57 Validación y creación de una variable de tipo Chat en MOFScript.

Validación para módulos wiki:

```
      if (modul.oclGetType()='Wiki'){

        var wiki:mlms.Wiki;

        wiki = modul;
```

Código Fuente 58 Validación y creación de una variable de tipo Wiki en MOFScript.

Validación para módulos announcement:

```
      if (modul.oclGetType()='Announcement'){

        var anno:mlms.Announcement;

        anno = modul;
```

Código Fuente 59 Validación y creación de una variable de tipo Announcement en MOFScript.

Validación para módulos news:

```
      if (modul.oclGetType()='News'){
```

```
var news:mlms.News;

news = modul;
```

Código Fuente 60 Validación y creación de una variable de tipo News en MOFScript.

Validación para módulos note:

```
if (modul.oclGetType()='Note'){

  var note:mlms.Note;

  note = modul;
```

Código Fuente 61 Validación y creación de una variable de tipo Note en MOFScript.

3.7 Pruebas y validaciones.

El objetivo de esta sección es validar el metamodelo planteado y comprobar si la hipótesis que se planteó al inicio del proyecto es *verdad "Es posible crear un nivel de abstracción tan alto que sea este compatible con diversas plataformas LMSs y así evitar el problema de la incompatibilidad entre diferentes plataformas LMSs planteado en (Bizonova and Ranc, 2008) a través de un metamodelo, DSL y MDE, igualmente el uso y aplicación de estas nuevas tendencias reducirá el tiempo en la creación de módulos para un LMS"*. Las pruebas medirán el tiempo y esfuerzo (usabilidad) de los usuarios para crear exactamente los mismos módulos en Moodle y la herramienta DSL creada, cada uno de los usuario creara 5 temas y cada uno de los temas tendrá en este orden: 1 Chat, 1 Forum, 1 Wiki, 1 Announcement, 1 News y 1 Note a todas las personas inscritas en ese curso, la información de cada módulo será la misma en todos los casos y los valores se han normalizado al mayor.

Antes de comenzar a realizar las pruebas se explicará cómo se crean cada uno de los módulos trabajados bajo cuatro enfoques: la herramienta DSL creada (KiwiDSM), la plataforma LMS Moodle, la plataforma LMS Claroline y la plataforma LMS Atutor.

3.7.1 Generación del modelo con la herramienta DSL KiwiDSM

A continuación se explicara como un usuario profesor puede crear módulos chat, foro, wiki, anuncios, noticias o notas, con la herramienta KiwiDSM creada e independiente de la plataforma LMS sobre la cual se vaya a desplegar, demostrando así que el metamodelo es compatible para diversas plataformas LMS.

Esta herramienta esta creada bajo entorno eclipse y funciona como un plugin para él, en la construcción de la herramientas se utilizaron conceptos y tecnologías como: MDE, EMF, Ecore, GMF, MOFScript, entre otras.

Para poder correr la herramienta KiwiDSM es necesario crear un “Java Project” en eclipse. Teniendo eclipse abierto se da clic en “File” ->“New”-> “Java Project”, como se muestra en la siguiente figura.

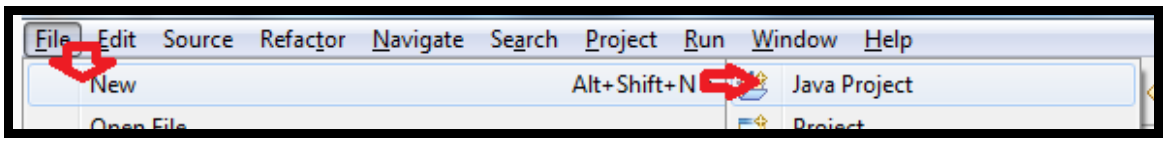


Figura 66 Creación de un Java Project en eclipse.

Se coloca un nombre al proyecto en “Project name” y se da clic en Finish, como lo muestra la siguiente figura.

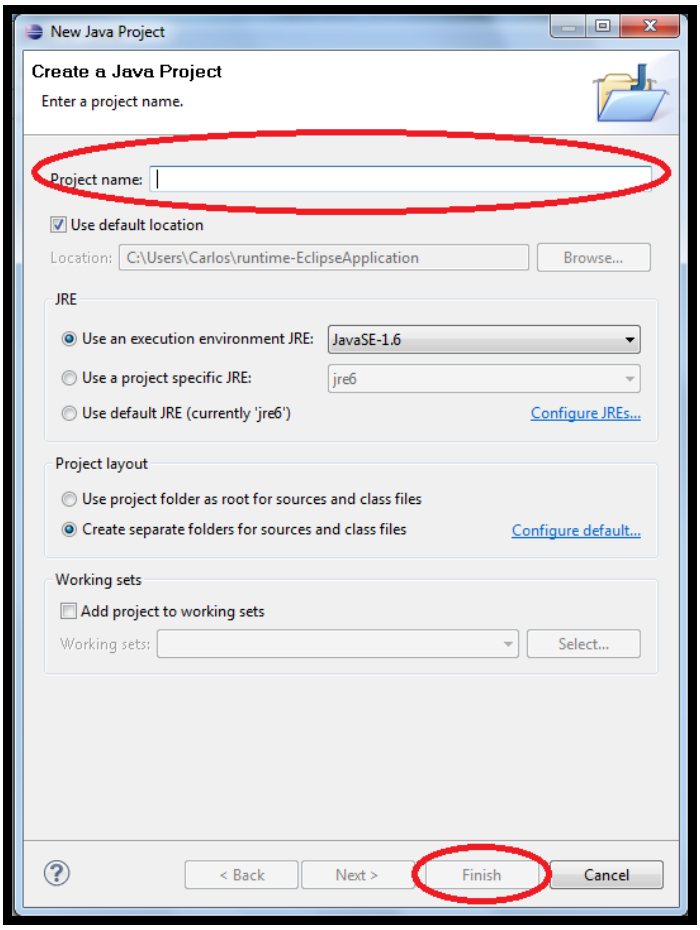


Figura 67 Nombre para un nuevo proyecto en eclipse.

Esto creara un nuevo proyecto en eclipse. En el “Package Explorer”, se vería como en la siguiente figura.

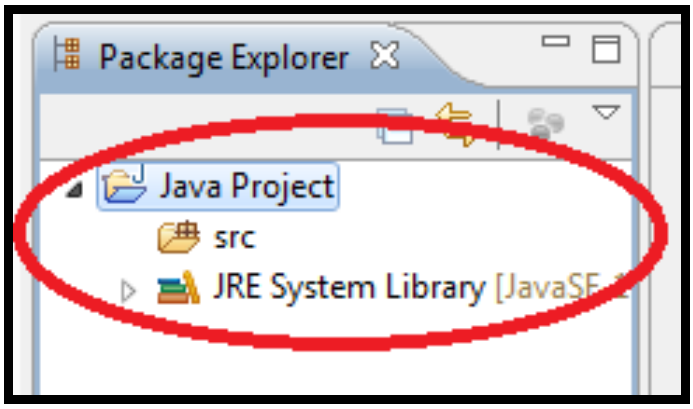


Figura 68 Package explorer de eclipse.

Luego se da clic derecho sobre el "Java Project" creado y se escoge la opción "New" -> "Example", como se ve en la siguiente figura.

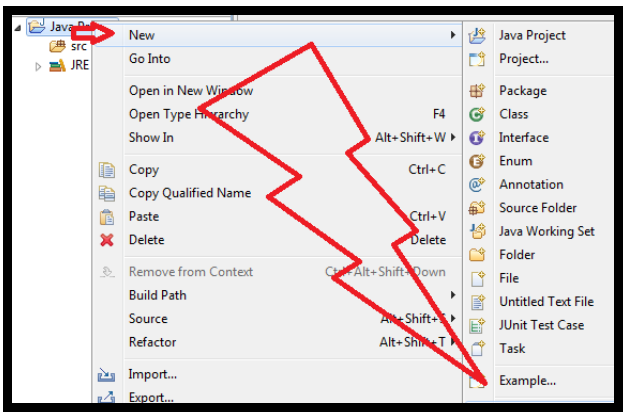


Figura 69 Creación de un nuevo proyecto ejemplo en eclipse.

Aparecerá una nueva ventana, en ella se seleccionará "Modellms Diagram" ó "Upmodelo Diagram" que corresponden a la herramientas de modelado para módulos de plataformas LMS KiwiDSM v1.0 y KiwiDSM v2.0 respectivamente, como lo muestra la siguiente figura.

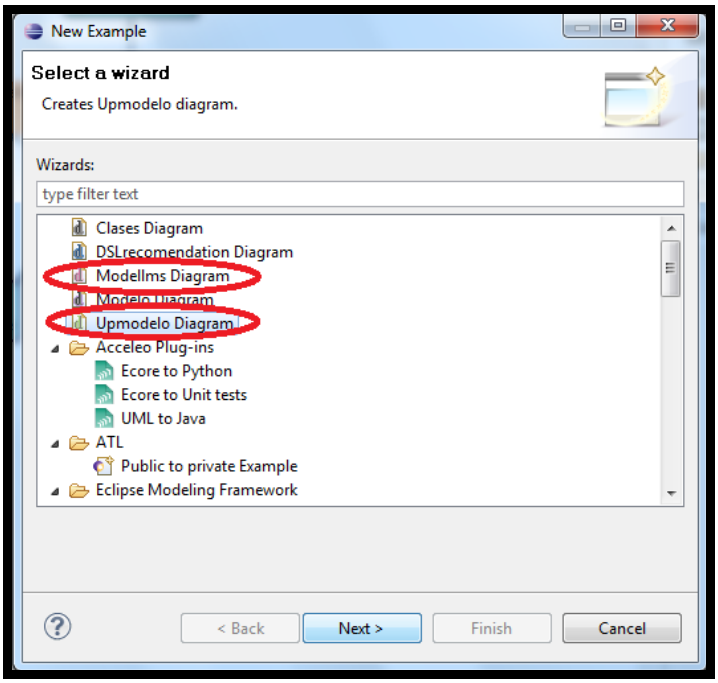


Figura 70 Creación de un proyecto de tipo "Modellms Diagram" ó ""Upmodelo Diagram".

Clic sobre el botón "next", apareceré una nueva ventana en donde se pone el nombre del fichero a crear en "File name" y se da "Finish", como lo muestra la siguiente figura.

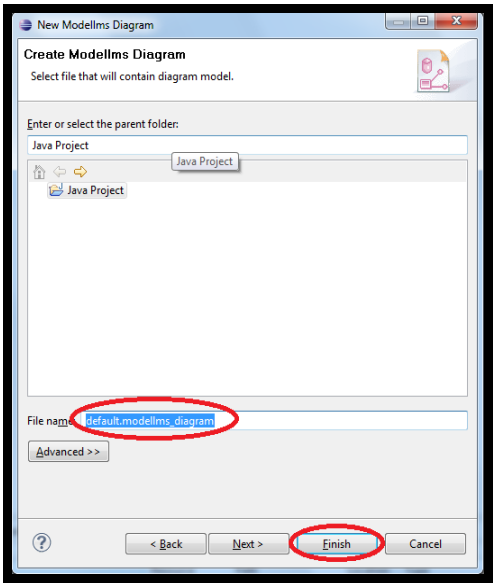


Figura 71 Nombre del proyecto de tipo "Modellms Diagram" ó "Upmodelo Diagram".

Así se ha creado un nuevo proyecto en la herramienta, esta se compone de un "package explorer", una paleta de herramientas "Palette" y un área de trabajo, tal y como se ve en la figura de abajo.

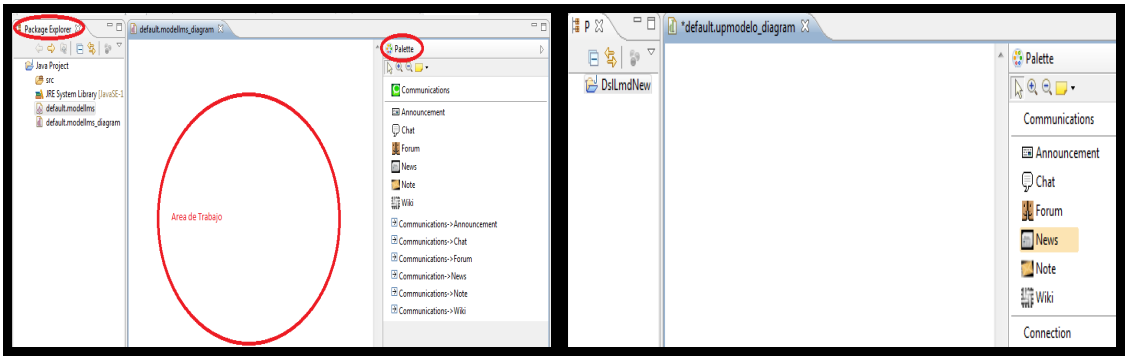


Figura 72 Vista de la herramienta DSL KiwiDSM v1.0 y v2.0 respectivamente.

El funcionamiento de la herramienta es muy sencillo, la paleta de herramientas "Palette" básicamente tiene dos tipos de herramientas a seleccionar Nodos y Conectores, los primeros corresponden a los módulos que tiene un LMS en el área de Comunicaciones y los otros son las conexiones que existen entre esos nodos, como se muestra en la siguiente figura.

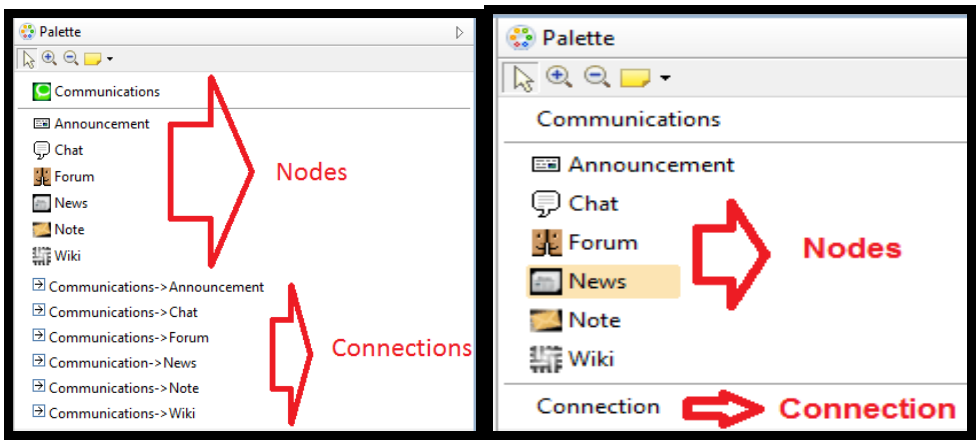


Figura 73 Paleta de herramientas de la herramienta DSL KiwiDSM v1.0 y v2.0 respectivamente.

Para crear los módulos basta con arrastrar los nodos de la "Pallete" al área de trabajo, rellenar los campos y conectarlos respetando las siguientes reglas:

- Un cursos solo puede tener un módulo de "Communications".
- El módulo de "Communications" tiene cero o muchos: "Announcements", "Chat", "Forum", "News", "Note", y "Wiki".

La herramienta valida los siguientes casos:

- Solo permite un módulo de "Communications".
- Los enlaces solo pueden corresponder entre el nodo "Communications" y su respectiva herramienta, por ejemplo "Communications -> Chat ", solo sirve para

conectar el nodo “Communications” con el nodo “Chat”, la herramienta no permite utilizarlo en otro caso.

- Cuando se genere el código a desplegar, aquellos nodos que estén sueltos (sin conexión) no serán tenidos en cuenta para la creación del curso.

Es muy importante considerar que los elementos modelados con esta herramienta se desplegaran en un solo tema del curso para Moodle y en un curso completo para Claroline y ATutor, con lo cual es substancial considerar los nombre de los campos en cada elemento (Nodo o Módulo) como genéricos, para no tener que cambiar el modelo cada vez que se despliegue sobre el curso.

A continuación se muestra la creación de módulos en KiwiDSM v1.0 y v2.0.

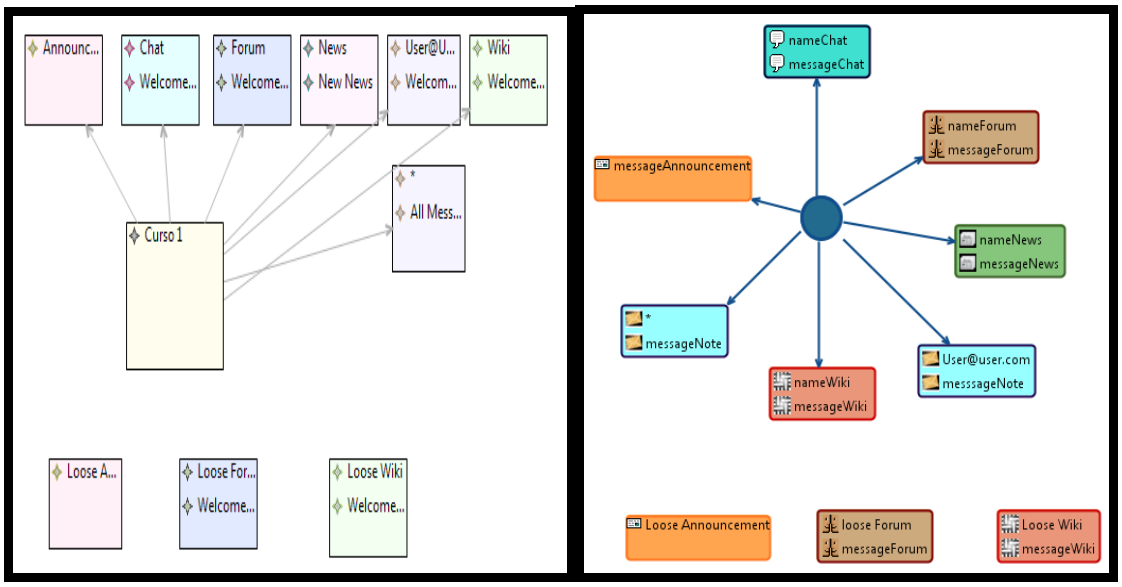


Figura 74 Creación de módulos con la herramienta DSL KiwiDSM v1.0 y v2.0 respectivamente.

En la anterior figura se pueden visualizar los módulos que están conectados con sus respectivas relación y los que no tiene conexión, estos últimos no serán desplegados en la plataforma LMS.

Observaciones a tener en cuenta en el modelado:

- Si se desea enviar una “Note” a todos los inscritos en un curso, en el campo “Send To” debe ir un *.

- Si se desea enviar una “Note” a un solo inscrito en un curso, en el campo “Send To” debe ir un el correo completo con el que la persona, como se inscribió en la plataforma virtual. Si este correo no corresponde con el de la persona el mensaje no será enviado.

- ***Recuerde diligenciar todos los campos en los nodos, pues aquellos campos en los cuales el usuario no ponga información serán tomados en blanco y se procesaran erradamente.***

- ***Todos los campos aparecen con información por defecto, pero esta información solo es de carácter orientativo, No implica información en los nodos.***

Guarde todos los cambios y envié al administrador de la aplicación el fichero generado con extensión .modellms, para este caso de ejemplo “default.modellms”.

Una vez reciba respuesta del Administrador, ingrese a la plataforma LMS (Moodle, Claroline o Atutor) con su perfil de profesor.

3.7.1.1 Cargue del módulo creado en Moodle

Una vez el Administrador reciba los ficheros enviado por el profesor, para este caso de ejemplo “default.modellms” y “default.modellms_diagram”:

1. Cambie la extensión del fichero “default.modellms” a “default.ecore”.

2. Abra eclipse y aplique las transformaciones MOFScrip que tiene el fichero “modellmsTransformationMoodle.m2t”.

3. Esta transformación generan dos ficheros “lib.php” y “mod_form.php” cópielas y reemplace las anteriores en la ruta en donde este el módulo “plantilladsl” previamente creado, que debe estar en “path_Moodle\mod\plantilladsl”.

4. Abra Moodle en modo Administrador y compile la plataforma nuevamente, para ello ingrese al navegador y digite “path_Moodle/admin/index.php”.

Con estos pasos ya el profesor podrá ver el módulo con todo lo que el creo en Moodle y lo podrá desplegar.

3.7.1.2 Cargue del módulo creado en Claroline

Una vez el Administrador reciba los ficheros enviado por el profesor, para este caso de ejemplo “default.modellms” y “default.modellms_diagram”:

1. Cambie la extensión del fichero “default.modellms” a “default.ecore”.

2. Abra eclipse y aplique las transformaciones MOFScrip que tiene el fichero “modellmsTransformationClaroline.m2t”.

3. Esta transformación generan dos ficheros “entry.php” y “plantillasdsl.lib.php”, comprímalas en un archivo que se llamara “UOPDSL2.zip” con la siguiente estructura:

- \entry.php

- \manifest.xml
- \lib\plantillasdsl.lib.php

4. Abra Claroline en modo Administrador y cargue el nuevo módulo, para ello valla a la opción “Administración de la plataforma” del menú, como lo muestra la siguiente figura.

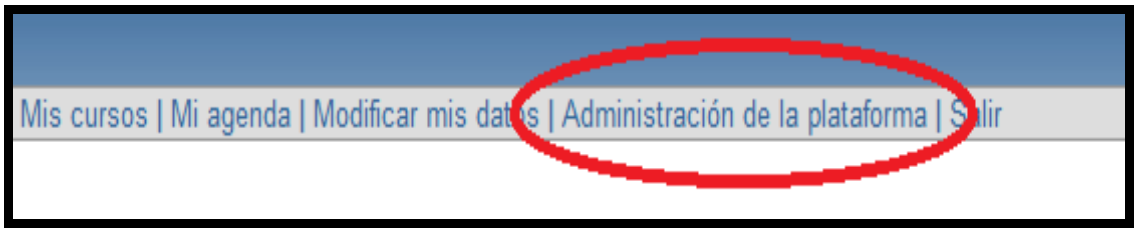


Figura 75 Administración de la plataforma Claroline

5. En el nuevo menú seleccione “Módulos”, como se muestra en la siguiente figura.

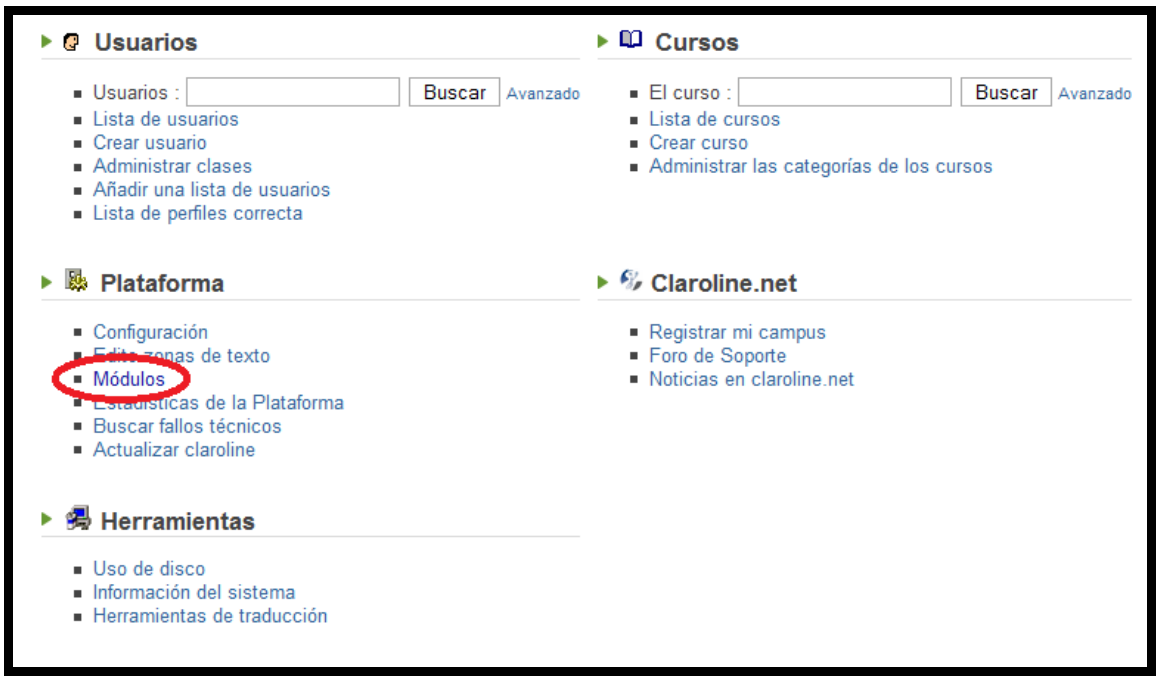


Figura 76 Ingreso a la administración de Módulos en Claroline.

6. En esta nueva ventana de clic sobre el vinculo “Instale un módulo”, luego en el botón examinar busque el archivo comprimido “UOPDSL2.zip” ábralo, de clic sobre el botón “Instale un módulo”, como se ve en la siguiente figura.

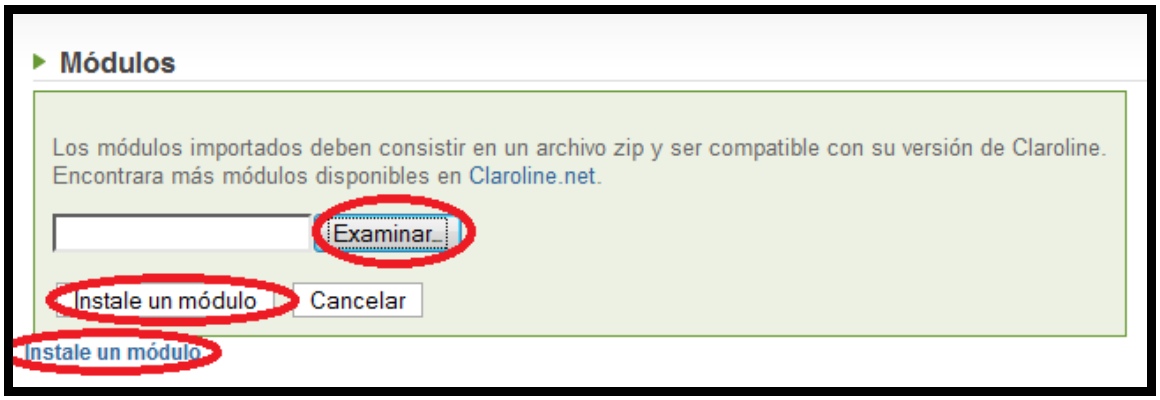


Figura 77 Cargue de un nuevo módulo en Claroline.

Si la instalación ha sido exitosa aparecerá un mensaje informado que el módulo ha sido instalado y este se desplegara en el listado de módulos disponibles, para que el profesor lo pueda visualizar debe activar el módulo para ello de clic sobre la opción activado del módulo  , como se ve en la siguiente figura.

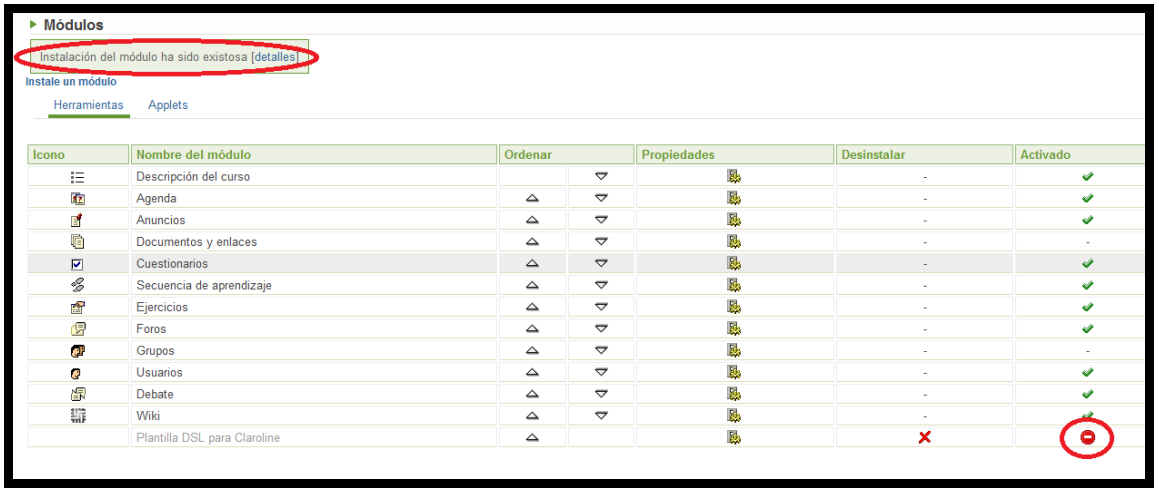


Figura 78 Activación de módulos en Claroline.

Con estos pasos ya el profesor podrá ver el módulo con todo lo que el creo en Claroline y lo podrá desplegar.

3.7.1.3 Cargue del módulo creado en ATutor

Una vez el Administrador reciba los ficheros enviado por el profesor, para este caso de ejemplo “default.modellms” y “default.modellms_diagram”:

1. Cambie la extensión del fichero “default.modellms” a “default.ecore”.

2. Abra eclipse y aplique las transformaciones MOFScrip que tiene el fichero “modellmsTransformationATutor.m2t”.

3. Esta transformación generará siete ficheros, es requerido ingresar las librerías php siguientes: “class.phpmailer.php” y “class.smtp.php” a la carpeta “lib” de forma manual. Al terminar el proceso anterior comprimir la carpeta plantilladsl.zip con la siguiente estructura:

- \index.php
- \module.php
- \module.sql
- \module.xml
- \module_install.php
- \module_uninstall.php
- \lib\dsllib.lib.php
- \lib\class.phpmailer.php
- \lib\class.smtp.php

4. Abra ATutor en modo Administrador y cargue el nuevo módulo, para ello valla a la opción “Modules”, “Install Modules”, seleccionar el fichero zip que se creó en el paso anterior e instalar, cumplir con los pasos requeridos para la instalación según el sistema operativo que se cuente, las instrucciones son vistas en el wizard de instalación de la plataforma, como se muestra en la siguiente figura.

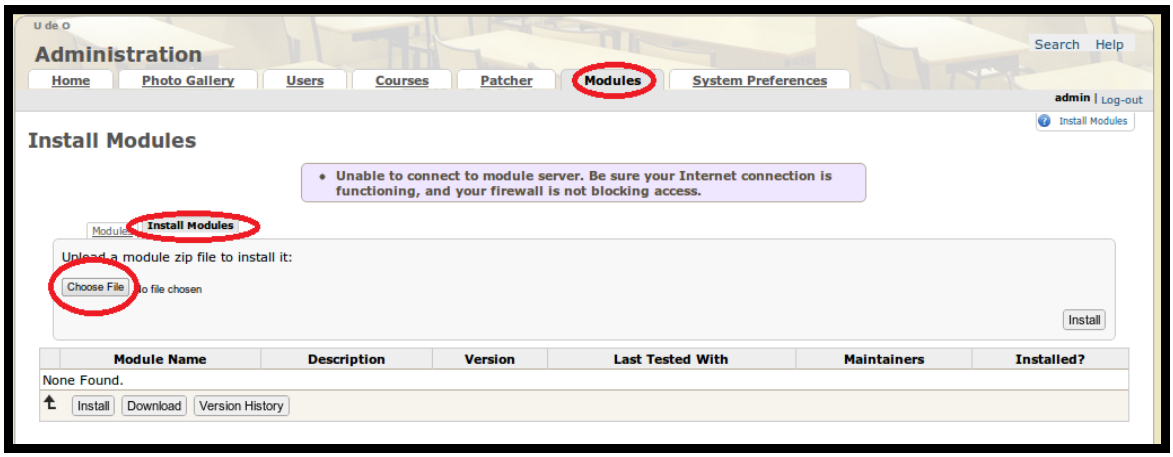


Figura 79 Cargue de un nuevo módulo en ATutor.

5. Al finalizar la instalación solo queda habilitar el módulo, para realizarlo solo se selecciona el módulo y se da clic en el botón “Enable”, para desinstalarlo solo clic en “Unisntall”, como se muestra en la siguiente figura.

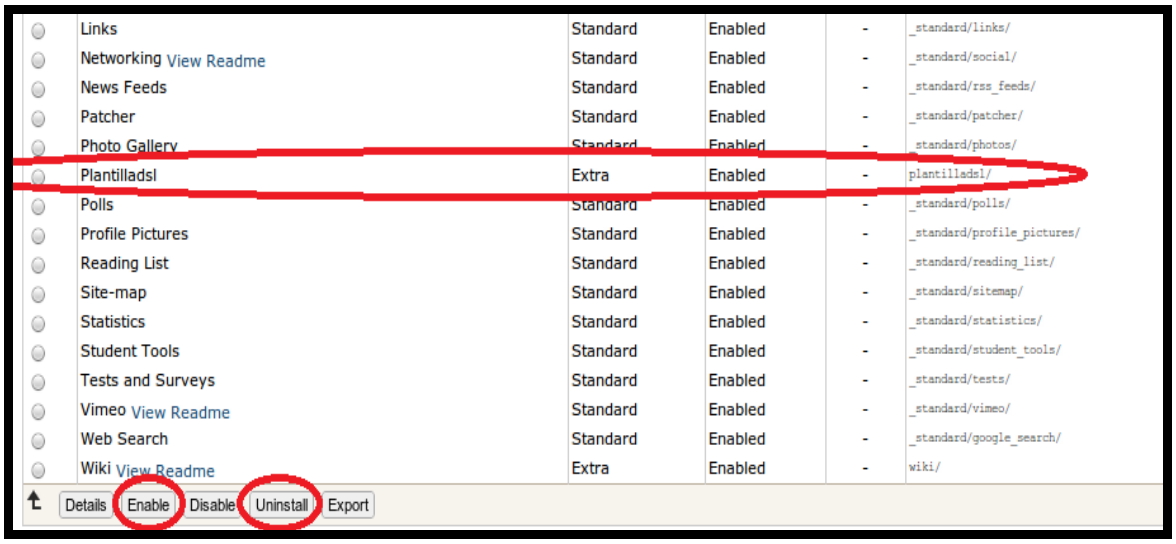


Figura 80 Activación de módulos en ATutor.

Con estos pasos ya el profesor podrá ver el módulo con todo lo que el creo en ATutor y lo podrá desplegar.

3.7.1.4 Despliegue del módulo creado en Moodle

Cambie a modo de edición. Ubíquese en cada uno de los temas y despliegue el ComboBox que dice “Add an activity”, seleccione la opción [[modulename]], como se ve en la siguiente figura.

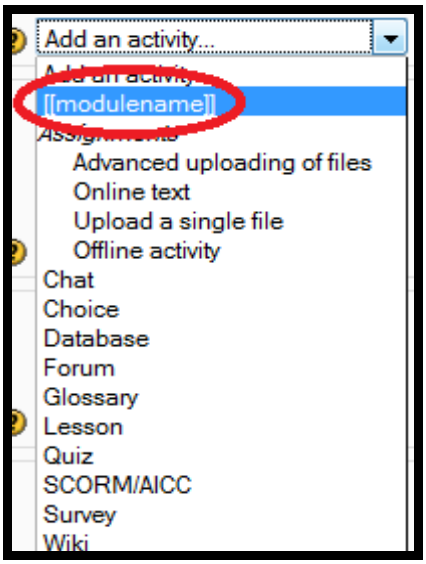


Figura 81 Adición del módulo creado con la herramienta KiwiDSM en Moodle.

Una vez seleccionada esta opción, los elementos modelados serán desplegados en la plataforma, Como se ven las siguientes figuras.

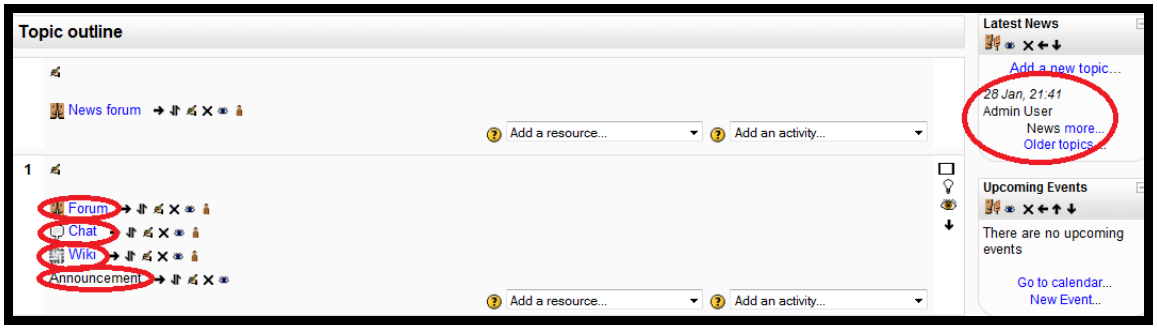


Figura 82 Despliegue de los módulos Forum Chat, Wiki, Announcement y News en Moodle, creados con la herramienta KiwiDSM.

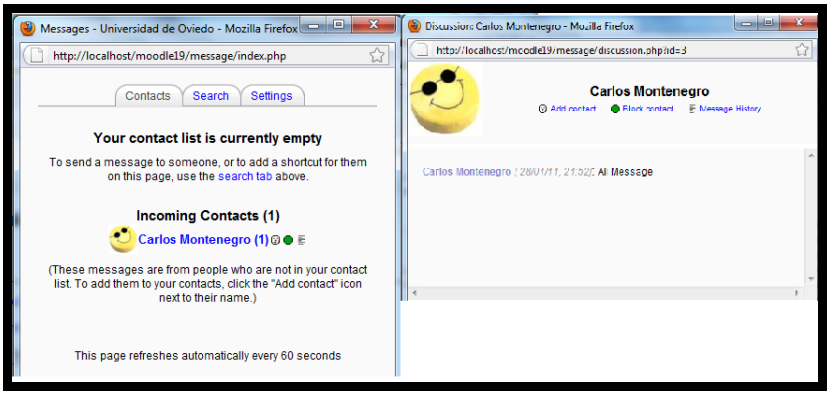


Figura 83 Despliegue del módulo Note en Moodle, creado con la herramienta KiwiDSM.

3.7.1.5 Despliegue del módulo creado en Claroline

Una vez reciba respuesta del Administrador, ingrese a Claroline con su perfil de profesor, luego ingrese al curso en el cual desea desplegar los módulos creados con la herramientas KiwiDSM, deberá aparecer una nueva opción en los módulos llamada “Plantilla DSL para Claroline”, para desplegar los módulos sencillamente de clic sobre esta nueva opción y los módulos se crearan acorde al modelo hecho con la herramienta KiwiDSM. Como lo muestran las siguientes figuras.

Figura 84 Adición del módulo creado con la herramienta KiwiDSM en Claroline.

Una vez seleccionada esta opción los elementos modelados serán desplegados en la plataforma, Como se ve a continuación.

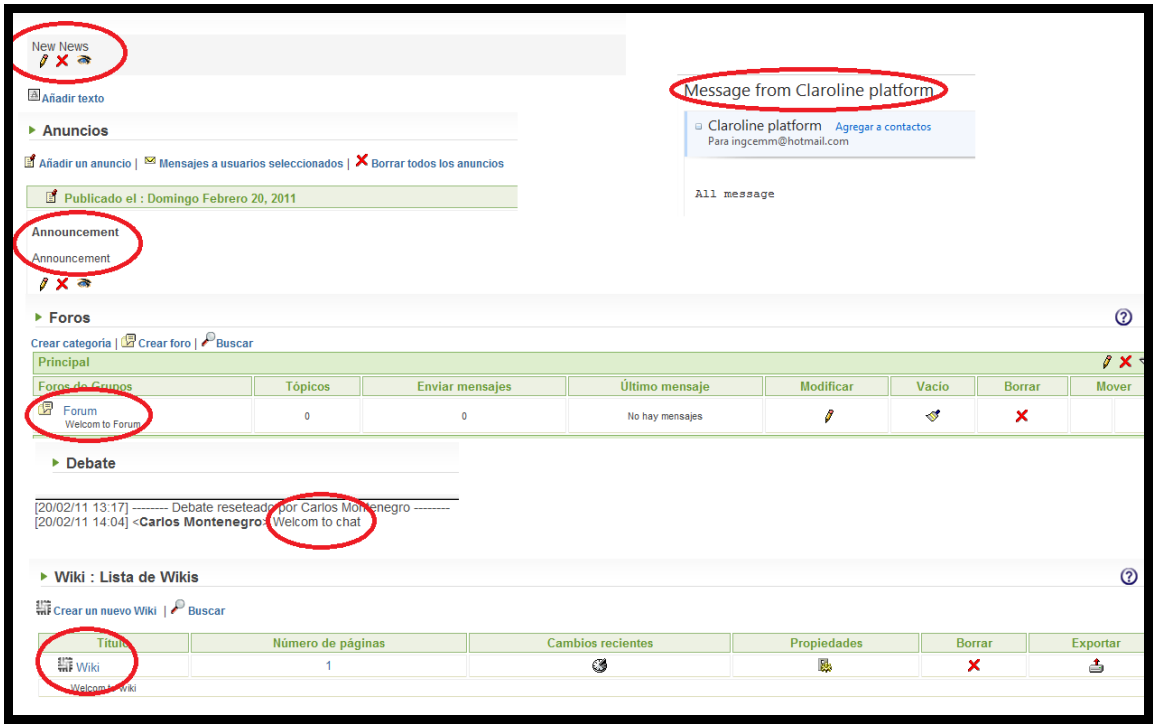


Figura 85 Despliegue de los módulos Forum Chat, Wiki, Announcement, News y Note en Claroline, creados con la herramienta KiwiDSM.

3.7.1.6 Despliegue del módulo creado en ATutor

Una vez reciba respuesta del Administrador, ingrese a ATutor con su perfil de profesor, luego ingrese al curso en el cual desea desplegar los módulos creados con la herramientas KiwiDSM, después ingresar a la pestaña “Manager” buscar en la serie de herramientas la opción llamada “Plantilladsl”, para desplegar los módulos sencillamente de clic sobre esta nueva opción.

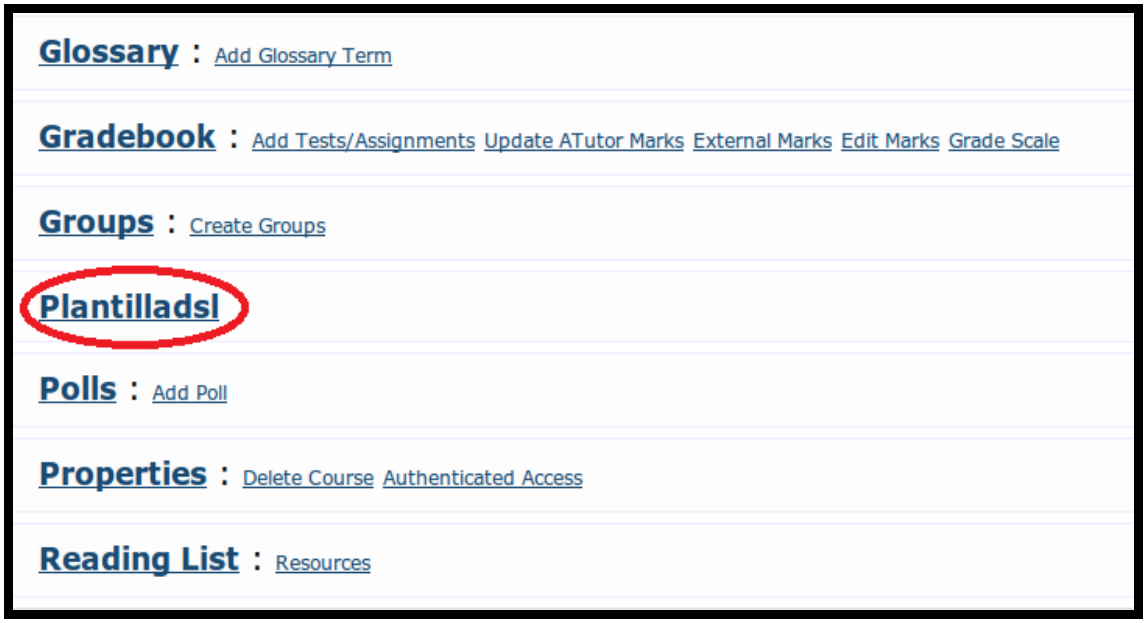


Figura 86 Adición del módulo creado con la herramienta DSL en ATutor.

Los módulos se crearan acorde al modelo hecho con la herramienta KiwiDSM. Como se muestra a continuación.

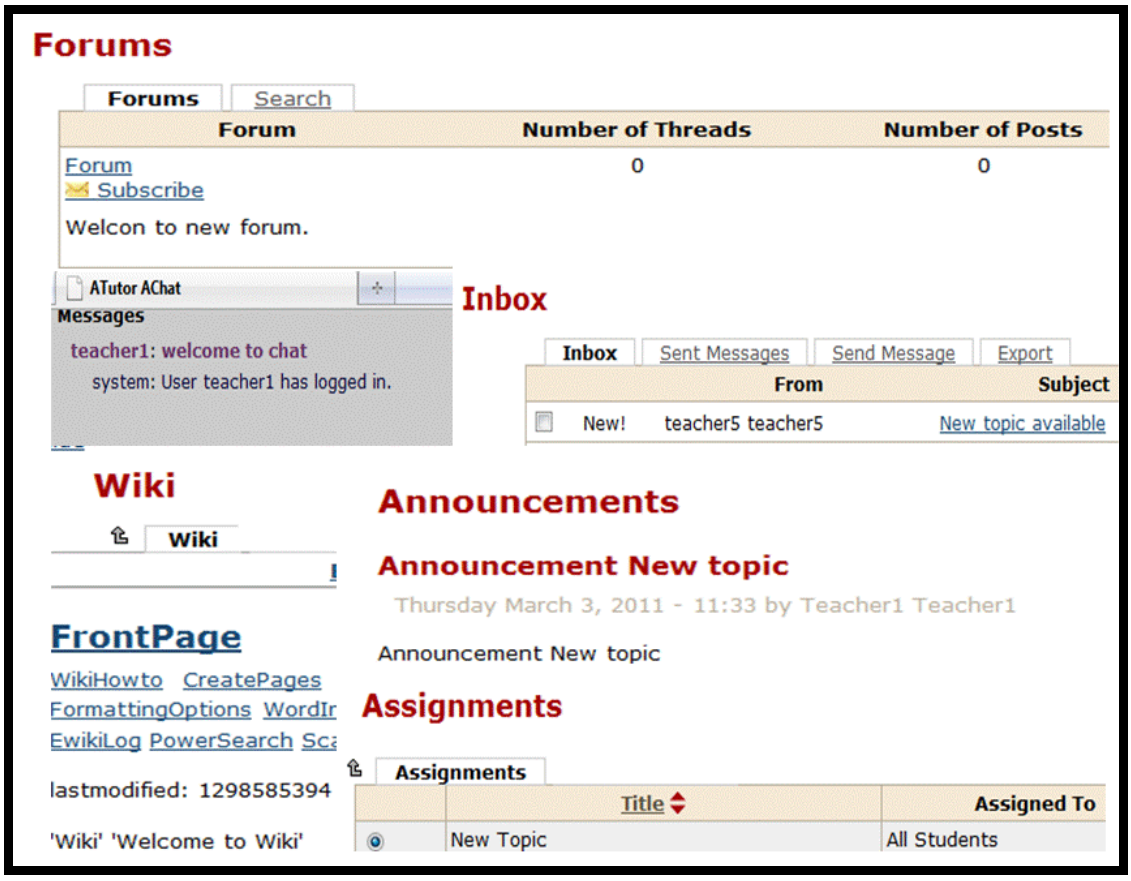


Figura 87 Despliegue de los módulos Forum Chat, Wiki, Announcement, News y Note en ATutor, creados con la herramienta KiwiDSM.

3.7.2 Generación y despliegue de módulos en Moodle

Aquí explicaremos como un usuario con rol de profesor ingresa a la plataforma y puede crear chats, foros, wikis, anuncios, noticias o notas.

1. Ingresar a la plataforma con su usuario y contraseña.
2. Seleccione el curso a trabajar.
3. Cambie la opción del curso a editable, para ello de clic sobre el botón “Turn editing on” que está ubicado en la parte superior derecha de la ventana, este botón debe cambiar a “Turn editing off”, ver la siguiente figura.

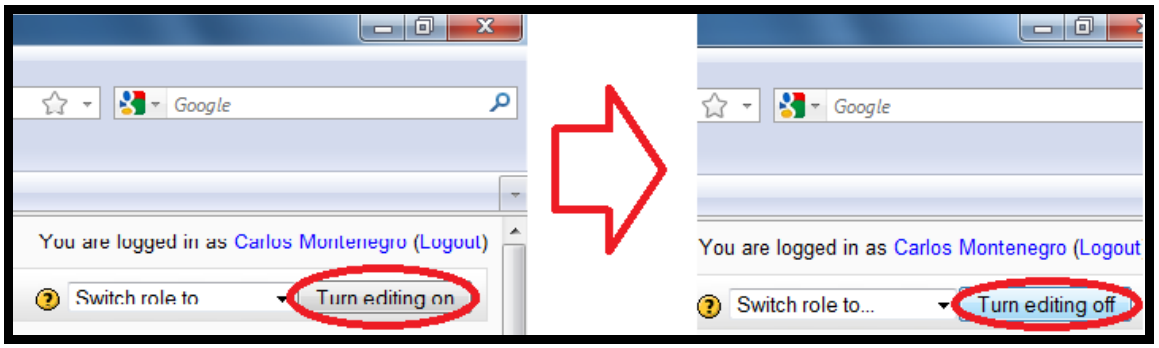


Figura 88 Cambiar a modo edición en Moodle.

4. Creación de Chat, Forum ó Wiki.

Despliegue el ComboBox que dice “Add an activity” ubicado en cada uno de los temas y seleccione la opción que desea agregar al tema del curso Chat, Forum ó Wiki, Como se muestra en la siguiente figura.

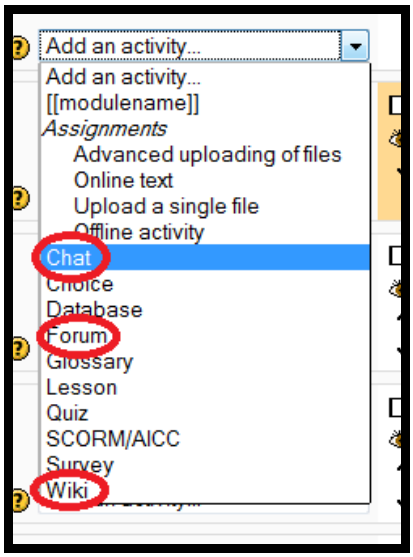


Figura 89 Adición de una actividad en Moodle.

a. Si usted selecciono Chat, vera un formulario como el de la siguiente figura, los campos marcados con rojo y * son obligatorios, llénelos y de clic sobre el botón “Save and return to course”, esto lo regresara nuevamente al curso y usted debe poder visualizar el chat creado, como lo muestra la figura de más abajo.

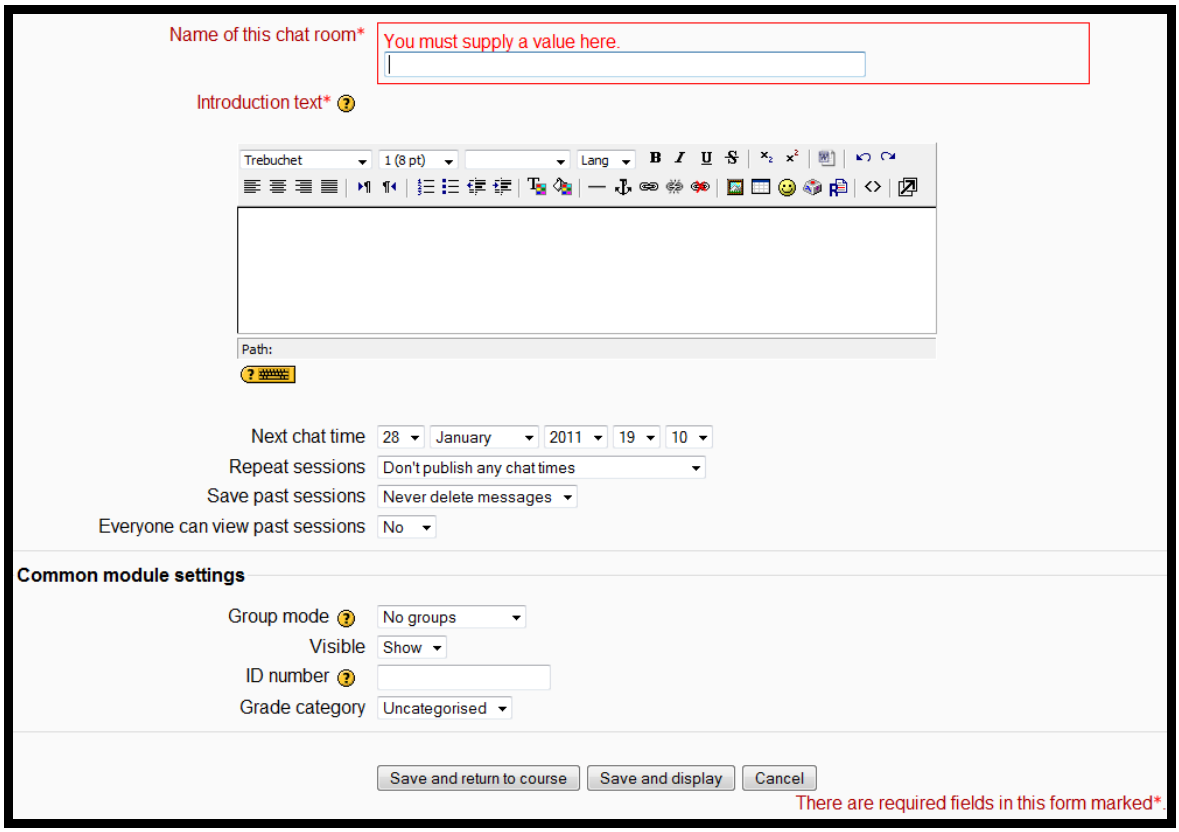


Figura 90 Formulario para la creación de un Chat en Moodle.

Figura 91 Chat desplegado en Moodle.

b. Si usted selecciono Forum, vera un formulario como el de la siguiente figura, los campos marcados con rojo y * son obligatorios, llénelos y de clic sobre el botón “Save and return to course”, esto lo regresara nuevamente al curso y usted debe poder visualizar el foro creado, como lo muestra la figura de más abajo.

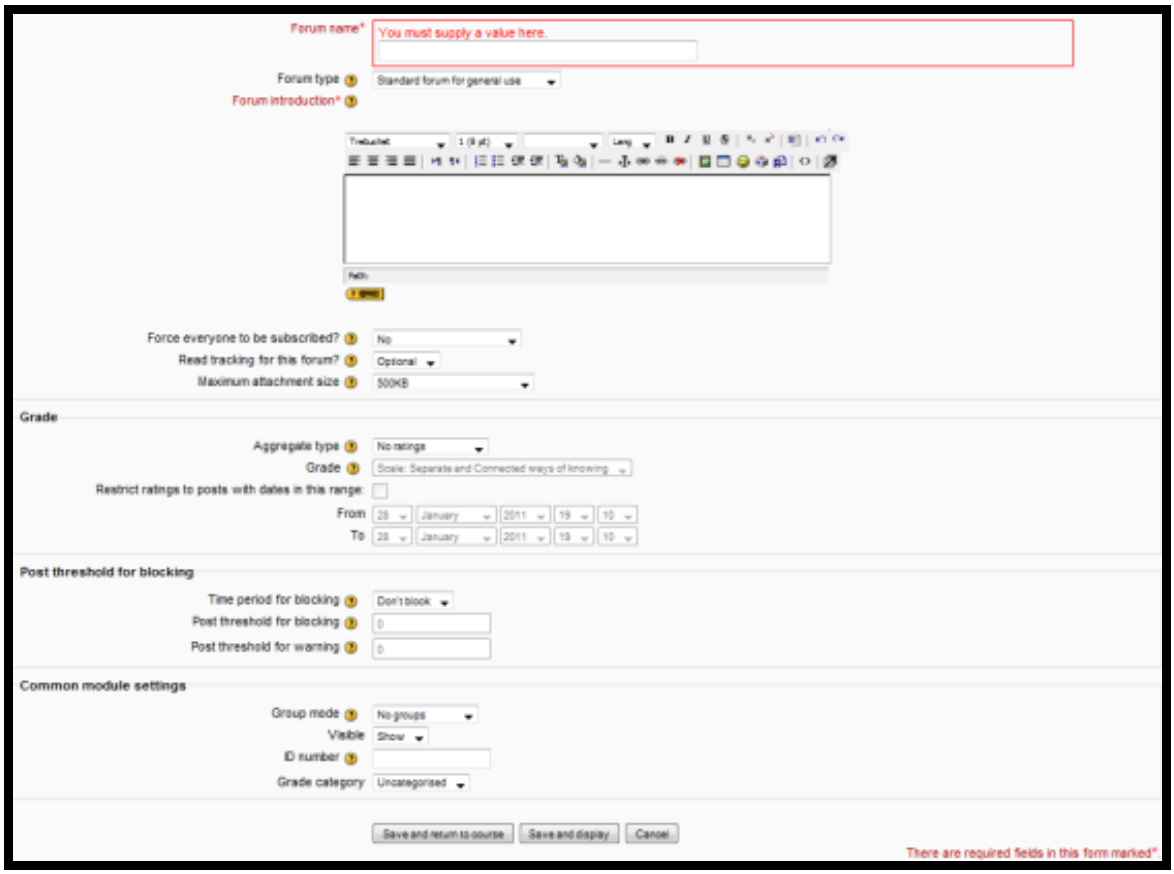


Figura 92 Formulario para la creación de un Foro en Moodle.

Figura 93 Foro desplegado en Moodle.

c. Si usted selecciono Wiki, vera un formulario como el de la siguiente figura, los campos marcados con rojo y * son obligatorios, llénelos y de clic sobre el botón “Save and return to course”, esto lo regresara nuevamente al curso y usted debe poder visualizar el wiki creado, como lo muestra la figura de más abajo.

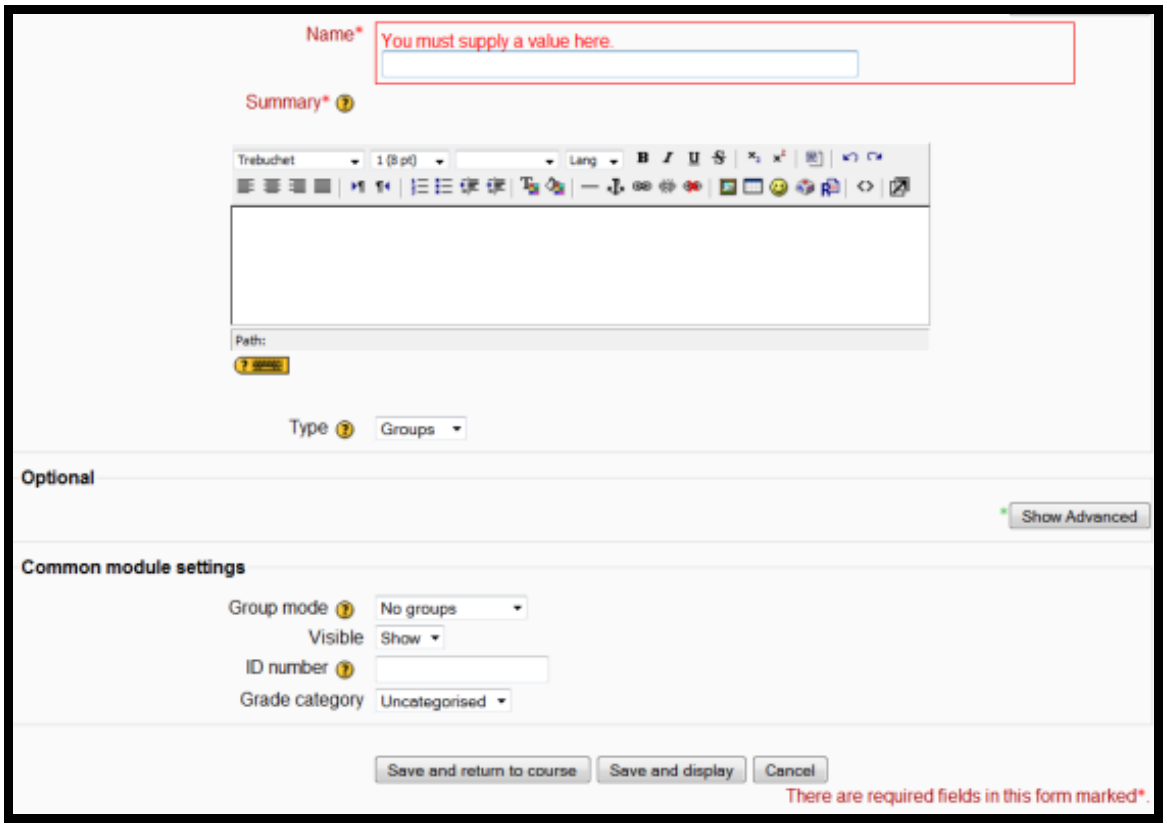


Figura 94 Formulario para la creación de una wiki en Moodle.

Figura 95 Wiki desplegada en Moodle.

5. Creación de Announcement

Despliegue el ComboBox que dice “Add a resource” ubicado en cada uno de los temas y selecciones la opción “Insert a label” como se ve en la siguiente figura.

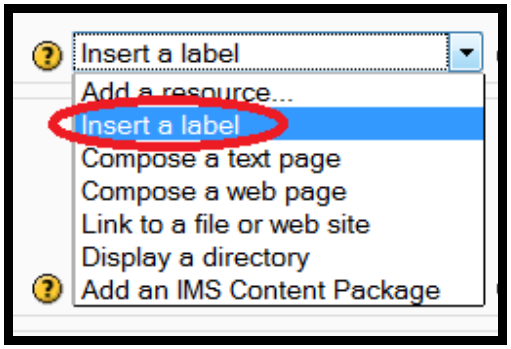


Figura 96 Adición de un recurso anuncio en Moodle.

Se mostrara un formulario como el de la siguiente figura, los campos marcados con rojo y * son obligatorios, llénelos y de clic sobre el botón “Save and return to course”, esto lo regresara nuevamente al curso y usted debe poder visualizar el Announcement creado, como lo muestra la figura de más abajo.

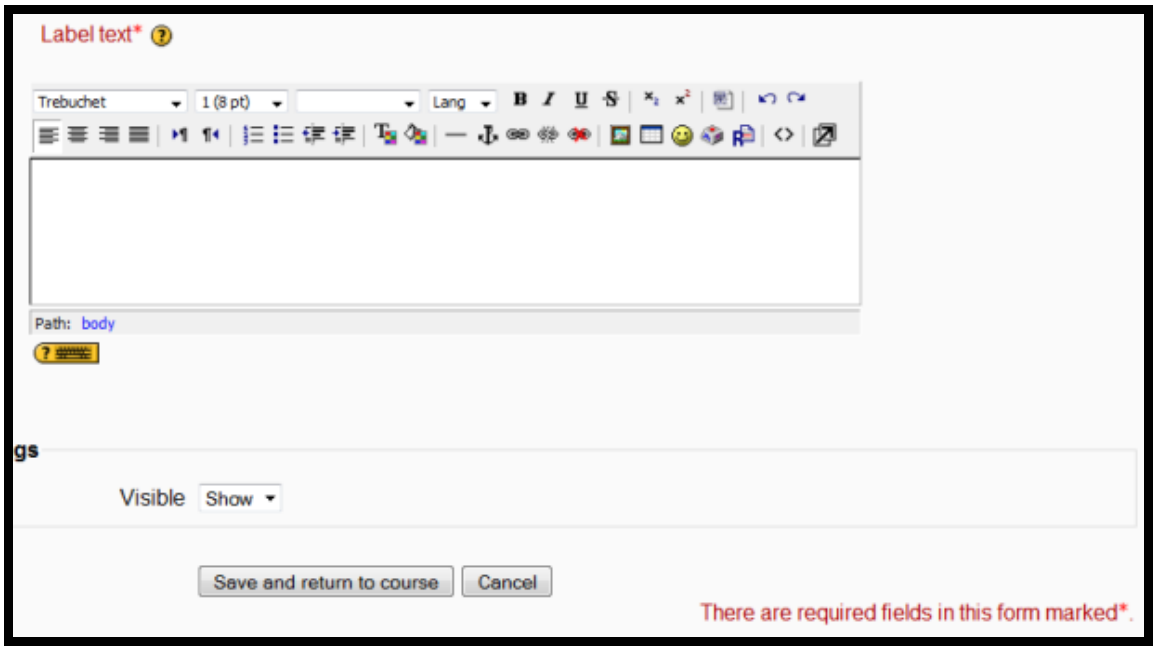


Figura 97 Formulario para la creación de un anuncio en Moodle.

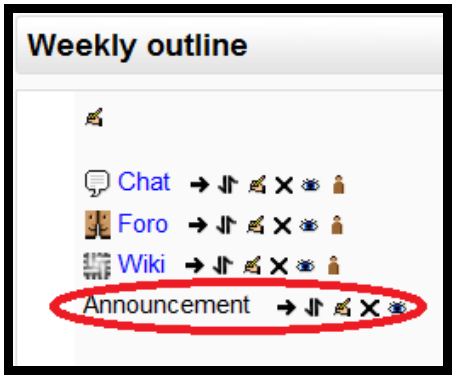


Figura 98 Anuncio desplegado en Moodle.

6. Creación de News.

En la parte superior derecha de la venta existe Widget con el título “Latest News”, de clic sobre “Add new topic …”, como se ve en la siguiente figura.

Figura 99 Adición de una noticia en Moodle.

Se mostrara un formulario como el de la siguiente figura, los campos marcados con rojo y * son obligatorios, llénelos y de clic sobre el botón “Post Forum”, espere unos segundos, esto lo regresara nuevamente al curso y usted debe poder visualizar la News creada, como lo muestra la figura de más abajo.

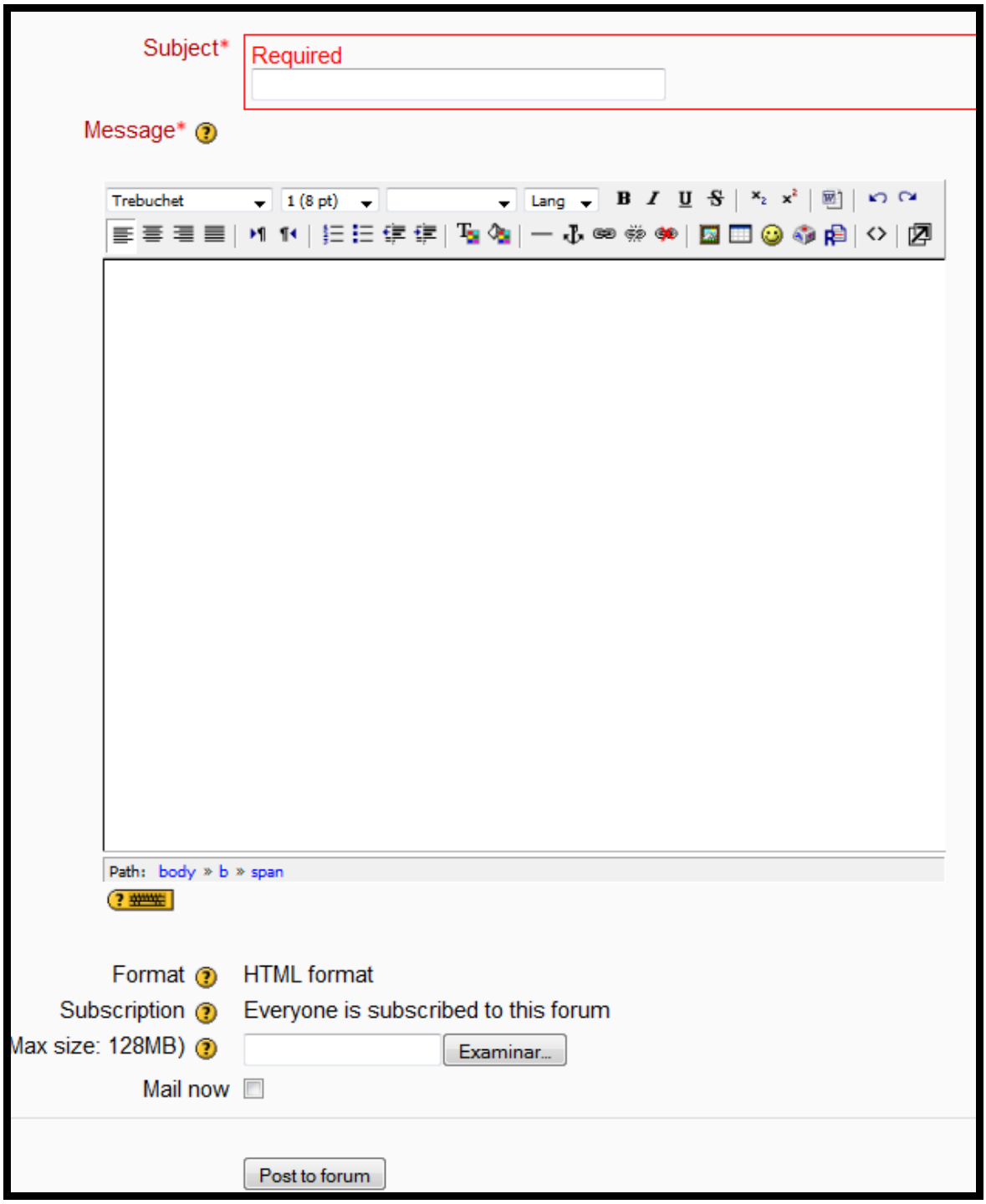


Figura 100 Formulario para la creación de una noticia en Moodle.

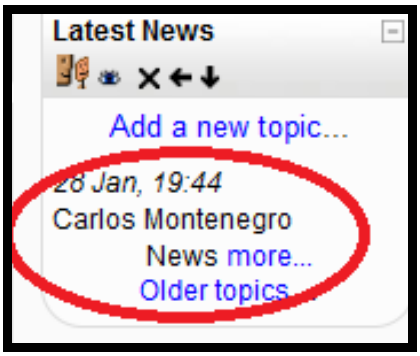


Figura 101 Noticia desplegada en Moodle.

7. Envió de Notes.

Las Notes en Moodle son llamadas messages, y para poder enviar uno, usted debe ingresar a la opción “Participants” del Wiget “People”, ubicado en la parte superior derecha de la ventana, como se ve en la siguiente figura.

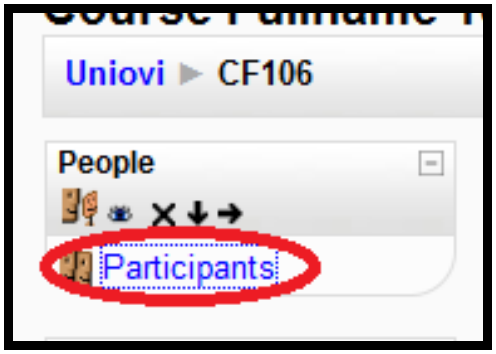


Figura 102 Selecciona de los inscritos en un curso para enviar una nota en Moodle.

Se mostrara un formulario con las personas inscritas en esta asignatura, usted deberá seleccionar a la persona que desea enviar el mensaje haciendo clic sobre su nombre, como lo muestra la figura siguiente.

Figura 103 Selecciona de un usuario para enviar una nota en Moodle.

En la pantalla que muestra la información de la persona y existe un botón en la parte inferior llamado “Send message”, de clic sobre él, como se ve en la próxima figura.

Figura 104 Adicionar una nota al usuario seleccionado en Moodle.

Y aparecerá una venta emergente, en donde se escribe el mensaje a enviar, como se ve en esta figura.

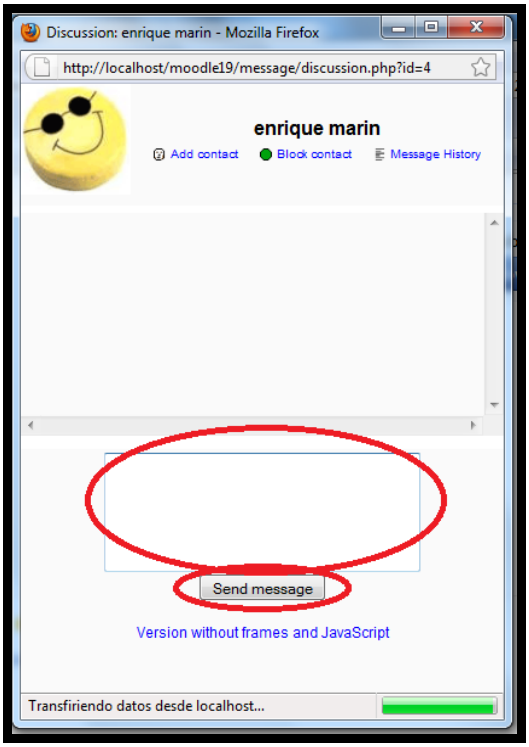


Figura 105 Formulario para el envió de una nota en Moodle.

Para enviar el mensaje se hace clic sobre el botón “Send message” y finalmente aparecerá en la parte superior de la venta un mensaje confirmando su envió, como lo muestra la figura de abajo.

Figura 106 Nota desplegada en Moodle.

3.7.3 Generación y despliegue de módulos en Claroline

Aquí explicaremos como un usuario con rol de profesor ingresa a la plataforma y puede crear chats, foros, wikis, anuncios, noticias o notas.

1. Ingresar a la plataforma con su usuario y contraseña.
2. Seleccione el curso a trabajar, como se ve en la siguiente figura.

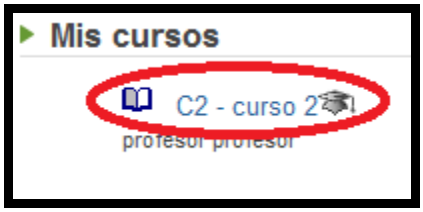


Figura 107 Lista de cursos en Claroline.

3. Existen dos modos de vista, usted debe seleccionar el modo de vista “Responsable de curso”, ya que esta le permite editar la información, ver la siguiente figura.

Figura 108 Modos de vista en Claroline.

4. Adición de información.

Para poder agregar la información usted debe seleccionar el módulo al cual desea agregar la información; Anuncio para Announcement y Note, Foro para Forum, Debate para Chat, Wiki para wiki o Añadir texto para News, como se ve en la siguiente figura.

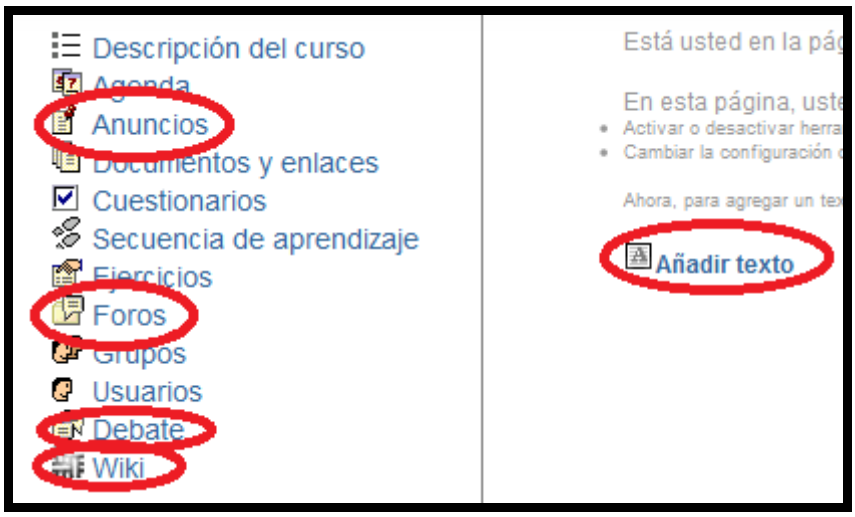


Figura 109 Módulos disponibles en Claroline.

a. Si usted selecciono Anuncio, aparecerá una vista con los anuncios que tiene el curso, allí pude seleccionar: Añadir un anuncio para publicar un nuevo anuncio (Announcement) o Mensajes a usuarios seleccionados para enviar un mensaje (Note) a los usuarios inscritos en el curso. Como lo muestra la siguiente figura.

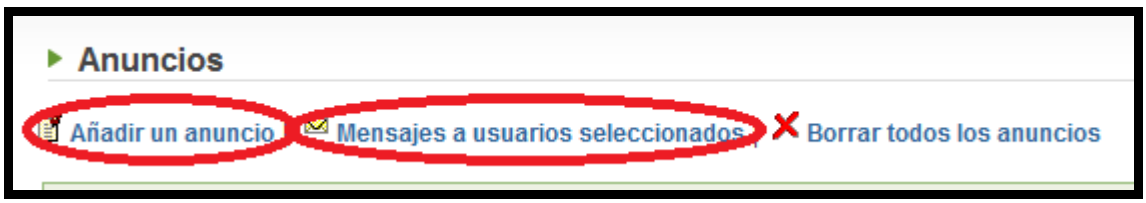


Figura 110 Opciones para un anuncio en Claroline.

i. Si usted selecciono Añadir un anuncio, aparecerá un formulario como el de la Figura siguiente, en este formulario se deben diligenciar los campos Titulo y Contenido, finalmente para publicar el anuncio se da clic sobre el botón validar, esto lo regresara nuevamente a la vista de Anuncios y vera su anuncio publicado en la plataforma, siguiente figura.

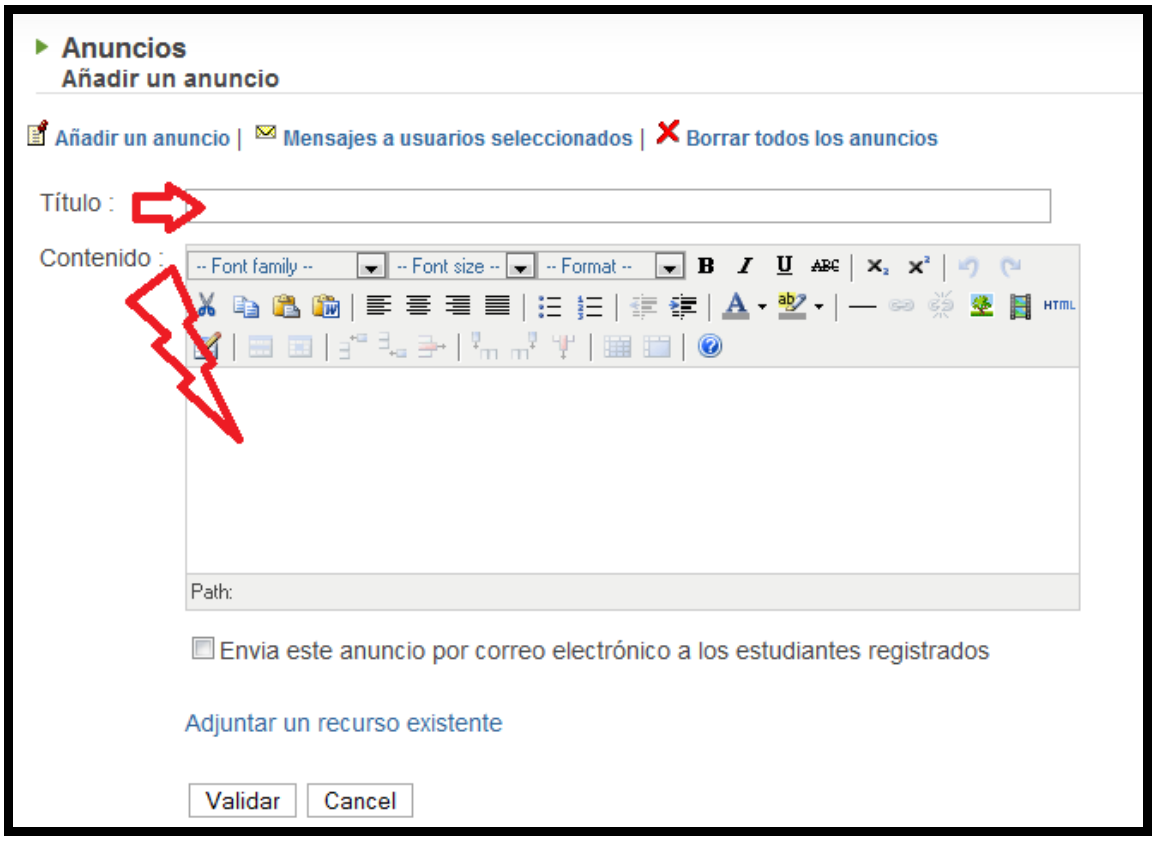


Figura 111 Formulario para la creación de un anuncio en Claroline.

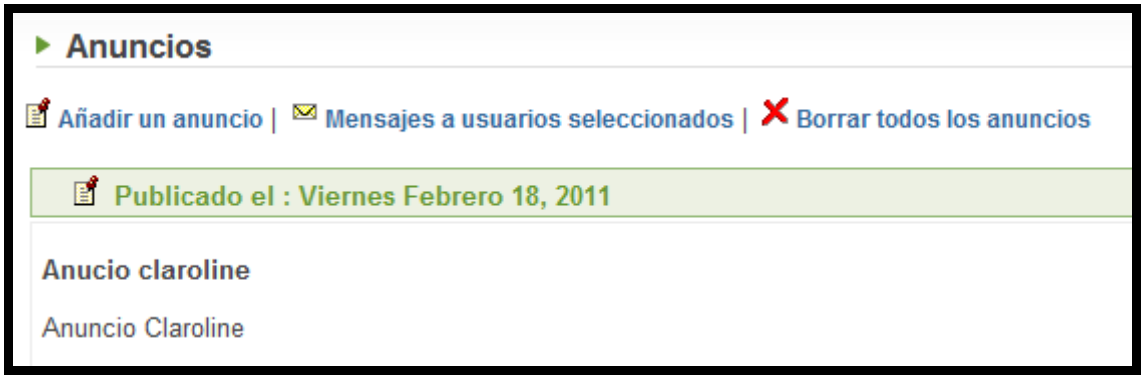


Figura 112 Anuncio desplegado en Claroline.

ii. Si usted selecciono Mensajes a usuarios seleccionados, aparecerá una vista como la de la siguiente figura, en esta vista se deben seleccionar los usuarios a los cuales se les va a enviar el mensaje de la Lista de usuarios y pasarlos a la lista de Usuario seleccionado, en el campo Anuncio se escribirá el mensaje a enviar y finalmente se da clic sobre el botón Asunto para enviar el mensaje, como se muestra en la siguiente figura. (Observación: El mensaje (Note) es enviado al correo de la persona seleccionada, por este motivo el servidor de correo saliente SMTP debe estar configurado en la plataforma). Una vez el mensaje es enviado aparecerá un mensaje de confirmación como lo muestra la figura siguiente.

Figura 113 Formulario para en el envió de Notes en Claroline.

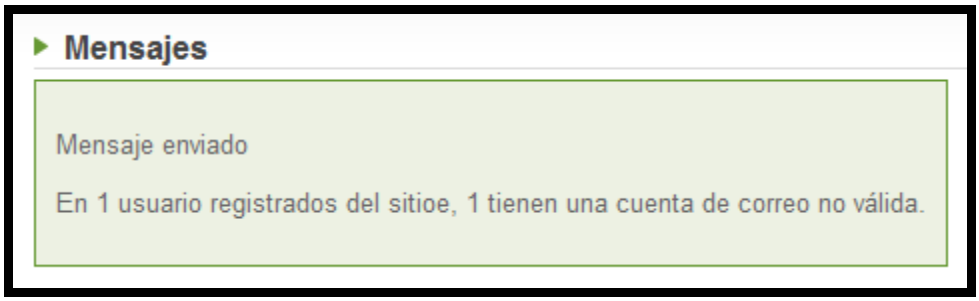


Figura 114 Confirmación de Note enviada en Claroline.

b. Si usted selecciono Foro, aparecerá una vista con los foros que tiene el curso, para crear un nuevo foro, debe dar clic sobre Crear foro como se muestra en la siguiente figura, aparecerá un nuevo formulario a diligenciar con el Nombre y la Descripción del foro, figura siguiente. Se llenan estos datos y se da clic en Validar y lo regresara nuevamente a la vista de foros con un mensaje que confirma la creación del foro.

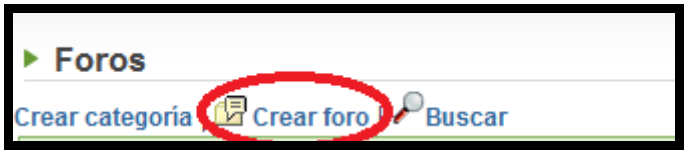


Figura 115 Creación de un nuevo foro en Claroline.

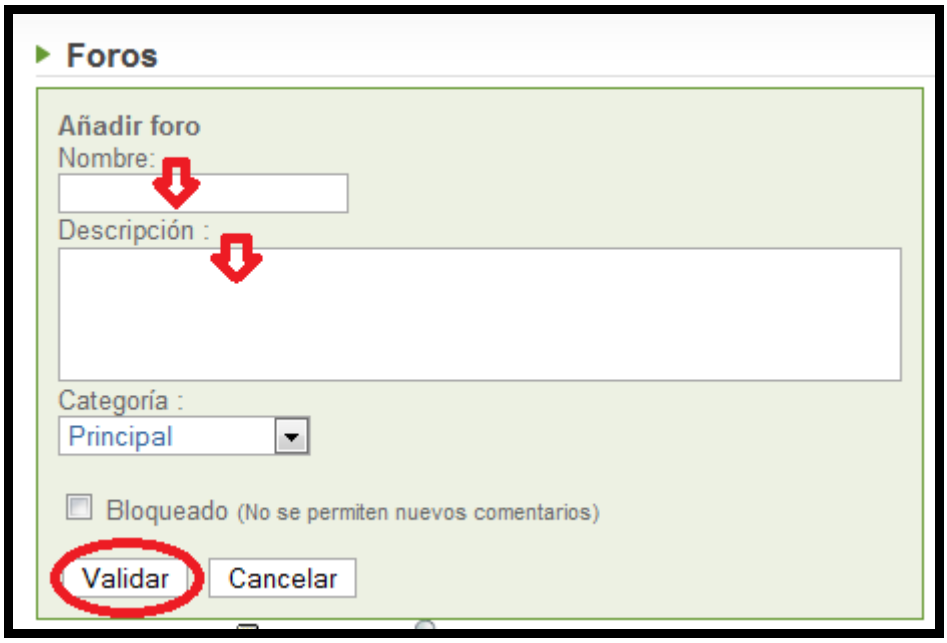


Figura 116 Formulario para la creación de un nuevo Foro en Claroline.

c. Si usted selecciono Debate (chat), aparecerá una vista que contiene la sala de debate del curso (chat), con los aportes que los usuarios han

realizado, para ingresar un nuevo aporte, se debe escribir en el campo de texto ubicado en la parte inferior y dar clic en >> para enviar el mensaje al chat, como se muestra en la siguiente figura. Los aportes se mostraran en cuadro de texto central de la sala de debates, indicando la fecha, hora, nombre del usuario y texto que se agrego, como se muestra en la figura siguiente.

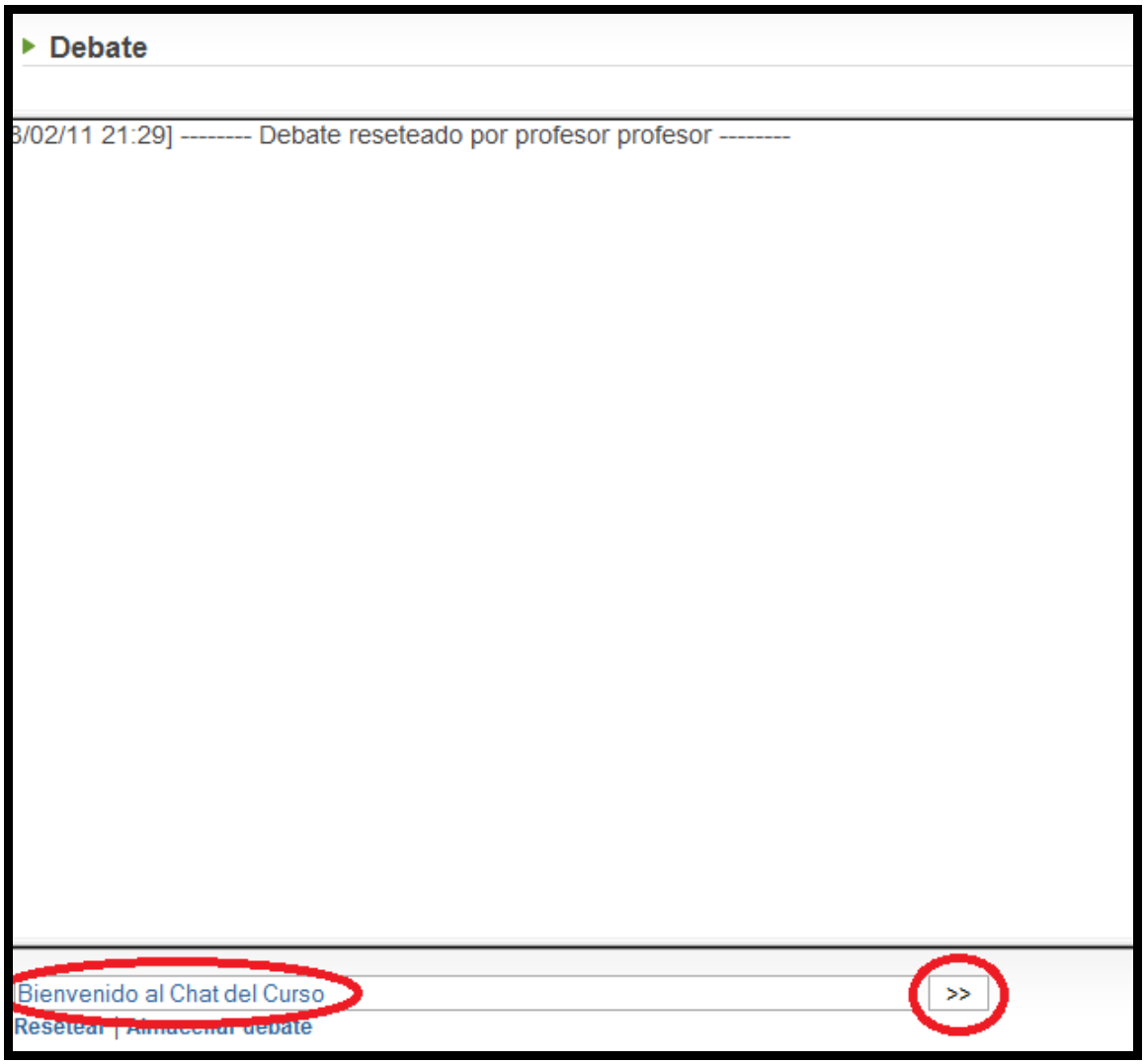


Figura 117 Formulario para ingresar datos al chat de un curso en Claroline.

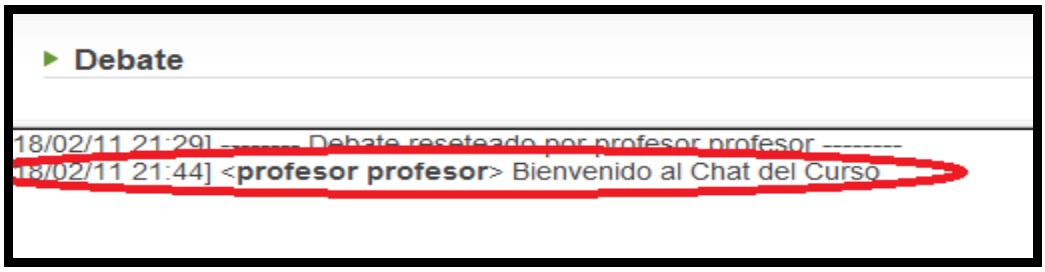


Figura 118 Información desplegada en el chat de un curso en Claroline.

d. Si usted selecciono Wiki, aparecerá una vista con las Wikis que tiene el curso, para crear una nueva Wiki, debe dar clic sobre Crear un nuevo Wiki como se muestra en la siguiente figura, aparecerá un nuevo formulario a diligenciar con el Titulo y la Descripción de la Wiki, figura siguiente. Se llenan estos datos y se da clic en Validar, esto lo regresara nuevamente a la vista de Wiki y podrá visualizar la nueva Wiki creada.

Figura 119 Creación de un nuevo wiki en Claroline.

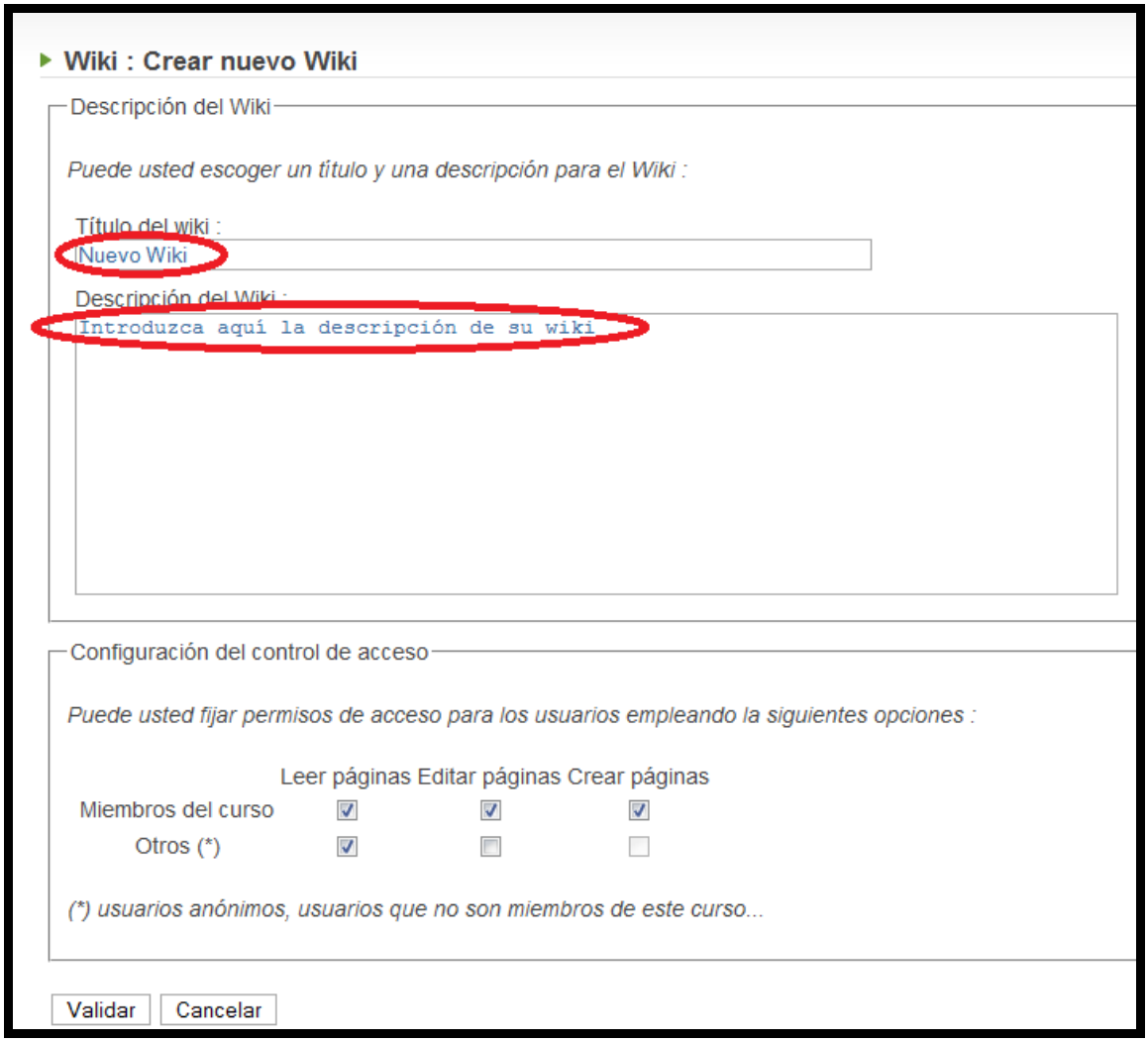


Figura 120 Formulario para la creación de un nuevo Foro en Claroline.

e. Si usted selecciono Anadir texto (news), aparecerá una vista con un editor, para ingresar el texto de la noticia (news), como lo muestra la figura siguiente. Para crear una nueva news, debe ingresar el texto y dar clic sobre Validar, esto lo regresara nuevamente a la vista de principal del curso y podrá visualizar la nueva noticia creada, figura siguiente.

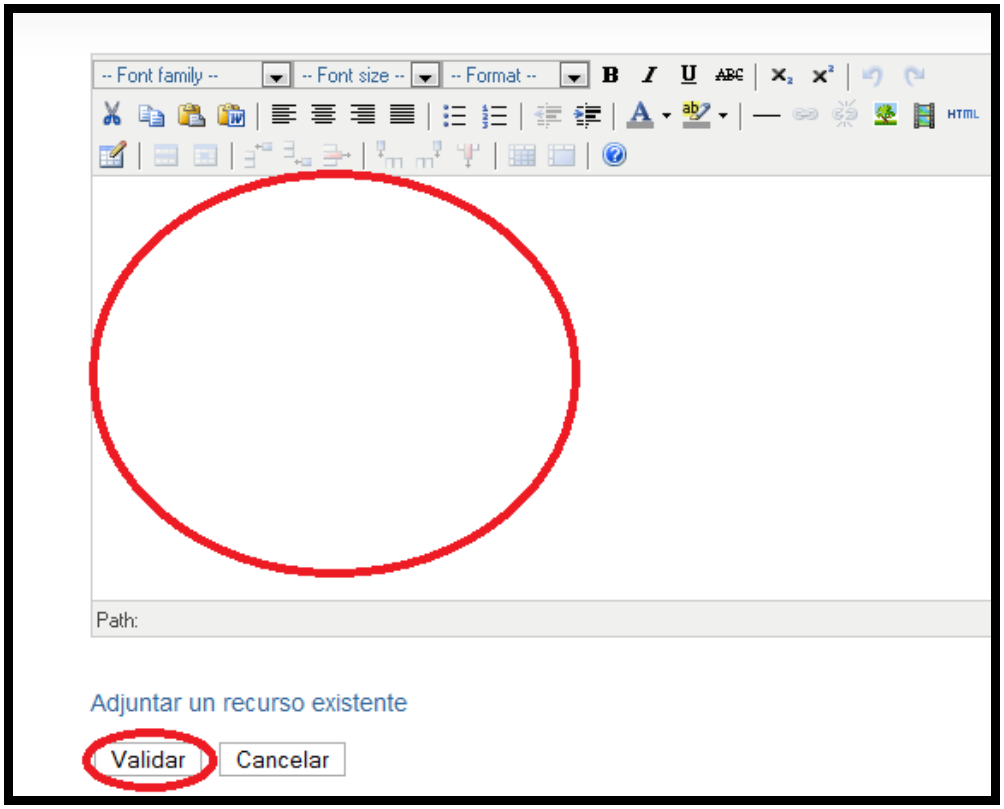


Figura 121 Formulario para la creación de una nueva news en Claroline.

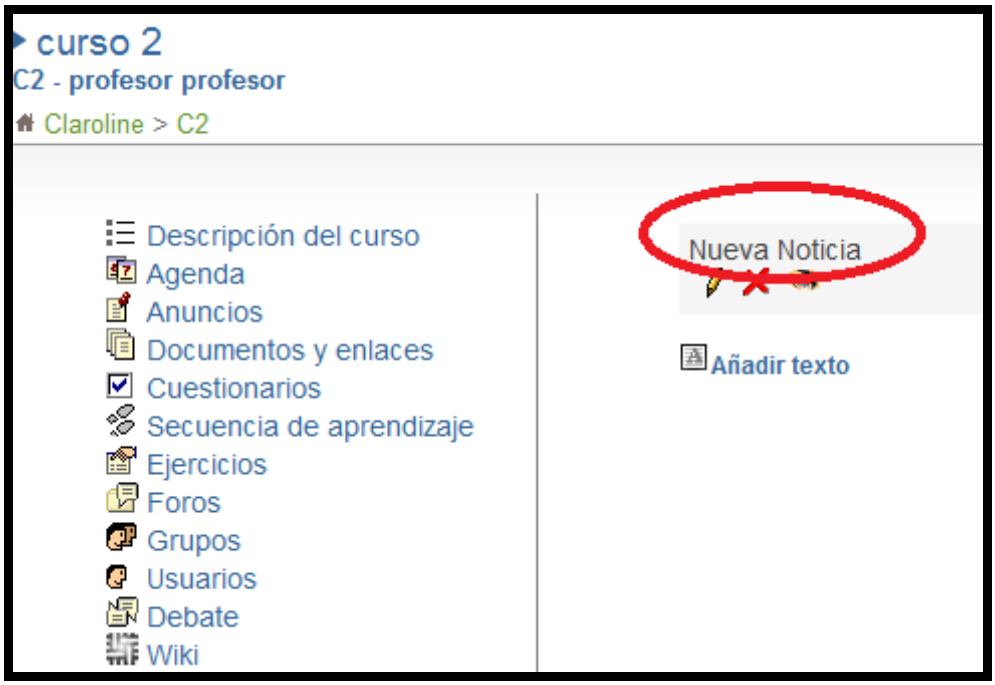


Figura 122 Publicación de una nueva news en Claroline.

3.7.4 Generación y despliegue de módulos en ATutor

Aquí explicaremos como un usuario con rol de profesor ingresa a la plataforma y puede crear chats, foros, wikis, anuncios, noticias o notas.

1. Ingresar a la plataforma con su usuario y contraseña.
2. Seleccione el curso a trabajar, como se muestra a continuación.

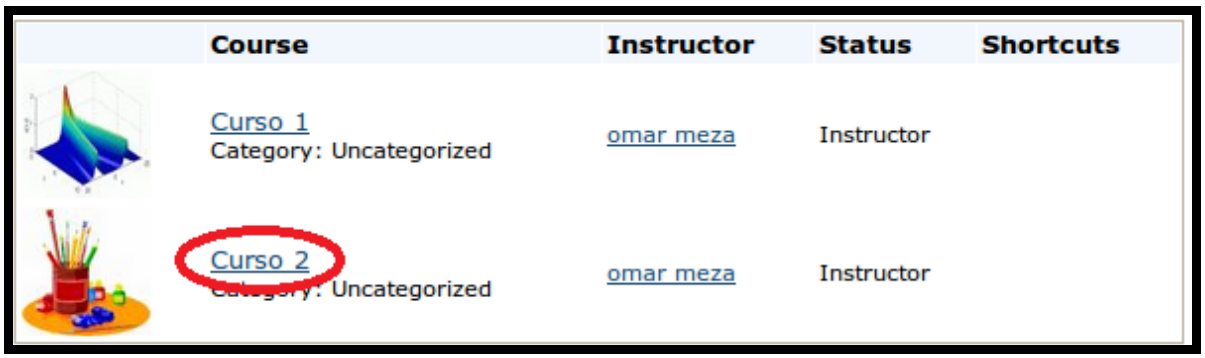


Figura 123 Selección de un curso ATutor.

3. Una vez ingresado al curso, selecciona la pestaña “Manage” para ingresar a las herramientas del mismo, como se ve a continuación.

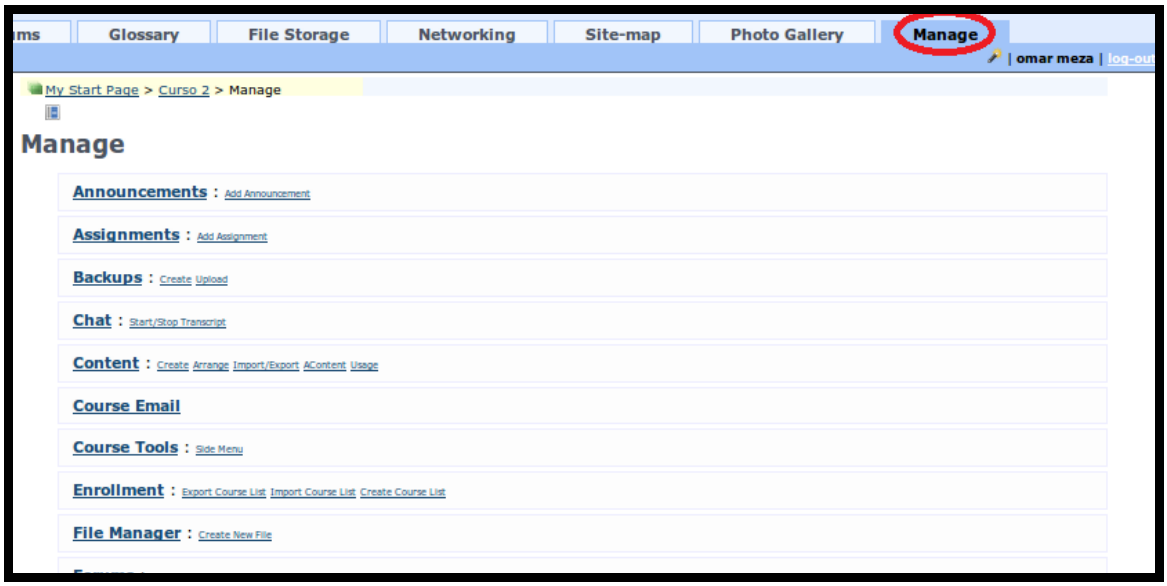


Figura 124 Ingreso a la administración de módulos en ATutor.

4. Adición de información.

Para poder agregar la información usted debe seleccionar la herramienta que desea, como se mostró en la figura anterior. A continuación se muestra una tabla comparativa de opciones para ingresar los datos requeridos:

Tabla 14 comparativa de opciones para ingresar los datos requeridos en Atutor.

Concepto	Herramienta ATutor	Operación
Anuncio	Announcements	Add Announcements
Noticia	Assignments	Add Assignment
Foro	Forums	Create Forum
Chat	Chat	Enter the Chat
Wiki	Wiki	EditThisPage
Note Todos	Course Email	Send
Note	MyContacts	1. Search People 2. Send Message

a. Al seleccionar ***Announcement*** aparecerá una vista con los anuncios que tiene el curso. Para agregar un anuncio es necesario regresar a la pestaña “Manage” como ya se mencionó, se hace clic en la opción “Add Announcement” ingresando el título y descripción como se muestra en la figura siguiente, una vez concluido este proceso el anuncio se verá reflejado en la página principal del curso, como se ve a continuación.

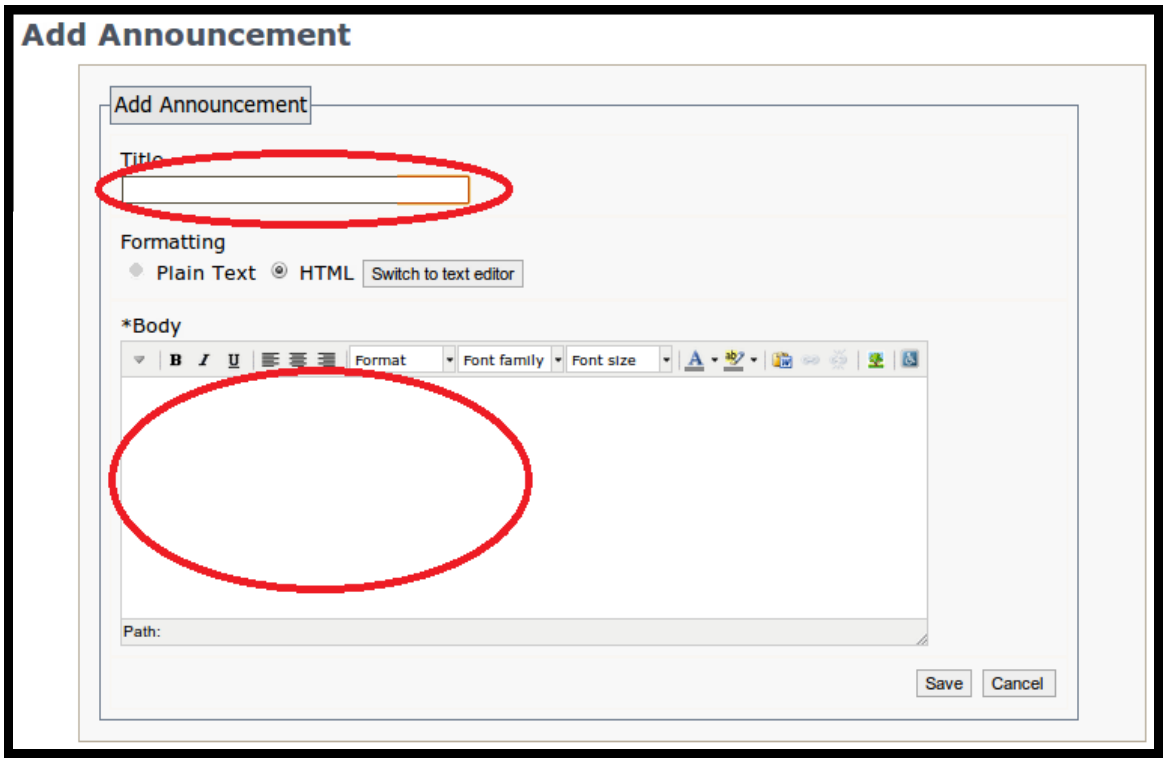


Figura 125 Formulario para la creación de un nuevo announcement en ATutor.

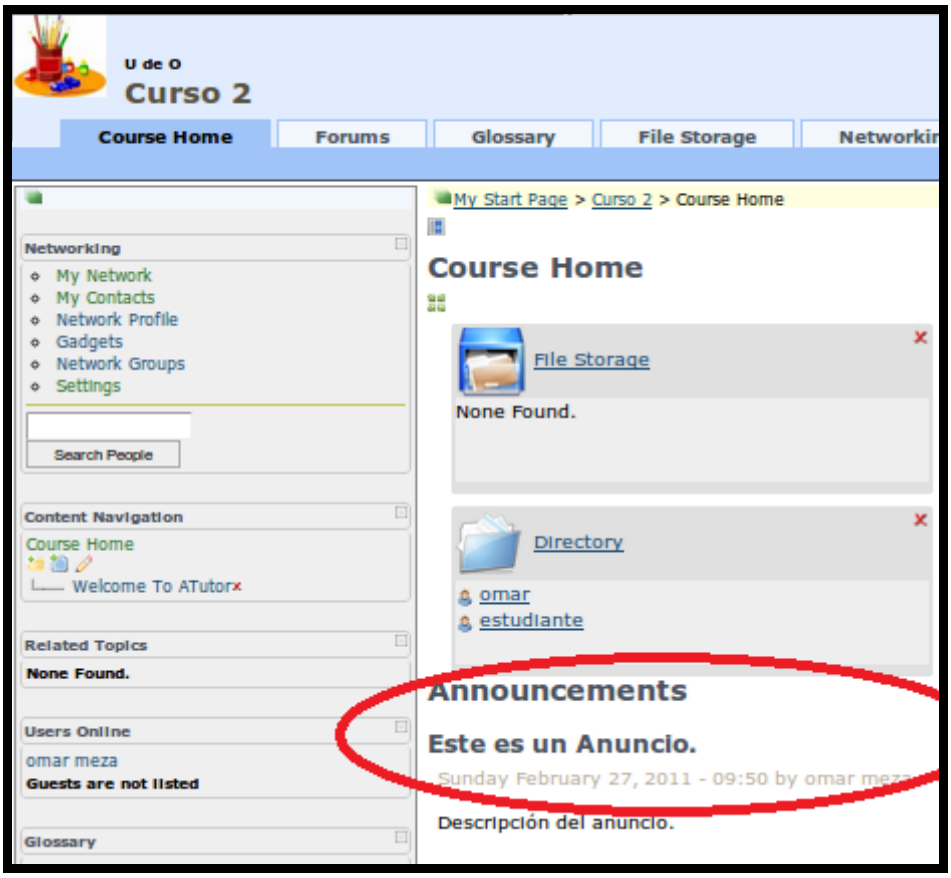


Figura 126 Publicación de un nuevo announcement en ATutor.

b. Al seleccionar ***Assignments*** aparecerá una vista con las noticias para los alumnos pertenecientes al curso. Para agregar una noticia es necesario regresar a la pestaña “Manage”, clic en “Add Assignment” e ingresar el título de la noticia, especificar si es para todos los alumnos o solo grupos específicos, en este caso se aplica para todos los alumnos, véase la siguiente figura.

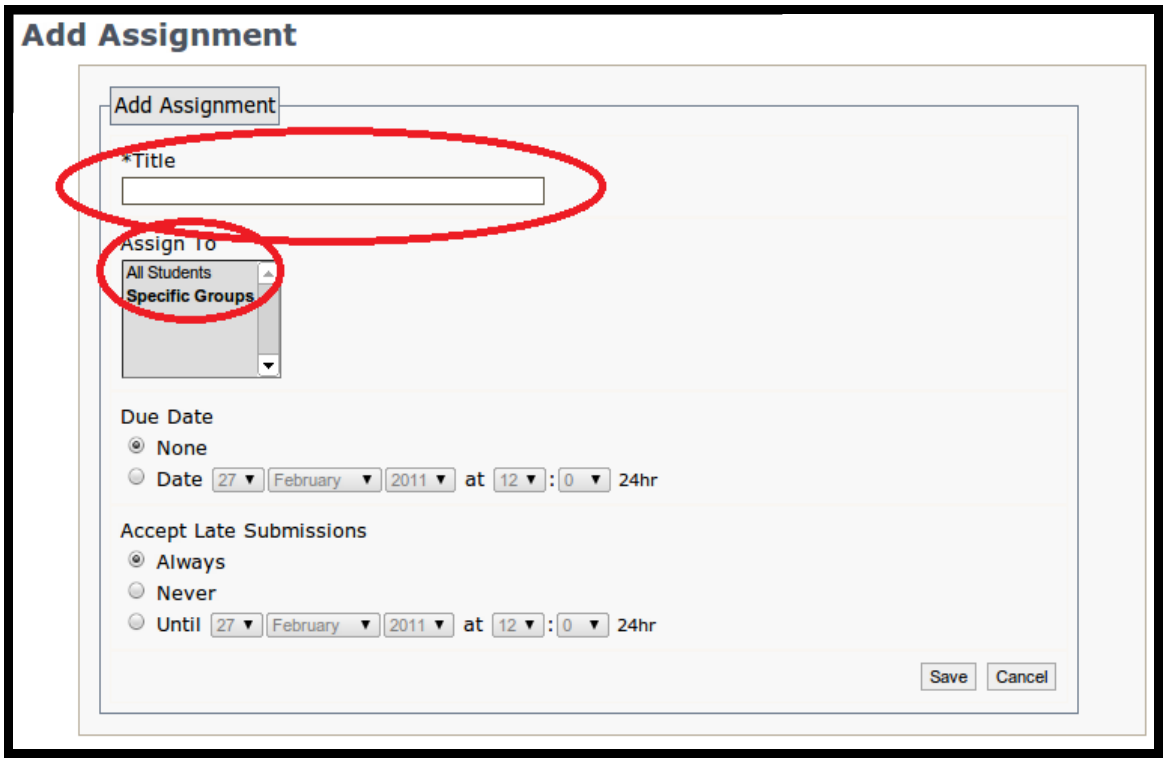


Figura 127 Formulario para la creación de una nueva news en ATutor.

c. Al seleccionar ***Forums*** aparecerá una vista con los foros para los alumnos pertenecientes al curso. Para agregar un nuevo foro es necesario regresar a la pestaña "Manage", clic en "Create Forum" e ingresar los datos correspondientes de título y descripción, véase la siguiente figura. Al finalizar el proceso enviará un mensaje de "Action completed succesfully" mostrando los foros del curso.

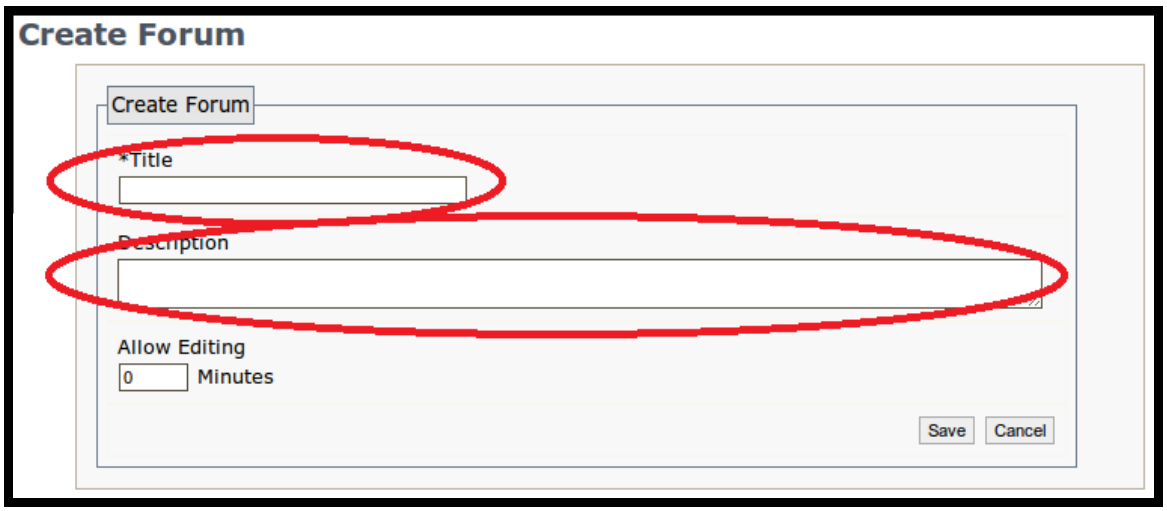


Figura 128 Formulario para la creación de un nuevo forum en ATutor.

d. Para ingresar al ***Chat*** se requiere ir a la página principal del curso "Course Home" dar clic en Chat y en "Enter the Chat", véase la siguiente figura.

Figura 129 Ingreso al chat en ATutor.

e. Al seleccionar ***Wiki*** aparecerá una página con opciones para modificar la actual, crear nuevas páginas, ayuda, actualización, entre otros. En este caso se toma en cuenta que el wiki es un módulo de terceros de nombre Ewiki, que ha sido incorporado a la plataforma. Para continuar con la creación del Wiki es necesario dar clic en "EditThisPage", modificarlo con nuestro texto y guardar. Al completar el proceso se verán reflejados los cambios en la página de wiki, como se muestra a continuación.

Figura 130 Edición de una wiki en ATutor.

f. Al seleccionar ***Course Email*** aparecerá una ventana para ingresar los datos a enviar, el subjet y el body. En este caso enviaremos el email a todos los usuarios enrolados, de no estar seleccionada la casilla "Enrolled" es necesario seleccionarla, véase la siguiente figura.

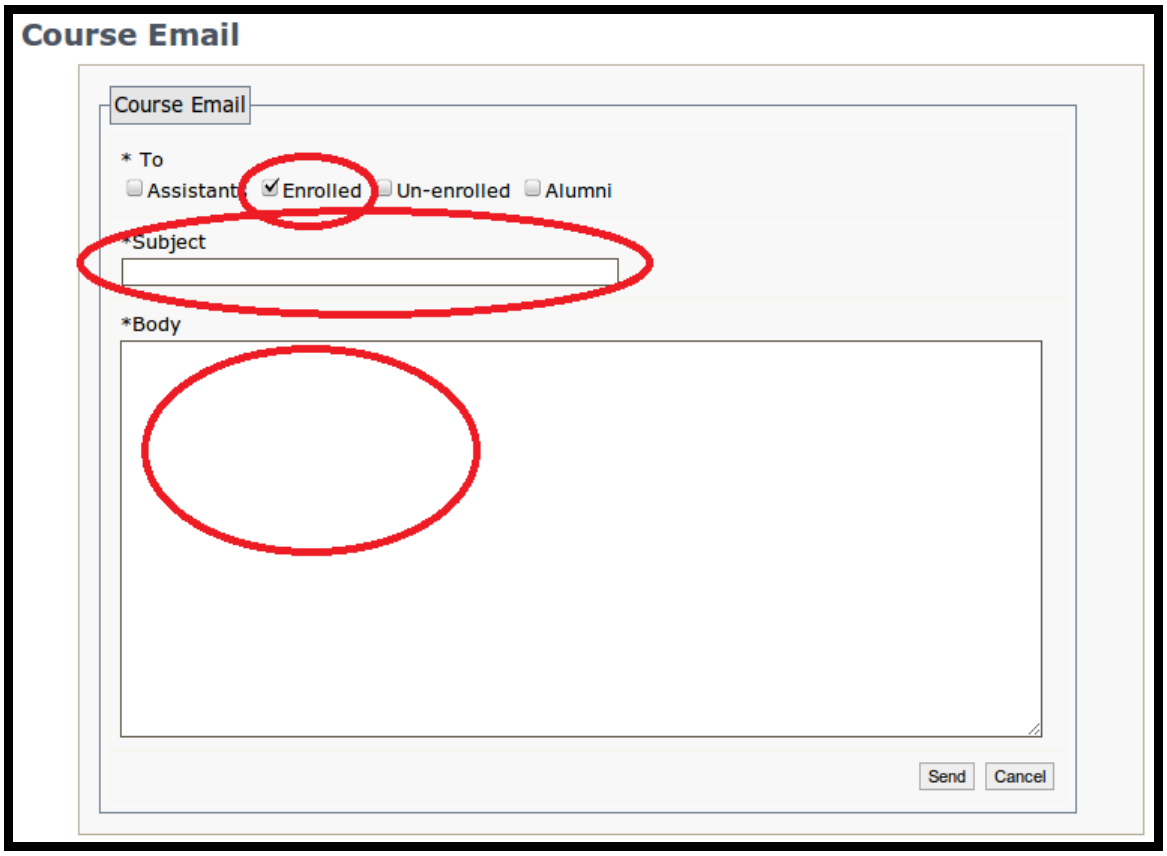


Figura 131 Formulario para el envío de una note en ATutor.

g. Para enviar un ***Email*** a un solo estudiante es necesario ir a la página principal del curso "Course Home", buscarlo en "My Contacts", clic en la opción "Send Message", rellenar el formulario y enviarlo, véase las siguientes figuras.

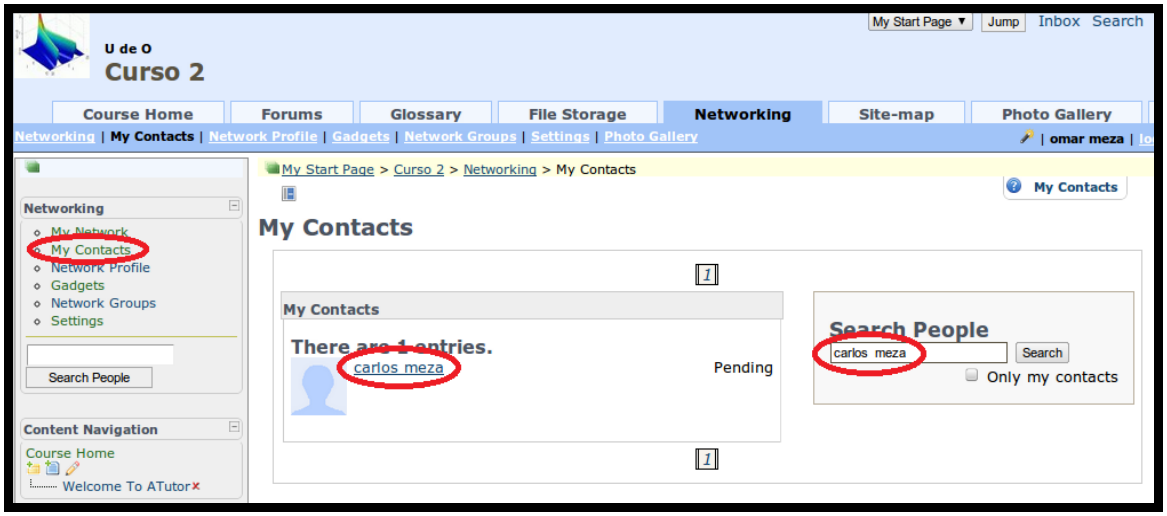


Figura 132 Selección de usuarios en ATutor para enviar una note.

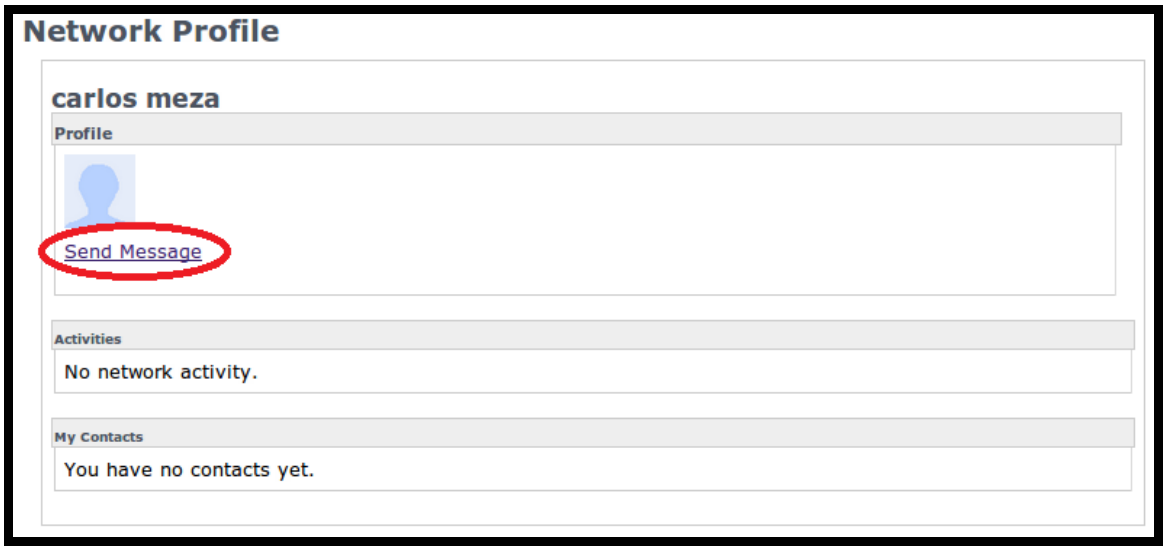


Figura 133 Acción para enviar un mensaje (note) en ATutor.

3.7.5 Pruebas de tiempo y esfuerzo

De acuerdo con (Yamada et al., 1993) podemos definir el esfuerzo como "cantidad de ocasiones en las que usuario selecciona o ingresa algún tipo de información al sistema". La medición de tiempos y esfuerzo se realizo para todos los sistemas y se midió: tiempo y esfuerzo en la creación de cada herramienta por tema, tiempo y esfuerzo en la creación de todas las herramientas que conforman un tema, tiempo y esfuerzo total en la creación de los cinco temas acompañados de sus herramientas. En las pruebas hemos considerado que el envió de las notas (mensajes) a los inscritos en el curso son una herramienta del tema. Para poder comparar los tiempos, es necesario que todos los módulos sean creados en el mismo orden para todos los sistemas.

3.7.5.1 Análisis de resultados para las pruebas de tiempo y esfuerzo entre KiwiDSM y Moodle

En el anexo digita (+Pruebas) se encuentra todas las mediciones de las pruebas realizadas, aquí se van a analizar los resultados finales que se obtuvieron, primero se realizara una comparativa entre la creación de los módulos en la plataforma Moodle contra la herramienta KiwiDSM; La siguiente figura muestra los tiempos promedio normalizados al mayor (el mayor tiempo para este caso es el tiempo total de crear todos los temas con sus módulos en Moodle) por tema, allí se ve que la diferencia al final de crear el primer tema, es tan solo de 1.75% más tiempo, en crear los módulos en Moodle que con la herramienta KiwiDSM, pero esta diferencia empieza a crecer desde el segundo tema en adelante, hasta tener una diferencia de 66.65% más rápido crear los módulos con la herramienta KiwiDSM que con Moodle, se observa que los tiempos para la creación de módulos en Moodle crecen más rápido que con la herramienta KiwiDSM.

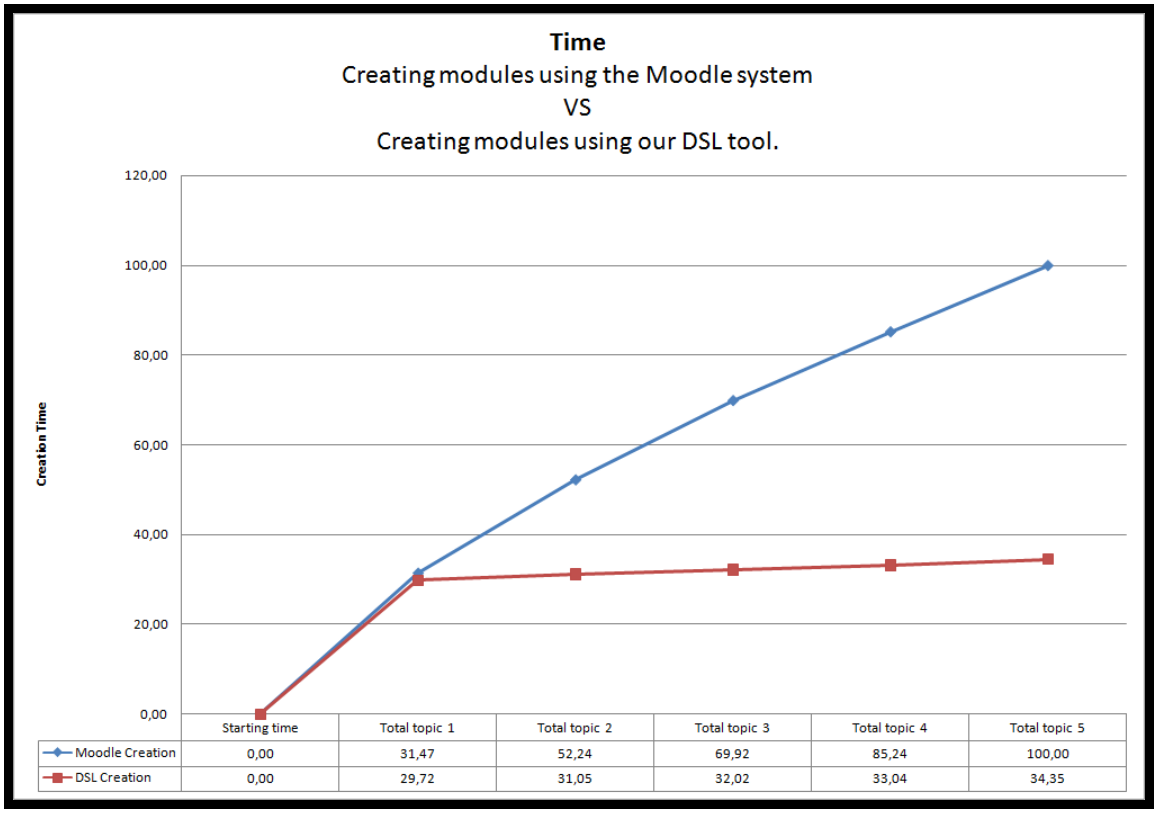


Figura 134. Comparación de tiempos entre la creación de módulos usando la plataforma Moodle Vs la creación de módulos con la herramienta KiwiDSM.

La siguiente figura muestra un comportamiento muy similar pero midiendo el esfuerzo promedio normalizado al mayor (el mayor esfuerzo para este caso se obtiene en la creación de todos los módulos para el primer tema con KiwiDSM y despliegue sobre Moodle), allí la diferencia al final de crear el primer tema, es de 12.68% menos esfuerzo con Moodle que con la herramienta KiwiDSM, pero desde el segundo tema en adelante, el esfuerzo en la creación de módulos con la herramienta KiwiDSM, es por lo menos 63% menor que al hacerlo en Moodle.

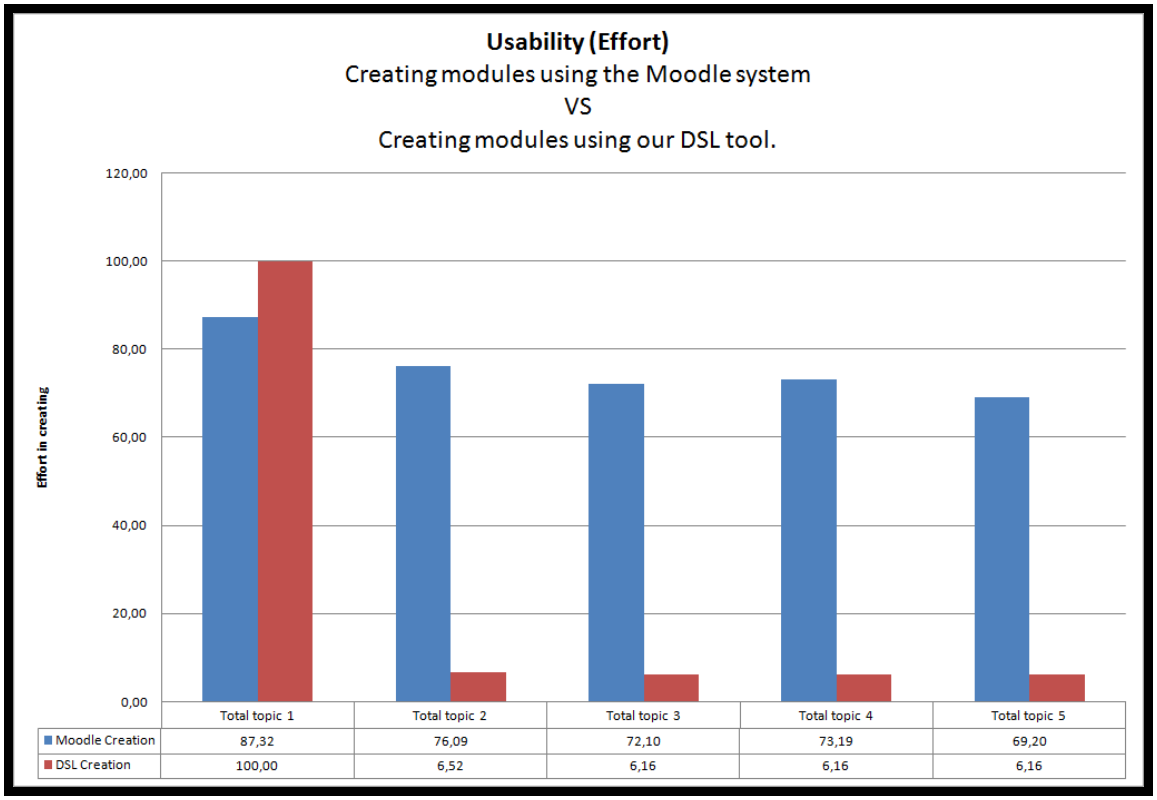


Figura 135 Comparación de usabilidad (esfuerzo) entre la creación de módulos usando la plataforma Moodle y la creación de módulos usando la herramienta KiwiDSM.

Las siguientes dos figuras muestran el tiempo promedio total y esfuerzo total normalizados al mayor valor (que para este caso se obtienen al crear todos los temas con sus módulos en Moodle), en la creación de 5 temas cada uno con sus respectivos módulos, y se concluye que la creación de módulos con la herramienta KiwiDSM, es por lo menos 65% más eficiente que al hacerlo en Moodle.

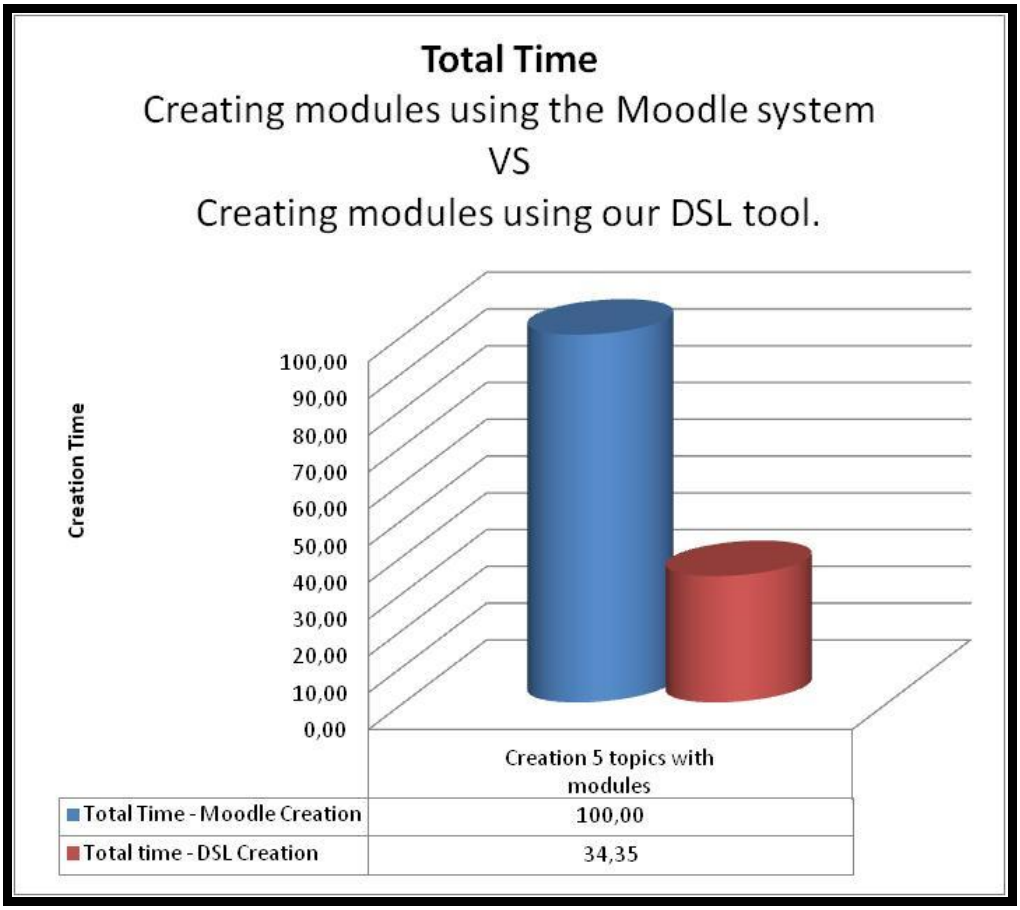


Figura 136 Comparación de tiempo total en la creación de módulos usando la plataforma Moodle Vs la herramienta KiwiDSM.

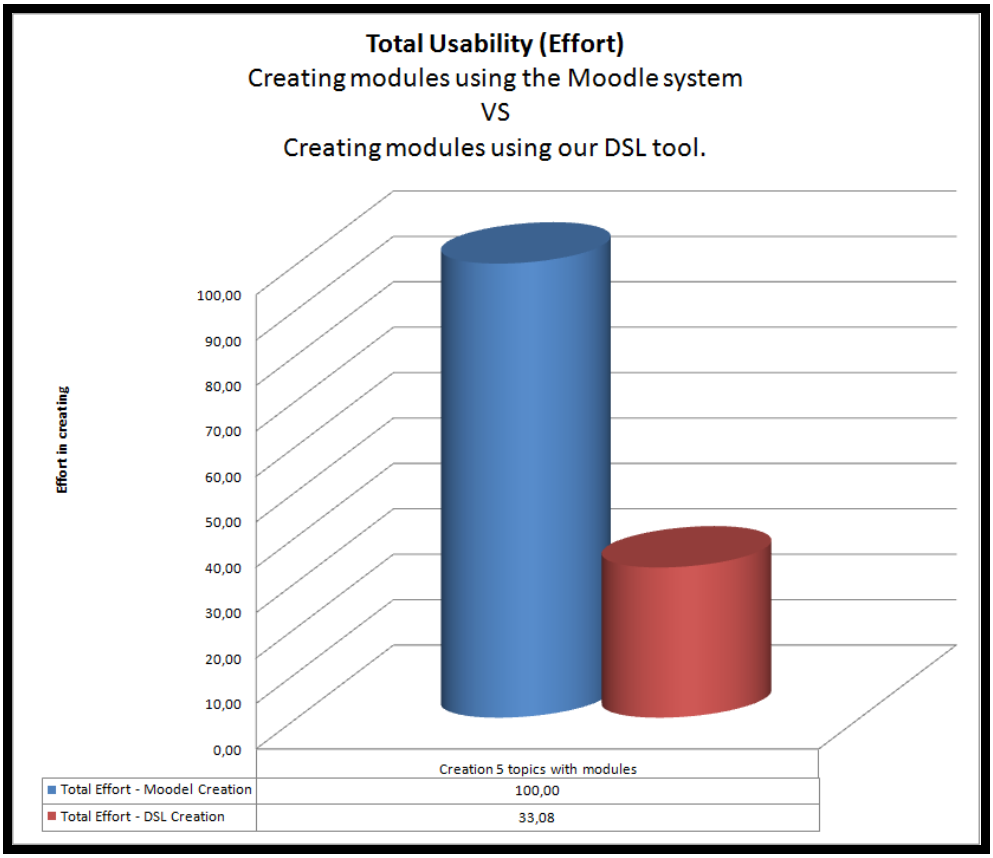


Figura 137 Comparación de usabilidad (esfuerzo) total en la creación de módulos usando la plataforma Moodle Vs la herramienta KiwiDSM.

3.7.5.2 Análisis de resultados para las pruebas de tiempo y esfuerzo entre KiwiDSM y Claroline

Ahora se analizaran los resultados para la comparativa entre la creación de los módulos en la plataforma Claroline contra la herramienta KiwiDSM, recuerda que en el anexo digital (+Pruebas) se encuentran las mediciones de las pruebas realizadas. La siguiente figura muestra los tiempos promedio normalizados al mayor (el mayor tiempo para este caso es el tiempo total de crear todos los temas con sus módulos en Claroline) por tema, allí se ve que la diferencia al final de crear el primer tema, es tan solo de 15.69% más tiempo, en crear los módulos en KiwiDSM que en Claroline, pero esta diferencia cambia a partir del segundo tema en adelante, logrando reducir hasta en un 51.62% los tiempos más rápido en crear los módulos con la herramienta KiwiDSM que con Claroline, se observa que los tiempos para la creación de módulos en Claroline crecen más rápido que con la herramienta KiwiDSM.

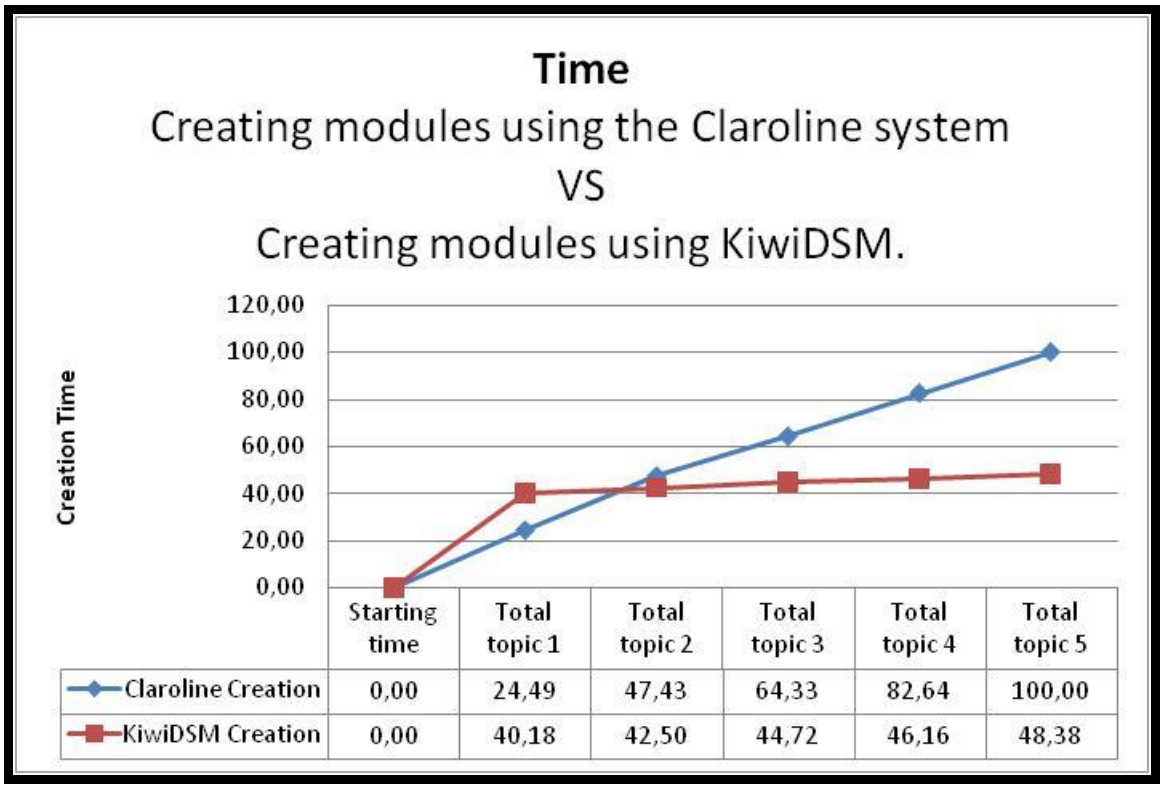


Figura 138. Comparación de tiempos entre la creación de módulos usando la plataforma Claroline Vs la creación de módulos con la herramienta KiwiDSM.

La siguiente figura muestra un comportamiento muy similar pero midiendo el esfuerzo promedio normalizado al mayor (el mayor esfuerzo para este caso se obtiene en la creación de todos los módulos para el primer tema con KiwiDSM y despliegue sobre Claroline), allí la diferencia al final de crear el primer tema, es de 18.39% menos esfuerzo con Claroline que con la herramienta KiwiDSM, pero desde el segundo tema en adelante, el esfuerzo en la creación de módulos con la herramienta KiwiDSM, es por lo menos 69.35% menor que al hacerlo en Claroline.

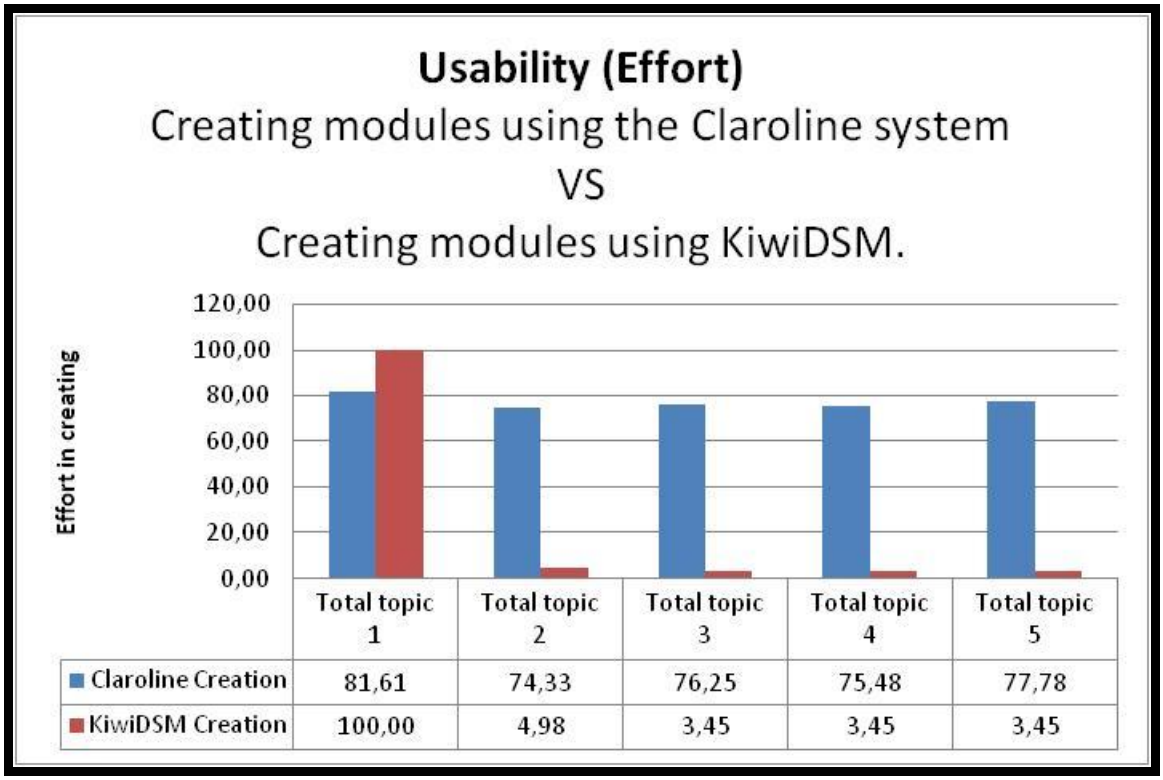


Figura 139 Comparación de usabilidad (esfuerzo) entre la creación de módulos usando la plataforma Claroline y la creación de módulos usando la herramienta KiwiDSM.

Las siguientes dos figuras muestran el tiempo promedio total y esfuerzo total normalizados al mayor valor (que para este caso se obtienen al crear todos los temas con sus módulos en Claroline), en la creación de 5 temas cada uno con sus respectivos módulos, y se concluye que la creación de módulos con la herramienta KiwiDSM, es por lo menos 51.62% más eficiente que al hacerlo en Claroline.

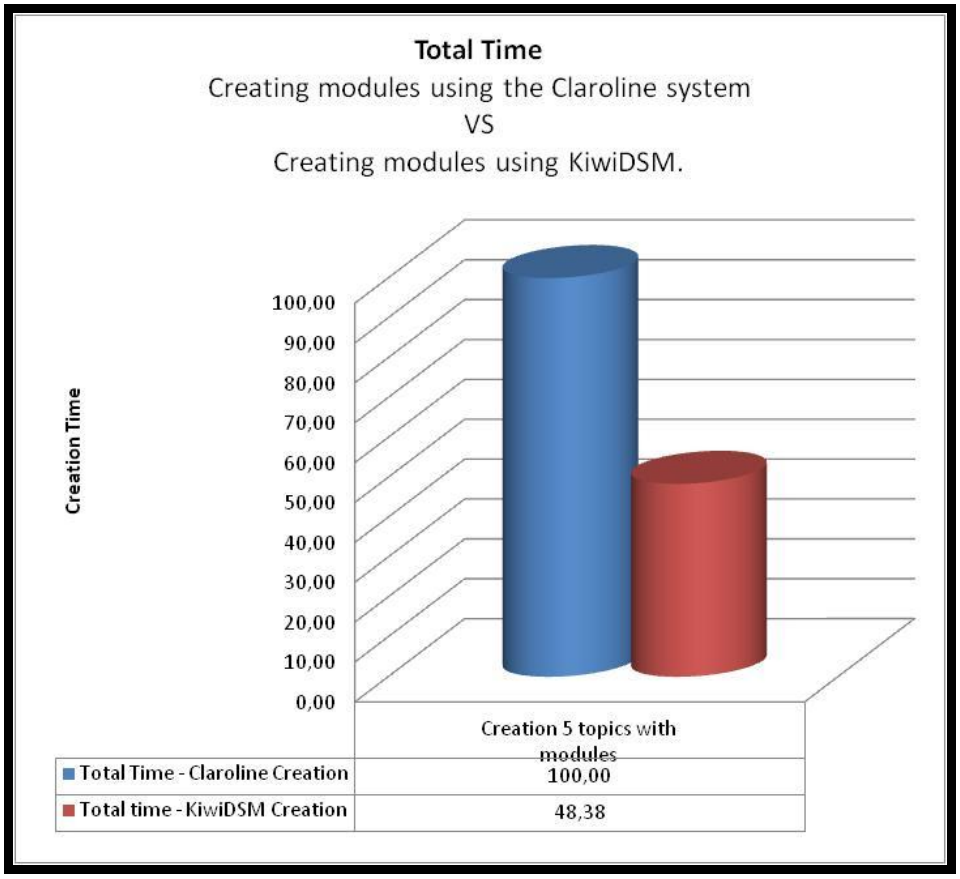


Figura 140 Comparación de tiempo total en la creación de módulos usando la plataforma Claroline Vs la herramienta KiwiDSM.

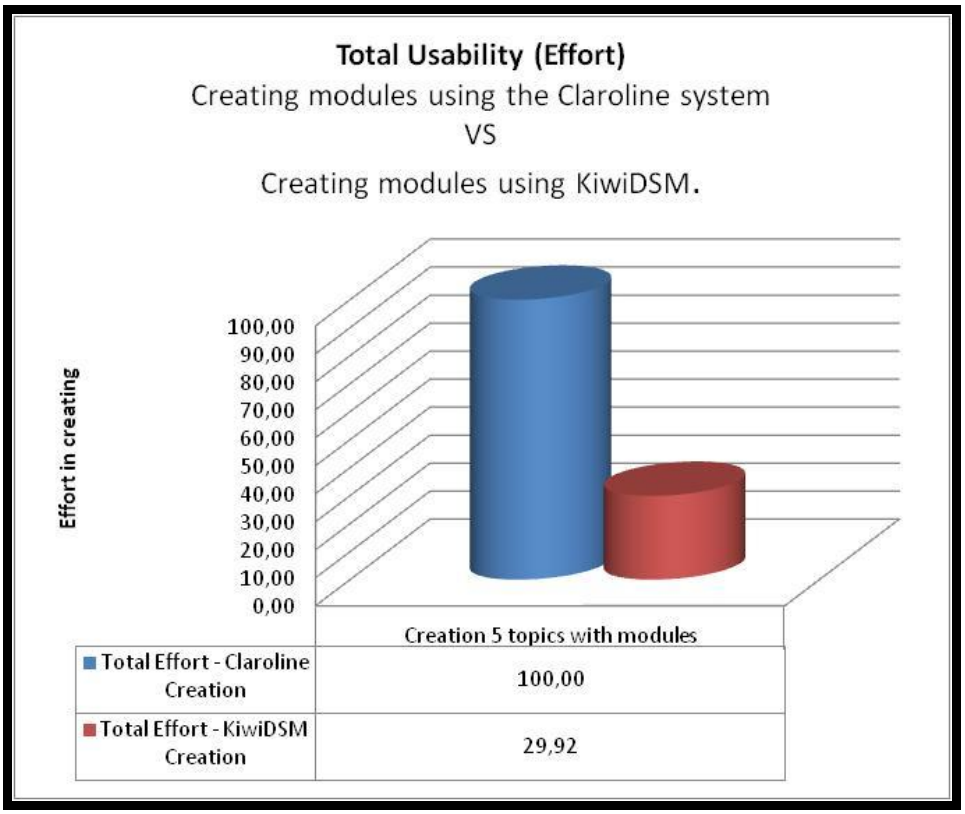


Figura 141 Comparación de usabilidad (esfuerzo) total en la creación de módulos usando la plataforma Claroline Vs la herramienta KiwiDSM.

3.7.5.3 Análisis de resultados para las pruebas de tiempo y esfuerzo entre KiwiDSM y ATutor

Finalmente se analizaran los resultados para la comparativa entre la creación de los módulos en la plataforma Atutor contra la herramienta KiwiDSM, en el anexo digital (+Pruebas) se encuentran las mediciones de las pruebas realizadas. La siguiente figura muestra los tiempos promedio normalizados al mayor (el mayor tiempo para este caso es el tiempo total de crear todos los temas con sus módulos en Atutor) por tema, allí se ve que la diferencia al final de crear el primer tema, es tan solo de 7.65% más tiempo, en crear los módulos en KiwiDSM que en Atutor, pero esta diferencia cambia a partir del segundo tema en adelante, logrando reducir hasta en un 58.87% los tiempos más rápido en crear los módulos con la herramienta KiwiDSM que con Atutor, se observa que los tiempos para la creación de módulos en Atutor crecen más rápido que con la herramienta KiwiDSM.

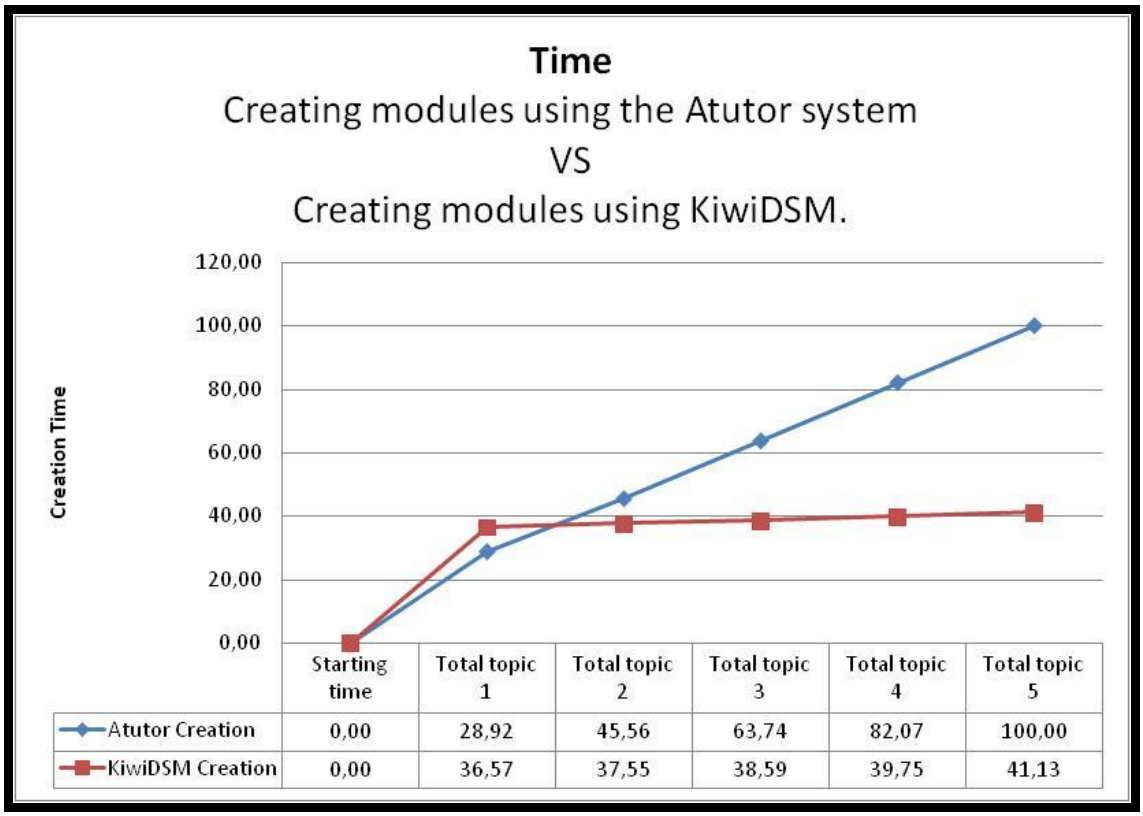


Figura 142. Comparación de tiempos entre la creación de módulos usando la plataforma Atutor Vs la creación de módulos con la herramienta KiwiDSM.

La siguiente figura muestra un comportamiento muy similar pero midiendo el esfuerzo promedio normalizado al mayor (el mayor esfuerzo para este caso se obtiene en la creación de todos los módulos para el primer tema con KiwiDSM y despliegue sobre Atutor), allí la diferencia al final de crear el primer tema, es de 17.98% menos esfuerzo con Atutor que con la herramienta KiwiDSM, pero desde el segundo tema en adelante, el esfuerzo en la creación de módulos con la herramienta KiwiDSM, es por lo menos 71.16% menor que al hacerlo en Atutor.

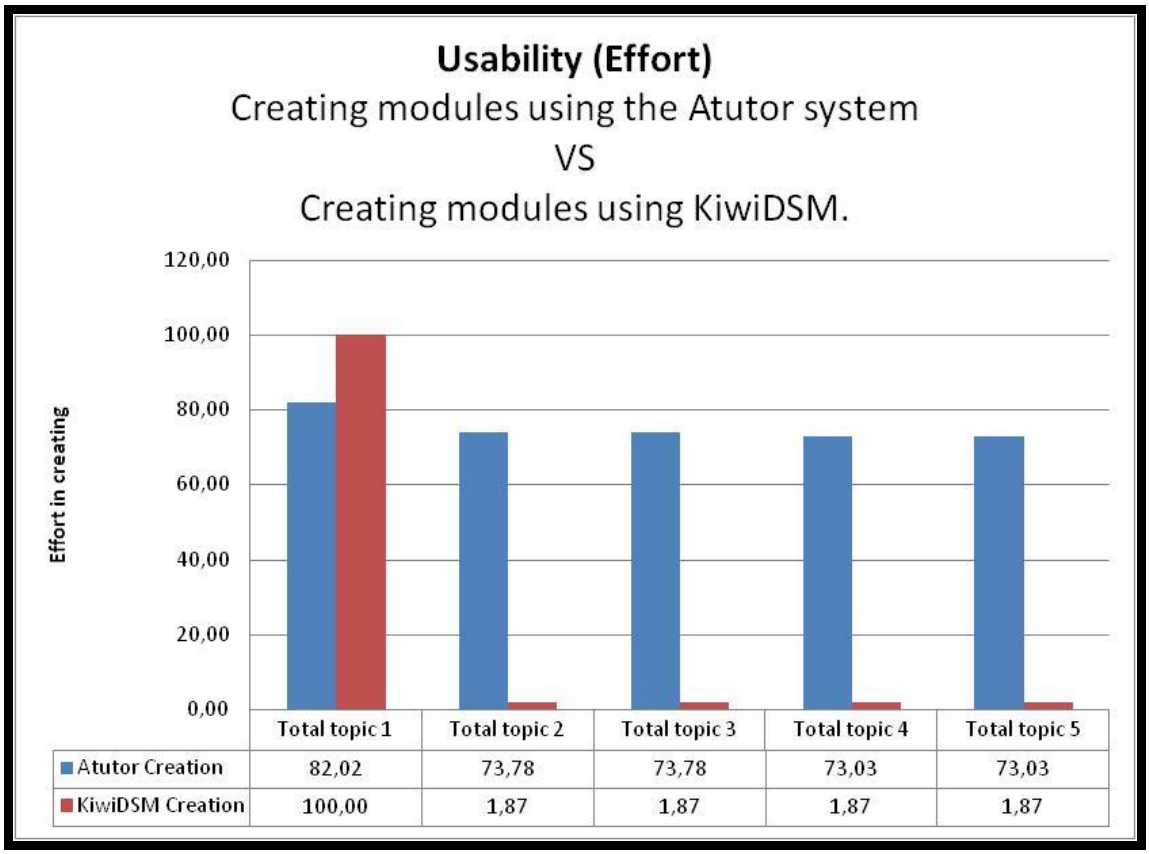


Figura 143 Comparación de usabilidad (esfuerzo) entre la creación de módulos usando la plataforma Atutor y la creación de módulos usando la herramienta KiwiDSM.

Las siguientes dos figuras muestran el tiempo promedio total y esfuerzo total normalizados al mayor valor (que para este caso se obtienen al crear todos los temas con sus módulos en Atutor), en la creación de 5 temas cada uno con sus respectivos módulos, y se concluye que la creación de módulos con la herramienta KiwiDSM, es por lo menos 58.87% más eficiente que al hacerlo en Atutor.

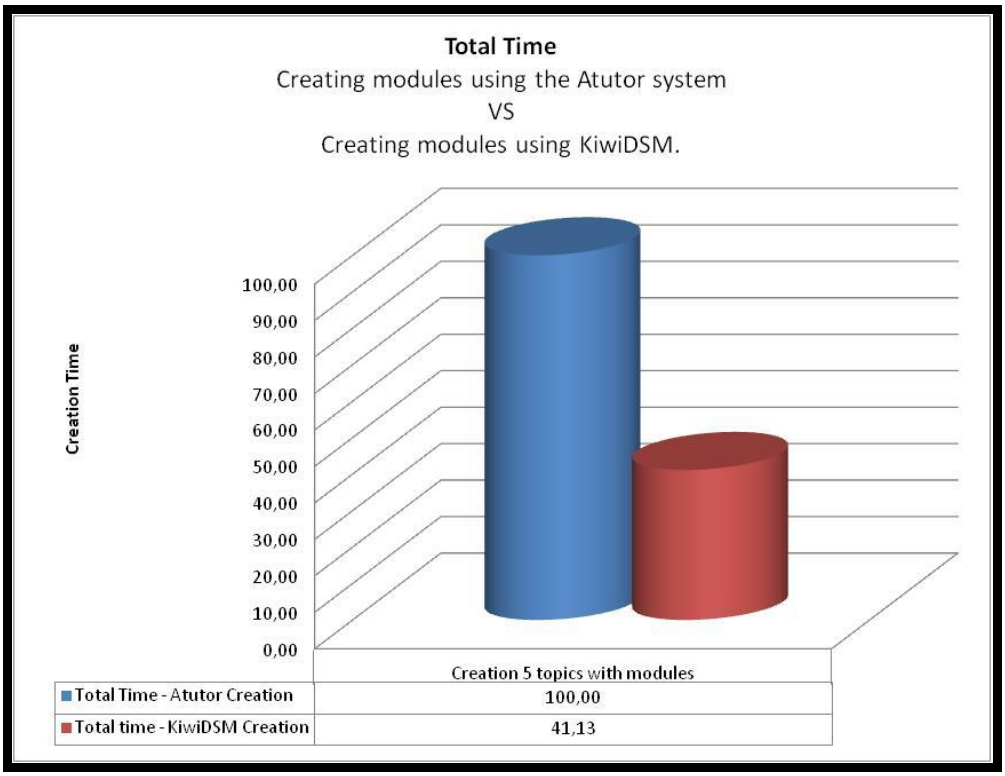


Figura 144 Comparación de tiempo total en la creación de módulos usando la plataforma Atutor Vs la herramienta KiwiDSM.

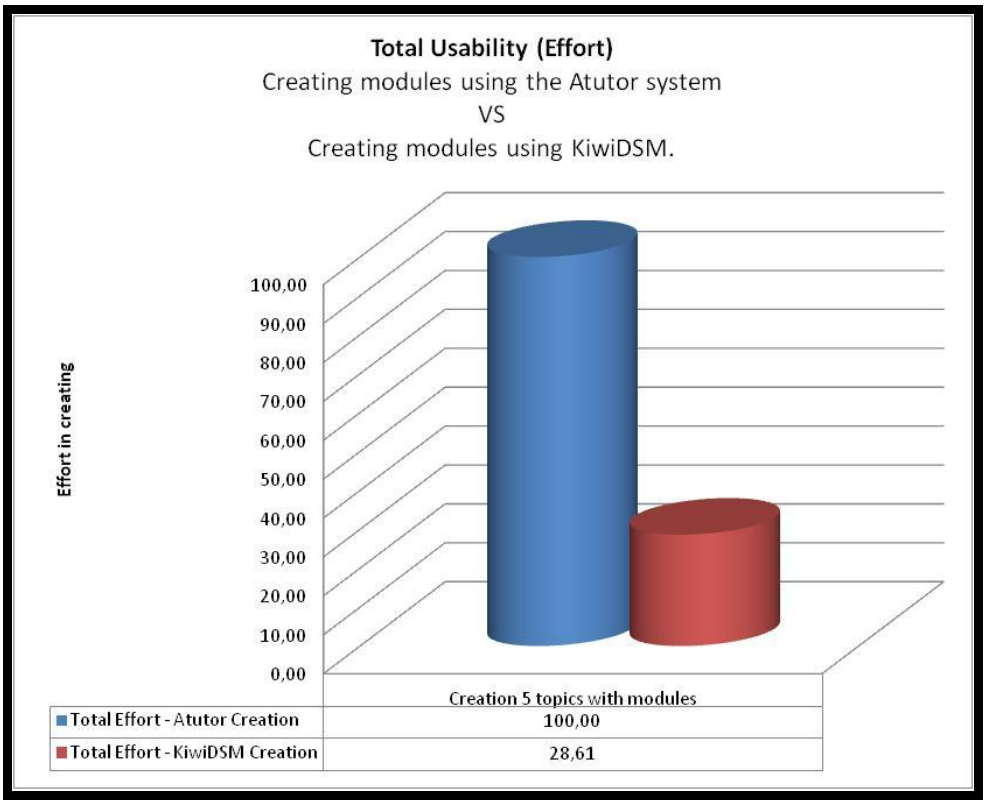


Figura 145 Comparación de usabilidad (esfuerzo) total en la creación de módulos usando la plataforma Atutor Vs la herramienta KiwiDSM.

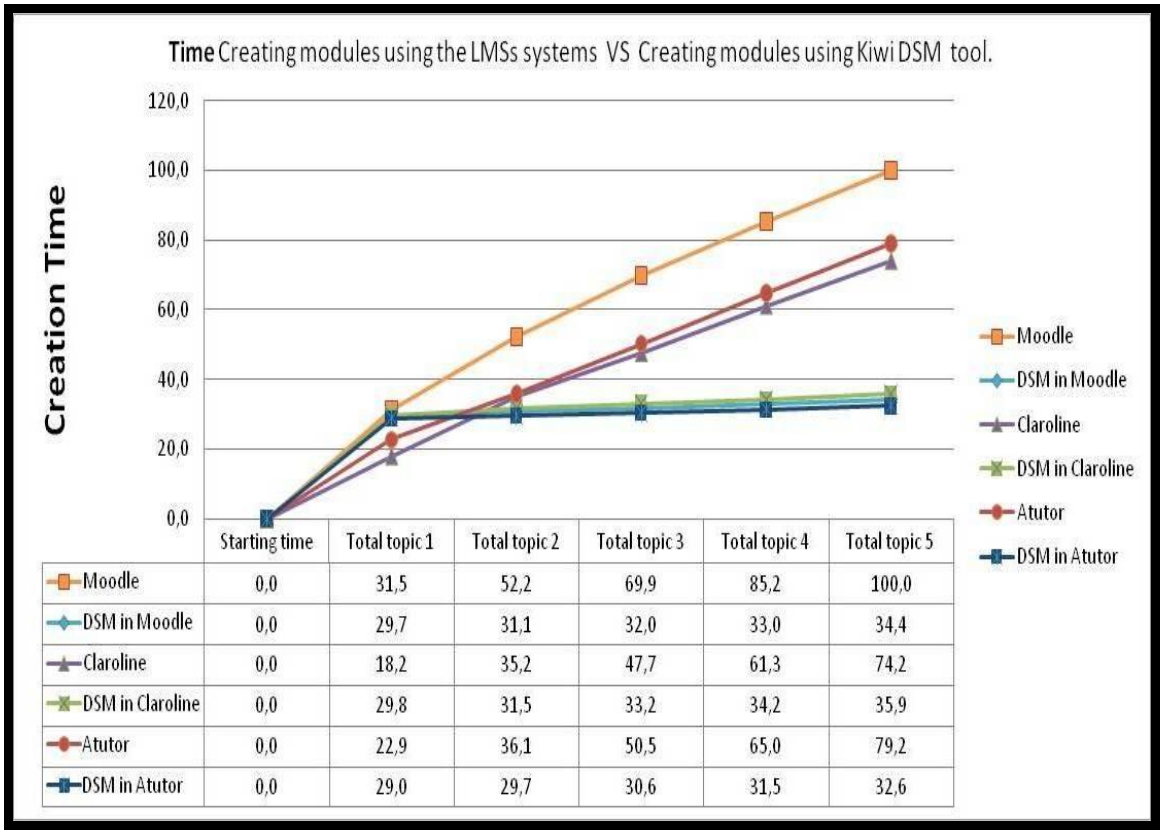
Las anteriores pruebas demuestran que al trabajar con modelos, se disminuye el tiempo y el esfuerzo del usuario, en la creación de módulos para una plataforma LMS, en este caso Moodle, Claroline y Atutor, además que el metamodelo funciona, ya que no se genero ningún fallo en la creación de los módulos con la herramienta DSL, que tiene como base fundamental el metamodelo LMS planteado.

3.7.5.4 Análisis de resultados para las pruebas de tiempo y esfuerzo entre el despliegue de módulos con KiwiDSM y las plataformas LMS trabajadas.

El objetivo de estas pruebas es analizar el comportamiento de crear módulos con KiwiDSM versus la creación de los mismos módulos con cada plataforma LMS, se emplearon los mismos modelos de pruebas que en las secciones anteriores y la normalización se ha realizado siempre al valor más alto, con el fin de obtener valores porcentuales.

La siguiente gráfica muestra una comparación respecto al tiempo en la creación de módulos con la propia plataforma LMS versus la utilización de KiwiDSM y se visualiza que la utilización de una herramienta DSL como KiwiDSM, reduce los tiempos a partir de la primera generación, y que esta diferencia crece a favor de KiwiDSM (gastando menos tiempo) respecto a cualquier plataforma LMS a medida que se generan más temas con sus respectivos módulos. Así se ve que el mayor y menor tiempo al finalizar

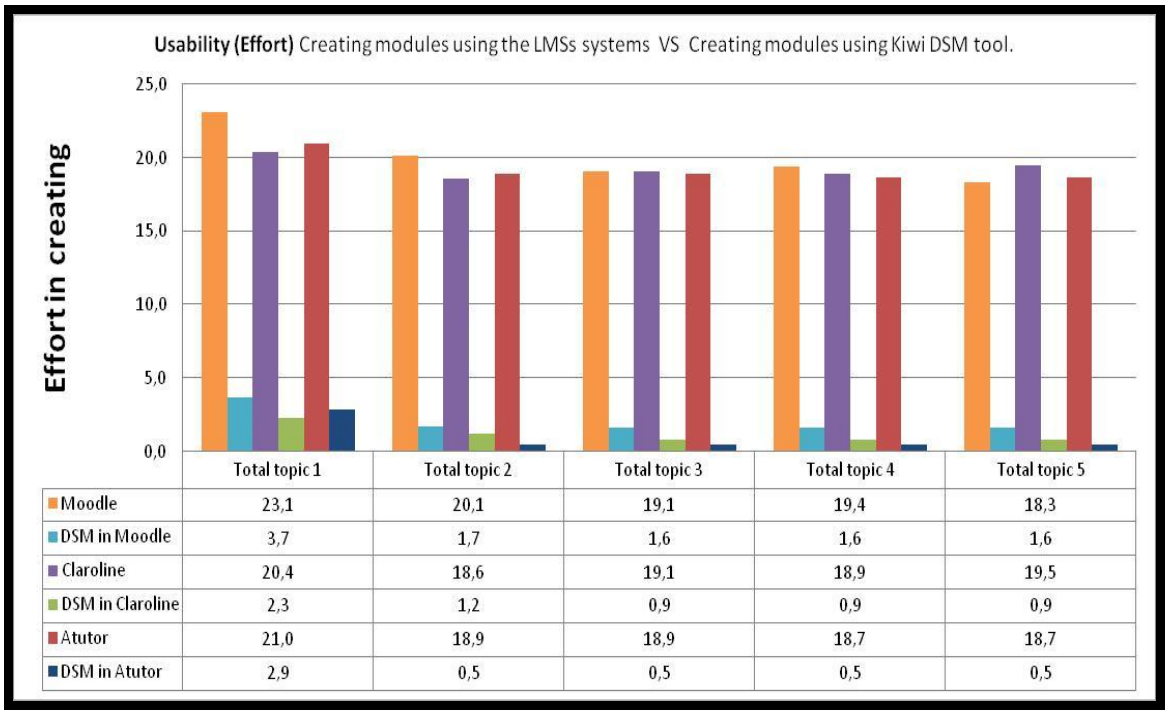
la creación del quinto tema con KiwiDSM es 35.9% y 32.6% respectivamente, mientras que con las plataformas LMS es de 74.2% y 100% respectivamente, mostrando así una reducción bastante considerable en tiempo que favorece la utilización de herramientas DSL en plataformas de educación virtual y que se podría extender a diversos campos de acción.



	Starting time	Total topic 1	Total topic 2	Total topic 3	Total topic 4	Total topic 5
Moodle	0,0	31,5	52,2	69,9	85,2	100,0
DSM in Moodle	0,0	29,7	31,1	32,0	33,0	34,4
Claroline	0,0	18,2	35,2	47,7	61,3	74,2
DSM in Claroline	0,0	29,8	31,5	33,2	34,2	35,9
Atutor	0,0	22,9	36,1	50,5	65,0	79,2
DSM in Atutor	0,0	29,0	29,7	30,6	31,5	32,6

Figura 146 Tiempo en la creación de módulos usando los sistemas LMS Vs la creación de los mismos módulos con KiwiDSM.

Un comportamiento muy similar se ve en la medición de esfuerzo, como lo muestra la siguiente grafica, en la cual y en contraste de lo que ocurre con el tiempo, la diferencia de esfuerzo se ve plasmada desde el principio en favor de hacer menos esfuerzo con KiwiDSM, logrando mínimas y máximas diferencias hasta del 16.7% y 20.8% respectivamente, en ambos casos aparecen en la generación del primer tema.



	Total topic 1	Total topic 2	Total topic 3	Total topic 4	Total topic 5
Moodle	23,1	20,1	19,1	19,4	18,3
DSM in Moodle	3,7	1,7	1,6	1,6	1,6
Claroline	20,4	18,6	19,1	18,9	19,5
DSM in Claroline	2,3	1,2	0,9	0,9	0,9
Atutor	21,0	18,9	18,9	18,7	18,7
DSM in Atutor	2,9	0,5	0,5	0,5	0,5

Figura 147 Esfuerzo (usabilidad) en la creación de módulos usando los sistemas LMS Vs la creación de los mismos módulos con KiwiDSM.

Por último en las siguientes graficas, se muestran consolidados los resultados totales de las pruebas respecto a tiempo y esfuerzo, en la creación de módulos con KiwiDSM versus la creación de módulos en cada plataforma y es evidente en la comparativa, la reducción en los dos casos logrando un máximo de reducción del 67.4% en tiempo y 72.5% en esfuerzo.

De esta forma y según las graficas e interpretaciones anteriores, al trabajar con KiwiDSM o herramientas DSL la disminución de tiempo y esfuerzo es notoria, aportando a la validación de la hipótesis inicial del trabajo, respecto a la disminución de tiempo y esfuerzo empleando lenguajes de dominio.

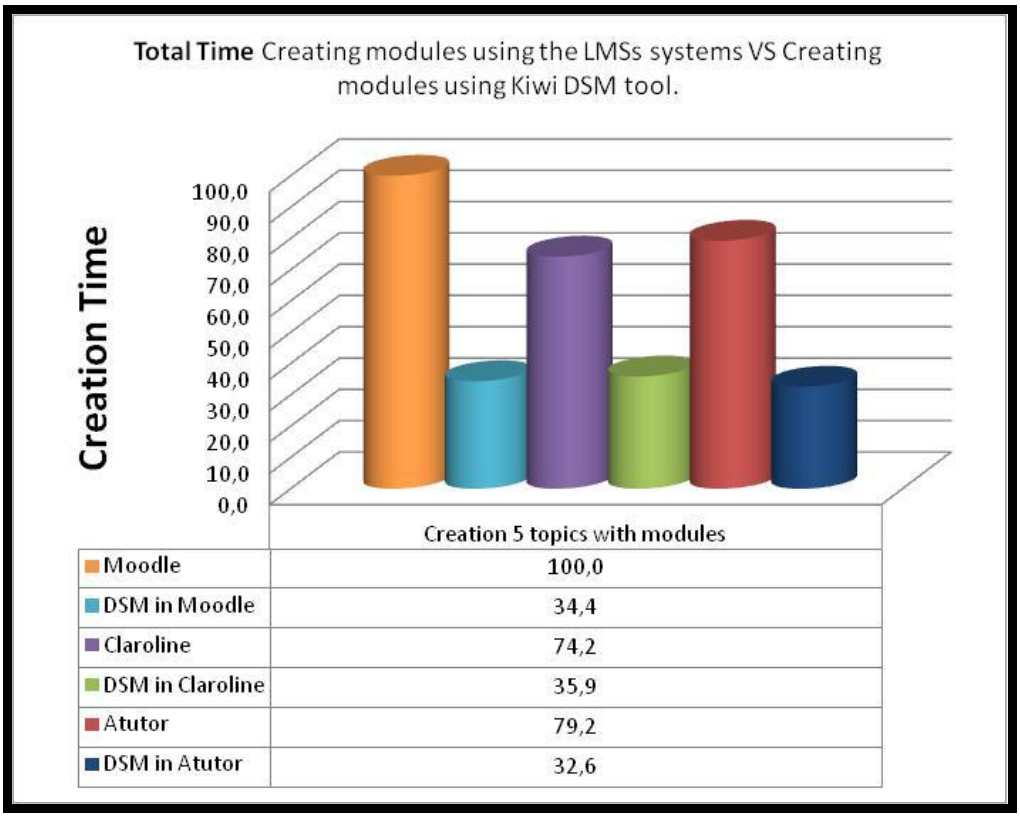


Figura 148 Tiempos totales en la creación de módulos usando los sistemas LMS Vs la creación de los mismos módulos con KiwiDSM.

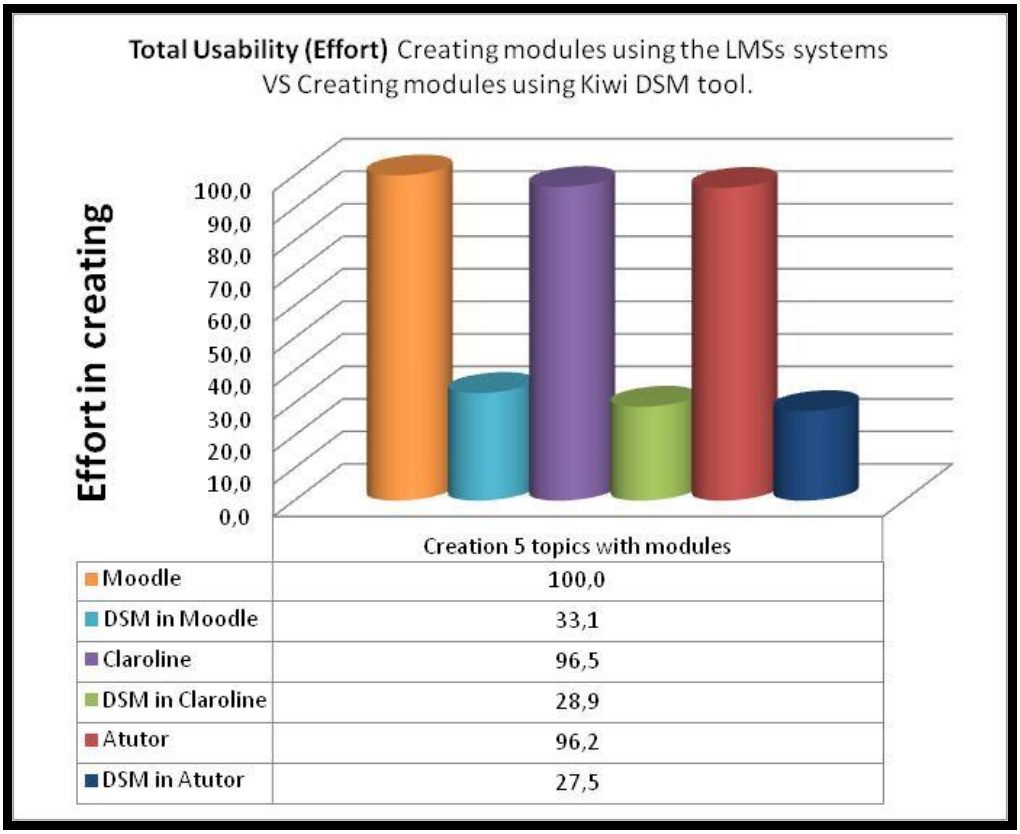


Figura 149 Esfuerzos (Usabilidad) totales en la creación de módulos usando los sistemas LMS Vs la creación de los mismos módulos con KiwiDSM.

3.7.5.5 Análisis de resultados para las pruebas de tiempo y esfuerzo entre el despliegue de módulos con KiwiDSM v1.0 vs KiwiDSM v2.0.

El objetivo de estas pruebas es analizar el comportamiento de crear módulos con KiwiDSM v1.0 y la mejora hecha en KiwiDSM v2.0, por ello solo es necesario contemplar la información del modelado sobre la herramienta, y nada de su despliegue sobre las diferente plataformas LMS, al igual que en los anteriores casos la normalización se ha realizado siempre al valor más alto, con el fin de obtener valores porcentuales.

En las siguientes grafica se puede visualizar la comparación en tiempo y esfuerzo, de crear los módulos con las dos versiones de la herramientas y se observa que las mejoras realizadas mejoran el rendimiento de los usuarios en la creación del modelo, esto se explica debido a que en Kiwi v2.0 es más claro el funcionamiento (usabilidad), permitiendo que no se confundan al momento de seleccionar el tipo de conexión utilizar, gran problema que tenia la primera versión, también ayuda bastante el hecho de haber cambiado la forma del nodo *"Communications"*, la asignación de colores diferentes a cada módulos, la coincidencia de las imágenes de la paleta de herramientas con el área de trabajo y la mejor visualización de las conexión en el área de trabajo.

Como se ve en las figuras siguientes, el rendimiento en tiempo mejora hasta un 16.45% y el esfuerzo se reduce hasta un 16.05% menos, en la nuevas versión de la herramienta KiwiDSM v2.0.

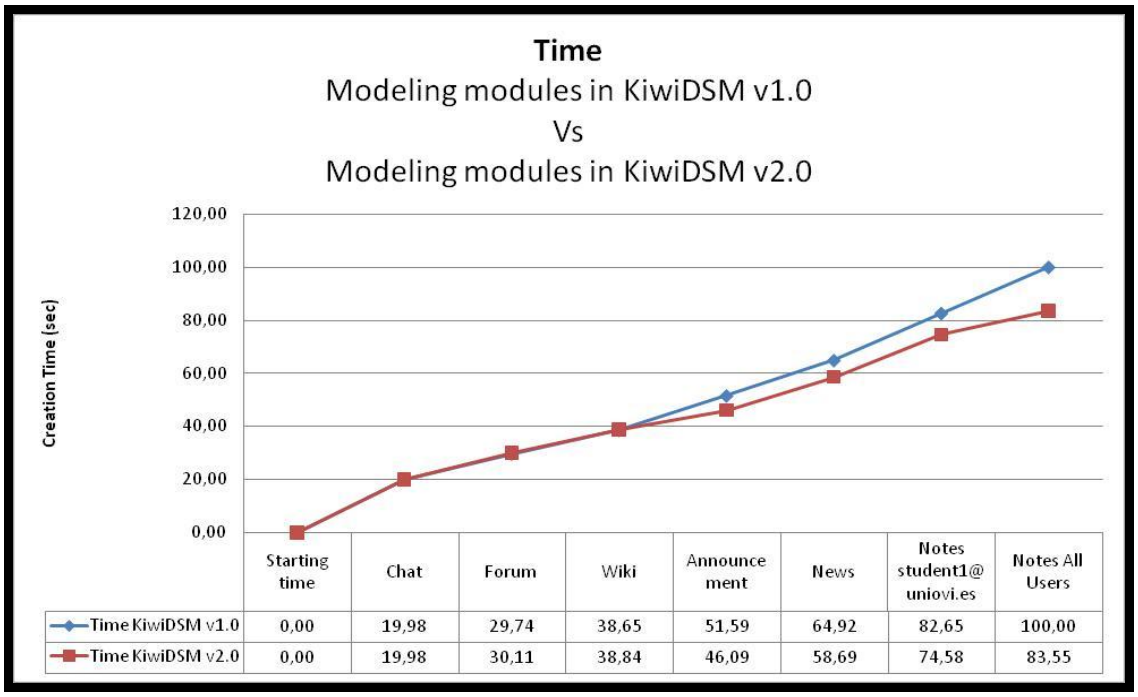


Figura 150 Tiempos totales en el modelamiento de módulos usando KiwiDSM v1.0 Vs KiwiDSM v2.0.

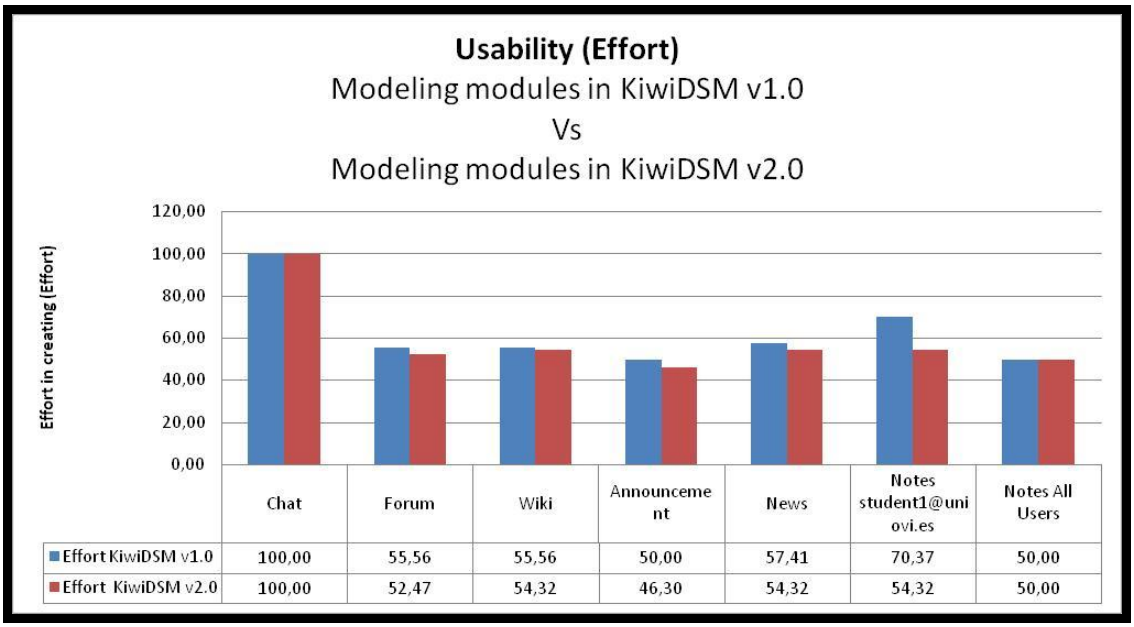


Figura 151 Esfuerzos (Usabilidad) totales en el modelamiento de módulos usando KiwiDSM v1.0 Vs KiwiDSM v2.0.

También se realizaron pruebas referentes al desenvolvimiento de KiwiDSM en los dos sistemas operativos más reconocidos Windows y Linux. Las pruebas sobre Windows se realizaron en la versión de Windows 7 Professional, con Eclipse Helios y como servidor web apache con php y mysql Xampp 1.7.3, mientras que para Linux se utilizo una distribución Ubuntu Desktop Edition 10.04 con Eclipse Helios y como servidor web apache con php y mysql Xampp 1.7.3, los resultados fueron exitosos y la aplicación funciona perfectamente en los dos sistemas operativos, esto es irrelevante pues realmente quien garantizan que todo funcione sobre los dos sistemas son los proyectos Eclipse y Xampp, si estos funcionan bien nuestra propuesta también funcionara bien ya que se basa totalmente sobre ellos.

3.7.6 Validación del metamodelo para sistemas de gestión del aprendizaje – módulos de comunicaciones

En la validación del metamodelo para sistemas de gestión del aprendizaje – módulos de comunicaciones, se hace necesario emplear un mecanismo formal de validación para este tipo de proyectos, ya que su aprobación generalmente se hace de manera empírica, determinando que si se cumple el objetivo el metamodelo es válido. Por esta razón se han empleado las redes de petri como mecanismo formal de validación, este método ya se ha empleado antes en trabajos como (Montañez, 2008, Romero and de Lara, 2006).

Para una descripción más extensa, el lector puede consultar por ejemplo (Peterson, 1981, Jensen, 1991). Las redes de Petri (Peterson, 1981) son un lenguaje gráfico y formal para la descripción, simulación y análisis de sistemas dinámicos discretos. Son adecuadas para modelar sistemas distribuidos y concurrentes, y han sido usadas con éxito por ejemplo en el diseño de protocolos de comunicación, diseño de sistemas operativos, diseño de hardware, sistemas embebidos, diseño de software y modelado del proceso de negocio.

Una red de Petri es un grafo bipartito, formado por dos tipos de nodos: lugares (Places) y transiciones. Los lugares se pueden conectar con las transiciones (y viceversa) por medio de arcos. Los lugares conectados con un arco a una transición son sus lugares de entrada. Los lugares que reciben un arco de una cierta transición, son sus lugares de salida. Los lugares contienen cero o más tokens. El número de tokens de cada uno de los lugares se conoce como el “marcado de la red”, y define unívocamente su estado. La simulación de una red de Petri consiste en el disparo de transiciones habilitadas. Una transición está habilitada si todos sus lugares de entrada contienen al menos un token. Si una transición se dispara, se elimina un token de cada uno de los lugares de entrada, y se añade un token a cada uno de sus lugares de salida.

Las pruebas que se hicieron al modelo de la herramienta KiwiDSM se basaron en propiedades de los requisitos definidos en el metamodelo, con el objetivo de validar que el modelo construido cumplía los requisitos del metamodelo.

Dichas pruebas se hicieron principalmente a los módulos creados en las plataformas LMS Moodle, Claroline y ATutor, que poseen las debidas transformaciones del modelo a ellas y que fueron explicadas en capítulos anteriores.

Se definieron un conjunto de pruebas que describen la actividad que se puede realizar con la herramienta KiwiDSM hasta su despliegue en las plataformas LMS Moodle, Claroline y ATutor, dichas pruebas corresponden a diferentes comportamientos que se puede dar para cumplir un requisito. Para cada prueba se hizo la siguiente formalización:

Prueba = { <**fórmula**, **resultado**> } donde,

fórmula Є a una composición <entrada, salida>

resultado = 1 (verdadero), si la fórmula modela un comportamiento válido y

0 (falso), si la fórmula modela un comportamiento inválido.

Para estas pruebas se incluyeron comportamientos válidos y no válidos. Si la prueba pasa satisfactoriamente en un comportamiento no válido indica que el modelo tiene fallas. Cada prueba se modelo con una *Red de Petri,* con el fin observar su comportamiento y así encontrar errores en el modelo.

Una *Red de Petri*: es una cuádrupla R = {P, T, I, O} donde,

P es un conjunto finito y no vacío de nodos, dadas como condiciones

T es un conjunto finito y no vacío de transiciones, dadas como eventos

$P \cap T = 0$

I:P x T → función de entrada

O:T x P → función de salida

A continuación se describirán las pruebas definidas para la creación de cada módulo.

3.7.6.1 Validación en Creación del módulo foro

Se tienen los siguientes nodos y transiciones:

P = {p1, p2, p3, p4, p5, pFin}

T = {t1, t2, t3, t4, t5, t6}

p1 = Esperando elementos en el Canvas.

p2 = Nodo Communications creado en el Canvas.

p3 = Nodo Forum creado en el Canvas.

p4 = Dato forumName asignado al Forum.

p5 = Dato message asignado al Forum.

pFin = Módulo Forum desplegado en la plataforma.

t1 = Se selecciono el nodo Communications de la paleta de herramientas y se coloco en el Canvas.

t2 = Se selecciono el nodo Forum de la paleta de herramientas y se coloco en el Canvas.

t3 = Se selecciono la conexión Communications -> Forum de la paleta de herramientas y se conecto el nodo Communications con el nodo Forum.

t4 = Se asigno información al campo forumName del nodo Forum.

t5 = Se asigno información al campo message del nodo Forum.

t6 = Se realizan las debidas transformaciones y despliegue sobre la plataforma LMS.

Red de petri para la creación de Módulo Forum.

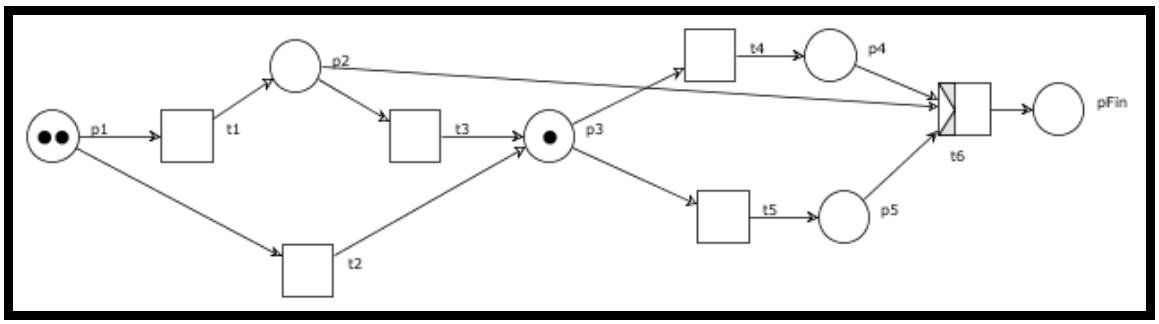


Figura 152 Red de petri para la creación de módulos Forum.

Comportamiento valido.

Prueba1 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, <p3, t4>, <p3, t5>, <t4, p4>, <t4, p5>, <t6, pFin>,1}

En esta prueba se obtuvo como resultado 1, lo que indica que este comportamiento se da en el modelo, y por consiguiente cumple con el requisito. A continuación se

presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

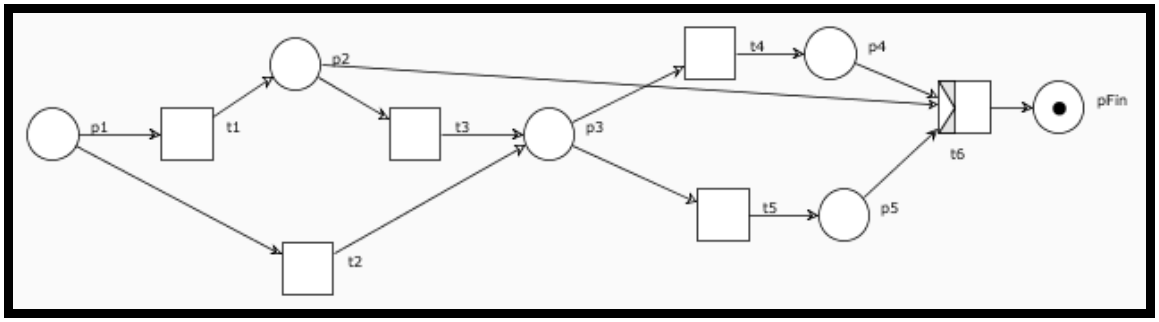


Figura 153 Red de petri comportamiento valido para la creación de módulos Forum.

Comportamiento no valido.

Prueba2 = {<p1, t2>, <t2, p3>, <p3, t4>, <t4, p4>, <p3, t5>, <t5, p5>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

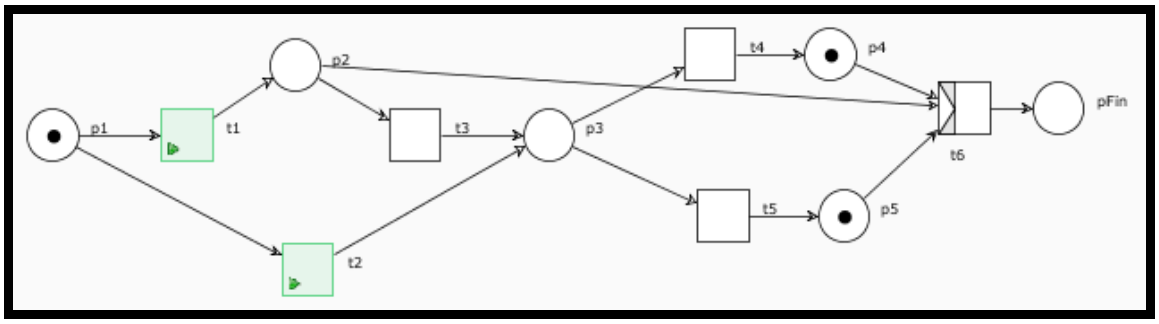


Figura 154 Red de petri comportamiento no valido 1 para la creación de módulos Forum.

Prueba3 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, <p3, t4>, <t4, p4>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada

transición.

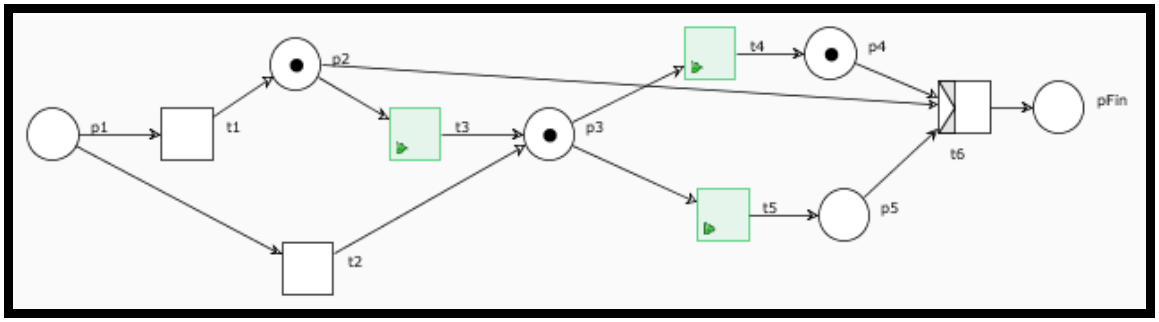


Figura 155 Red de petri comportamiento no valido 2 para la creación de módulos Forum.

Prueba4 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, <p3, t5>, <t5, p5>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

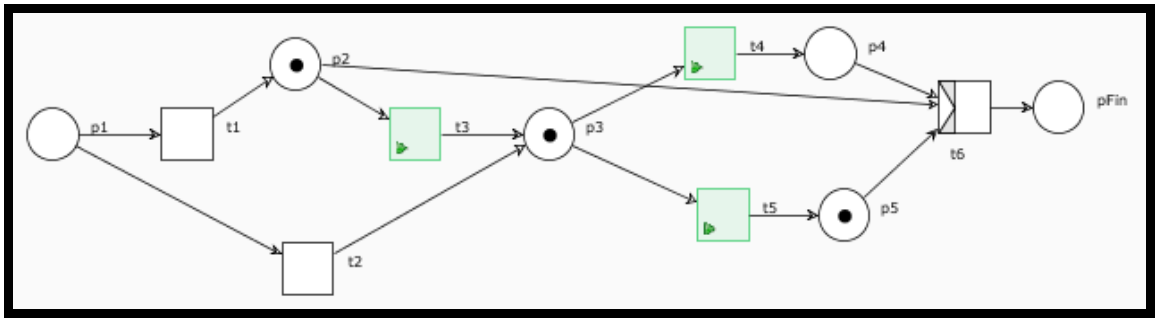


Figura 156 Red de petri comportamiento no valido 3 para la creación de módulos Forum.

Prueba5 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

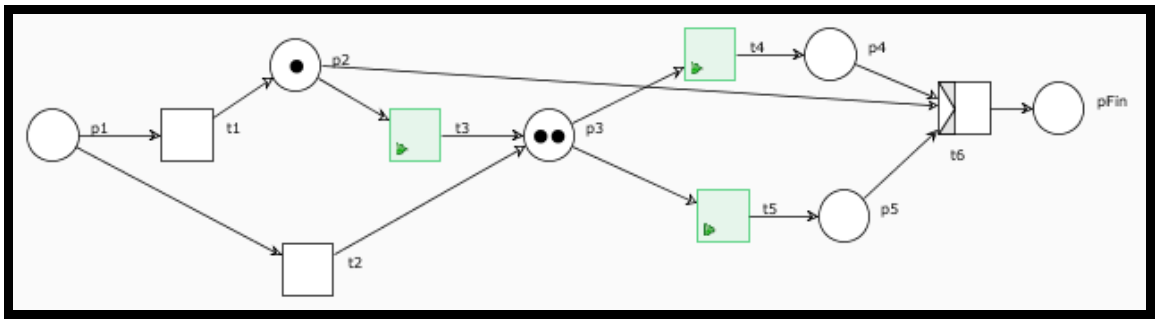


Figura 157 Red de petri comportamiento no valido 4 para la creación de módulos Forum.

3.7.6.2 Validación en Creación del Módulo Chat

Se tienen los siguientes nodos y transiciones:

P = {p1, p2, p3, p4, p5, pFin}

T = {t1, t2, t3, t4, t5, t6}

p1 = Esperando elementos en el Canvas.

p2 = Nodo Communications creado en el Canvas.

p3 = Nodo Chat creado en el Canvas.

p4 = Dato roomName asignado al Chat.

p5 = Dato sendText asignado al Chat.

pFin = Módulo Chat desplegado en la plataforma.

t1 = Se selecciono el nodo Communications de la paleta de herramientas y se coloco en el Canvas.

t2 = Se selecciono el nodo Chat de la paleta de herramientas y se coloco en el Canvas.

t3 = Se selecciono la conexión Communications -> Chat de la paleta de herramientas y se conecto el nodo Communications con el nodo Chat.

t4 = Se asigno información al campo roomName del nodo Chat.

t5 = Se asigno información al campo sendText del nodo Chat.

t6 = Se realizan las debidas transformaciones y despliegue sobre la plataforma LMS.

Red de petri para la creación de Módulo Chat.

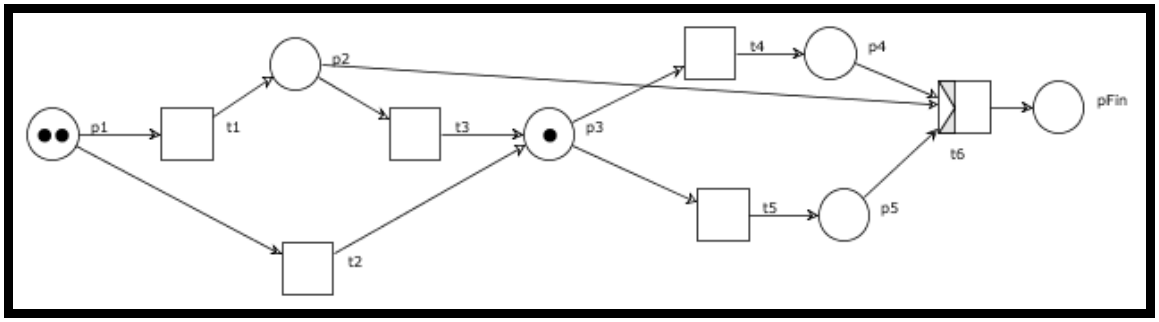


Figura 158 Red de petri para la creación de módulos Chat.

Comportamiento valido.

Prueba1 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, <p3, t4>, <p3, t5>, <t4, p4>, <t4, p5>, <t6, pFin>,1}

En esta prueba se obtuvo como resultado 1, lo que indica que este comportamiento se da en el modelo, y por consiguiente cumple con el requisito. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

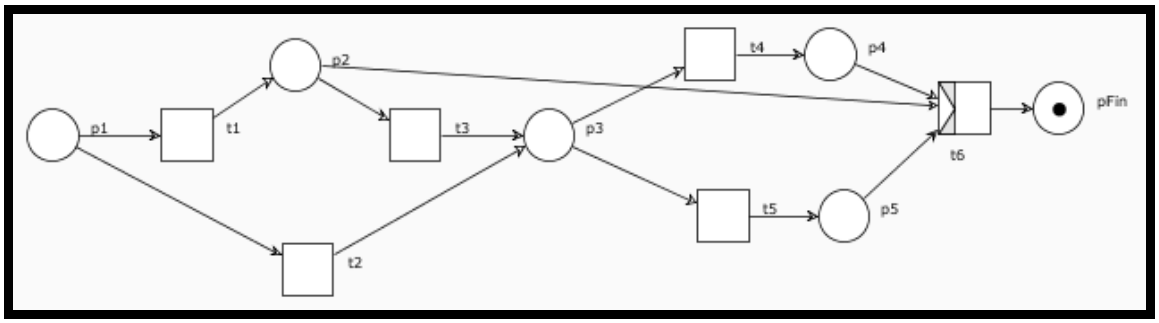


Figura 159 Red de petri comportamiento valido para la creación de módulos Chat.

Comportamiento no valido.

Prueba2 = {<p1, t2>, <t2, p3>, <p3, t4>, <t4, p4>, <p3, t5>, <t5, p5>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

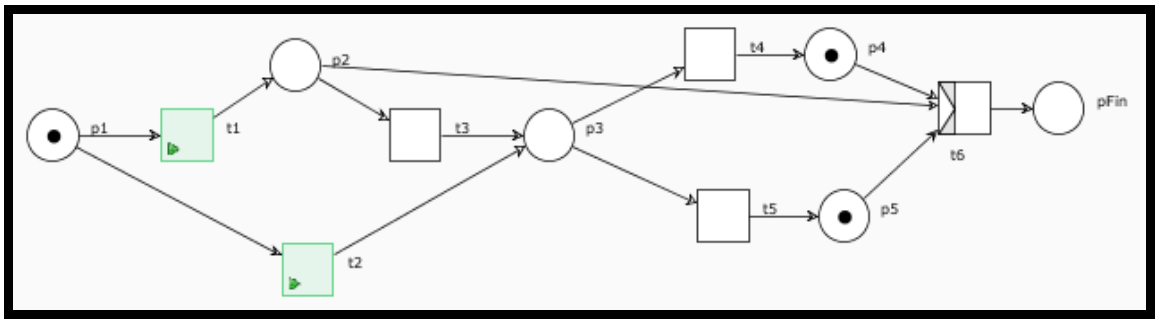


Figura 160 Red de petri comportamiento no valido 1 para la creación de módulos Chat.

Prueba3 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, <p3, t4>, <t4, p4>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

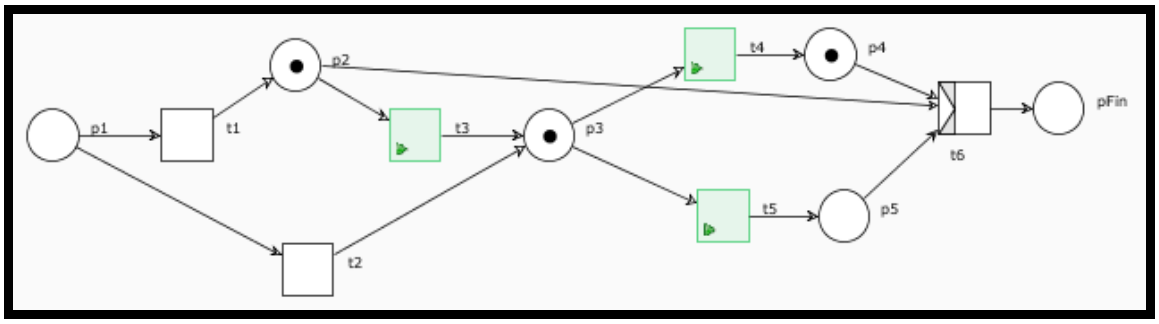


Figura 161 Red de petri comportamiento no valido 2 para la creación de módulos Chat.

Prueba4 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, <p3, t5>, <t5, p5>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

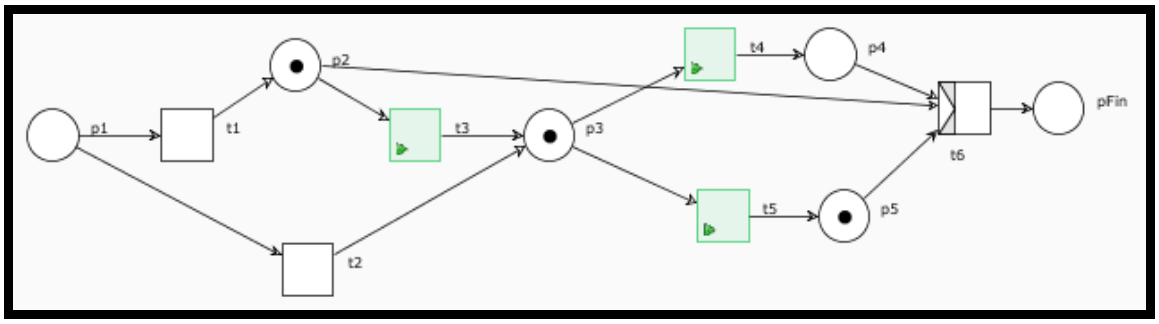


Figura 162 Red de petri comportamiento no valido 3 para la creación de módulos Chat.

Prueba5 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

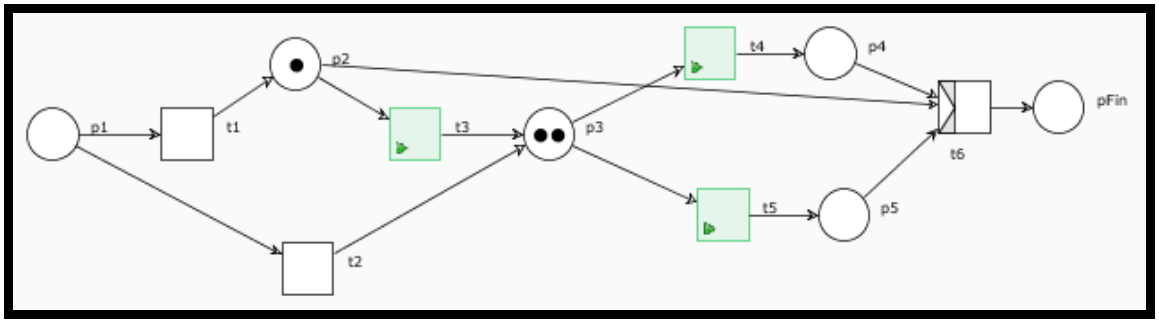


Figura 163 Red de petri comportamiento no valido 4 para la creación de módulos Chat.

3.7.6.3 Validación en Creación del Módulo Wiki

Se tienen los siguientes nodos y transiciones:

P = {p1, p2, p3, p4, p5, pFin}

T = {t1, t2, t3, t4, t5, t6}

p1 = Esperando elementos en el Canvas.

p2 = Nodo Communications creado en el Canvas.

p3 = Nodo Wiki creado en el Canvas.

p4 = Dato wikiName asignado al Chat.

p5 = Dato wikiBody asignado al Chat.

pFin = Módulo Wiki desplegado en la plataforma.

t1 = Se selecciono el nodo Communications de la paleta de herramientas y se coloco en el Canvas.

t2 = Se selecciono el nodo Wiki de la paleta de herramientas y se coloco en el Canvas.

t3 = Se selecciono la conexión Communications -> Wiki de la paleta de herramientas y se conecto el nodo Communications con el nodo Wiki.

t4 = Se asigno información al campo wikiName del nodo Wiki.

t5 = Se asigno información al campo wikiBody del nodo Wiki.

t6 = Se realizan las debidas transformaciones y despliegue sobre la plataforma LMS.

Red de petri para la creación de Módulo Wiki.

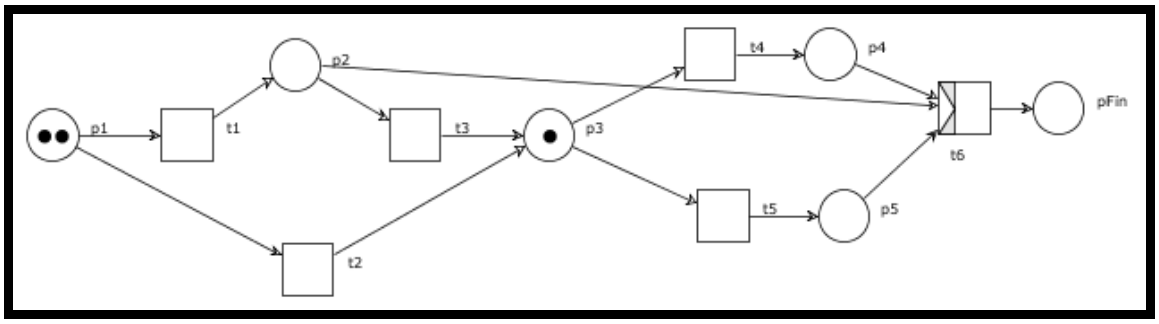


Figura 164 Red de petri para la creación de módulos Wiki.

Comportamiento valido.

Prueba1 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, <p3, t4>, <p3, t5>, <t4, p4>, <t4, p5>, <t6, pFin>,1}

En esta prueba se obtuvo como resultado 1, lo que indica que este comportamiento se da en el modelo, y por consiguiente cumple con el requisito. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

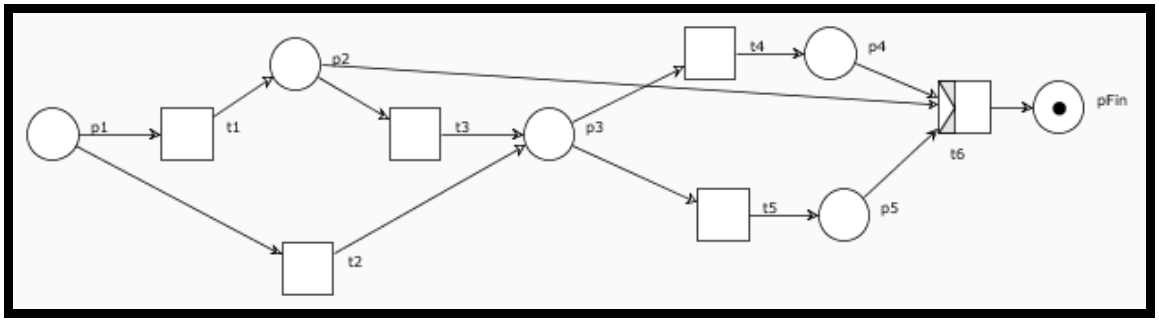


Figura 165 Red de petri comportamiento valido para la creación de módulos Wiki.

Comportamiento no valido.

Prueba2 = {<p1, t2>, <t2, p3>, <p3, t4>, <t4, p4>, <p3, t5>, <t5, p5>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la

red de petri, simulando el comportamiento y las funciones de entrada y salida en cada transición.

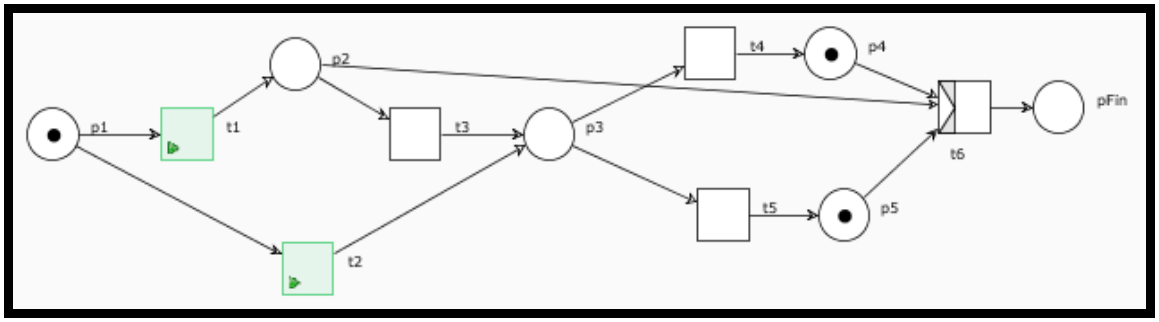


Figura 166 Red de petri comportamiento no valido 1 para la creación de módulos Wiki.

Prueba3 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, <p3, t4>, <t4, p4>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

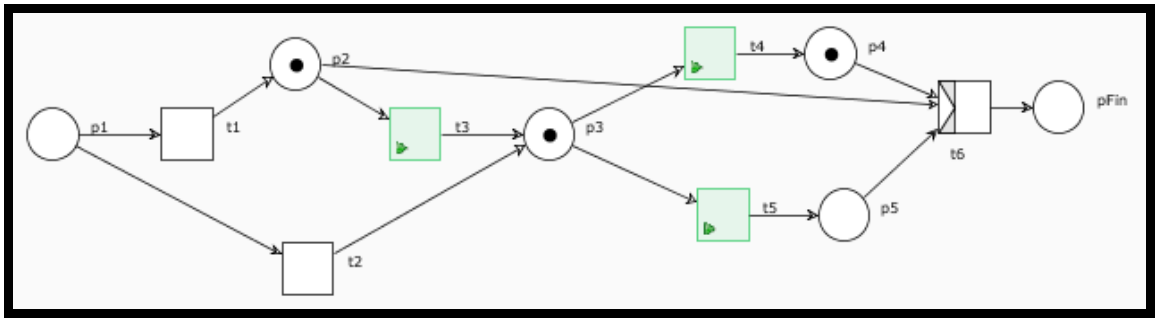


Figura 167 Red de petri comportamiento no valido 2 para la creación de módulos Wiki.

Prueba4 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, <p3, t5>, <t5, p5>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

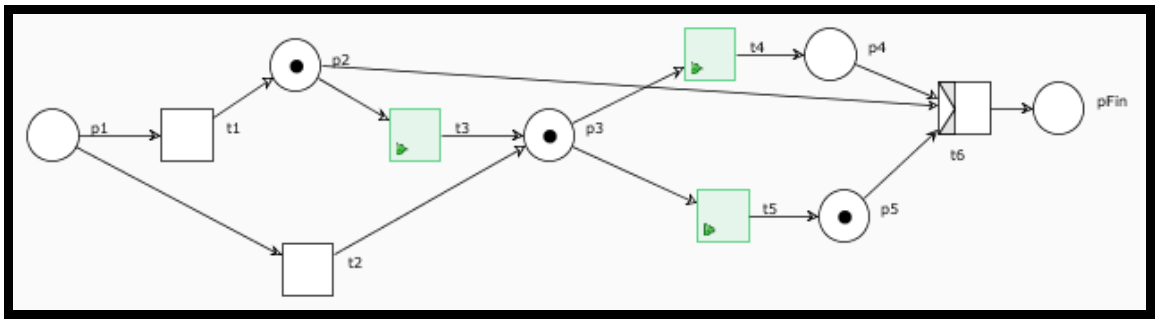


Figura 168 Red de petri comportamiento no valido 3 para la creación de módulos Wiki.

Prueba5 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

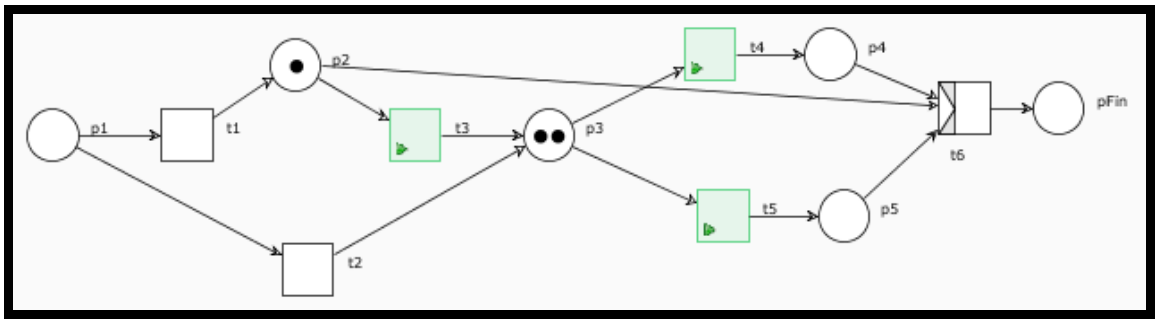


Figura 169 Red de petri comportamiento no valido 4 para la creación de módulos Wiki.

3.7.6.4 Validación en Creación del Módulo Announcement

Se tienen los siguientes nodos y transiciones:

P = {p1, p2, p3, p4, p5, pFin}

T = {t1, t2, t3, t4, t5, t6}

p1 = Esperando elementos en el Canvas.

p2 = Nodo Communications creado en el Canvas.

p3 = Nodo Announcements creado en el Canvas.

p4 = Dato announcementText asignado al Announcements.

pFin = Módulo Announcements desplegado en la plataforma.

t1 = Se selecciono el nodo Communications de la paleta de herramientas y se coloco en el Canvas.

t2 = Se selecciono el nodo Announcements de la paleta de herramientas y se coloco en el Canvas.

t3 = Se selecciono la conexión Communications -> Announcements de la paleta de herramientas y se conecto el nodo Communications con el nodo Announcements.

t4 = Se asigno información al campo announcementText del nodo Announcements.

T5 = Se realizan las debidas transformaciones y despliegue sobre la plataforma LMS.

Red de petri para la creación de Módulo Announcement.

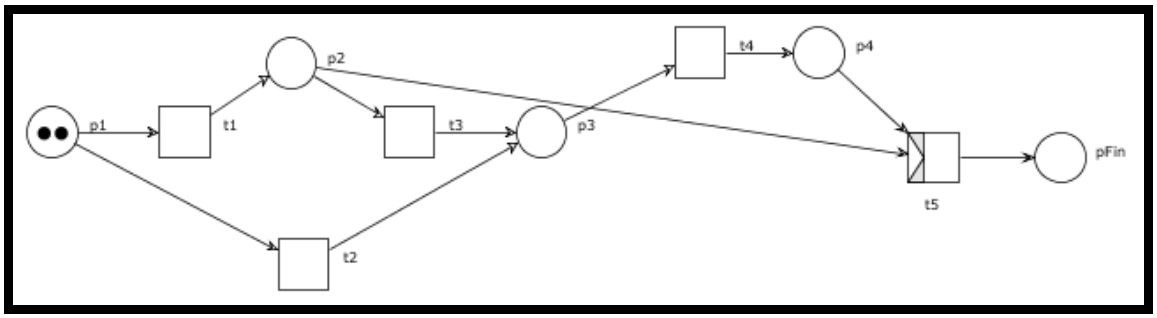


Figura 170 Red de petri para la creación de módulos Announcement.

Comportamiento valido.

Prueba1 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, <p3, t4>, <t4, p4>, <t5, pFin>,1}

En esta prueba se obtuvo como resultado 1, lo que indica que este comportamiento se da en el modelo, y por consiguiente cumple con el requisito. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

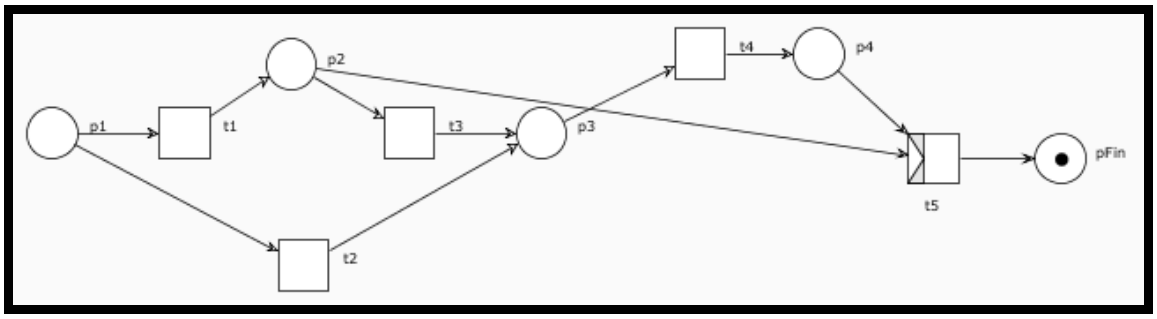


Figura 171 Red de petri comportamiento valido 4 para la creación de módulos Announcement.

Comportamiento no valido.

Prueba2 = {<p1, t2>, <t2, p3>, <p3, t4>, <t4, p4>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

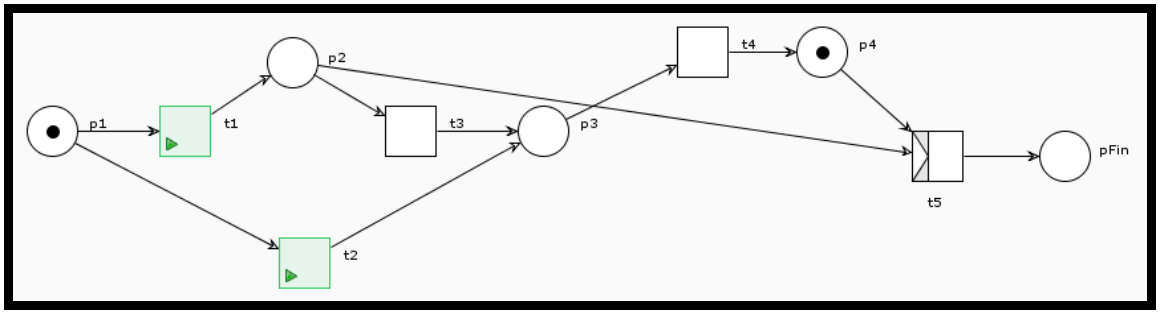


Figura 172 Red de petri comportamiento no valido 1 para la creación de módulos Annoncement.

Prueba3 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

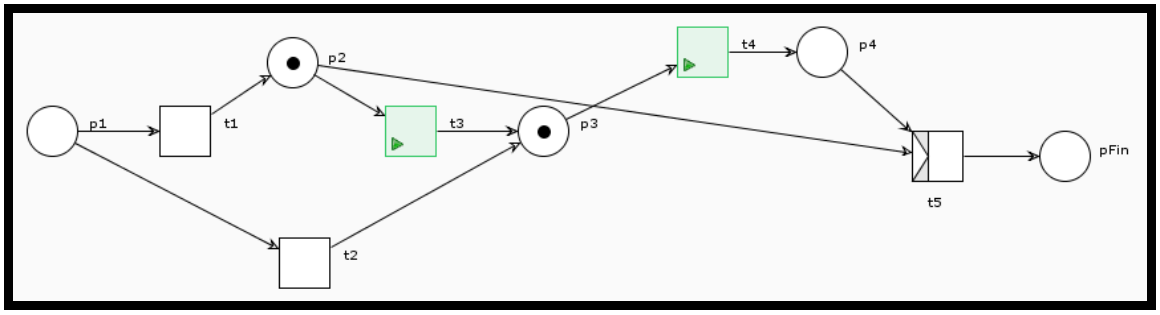


Figura 173 Red de petri comportamiento no valido 2 para la creación de módulos Announcement.

3.7.6.5 Validación en Creación del Módulo News

Se tienen los siguientes nodos y transiciones:

P = {p1, p2, p3, p4, p5, pFin}

T = {t1, t2, t3, t4, t5, t6}

p1 = Esperando elementos en el Canvas.

p2 = Nodo Communications creado en el Canvas.

p3 = Nodo News creado en el Canvas.

p4 = Dato newsTitle asignado al News.

p5 = Dato newsBody asignado al News.

pFin = Módulo News desplegado en la plataforma.

t1 = Se selecciono el nodo Communications de la paleta de herramientas y se coloco en el Canvas.

t2 = Se selecciono el nodo News de la paleta de herramientas y se coloco en el Canvas.

t3 = Se selecciono la conexión Communications -> News de la paleta de herramientas y se conecto el nodo Communications con el nodo News.

t4 = Se asigno información al campo newsTitle del nodo News.

t5 = Se asigno información al campo newsBody del nodo News.

t6 = Se realizan las debidas transformaciones y despliegue sobre la plataforma LMS.

Red de petri para la creación de Módulo News.

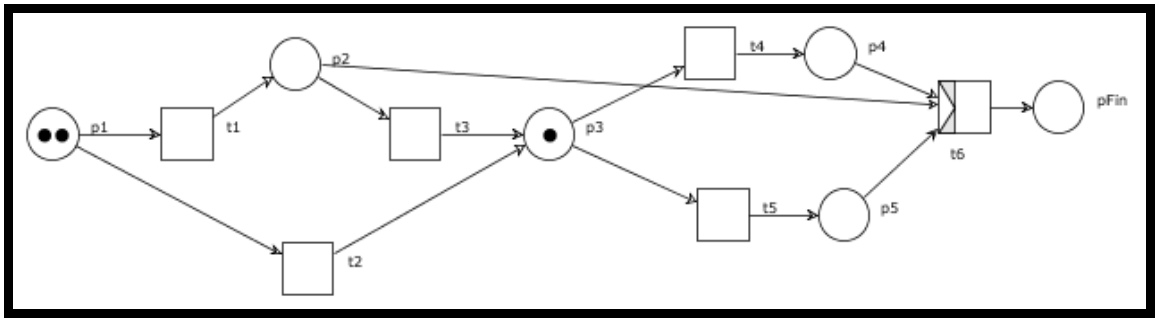


Figura 174 Red de petri para la creación de módulos News.

Comportamiento valido.

Prueba1 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, <p3, t4>, <p3, t5>, <t4, p4>, <t4, p5>, <t6, pFin>,1}

En esta prueba se obtuvo como resultado 1, lo que indica que este comportamiento se da en el modelo, y por consiguiente cumple con el requisito. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

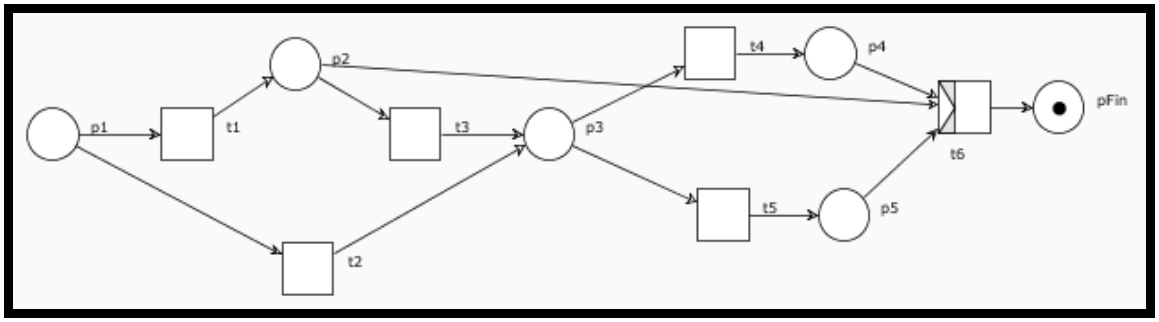


Figura 175 Red de petri comportamiento valido para la creación de módulos News.

Comportamiento no valido.

Prueba2 = {<p1, t2>, <t2, p3>, <p3, t4>, <t4, p4>, <p3, t5>, <t5, p5>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

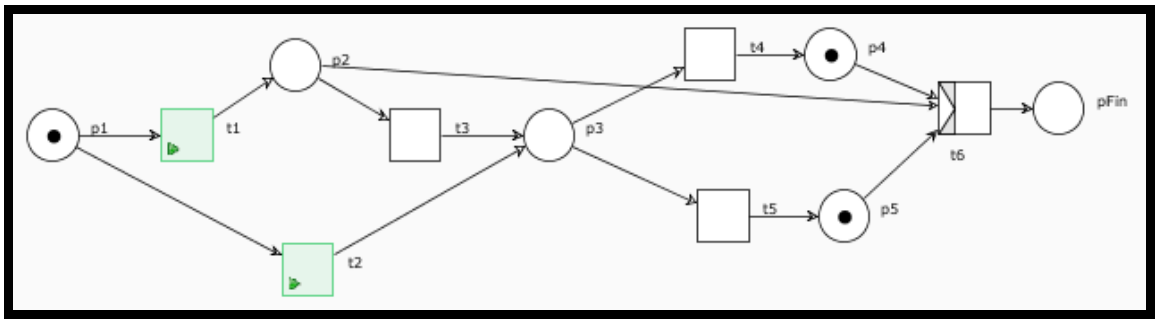


Figura 176 Red de petri comportamiento no valido 1 para la creación de módulos News.

Prueba3 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, <p3, t4>, <t4, p4>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

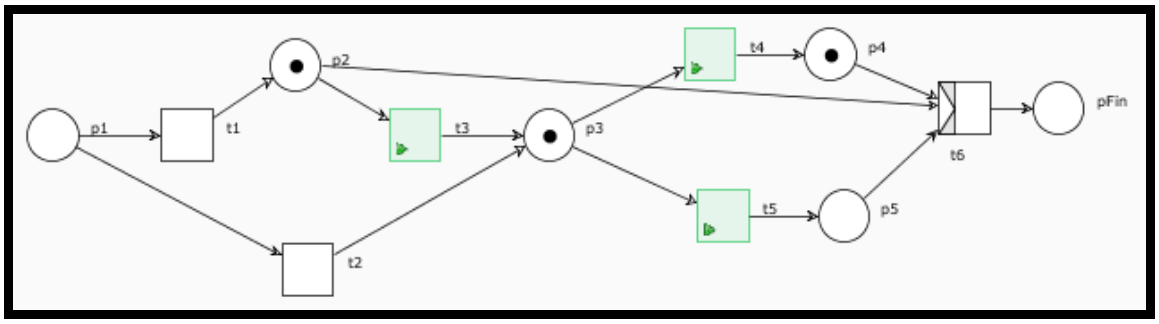


Figura 177 Red de petri comportamiento no valido 2 para la creación de módulos News.

Prueba4 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, <p3, t5>, <t5, p5>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

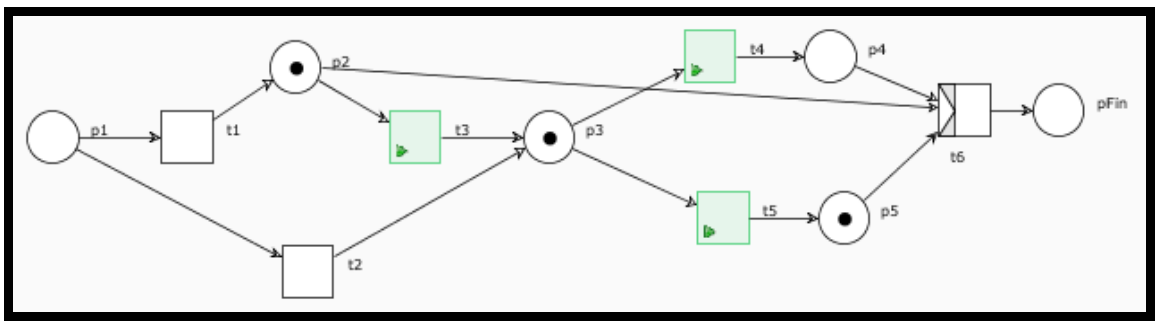


Figura 178 Red de petri comportamiento no valido 3 para la creación de módulos News.

Prueba5 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

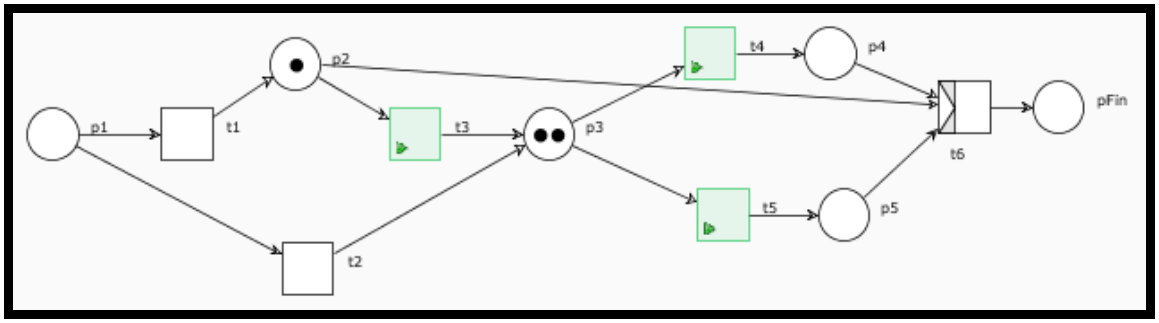


Figura 179 Red de petri comportamiento no valido 4 para la creación de módulos News.

3.7.6.6 Validación en Creación del Módulo Note

Se tienen los siguientes nodos y transiciones:

P = {p1, p2, p3, p4, p5, pFin}

T = {t1, t2, t3, t4, t5, t6}

p1 = Esperando elementos en el Canvas.

p2 = Nodo Communications creado en el Canvas.

p3 = Nodo Note creado en el Canvas.

p4 = Dato sendTo asignado al Note.

p5 = Dato message asignado al Note.

pFin = Módulo Note desplegado en la plataforma.

t1 = Se selecciono el nodo Communications de la paleta de herramientas y se coloco en el Canvas.

t2 = Se selecciono el nodo Note de la paleta de herramientas y se coloco en el Canvas.

t3 = Se selecciono la conexión Communications -> Note de la paleta de herramientas y se conecto el nodo Communications con el nodo Note.

t4 = Se asigno información al campo sendTo del nodo Note.

t5 = Se asigno información al campo message del nodo Note.

t6 = Se realizan las debidas transformaciones y despliegue sobre la plataforma LMS.

Red de petri para la creación de Módulo Note.

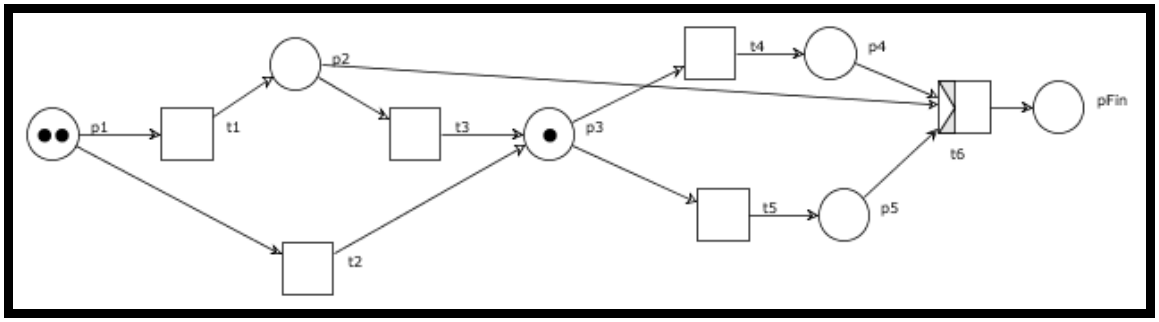


Figura 180 Red de petri para la creación de módulos Note.

Comportamiento valido.

Prueba1 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, <p3, t4>, <p3, t5>, <t4, p4>, <t4, p5>, <t6, pFin>,1}

En esta prueba se obtuvo como resultado 1, lo que indica que este comportamiento se da en el modelo, y por consiguiente cumple con el requisito. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

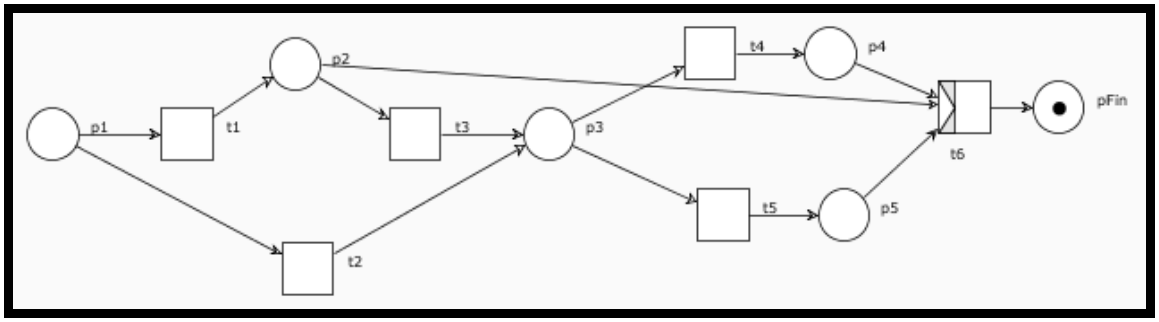


Figura 181 Red de petri comportamiento valido para la creación de módulos Note.

Comportamiento no valido.

Prueba2 = {<p1, t2>, <t2, p3>, <p3, t4>, <t4, p4>, <p3, t5>, <t5, p5>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

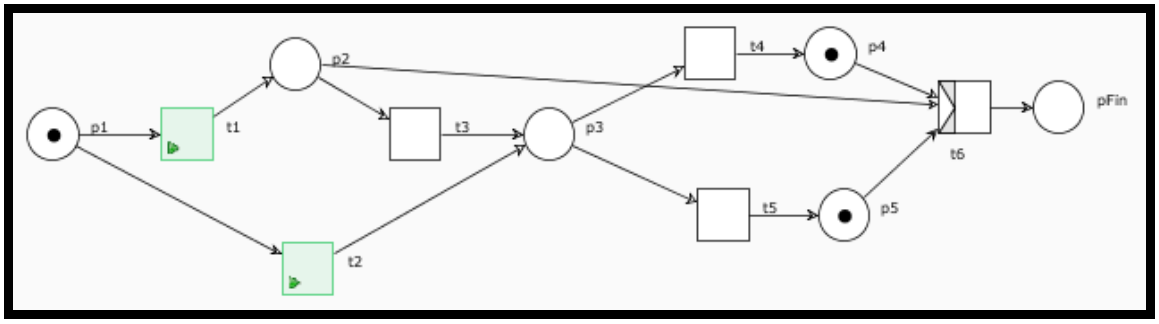


Figura 182 Red de petri comportamiento no valido 1 para la creación de módulos Note.

Prueba3 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, <p3, t4>, <t4, p4>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

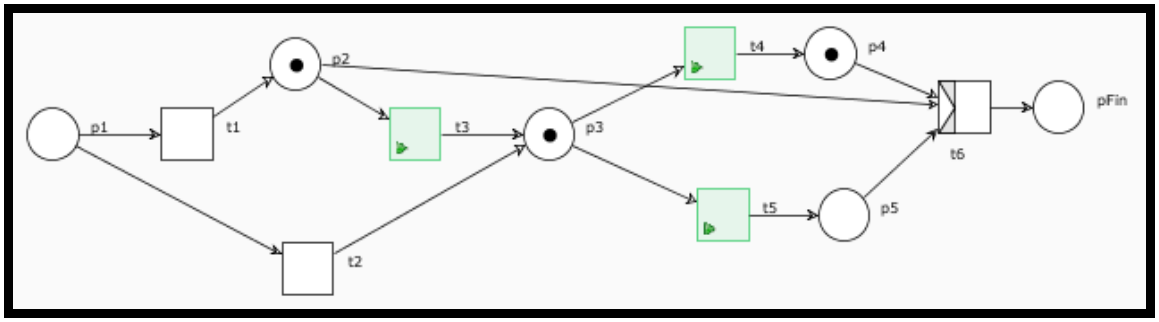


Figura 183 Red de petri comportamiento no valido 2 para la creación de módulos Note.

Prueba4 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, <p3, t5>, <t5, p5>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

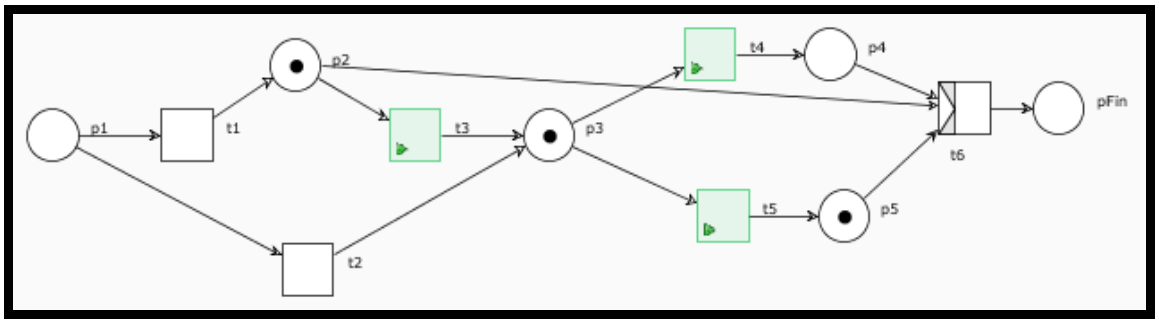


Figura 184 Red de petri comportamiento no valido 3 para la creación de módulos Note.

Prueba5 = {<p1, t1>, <t1, p2>, <p1, t2>, <t2, p3>, 0}

En esta prueba se obtuvo como resultado 0, lo que indica que este comportamiento no se da en el modelo. A continuación se presenta gráficamente el final de la prueba en la *red de petri*, simulando el comportamiento y las funciones de entrada y salida en cada transición.

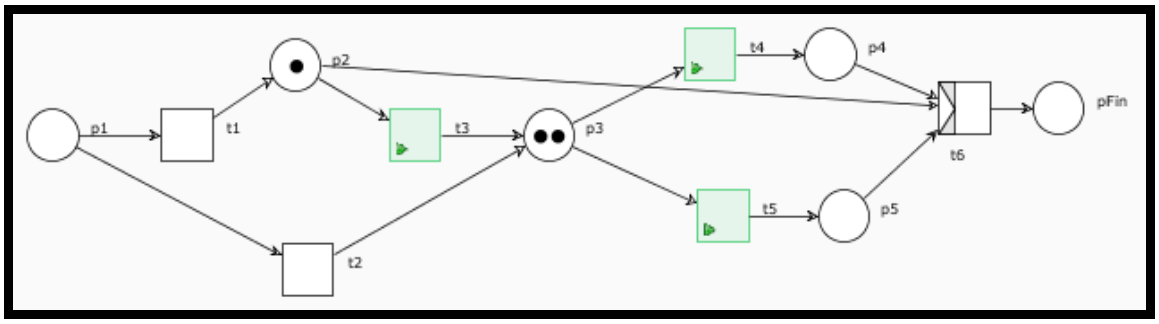


Figura 185 Red de petri comportamiento no valido 4 para la creación de módulos Note.

4 Conclusiones e investigación futura

1. La sección 3.1 muestra una forma de modelar módulos de un sistema de gestión del aprendizaje con la utilización de mapas de conocimiento y basados en ese modelamiento determinar cuáles módulos son comunes en diferentes sistemas de gestión del aprendizaje. Este mecanismo de modelamiento y determinación de similaridad puede ser aplicable en diversos contextos.

2. Tomando como referente la metodología definida por los autores de protegé acerca de cómo construir una ontología, en la sección 3.2 se ha creado una ontología para sistemas de gestión del aprendizaje que hace énfasis en los elementos del nivel A del IMS-LD para un LMS. El punto de partida para la construcción de esta ontología fueron los modelos obtenidos con mapas de conocimiento sobre diversas plataformas LMS, trabajo que se había realizado previamente.

3. Esta tesis es el punto de partida para la construcción de un metamodelo LMS más completo, que integre mas módulos y plataformas LMS, de igual forma se podría realizar con la definición y validación del metamodelo planteado, un proceso inverso en el cual, los módulos que ya estén creados en un LMS especifico se conviertan a instancias del metamodelo y de esta forma se podría hacer un proceso de transformación a cualquier otra plataforma LMS.

4. En este trabajo se identifico una similitud entre el análisis del dominio hecho para ontologías con el de metamodelos, aquí se planteo una ontología y se utilizo como insumo principal para la construcción del metamodelo, haciendo el proceso de conversión de la ontología al metamodelo manualmente, pero este proceso podría llegar a automatizarce y allí encontramos otro nicho de investigación futura.

5. Se ha proporcionado una herramienta DSL que se basa en el metamodelo para sistemas de gestión del aprendizaje creado aquí, la herramienta funciona para la creación de módulos independientes de la plataforma y solo cubre algunos los módulos de comunicaciones. Las pruebas a la herramienta han demostrado que los tiempos y esfuerzos necesarios en la creación de diversos módulos en varias plataformas LMS, se minimizan en más del 50% y que este rendimiento mejora a medida que se utiliza más en la plataforma.

6. En esta tesis se han explorado los niveles M0, M1, M2 y M3 planeados en la ingeniería dirigida por modelos, El metamodelo LMS construido representa el nivel M2, el meta-metamodelo Ecore es el nivel M3, el resultado de modelar los módulos con la herramientas DSL construida representa al nivel M1 y finalmente, las conversiones y despliegue de esos módulos construidos en las plataformas moodle, atutor y claroline representan el nivel M0, de esta forma hemos explorado en un caso práctico todos los niveles planeados en MDE,

más adelante se pretende comparar las tecnologías utilizadas para este trabajo con otras de uso comercial como DSLtools de Microsoft.

7. El alcance de esta propuesta se limito a la utilización de MDE para crear un modelo independiente de la plataforma y lograr un despliegue real de este sobre varias plataformas LMS, una propuesta que queda abierta a trabajar; seria partiendo de un curso ya implementado en un LMS y aplicando las debidas transformaciones obtener las instancias de este curso acordes al metamodelo planteado, para finalmente aplicar lo hecho hasta el momento y desplegarlo sobre otras plataformas LMS.

8. Respecto al trabajo hecho en la construcción de la herramienta DSL, se ha proporcionado una solución a la problemática de poseer muchos conectores en la herramienta, aplicando un mecanismo que adapta el patrón de diseño Fabrica Abstracta a los metamodelos y logra eliminar muchos conectores, proporcionando los mínimos posibles, se deja abierta la posibilidad de plantear mas mecanismos que se apoyen en los patrones actuales de la orientación a Objetos aplicables a los problemas en los metamodelos.

9. Basándose en los resultados de las pruebas, se puede afirmar que MDE es aplicable en el contexto de diseño de cursos para LMS y que reduce significativamente el tiempo y esfuerzo en la construcción y despliegue de cursos sobre plataformas LMS.

10. Una de las áreas más claras para seguir avanzando en la madures de este trabajo, es la integración de mas plataformas LMS, lo único que se debe hacer es adaptar las plantillas de transformación de modelo a texto (M2T) que están hechas en MOFScript, a las necesidades de cada LMS, claro para ello es necesario realizar un proceso exploratorio sobre cada plataforma LMS.

11. Respecto a la validación del metamodelo, las redes de petri son una excelente opción, que permite validar no solo metamodelos si no que es aplicable a diversos contextos.

12. Se han proporcionado mecanismos formales para realizar pruebas y análisis a una herramienta DSL, que se pueden complementar según los criterios que se requieran medir.

13. Dada la necesidad que no se encontró una guía consolidada para la aplicación de MDE, se ha generado un manual muy completo para la utilización de EMF, GMF y MOFScript en Eclipse, este manual proporciona una guía muy detallada sobre la aplicación de MDE con eclipse a un caso hipotético de una librería.

14. Durante el desarrollo de la propuesta se puedo observar que los tiempos de aprendizaje en el desarrollo de módulos con modelos en cada iteración se fueron reduciendo, notando que cuando se creó la primera versión finalizada

para la plataforma Moodle, en comparación con la segunda versión finalizada para Atutor, el tiempo fue aproximadamente la mitad y luego para la tercera plataforma Claroline, el tiempo disminuyo aun mas, esto indica que a medida que se conoce más el enfoque de desarrollo dirigido por modelos, los tiempos disminuyen, en otras palabras a medida que el grado de conocimiento sobre MDE crece, los tiempos de desarrollo disminuyen.

5 Aportaciones

Como parte del desarrollo de esta tesis se han generado los siguientes tipos de aportaciones:

Publicaciones:

El proceso de divulgación ha sido continuo en este trabajo y se han logrado algunas publicaciones como:

Plataforma de seguridad basado en autenticidad de contenidos sobre conjunto de especificaciones SCORM, artículo publicado en la revista ingeniería y competitividad de la universidad del valle (Colombia), con ISSN 0123-3033 que está clasificada en Publindex como grado A2, el artículo trata sobre los mecanismos que se presentan en la mayoría de plataformas Learning Content Management Systems, (LCMS), no permiten evaluar el concepto de "autenticidad" en contenidos que se comparten en cada una de ellas con un buen grado de aceptación, por lo tanto, el siguiente artículo tiene como finalidad plantear un Modelo de seguridad informático sobre plataformas de aprendizaje virtual LCMS, mediante sus contenidos a través de las especificaciones dadas por Sharable Content Object Reference (SCORM), lo cual permita garantizar la autenticidad de contenidos mediante conceptos de firmas digitales e identificación de protocolos y mecanismos que garanticen este tipo de actividades. Para llevar a cabo esta actividad, se plantean alternativas de modelos de seguridad, partiendo por el análisis de los mecanismos de seguridad informáticos que se trabajan actualmente sobre la mayoría de plataformas LCMS, al igual que la identificación de componentes y variables desde el punto de vista del desarrollo de contenidos mediante las especificaciones SCORM. Finalmente se propone un modelo de seguridad para su implementación sobre el desarrollo de objetos de aprendizaje que permitan identificar niveles de confianza en los diferentes contenidos que se comparten dentro de una Plataforma LCMS.

Modeling and comparison study of modules in open source lms platforms with campstool, artículo publicado en International Journal of Artificial Intelligence and Interactive Multimedia, con ISSN 1989-1660, este trata sobre la realización de una comparación entre diversos LMSs con el fin de conseguir una primera aproximación de los módulos comunes entre ellos, y de así iniciar la construcción de la Ontología compatible con todos los LMS.

Definition of trust levels in virtual learning platforms through semantic language, artículo publicado en International Journal of Artificial Intelligence and Interactive Multimedia, con ISSN 1989-1660, allí se plantea una estrategia que desde el punto de vista semántico permita identificar las variables necesarias para identificar los niveles de confianza de los contenidos que se publican y comparten en una plataforma

LCMS (Learning Content Management System), lo que permite valorar los contenidos antes de ser publicados en un espacio virtual.

Comparison of modules between different open source lms platforms through knowledge maps, artículo aceptado en International Journal of Engineering and Industries, con ISSN: 2093-5641, en el cual se presenta un estudio realizado sobre cinco plataformas LMS y determina cuales son los módulos y características de estos comunes entre las plataformas LMS, el estudio se realizo indagando en la arquitectura para generar un mapa de conocimiento por cada plataforma LMS.

Herramienta de lenguaje de dominio específico (DSL) para la creación de módulos en sistemas de gestión del aprendizaje (LMS) con despliegue sobre moodle, artículo presentado en la sexta conferencia ibérica de sistemas y tecnologías de la información, a celebrar entre el 15 y 18 de Junio de 2011, en Chaves, Portugal, en este articulo se muestra un metamodelo bajo ecore, después la construcción de una herramienta DSL (Domain Specific Languaje) grafica basada en ese metamodelo y con apoyo en Eclipse Modeling Framework (EMF), Meta Object Facility (MOF) y Graphical Modeling Framework (GMF). Finalmente se muestra como transformar un modelo construido con el DSL a código desplegable para la plataforma moodle en este caso, esto se hará con MOFScript. Al terminar el artículo se realizara una discusión de las pruebas de la herramienta.

Modelado especifico de dominio para la construcción de módulos de LMS independientes de la plataforma, artículo presentado en el simposio doctoral de la sexta conferencia ibérica de sistemas y tecnologías de la información, a celebrar entre el 15 y 18 de Junio de 2011, en Chaves, Portugal, allí se plantea realizar una modelado específico de dominio para la construcción de módulos para plataformas LMS independientes de la plataforma.

Domain specific language for the generation of learning management systems modules, artículo enviado a journal of systems and software, con ISSN 0164-1212 y factor de impacto 1.340 según Thomson Reuters Journal Citation Reports 2009, en este artículo se habla sobre la creación de un lenguaje especifico de dominio para la generación de módulos en sistemas de gestión del aprendizaje, aplicando conceptos de web semántica e ingeniería dirigida por modelos.

KiwiDSM: Visual tool using a Domain Specific Modeling for the construction of LMS modules independent of the platform, artículo enviado a journal of visual languages and computing, con ISSN 1045-926X y factor de impacto 1.082 según Thomson Reuters Journal Citation Reports 2009, allí se habla sobre la creación de una herramienta de modelado de dominio especifico para la creación de modulos de un LMS y su despliegue sobre tres plataformas moodle, claroline y atutor.

Generation of metamodel in Ecore with start point in an ontology for Learning Management Systems (LMS), artículo enviado a Journal of Web Engineering, con

ISSN 1540-9589 y factor de impacto 0.531 según Thomson Reuters Journal Citation Reports 2009, aquí se habla sobre la creación de un metamodelo basado en ecore que tiene como punto de partida una ontología previamente creada, el metamodelo creado es global y compatible con todos los LMSs estudiados.

Designing and deployment of platform independent LMS modules using visual DSLs, artículo enviado a Journal Software & System Modeling, con ISSN 1619-1374 y factor de impacto 1.533 según Thomson Reuters Journal Citation Reports 2009, este articulo trata sobre Este articulo trata sobre el diseño y creación de módulos para LMSs independientes de la plataforma utilizando lenguajes visuales, se muestra el resultado final del Framework desarrollado con pruebas sobre varias plataformas LMS y un método de validación a través de redes de petri.

Aplicación de ingeniería dirigida por modelos (MDA), para la construcción de una herramienta de modelado de dominio especifico (DSM) para la creación de módulos en sistemas de gestión del aprendizaje (LMS) independientes de la plataforma, artículo aceptado en la revista DYNA, con ISSN 0012-7353 y factor de impacto 0.054 según Thomson Reuters Journal Citation Reports 2009, allí se plantea realizar un modelado específico de dominio para la construcción de módulos en sistemas de gestión del aprendizaje (LMS) independientes de la plataforma. Para esto, el punto de partida es un metamodelo para la construcción de un lenguaje específico de dominio (DSL) que con ingeniería dirigida por modelos (MDE) y aplicando las debidas transformaciones se consiga de un modelo independiente de la plataforma, el despliegue de este modelo se realizara en varias plataformas LMSs.

KiwiDSM: Herramienta de lenguaje de dominio específico e ingeniería dirigida por modelos para la creación de módulos independientes de la plataforma con despliegue sobre Atutor, articulo enviado a la Revista de Ingeniería de la Universidad Distrital “Francisco José de Caldas”, con ISSN 0121-750X, Este artículo presenta la creación de KiwiDSM: herramienta de lenguaje de dominio específico (DSL), que apoyada en ingeniería dirigida por modelos (MDE), sirve para modelar módulos que conforman un sistema de gestión del aprendizaje (LMS) en el área de comunicaciones, esta herramienta es independiente de la plataforma. Para la realización de pruebas, el despliegue del modelo hecho con KiwiDSM se hará sobre la plataforma Atutor. Estas pruebas evidenciaran que el trabajar con MDE reduce el tiempo y esfuerzo en él la creación y despliegue sobre Atutor de los módulos modelados, y que el metamodelo planteado cubre los conceptos manejados al menos en Atutor.

Herramienta de modelado de dominio específico (DSM) para la creación de módulos en sistemas de gestión del aprendizaje (LMS), articulo aceptado en la Revista científica de la Universidad Distrital “Francisco José de Caldas”, con ISSN 0124-2253, En este articulo se mostrara un metamodelo bajo ecore, después la construcción de una herramienta DSL (Domain Specific Languaje) grafica basada en ese metamodelo y con apoyo en Eclipse Modeling Framework (EMF), Meta Object Facility (MOF) y Graphical Modeling Framework (GMF). Finalmente se mostrara como transformar un

modelo construido con el DSL a código desplegable para la plataforma moodle en este caso, esto se hará con MOFScript.

Propuesta arquitectónica de servicios funcionales a través de una plataforma de computación grid para la universidad distrital. Ponencia presentada en el Latin American and Caribbean Conference for Engineering and Technology 7 º LACCEI Extended, con ISBN 0-9822896-2-6, realizado en San Cristóbal Venezuela en 2009.

Diseño y construcción de un portal web especializado en neumología para la unidad de cuidado intensivo del hospital de santa clara. Ponencia aceptada en el Latin American and Caribbean Conference for Engineering and Technology 7 º LACCEI Extended, con ISBN 0-9822896-2-6, realizado en San Cristóbal Venezuela en 2009

Modelo informático para autenticidad de contenidos mediante el concepto de web of trust sobre plataformas virtuales lcms. Ponencia presentada en el Latin American and Caribbean Conference for Engineering and Technology 8 º LACCEI Extended, realizado en Arequipa Perú en 2010.

Prototipo de telemedicina móvil para asistencia médica domiciliaria y remota. Ponencia presentada en el Latin American and Caribbean Conference for Engineering and Technology 8 º LACCEI Extended, realizado en Arequipa Perú en 2010.

El proceso de testing y la implementación de la herramienta mantis bugtracker en el proceso. Ponencia presentada en el V Simposio Internacional de Sistemas De Información E Ingeniería De Software En La Sociedad Del Conocimiento (SISOFT 2010), realizado en Bogotá Colombia en 2010.

Teóricas:

Se realizo el modelamiento y posterior comparación de cinco sistemas de gestión del aprendizaje (Moodle, ATutor, Claroline, Sakai y DotRLN), obteniendo como resultado final un cuadro comparativo de los módulos comunes que hay entre ellos, dicho modelamiento se realizo a través de un largo proceso de utilización y apoyados en la diagramación con mapas de conocimiento por cada sistema.

Se ha generado una ontología para sistemas de gestión del aprendizaje, que modela algunos módulos comunes en plataformas LMS, esta ontología utiliza la metodología par desarrollo de ontologías plateada por protegé, “Desarrollo de Ontologías-101: Guía Para Crear Tu Primera Ontología” (Natalya and Deborah, 2005).

Se ha creado un metamodelo basado en la ontología previamente creada, este metamodelo representa el nivel M2 planeado en la ingeniería dirigida por modelos, el meta-metamodelo ecore es el nivel M3 y es sobre lo que se ha construido el metamodelo LMS. Se ha elaborado una herramienta DSL llamada KiwiDSM que se

basa en el metamodelo anterior, la herramienta funciona para la creación de módulos independientes de la plataforma y solo cubre los módulos de comunicaciones: foros, chats, wikis, anuncios, noticias y mensajes o notas. El resultado de modelar los módulos con KiwiDSM representa al nivel M1de MDE, por último, las conversiones y despliegue de esos módulos construidos en las plataformas Moodle, ATutor y Claroline representan el nivel M0. De esta forma se han explorado en un caso práctico todos los niveles planeados en la ingeniería dirigida por modelos (MDE).

Se han proporcionado mecanismos para la realización de pruebas de tiempo y esfuerzo, que permiten analizar los resultados respecto a la eficiencia de trabajar con ingeniería dirigida por modelos, respecto a las formas tradicionales, en este caso aplicado a la creación de módulos para sistemas de gestión del aprendizaje y específicamente algunos módulos de comunicaciones. De igual forma se presenta una estrategia para validación de modelos con redes de petri, que puede ser aplicable a otros contextos.

Científico practicas:

Se ha generado KiwiDSM v2.0, que es una aplicacion de lenguaje de dominio especifico grafica, que permite el modelamiento y despliegue de módulos de comunicaciones para tres sistemas de gestión del aprendizaje, esta aplicación es escalable permitiendo de manera fácil integrar mas módulos o mas plataformas para su despliegue.

Se han proporcionado guías rápidas para el desarrollo de módulos en las plataformas de gestión del aprendizaje Moodle, ATutor y Claroline.

También se proporciona un manual que integra la creación de metamodelos sobre ecore, la construcción de herramientas de dominio especifico graficas en eclipse con EMF y GMF, incluyendo la instalación de los plugins necesarios. Este aporte se hace necesario ya que no existen manuales que integren todas estas tecnologías, por el contrario existe mucha documentación suelta de cada una.

6 Anexo I

Mapas De Conocimiento Por Cada LMS

Carlos Enrique Montenegro Marín

1. Mapa De Conocimiento Para Atutor

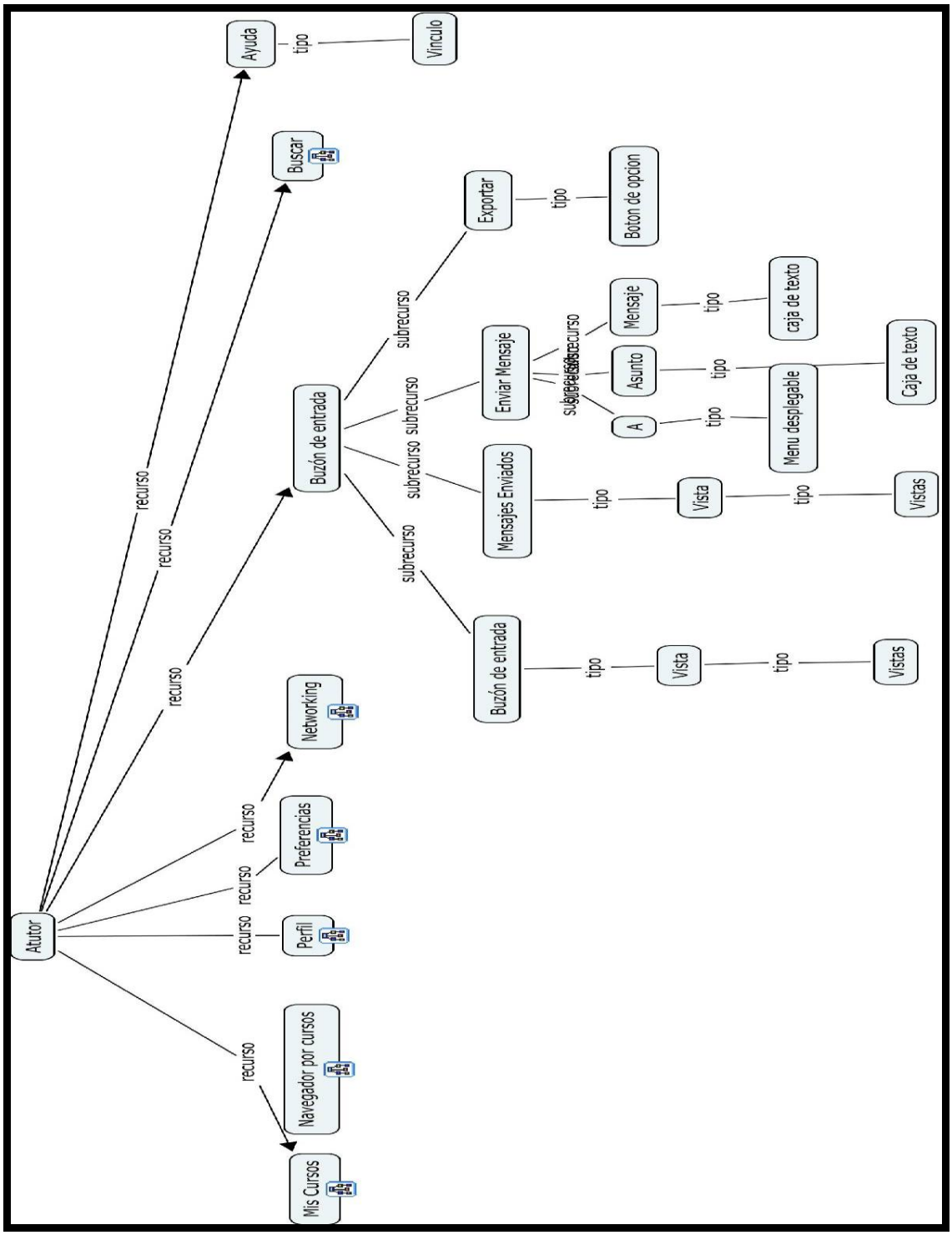


Figura 186 Vista principal con CmapTools de ATutor.

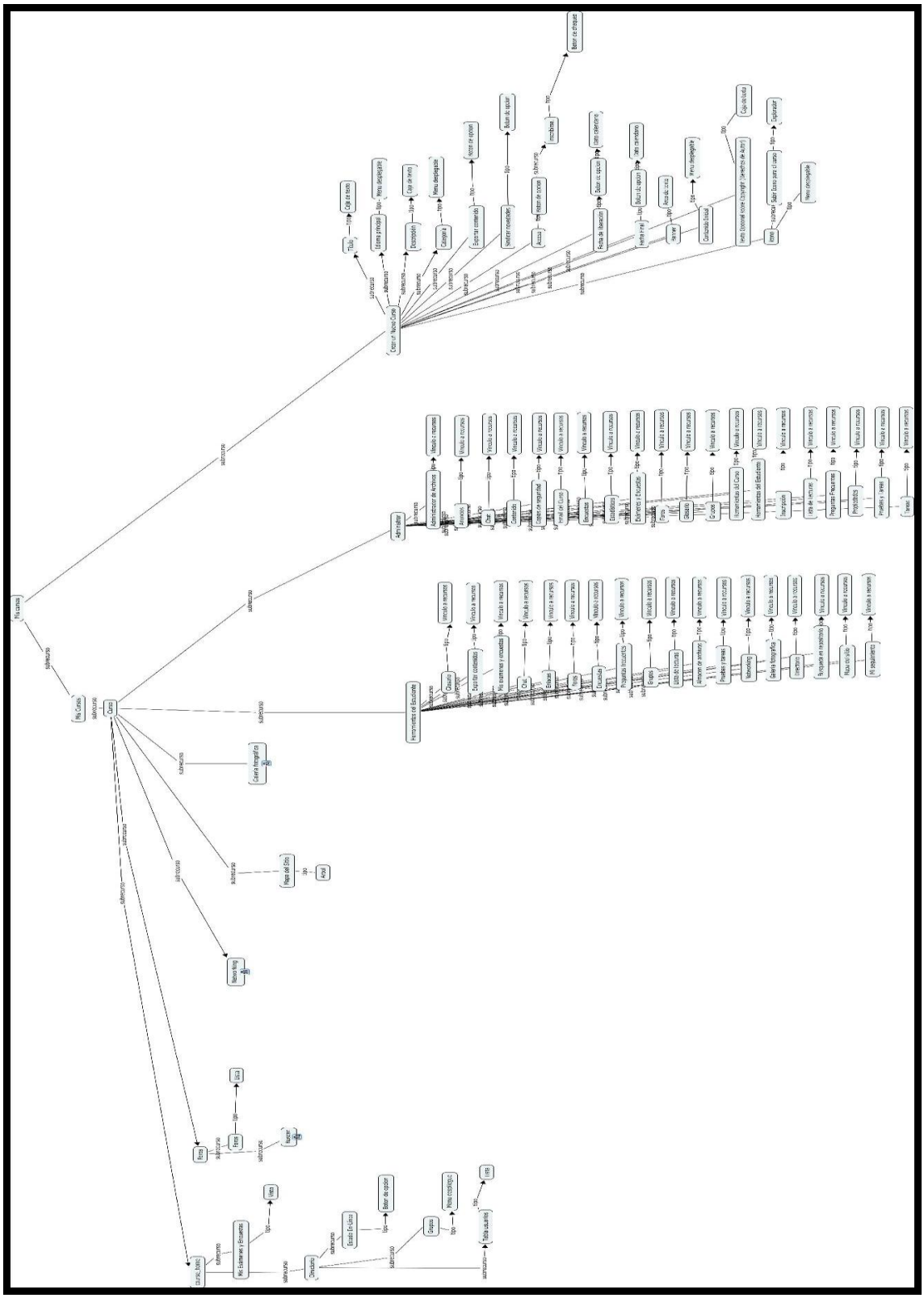


Figura 187 Vista recurso Mis Cursos con CmapTools de ATutor.

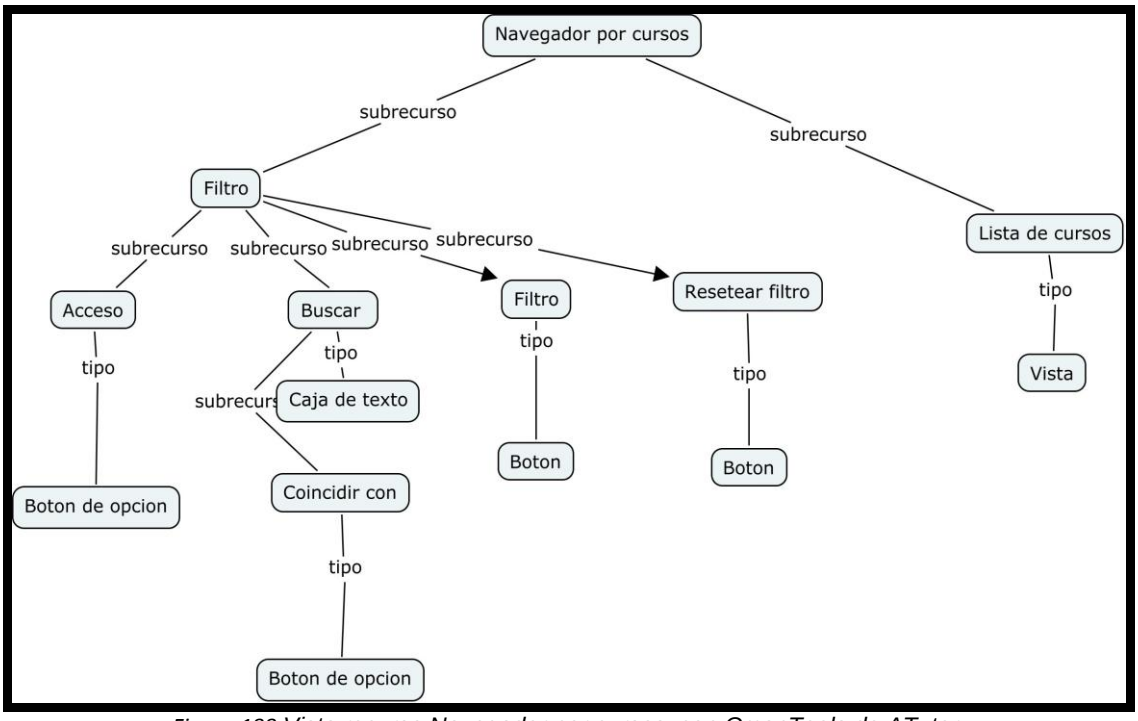


Figura 188 Vista recurso Navegador por cursos con CmapTools de ATutor.

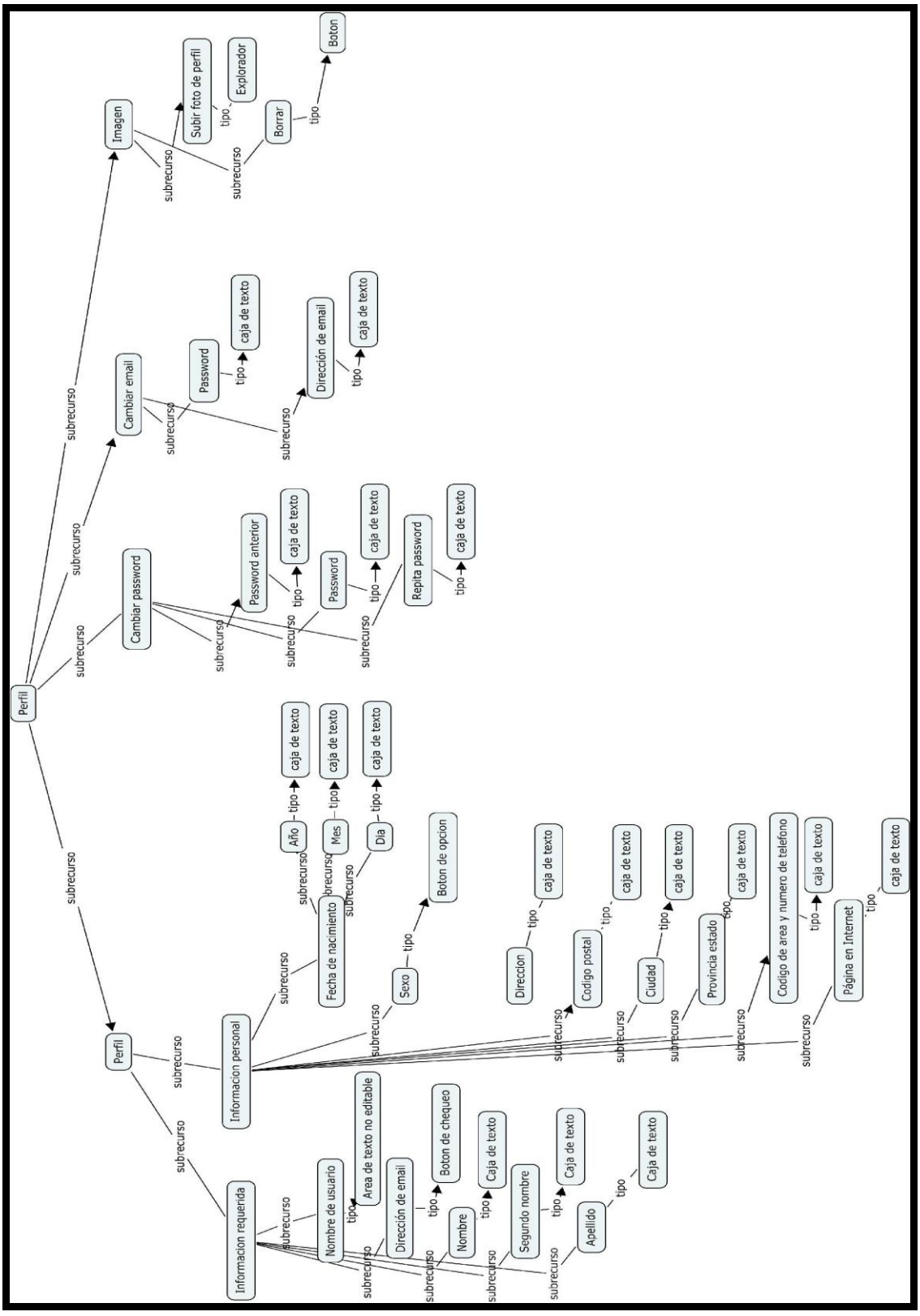


Figura 189 Vista recurso Perfil con CmapTools de ATutor.

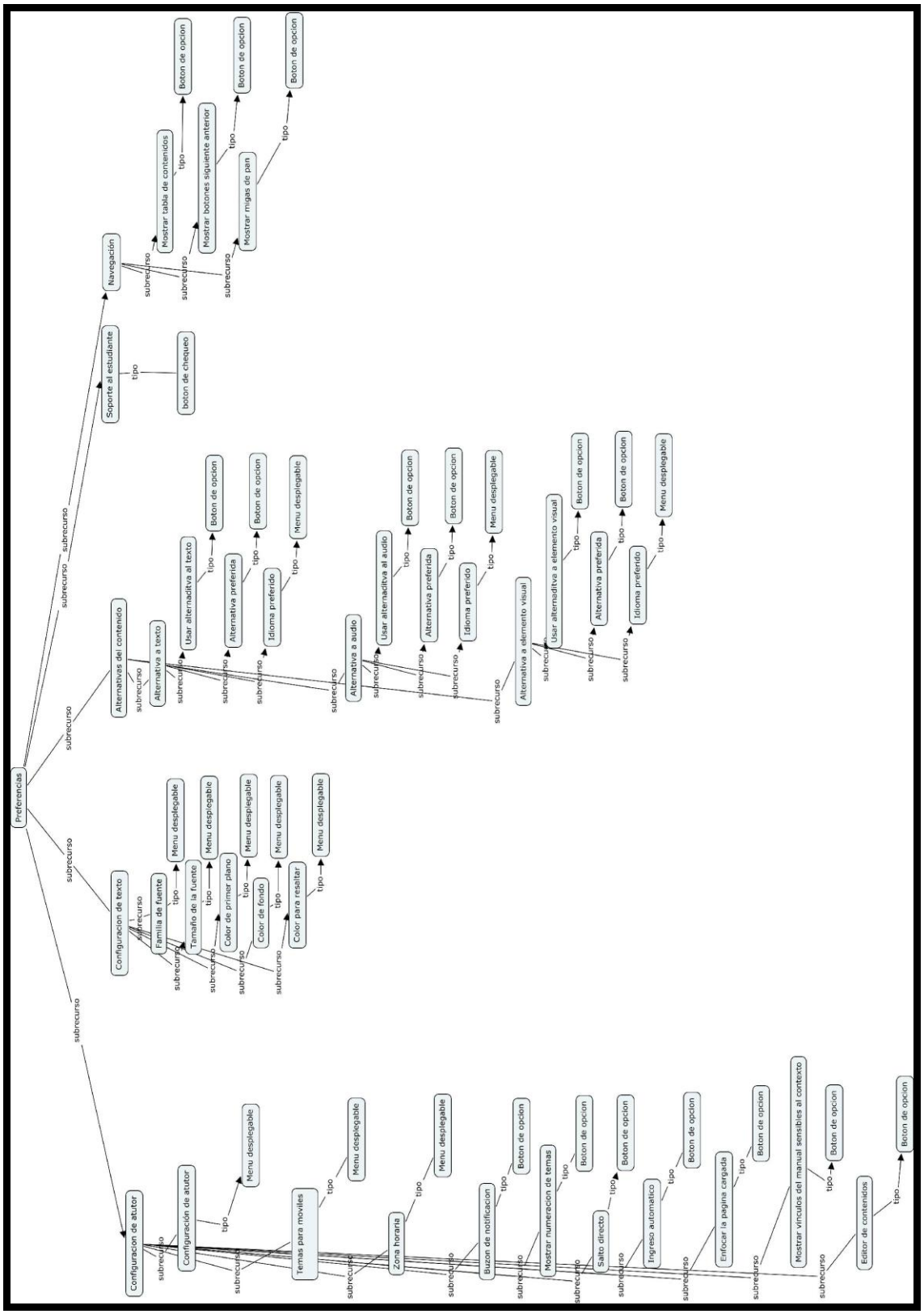


Figura 190 Vista recurso Preferencias con CmapTools de ATutor.

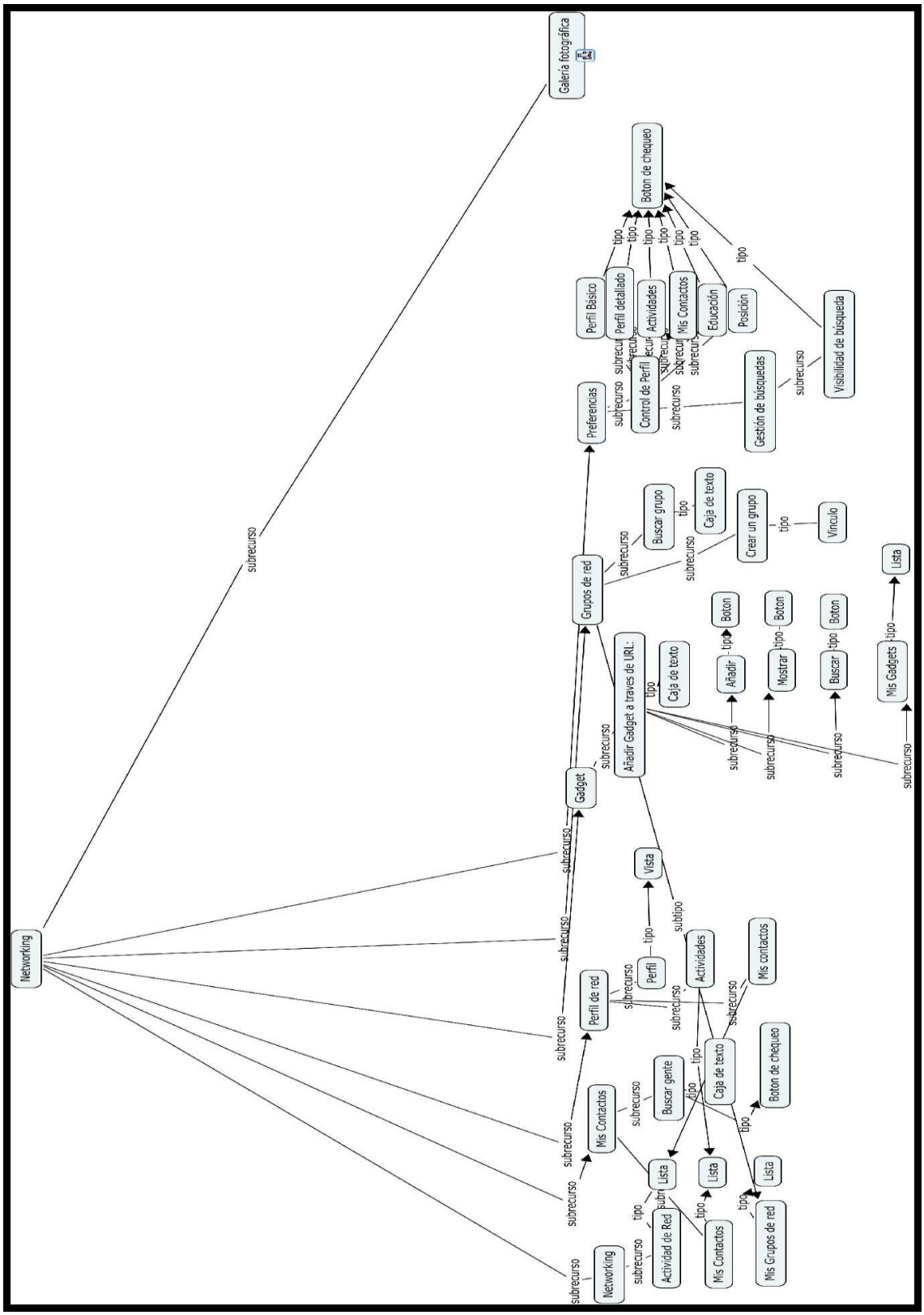


Figura 191 Vista recurso Networking con CmapTools de ATutor.

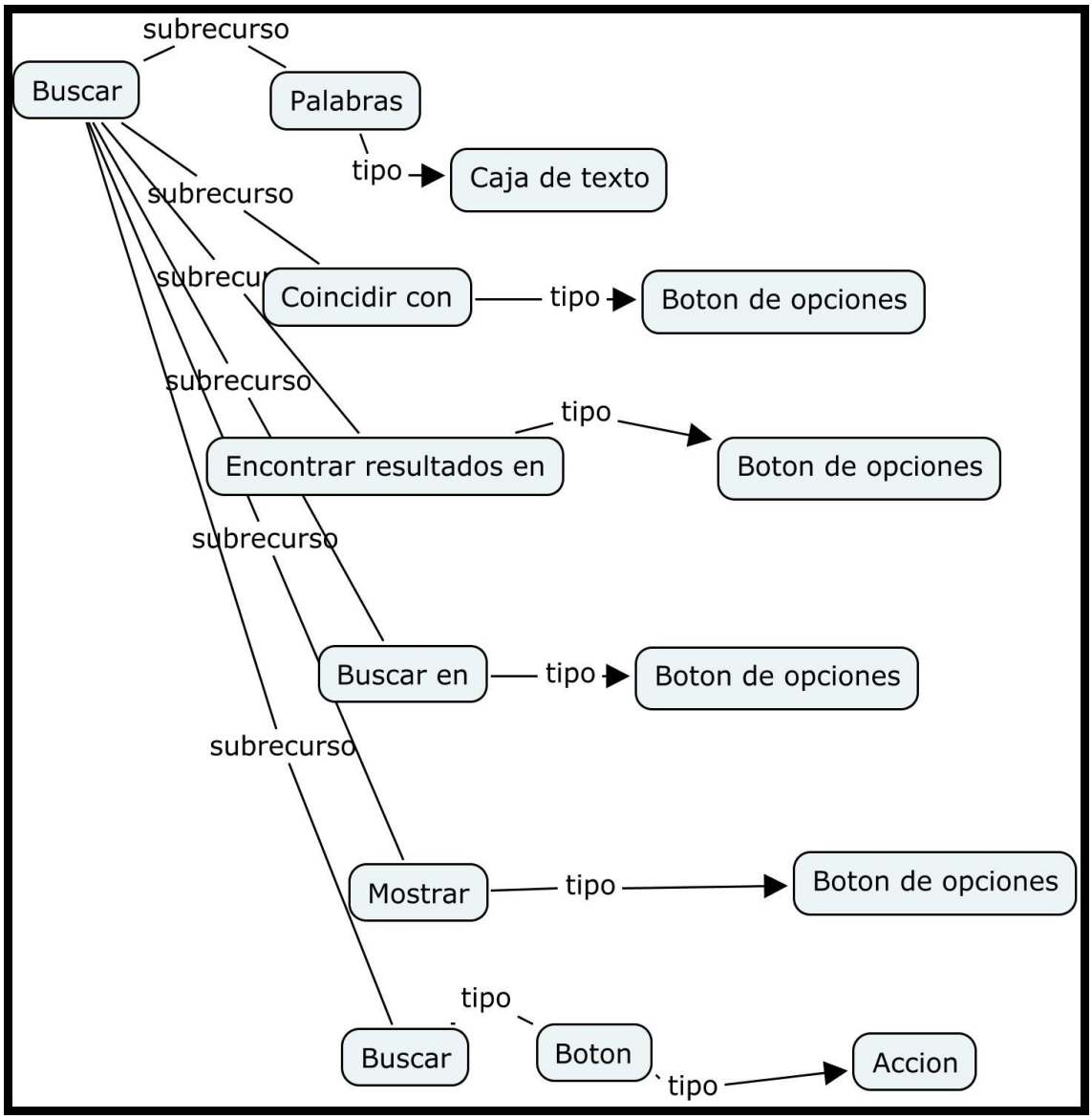


Figura 192 Vista recurso Buscar con CmapTools de ATutor.

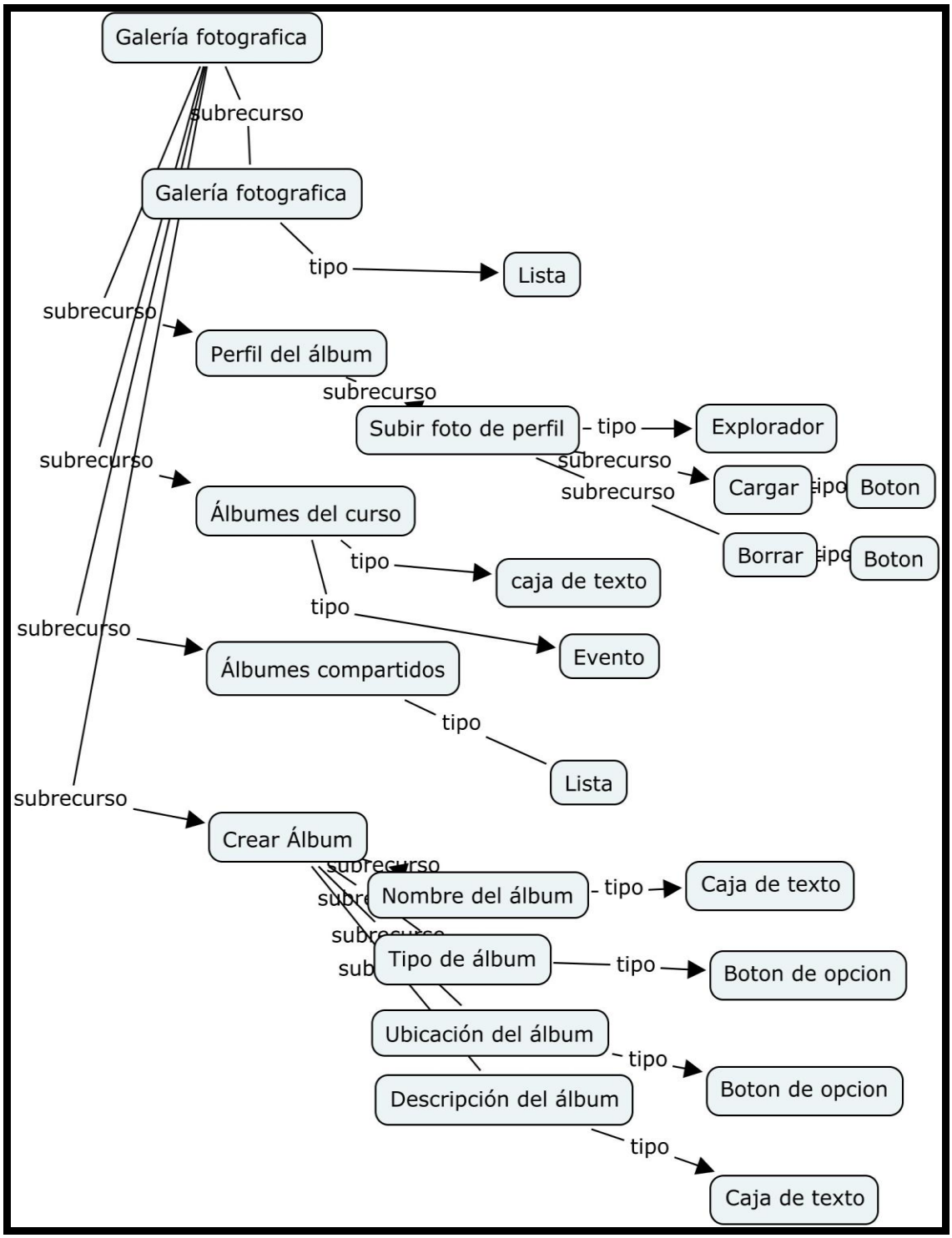


Figura 193 Vista recurso Galería fotográfica con CmapTools de ATutor.

2. Mapa De Conocimiento Para Claroline

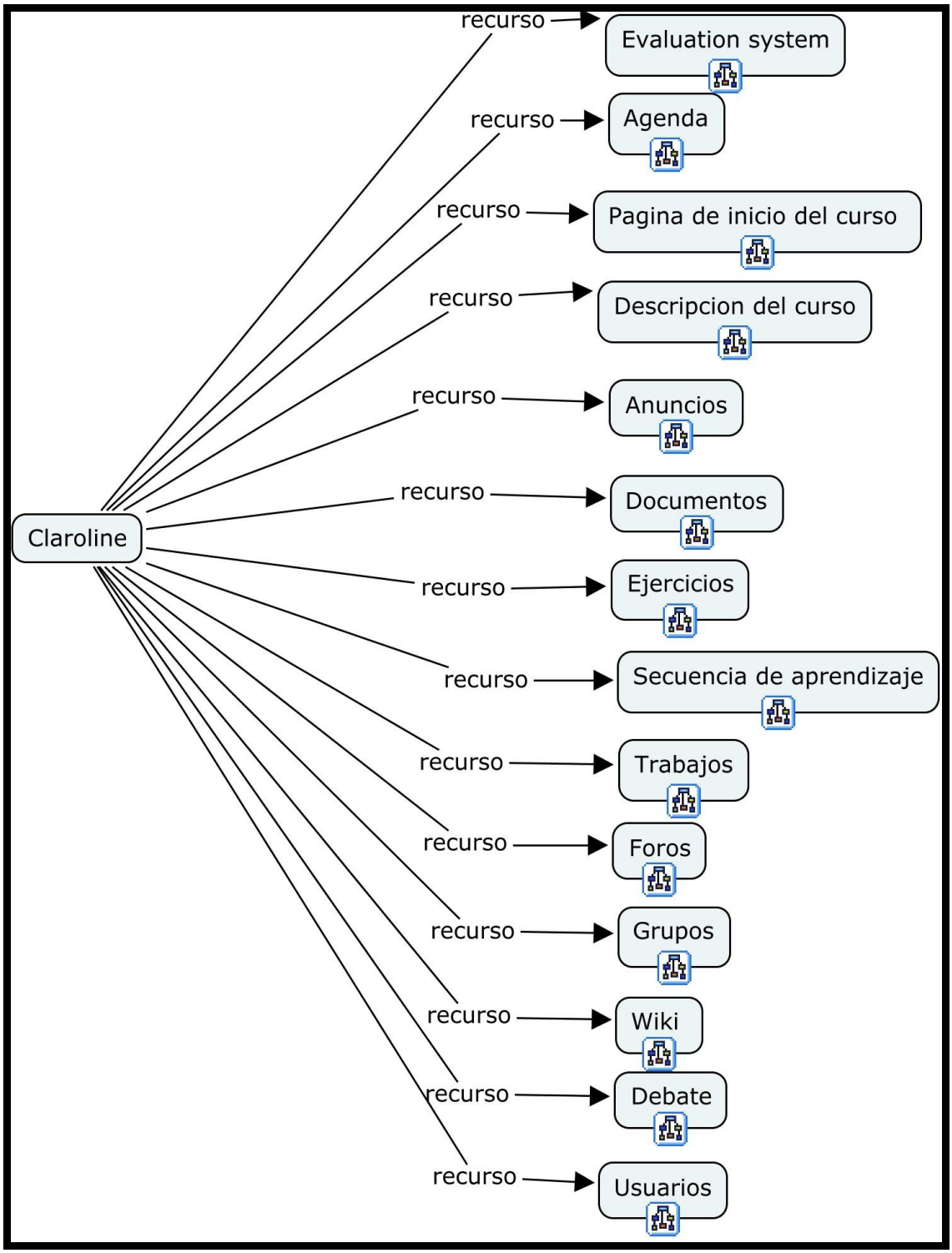


Figura 194 Vista Principal con CmapTools de Claroline.

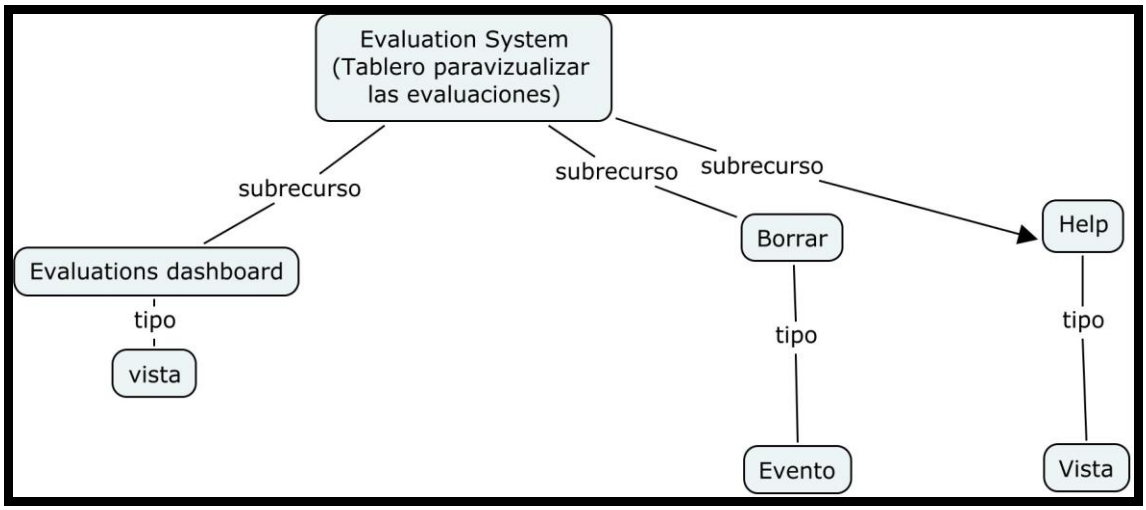


Figura 195 Vista recurso Evaluation System con CmapTools de Claroline.

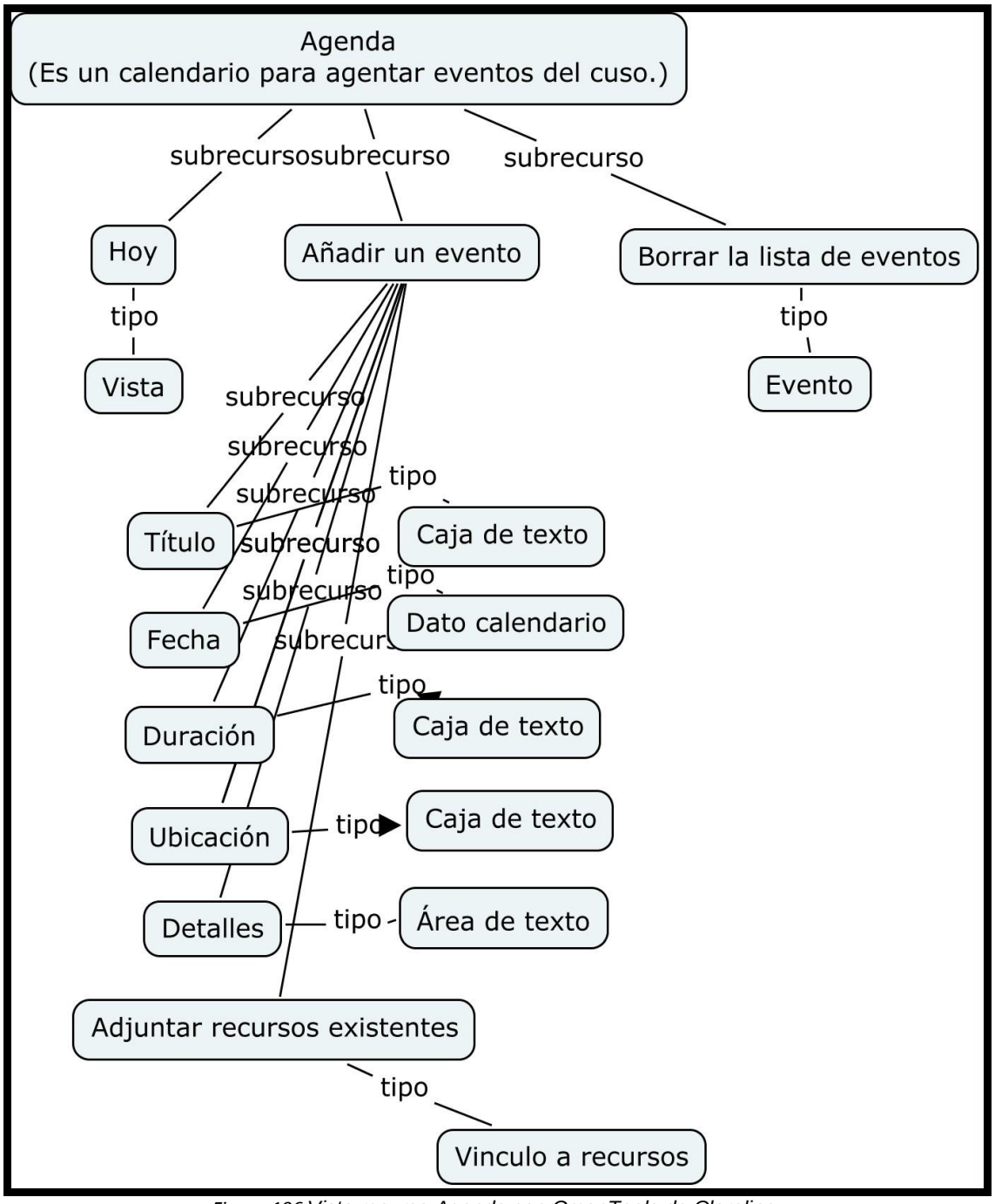


Figura 196 Vista recurso Agenda con CmapTools de Claroline.

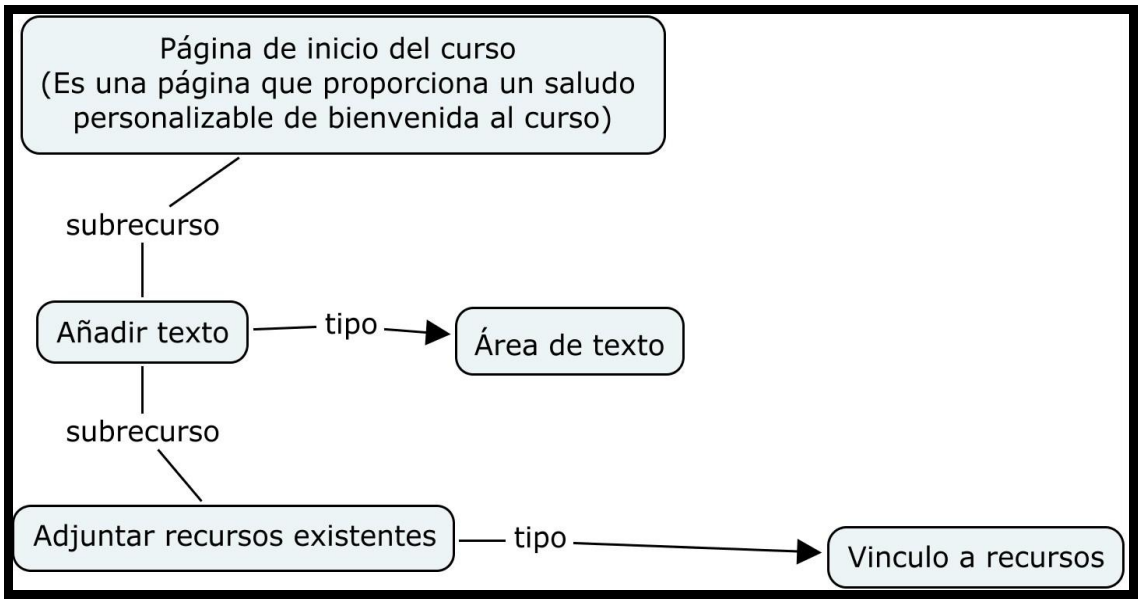


Figura 197 Vista recurso Página de inicio del curso con CmapTools de Claroline.

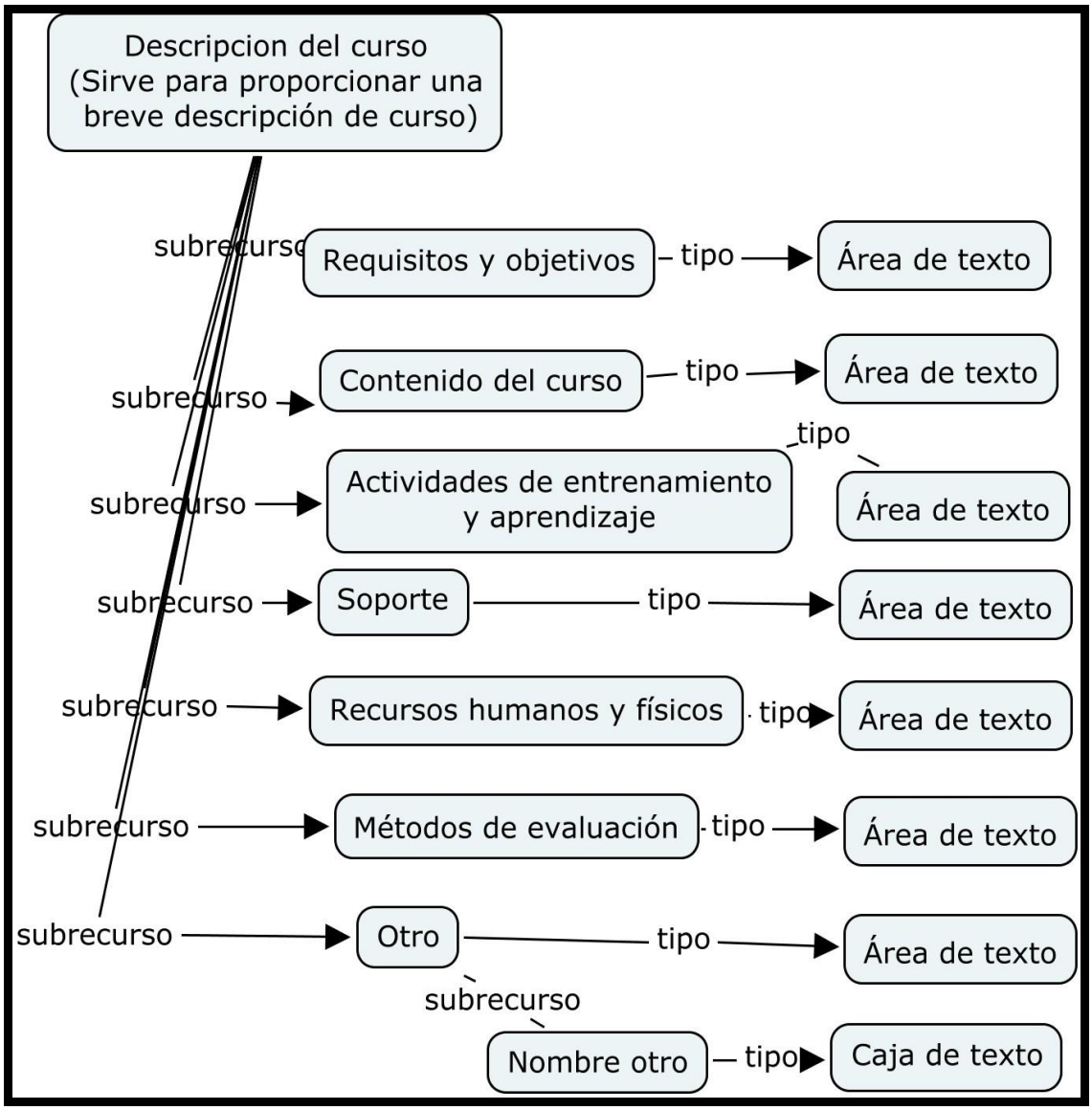


Figura 198 Vista recurso Descripción del curso con CmapTools de Claroline.

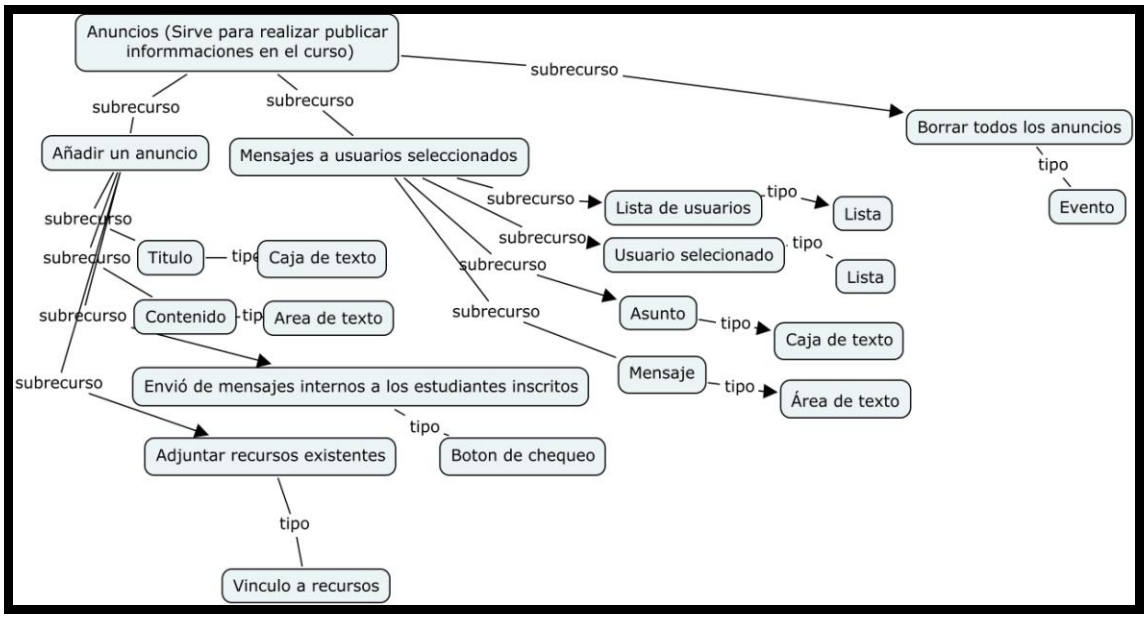


Figura 199 Vista recurso Anuncios con CmapTools de Claroline.

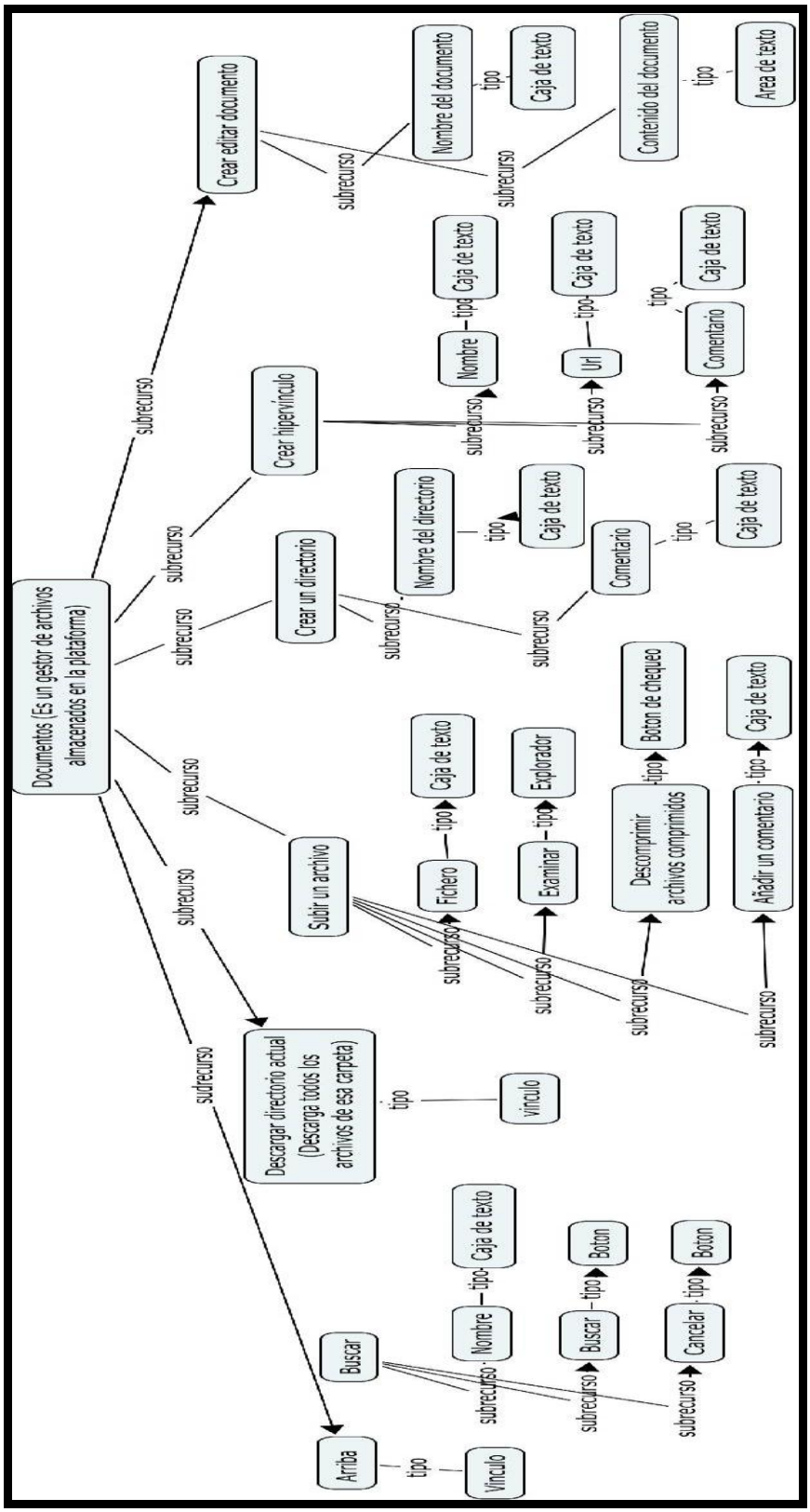


Figura 200 Vista recurso Documentos con CmapTools de Claroline.

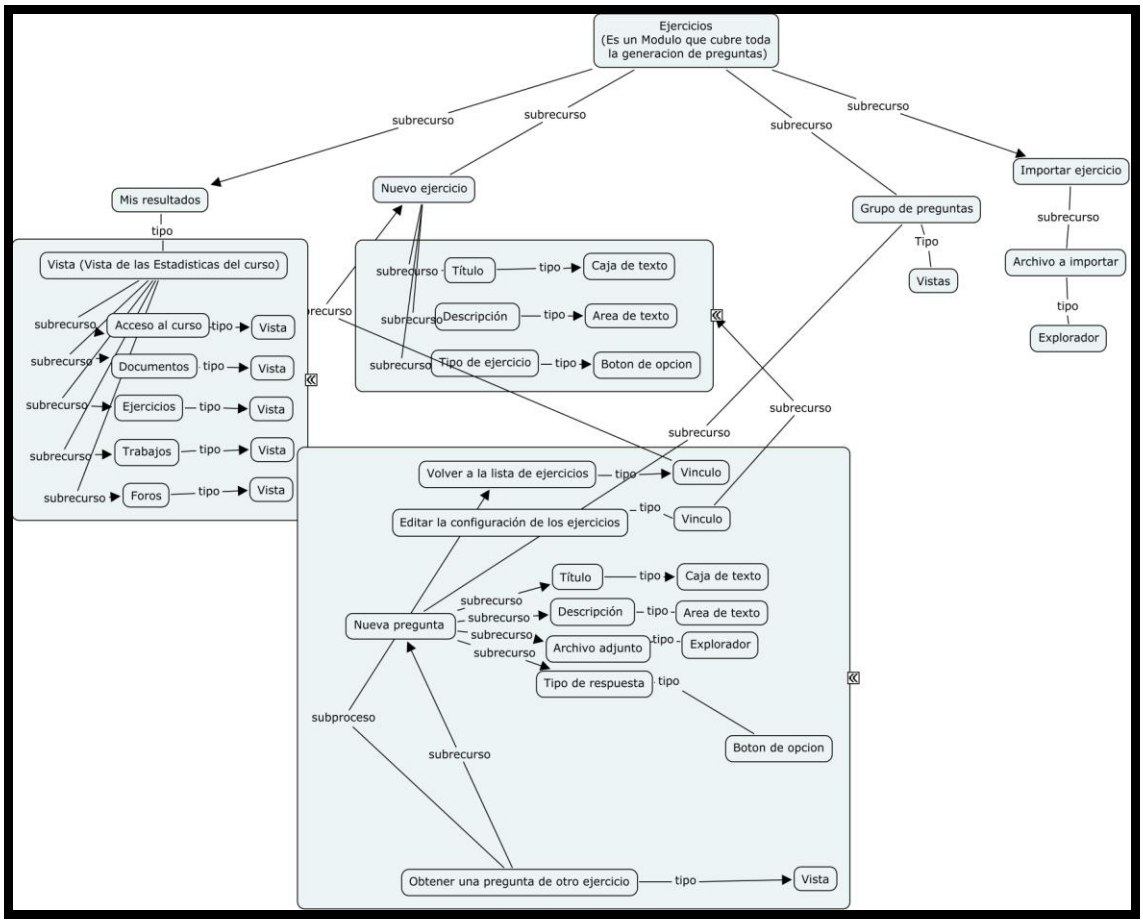


Figura 201 Vista recurso Ejercicios con CmapTools de Claroline.

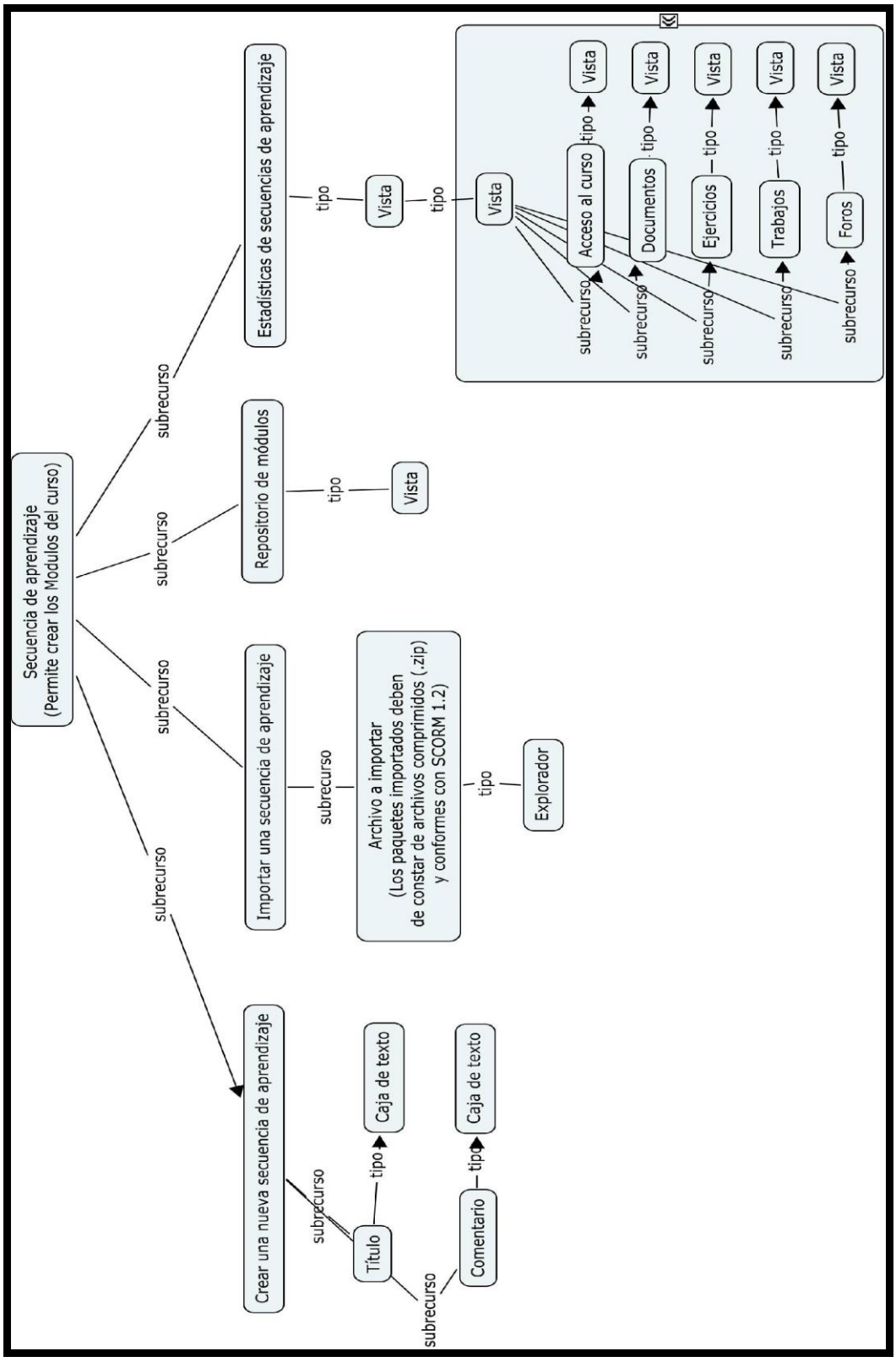


Figura 202 Vista recurso Secuencia de aprendizaje con CmapTools de Claroline.

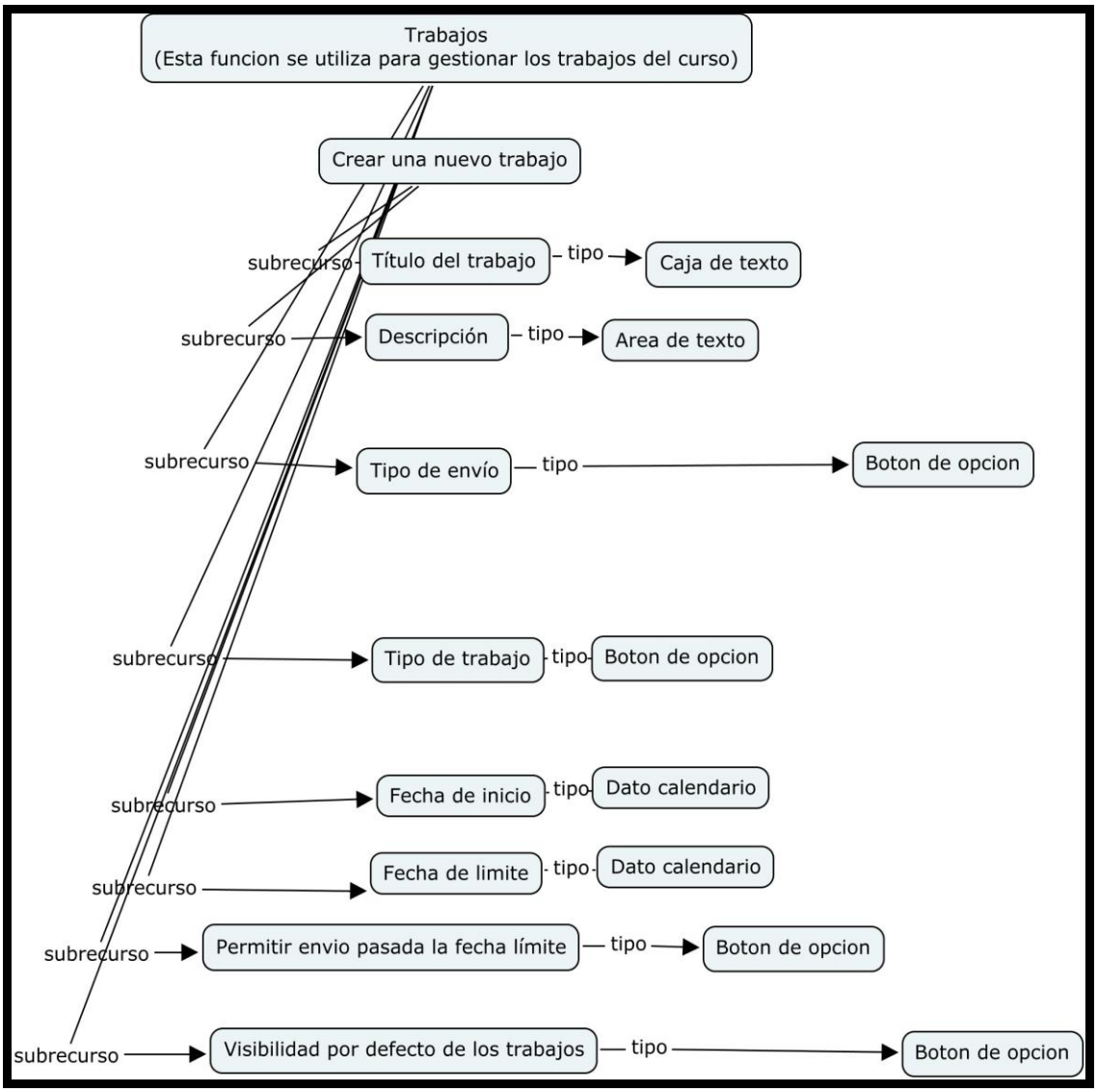


Figura 203 Vista recurso Trabajos con CmapTools de Claroline.

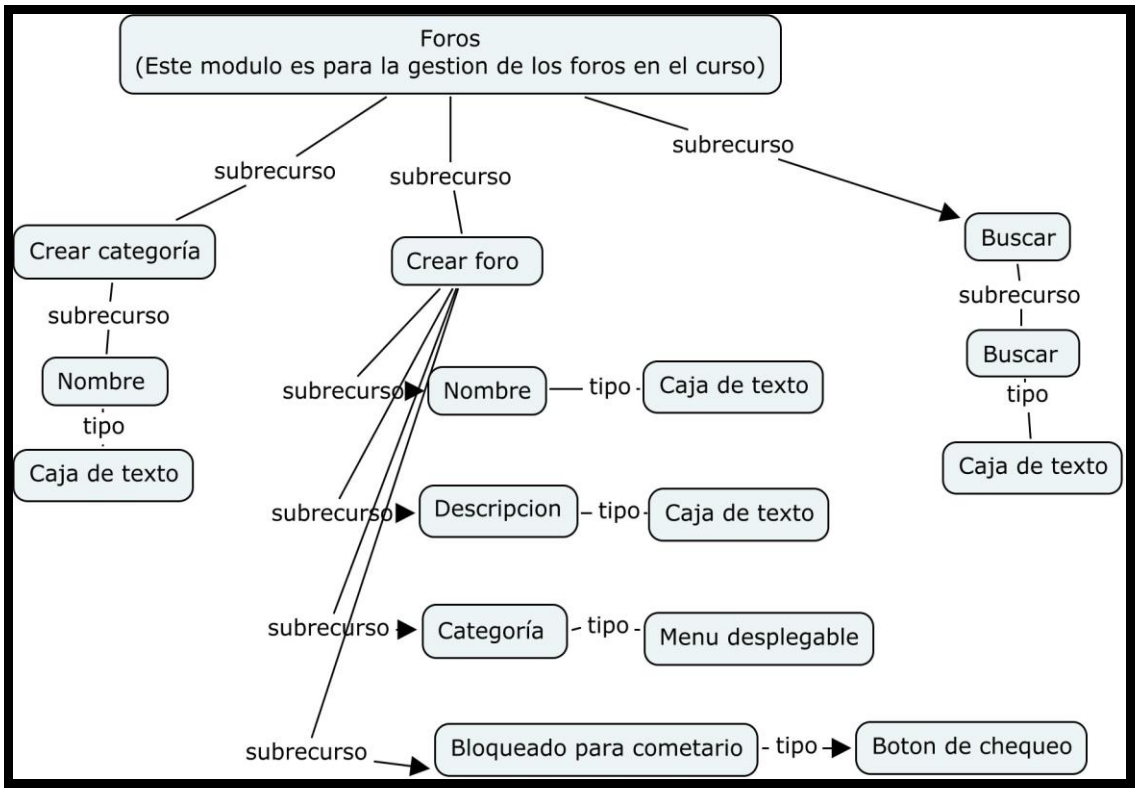


Figura 204 Vista recurso Foros con CmapTools de Claroline.

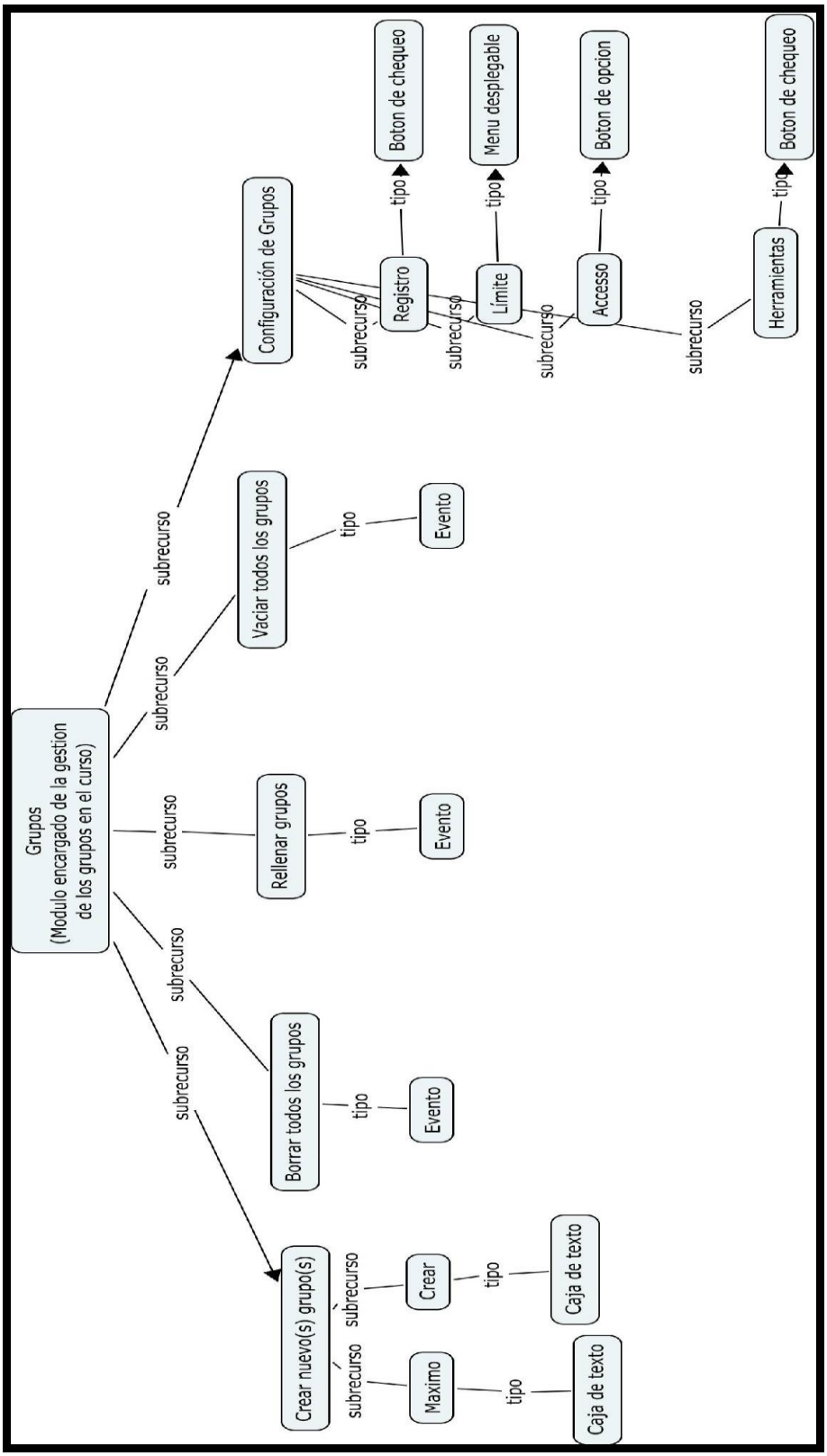


Figura 205 Vista recurso Grupos con CmapTools de Claroline.

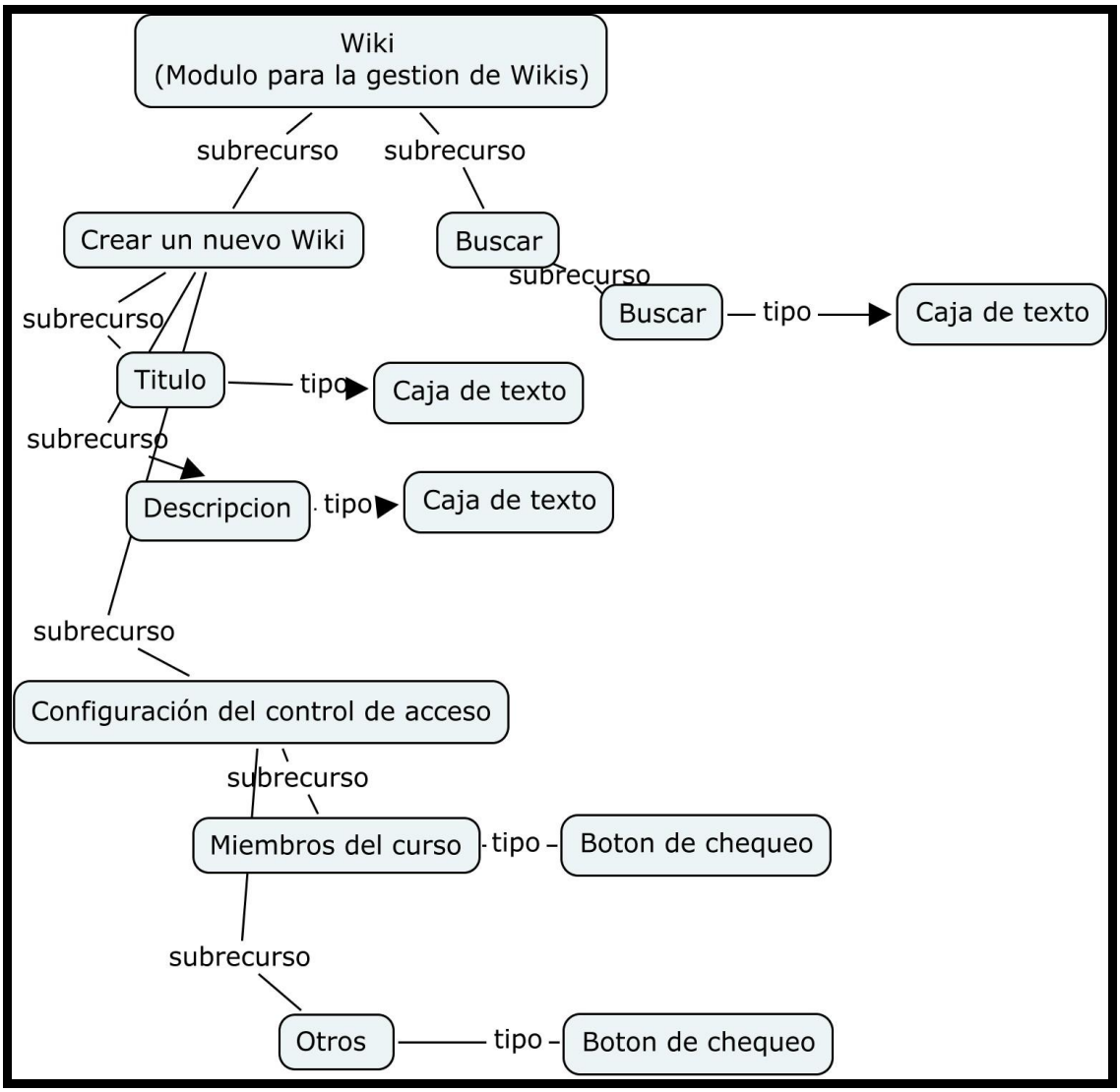


Figura 206 Vista recurso Wiki con CmapTools de Claroline.

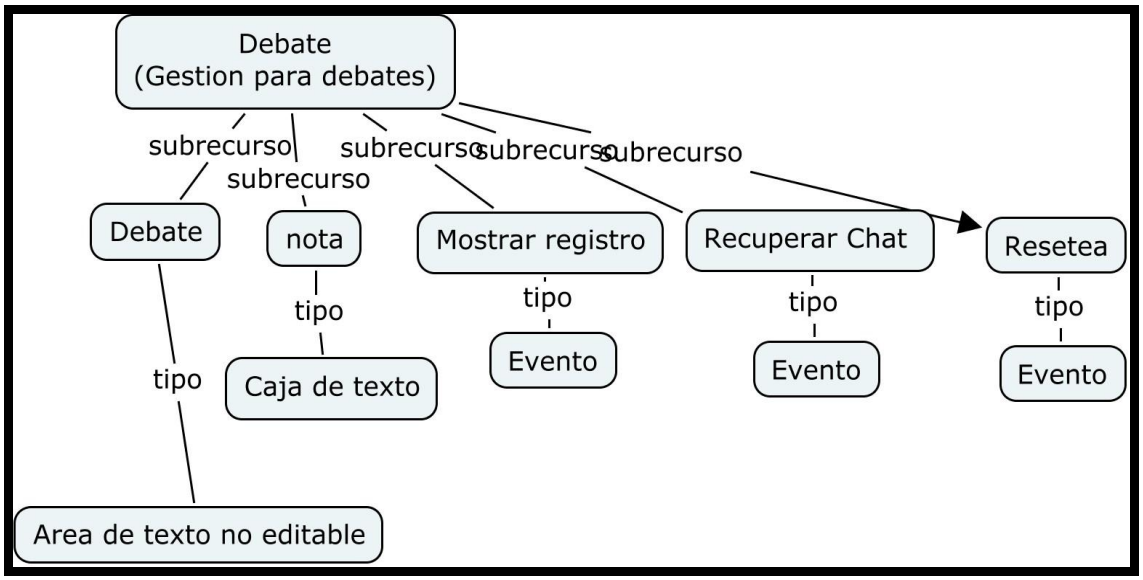


Figura 207 Vista recurso Debate con CmapTools de Claroline.

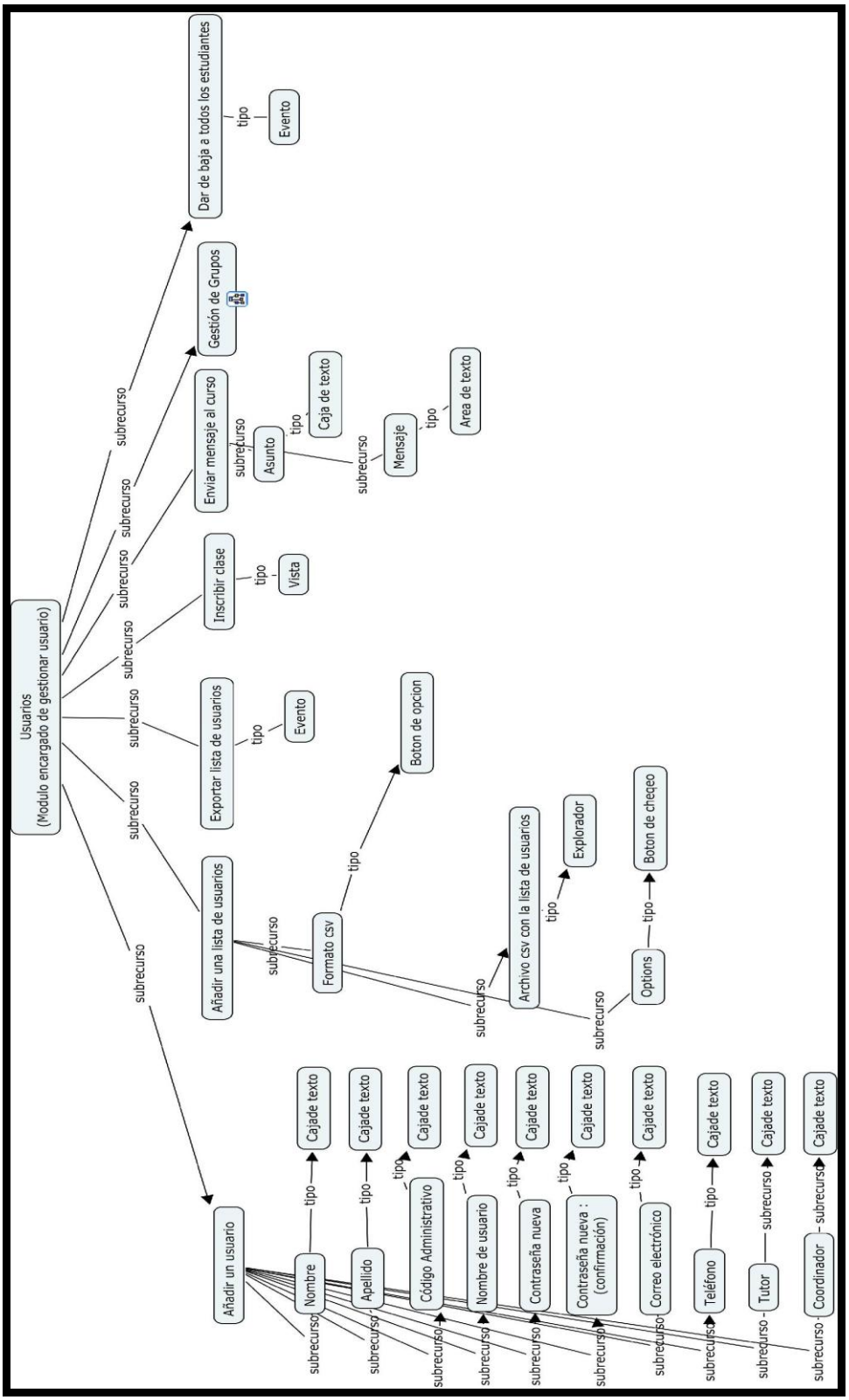


Figura 208 Vista recurso Usuarios con CmapTools de Claroline.

3. Mapa De Conocimiento Para Moodle

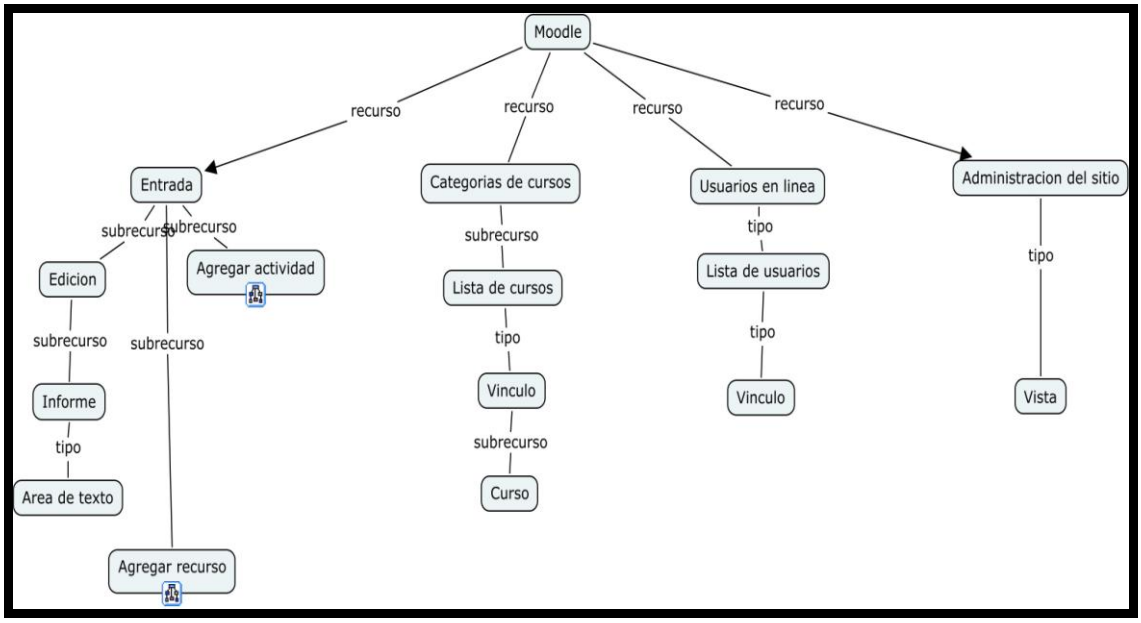


Figura 209 Vista principal con CmapTools de Moodle.

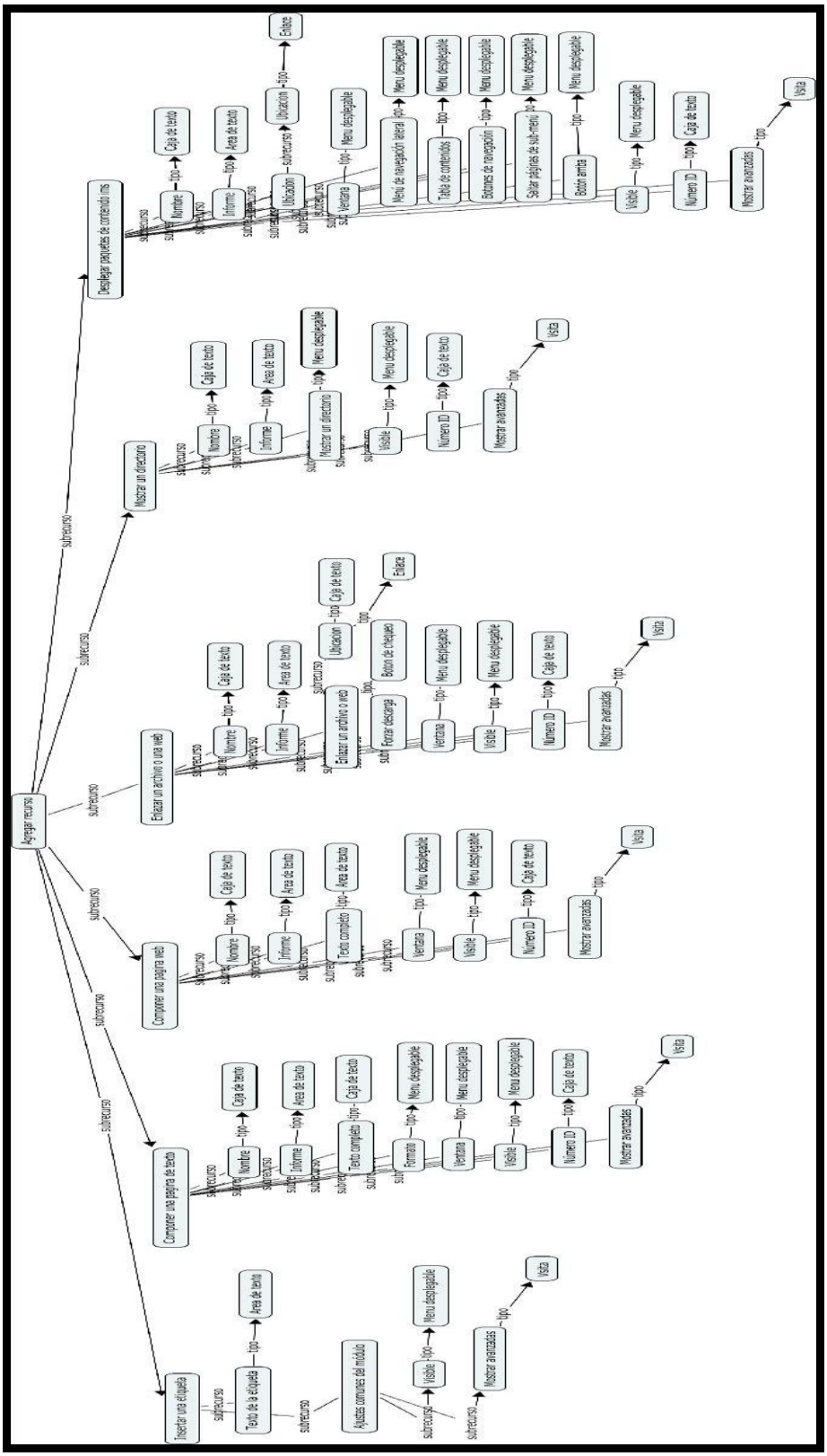


Figura 210 Vista recurso Agregar recurso con CmapTools de Moodle.

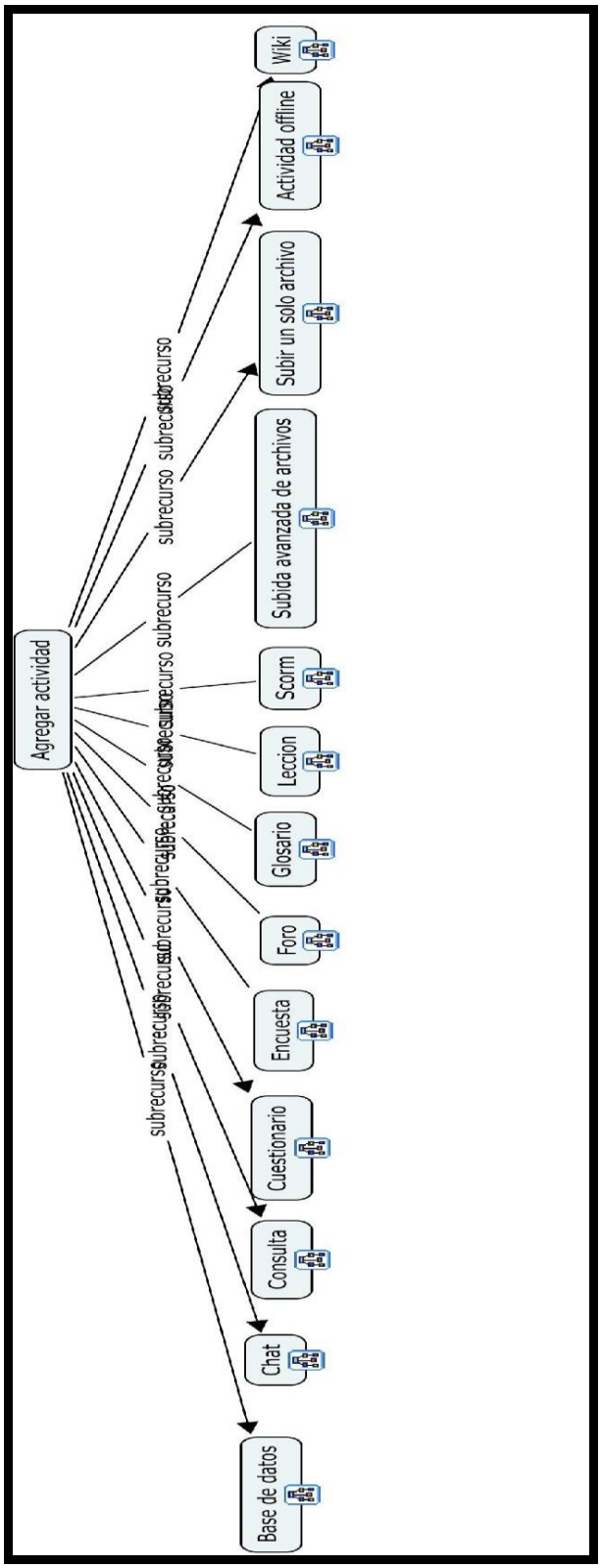


Figura 211 Vista recurso Agregar actividad con CmapTools de Moodle.

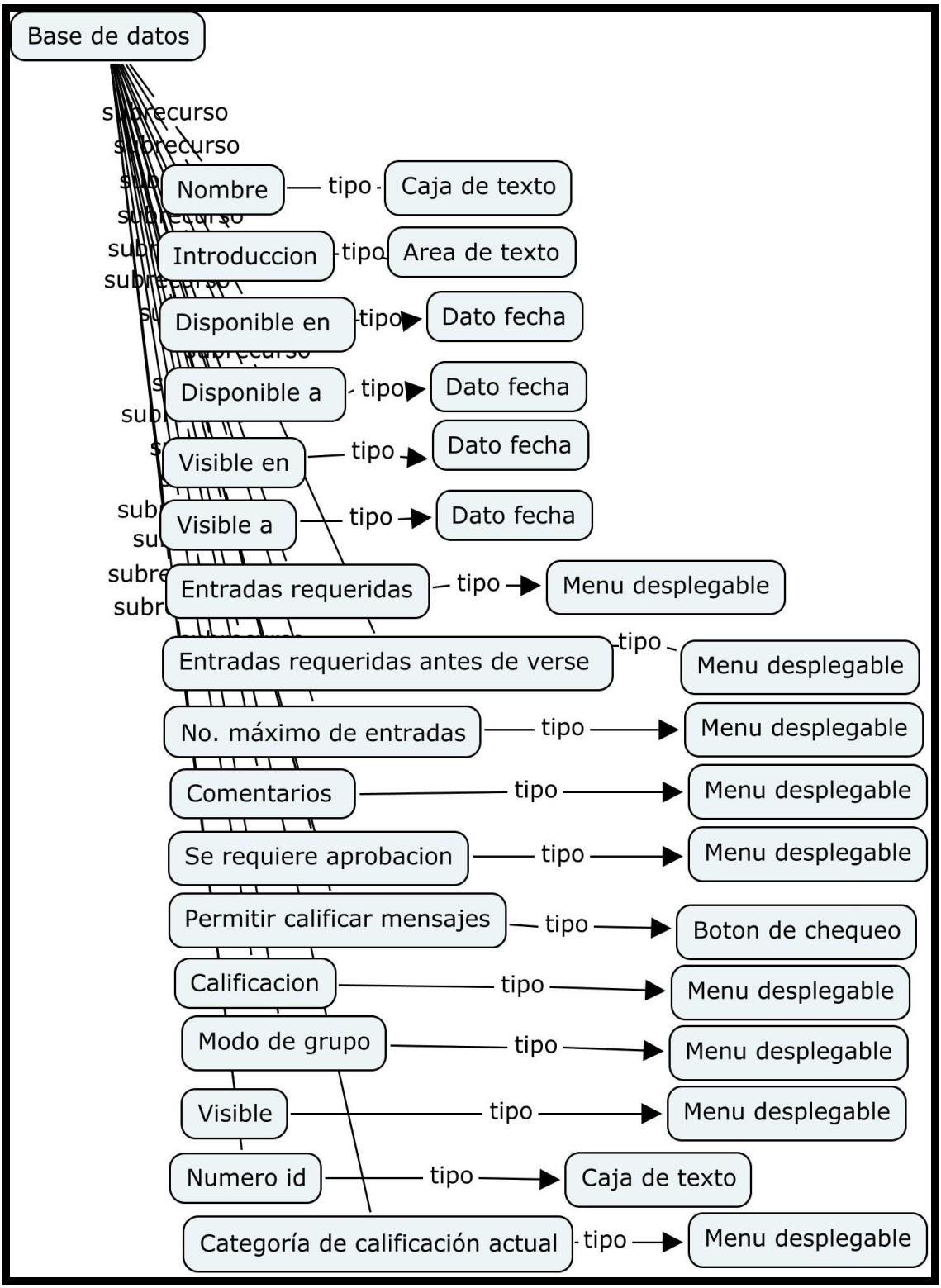


Figura 212 Vista recurso Base de datos con CmapTools de Moodle.

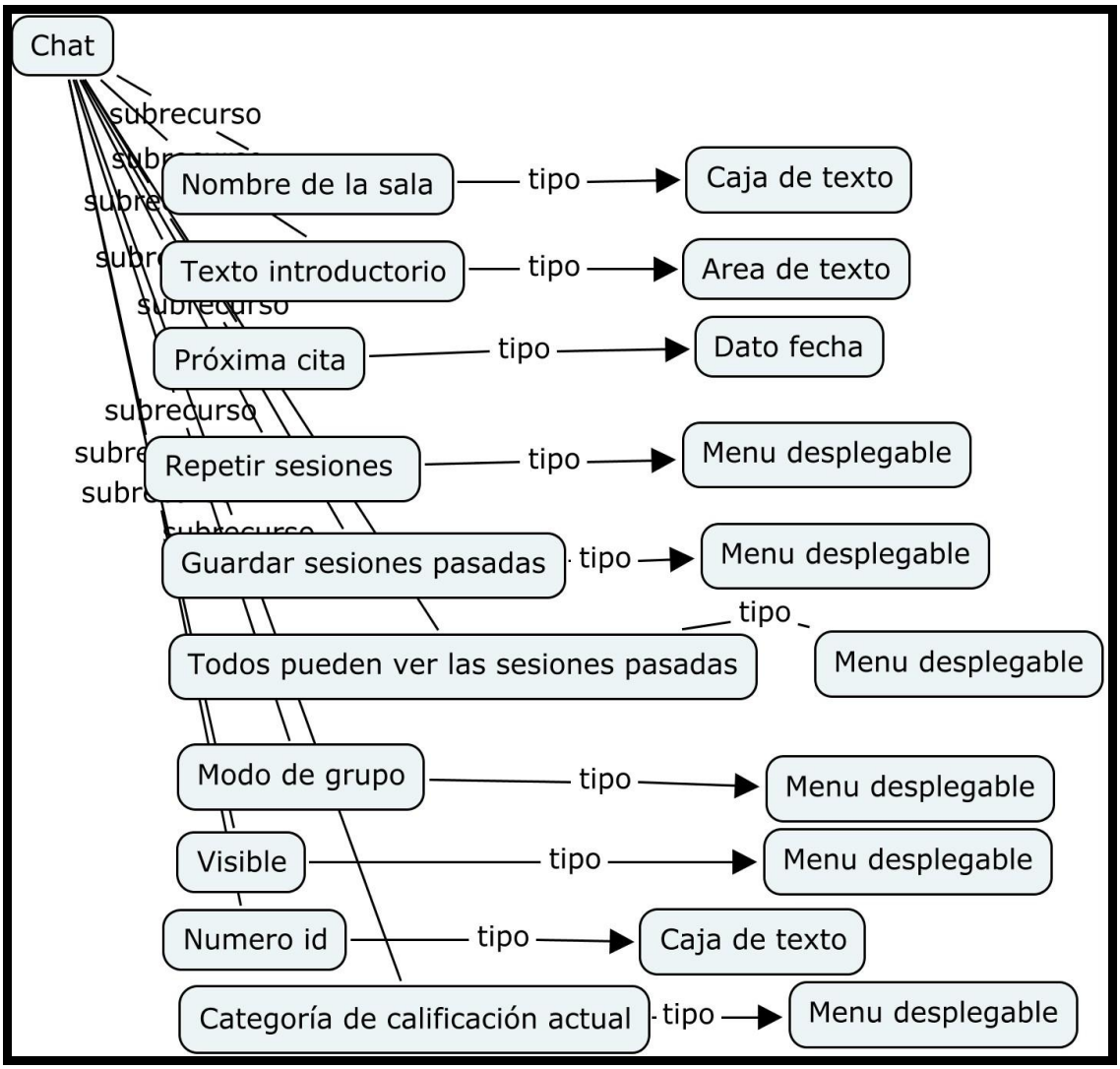


Figura 213 Vista recurso Chat con CmapTools de Moodle.

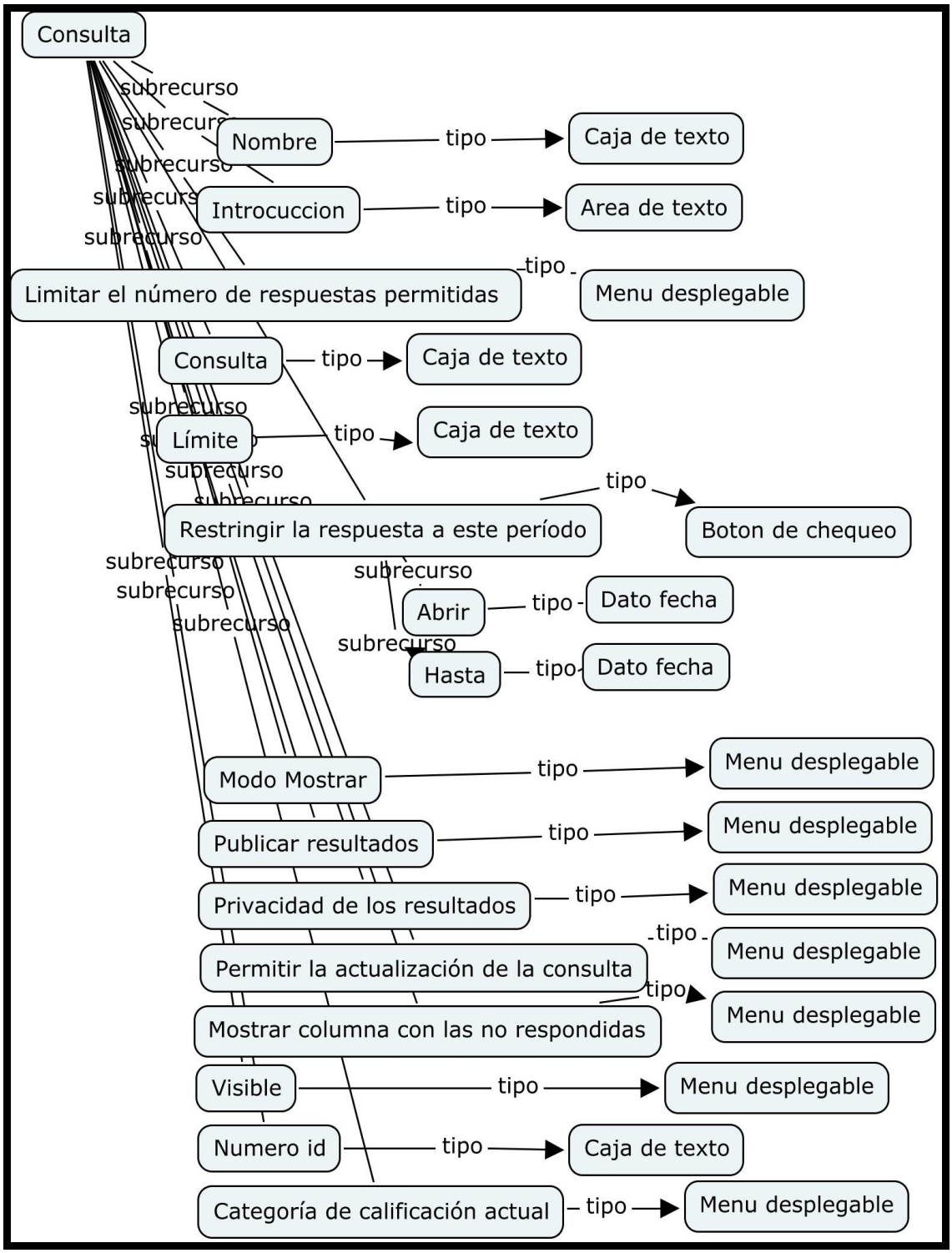


Figura 214 Vista recurso Consulta con CmapTools de Moodle.

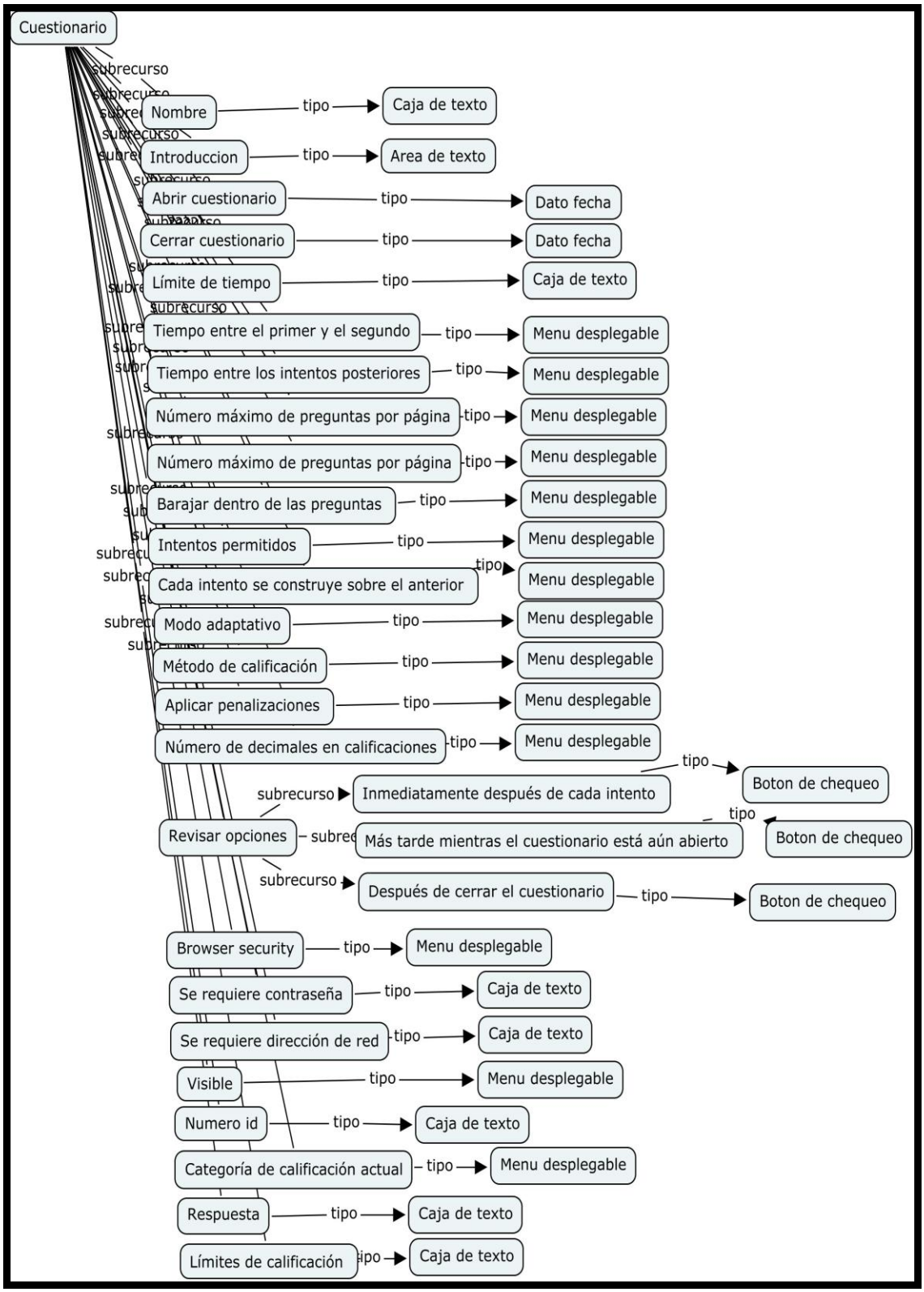


Figura 215 Vista recurso cuestionario con CmapTools de Moodle.

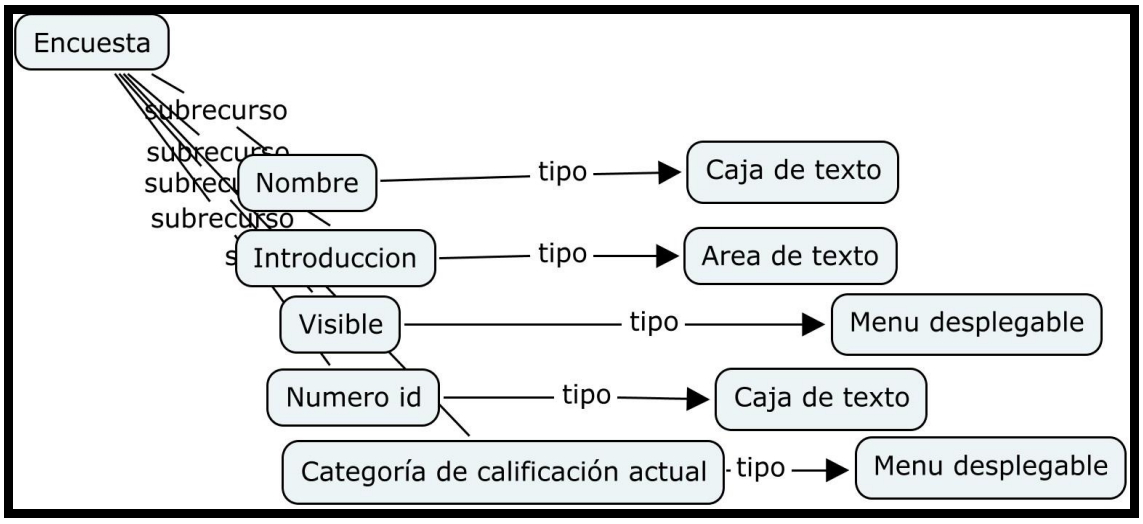


Figura 216 Vista recurso Encuesta con CmapTools de Moodle.

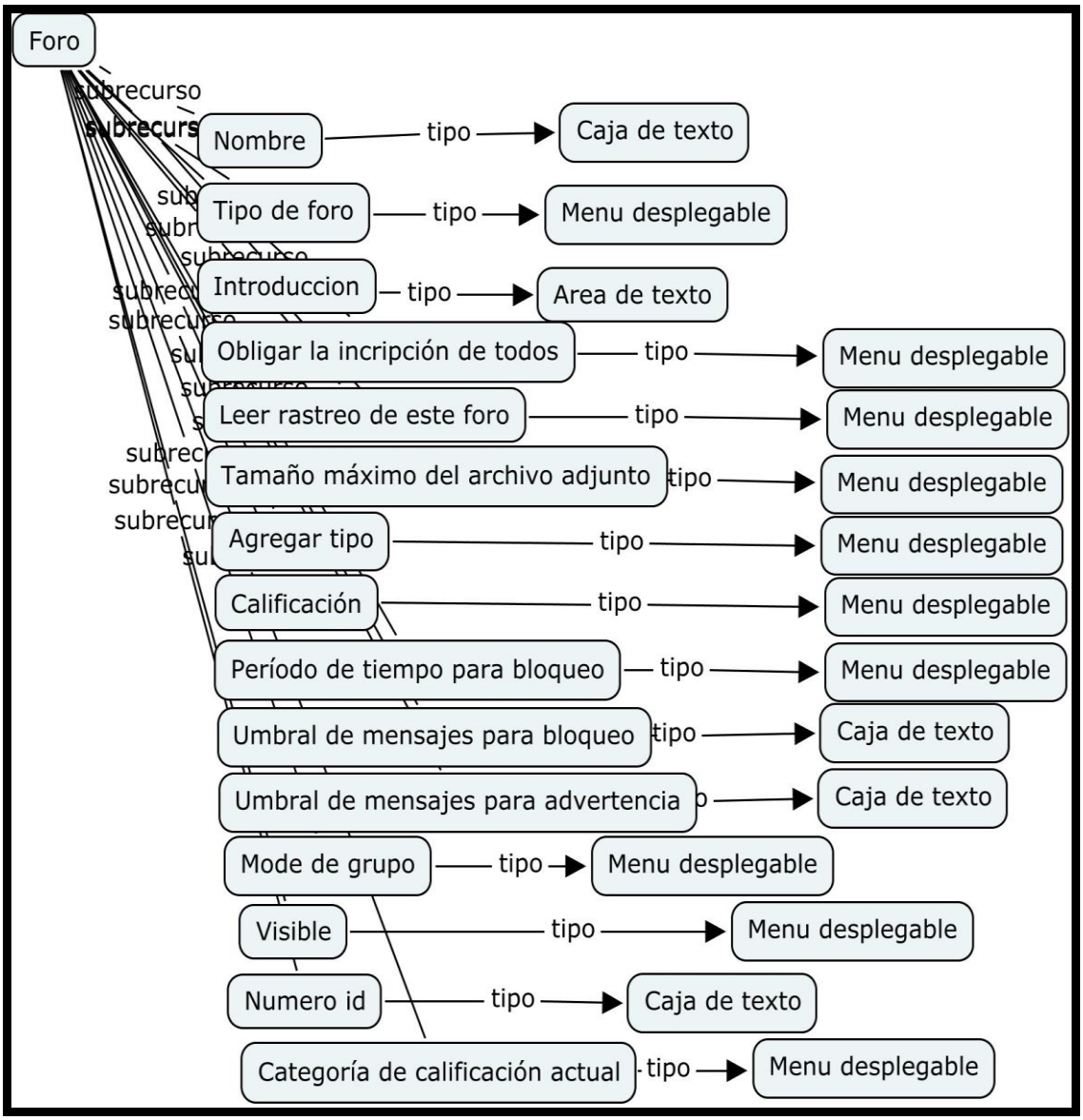


Figura 217 Vista recurso Foro con CmapTools de Moodle.

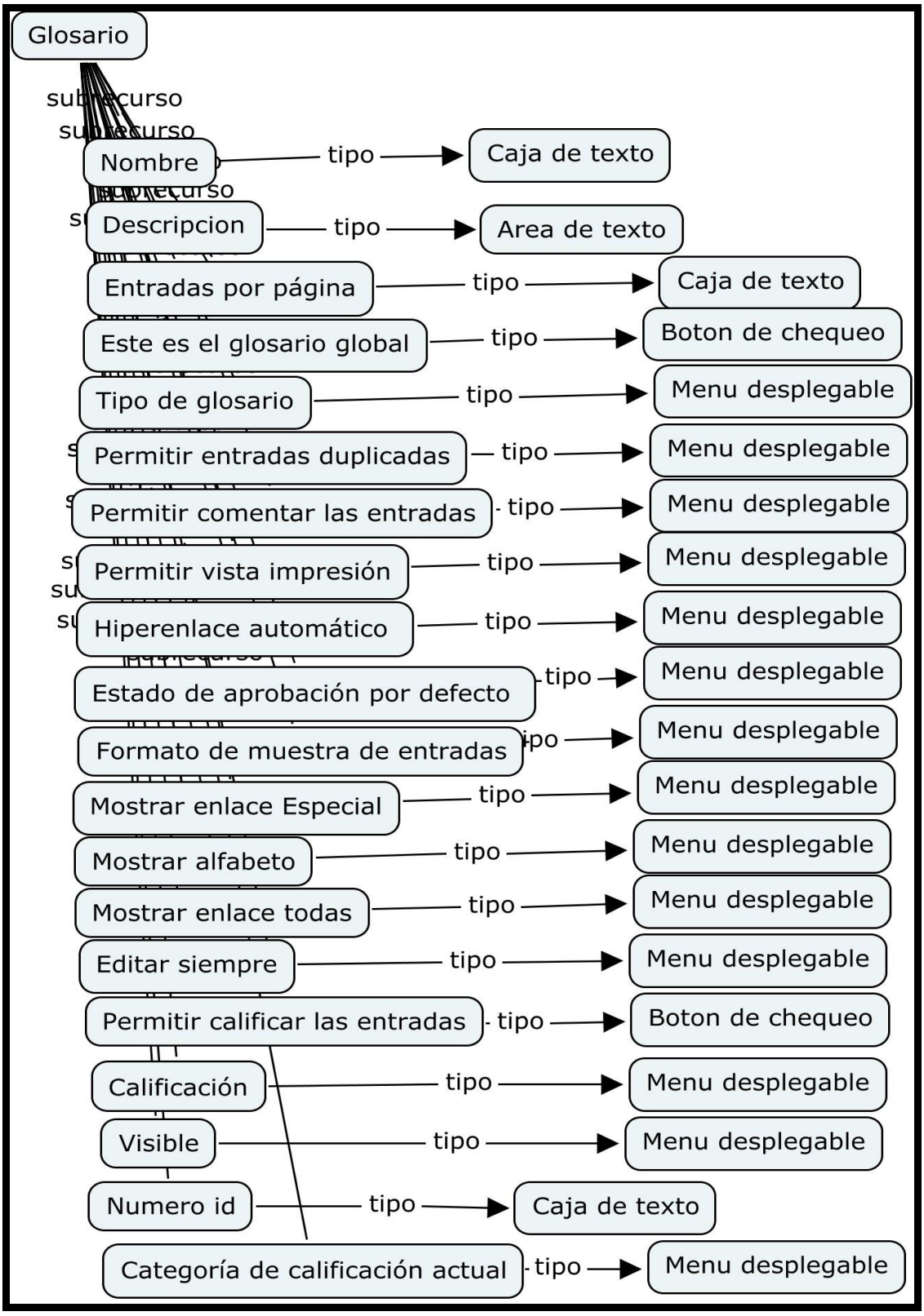


Figura 218 Vista recurso Glosario con CmapTools de Moodle.

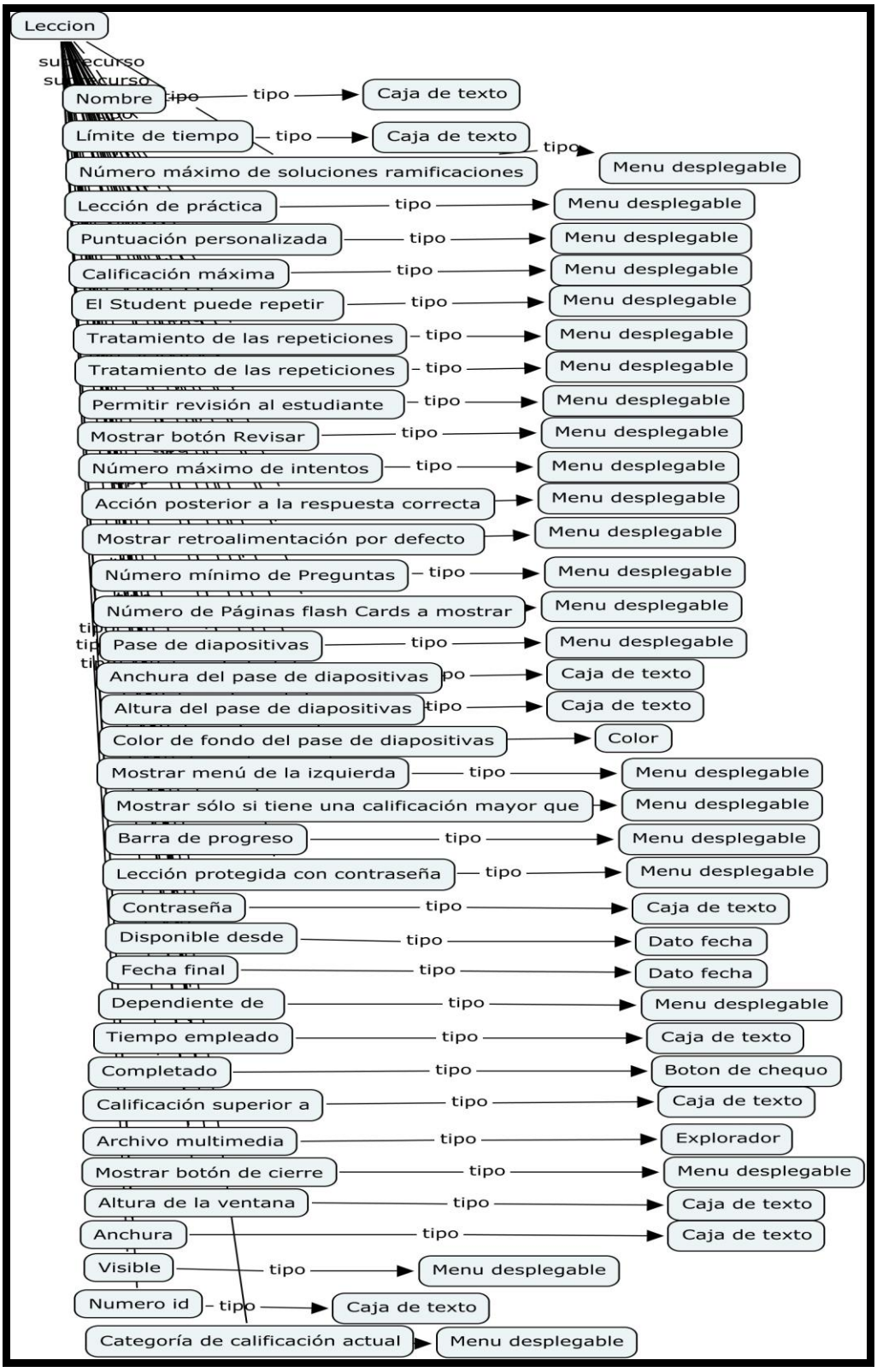


Figura 219 Vista recurso Lección con CmapTools de Moodle.

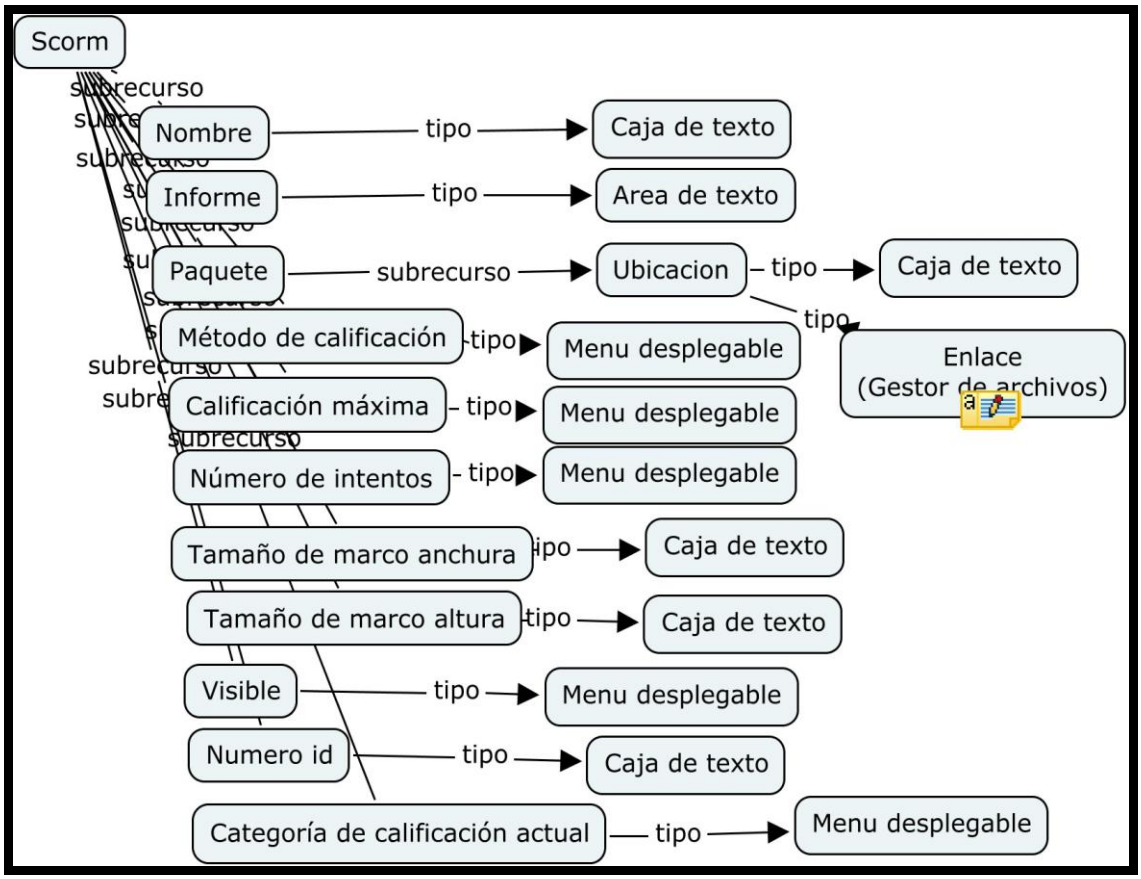


Figura 220 Vista recurso Scorm con CmapTools de Moodle.

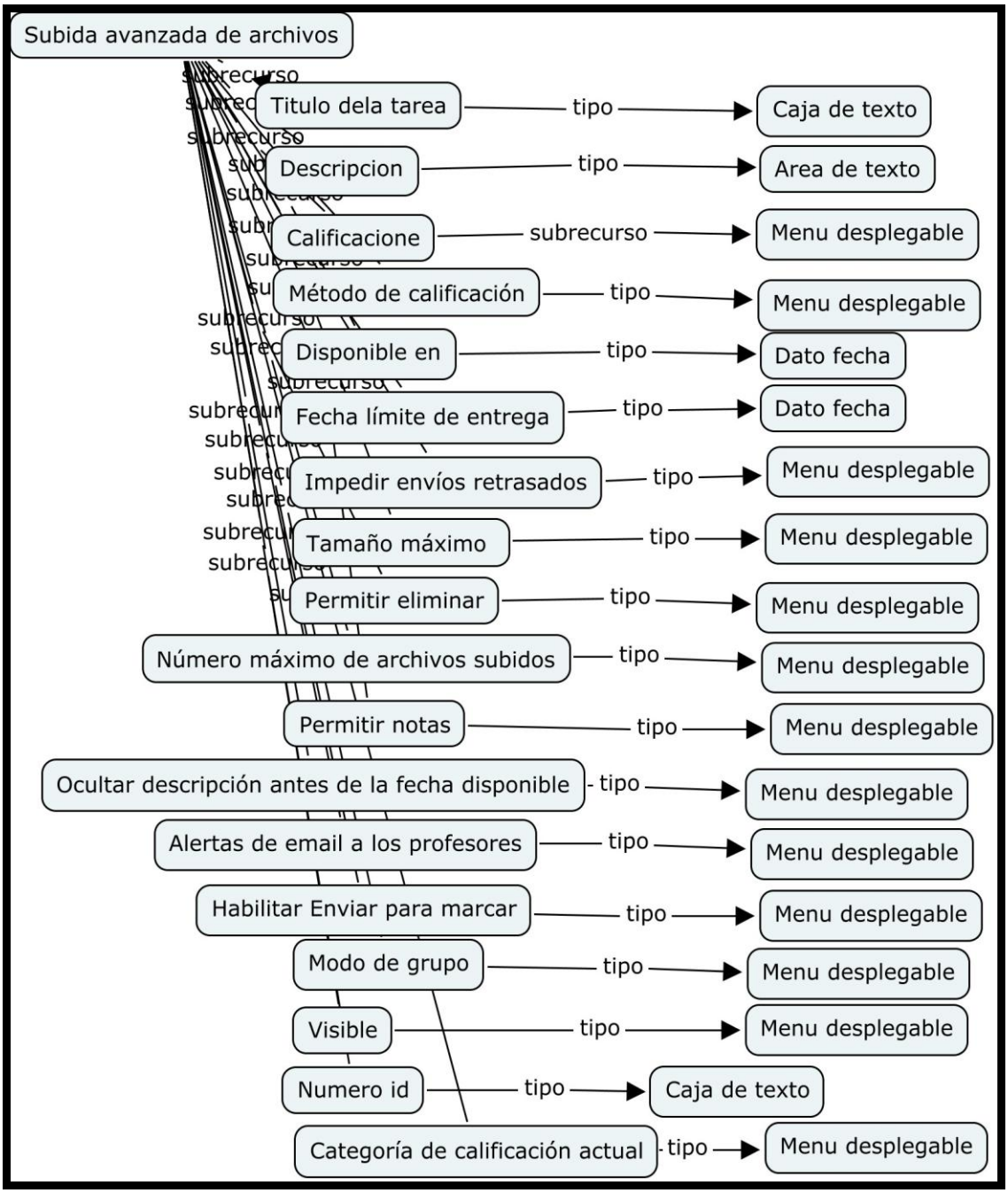


Figura 221 Vista recurso Subida avanzada de archivos con CmapTools de Moodle.

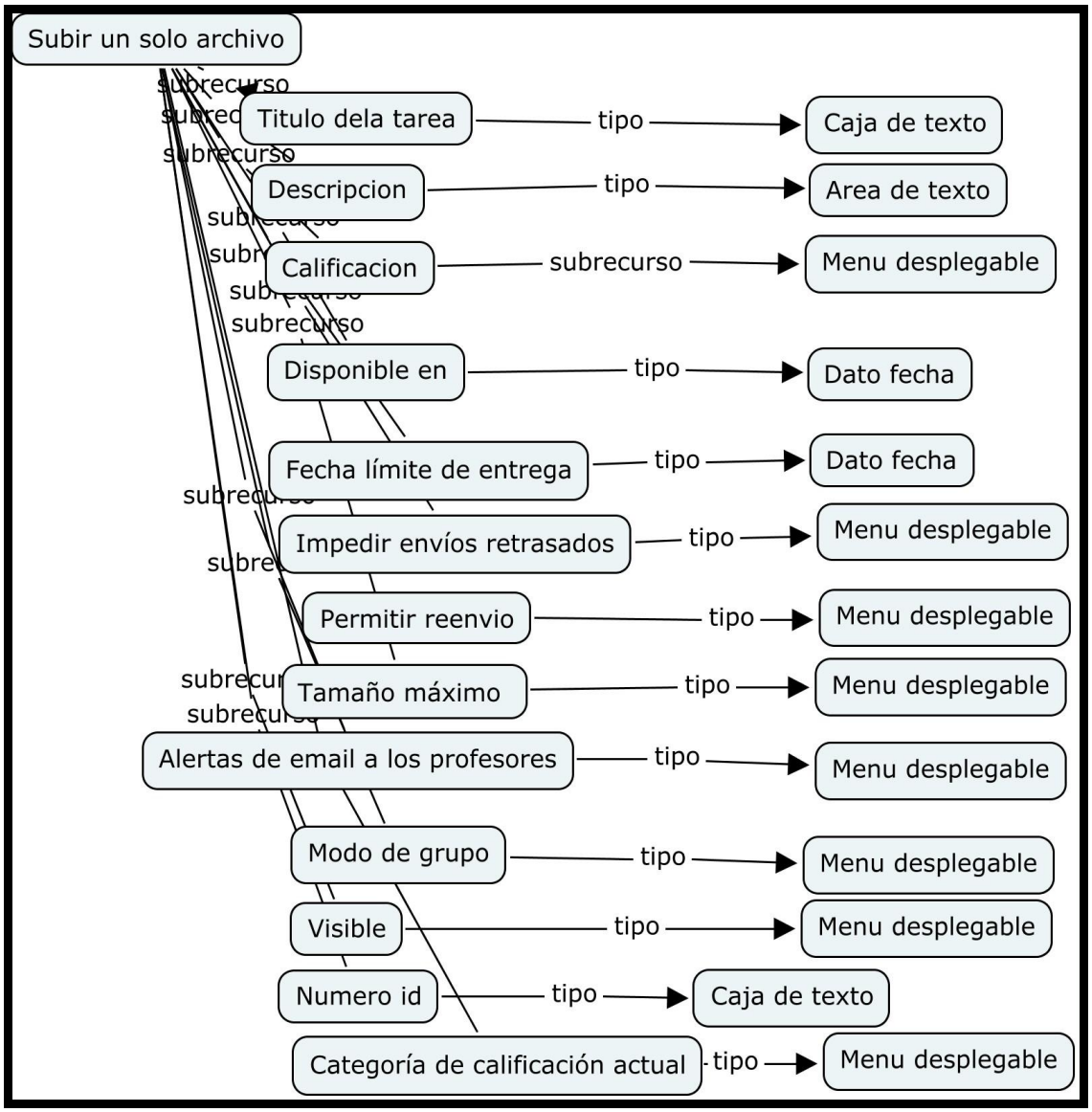


Figura 222 Vista recurso Subir un solo archivo con CmapTools de Moodle.

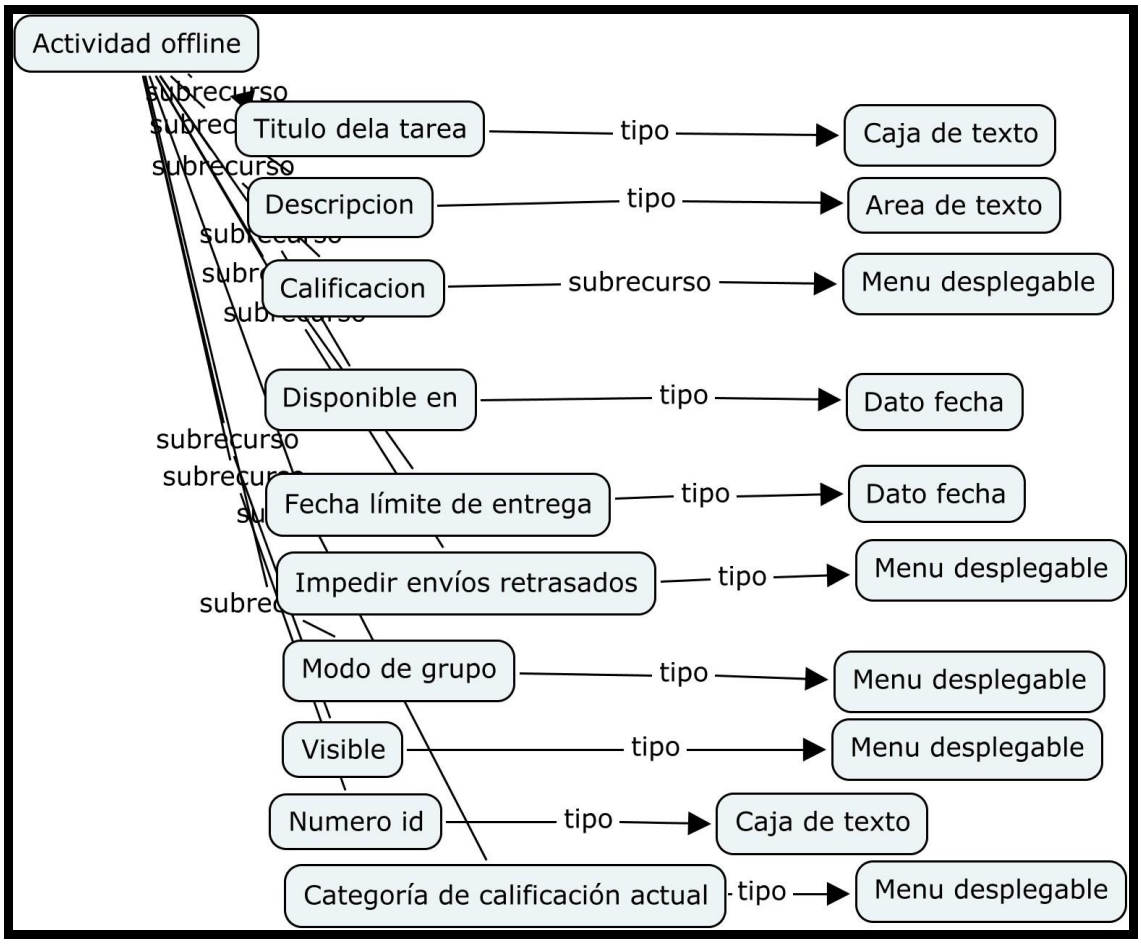


Figura 223 Vista recurso Actividad offline con CmapTools de Moodle.

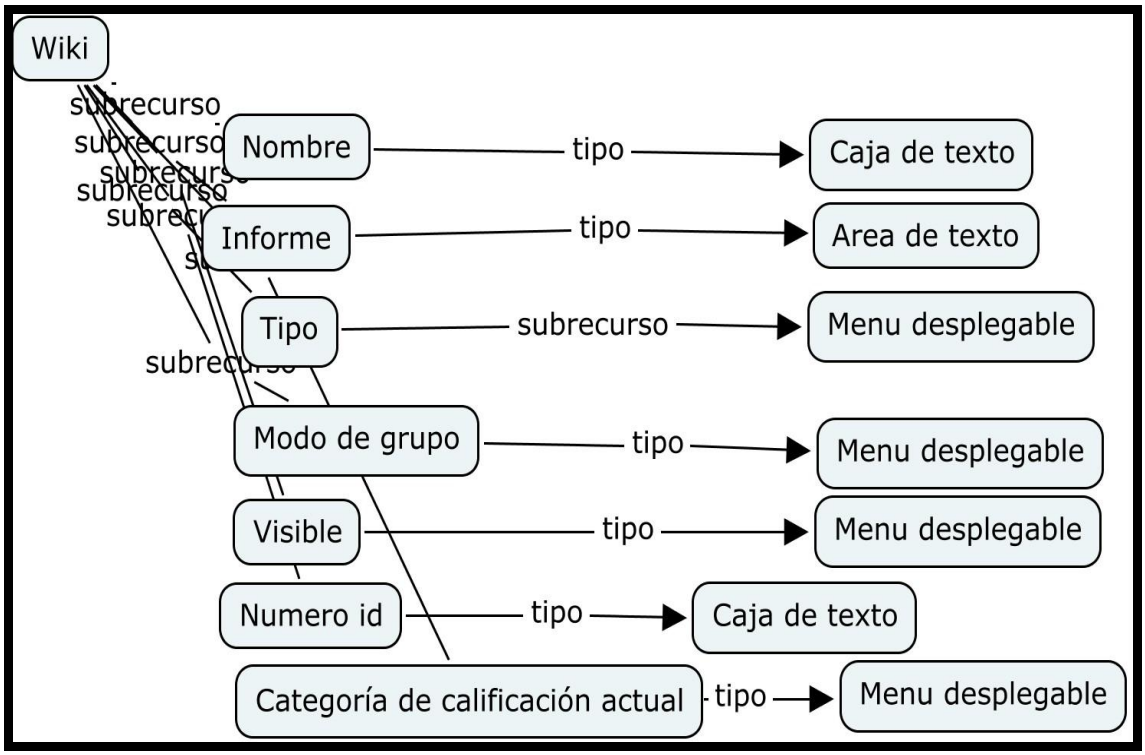


Figura 224 Vista recurso Wiki con CmapTools de Moodle.

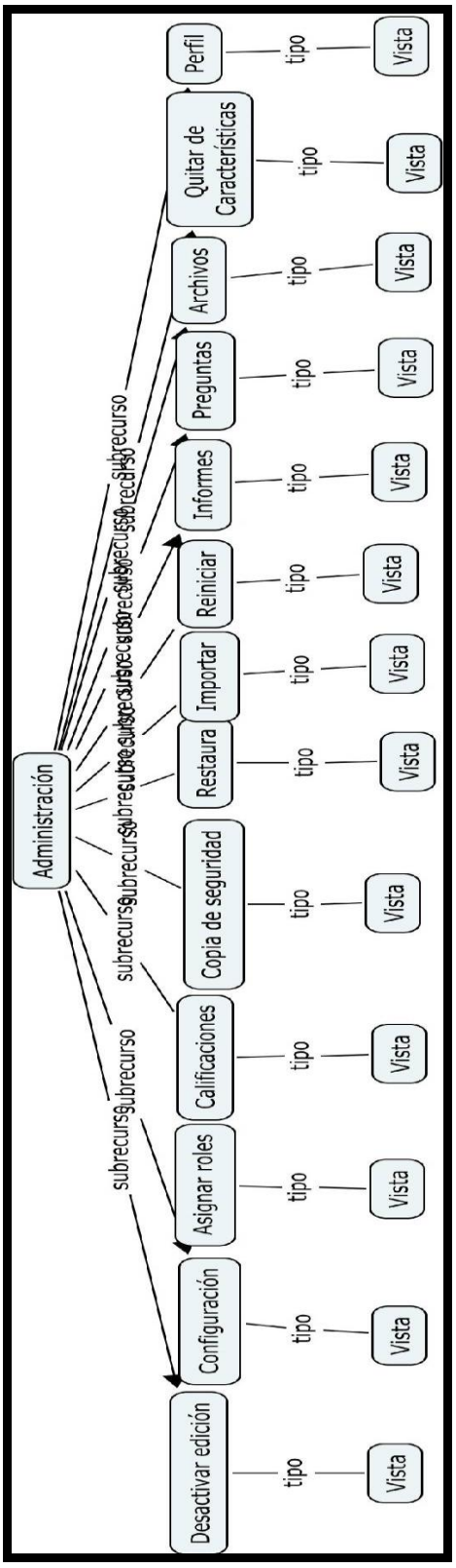


Figura 225 Vista recurso Administración con CmapTools de Moodle.

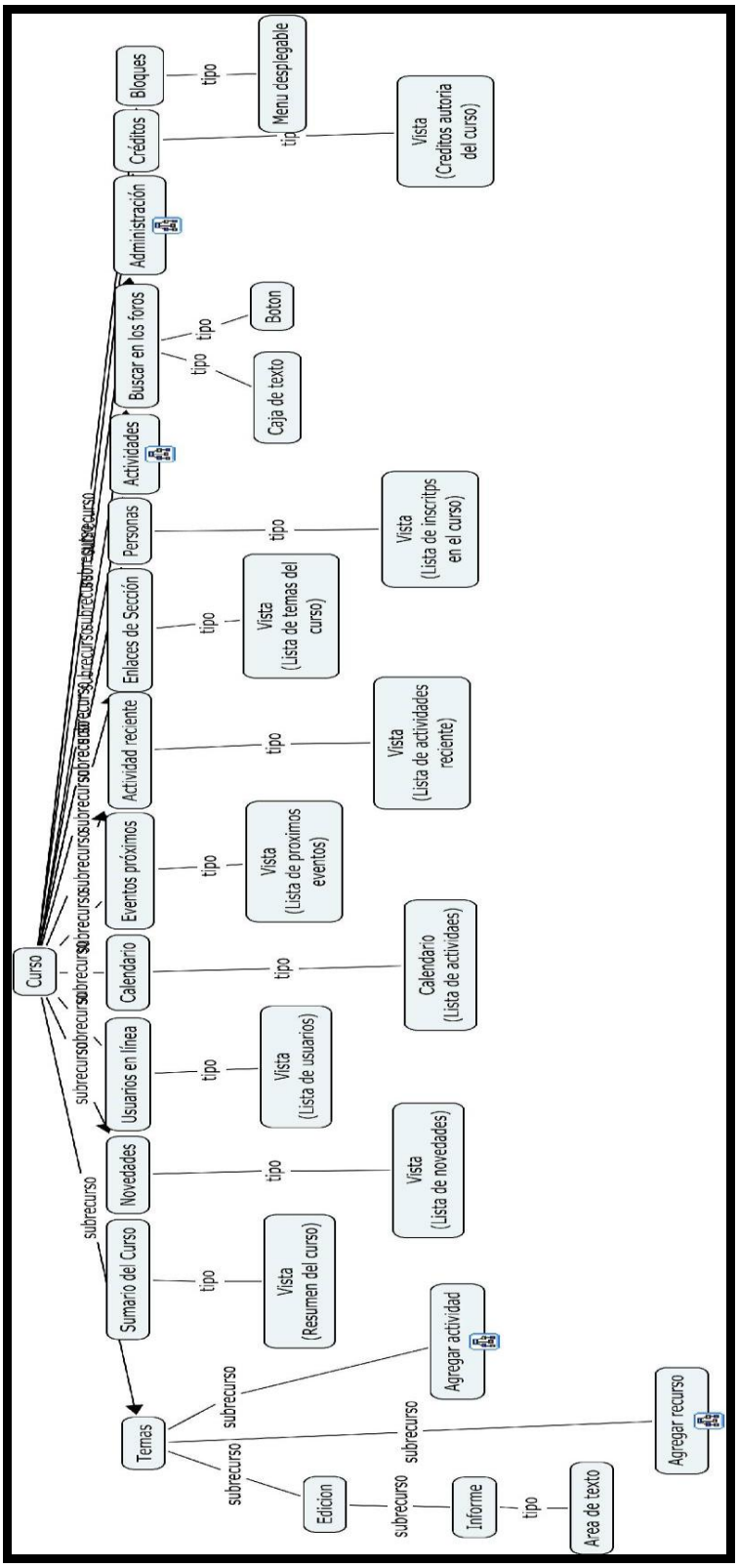


Figura 226 Vista recurso Curso con CmapTools de Moodle.

4. Mapa De Conocimiento Para DotLRN

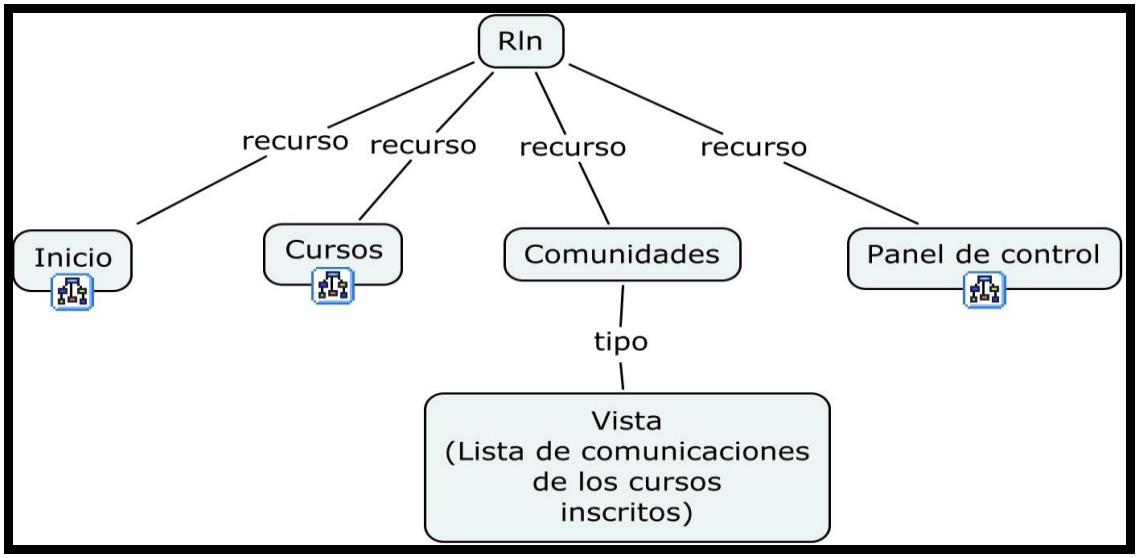


Figura 227 Vista Principal con CmapTools de .LRN.

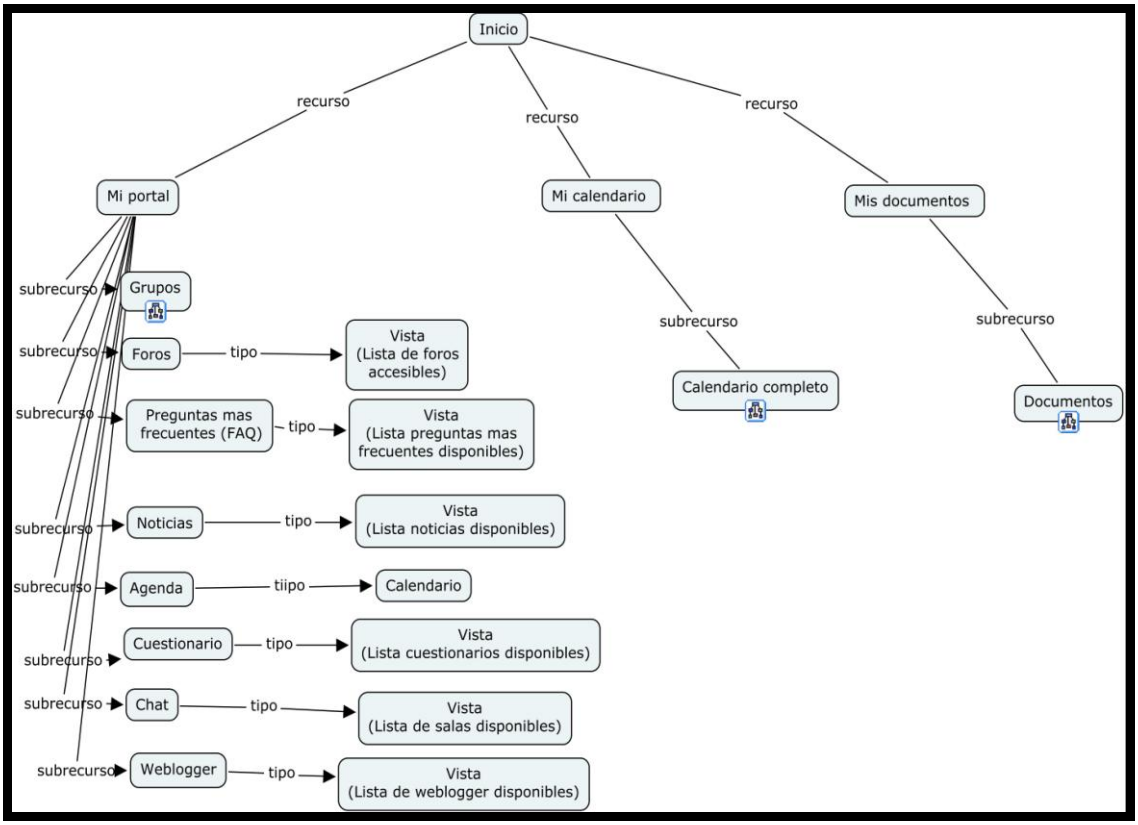


Figura 228 Vista recurso Inicio con CmapTools de .LRN.

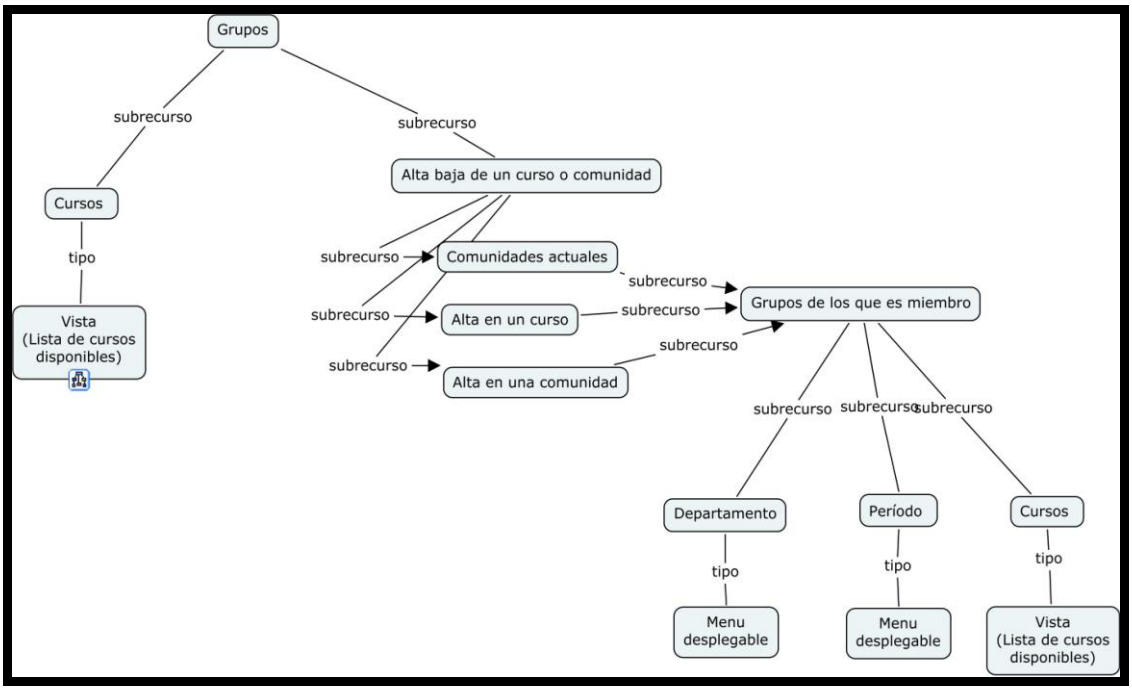


Figura 229 Vista recurso Grupos con CmapTools de .LRN.

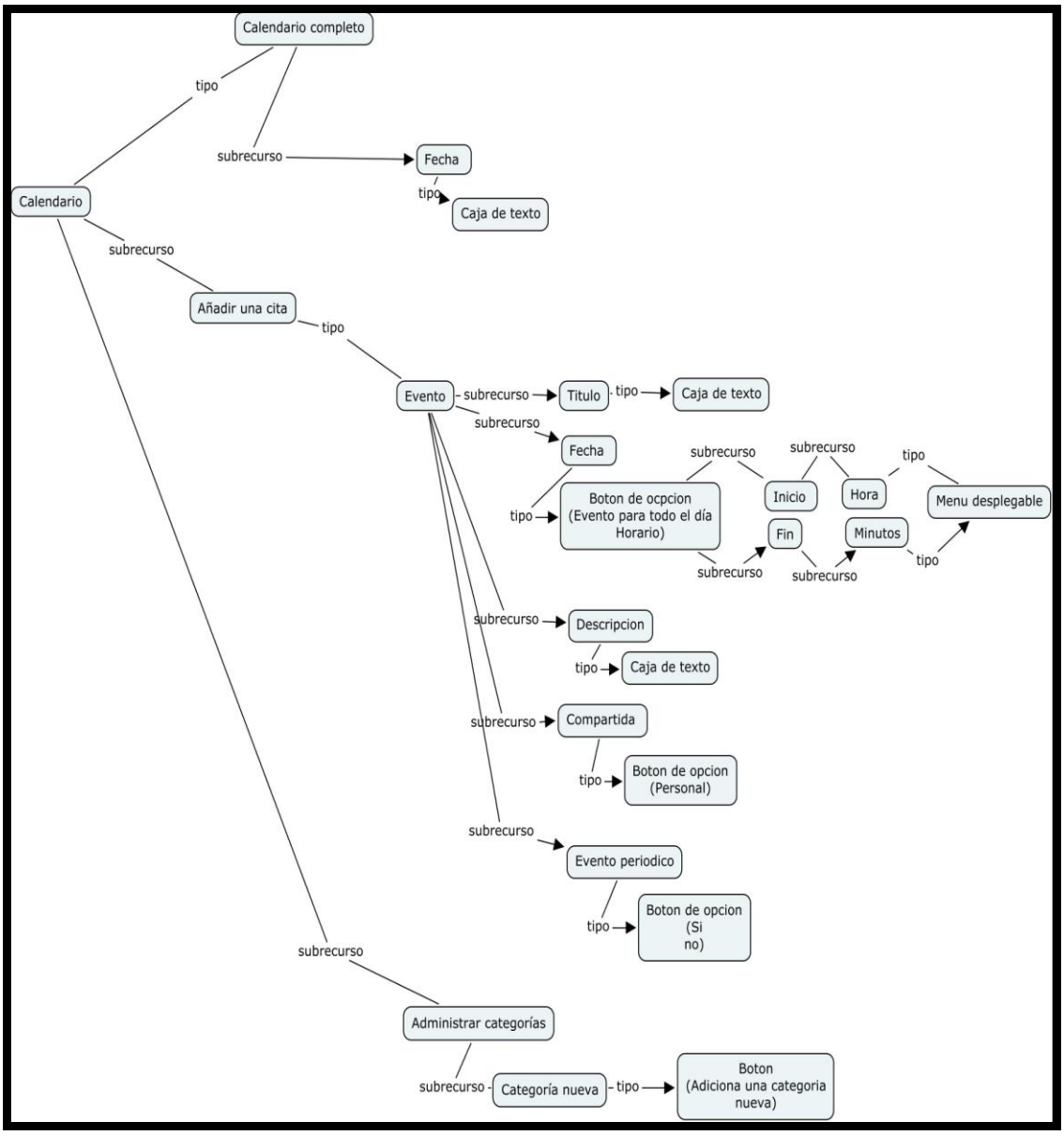


Figura 230 Vista recurso Calendario completo con CmapTools de .LRN.

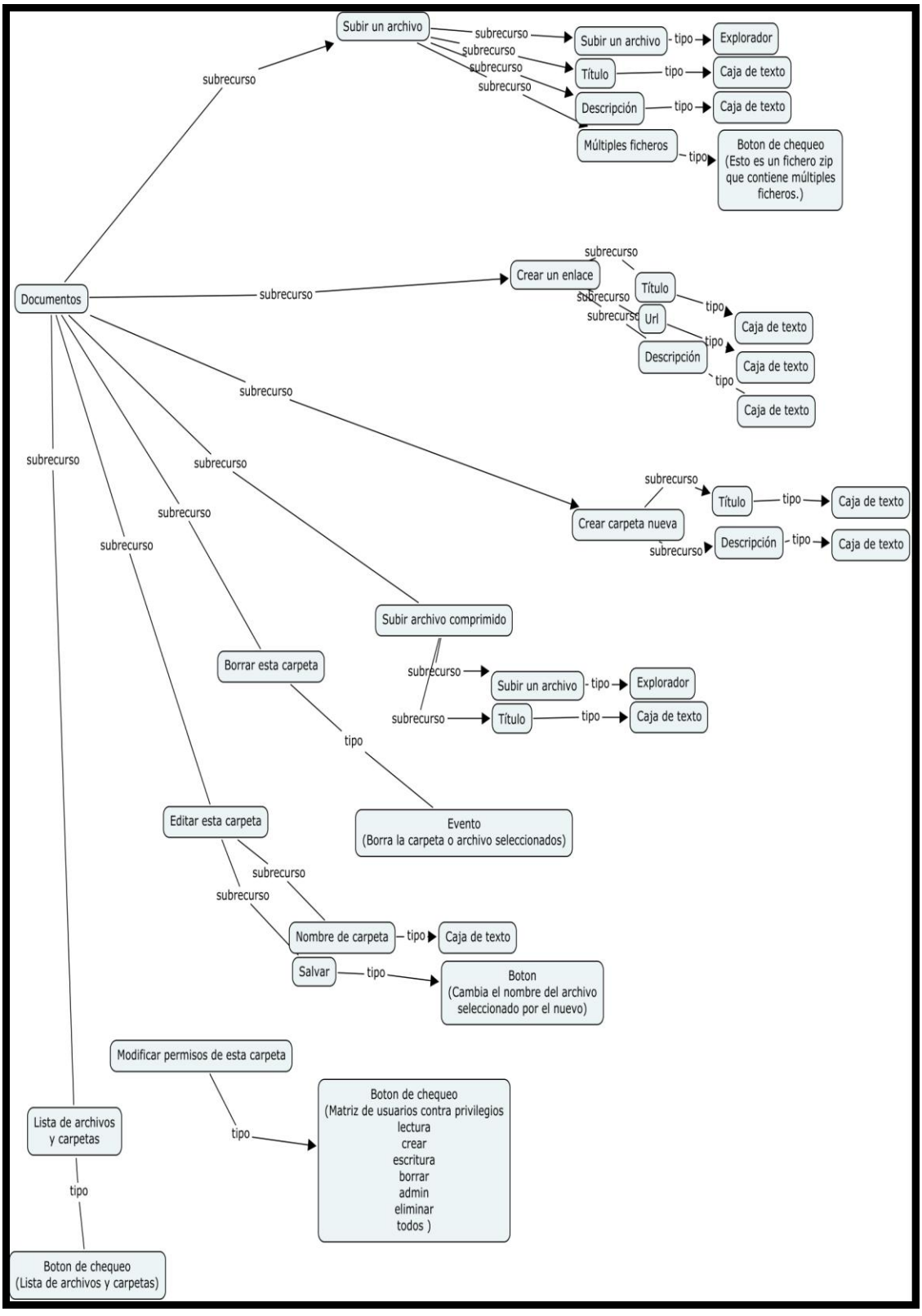


Figura 231 Vista recurso Documentos con CmapTools de .LRN.

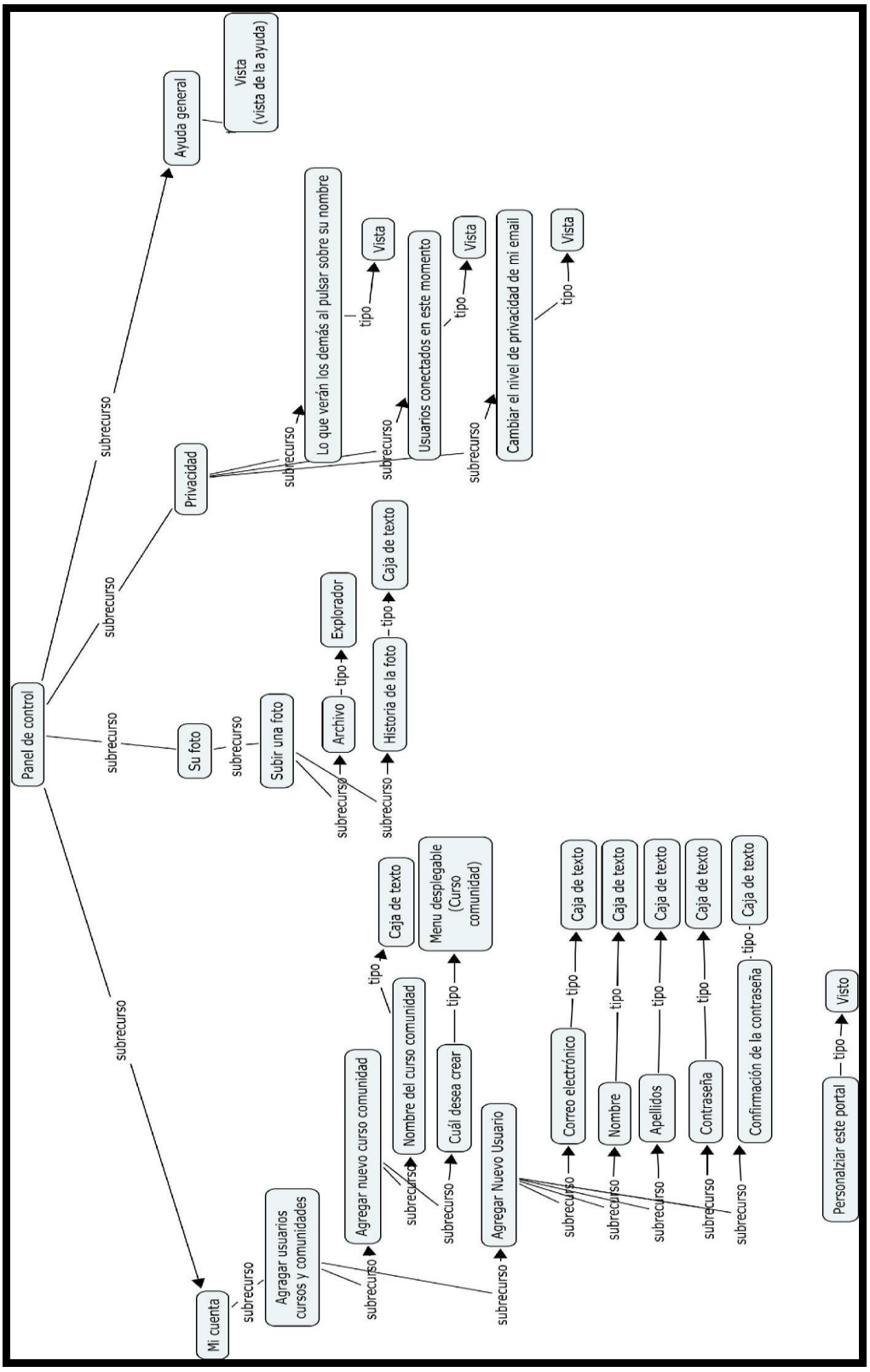


Figura 232 Vista recurso Panel de control con CmapTools de .LRN.

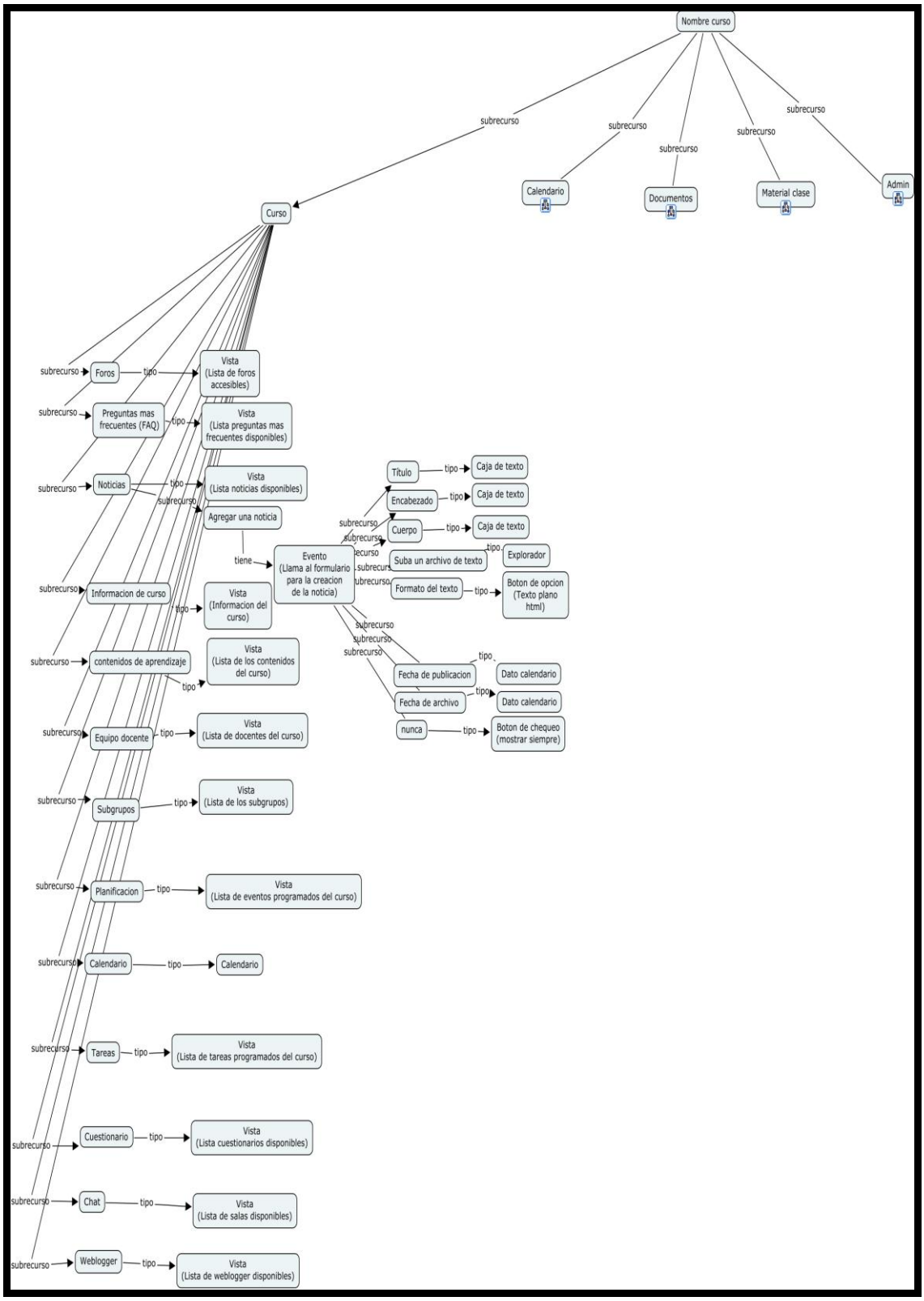


Figura 233 Vista recurso Nombre curso con CmapTools de .LRN.

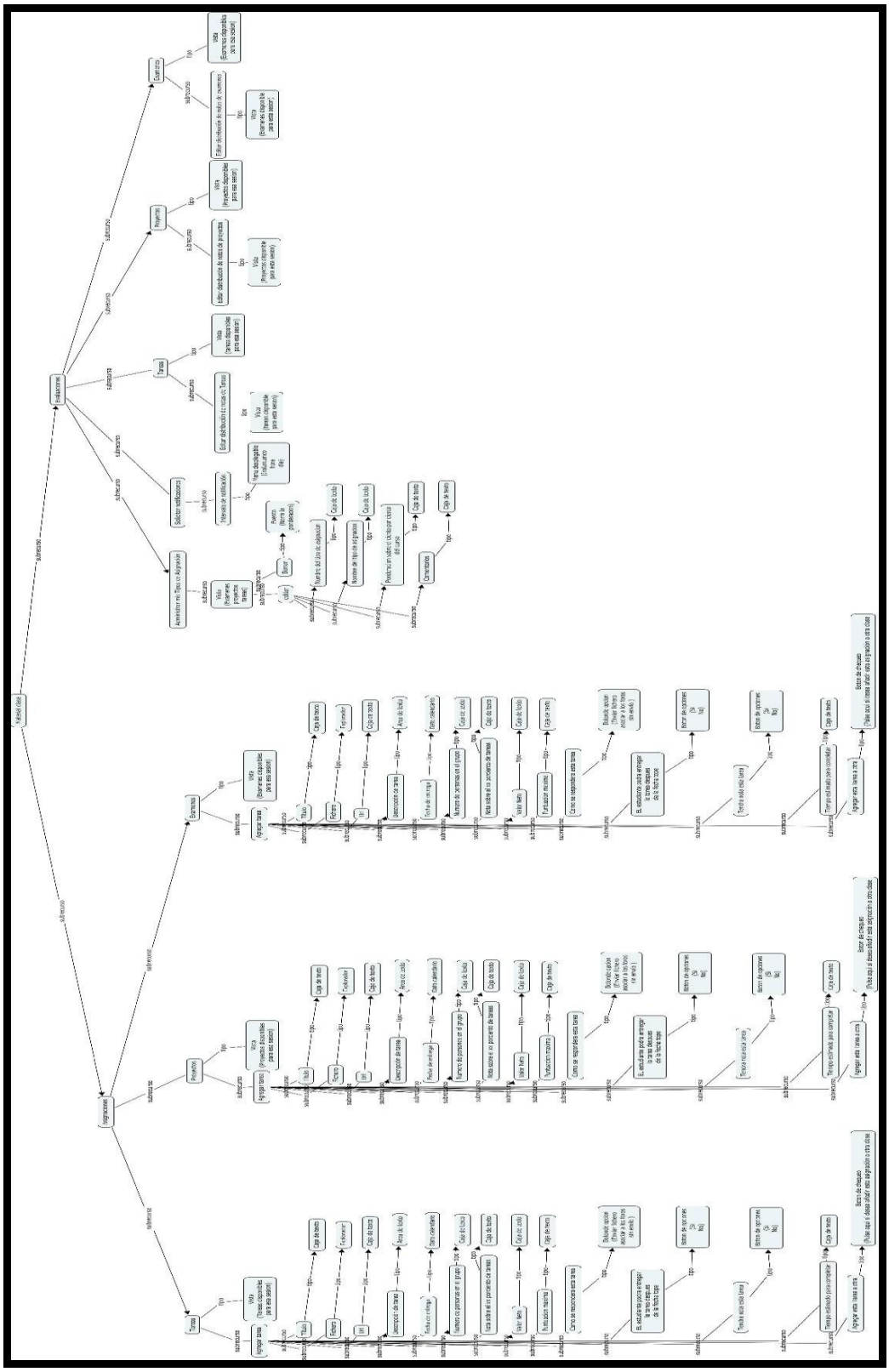


*Figura 234 Vista recurso Material clase con CmapTools de .L**R**N.*

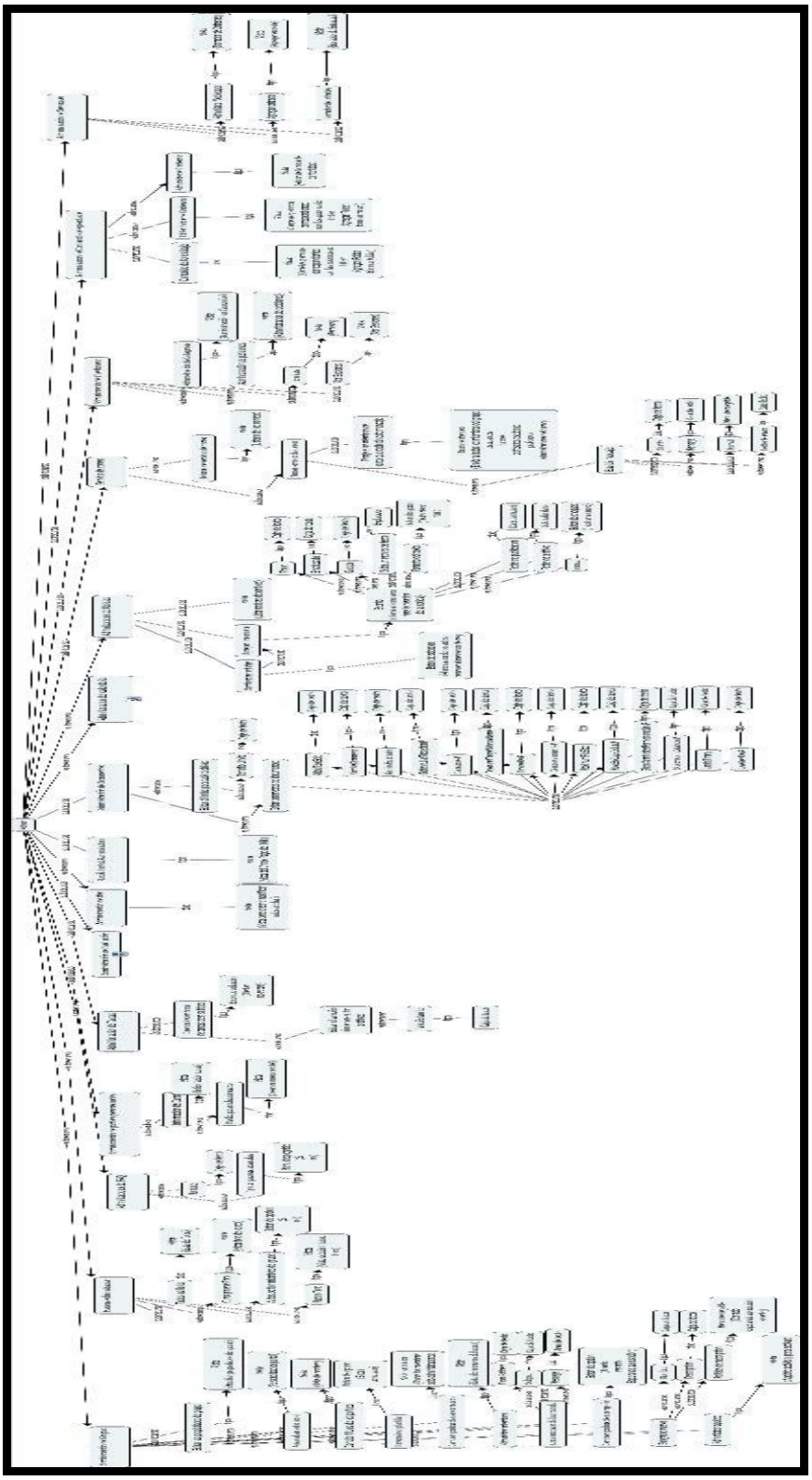


Figura 235 Vista recurso Admin con CmapTools de .LRN.

5. Mapa De Conocimiento Para Sakai

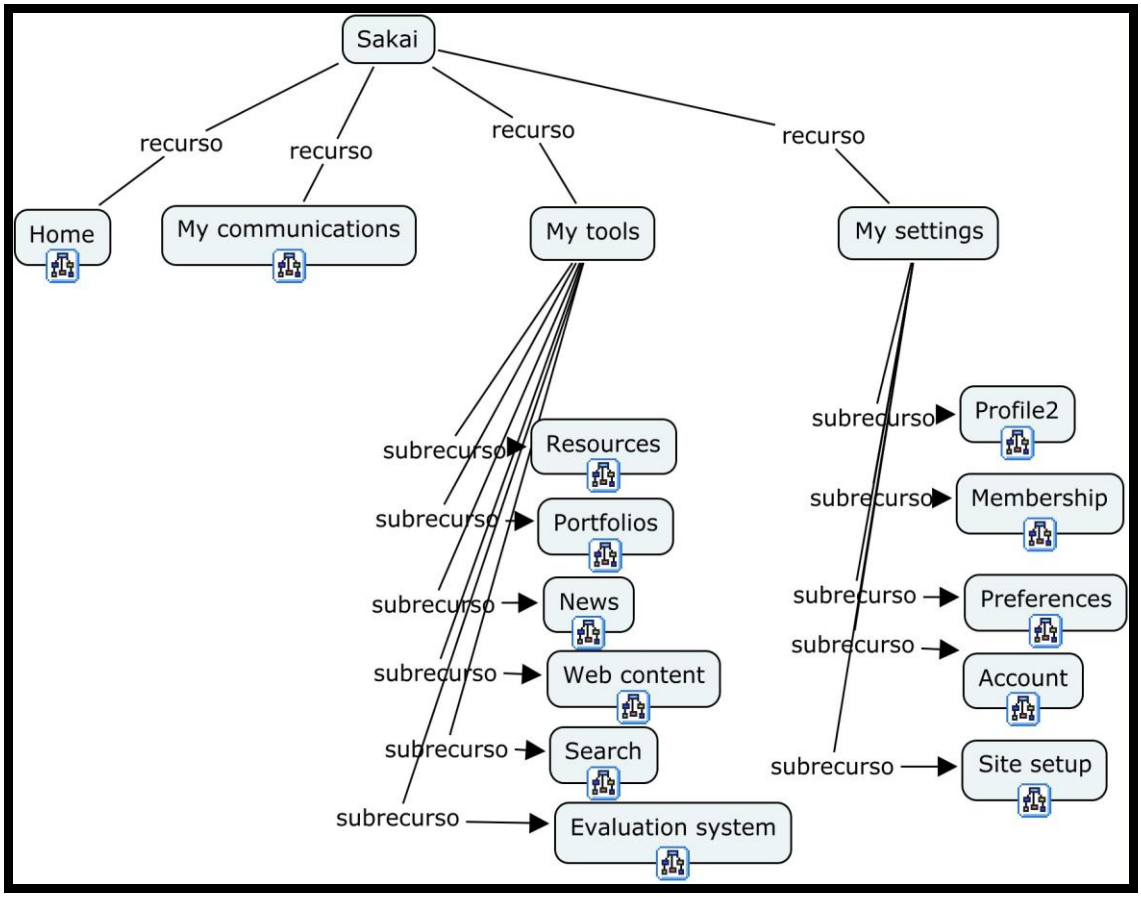


Figura 236 Vista principal con CmapTools de Sakai.

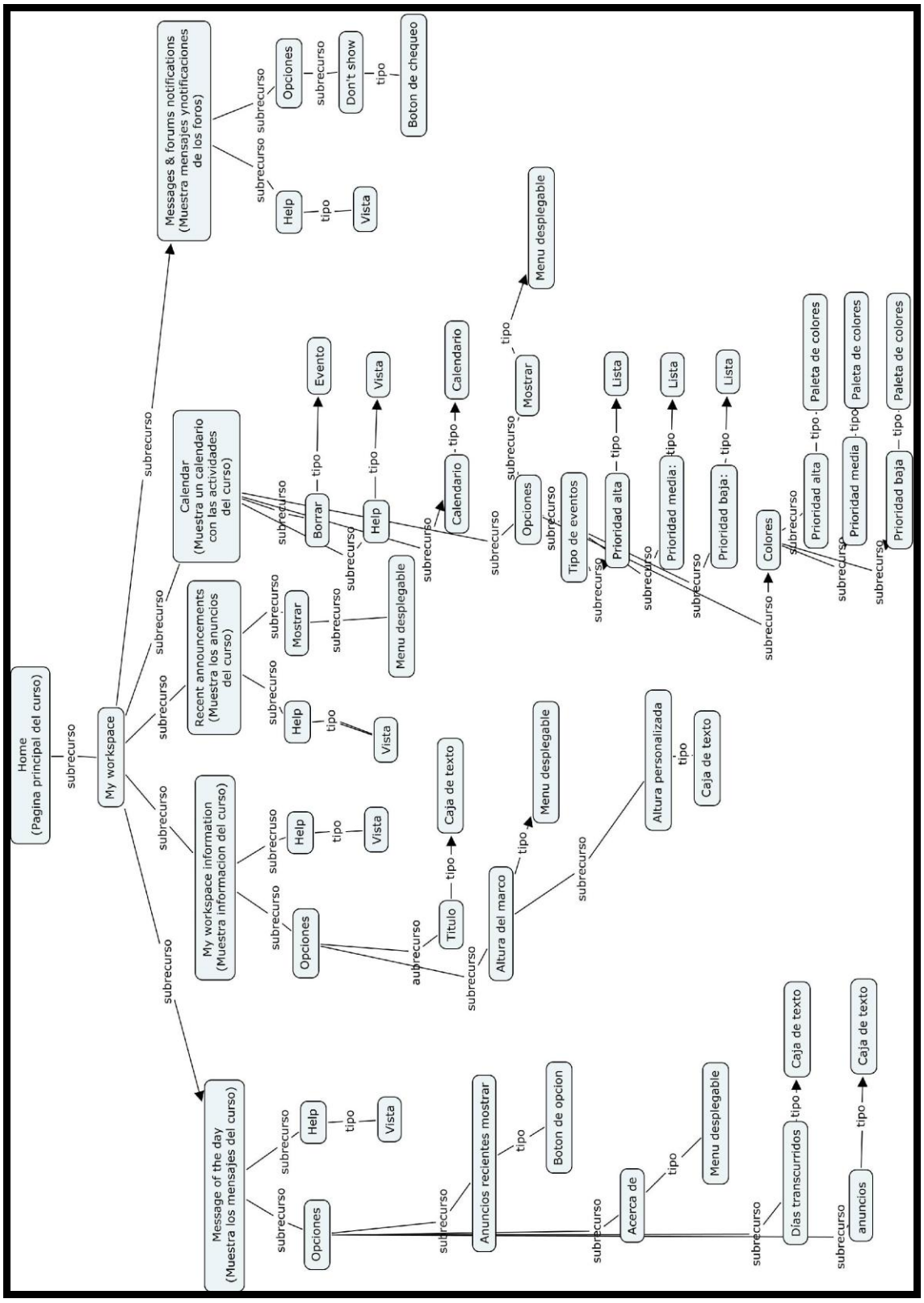


Figura 237 Vista recurso Home con CmapTools de Sakai.

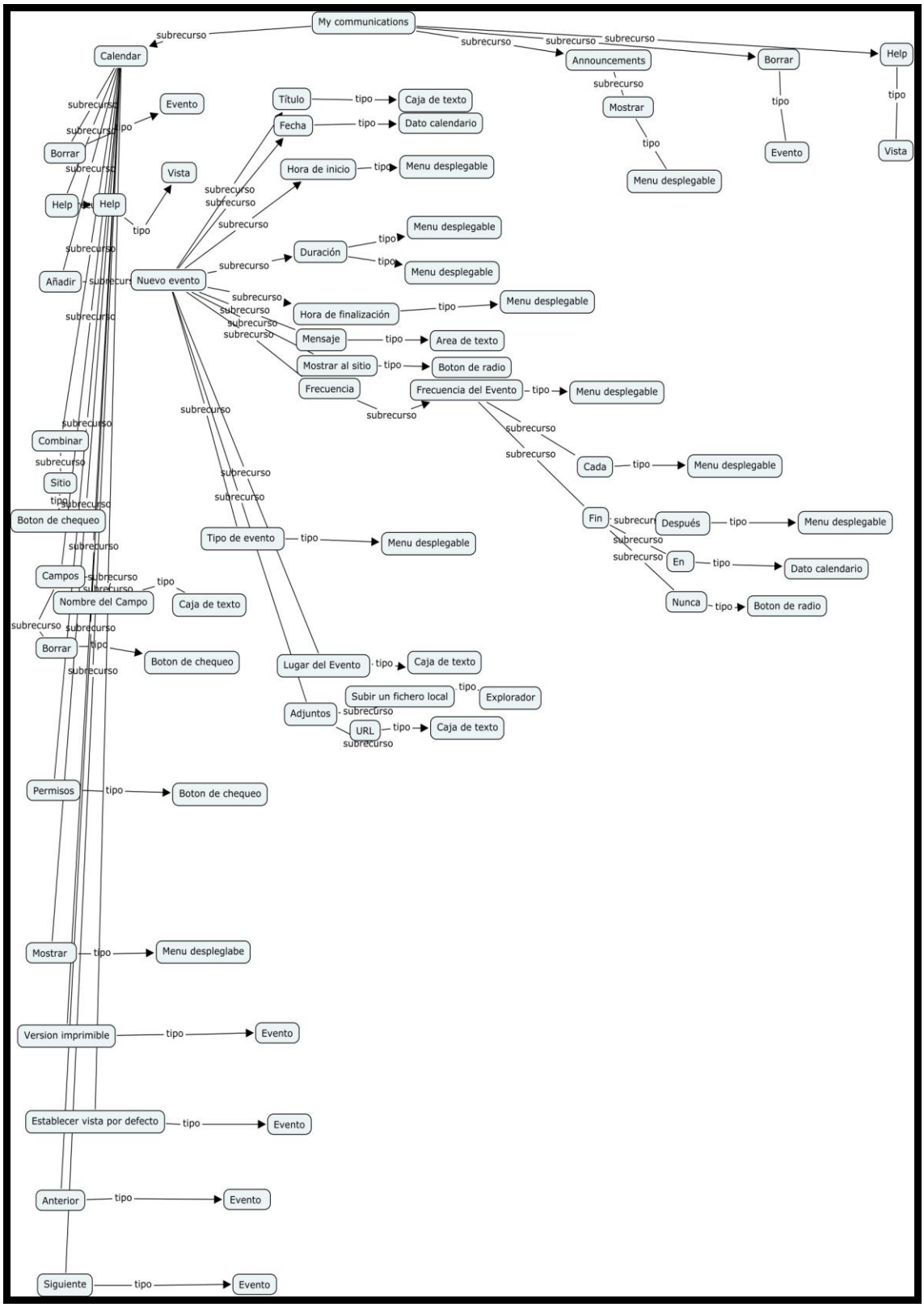


Figura 238 Vista recurso My communications con CmapTools de Sakai.

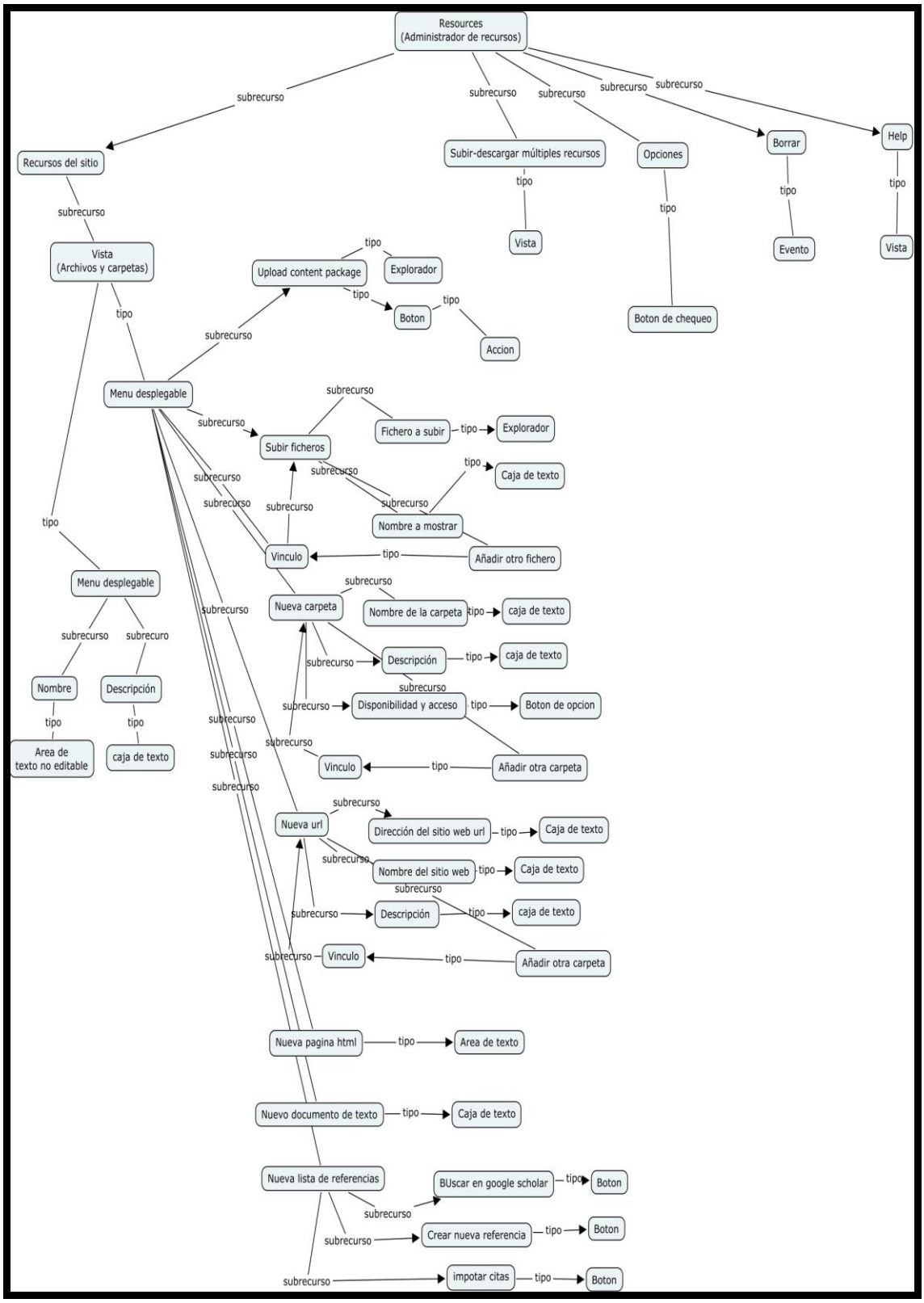


Figura 239 Vista recurso Resources con CmapTools de Sakai.

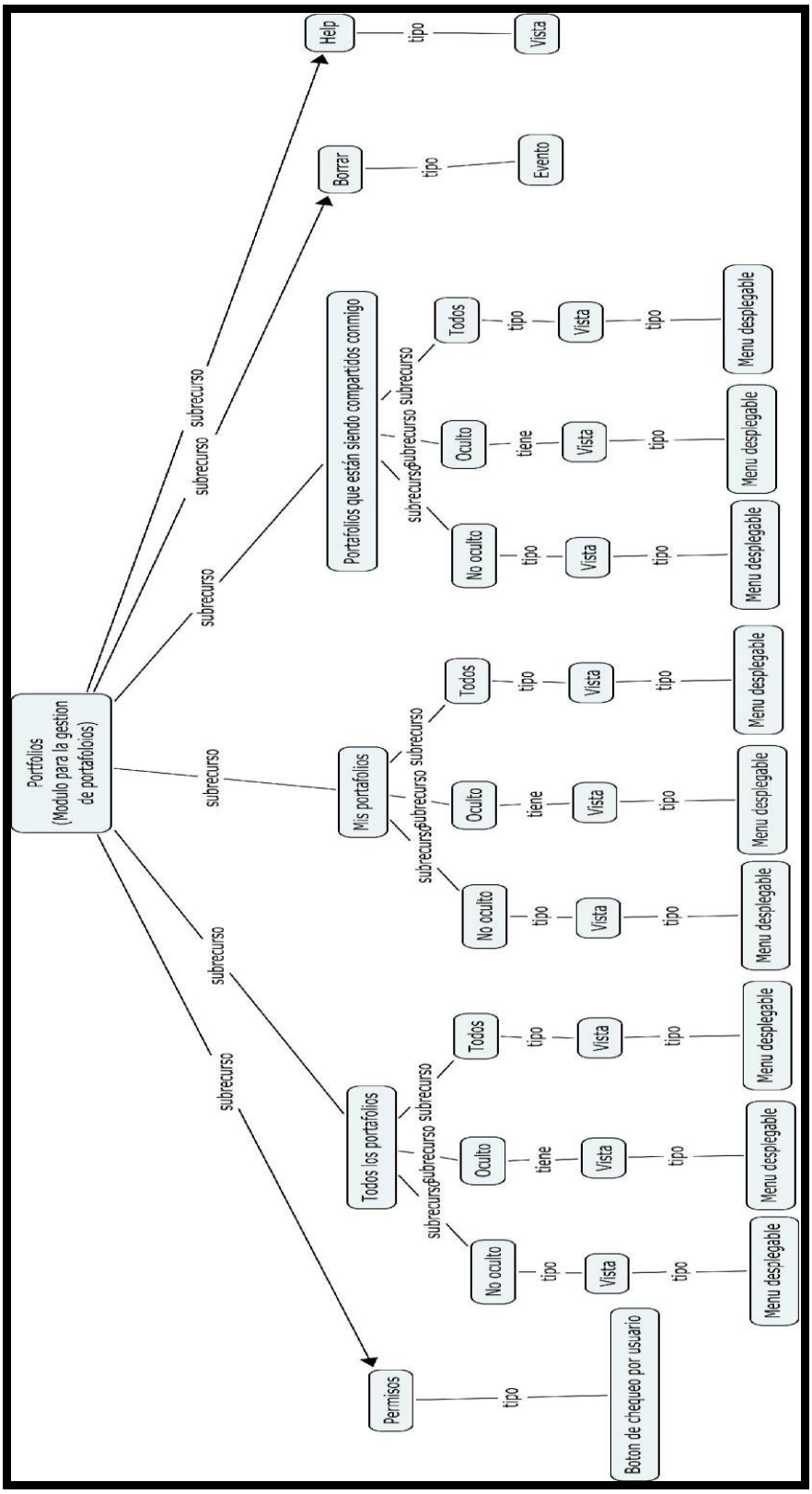


Figura 240 Vista recurso portfolios con CmapTools de Sakai.

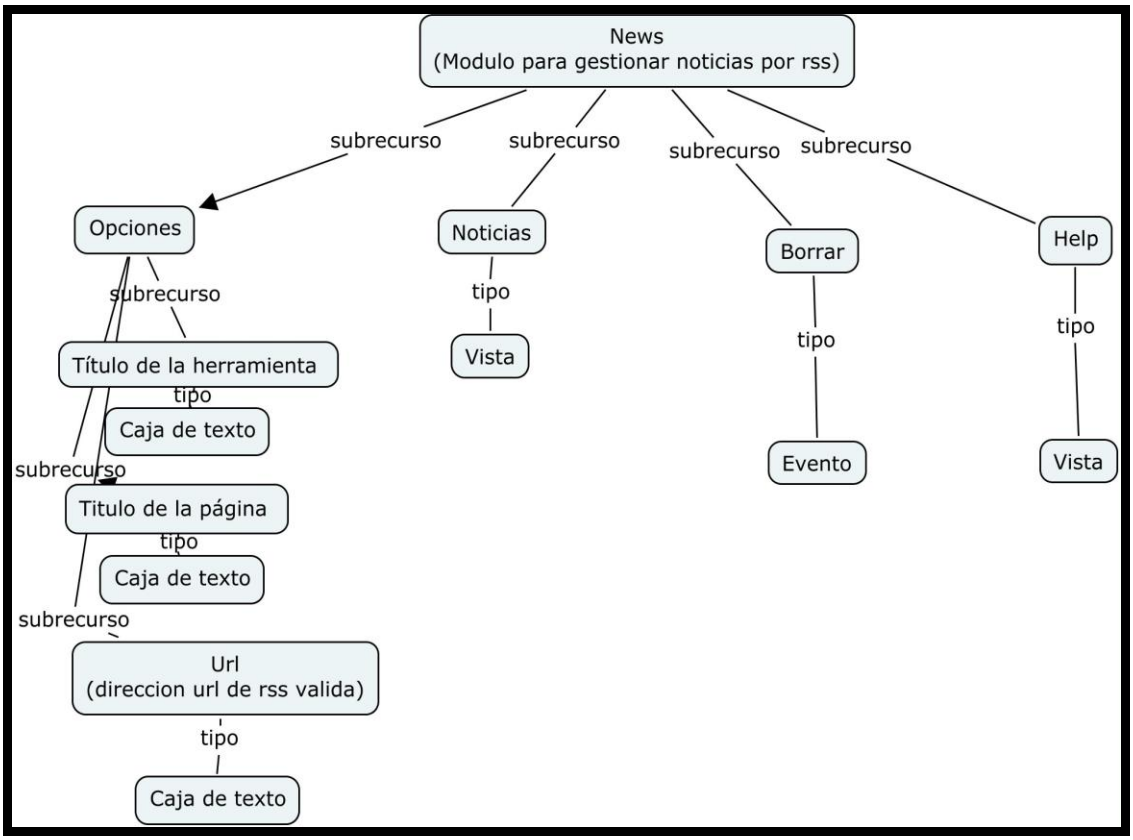


Figura 241 Vista recurso News con CmapTools de Sakai.

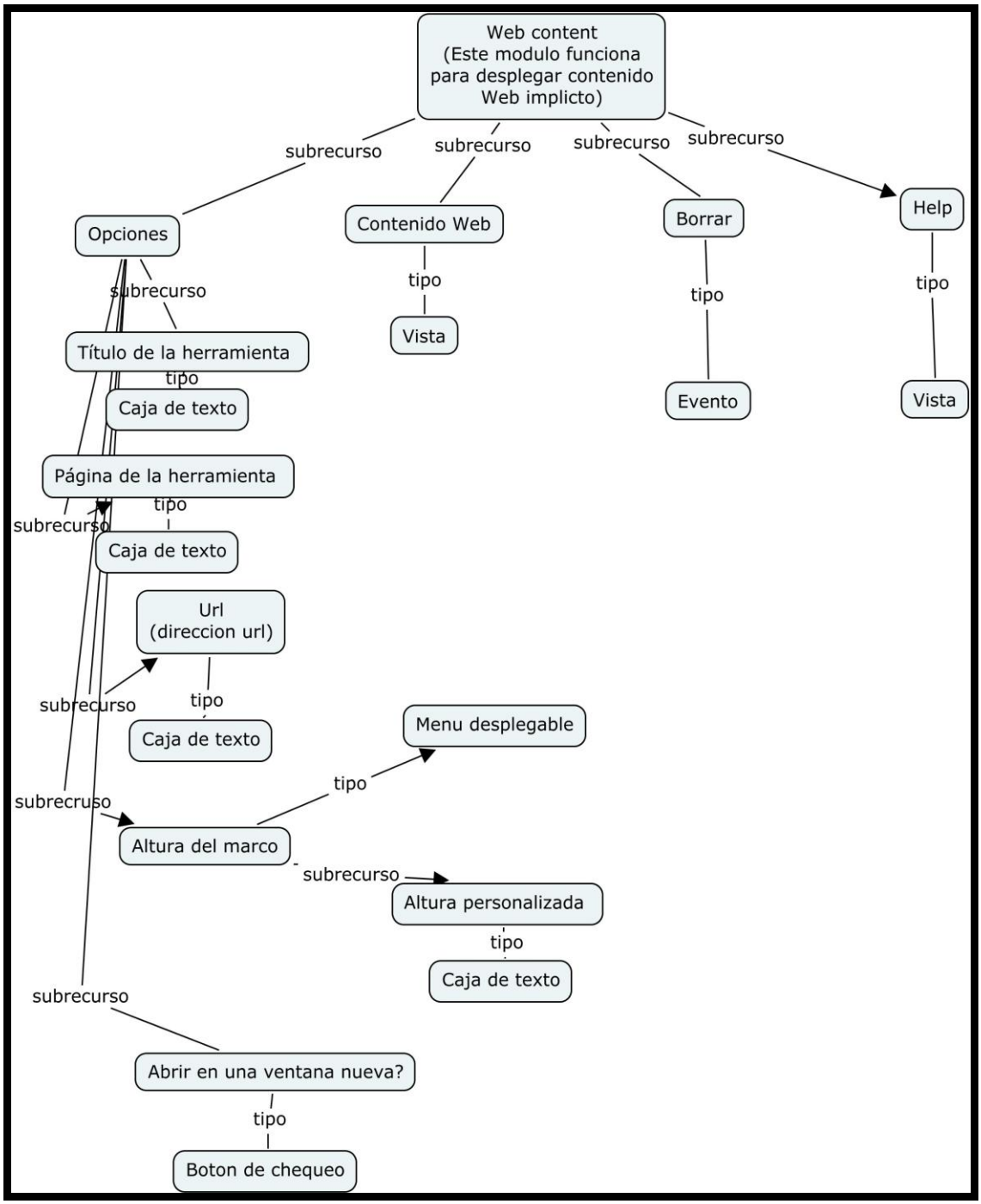


Figura 242 Vista recurso Web content con CmapTools de Sakai.

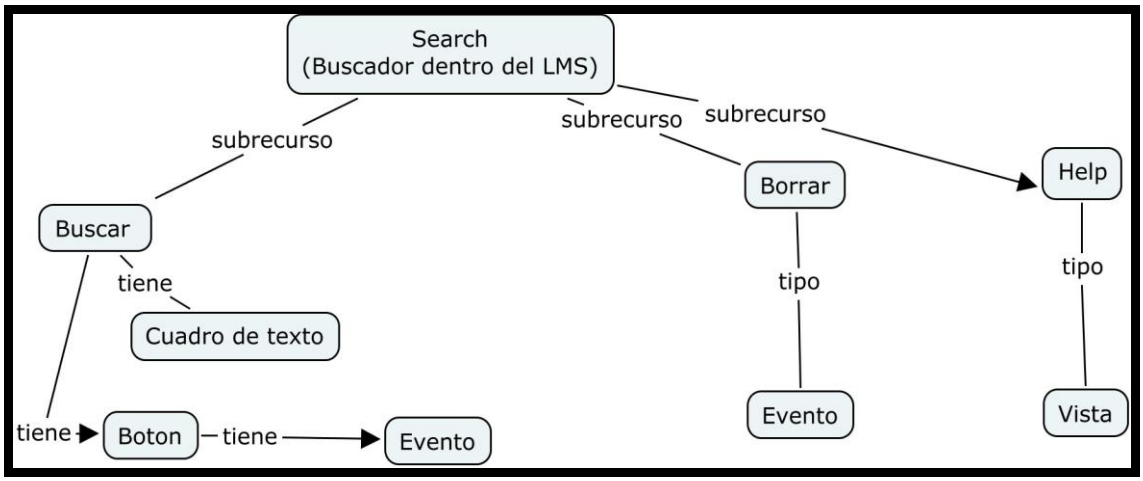


Figura 243 Vista recurso Search con CmapTools de Sakai.

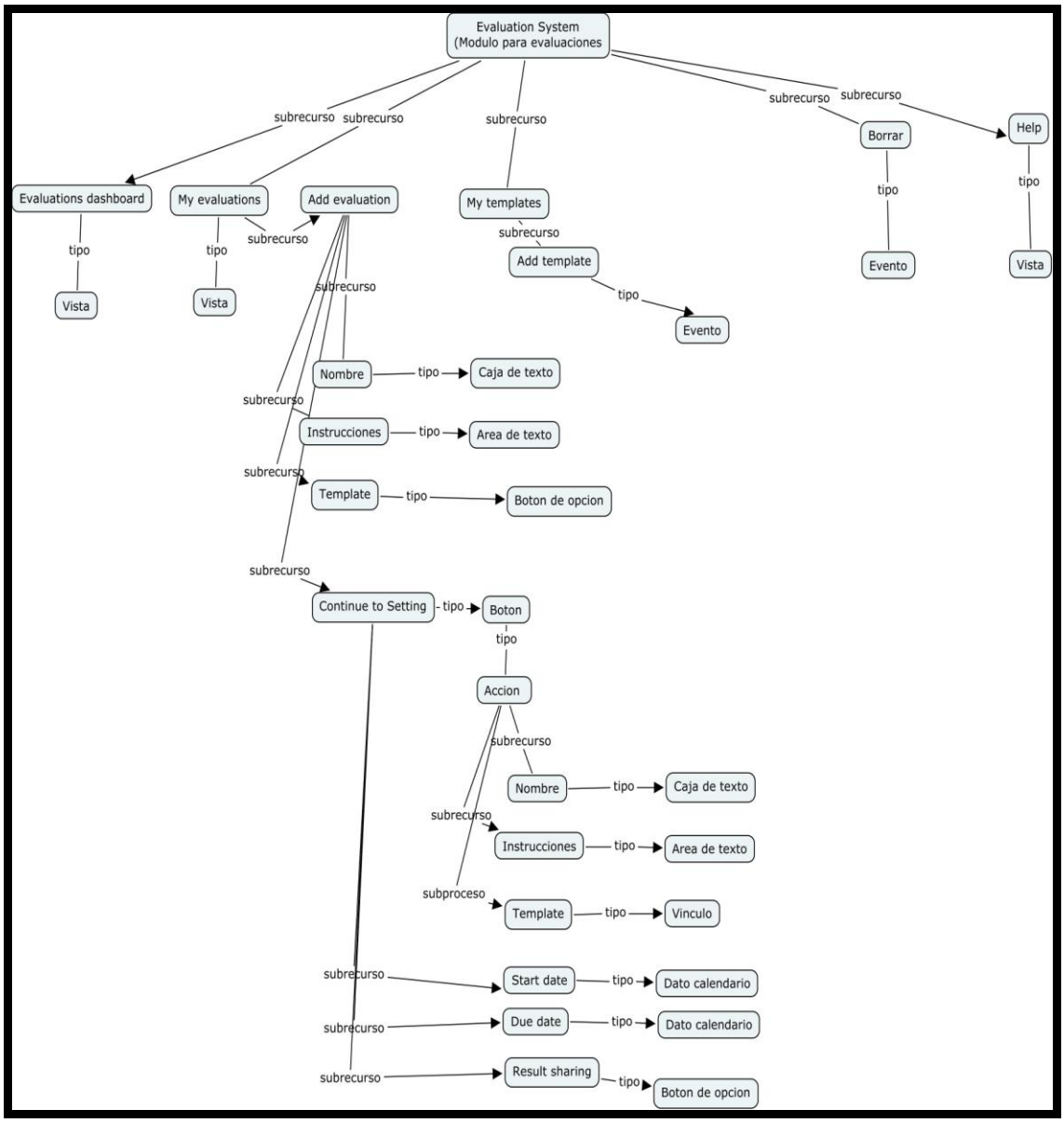


Figura 244 Vista recurso Evaluation System con CmapTools de Sakai.

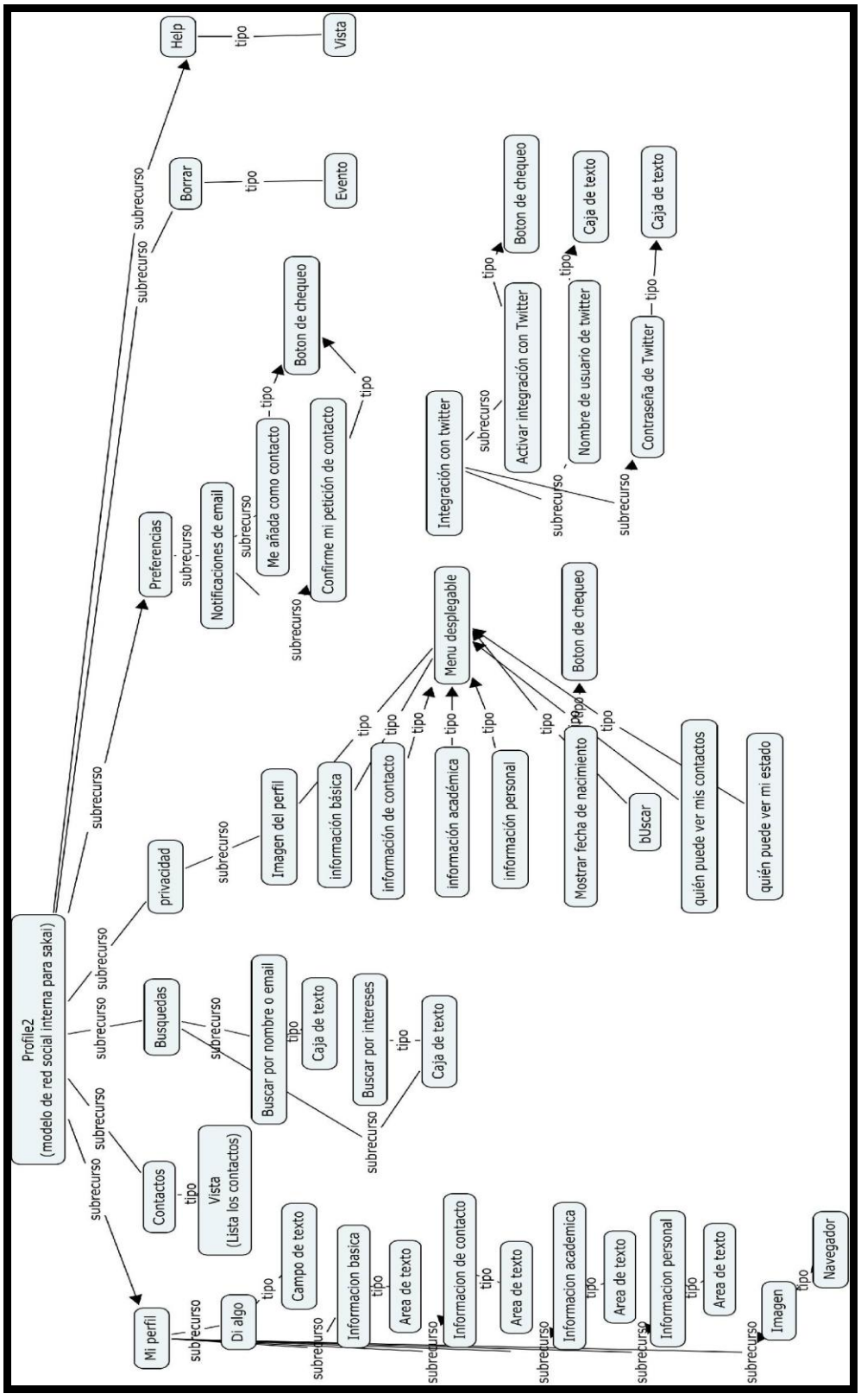


Figura 245 Vista recurso Profile 2 con CmapTools de Sakai.

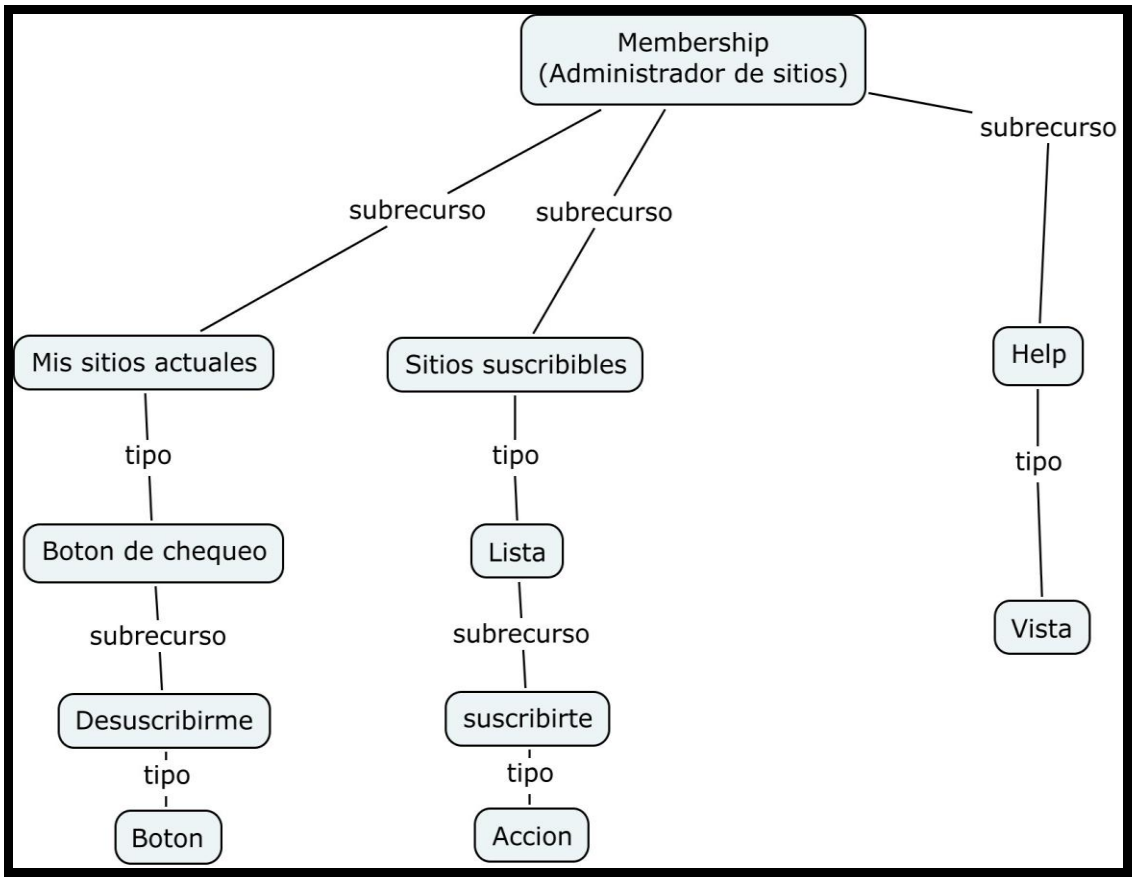


Figura 246 Vista recurso Membership con CmapTools de Sakai.

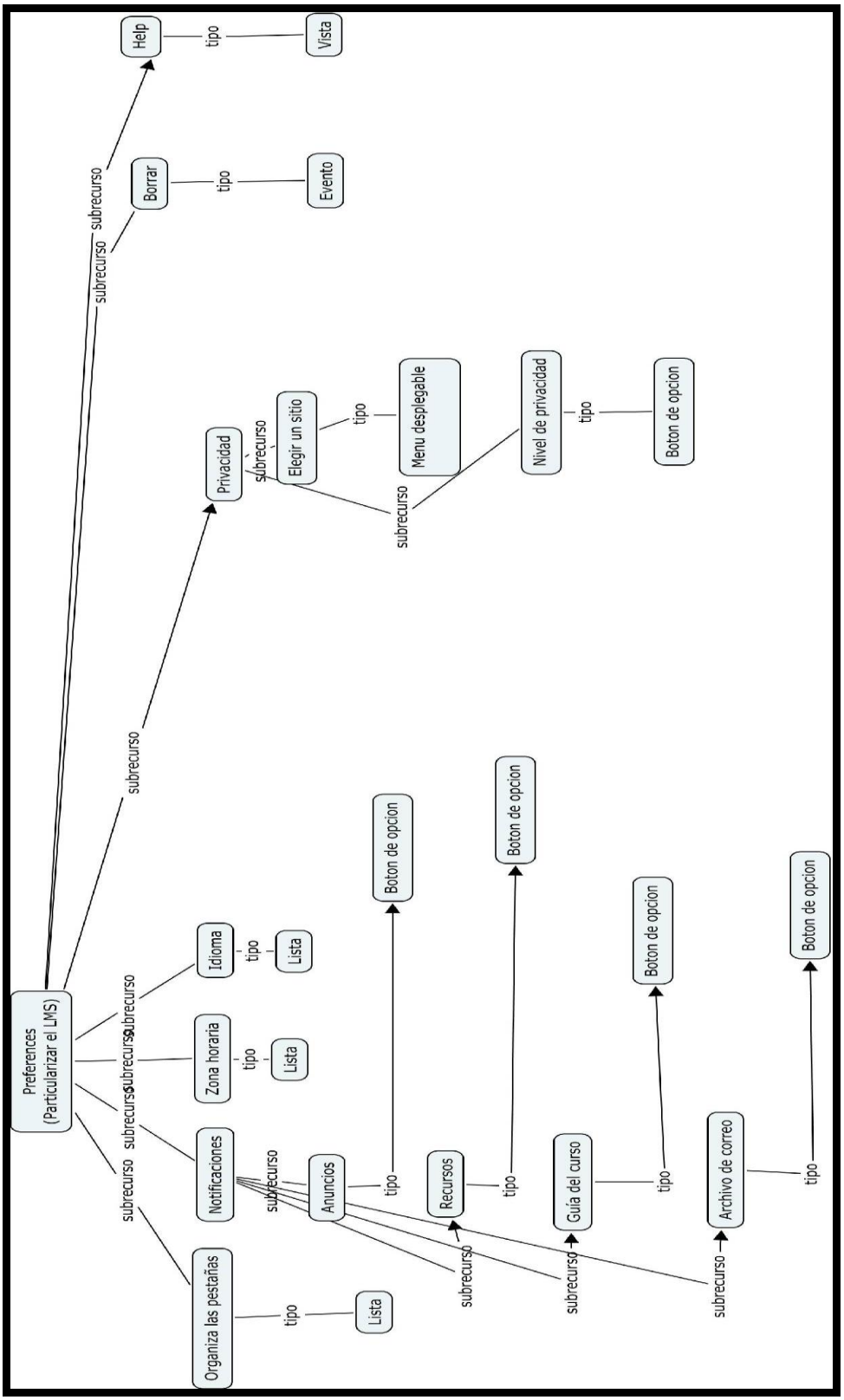


Figura 247 Vista recurso Preferences con CmapTools de Sakai.

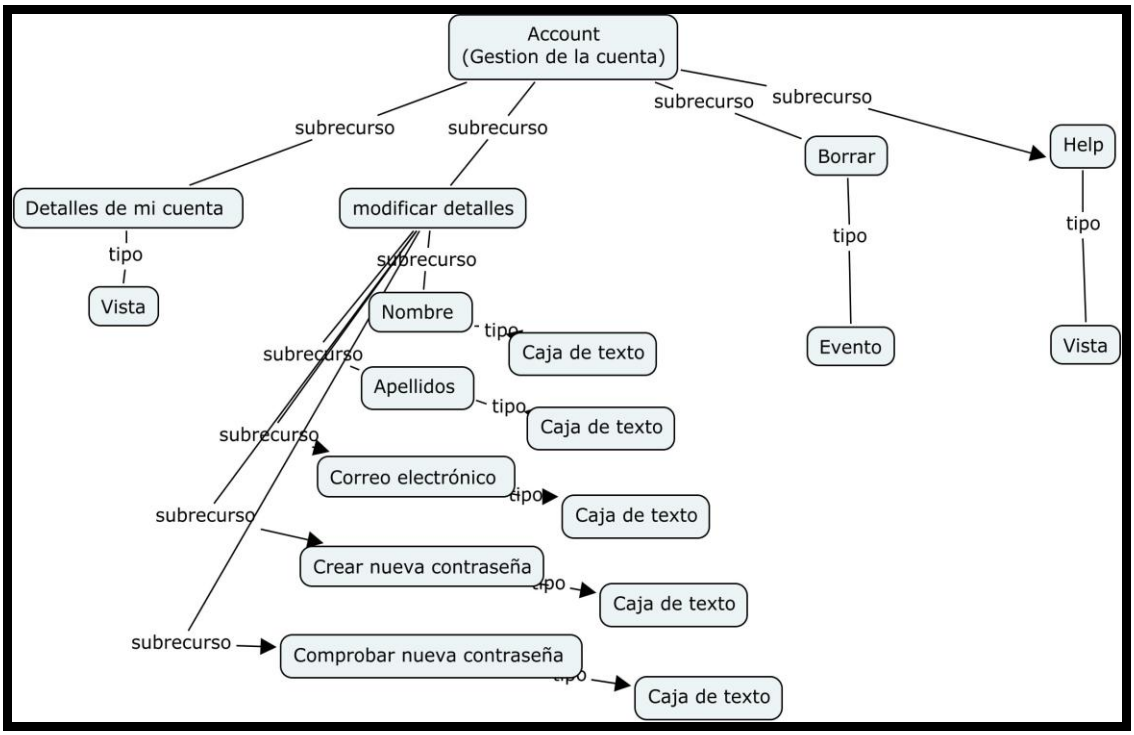


Figura 248 Vista recurso Account con CmapTools de Sakai.

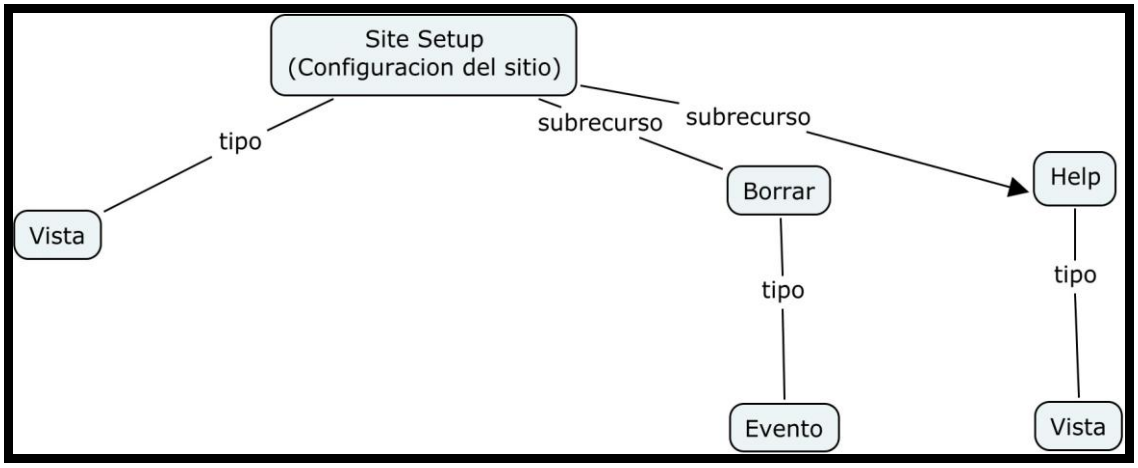


Figura 249 Vista recurso Site setup con CmapTools de Sakai.

7 Anexo II

Manual EMF y GMF – Instalación de MOFScript

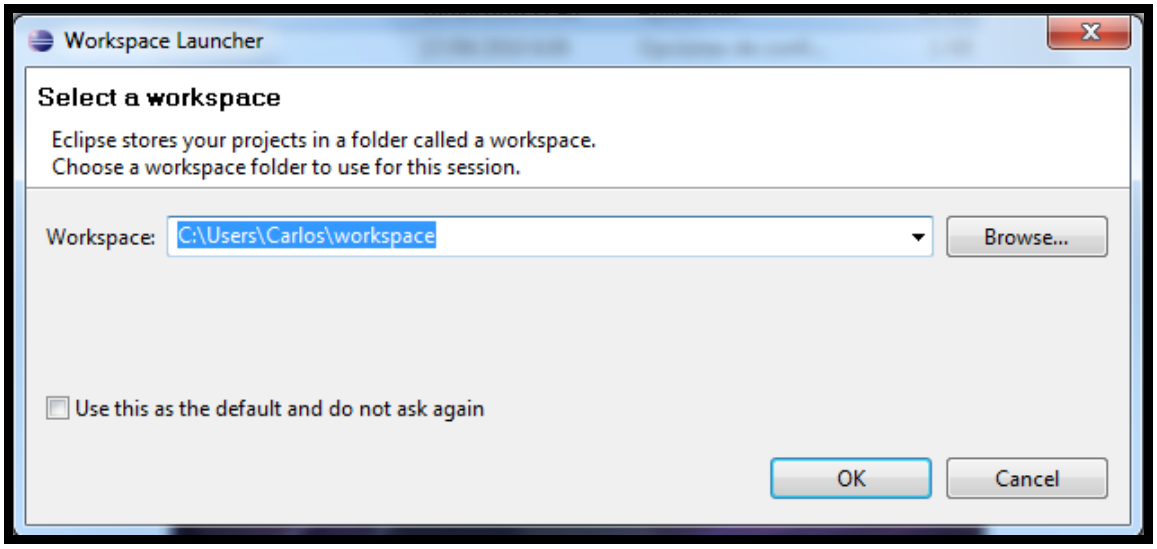
Guía para el desarrollador

Esta es una sencilla guía para aprender a manejar EMF y GMF bajo Eclipse.

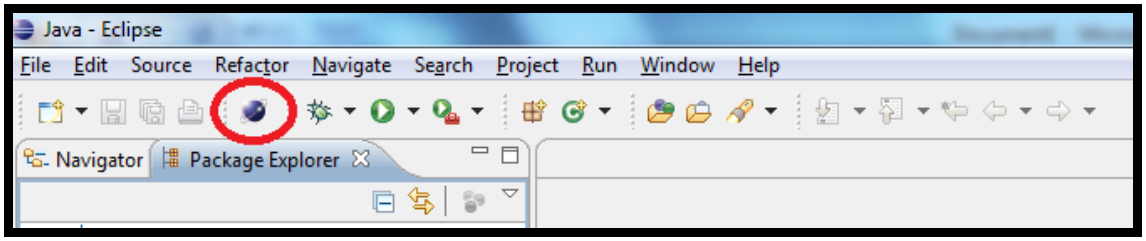
Carlos Enrique Montenegro Marín

1. Instalación de Eclipse Modeling Framework para Helios Service Release 1

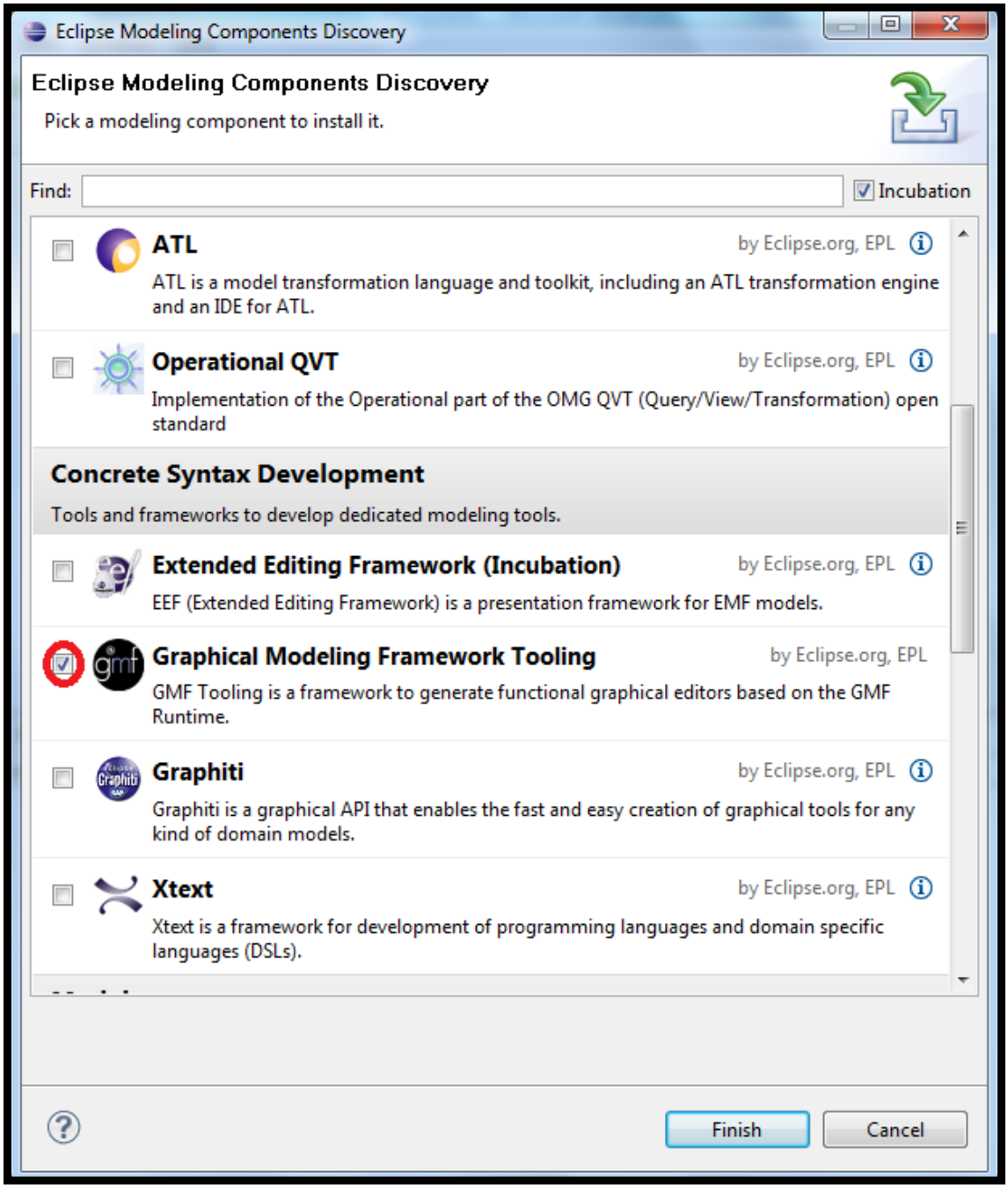
1. Ingresar a la Web de eclipse Project http://www.eclipse.org/
2. Seleccione Descargas http://www.eclipse.org/downloads/
3. Descargue el paquete Eclipse Modeling Tools (includes Incubating components)
4. Para la instalación sencillamente descomprima el fichero en cualquier ruta con permisos de lectura y escritura preferiblemente, para este caso la hemos descomprimido en "C:\eclipse".
5. Ejecute el fichero "eclipse. exe" para ejecutar eclipse
6. Seleccione el área de trabajo para almacenar los trabajos realizados, como se muestra en la siguiente figura.



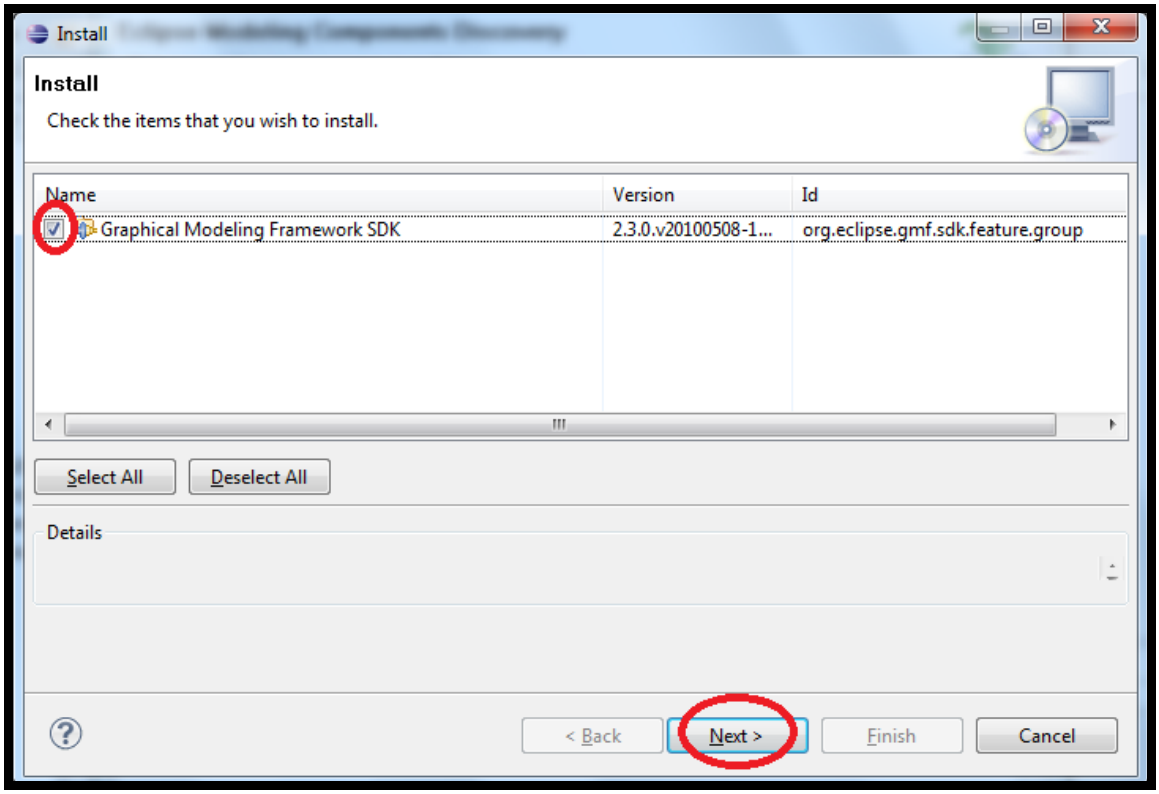
7. Instalar los componentes de modelado necesario para trabajar, para ello haga clic sobre el icono "Install Modeling Components" , ubicado en la barra de herramientas de eclipse, como se muestra en la siguiente figura.



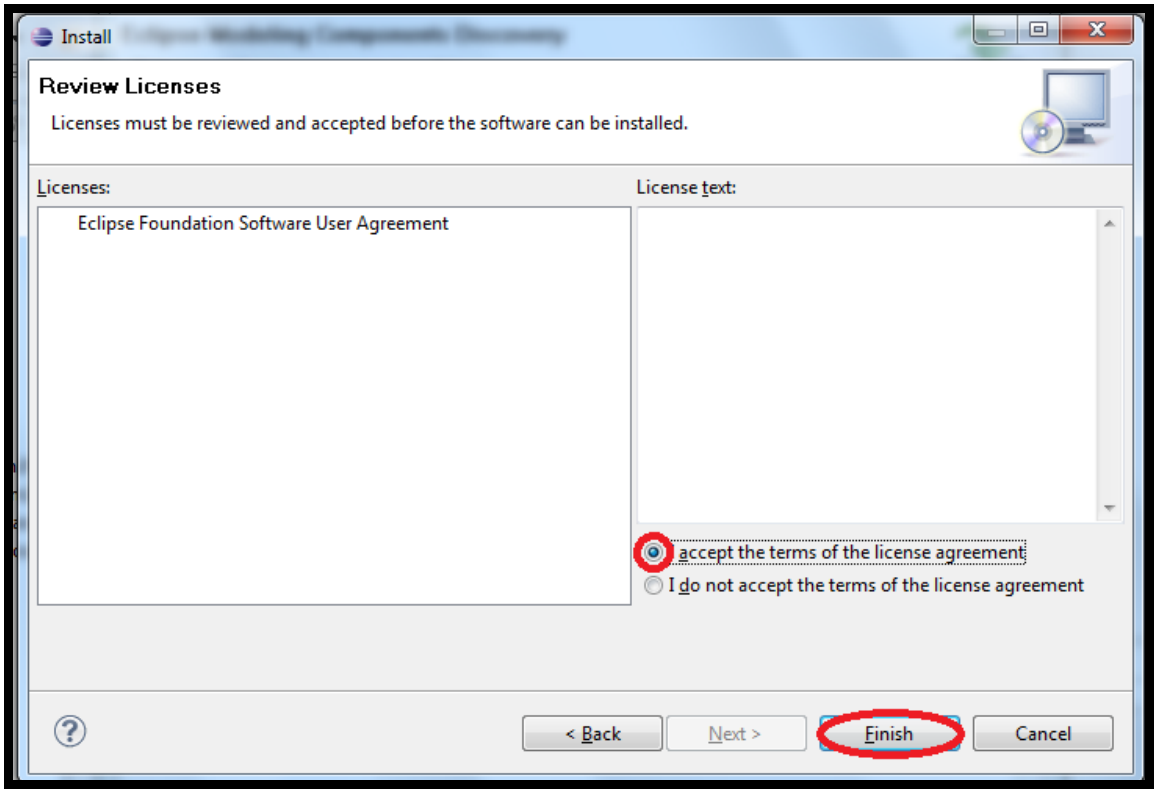
8. Aparecerá una nueva ventana con los componentes que deseemos instalar, para nuestro caso seleccionaremos únicamente "Graphical Modeling Framework Tooling (GMF)" de esta lista, como se muestra en la siguiente figura.



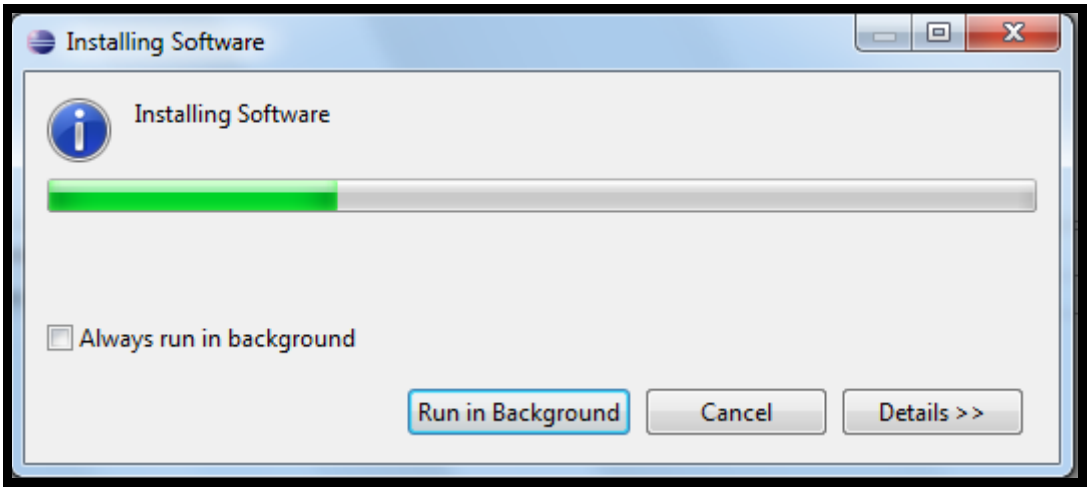
9. Esperamos un momento y confirmamos la instalación del plugin dando clic en “next” para la ventana que aparece, como se muestra en la siguiente figura.



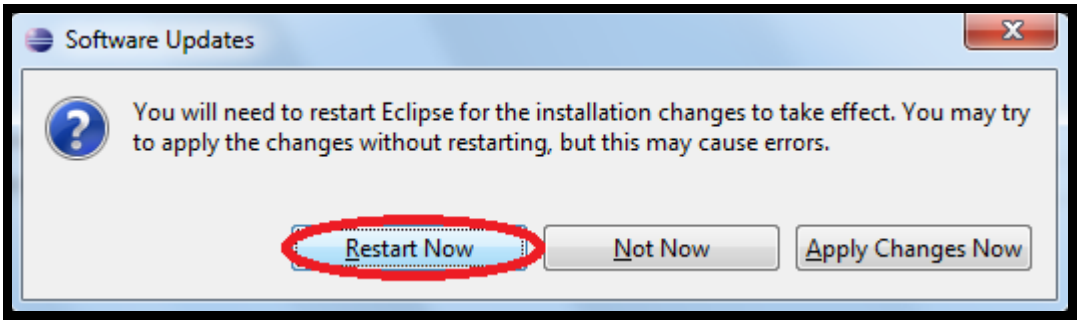
10. Esperamos un momento, volvemos a dar clic en “next” para la siguiente pantalla aceptamos los términos de licenciamiento, y damos clic en “Finish”, como se muestra en la siguiente figura.



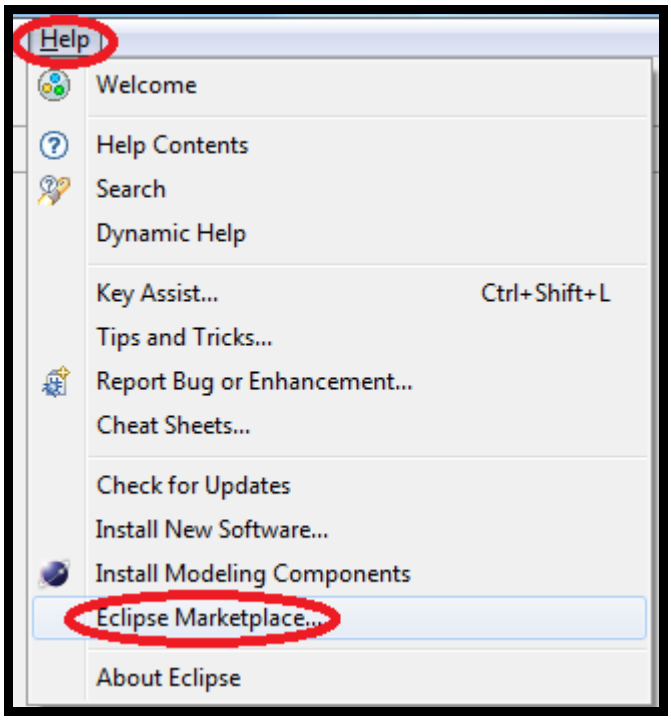
11. Aparecerá un cuadro de dialogo en el cual se puede visualizar el proceso de instalación, como se muestra a continuación.



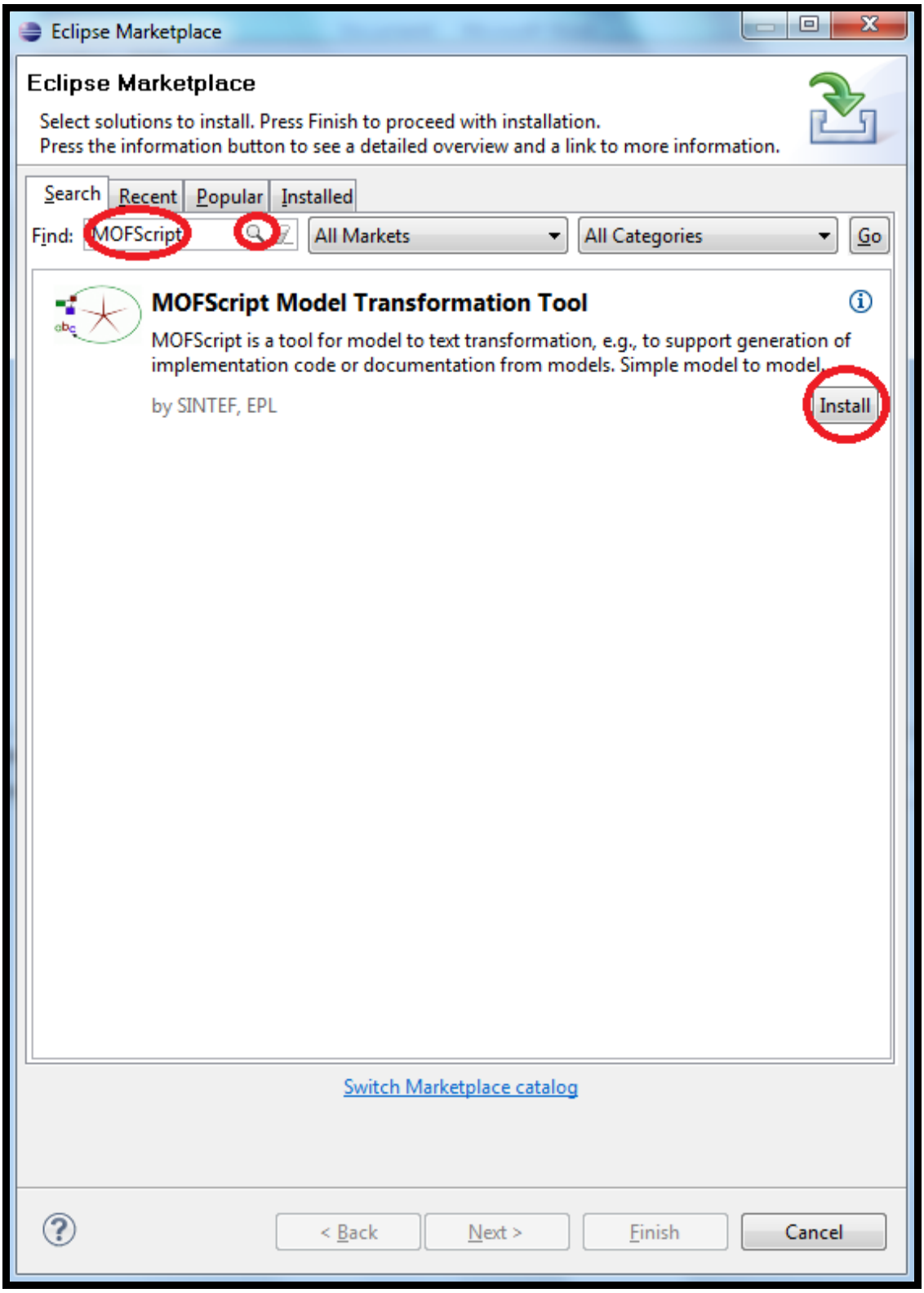
12. Finalmente aparece una cuadro de dialogo informado sobe las actualización realizadas y nos pedirá que reiniciamos nuestro eclipse, con lo cual en este cuadro seleccionaremos la opción" Restar Now", como se muestra a continuación.



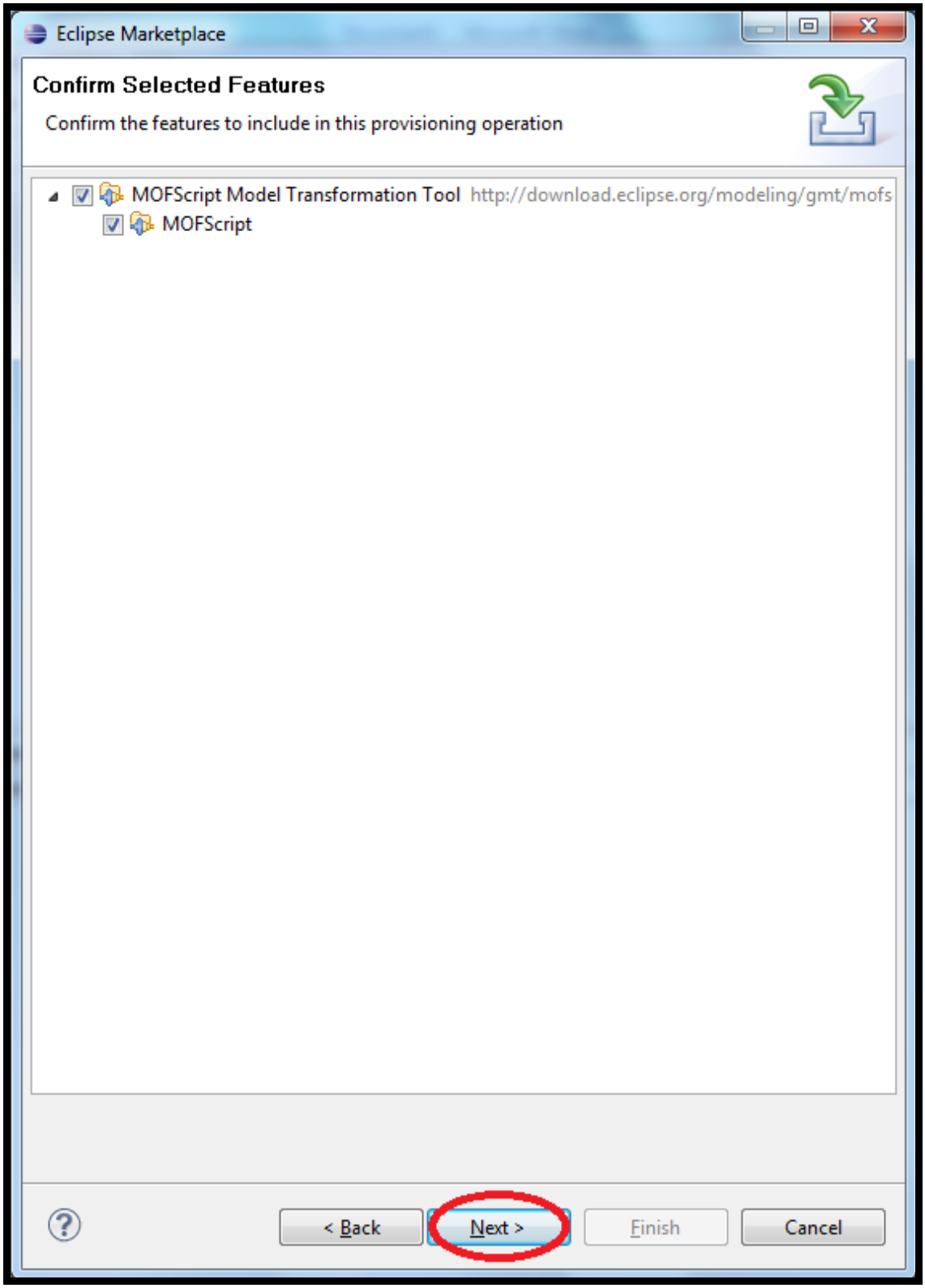
13. Nuestro eclipse se reiniciara y con esto ya tendremos el componente de GMF instalado, este componente se utilizará para el desarrollo de la sintaxis concreta.
14. El otro plugin que debemos instalar en eclipse es MOFScript para las transformaciones de modelo a texto, teniendo abierto nuestro eclipse, se hace clic sobre "Help" y se selecciona "Eclipse Marketplace…" como se muestra a continuación.



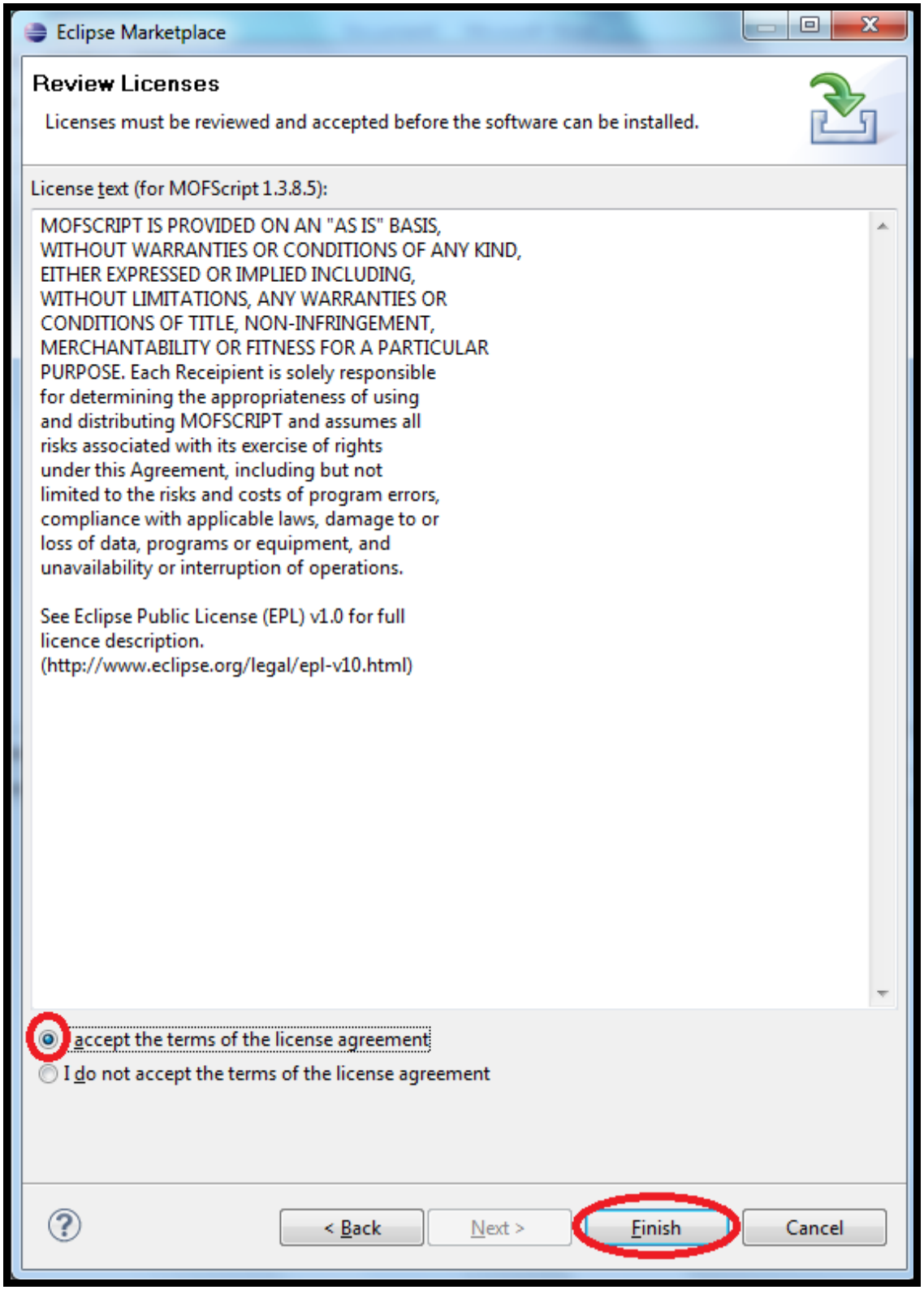
15. Esto abrirá una nueva venta con información del Marketplace para eclipse, en el campo “Find:” escribiremos MOFScrip y daremos clic en” find” o “Enter”, esto buscara el plugin de MOFScript, para instalar sencillamente damos clic sobre el botón “Install” que allí aparece, como se muestra en la siguiente figura.



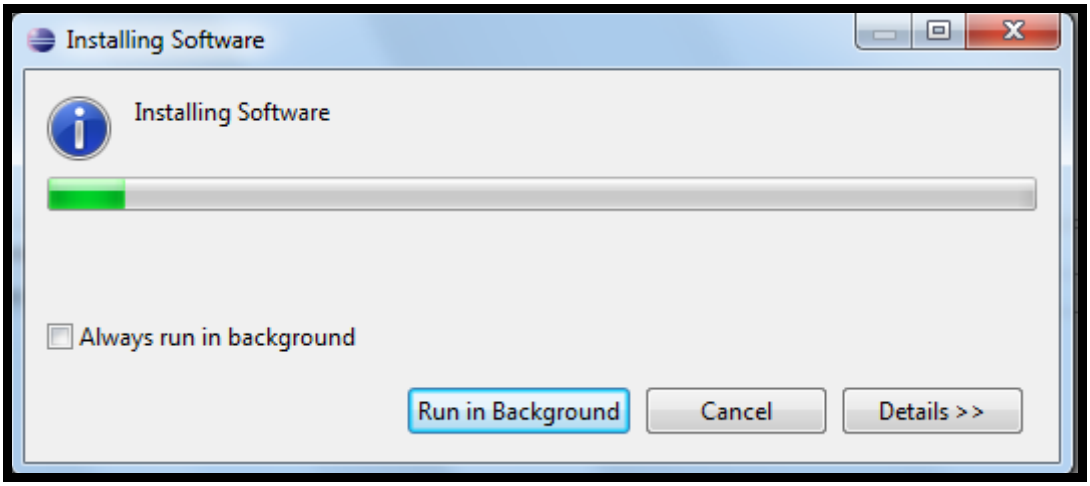
16. Esperamos un momento a que se descargue el plugin y en el nuevo cuadro de dialogo damos clic en “Next”, como se muestra a continuación.



17. En el nuevo cuadro de dialogo, se aceptan las condiciones de licenciamiento y se da clic en “Finish”, como se muestra a continuación.



18. Esto permitirá que el plugin sea instalado, aparecerá un cuadro de dialogo mostrando el progreso de la instalación como se ve a continuación, el proceso demorara un momento.



19. Si parece cualquier información en medio de la instalación acepte y continúe, finalmente aparecerá un cuadro de dialogo informado sobre la instalación del plugin y solicitará que reinicie su eclipse, de clic en “Restart Now” y al reiniciar el eclipse el plugin estará instalado.

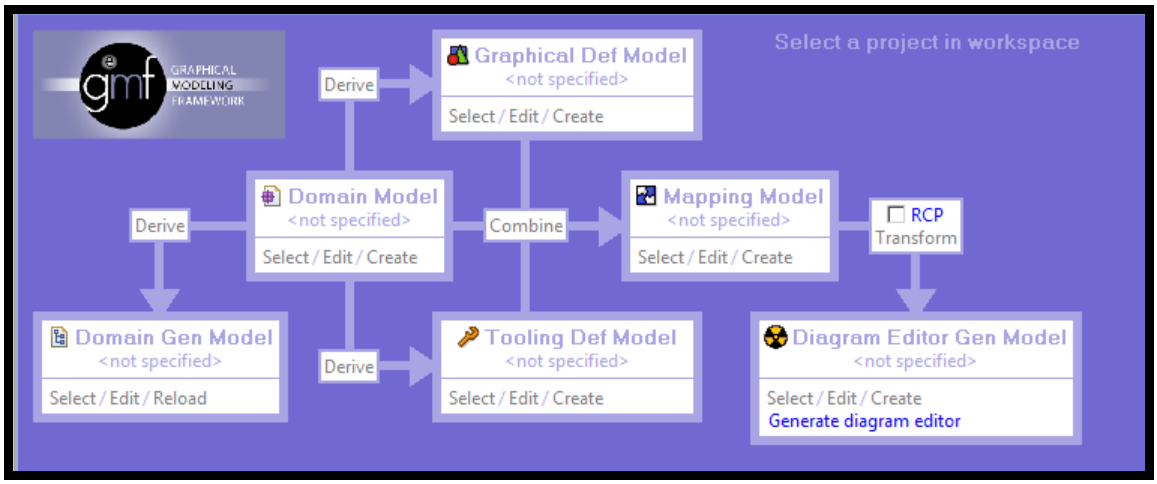
Los demás componentes que necesitamos para este tutorial como Eclipse Modeling Framework (EMF) y GEF ya venían preinstalados o se instalaron dentro de los plugins agregados.

2. Caso ejemplo para el desarrollo del tutorial

Para el desarrollo de este tutorial vamos a partir del siguiente caso hipotético; Una librería desea desarrollar un herramienta grafica que les ayude en el proceso de modelar sus stock de existencias, para ello han dicho que la librería tiene muchos libros con identificado ISBN y el nombre del libro, cada libro posee un único resumen y el libro puede ser impreso por al menos una editorial que posee un nombre. Bien pues la idea es modelar este sencillo caso y construir esa herramienta grafica que apoye el modelado.

3. Creación de una herramienta especifica de dominio en eclipse

Este tutorial tiene como objetivo enseñar el manejo de EMF y GMF para la creación de herramientas de modelado específicas de dominio. GMF establece un proceso muy específico para construir una herramienta de este tipo. Éste se compone de los siguientes subprocesos: definición del metamodelo o modelo a seguir, generación de código, definición de la metáfora gráfica, definición de las herramientas del modelo, especificación de la correspondencia entre los elementos del modelo y la metáfora gráfica, y generación de la herramienta, como se muestra en la siguiente figura.



Como se puede observar en la figura anterior, el centro del proyecto es el Domain Model, es decir el modelo del dominio o metamodelo que será el origen del proceso y del cual se derivarán el resto de subprocesos. El modelo del dominio se define utilizando EMF mediante un lenguaje de definición de modelos denominado Ecore. Todos y cada uno de los subprocesos de GMF están relacionados con el modelo y como indica su nombre, Ecore, constituirá el centro del proyecto en todo momento. A continuación se explican cada uno de los subprocesos que conforman el desarrollo del proyecto.

3.1 Definición del modelo

GMF precisa de un metamodelo y para ello se sirve de EMF, que se trata de un marco de modelado que soporta y genera documentos XMI con un esquema XML Ecore con la especificación del dominio. Es por ello, que este tutorial no sólo es de GMF, sino que también introduce los conocimientos básicos de EMF.

3.2 Definición de la metáfora gráfica

Una vez definido el dominio del modelo como lo muestra la figura anterior, se puede empezar a diseñar la definición gráfica de las primitivas de modelado. La definición gráfica consiste en decidir qué primitivas de modelado harán la función de nodos (elementos de la herramienta de modelado que se desea construir), cuáles serán conectores (enlaces entre los elementos de la herramienta de modelado) y cuáles etiquetas (propiedades de los elementos y enlaces de la herramienta de modelado).

3.4 Definición de herramientas

La creación y especificación del panel de herramientas se ha de realizar después de la definición de la metáfora gráfica como o muestra la figura anterior. Consideramos panel de herramientas como la paleta de la herramienta de modelado a crear que proporcionará un funcionalidad “drag and drop”, mostrándonos las primitivas de modelado en la forma que se

han diseñado en la definición gráfica. En este paso también se definen los iconos del la herramienta de modelado, que constituyen la iconografía de la paleta.

3.5 Correspondencia entre elementos del modelo y gráficos

La correspondencia entre los elementos del modelo, la metáfora gráfica y herramientas se ha de realizar después de hacer la especificación de las herramientas como lo muestra la figura anterior. En este escenario todo lo creado anteriormente cobra un sentido, ya que, que se relacionan y asocian todos y cada uno de los elementos creados con anterioridad. Este paso es el último del proceso de creación que se ve en la figura anterior. La creación del generador de la herramienta sin errores indica que todas las anteriores especificaciones se han realizado correctamente y mantienen coherencia.

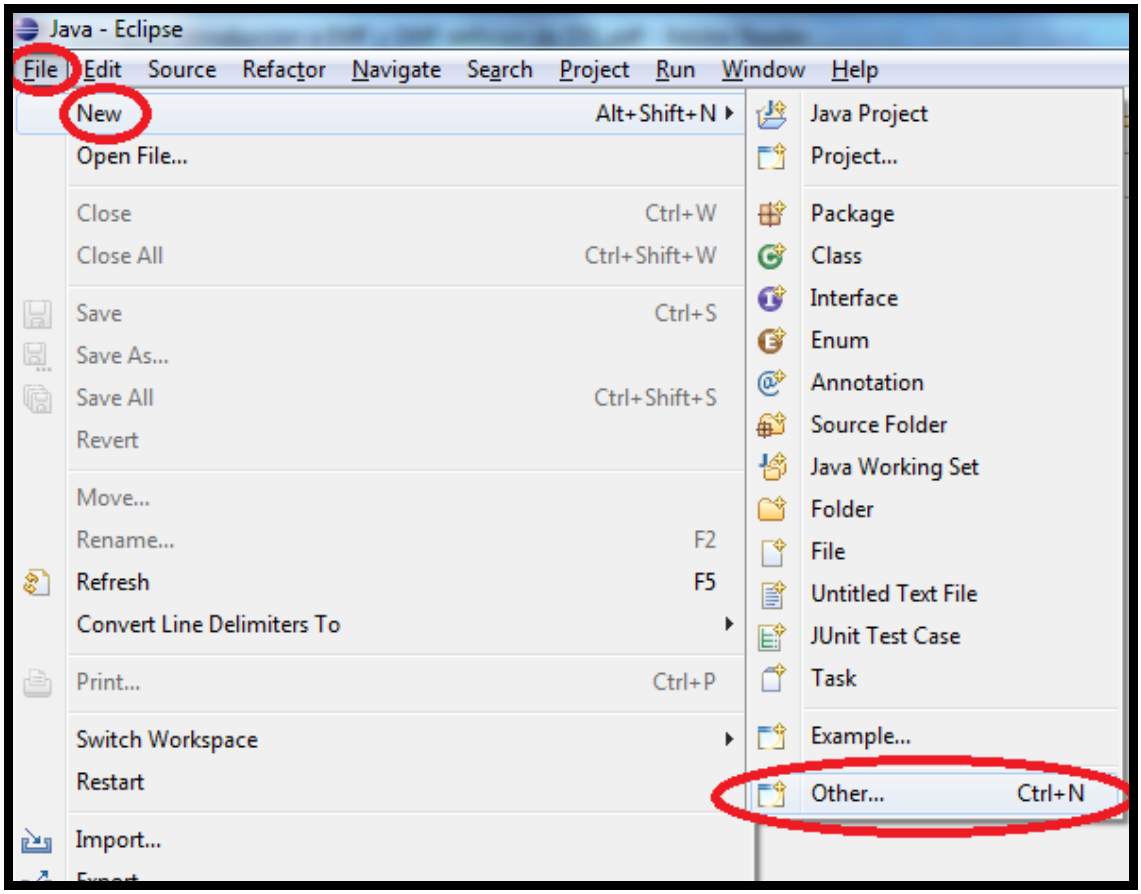
4. EMF (Eclipse Modeling Framework)

El Eclipse Modeling Framework (EMF) es una herramienta que proporciona una estructura de modelado y facilidades para la generación de código al objeto de construir herramientas u otras aplicaciones basadas en un modelo de datos estructurado. A partir de una especificación XML de un modelo, EMF suministra herramientas y soporte de ejecución para producir un conjunto de clases Java en base a ese modelo, un conjunto de clases Adapter, que permiten su visualización y edición basándose en comandos del modelo, y un editor básico.

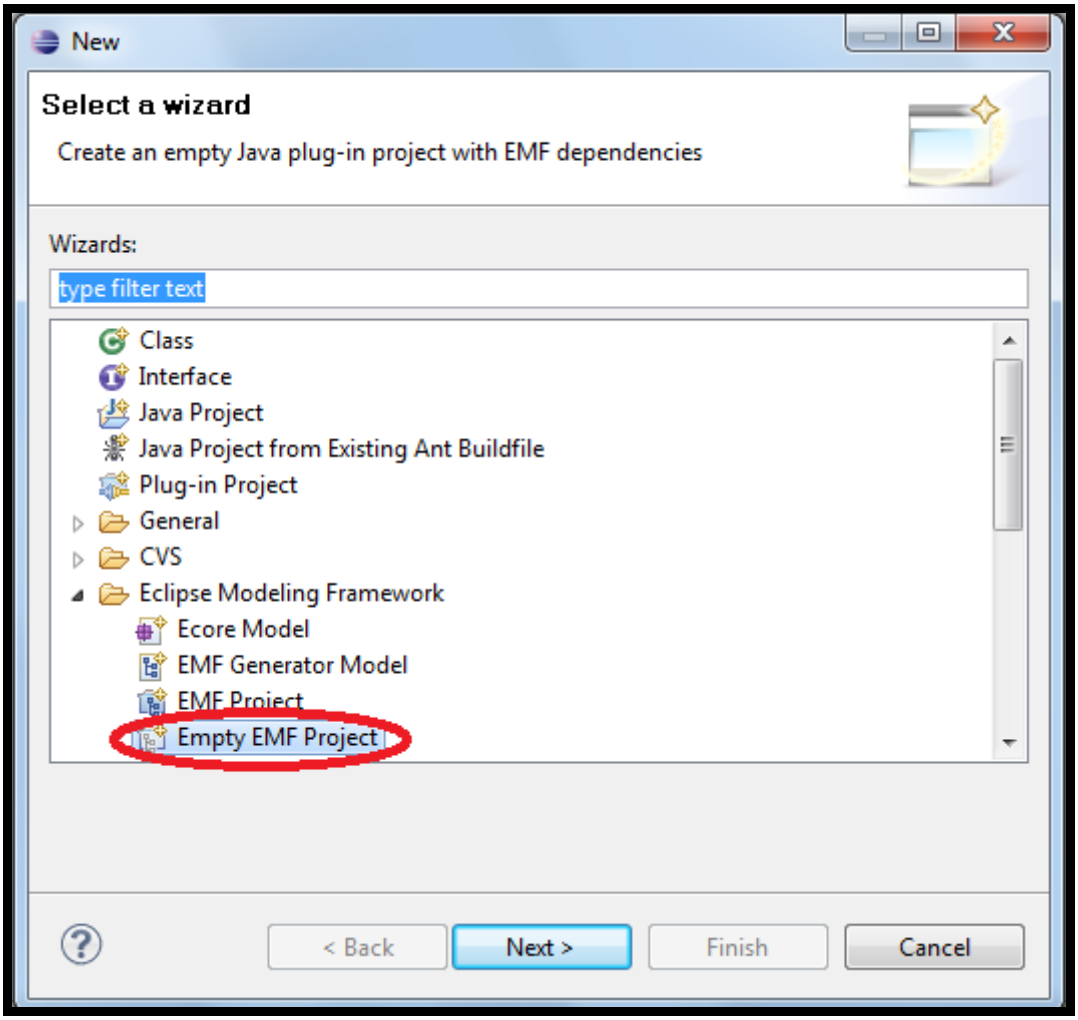
EMF permite importar modelos que han sido previamente especificados usando documentos Ecore, de los que se hablará más adelante. Una de las características más importantes de EMF es que suministra mecanismos de interoperabilidad con otras herramientas y aplicaciones basadas en EMF.

4.1 Creación de un modelo EMF

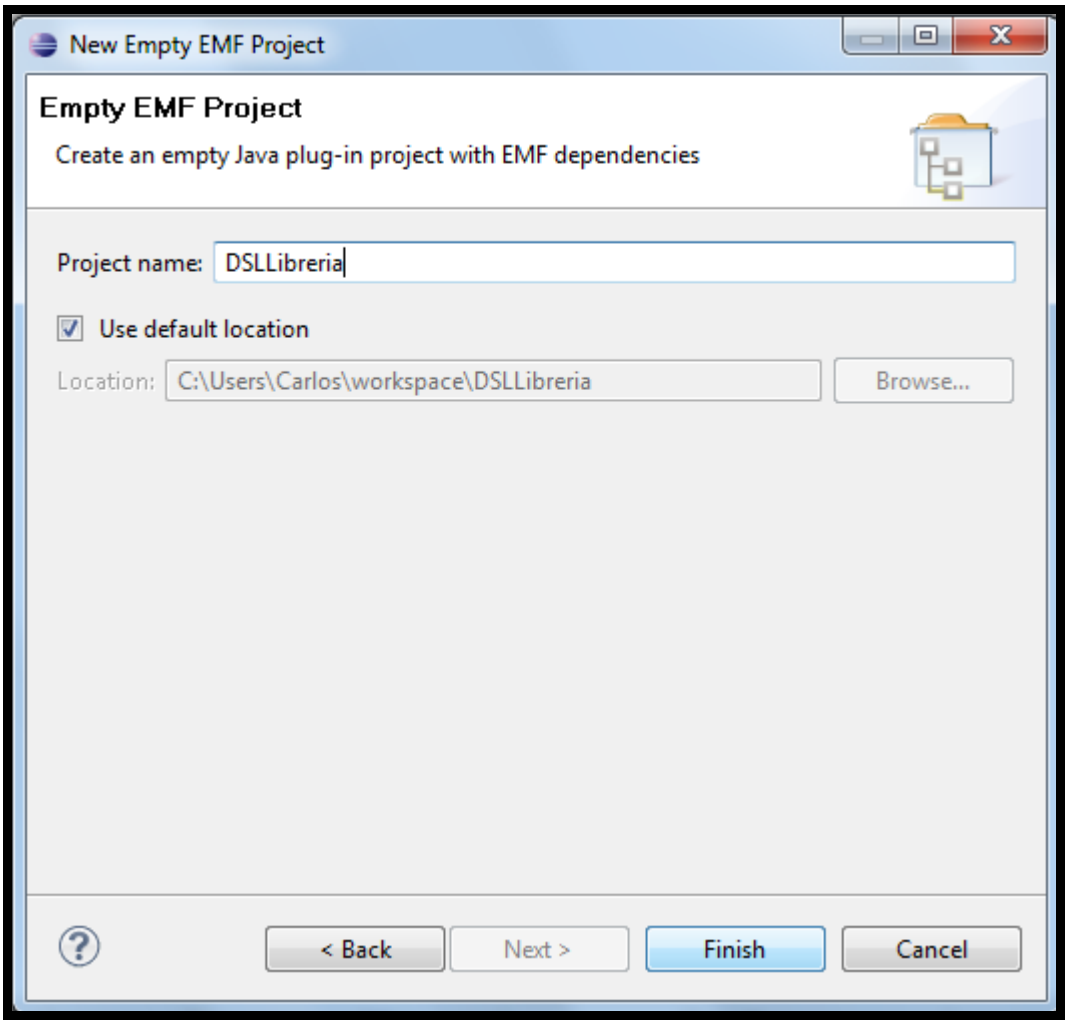
Para trabajar con Eclipse siempre es necesario crear un proyecto con el que trabajar, el caso de EMF no es una excepción. Por lo tanto, se ha de crear un proyecto mediante el menú de Eclipse “File ->New->Project “, como se muestra a continuación.



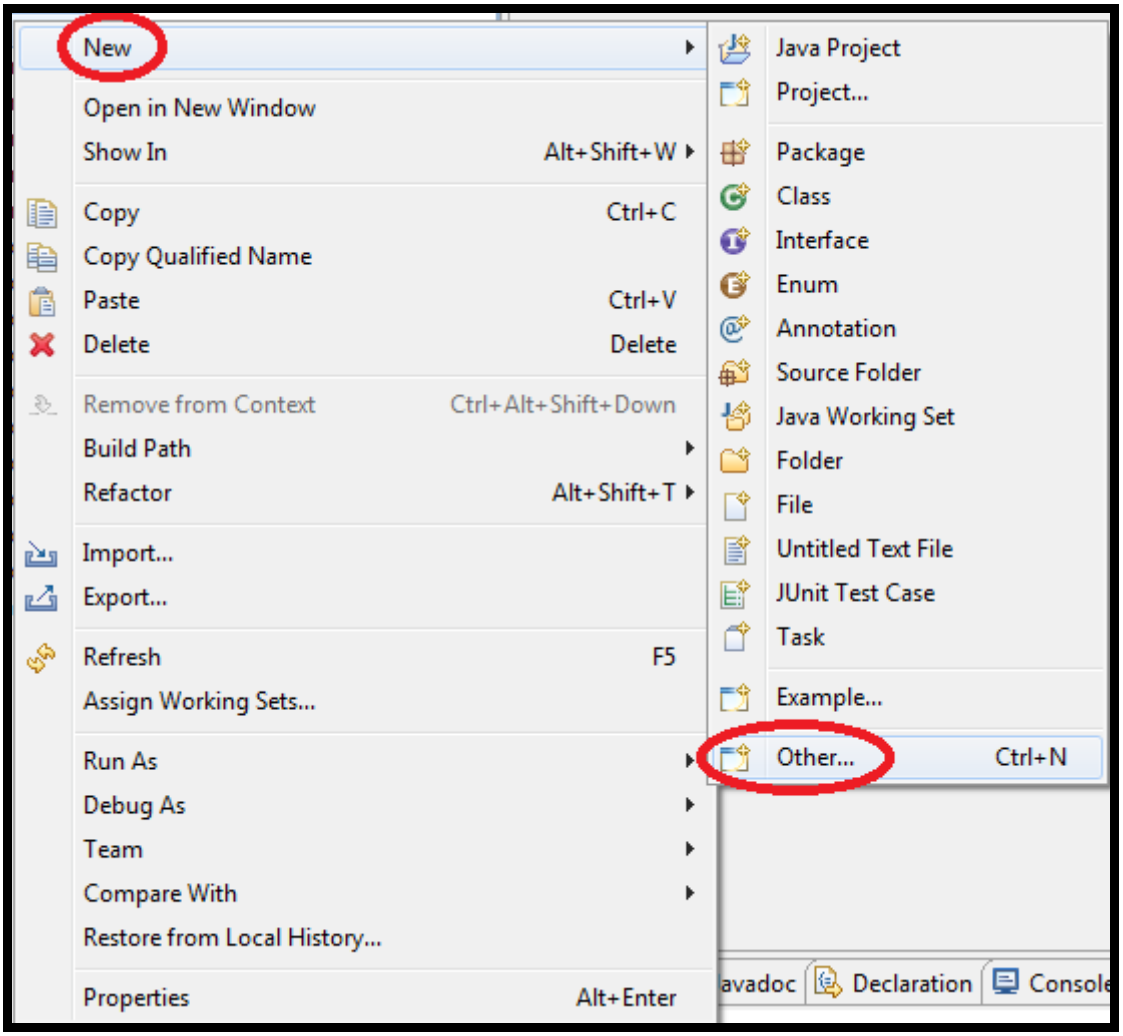
A continuación, se ha de elegir el tipo de proyecto que se desea crear. En nuestro caso se ha de elegir un “Empty EMF Project”, para crear el nuevo proyecto vea la siguiente figura.



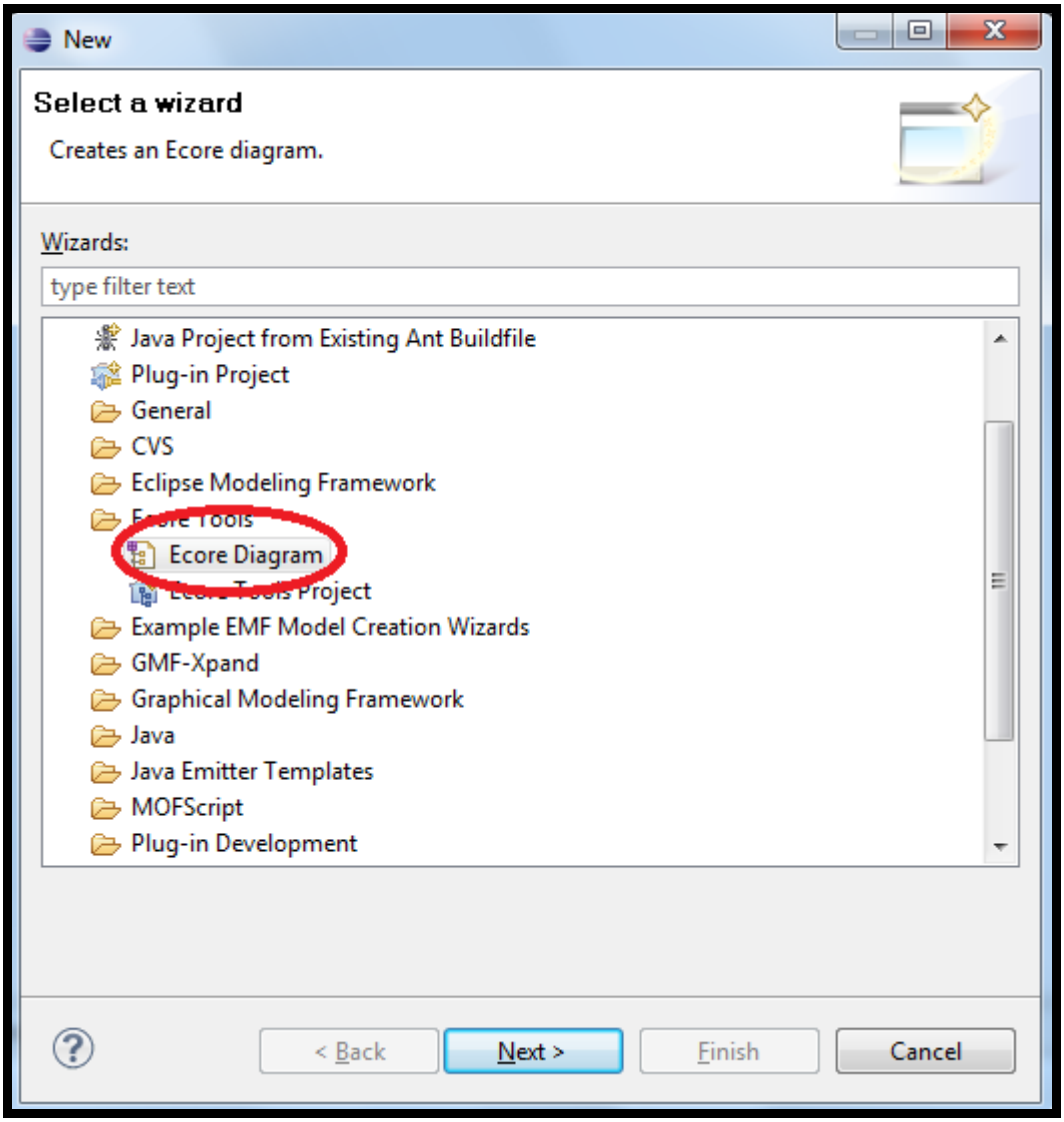
Posteriormente, se pulsa “Next” y se le da nombre al proyecto, para terminar la definición del proyecto hacer clic en “Finish”, como se muestra a continuación.



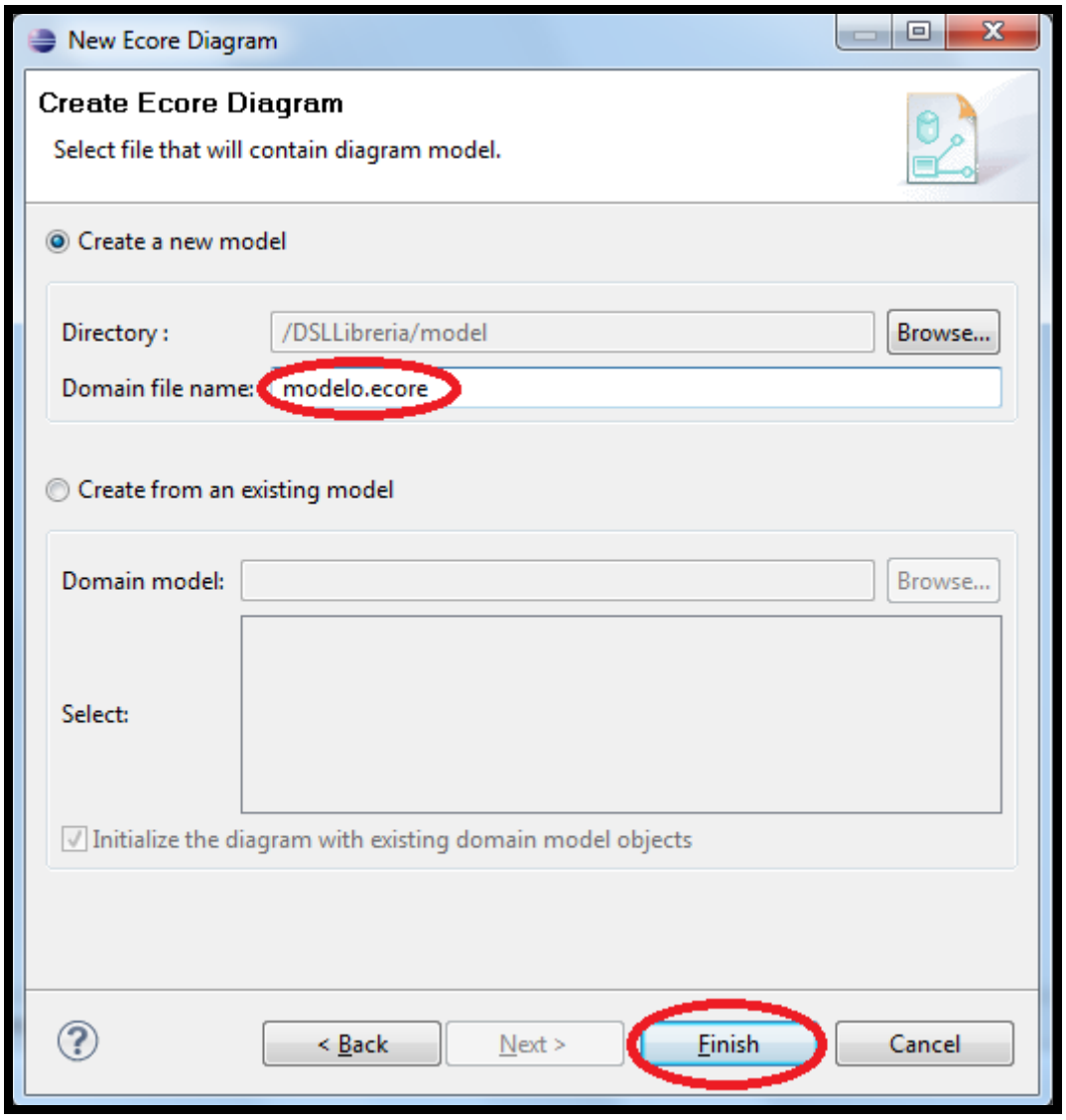
EMF nos permite crear modelos basados en Ecore. Pulsando con el botón derecho sobre la carpeta “model” del proyecto creado previamente en “New -> Other” como se muestra en la siguiente figura, podemos crear un nuevo modelo seleccionando en la carpeta “Other” el tipo de documento denominado “Ecore Diagram”. Este tipo de documento es el que permite especificar modelos Ecore, y para el caso de estudio definir el diagrama Ecore relativo a la librería. Además, al crear un diagrama Ecore, EMF genera al mismo tiempo un documento XMI con extensión .ecore, que asocia al diagrama del modelo, y que es utilizado para la generación de código.



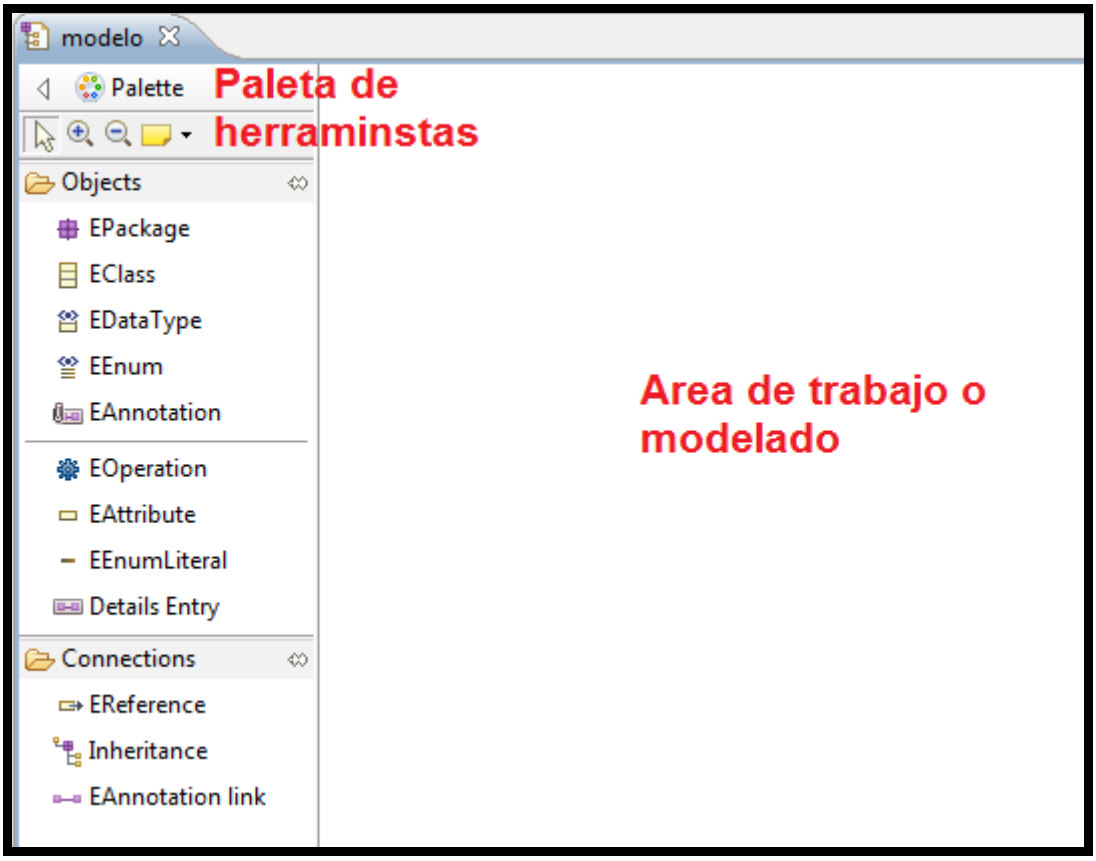
Una vez seleccionado el tipo de elemento en “Ecore Tools -> Ecore Diagram”, pulsamos “Next” para posteriormente darle nombre, ver la siguiente figura.



El nombrado consiste dar un nombre al documento ".ecore" este mismo nombre se asignara al fichero ".ecore_diagram", por ejemplo "modelo" y pulsar "Finish". Estos pasos se ven en la siguiente figura.



Se crean los dos documentos y se abre automáticamente el documento con extensión ".ecore_diagram". Este documento presenta un área de trabajo o modelado y una paleta de herramientas como se muestra en la siguiente figura.



La paleta de herramientas incluye los diferentes tipos de elementos y relaciones que permiten definir un modelo Ecore. En la paleta se puede distinguir la herramienta “EClass”, con la que podremos crear nuevas clases, la herramienta “EPackage”, con la que crearemos nuevos paquetes, “EAnnotation”, con la que se crearán anotaciones y comentarios, “EDataType” con la que crear nuevos tipos de datos, “EAttribute” con la que agregar atributos a las clases, “EOperation” con la que añadir operaciones a las clases y por último, las referencias “EReference” , herencia “Inheritance” y vínculo de anotaciones “EAnnotation link” para vincular las anotaciones . El modelado se realiza mediante “drag and drop” para cualquiera de las herramientas de la paleta, esto quiere decir que si pulsamos por ejemplo sobre la herramienta de creación de clases y acto seguido pulsamos sobre el área de trabajo se nos crea una nueva clase. El nombre de las clases en Eclipse debe comenzar siempre por letra mayúscula. Por otro lado, es posible añadirles tantos atributos y operaciones como se deseen. Hay dos posibilidades de hacerlo: una de ellas consiste en arrastrar desde la paleta de modelado y soltar sobre la clase los atributos y operaciones, y la otra es a través de un menú emergente que aparece superponiendo el puntero sobre la clase en cuestión.

Las relaciones entre clases se instancian gráficamente mediante las relaciones de “EReference” y “Inheritance”. Para poder relacionar dos clases se debe seleccionar la herramienta y acto seguido seleccionar la clase de la que parte la relación (en el caso de herencia, la clase hija, y en el caso de la referenciación, la clase compuesta) y después arrastrar y seleccionar la clase

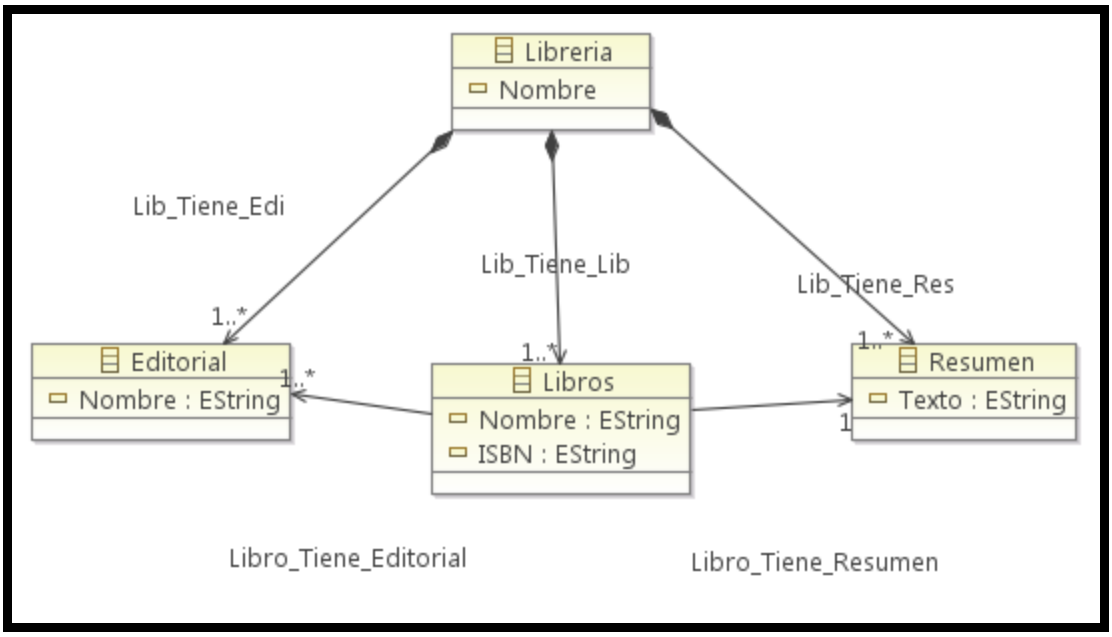
destino de la relación (en el caso de herencia, la clase padre, y en el caso de la referenciación, la clase componente). Se ha de tener en cuenta, que las asociaciones en EMF son siempre unidireccionales. Si se desea modelar una relación bidireccional, se realiza mediante dos asociaciones que relacionen las dos mismas clases en sentidos opuestos y fusionarlas en una sola bidireccional. Para ello, se ha de seleccionar dentro de una de las asociaciones, su opuesta en la propiedad Eopposite.

Para definir el tipo de un atributo, clase o relación es necesario editar sus propiedades. Esto se realiza pulsando con el botón derecho encima del elemento del que se desean ver las propiedades y seleccionar la opción “Show Properties View”. A continuación, aparece una ventana en la parte inferior del entorno de Eclipse mediante la que se pueden editar todas las propiedades de cada elemento. Entre las propiedades que definen a una clase están el nombre, si pertenecen a alguna subclase, si son de algún tipo predefinido, etc. Para editar el tipo de los atributos, se ha de pinchar en la propiedad “EType” y seleccionar el tipo de datos adecuado para cada atributo, de entre todos los que se muestran en la lista, “EString” para las cadenas de caracteres, “EInt” para los números enteros, EFloat para los números reales, “Echar” para los caracteres y “EBoolean” para los booleanos, entre otros. Todos los atributos de las clases del caso de estudio son de tipo “EString”.

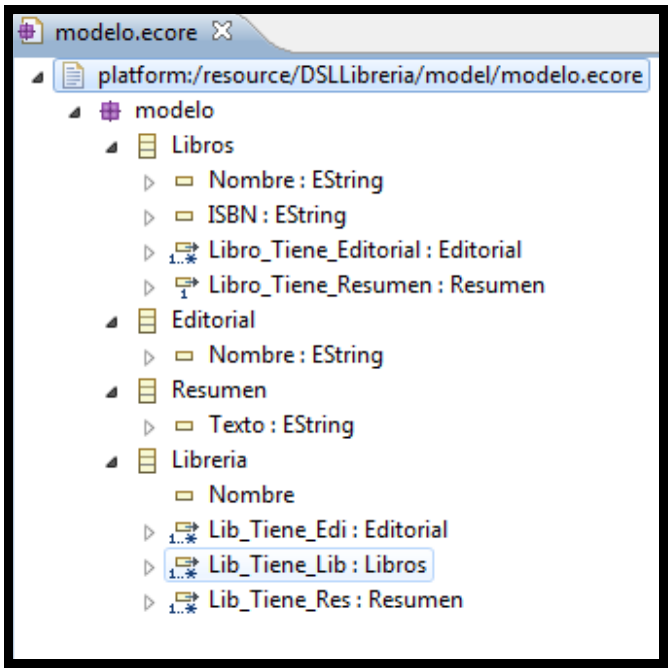
También, podemos crear enumeraciones de elementos, es decir, listas de tipos de datos. Esta definición se realiza sobre el área de trabajo en el que definimos nuestro modelo Ecore. Para ello añadimos un objeto de tipo “EEnum” de la barra de herramientas de modelado al área de trabajo mediante “drag and drop”. Posteriormente, le damos un nombre en el campo “Name”. Este será el nombre que tendrá nuestro tipo de datos enumerado. Para crear la lista sólo nos queda definir dentro del elemento “EEnum” tantos “EEnumLiteral” como número de tipos de datos queramos introducir. “EEnumLiteral” se encuentra también en la barra de herramientas de modelado. A cada “EEnumLiteral” se le ha de asociar su nombre, que será el nombre con el que aparecerá en la enumeración. Para ello, introducimos el mismo nombre para el campo “Name” y Literal del “EEnumLiteral”.

Dado que EMF expresa en formato XMI el modelo correspondiente a un diagrama ecore, conviene resaltar que para definir un modelo correctamente en EMF es necesario incluir un elemento raíz (una clase o EClass) que relacione mediante agregación todas las clases del modelo, en nuestro caso de estudio la EClass Libreria agrega las clases Libros, Resumen y Editorial mediante tres relaciones de agregación. A su vez la clase Libros, está compuesta por la clase Resumen y Editorial, teniendo en cuenta que un libro tiene un resumen y una editorial. Por ello, al crear una relación se le asigna un nombre y se especifica su cardinalidad. Para esto último, existen dos propiedades dentro de la vista de propiedades, “Lower Bound:” en la cual se especifica la cardinalidad mínima, y “Upper bound:”, en la que se especifica la cardinalidad máxima. Cabe destacar que para conseguir la cardinalidad N, se debe poner el valor “-1” y automáticamente, al dejar de editar la relación, se cambia el valor a “*”.

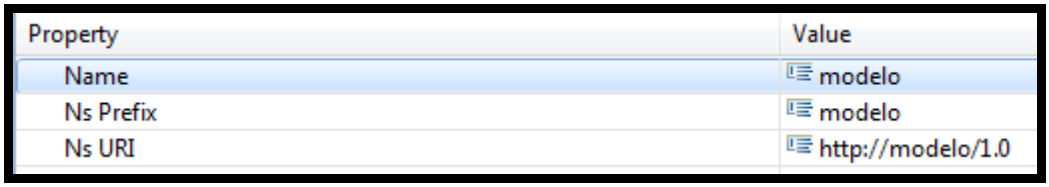
Una vez conocidas las herramientas de modelado, es posible proceder a modelar el caso de estudio, que se muestra a continuación.



Por último, hay que dar un identificador al paquete del “modelo.ecore”. Para ello cerramos el diagrama y hacemos doble clic sobre “modelo.ecore” en el árbol de documentos de proyecto situado en la ventana izquierda “Package Explorer”. Si desplegamos totalmente el “modelo.ecore” podemos ver todas nuestras clases, con sus atributos y relaciones, estructuradas en un árbol como se muestra a continuación.



Si nos fijamos, hay un icono con forma de paquete morado en la raíz de árbol con el nombre null. Si se hace click con el botón derecho sobre ese paquete y se selecciona Show Properties View. Aparece la ventana con sus tres propiedades Name, Ns Prefix y Ns URI que se han de rellenar con el nombre del modelo como se muestra a continuación.



Finalmente, guardamos el proyecto y ya tenemos nuestro modelo EMF creado.

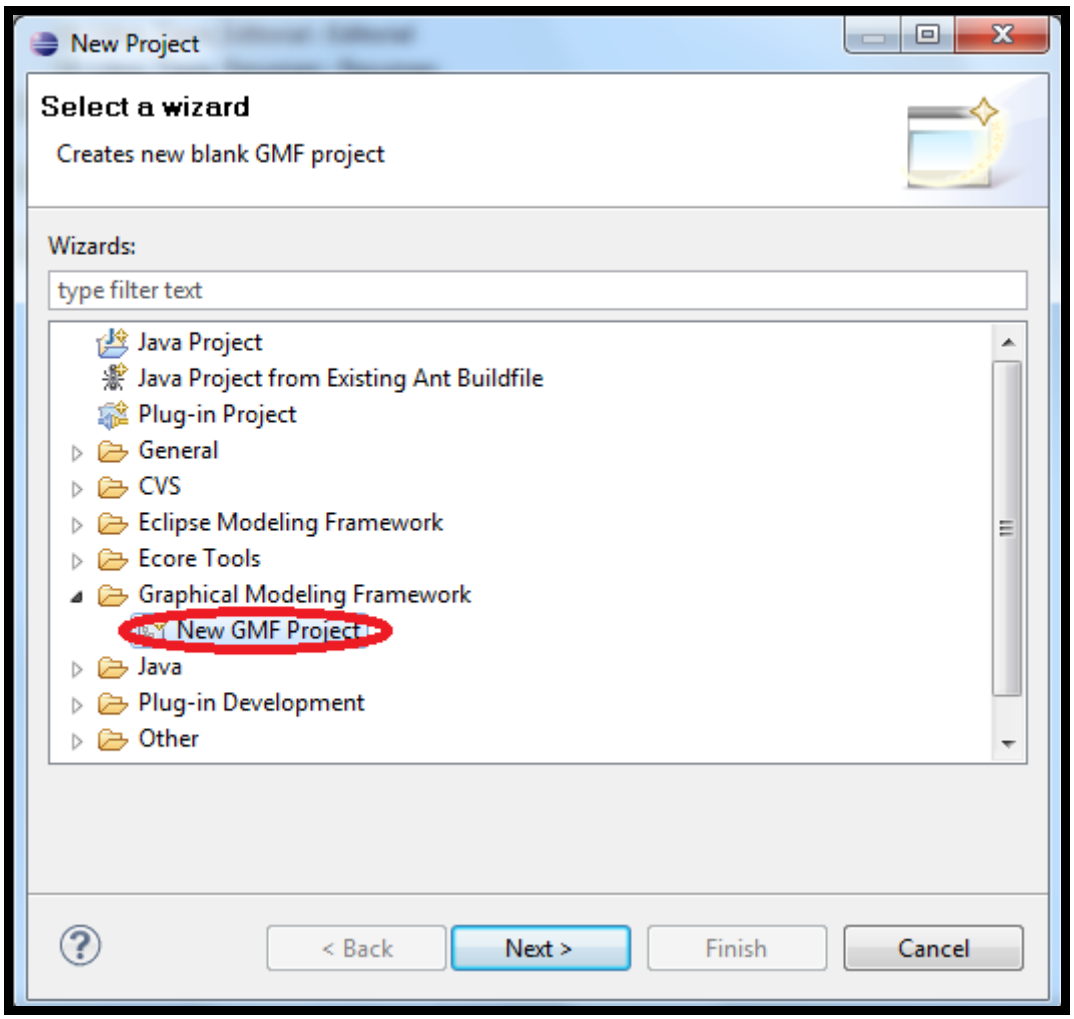
5. GMF (Graphical Modeling Framework)

Graphical Modeling Framework (GMF) proporciona una herramienta para la generación de editores gráficos basados en EMF y GEF. Este último es el que permite al desarrollador crear un editor gráfico completo de forma rápida a partir de un modelo de una aplicación. Una característica destacable en GMF es la reutilización de la definición gráfica para diferentes dominios y aplicaciones, esto es, se pueden reutilizar las metáforas gráficas ya definidas para conceptos de diferentes dominios y aplicaciones. Esta característica se consigue modelando por separado las componentes gráficas que se corresponden con cada uno de los elementos del dominio y la definición de la paleta de herramientas, la cual tendrá una herramienta por cada primitiva. Para completar el proceso de generación de un editor gráfico de dominio, GMF

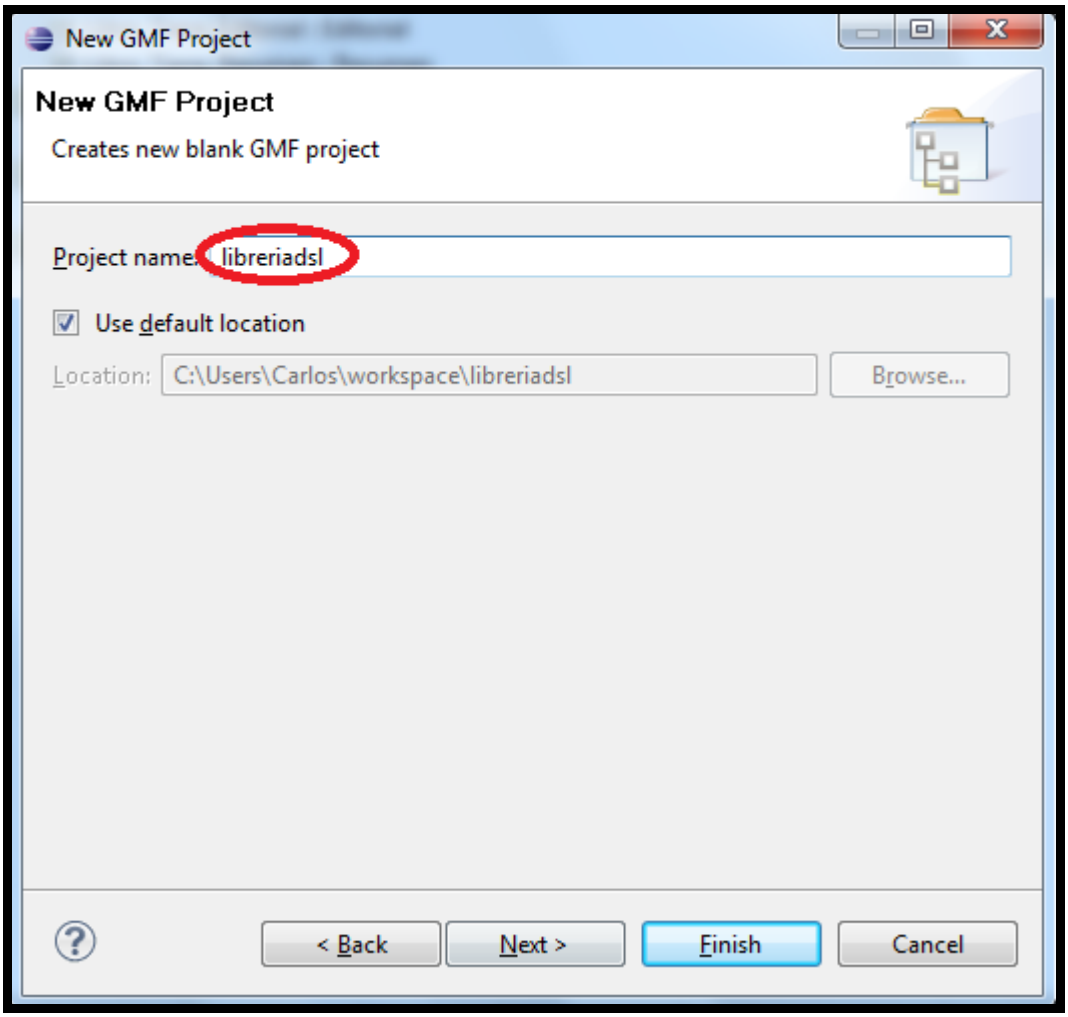
proporciona una definición de mapping o correspondencia mediante la que se asocia cada primitiva de modelado con su componente gráfica y con su herramienta dentro del editor que se está generando. El proceso de construcción de una herramienta de modelado con GMF se ha introducido en secciones anteriores, ahora se procederá a aplicar dicho proceso mediante la elaboración de este tutorial.

5.1 Creación de un proyecto GMF

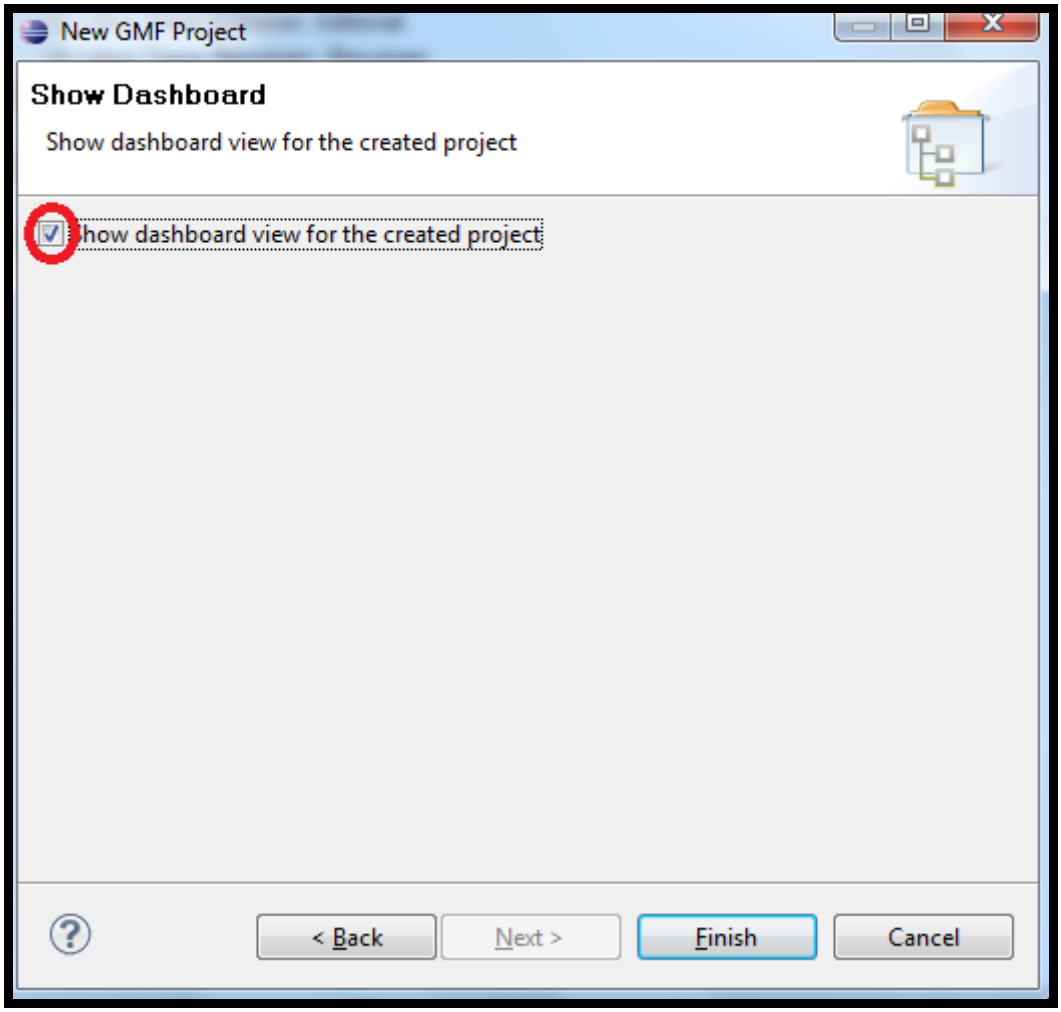
Para crear un nuevo proyecto en eclipse vamos a "File -> New -> Project" y en la carpeta "Graphical Modeling Framework" seleccionamos "New GMF Project", como se muestra a continuación.



Le damos un nombre al proyecto, en nuestro caso “libreríadsl” y pulsamos “Next”. Es importante que el proyecto no se llame como ninguna de las clases ya que podría dar errores de compilación al generar el código, como se muestra a continuación.



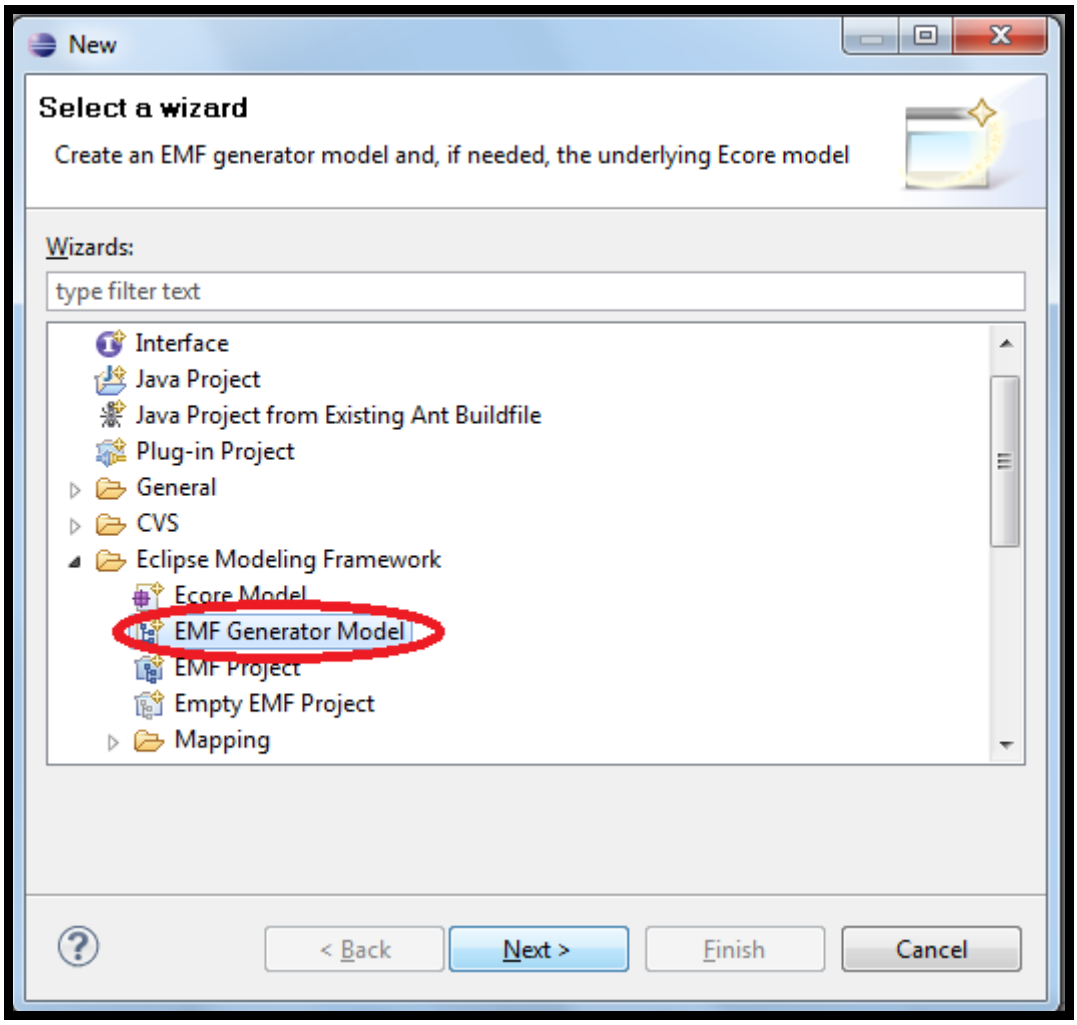
Activamos la casilla “Show dashboard view for the created Project” para obtener una vista del estado de nuestro proyecto y pulsamos “Finish”. En caso de que se cierre esta vista, estará accesible desde “Window ->Show view -> Other -> General -> GMF dashboard”. Esto se muestra a continuación.



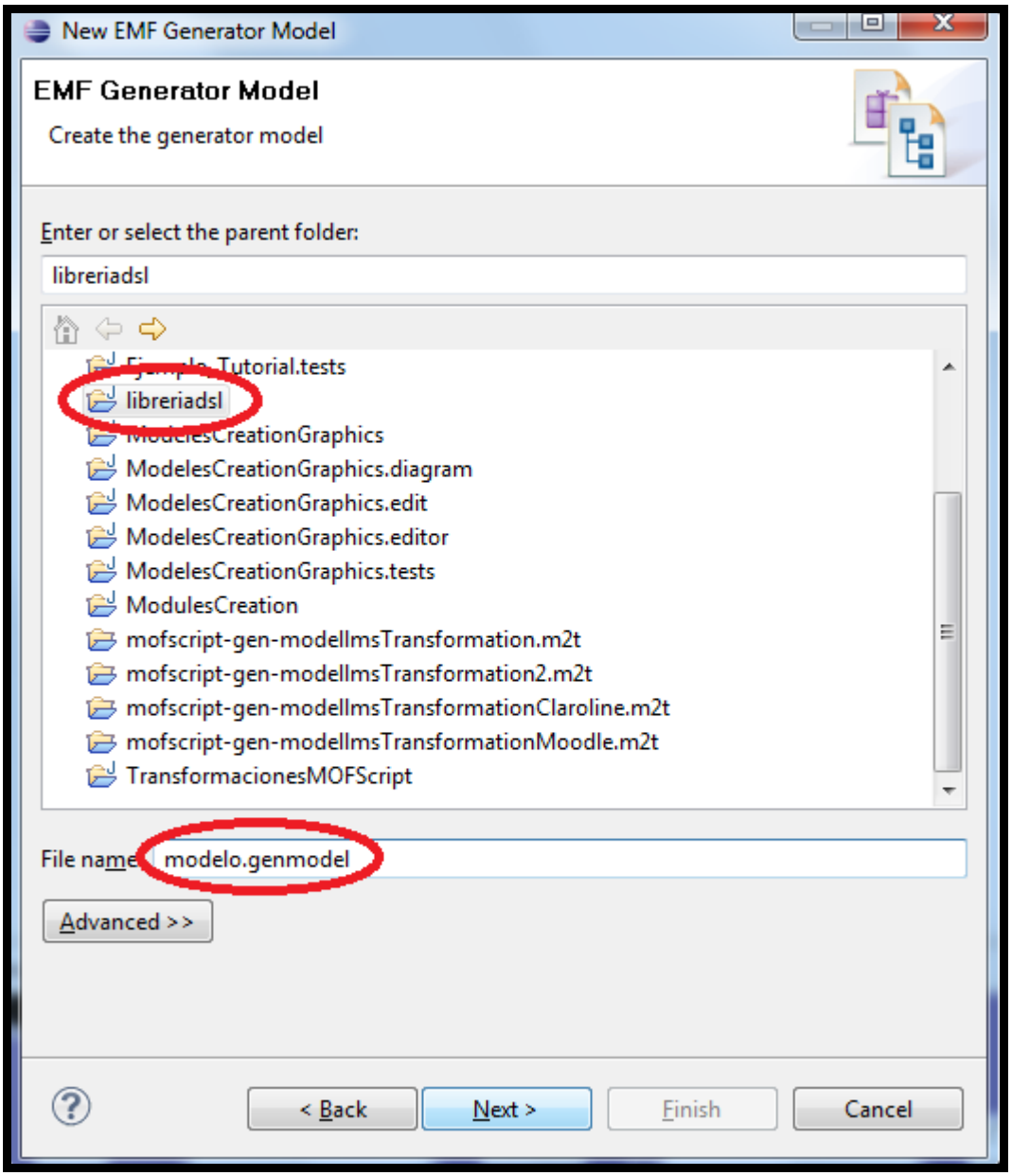
En este proyecto será donde se cree una herramienta de modelado específico para el caso de estudio utilizado en este tutorial.

5.2 Creación o importación de un modelo

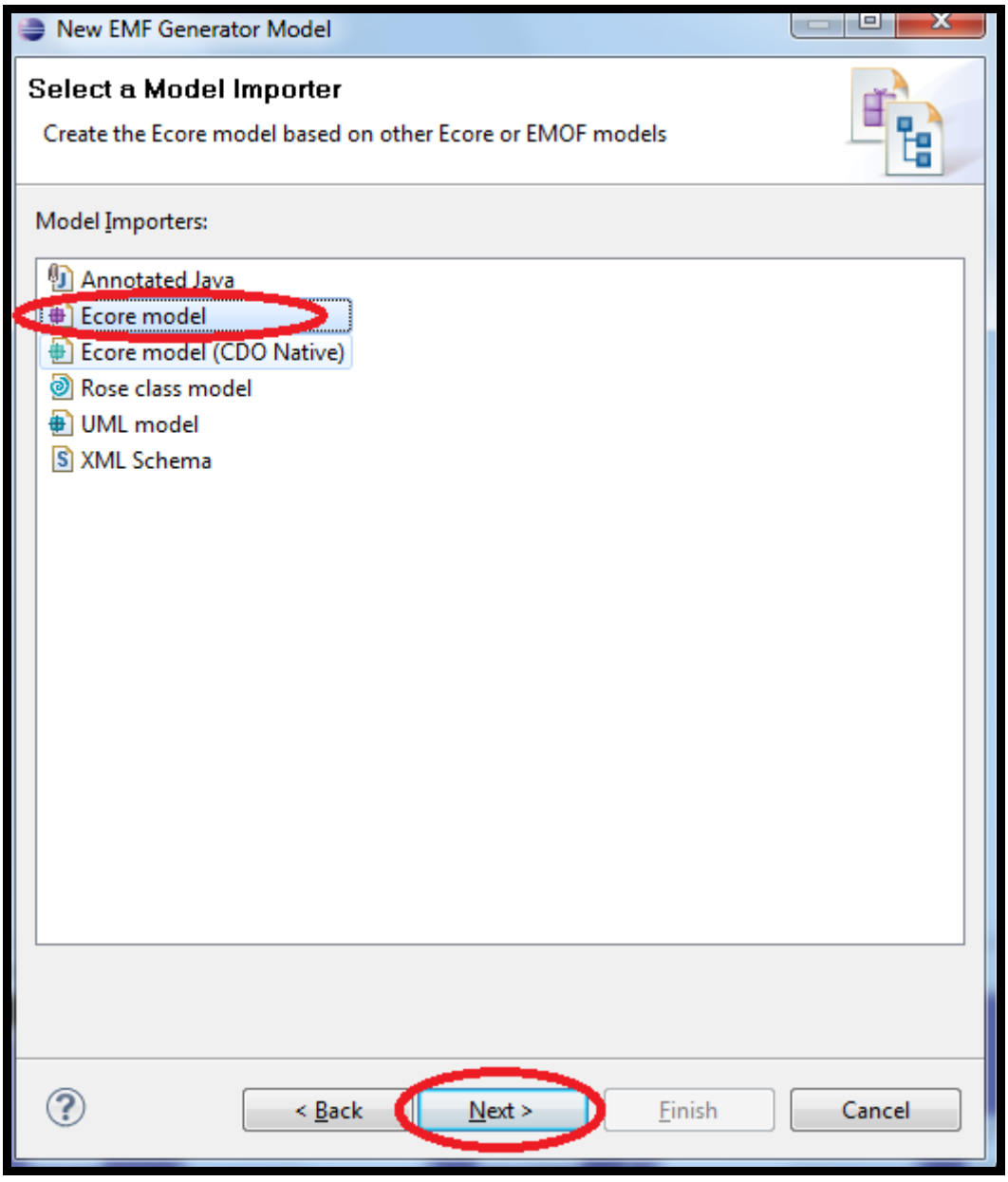
El objetivo principal de GMF es proporcionar un editor gráfico para modelar diferentes entornos partiendo de un modelo o metamodelo de entrada. Por esta razón, el centro del proceso de desarrollo de un entorno de modelado en GMF es un modelo. Dicho modelo es posible crearlo dentro de GMF desde cero o bien importar un modelo realizado previamente, como es nuestro caso. Para ello, Clicamos con el botón derecho sobre el proyecto “GMF New -> Other”, en la carpeta “Eclipse Modeling Framework” seleccionamos “EMF Generator Model”, como se muestra a continuación.



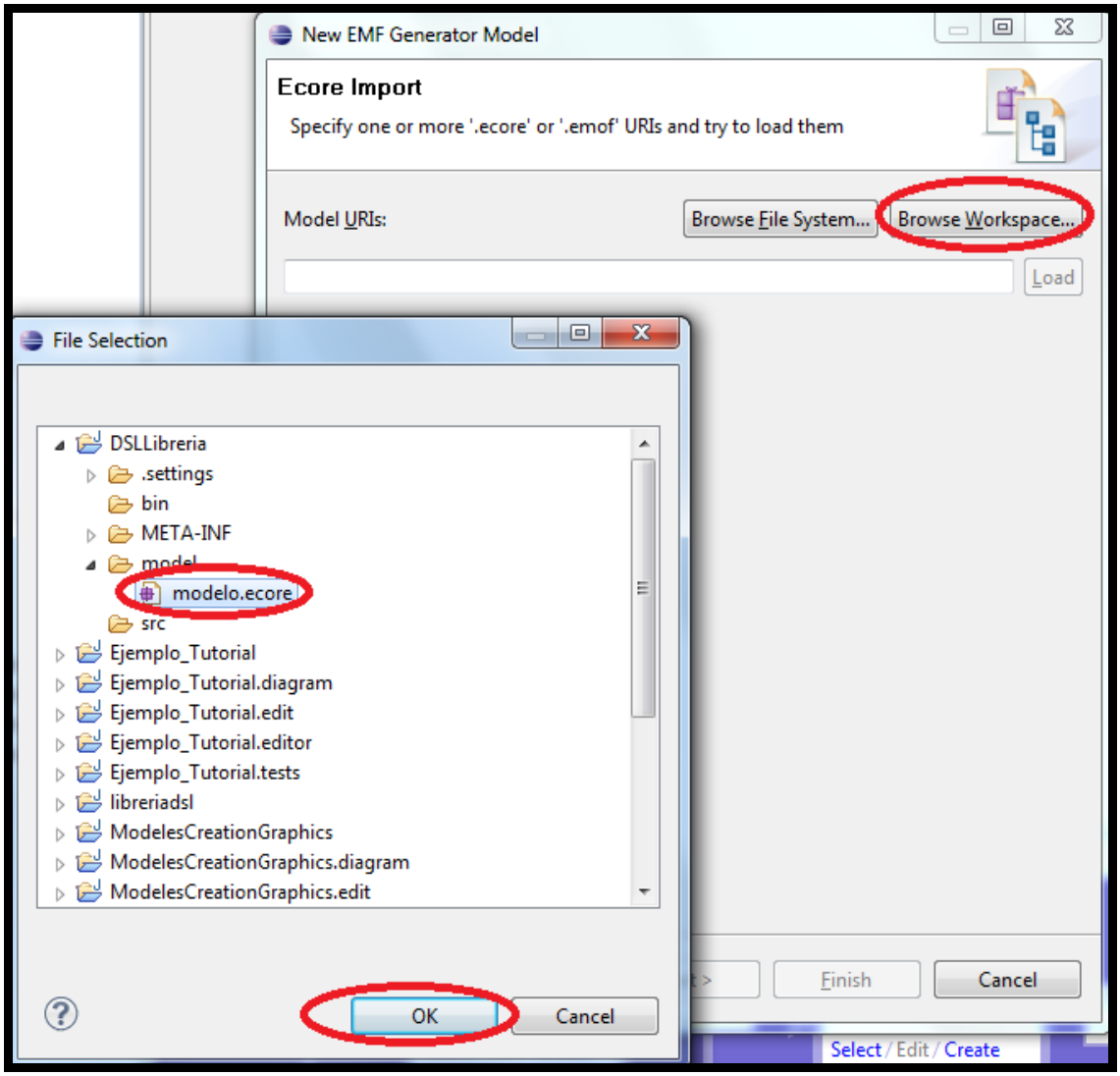
Seleccionamos "libreriadsl" como directorio raíz, le ponemos como nombre "modelo.genmodel" y pulsamos "Next", como se muestra a continuación.



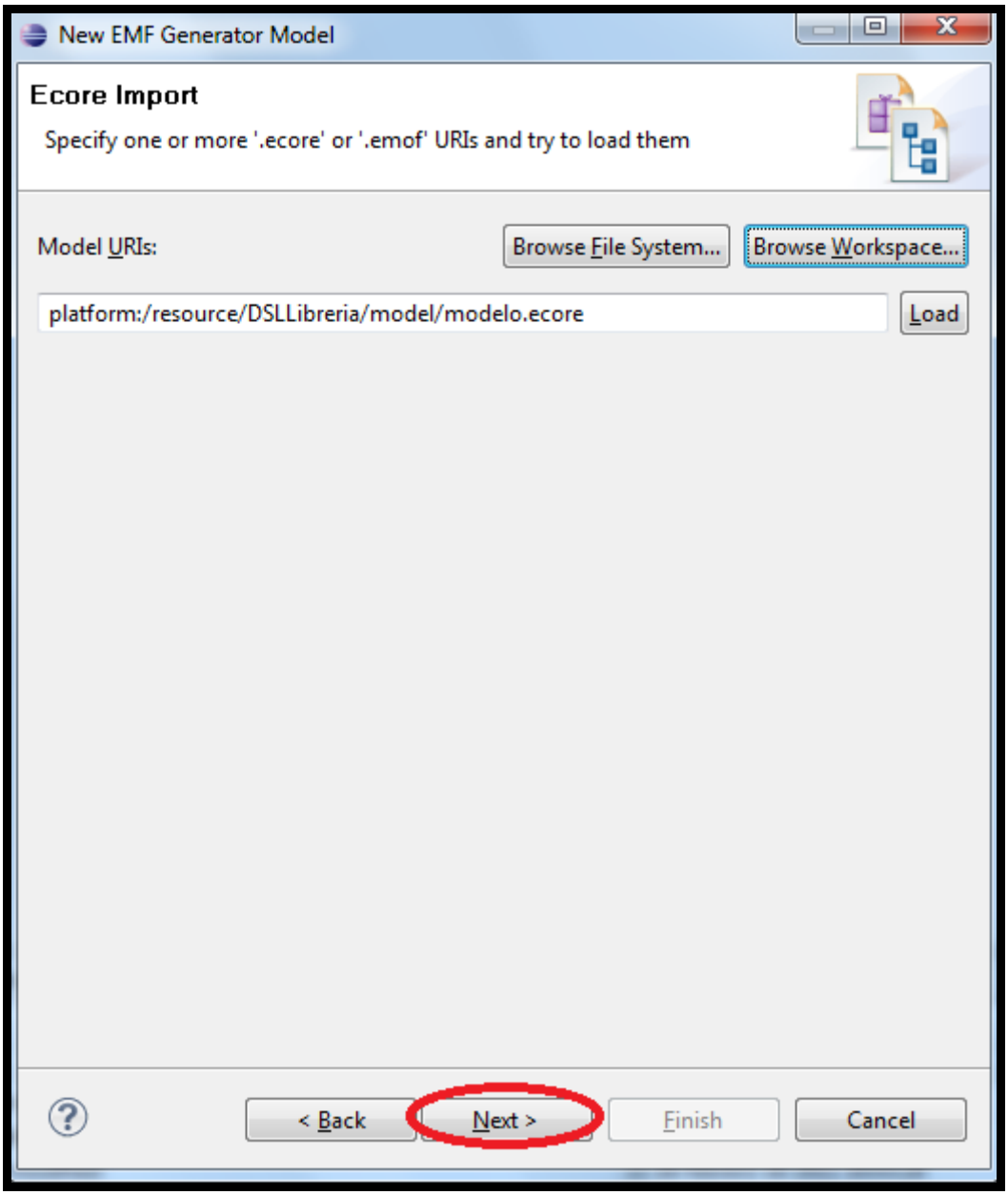
Seleccionamos “Ecore model” como modelo de entrada y pulsamos “Next”, como se ve a continuación.



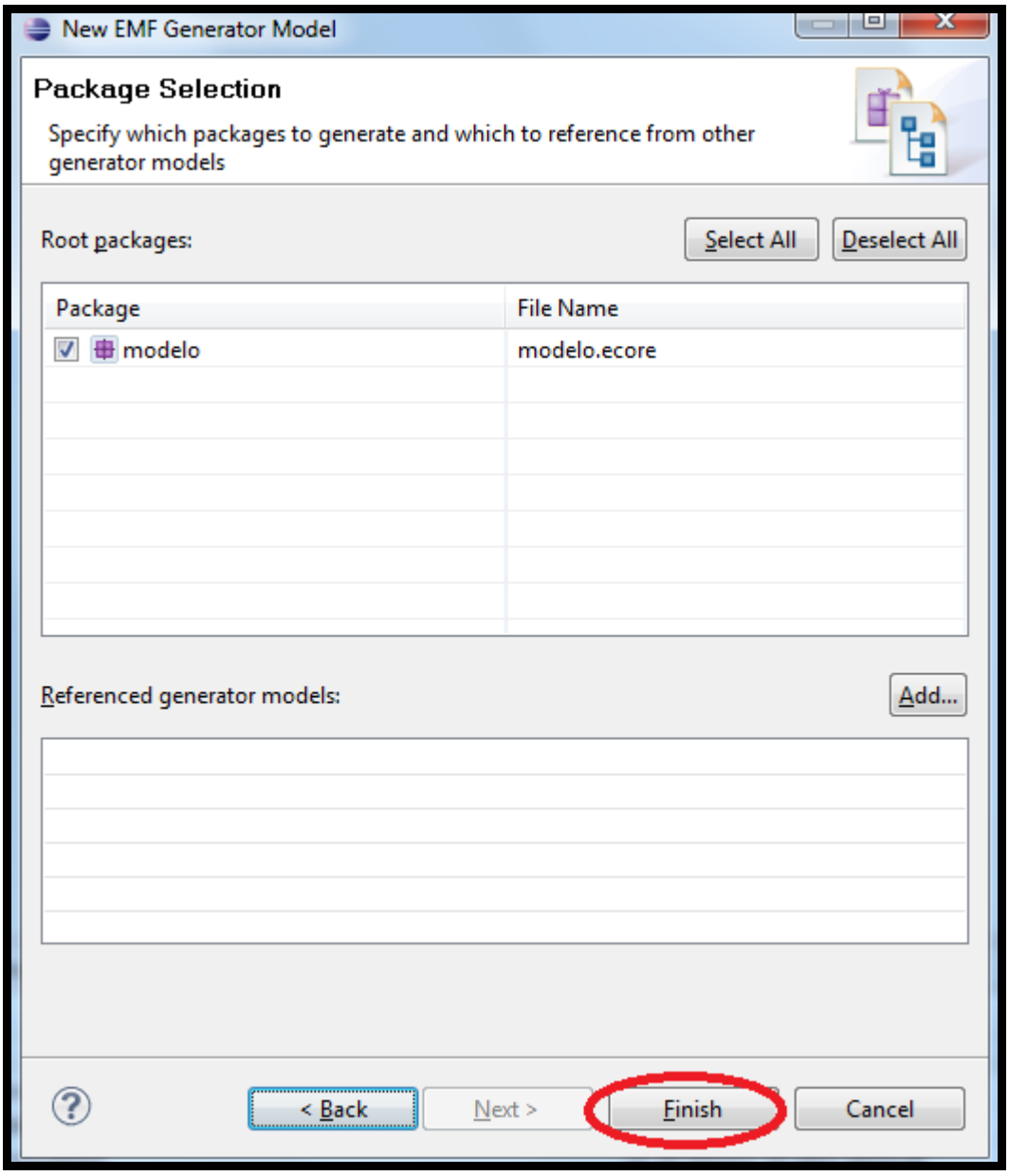
Pulsamos “Browse Workspace”, si el modelo Ecore se encuentra en nuestro espacio de trabajo en un proyecto abierto, o “Browse File System” si el modelo está fuera del espacio de trabajo, seleccionamos “modelo.ecore”, como modelo de entrada dentro de la carpeta “DSLLibreria” y “model” seleccionamos el fichero, y pulsamos Next, como se muestra a continuación



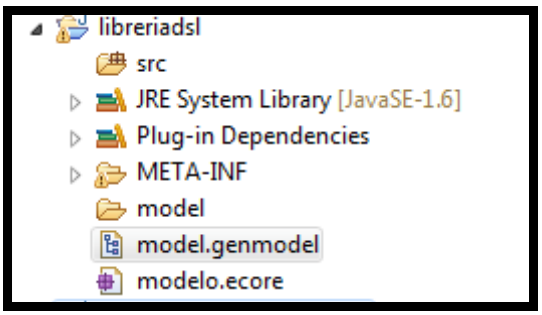
Pulsamos “Next” en la siguiente pantalla como se muestra abajo.



Y por ultimo “Finish”, como se ve en la síguete figura.



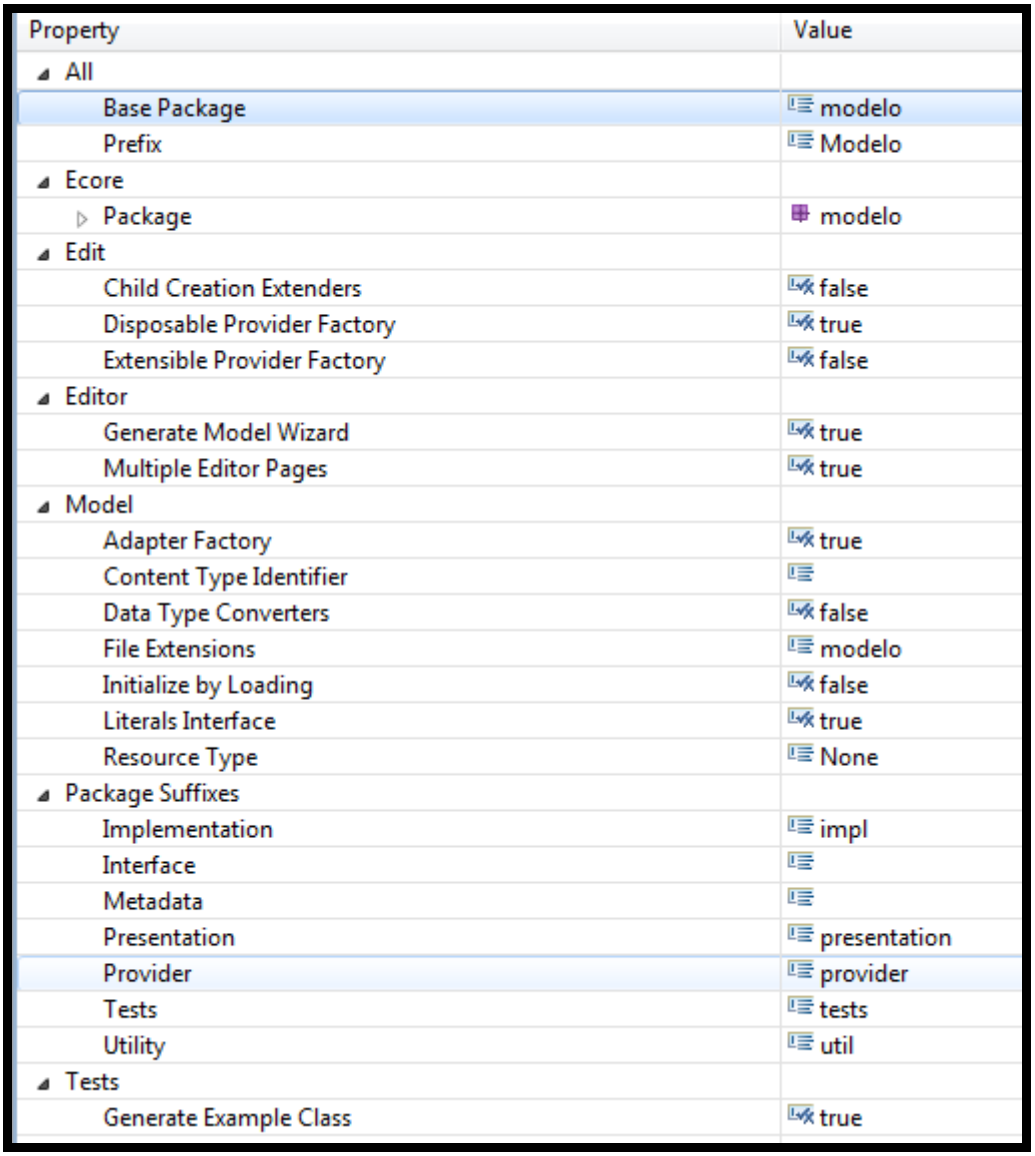
Si desplegamos el explorador de paquetes, debemos tener una vista similar a la de la siguiente figura.



5.3 Generación del código del modelo

En el apartado anterior se ha creado un nuevo elemento con extensión “genmodel”. Este elemento es un modelo que permite transformar automáticamente el modelo Ecore que hemos definido a código fuente. El código se genera aplicando patrones de transformación. El resultado de la generación es un conjunto de clases Java, que serán utilizadas más adelante en el proceso de creación de la herramienta de modelado específico de dominio para que todos los elementos que constituyen la herramienta se comporten tal y como el modelo ecore establece. Este proceso de generación se sirve de EMF y el modelo se ha importado como un modelo EMF dentro del proyecto.

Antes de generar el código del modelo es preciso configurar el modelo “genmodel” para especificar su paquete base. Para ello hacemos doble clic en “modelo.genmodel”. Se abre una ventana con un árbol similar al de “modelo.ecore”. Clicamos con el botón derecho sobre el paquete morado, de nombre “Modelo”, y seleccionamos “Show properties view”. Se nos abre una tabla que debemos rellenar como se muestra a continuación, principalmente el campo “Base Package”.



Ahora está todo listo para generar el código del modelo. Clicamos con el botón derecho sobre el paquete morado y seleccionamos “Generate Model Code”. Se generará automáticamente un plugin con el código del modelo. A continuación, realizamos el mismo proceso seleccionando “Generate Edit Code” y “Generate Editor Code”. Este último será el intérprete de Eclipse que nos permita ver nuestro modelo específico de dominio en forma de árbol tal y como nos muestra los documentos Ecore y cualquier componente de GFM.

5.4 Definición gráfica del modelo

La definición gráfica del modelo consiste en definir el aspecto que van a tener las primitivas de modelado en nuestro editor gráfico. Así, GMF de una forma totalmente automática, nos permite personalizar el aspecto de la aplicación de construcción de modelos específicos de dominio a nuestro gusto. Para este tutorial se han elegido las siguientes metáforas gráficas:

Primitiva gráfica Libros: Rectángulo con picos romos, bordes de color azul oscuro y lineal e interior de color azul claro.

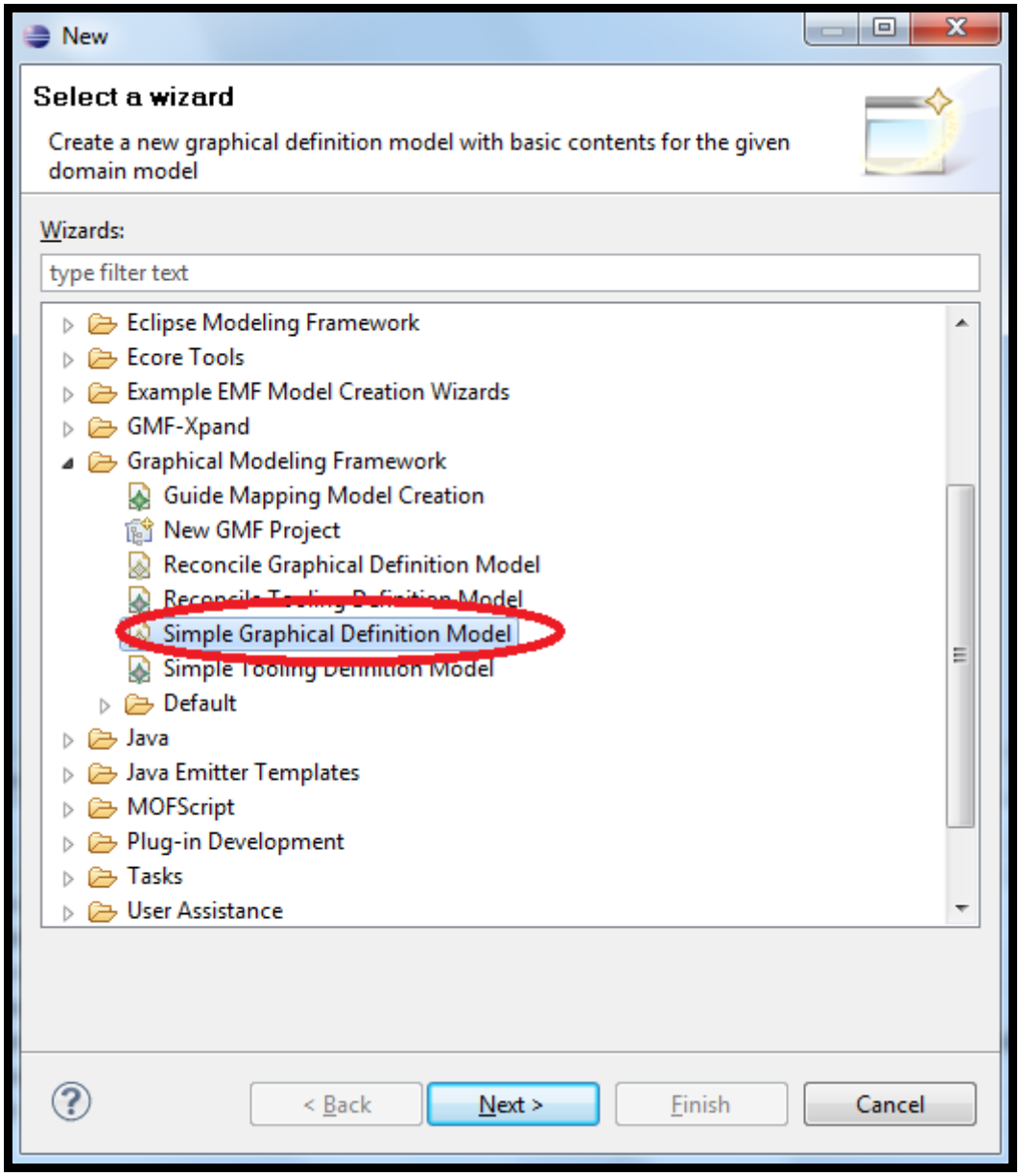
Primitiva gráfica Editorial: Elipse con los bordes punteados de color gris e interior de color gris claro.

Primitiva gráfica Resumen: Rectángulo, con los bordes de rayas discontinuas color naranja e interior amarillo claro.

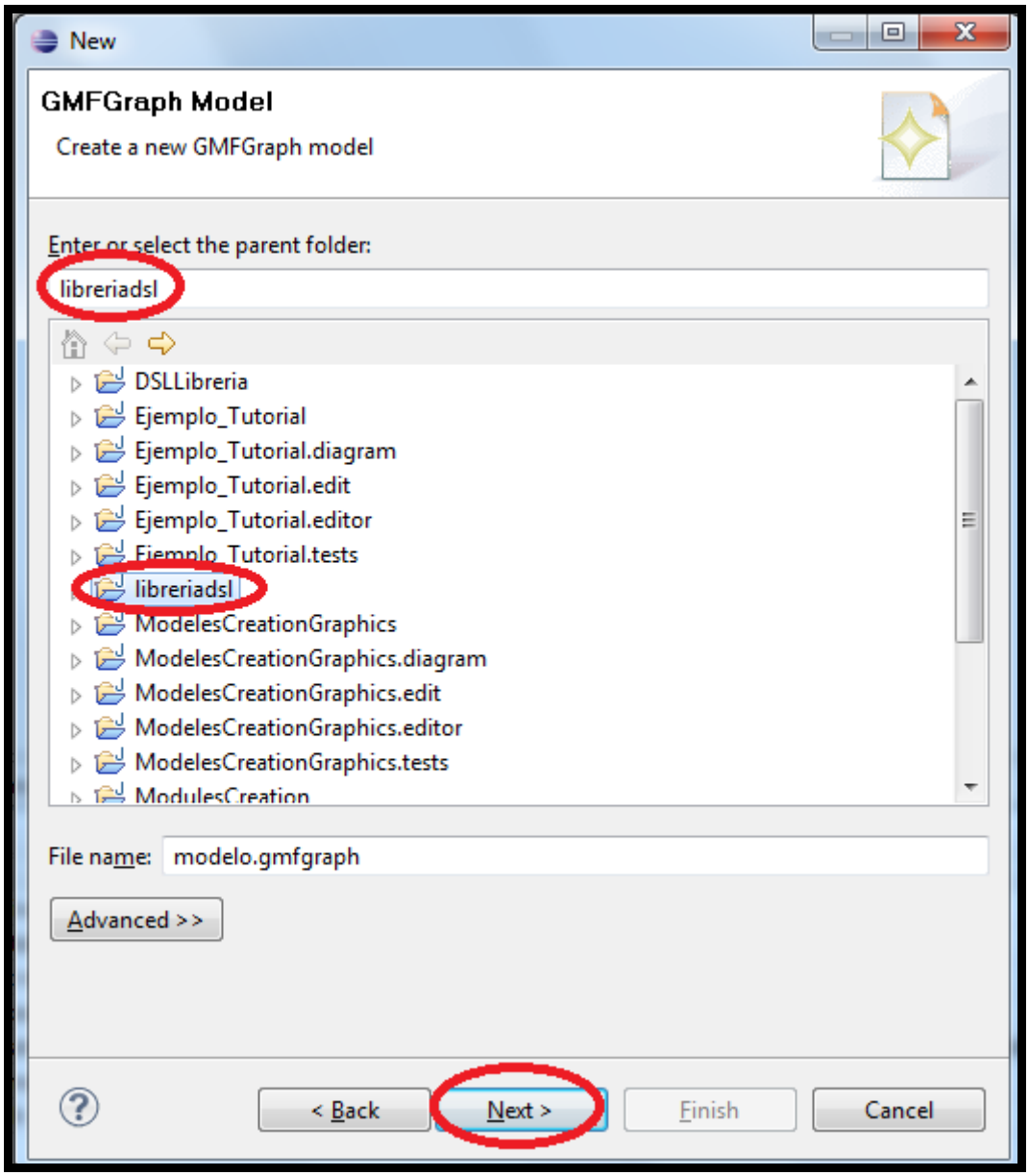
Primitiva gráfica Libro_Tiene_Editorial: Línea fina de color azul claro.

Primitiva gráfica Libro_Tiene_Resumen: Línea gruesa discontinua de color azul oscuro.

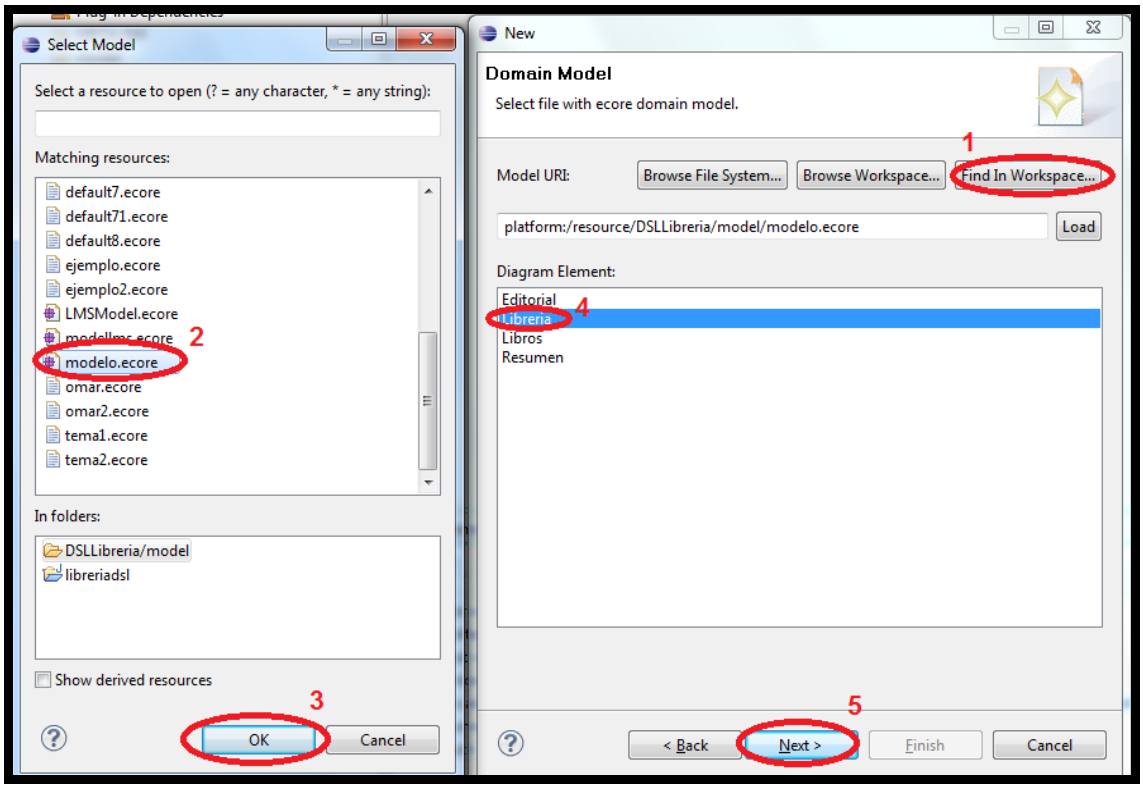
Una vez elegido el aspecto de la aplicación, podemos empezar a crear la definición gráfica del modelo. Para ello hacemos “File -> New -> Other”, vamos a la carpeta “Graphical Modeling Framework” y seleccionamos “Simple Graphical Definition Model” y pulsamos “Next”, como se muestra a continuación.



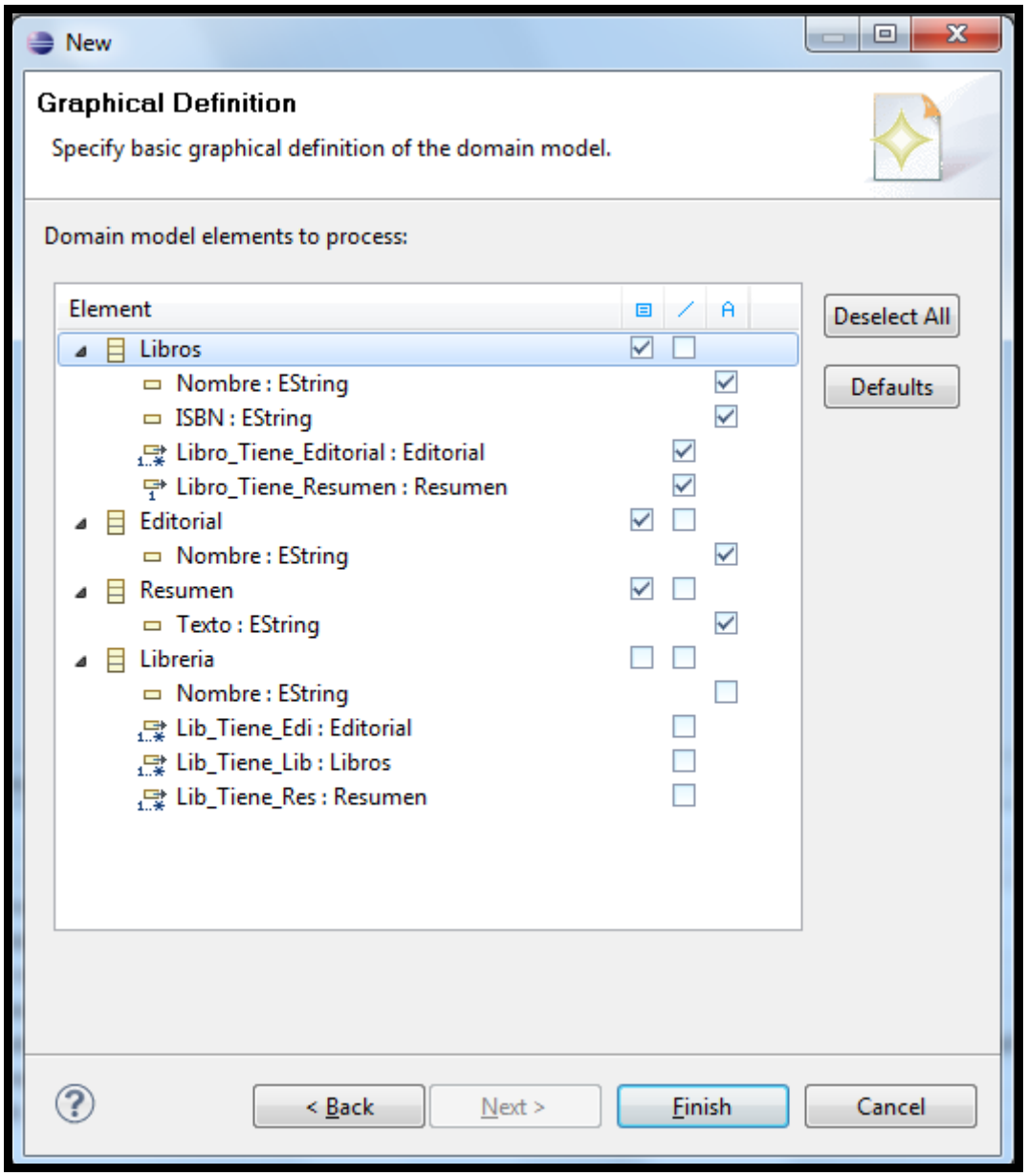
Cambiamos el nombre a "modelo.gmfgraph" y pulsamos "Next" comprobando que el directorio raíz marcado es tienda, como se muestra a continuación.



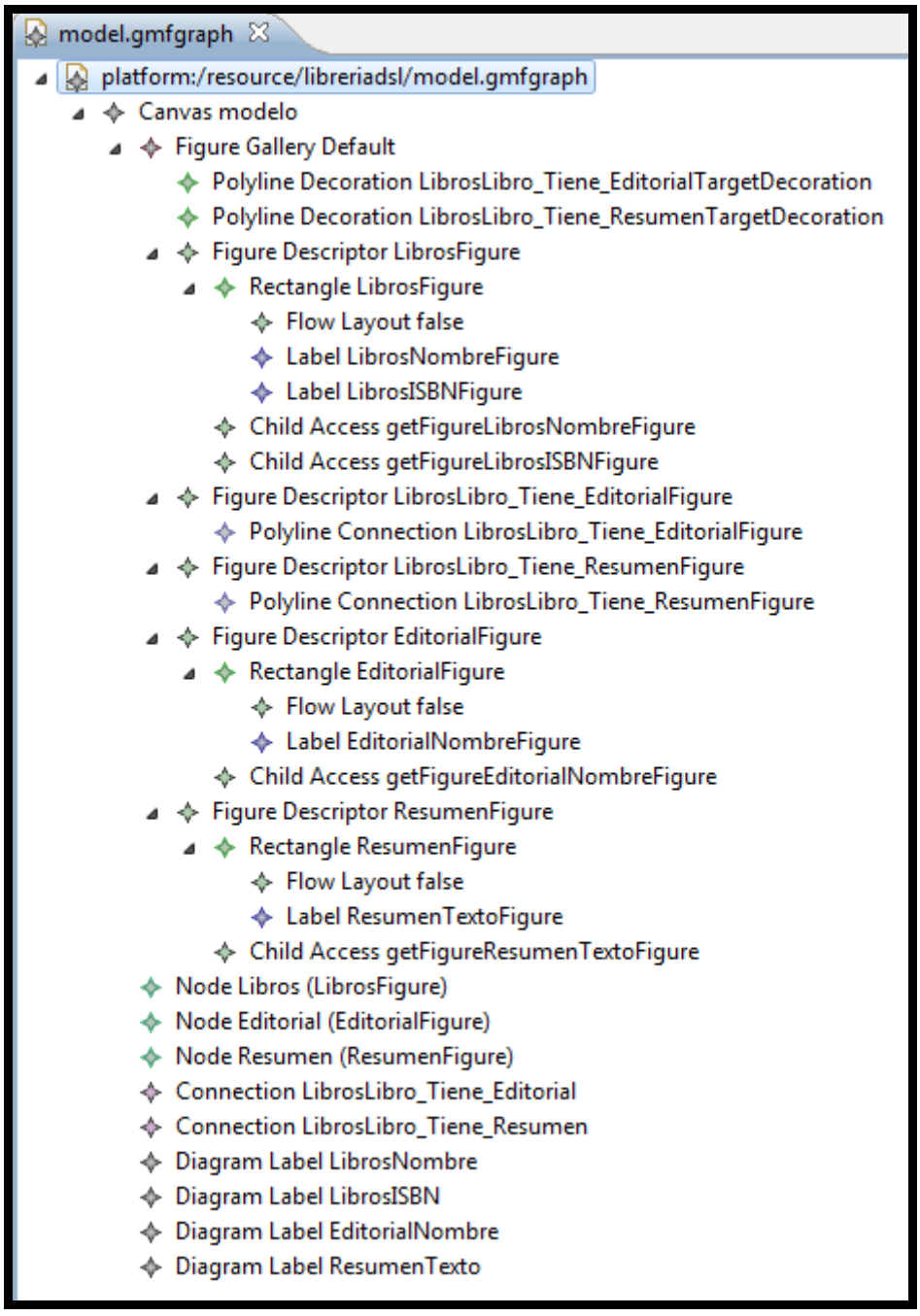
Seleccionamos “modelo.ecore” como modelo de entrada, si no aparece por defecto, pulsamos en “Find in Workspace” para seleccionarlo. Después, seleccionamos el elemento raíz de nuestro modelo, es decir, el que contiene al resto, en nuestro caso es “Librería” y pulsamos Next, como se muestra a continuación.



En este paso nos aparecerá una ventana como la de la siguiente figura, en la que podremos elegir qué elementos del modelo del dominio actuarán como nodos, cuáles como enlaces y cuáles como etiquetas. En la ventana aparecen representados en una tabla de tres columnas en la que se encuentran colocados de izquierda a derecha, y están representados gráficamente en la cabecera de la siguiente forma: cuadrado con rayas horizontales (nodos), línea inclinada (enlaces), y letra A (etiquetas). Cada elemento del modelo tiene asociad un checkbox (casilla de selección) en cada columna de la tabla que es posible seleccionar, para así elegir el tipo de representación gráfica que se desea. En nuestro caso, hacemos la selección como se muestra en la siguiente figura y pulsamos “Finish”.



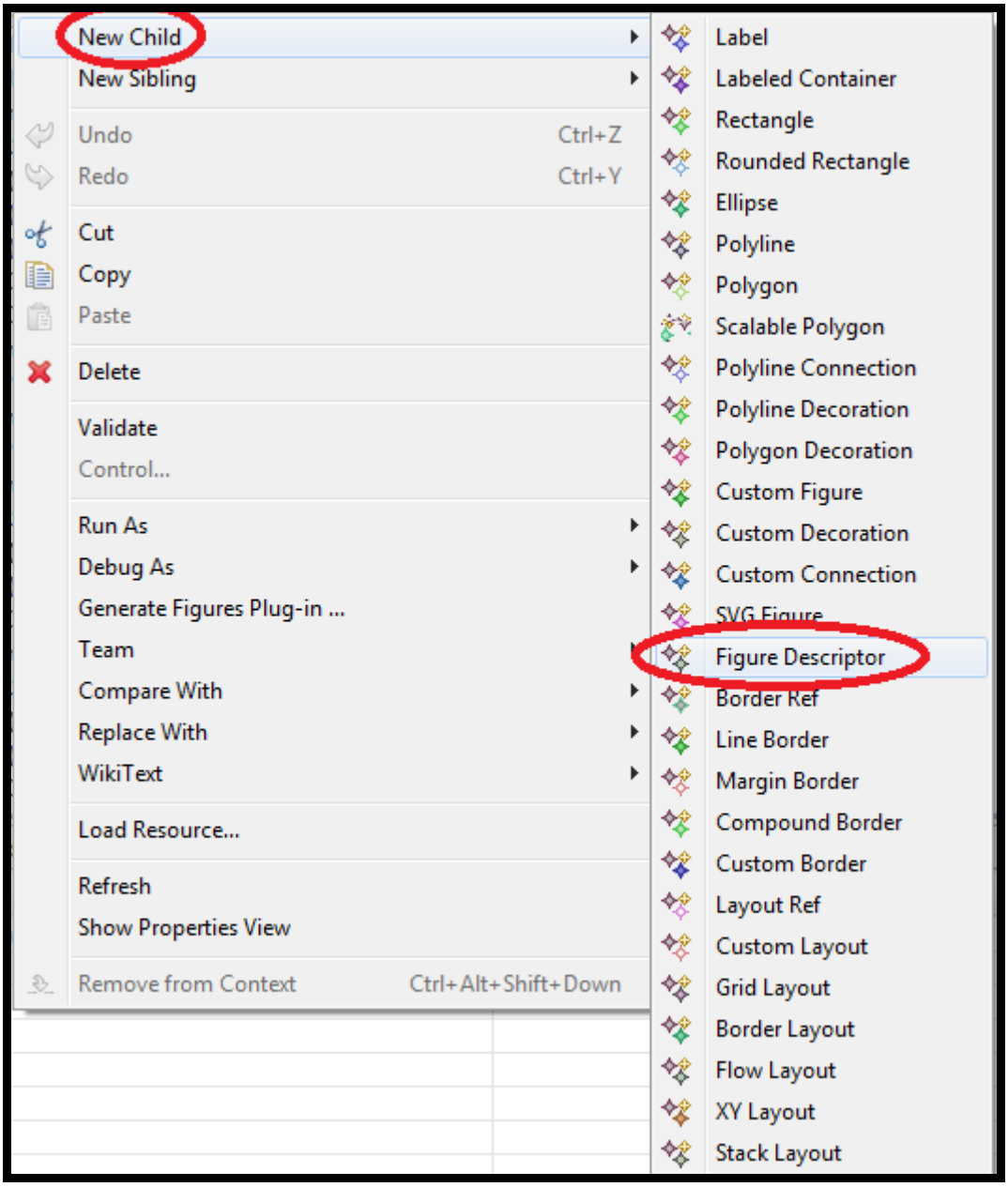
Al pulsar “Finish” aparecerá en la ventana el “modelo.gmfgraph”. Este tiene estructura de árbol, si desplegamos el elemento raíz, nos aparecerá una vista similar a la siguiente.



A continuación, seleccionamos “Figure Gallery Default” observando que los enlaces van determinados por un “Polyline Decoration” y el nombre de la primitiva que representa ese enlace. Desplegando los nodos del modelo (Libro, Editorial, Resumen) vemos que vienen determinados por un descriptor de figura que contiene la figura “Rectangle” y contiene las funciones de acceso a esas etiquetas “Child Access getFigure”. Si a su vez la desplegamos la figura “Rectangle” de cada nodo, vemos que contiene sus etiquetas “Label” y la alineación de las etiquetas “Flow Layout”, aparte de esto, por cada figura existe un nodo “Node” de acceso al “FigureDescriptor” y por cada enlace una “Connection” de acceso al “Polyline Decoration”.

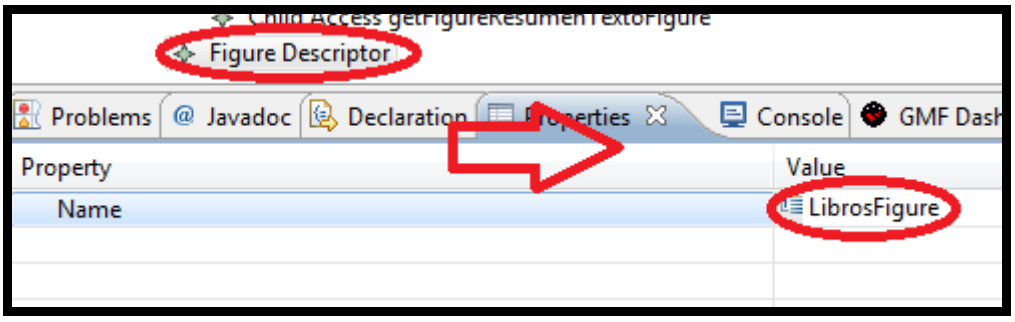
Sabiendo todo esto, vamos a personalizar la interfaz de la aplicación. Para empezar, vamos a crear el diseño del Libro.

Hacemos clic con el botón derecho sobre “Figure Gallery Default -> New Child -> Figure Descriptor”, como se muestra a continuación.

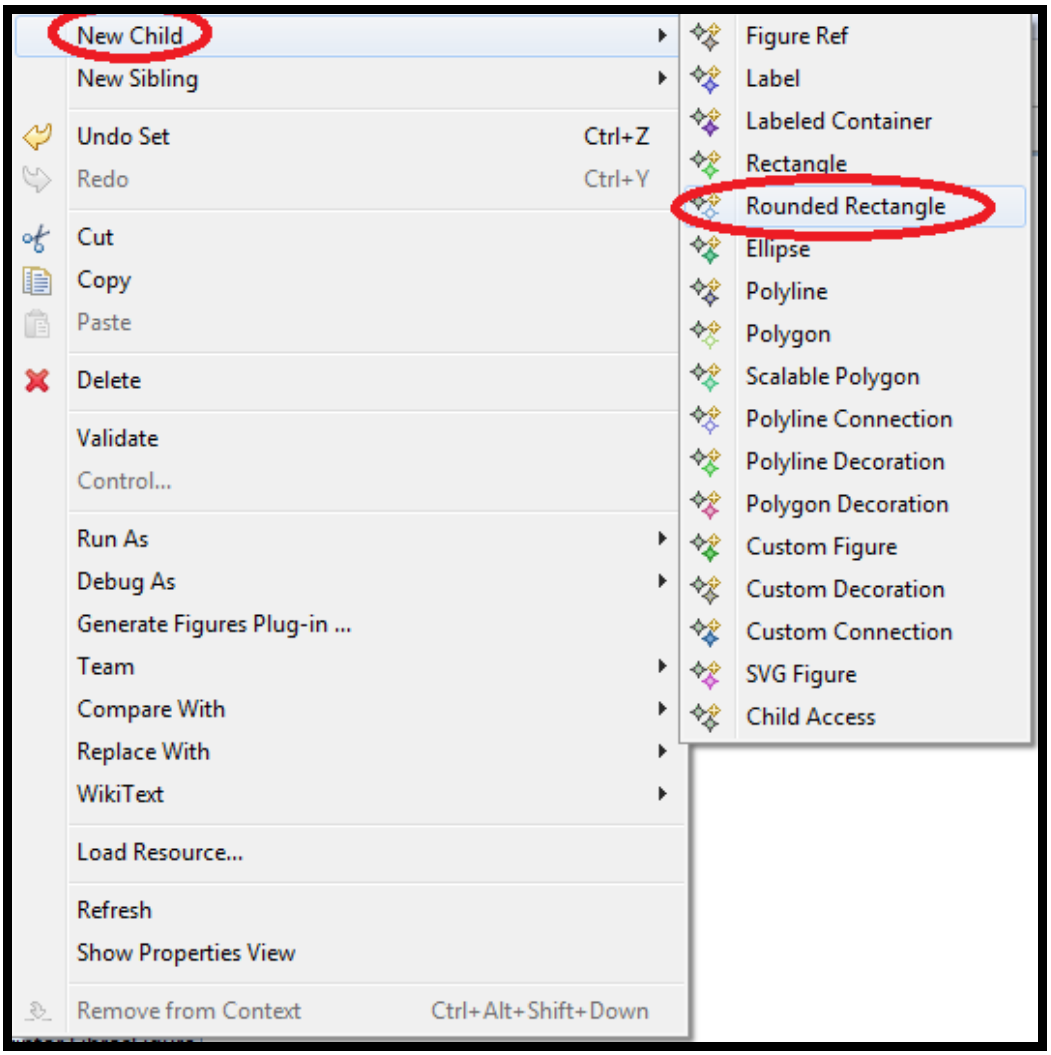


Para personalizar las metáforas gráficas de las primitivas de modelado crearemos nuevos descriptores gráficos. Por ejemplo, para la primitiva Libros, se crea un nuevo descriptor,

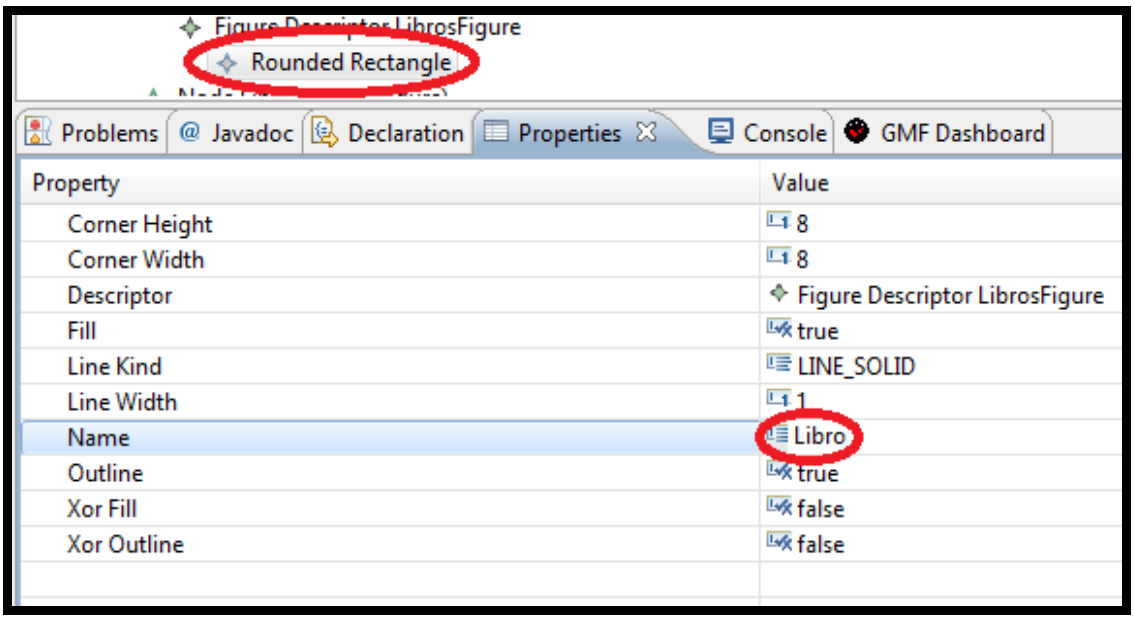
hacemos clic con el botón derecho sobre él y seleccionamos “Show Properties View”. Le ponemos el nombre del descriptor que tenemos del Libro, LibrosFigure, como se muestra a continuación.



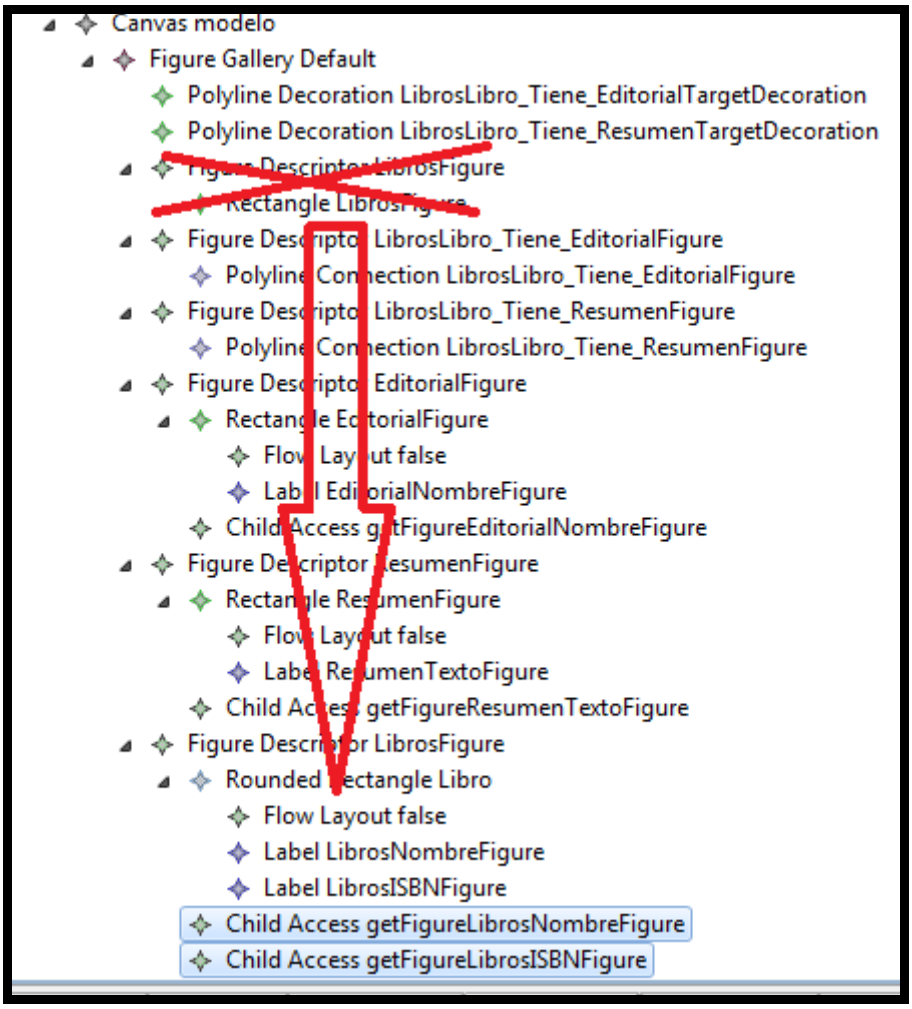
Luego hacemos clic sobre el descriptor de nuevo con el botón derecho y seleccionamos “New Child -> Rounded Rectangle”, ya que era lo que habíamos definido en el diseño.



Le pondremos “Libro” de nombre, como se muestra a continuación.

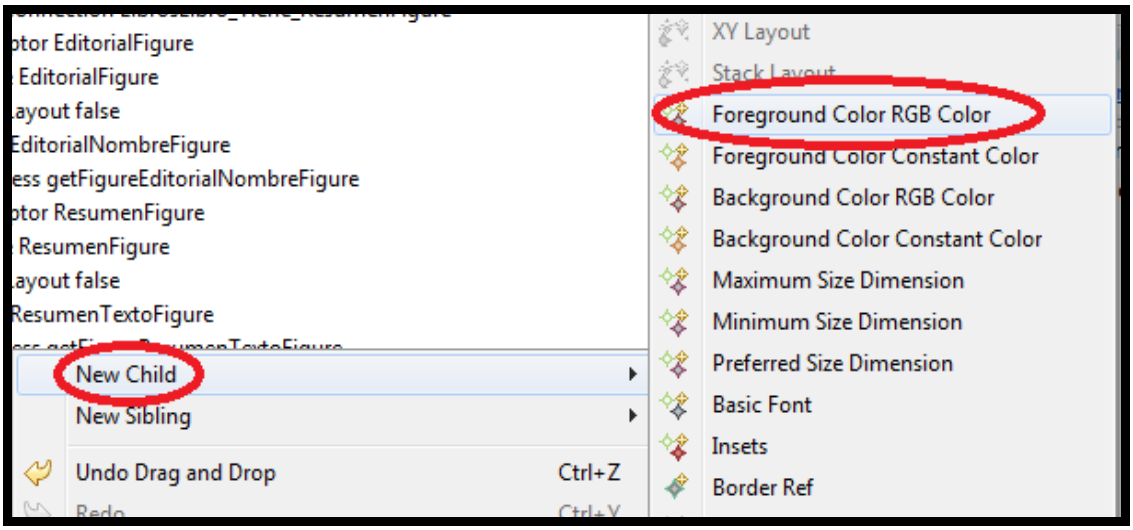


Seleccionamos todos los elementos del “FigureDescriptor LibrosFigure” anterior y los trasladamos arrastrándolos al lugar donde corresponden (el nuevo descriptor creado), excepto el “Rectangle LibrosFigure”. Por lo tanto, los “Labels” y el “Flow Layout” irán en la jerarquía dentro del “Rounded Rectangle”, y los “Child Access” irán dentro del “Figure Descriptor”, como se muestra a continuación.

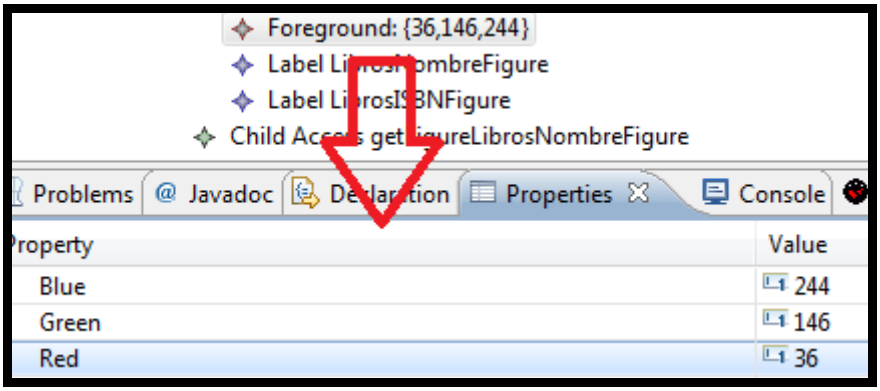


Una vez que hemos trasladado todos los elementos al nuevo Descriptor, borramos el antiguo.

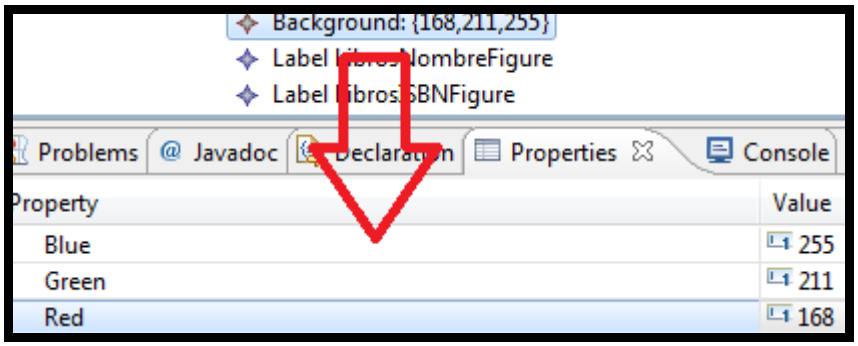
Ahora sólo queda darle color. Para ello, pulsamos con el botón derecho sobre el “Rounded Rectangle libro ->New Child -> Foreground RGB color”, como se ve a continuación.



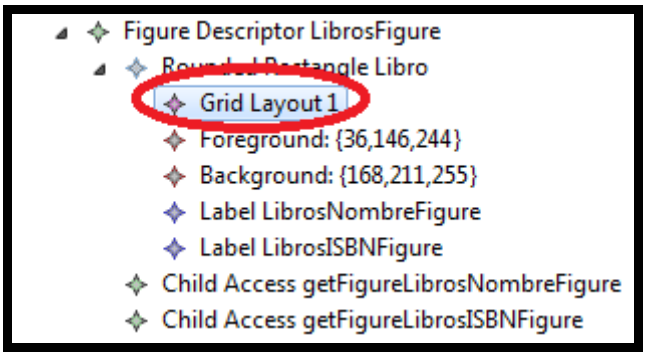
Luego introducimos los valores en RGB, para este caso como se muestra en la siguiente figura.



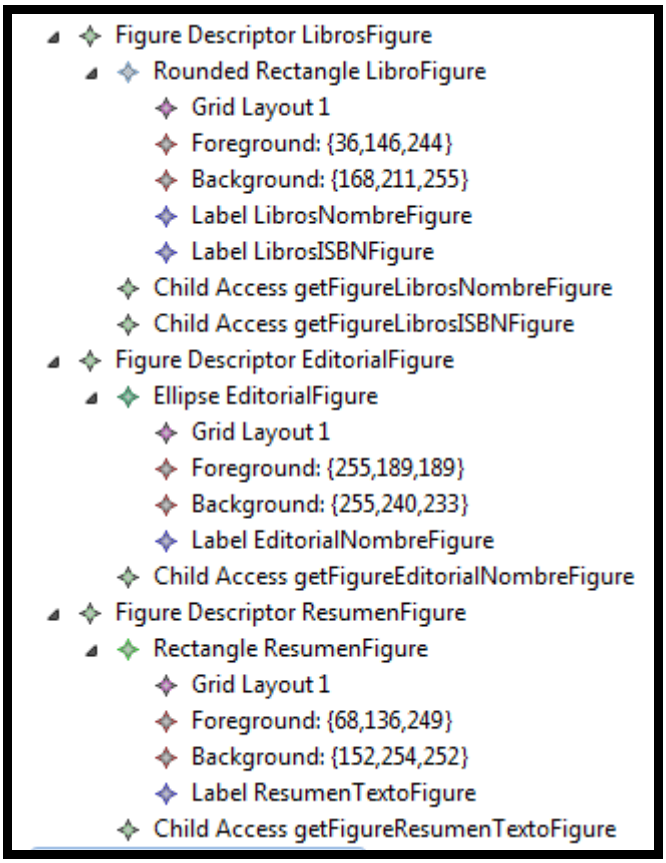
Seguimos el mismo proceso seleccionando “Background RGB Color” e Introducimos los valores siguiente figura.



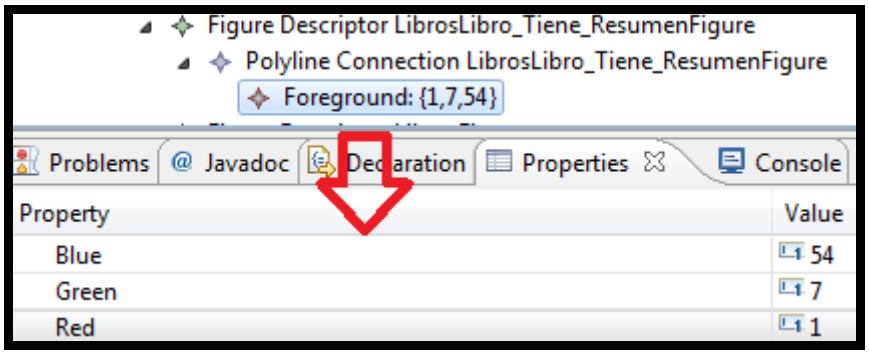
Por último borramos “Flow layout” e insertamos “Grid layout” de la misma forma, a través de “New Child”. Esto se hace para que las etiquetas estén en columnas y no en filas, como se muestra en la siguiente figura.



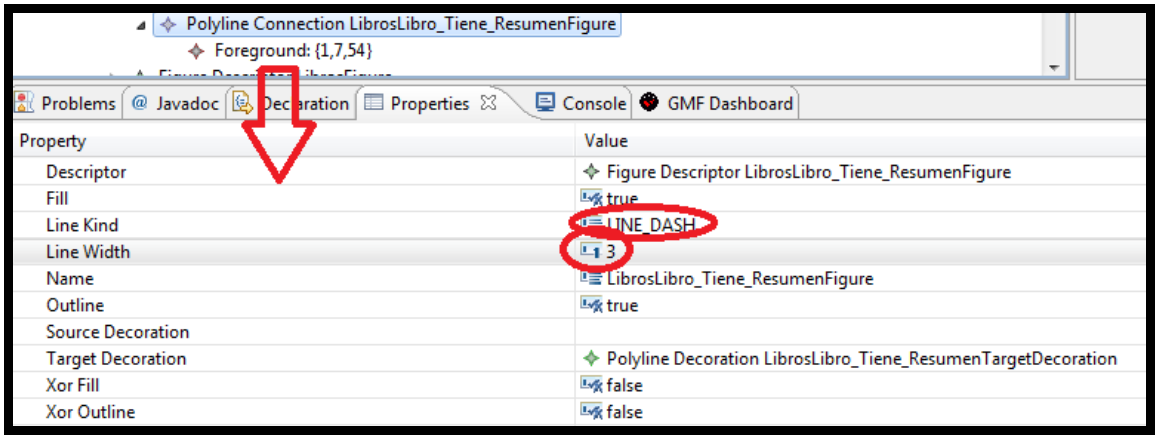
Con el resto de nodos, seguimos el mismo proceso para definir su primitiva, la siguiente figura muestra como deberían quedar.



Ahora haremos la definición gráfica de un enlace, en particular de “Libro_Tiene_Resumen”. Para ello desplegamos el descriptor de ese enlace y clicamos sobre “PolylineConnection” añadiendo un “Background RGB color” con los valores de la síguete figura.

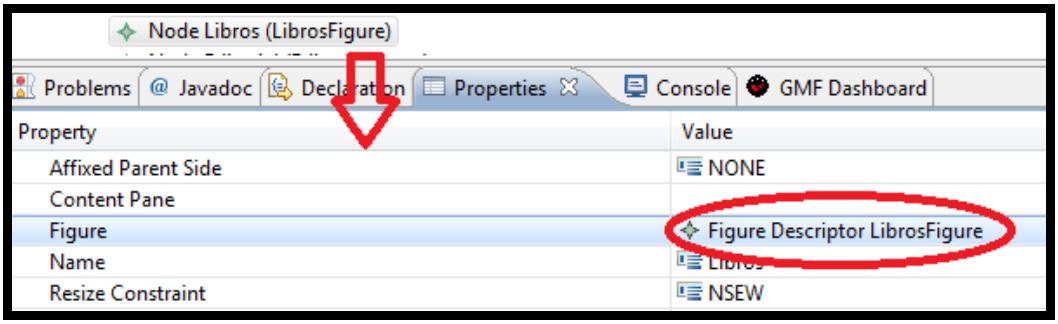


Y por último, ya que este enlace tiene un grosor determinado y es punteado, lo modificaremos clicando en "Polyline Connection" con el botón derecho y "Show Porperties View". Como se ve en la siguiente figura.

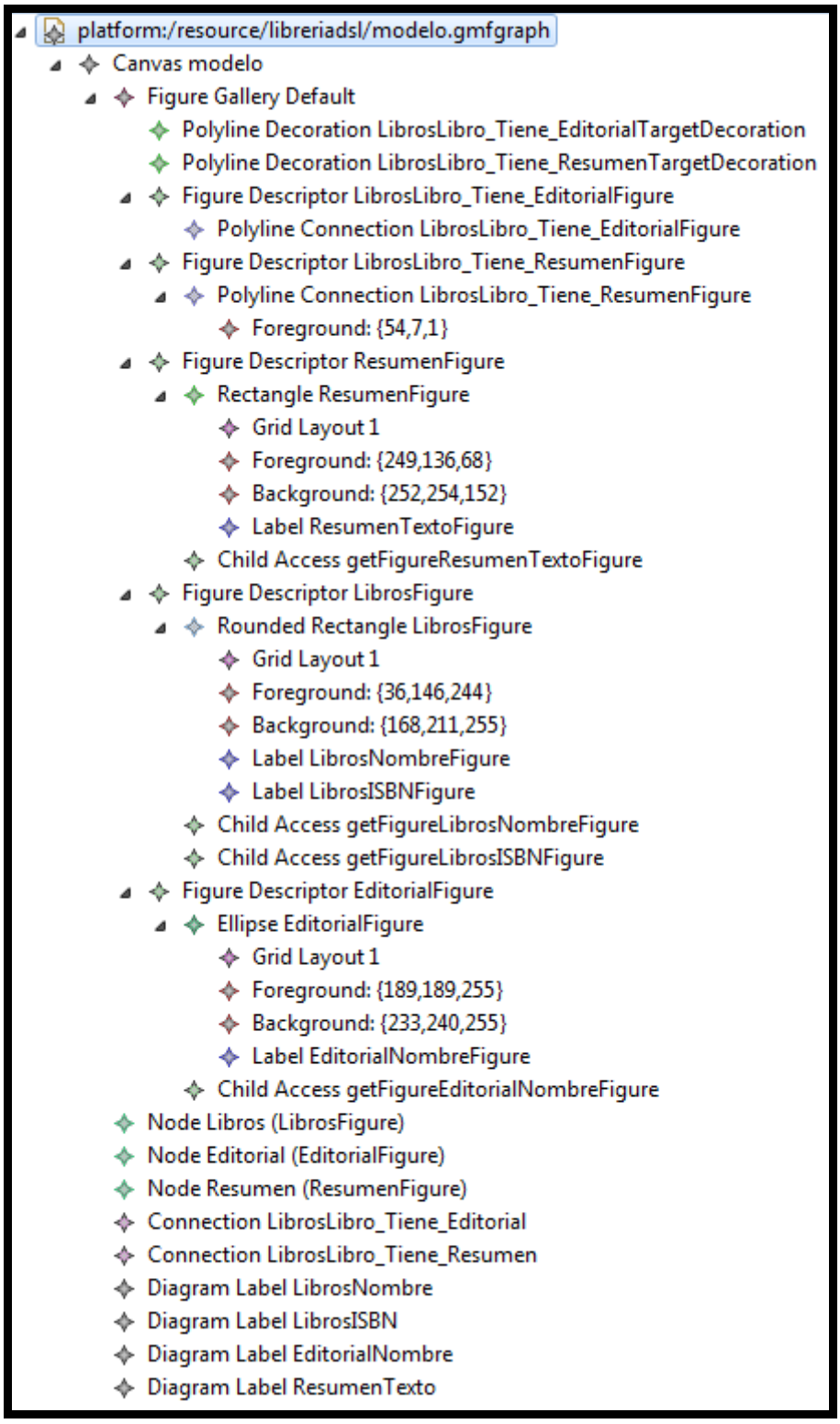


La definición de los bordes del resto de las figuras se realiza igual, accediendo a las propiedades y cambiando el "Line Kind" y el "Line Width".

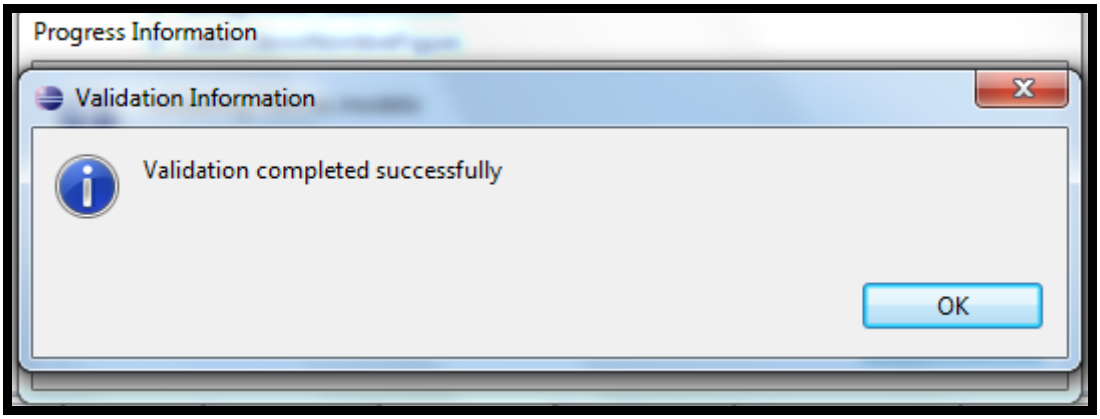
Una vez realizado el diseño hay que comprobar que los nodos de acceso a las figuras apuntan a las figuras correctas. Para ello, vamos a "Node Libro", hacemos clic con el botón derecho y seleccionamos "Show properties view". En el campo "Figure", desplegamos y seleccionamos la figura "LibrosFigure" como se muestra a continuación. También se ha de comprobar que los "Diagram Label" apuntan a la etiqueta y a la figura "Figure" a la que representan.



Tras definir los nodos ya tendremos la definición gráfica completa. La siguiente figura muestra la vista del árbol desplegado.

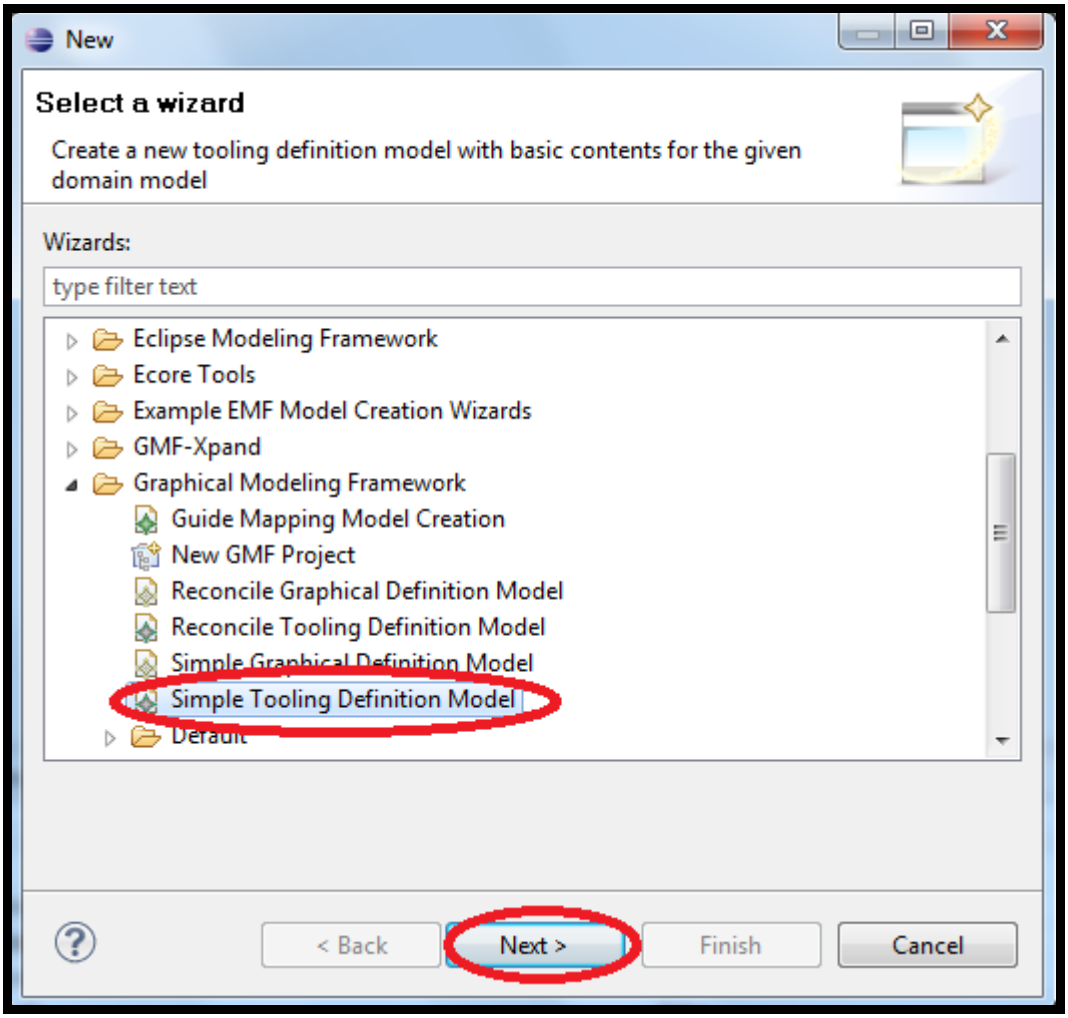


Antes de pasar al siguiente apartado es conveniente validar la definición gráfica anterior. Para ello, se ha de hacer clic sobre “Canvas modelo” con el botón derecho y seleccionar “Validate”. Si es correcto, pasar al siguiente punto, si no comprobar los campos “Figure” de los nodos y los “Diagram Label”. Deberá aparecer un mensaje como el de la siguiente figura.

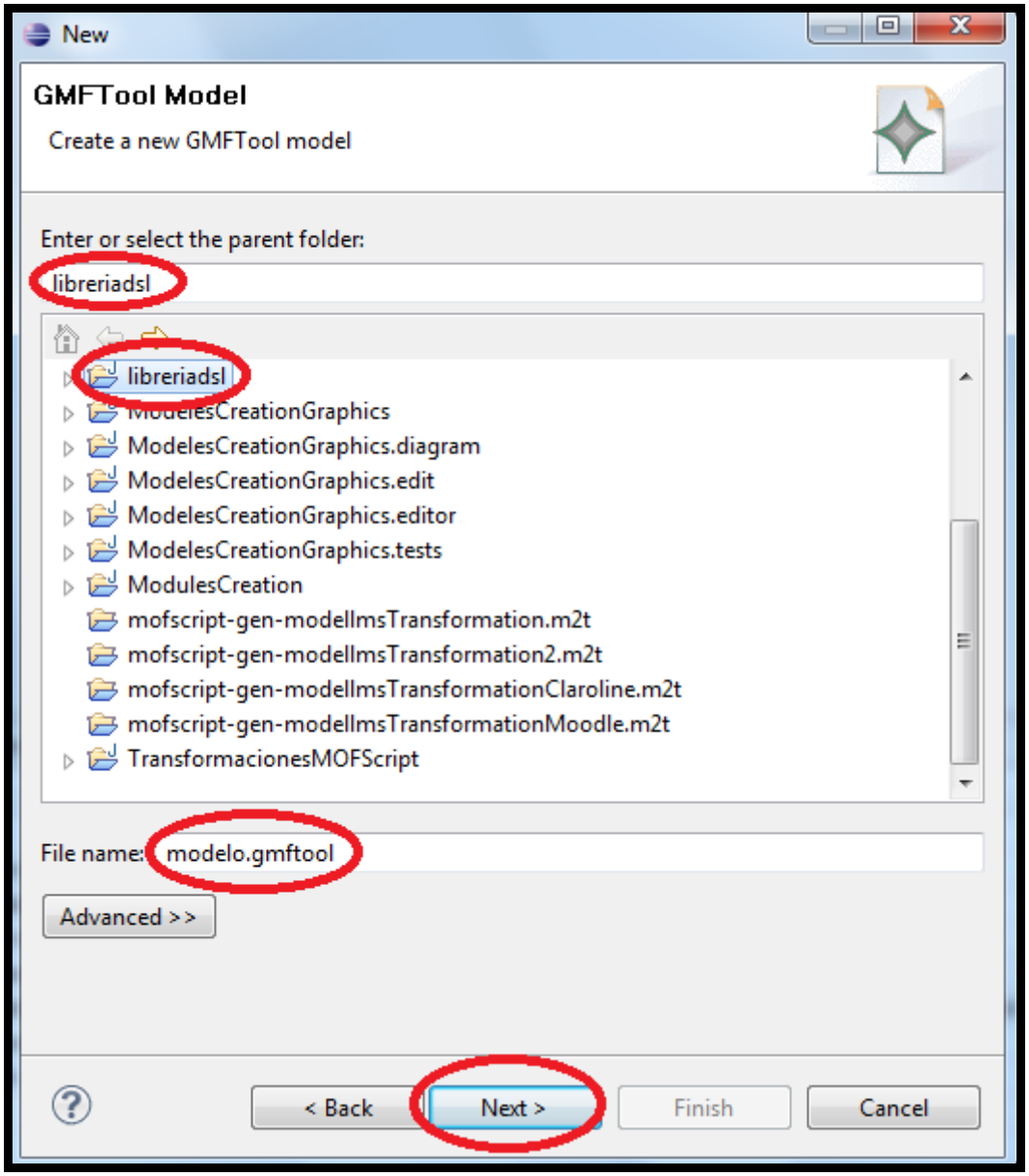


5.5 Definición de la paleta de herramientas de modelado

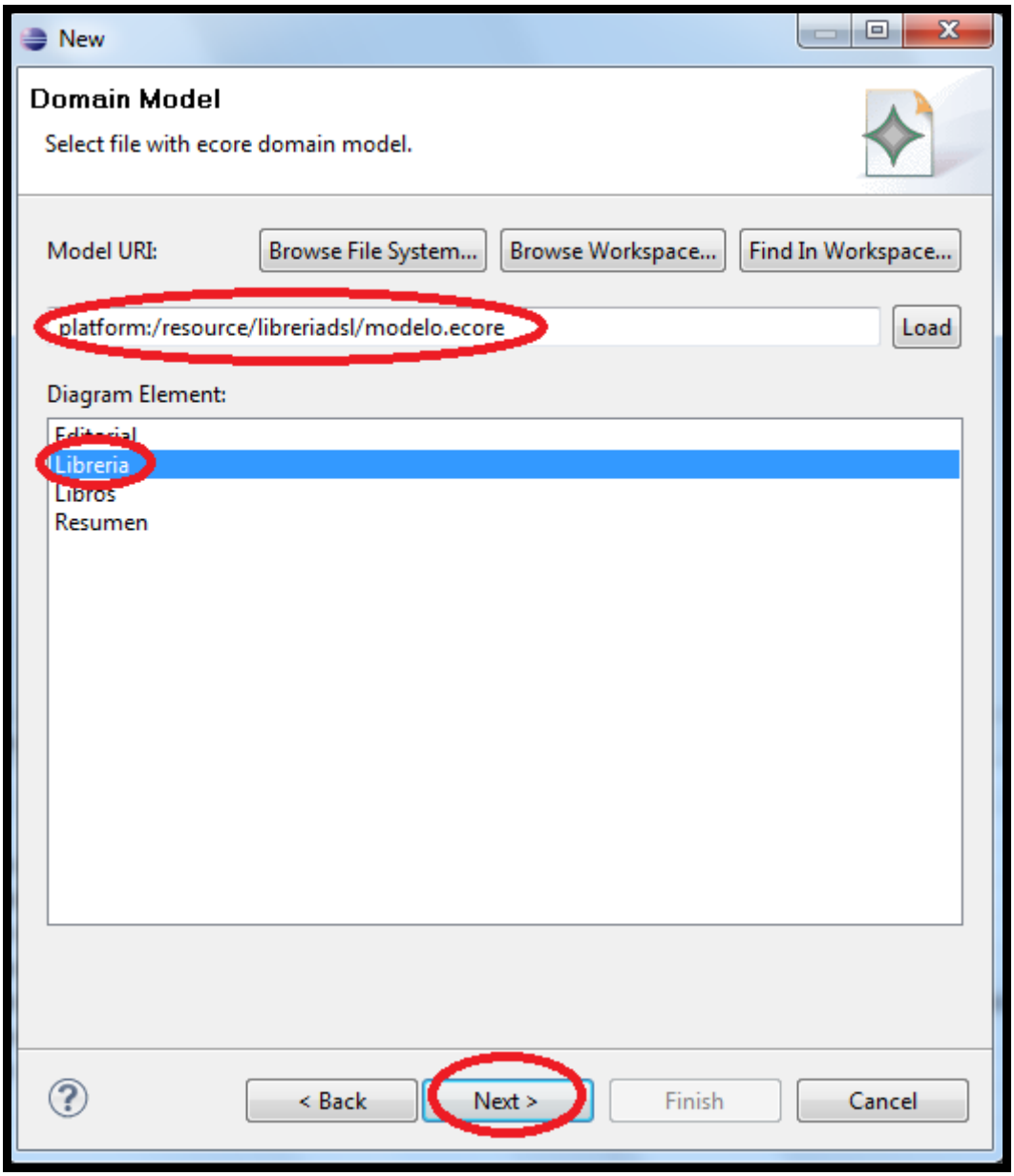
En este apartado definiremos la paleta de la aplicación con la que construiremos los modelos. Para ello, se ha de seleccionar “File -> New -> Other”, nos dirigimos a la carpeta “Graphical Modeling Framework” y seleccionamos “Simple Tooling Definition Model” y pulsamos “Next”, como se muestra a continuación.



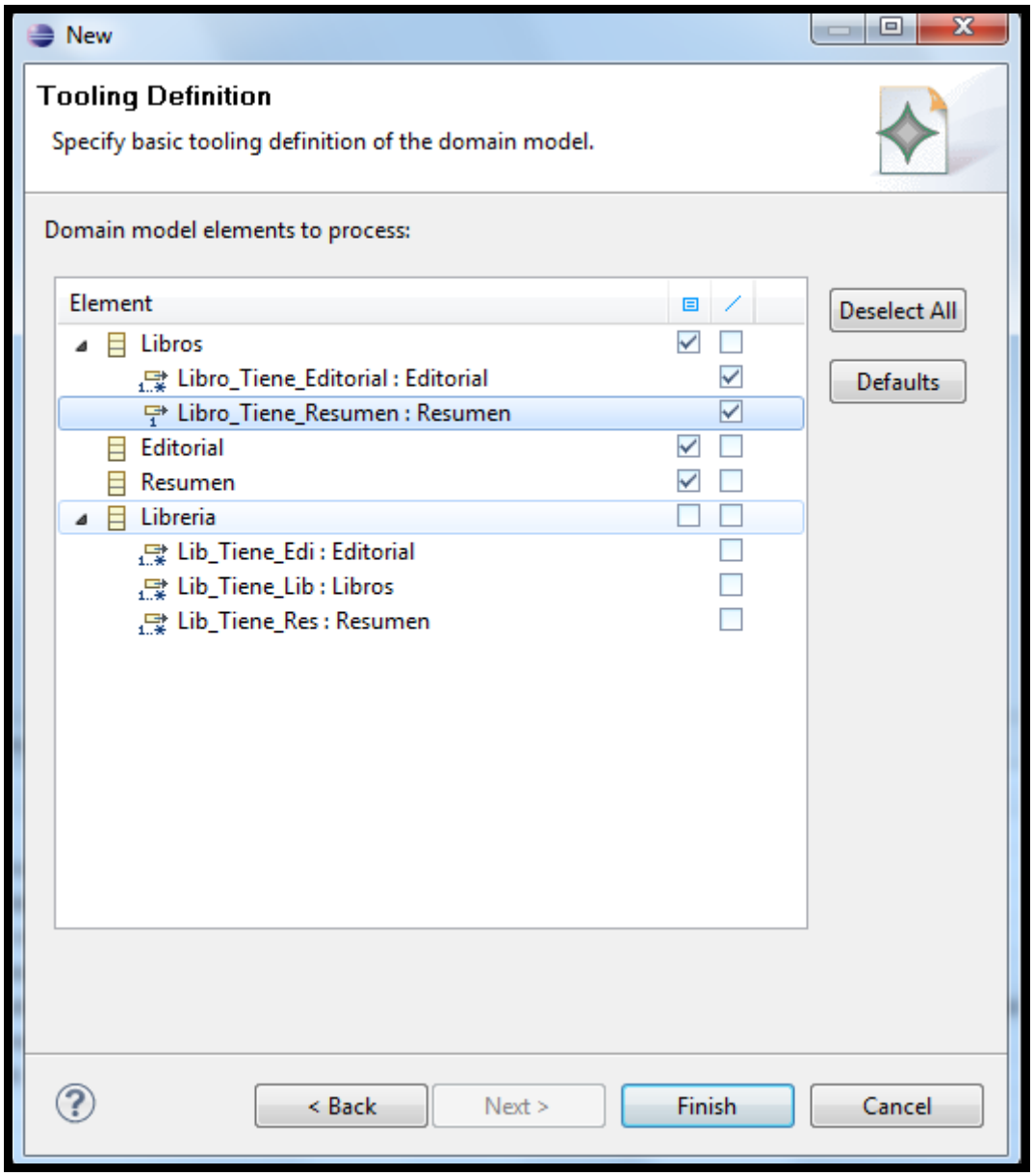
Llamamos a la definición de la paleta de herramientas “modelo.gmftool” y pulsamos “Next”, comprobando que el directorio raíz marcado es, como se muestra a continuación.



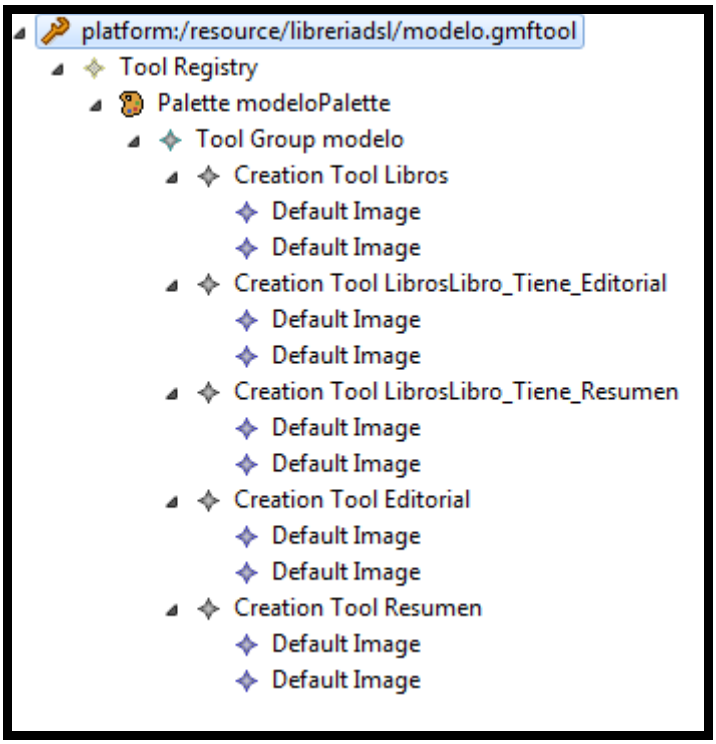
Seleccionamos “modelo.ecore” como modelo de entrada, si no aparece por defecto, pulsamos en “Find in Workspace” para seleccionarlo. Después, seleccionamos el elemento raíz de nuestro modelo, es decir, el que contiene al resto, en nuestro caso es “Librería” y pulsamos “Next”, como se muestra a continuación.



Al igual que en la definición gráfica, se nos presentan los elementos del modelo y nos permite elegir aquellos que van aparecer como nodos o como enlaces en la barra de herramientas. En nuestro caso, marcamos la selección como en la siguiente figura y pulsamos “Finish”.



Nos aparece un nuevo árbol, que representa la definición de herramientas de la aplicación a crear, como se muestra a continuación.



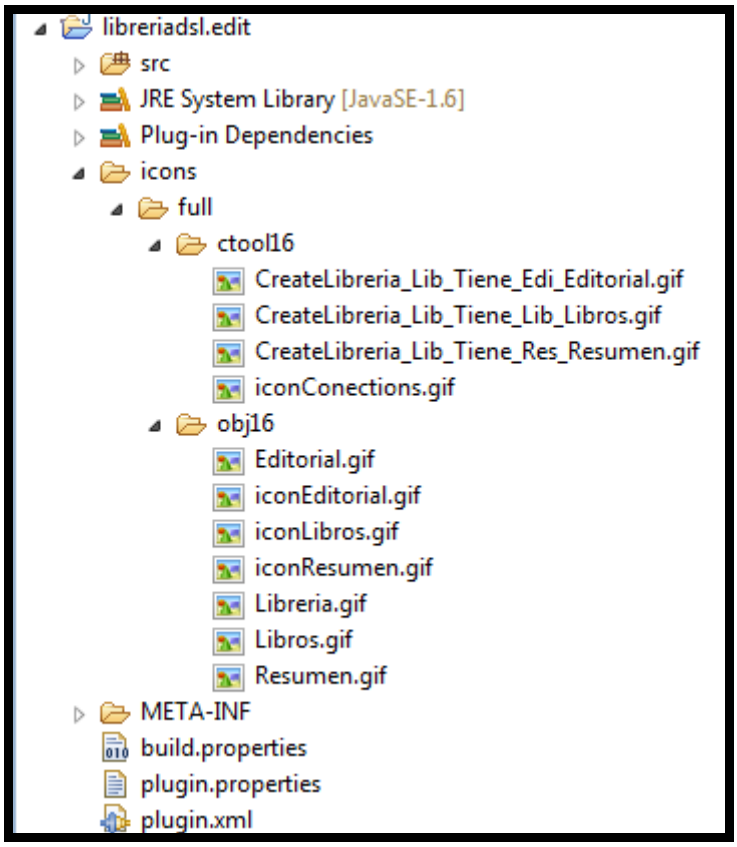
Cabe destacar que por cada primitiva seleccionada se crea una herramienta de creación y dos imágenes por defecto que corresponden a los iconos de la aplicación. El icono por defecto es ✦. Para distinguir unas herramientas de otras vamos a asignarles distintos iconos. Para ello, se han de copiar todos los iconos que vayamos a utilizar en “tienda.edit/icons/full/obj16”, no hay que borrar los iconos que están. Para nuestro caso utilizaremos los siguientes iconos:

Resumen

Libros

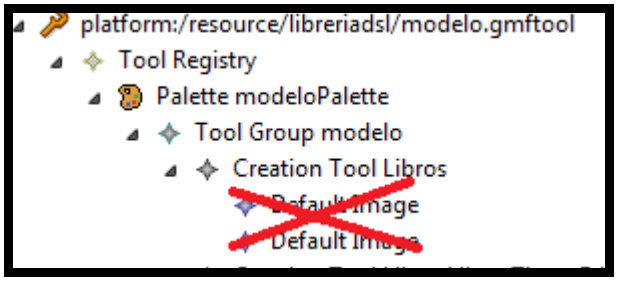
Editorial

Conexiones

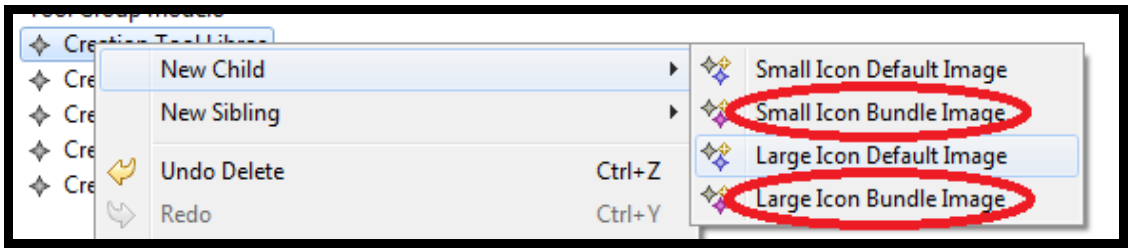


Posteriormente, procedemos a asignarle a cada nodo su icono. A continuación, mostramos un ejemplo para el caso del nodo Libros:

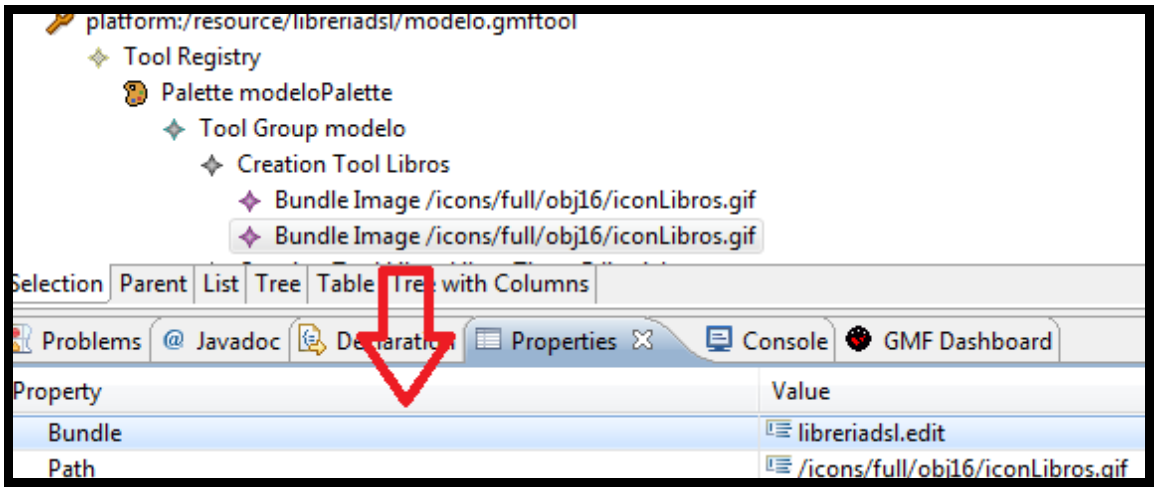
Borramos los dos “default image” que cuelgan de “Creation Tool Libros”.



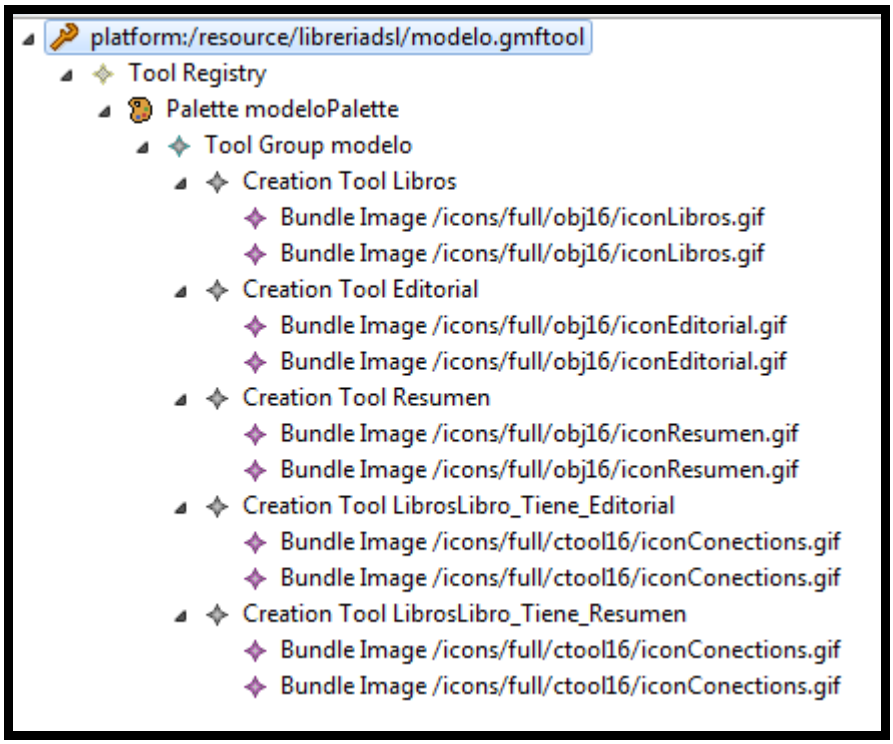
Hacemos clic con el botón derecho sobre “Creation Tool Libros”, seleccionamos “New Child” y creamos un “Small Icon Bundle Image” y un “Large Icon Bundle Image”, como se muestra a continuación.



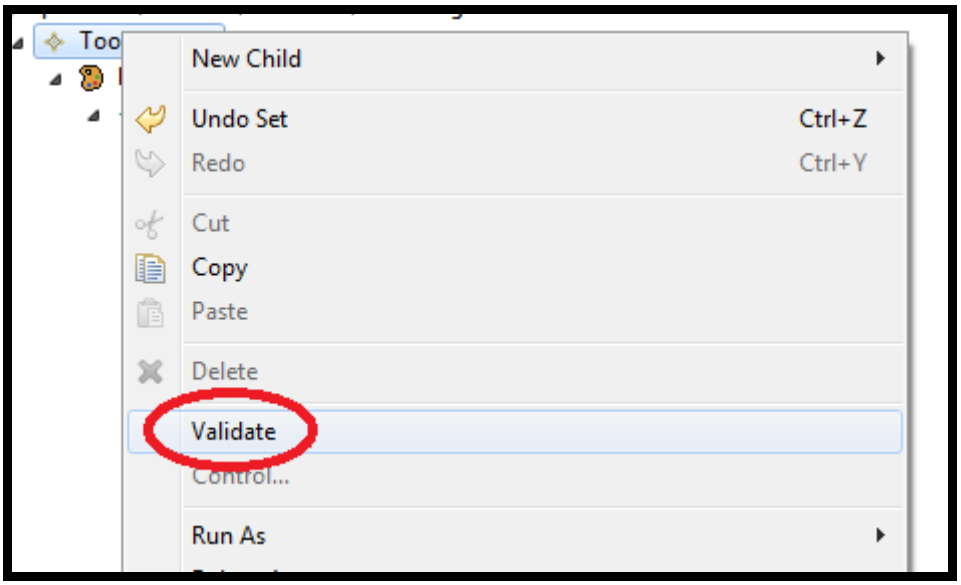
Definimos las propiedades de ambos como muestra en la siguiente figura.



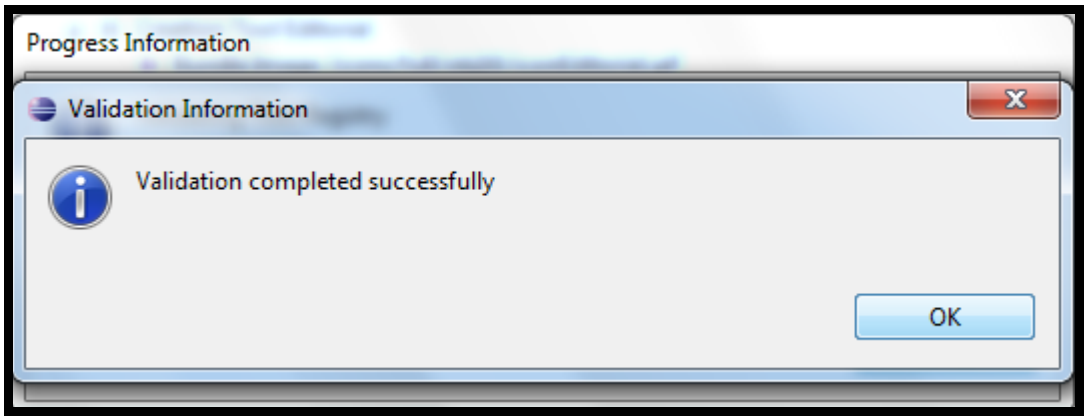
El resto de iconos tienen la misma ruta y hay que cambiar el nombre según el nombre del archivo. Una vez hecho todo el proceso con el resto de las primitivas, el árbol de definición debe quedar como el que aparece en siguiente figura.



A continuación se ha de validar la definición de la barra de herramientas. Para ello, se ha de hacer clic sobre “Tool Registry” con el botón derecho y seleccionar “Validate”.

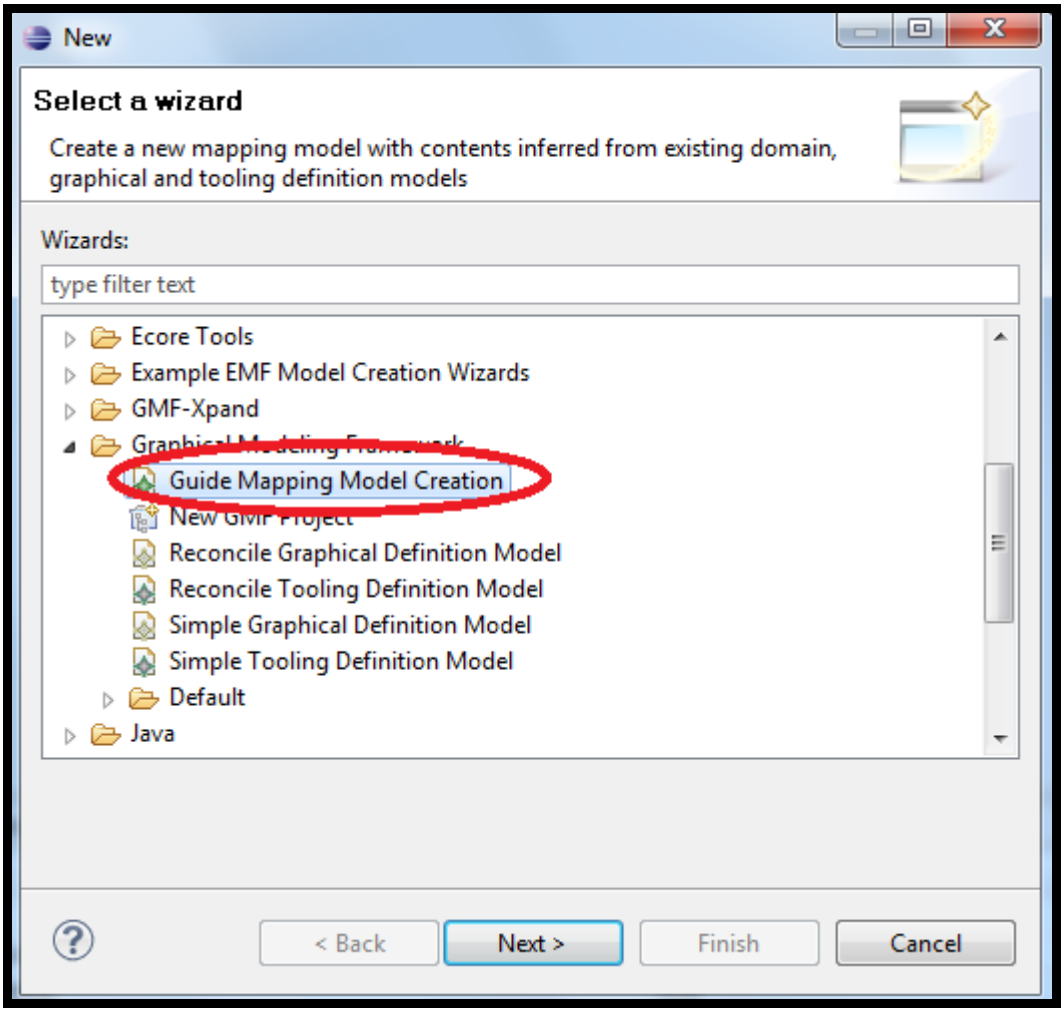


Si todo está bien aparecerá un mensaje como el de la siguiente figura.

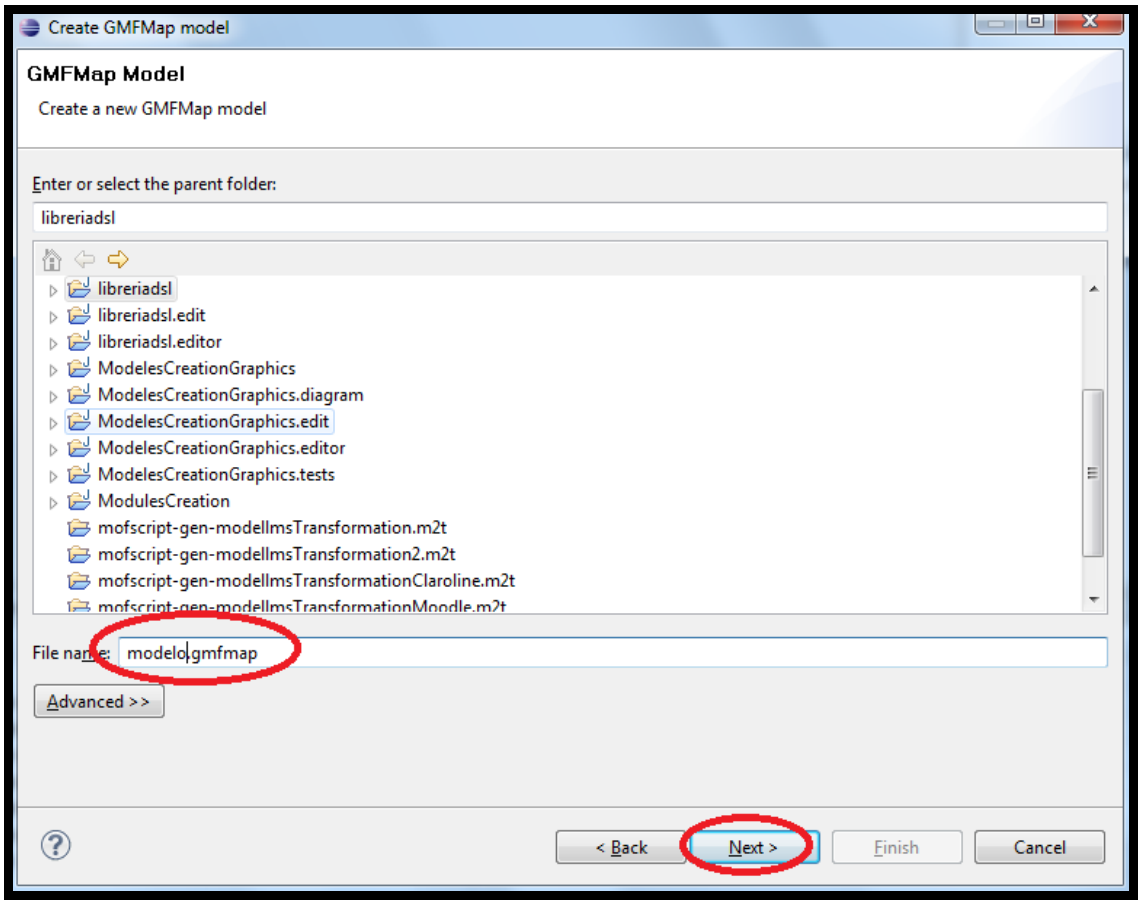


5.6 Mapping

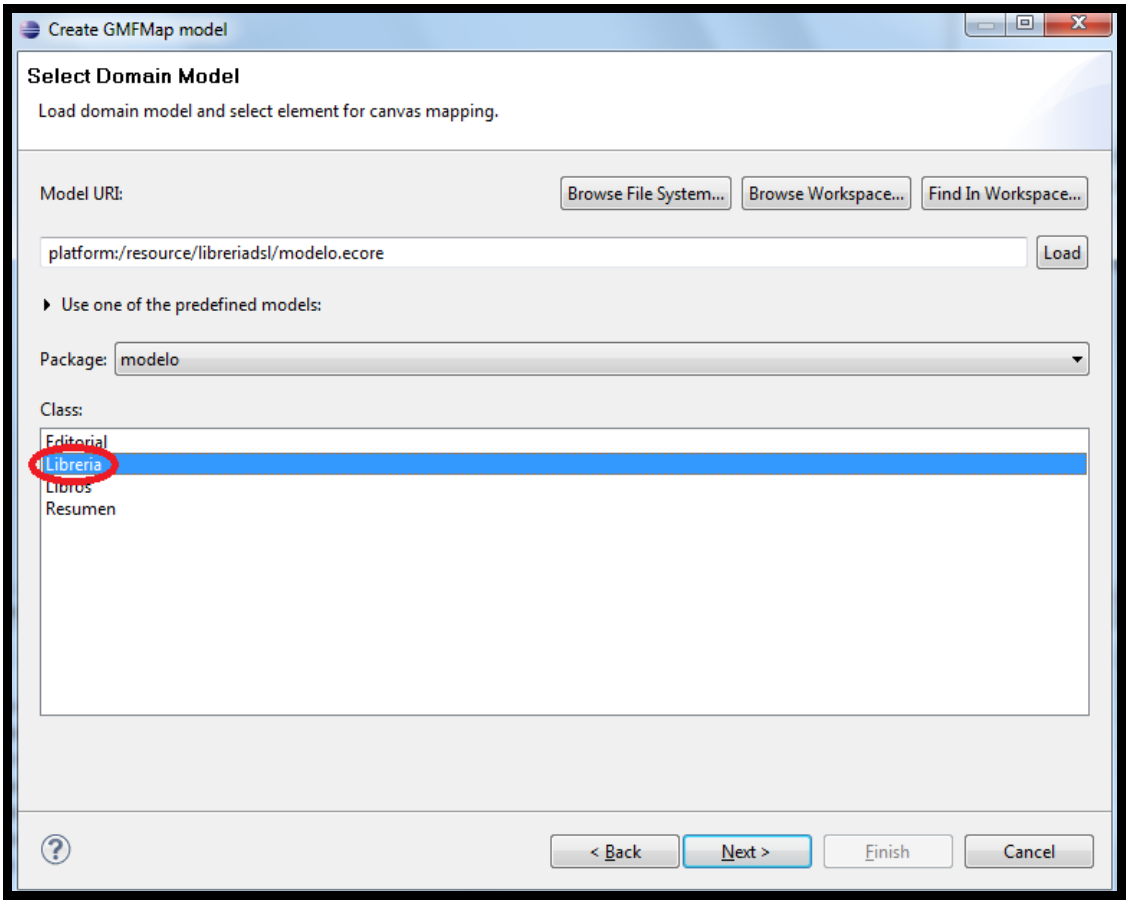
El proceso de “mapping” o correspondencia es aquel en el que todo lo que hemos creado cobra un sentido y se une para formar la herramienta de construcción de modelos específicos de dominio. Será en este paso donde conozcamos si toda la creación de esta herramienta tiene consistencia. Para comenzar, se ha de hacer clic en “File ->New -> Other” y en la carpeta “Graphical Modeling Framework” seleccionar “Guide Mapping Model Creation", como se muestra a continuación.



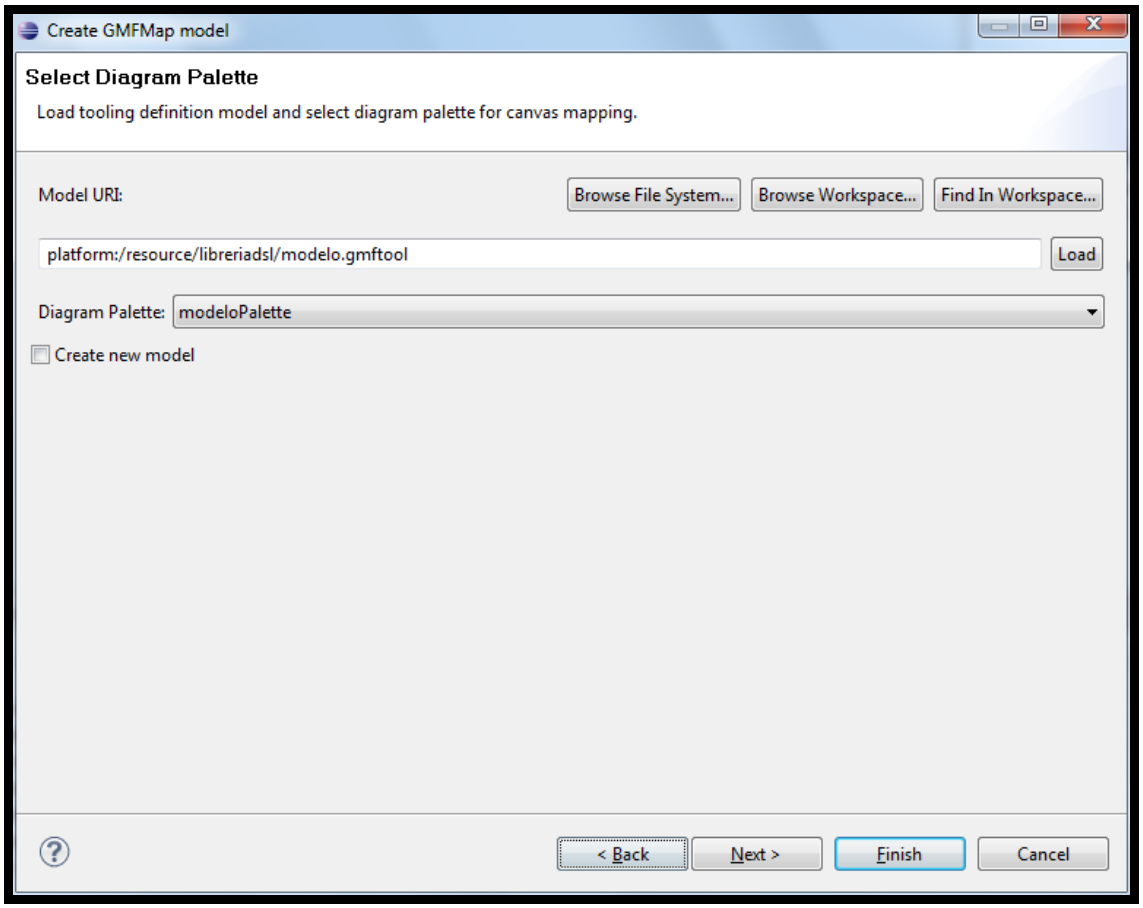
Seleccionamos “libreriadsl” como carpeta raíz y llamamos al documento “modelo.gmfmap” y pulsamos “Next”, como se muestra a continuación.



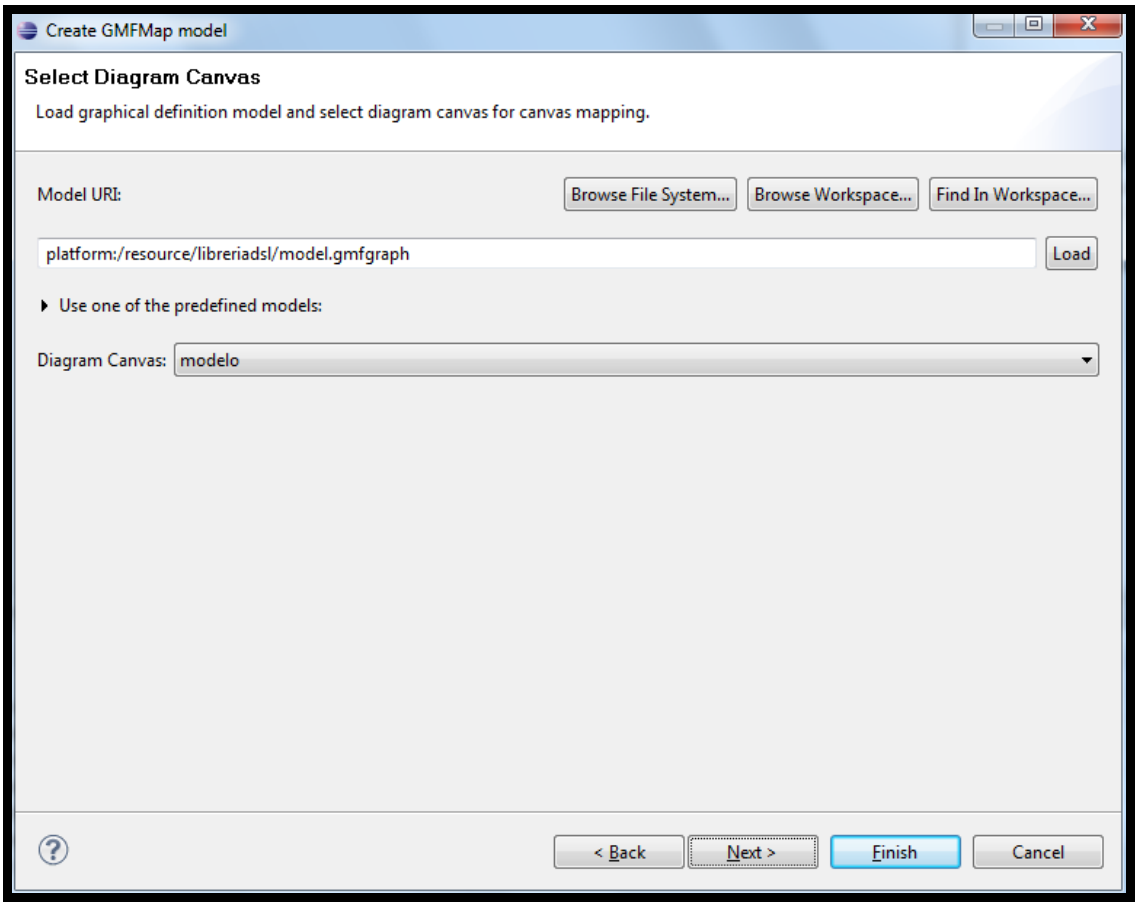
Seleccionamos “modelo.ecore” como modelo de entrada, si no aparece por defecto, pulsamos en “Find in Workspace” para seleccionarlo. Después, seleccionamos el elemento raíz de nuestro modelo, es decir, el que contiene al resto, en nuestro caso es “Libreria” y pulsamos Next, como se muestra a continuación.



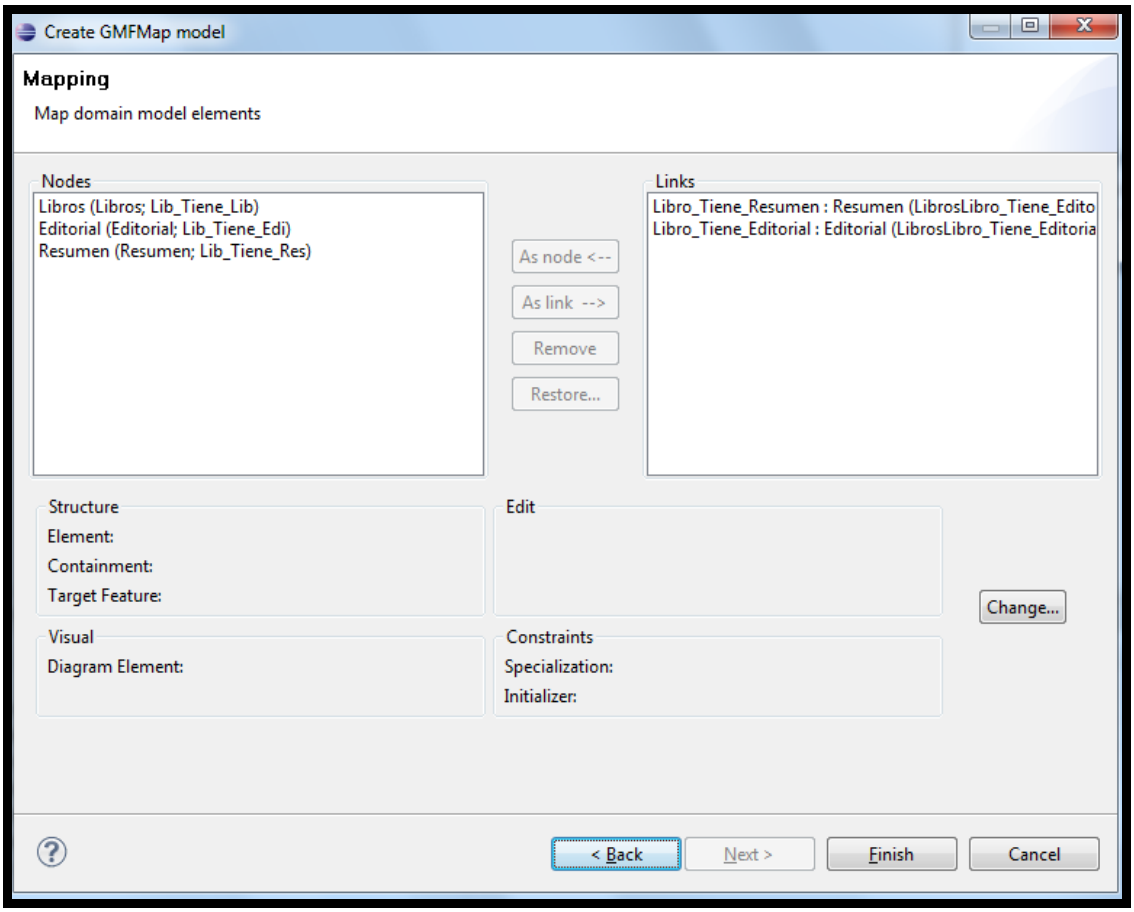
Seleccionamos el "modelo.gmftool", si no aparece por defecto, pulsamos en "Find in Workspace" para seleccionarlo y pulsamos "Next", como se ve a continuación.



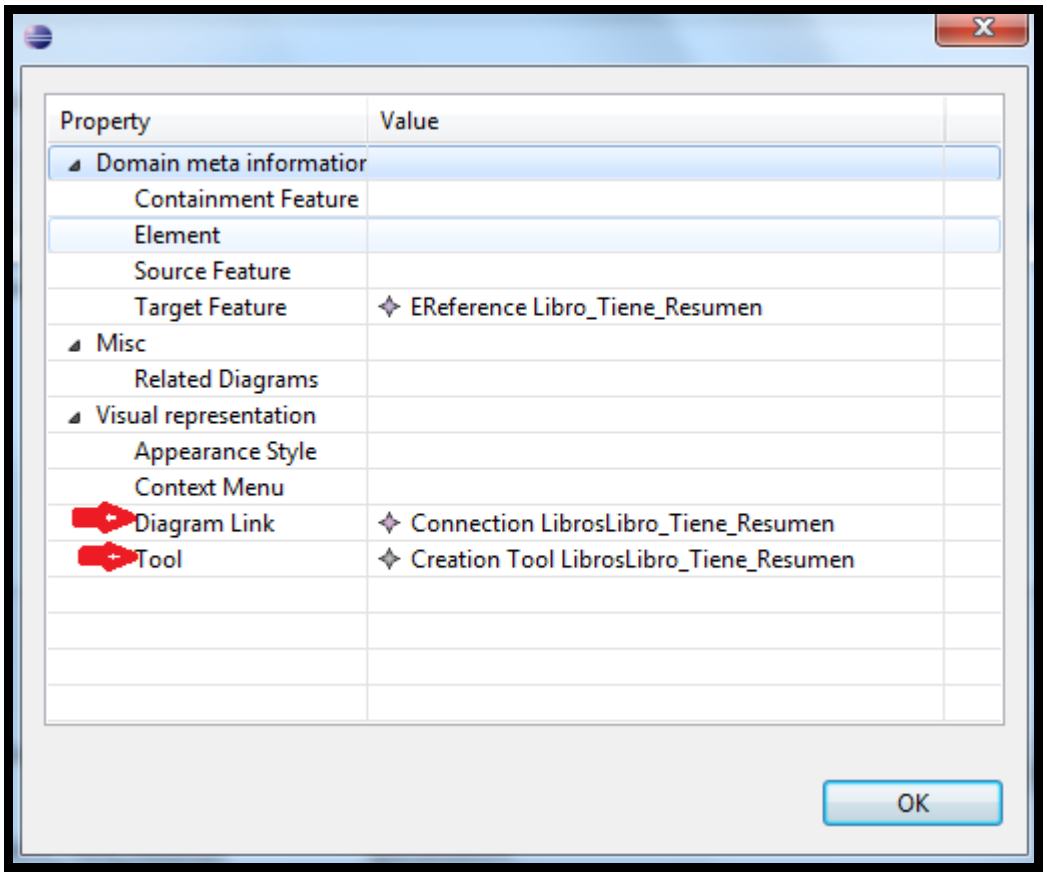
Seleccionamos “modelo.gmfgraph”, si no aparece por defecto, pulsamos en “Find in Workspace” para seleccionarlo, y pulsamos “Next”, como se muestra a continuación.



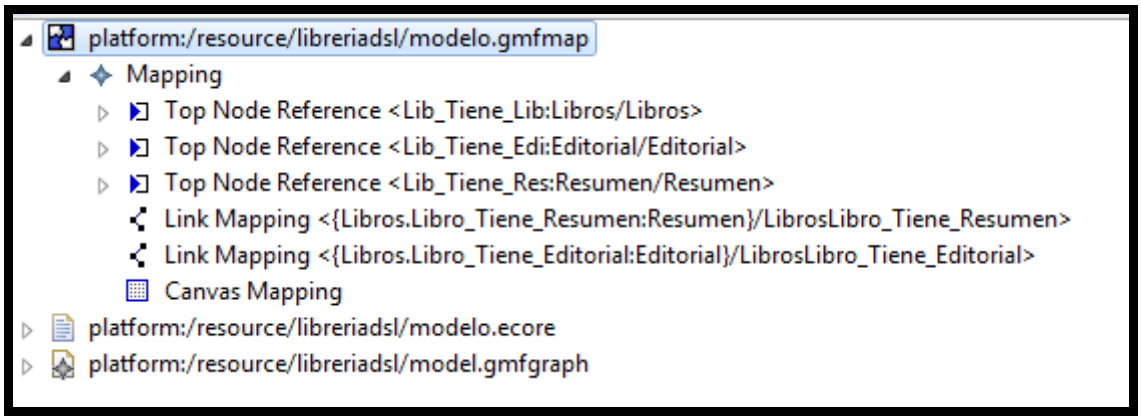
A continuación aparece una ventana como la de la siguiente figura, en la que se especifican los nodos y los enlaces (links).



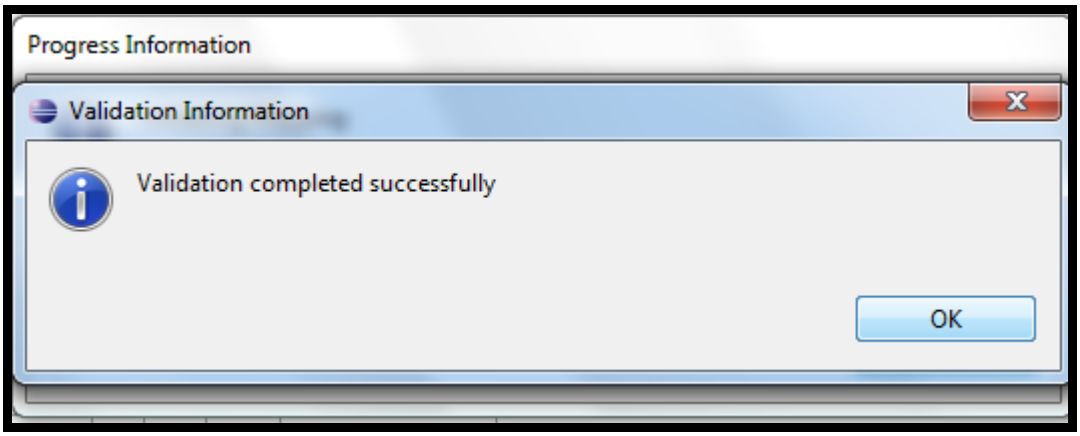
Marcamos “Libro_Tiene_Resumen” y pulsamos “Change”, cambiamos los valores de “Diagram Link” y “Tool” a las metáforas y herramientas que hemos definido para “Libro_Tiene_Resumen”, y “Libro_Tiene_Editorial”, también revisamos los nodos “Libros”, “Editorial” y “Resumen” por ultimo hacemos clic en “Finish”, como se muestra a continuación.



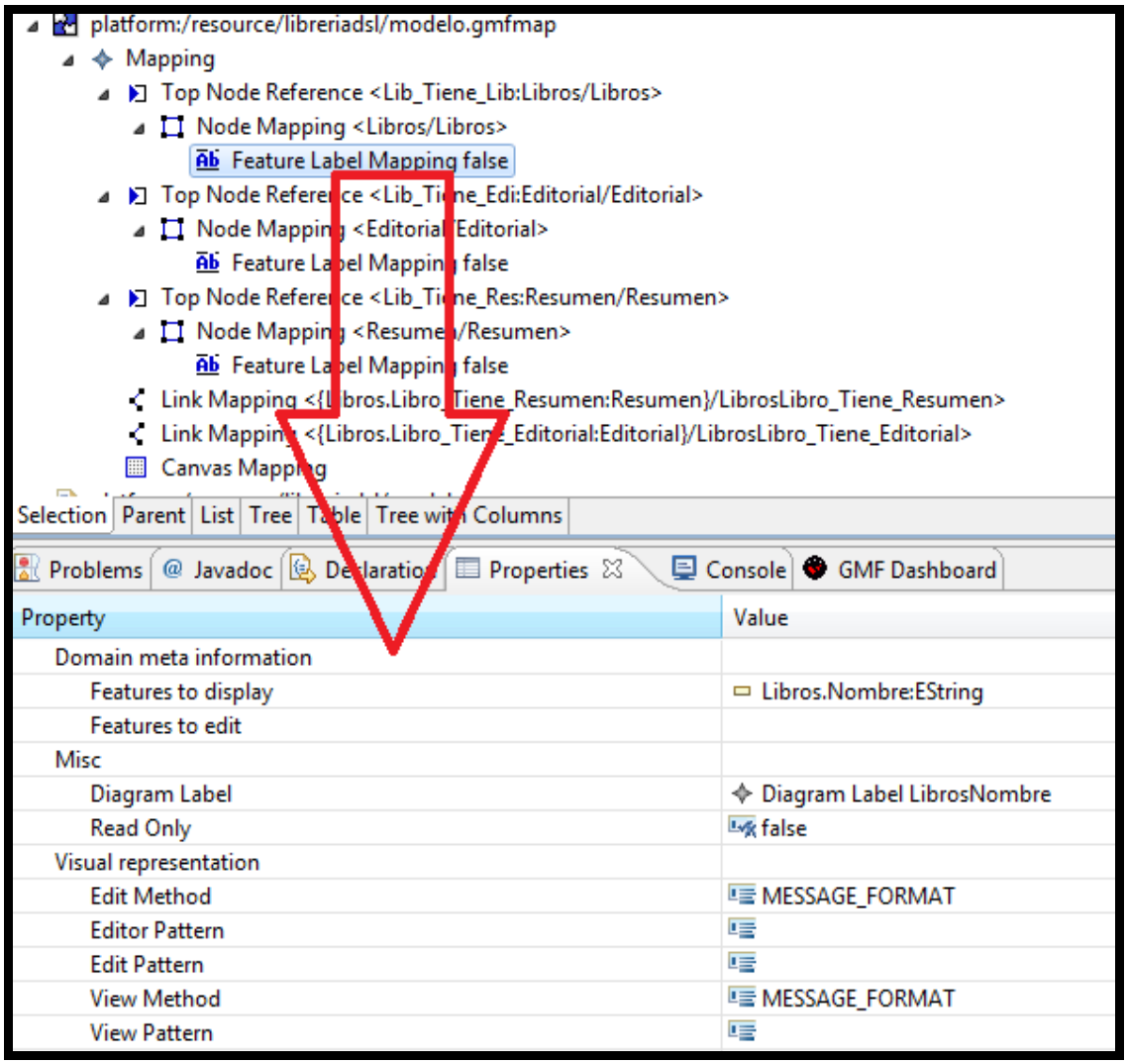
Una vez que hayamos pulsado “Finish” nos aparecerá el árbol de especificación del proceso de “mapping” como se ve en la siguiente figura.



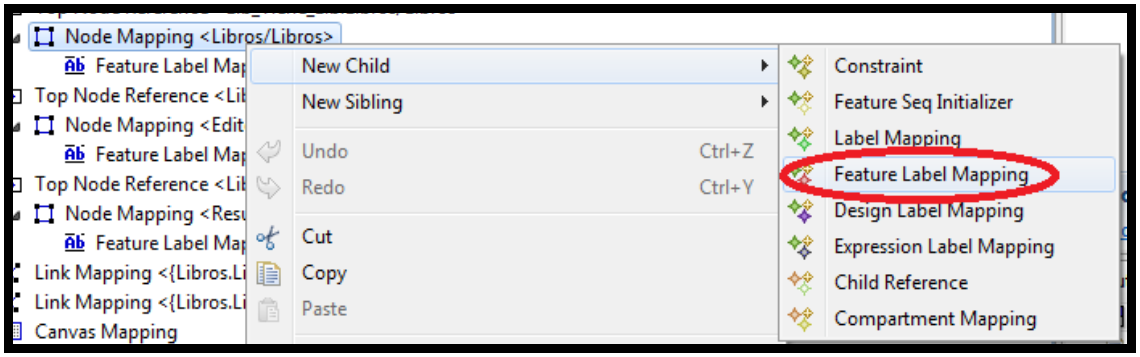
En este paso debemos comprobar que el “mapping” se ha realizado correctamente. Para ello, se ha de hacer clic sobre “Mapping” con el botón derecho y seleccionar “Validate”, si todo está bien aparecerá un mensaje de confirmación como el siguiente.



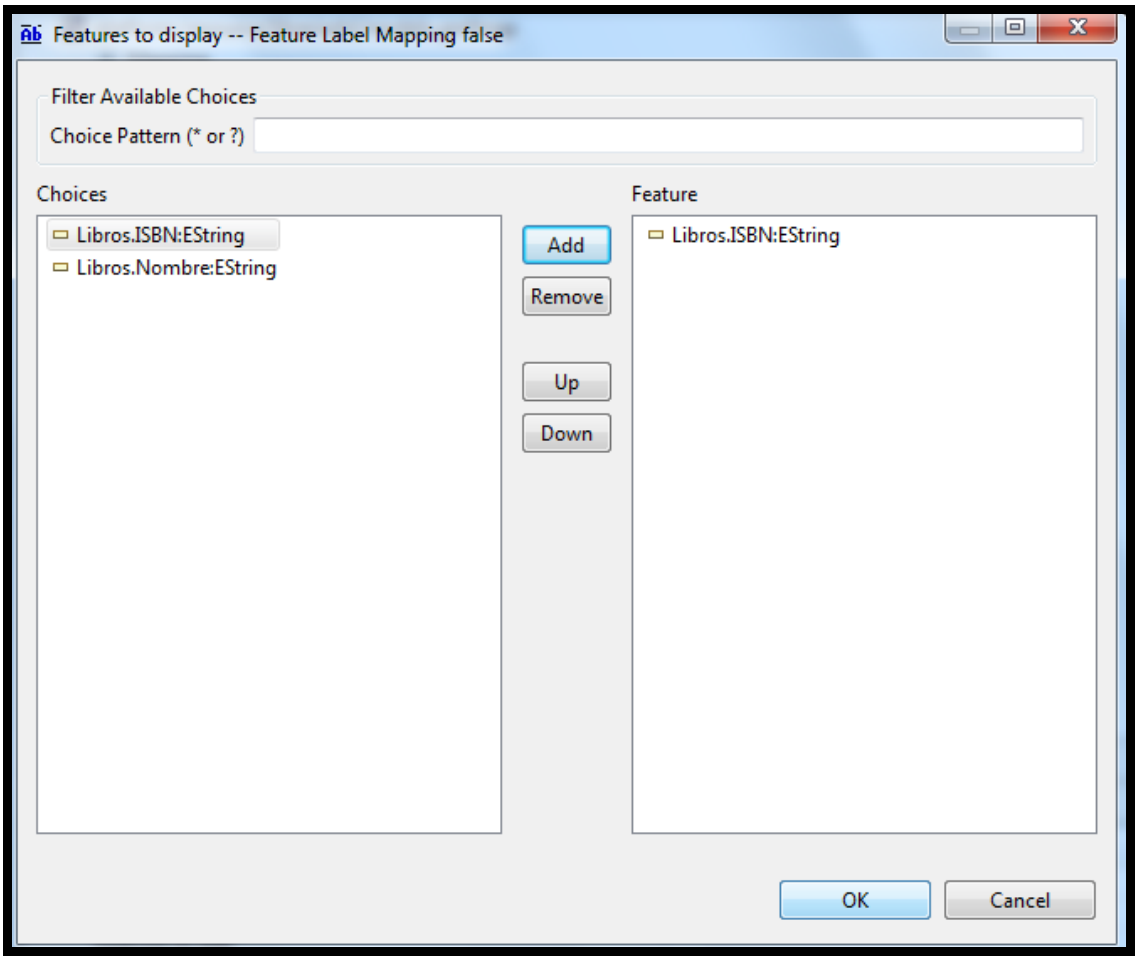
Como se puede observar en la siguiente figura, cada nodo cuelga de un nodo de referencia que indica qué relación existe entre la librería y el nodo en cuestión. Si nos fijamos un poco más en detalle, podemos darnos cuenta que muchas de las etiquetas que hemos declarado en el modelo no aparecen. Este es el momento de crearlas. Vamos a crear la etiqueta de ISBN o del libro, ya que so hacemos clic sobre el “Feature Label Mapping false” que aparece anidado dentro de “Node mapping <Libros/Libros>” veremos que solo hay un propiedad creada. Por lo tanto, hay que añadir la que nos falte, en este caso es ISBN, como se ve a continuación.



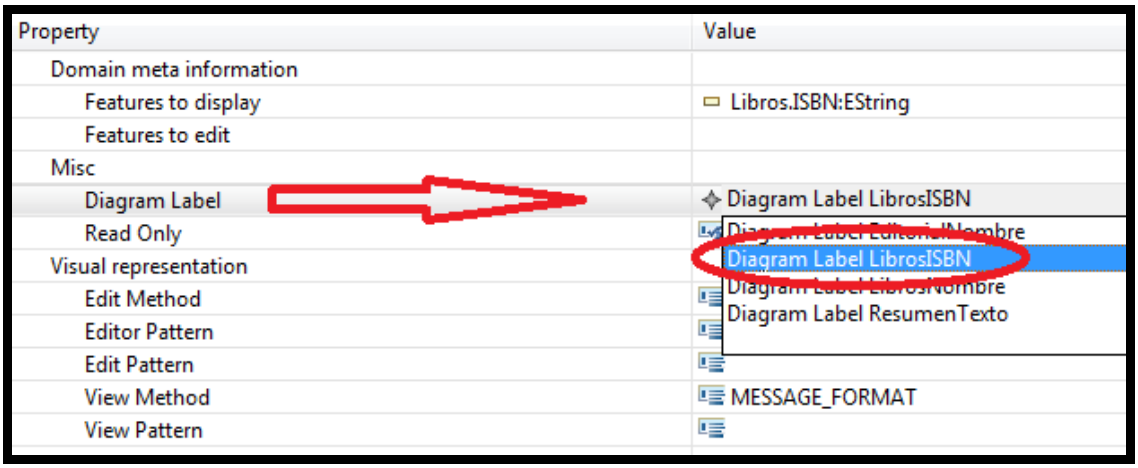
Para agregar la propiedad hacemos clic con el botón derecho sobre "Node mapping <Libros/Libros>" y seleccionamos "New Child -> Feature label mapping", como se a continuación.



Vamos a propiedades. En el campo "Features to Display" seleccionamos "ISBN" y pulsamos "add", continuación pulsamos OK. Como se muestra a continuación.



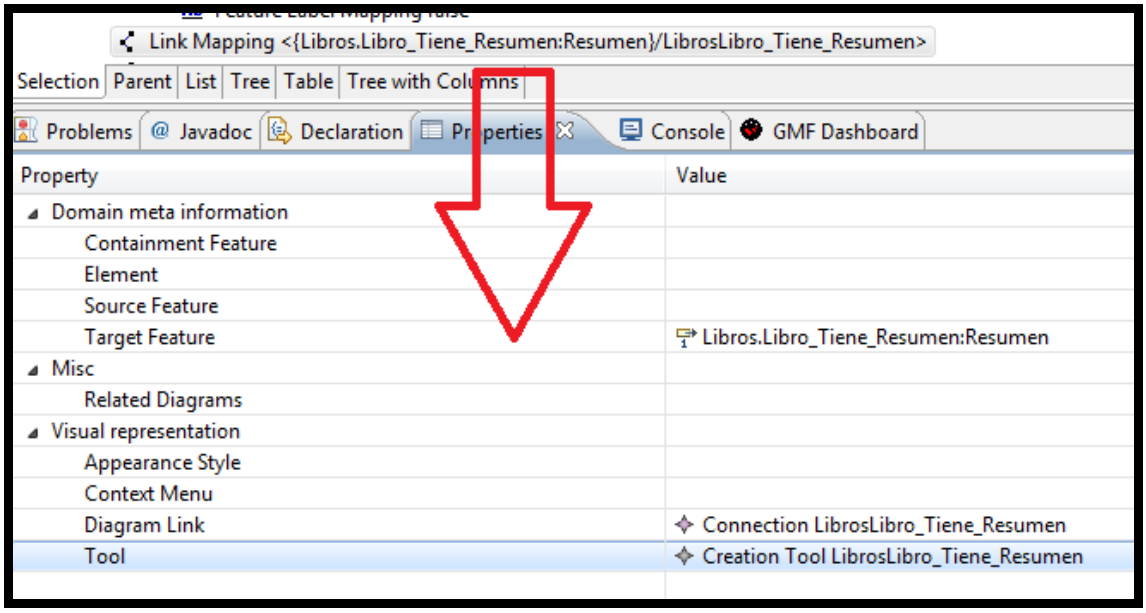
Luego cambiamos el campo “Diagram label” por la etiqueta que corresponda, en este caso “Diagram Label LibrosISBN”, como se a continuación.



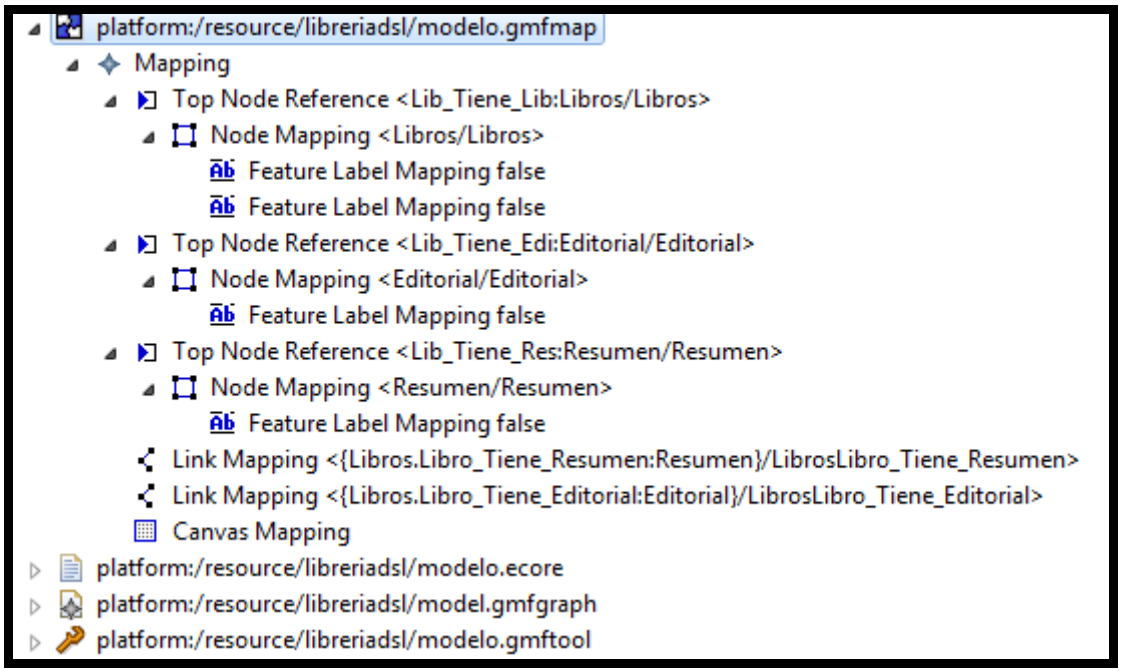
El proceso es similar para adicionar más propiedades de los labeles, esto se debe hacer cuando sea necesario y es casi siempre que hay más de un label. Para nuestro caso no es necesario, pues los demás label si aparecen.

A continuación hay que comprobar que cada nodo está asociado con su herramienta de creación y su componente gráfica. Veamos el caso de Libros:

Pulsamos con el botón derecho sobre el "Link mapping <{Libros.Libro_Tiene_Resumen:Resumen}/LibrosLibro_Tiene_Tesumen>", seleccionamos "Show Properties View", observamos si los campos "Diagram Link" y "Tool" coinciden con el enlace. Si no es así lo cambiamos. En el caso de "Libro_Tiene_Resumen" quedaría como se ve a continuación.

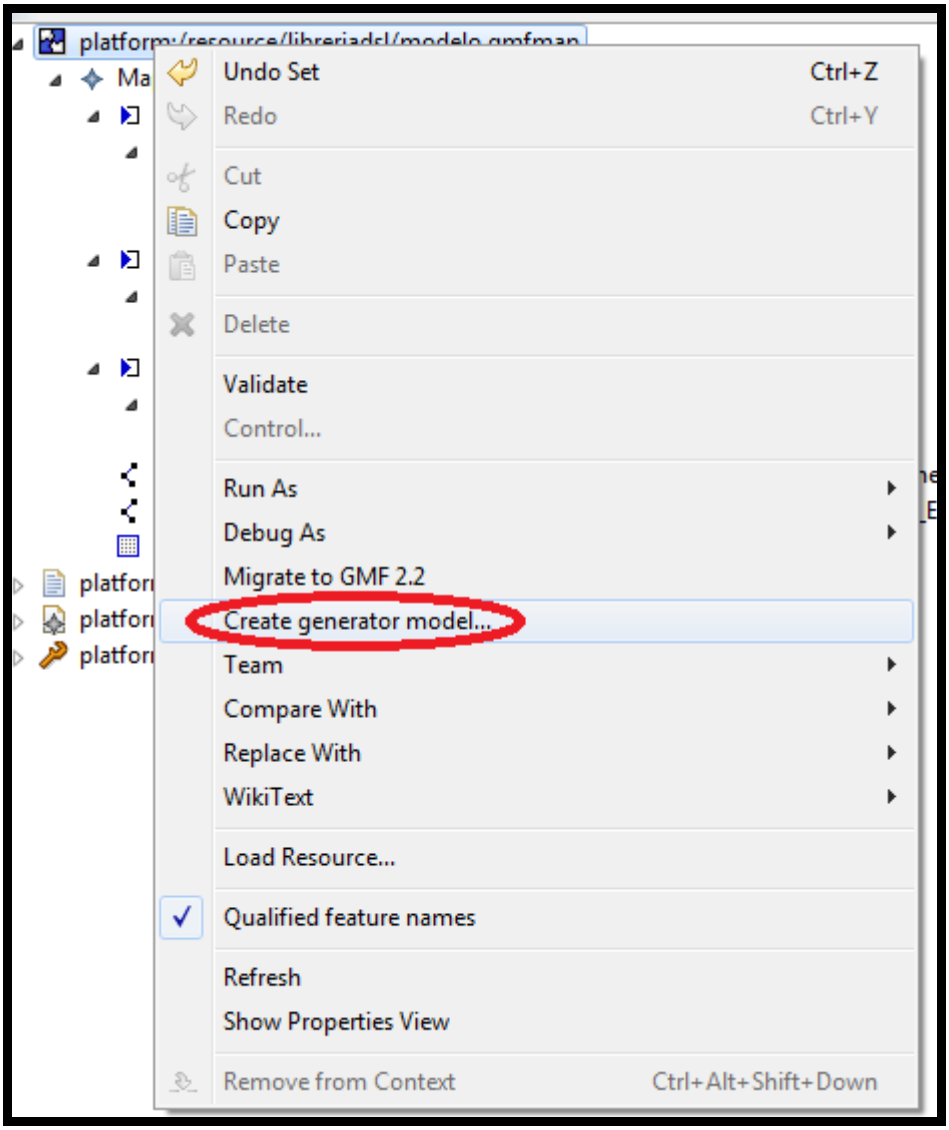


Este proceso se debe realizar con todos los nodos y los enlaces. Una vez completado se ha de proceder a la validación del modelo de correspondencias. Validamos el proceso del mismo modo que se hizo anteriormente. El árbol desplegado debe quedar como se muestra a continuación.

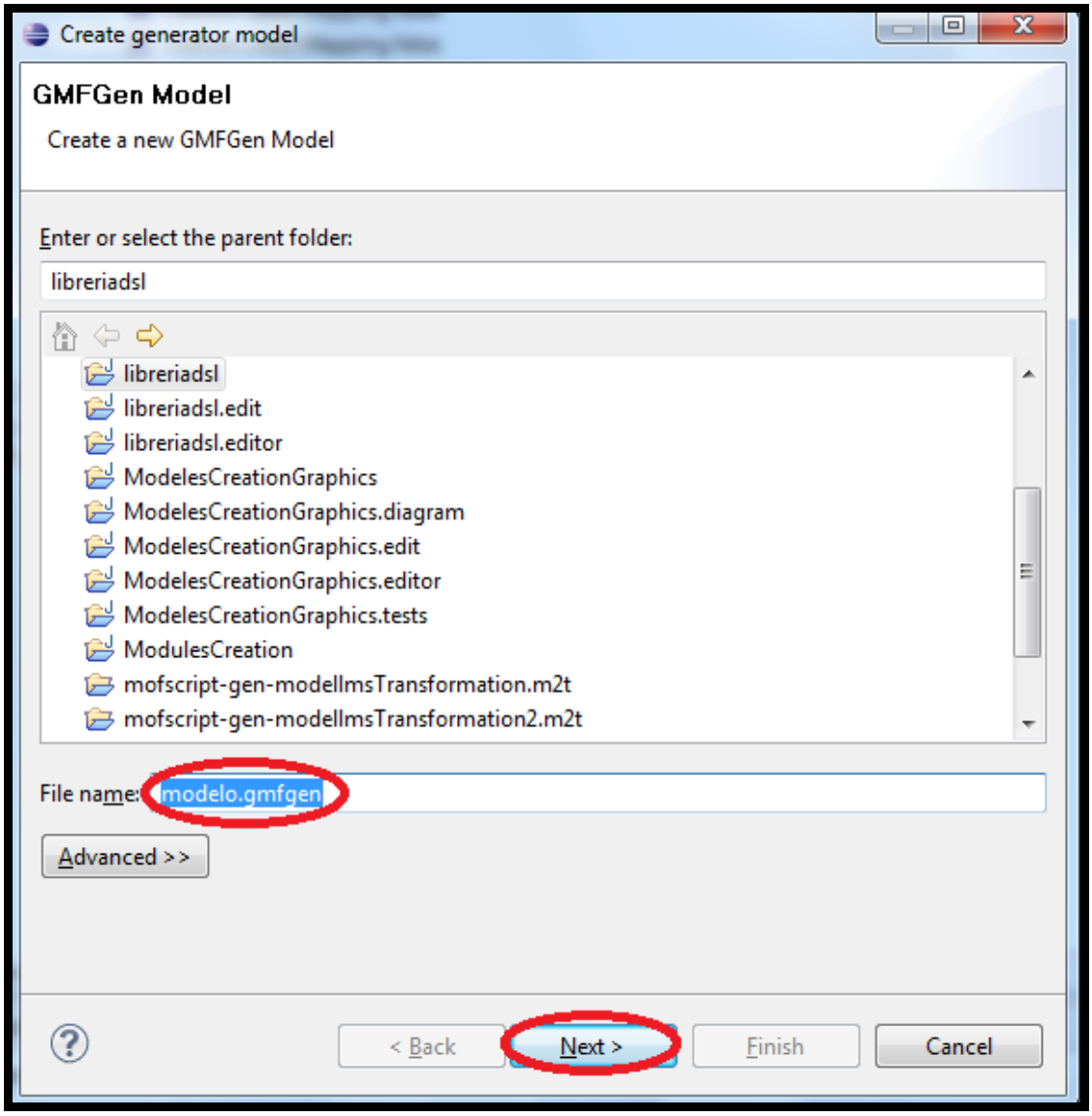


5.7 Generación de código

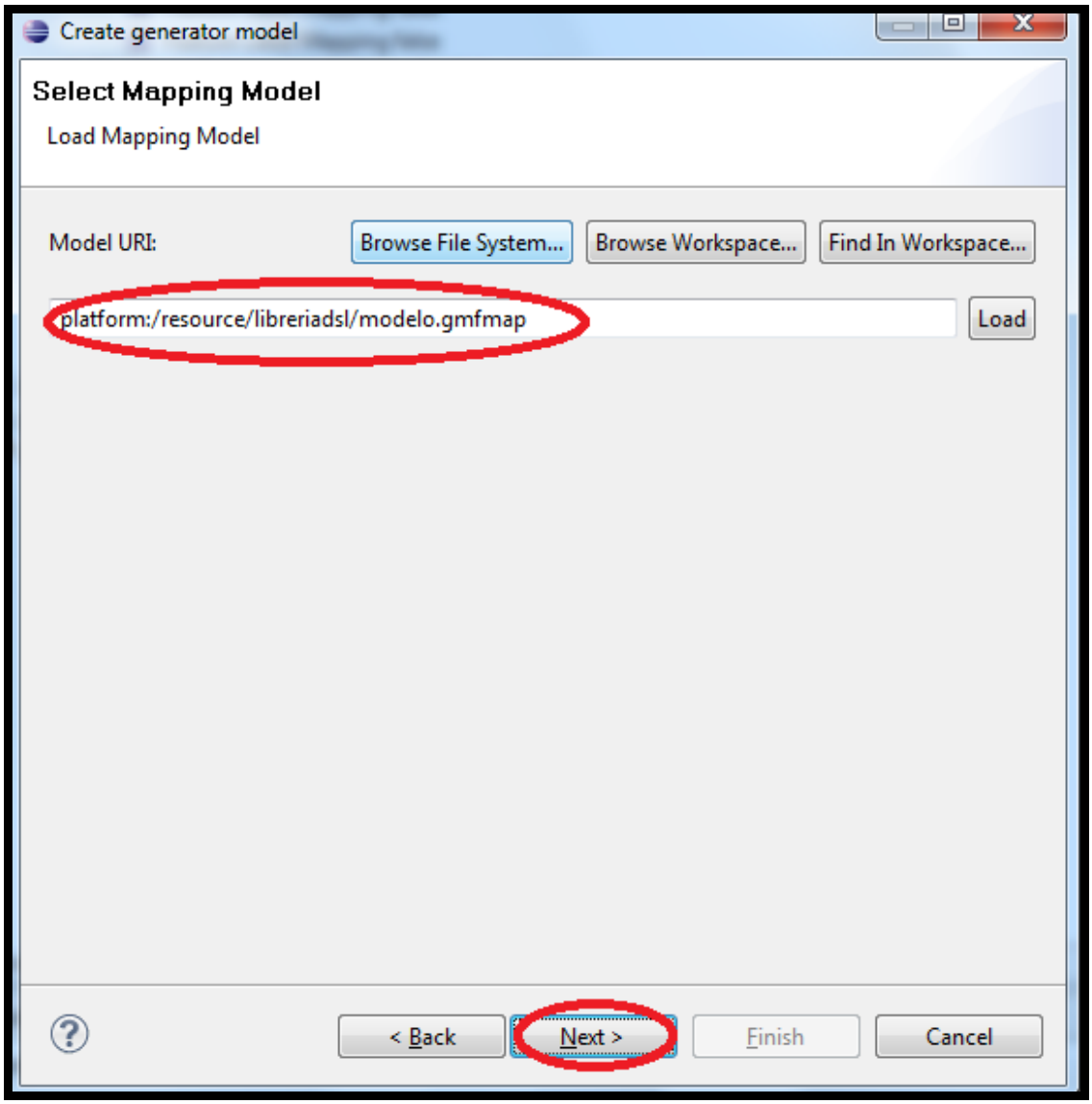
Una vez acabado el “mapping”, ya está el proceso de definición de nuestra herramienta completado. Ahora solamente falta generar su código, para poder ser ejecutada. Para poder generar el código, debemos tener un generador. Este elemento nos lo proporciona el “modelo.gmfmap”. Para conseguirlo debemos hacer clic con el botón derecho sobre la raíz del árbol de “mapping” y seleccionar “Create Generator Model….”, como se ve a continuación.



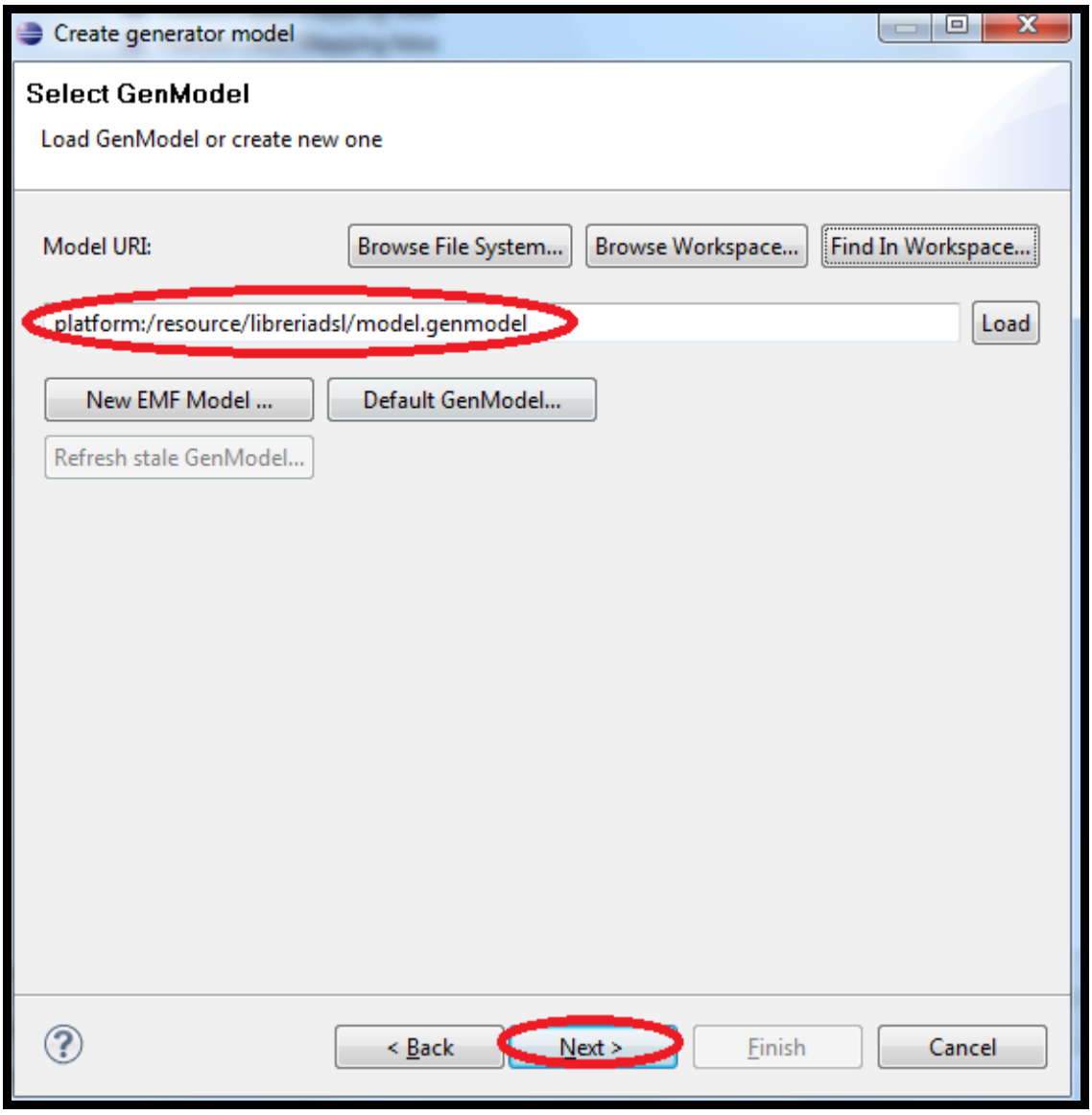
Introducimos el nombre del generador, en nuestro caso “modelo.gmfgen” y pulsamos “Next”, como se muestra a continuación.



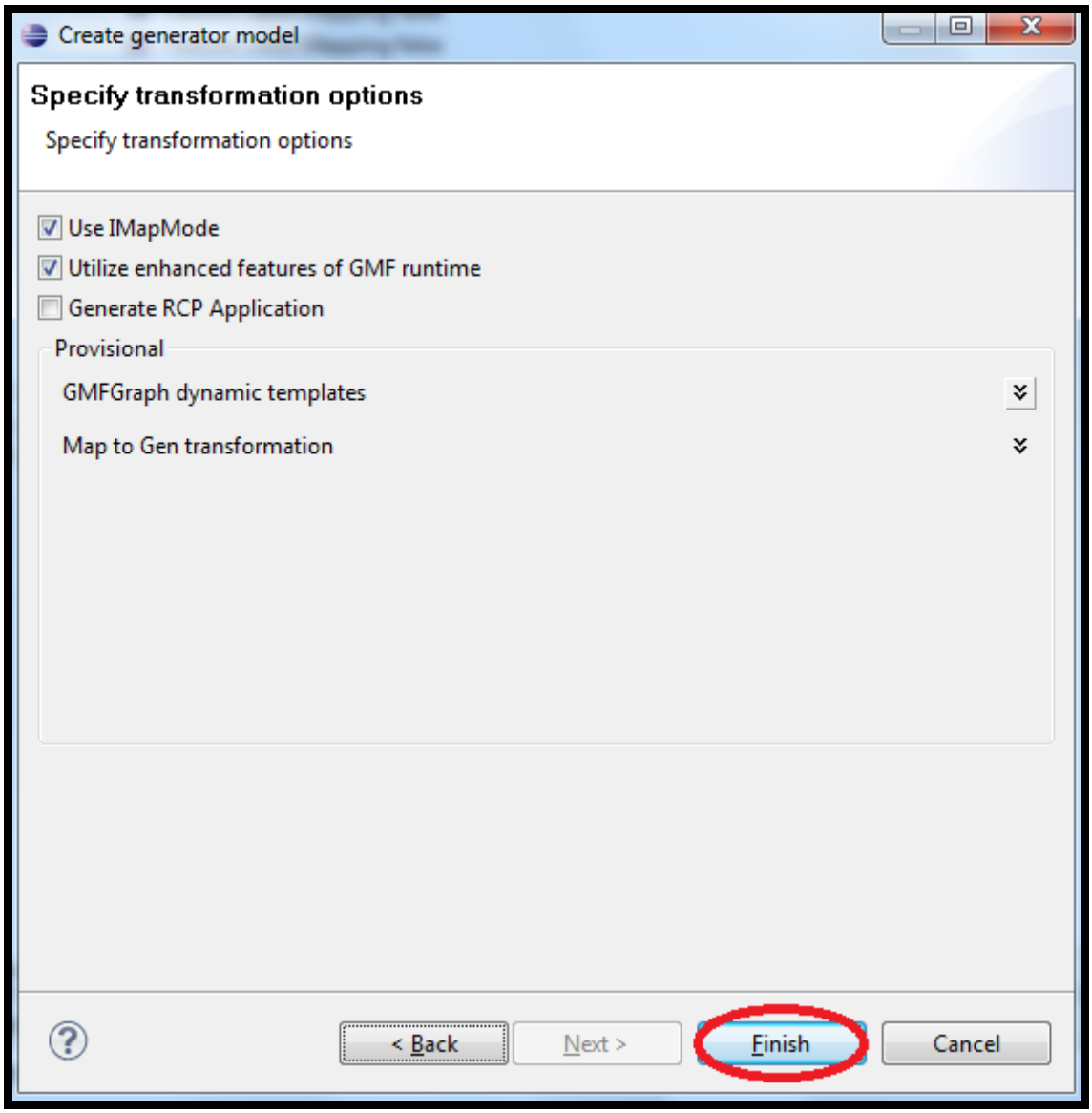
Seleccionamos “modelo.gmfmap”, si no aparece por defecto, pulsamos en “Find in Workspace” para seleccionarlo, pulsamos “Next”, como se mustr5a a continuación.



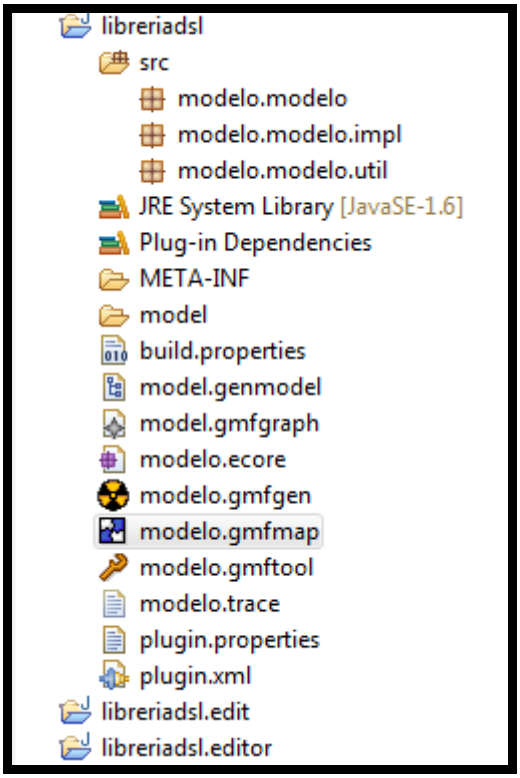
Seleccionamos “modelo.genmodel”, si no aparece por defecto, pulsamos en “Find in Workspace” para seleccionarlo, pulsamos Next, como se ve a continuación.



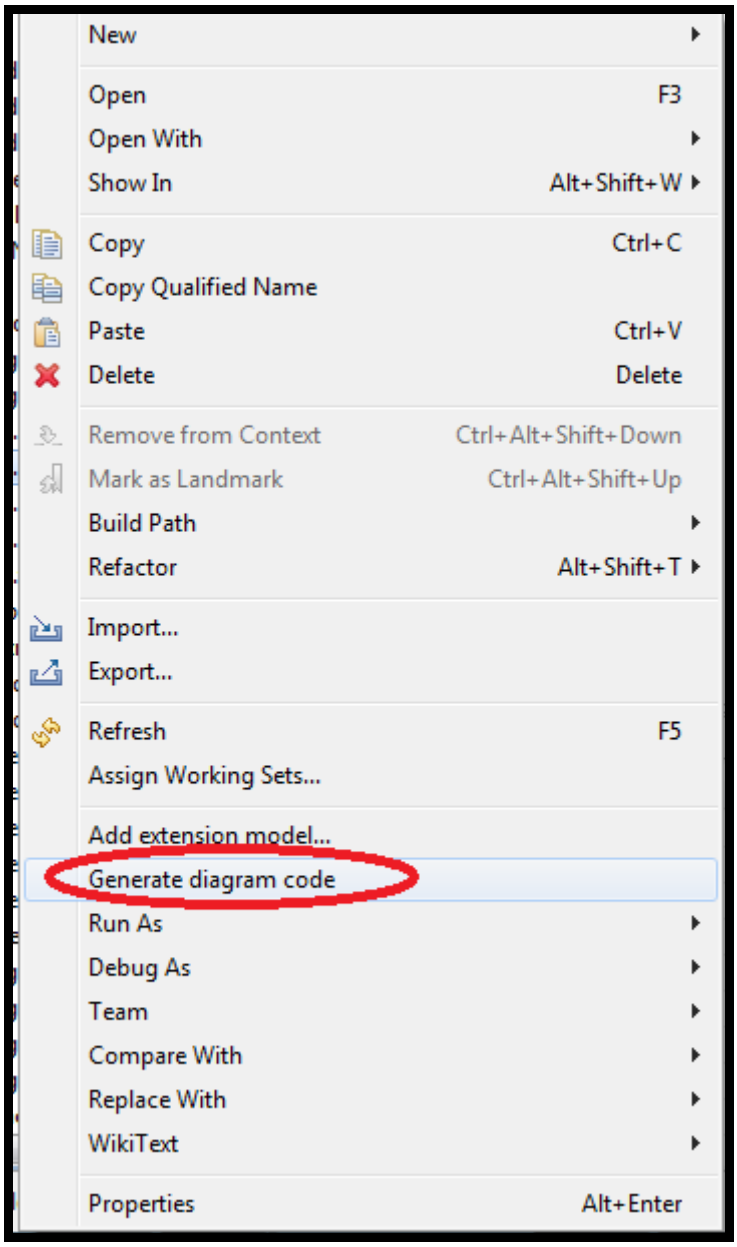
Por último hacemos clic en “Finish”, como se ve a continuación.



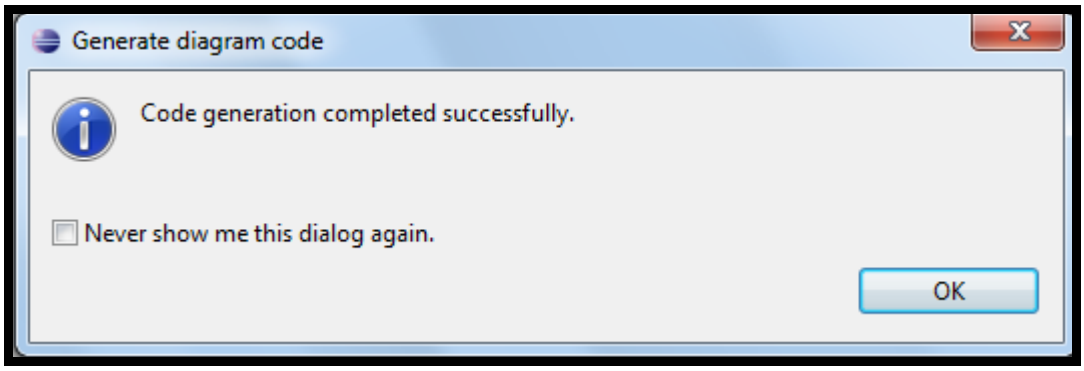
El explorador de paquetes debe quedar cómo se muestra a continuación.



Finalmente, pulsamos con el botón derecho sobre “modelo.gmfgen” y seleccionamos “Generate Diagram Code”, como se muestra a continuación.



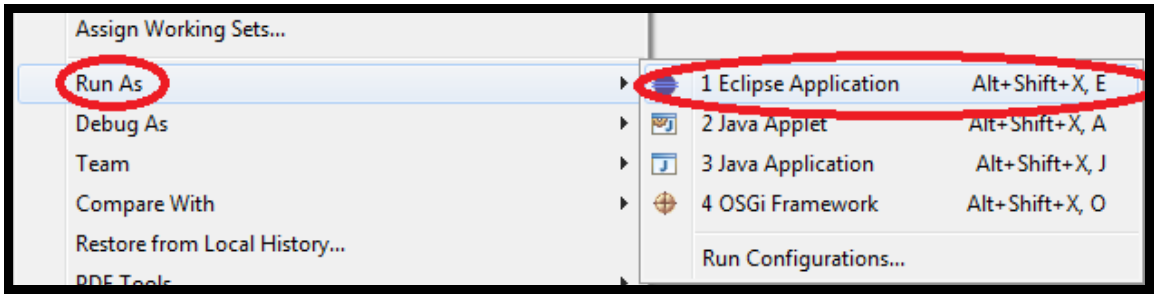
Si se ha generado correctamente aparecerá un mensaje de confirmación. Como se ve continuación.



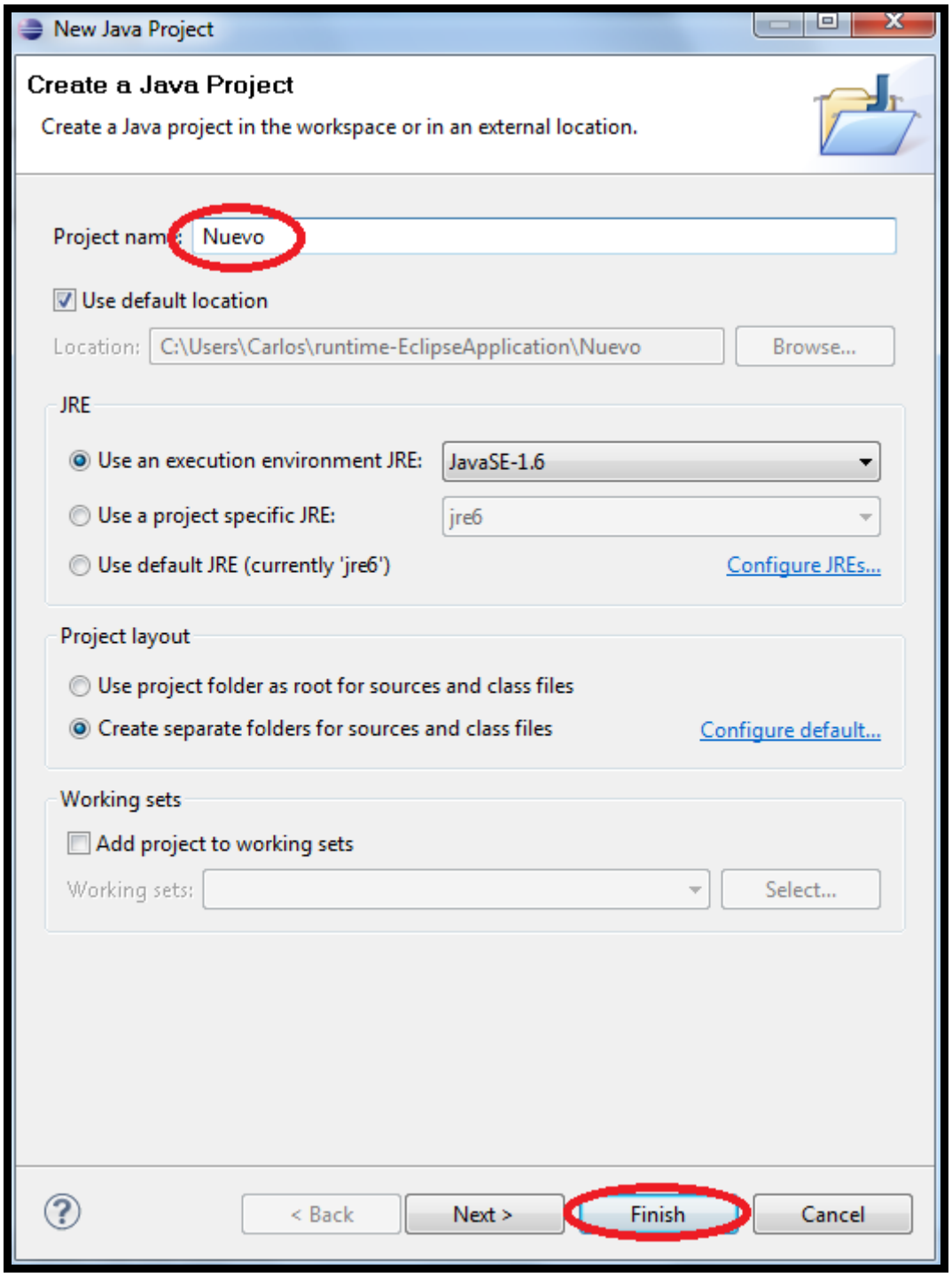
Y Se genera una carpeta con el nombre “libreriadsl.diagram”.

5.8 Ejecución de la herramienta de creación de diagramas

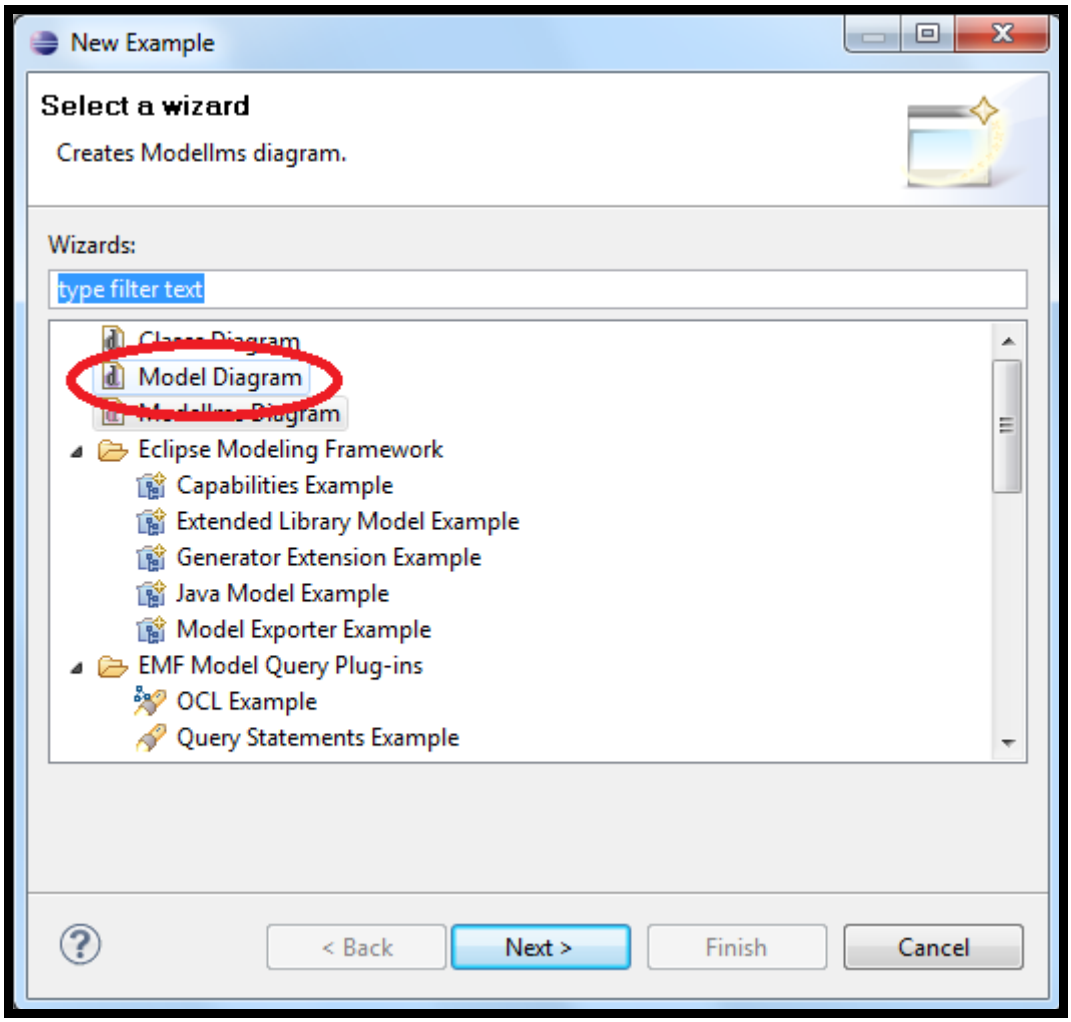
En este punto, ya disponemos de nuestra aplicación. Para ejecutarla, hay que seleccionar la carpeta “libreriadsl.diagram” y pulsar sobre ejecutar (flecha verde de la barra de herramientas). Seleccionar “Run as ->Eclipse Application” o haciendo clic derecho sobre la carpeta “libreriadsl.diagram” y seleccionado “Run As -> Eclipse Application”, como se muestra a continuación.



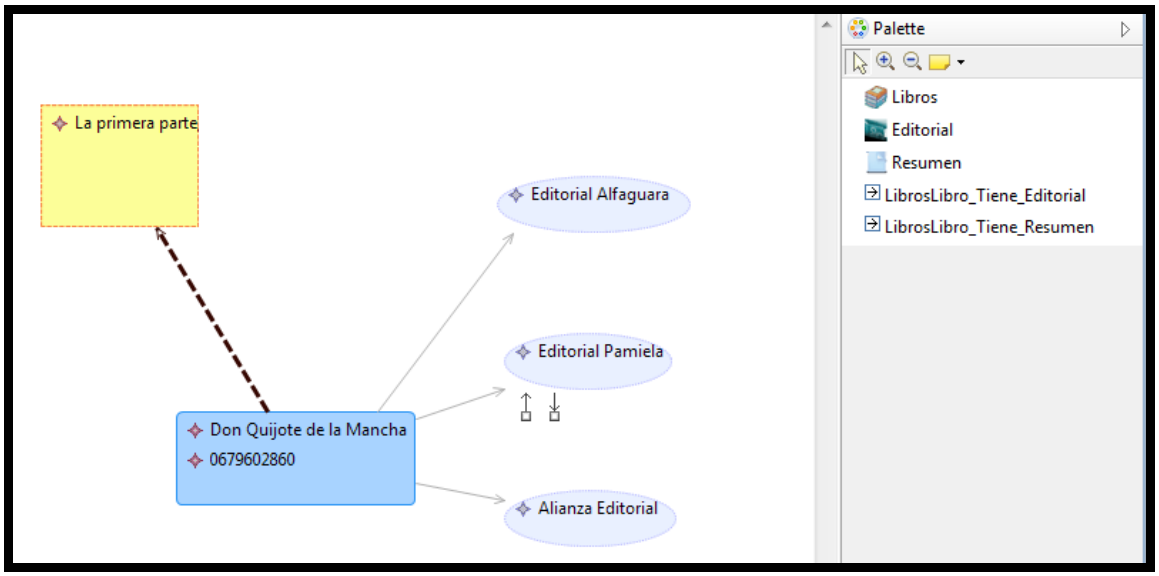
Se nos abrirá una nueva ventana de Eclipse. Creamos un nuevo proyecto “java File -> New -> Java Project”. Lo llamamos Nuevo y pulsamos Finish, como se muestra a continuación.



Ahora pulsamos con el botón derecho sobre “Nuevo” y seleccionamos “New -> Example -> “Model Diagram”. Asignamos los nombres que queramos a los nuevos documentos, uno será el diagrama y otro el fichero asociado con el lenguaje específico de dominio.



Nuestra aplicación debería tener el aspecto como el que se muestra a continuación, en la que se ha modelado un Libro con un resumen y tres editoriales.



Nota: Si desea realizar modificaciones sobre cualquier elemento, estas se pueden realizar en el fichero ".gmfgraph", las guarda, valida el modelo de nuevo, para que los cambio apliquen, debe borrar el fichero ".gmfgen" y la carpeta que se genero a partir de este ".diagram", por ultimo volver a generar ".gmfgen" validarlo y generar él ".diagram". Con esto usted podrá ejecutar nuevamente la herramienta y visualizará los cambios.

8 Anexo III

Guía pruebas

Carlos Enrique Montenegro Marín

Guía pruebas sobre Moodle

Carlos Enrique Montenegro Marín

Guía de Pruebas para una herramienta que realiza un modelado específico de dominio para la construcción de módulos de LMS independientes de la plataforma.

Objetivo: Probar el rendimiento de utilizar la herramienta que realiza un modelado específico de dominio para la construcción de módulos de LMS independientes de la plataforma, contra la creación de estos módulos manualmente. Las pruebas evaluaran rendimiento y esfuerzo del profesor en la creación de módulos para la plataforma.

Como MDA plantea la creación de aplicaciones partiendo desde un dominio específico hasta poder desplegar el modelo creado en una plataforma especifica. El despliegue se realizara bajo la plataforma LMS Moodle 1.9.

Partiendo del caso hipotético; en el cual un profesor desea impartir una asignatura apoyándose en una plataforma virtual y bajo las siguientes consideraciones:

- La asignatura consta de cinco temas.
- Por cada uno de los temas se desea crear un foro, un chat, un anuncio, una wiki, y una noticia.
- Cada vez que creen las herramientas de un tema, se quiere enviar una nota a un estudiante en particular y luego a todos los inscritos en el curso, informado que "un nuevo tema está disponible en la plataforma virtual".

La medición de tiempos se realizara para los dos ambientes y se tomara:

- Tiempo en la creación de cada herramienta por tema.
- Tiempo en la creación de todas las herramientas que conforman un tema.
- Tiempo total en la creación de los cinco temas acompañados de sus herramientas.

Nota: Se considera que el envió de las notas a los inscritos también es una herramienta del tema.

La medición de esfuerzo mide la cantidad de ocasiones que el usuario tuvo que seleccionar o ingresar algún tipo de dato al sistema, y se realizara para:

- Esfuerzo en la creación de cada herramienta por tema.
- Esfuerzo en la creación de todas las herramientas que conforman un tema.
- Esfuerzo total en la creación de los cinco temas acompañados de sus herramientas.

Nota: Se considera que el envió de las notas a los inscritos también es una herramienta del tema.

Para poder comparar los tiempos es necesario que en la herramienta DSL, se cree el modulo y a continuación su enlace, **NO** crear todos los nodos y luego todos los enlaces.

Como la prueba busca medir el tiempo y nivel de esfuerzo en agregar cada modulo, se le proporcionaran los datos al usuario para ingresar:

Chat	Name	Chat
	Text	Welcome to Chat
Forum	Name	Forum
	Text	Welcome to Forum
Wiki	Name	Wiki
	Text	Welcome to Wiki
Announcement	Text	New topic
News	Name	New topic
	Text	New topic available
Notes student1@uniovi.es	Send to	student1@uniovi.es
	Text	New topic available
Notes All Users	Text	New topic available

Guía para el profesor en la creación de herramientas de comunicación en moodle.

1. Ingresar a la plataforma con su usuario y contraseña.
2. Seleccione el curso a trabajar.
3. Cambie la opción del curso a editable, para ello de clic sobre el botón “Turn editing on” que está ubicado en la parte superior derecha de la ventana, este botón debe cambiar a “Turn editing off”, ver Figura 1.

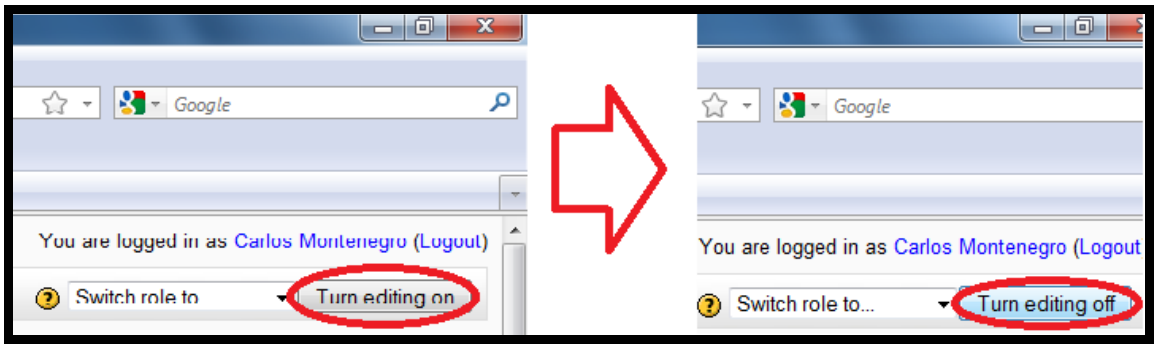


Figura 1.

4. **Creación de Chat, Forum ó Wiki.**

Despliegue el ComboBox que dice “Add an activity” ubicado en cada uno de los temas y selecciones la opción que desea agregar al tema del curso Chat, Forum ó Wiki, Figura 2.

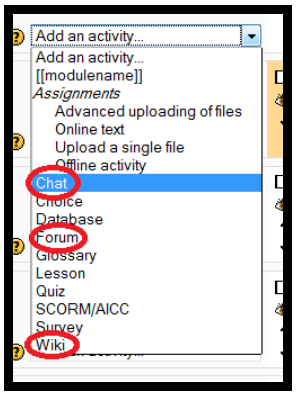


Figura 2.

a. Si usted selecciono Chat, vera un formulario como el de la Figura 3, los campos marcados con rojo y * son obligatorios, llénelos y de clic sobre el botón “Save and return to course”, esto lo regresara nuevamente al curso y usted debe poder visualizar el chat creado, Figura 4.

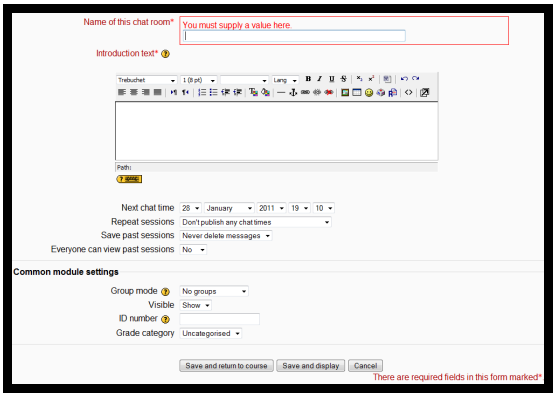


Figura 3.

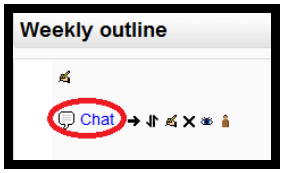


Figura 4.

b. Si usted selecciono Forum, vera un formulario como el de la Figura 5, los campos marcados con rojo y * son obligatorios, llénelos y de clic sobre el botón “Save and return to course”, esto lo regresara nuevamente al curso y usted debe poder visualizar el foro creado, Figura 6.

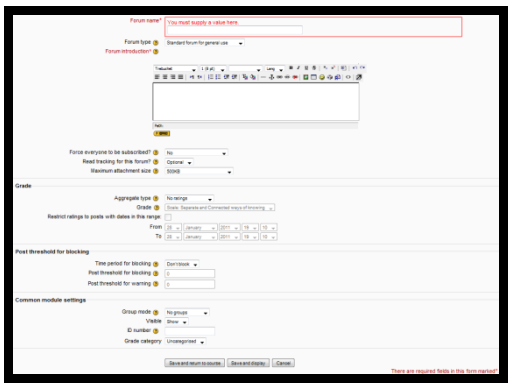


Figura 5.

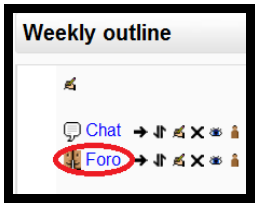


Figure 6.

c. Si usted selecciono Wiki, vera un formulario como el de la Figura 7, los campos marcados con rojo y * son obligatorios, llénelos y de clic sobre el botón “Save and return to course”, esto lo regresara nuevamente al curso y usted debe poder visualizar el wiki creado, Figura 8.

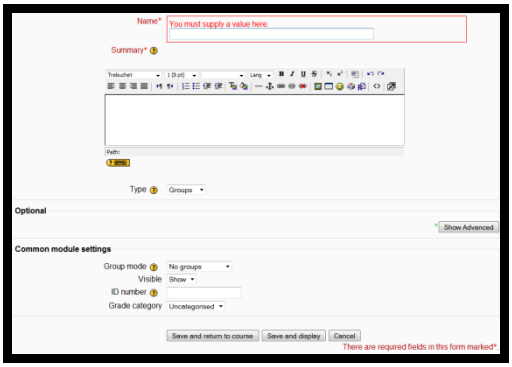


Figura 7.

Figure 8.

5. Creación de Announcement.

Despliegue el ComboBox que dice “Add a resource” ubicado en cada uno de los temas y selecciones la opción “Insert a label” Figura 9.

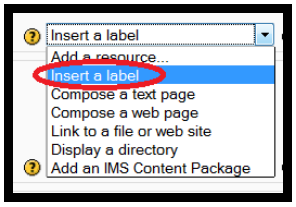


Figura 9.

Se mostrara un formulario como el de la Figura 10, los campos marcados con rojo y * son obligatorios, llénelos y de clic sobre el botón “Save and return to course”, esto lo regresara nuevamente al curso y usted debe poder visualizar el Announcement creado, Figura 11.

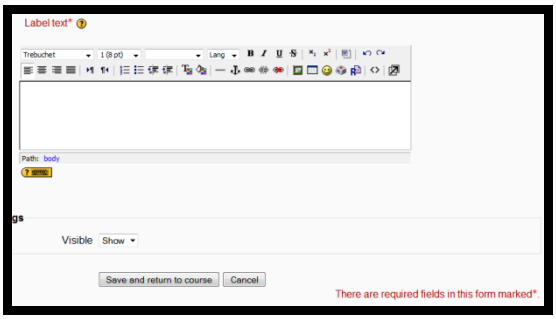


Figura 10.

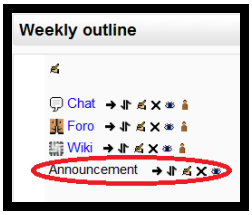


Figura 11.

6. Creación de News.

En la parte superior derecha de la venta existe Widget con el título “Latest News”, de clic sobre “Add new topic …”, Vea Figura 12.

Figura 12.

Se mostrara un formulario como el de la Figura 13, los campos marcados con rojo y * son obligatorios, llénelos y de clic sobre el botón “Post Forum”, espere unos segundos, esto lo regresara nuevamente al curso y usted debe poder visualizar la News creada, Figura 14.

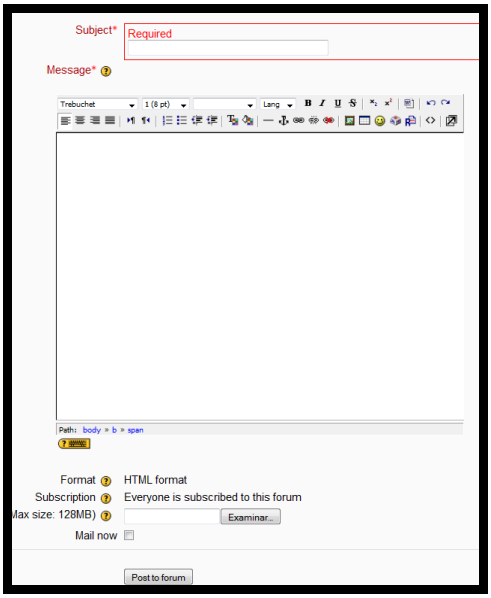


Figura 13.

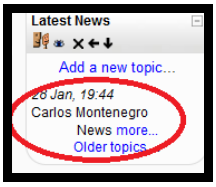


Figura 14.

7. Envió de Notes.

Las Notes en moodle son llamadas messages, y para poder enviar uno, usted debe ingresar a la opción “Participants” del Wiget “People”, ubicado en la parte superior derecha de la ventana, Figura 15.

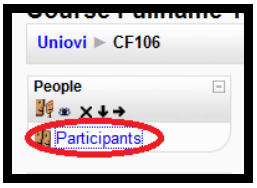


Figura 15.

Se mostrara un formulario con las personas inscritas en esta asignatura, usted deberá seleccionar a la persona que desea enviar el mensaje haciendo clic sobre su nombre, Figura 16.

Figura 16.

En la pantalla que muestra la información de la persona existe un botón en la parte inferior llamado “Send message”, de clic sobre él, Figura 17.

Figura 17.

Y aparecerá una venta emergente, en donde se escribe el mensaje a enviar, Figura 18.

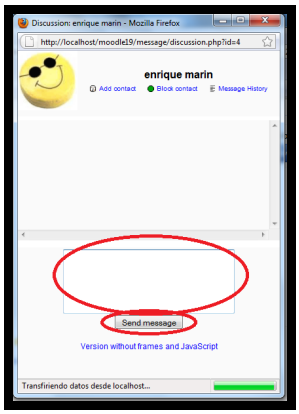


Figura 18.

Para enviar el mensaje se hace clic sobre el botón “Send message” y finalmente aparecerá en la parte superior de la venta un mensaje confirmando su envió, Figura 19.

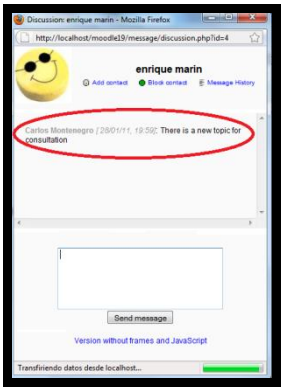


Figura 19.

Guía para el profesor en la creación de módulos de comunicación en la herramienta que realiza un modelado específico de dominio para la construcción de módulos de LMS independientes de la plataforma.

Esta herramienta esta creada bajo entorno eclipse y funciona como un plugin para él, en la construcción de la herramientas se utilizaron conceptos y tecnologías como: MDE, EMF, Ecore, GMF, MOFScript, entre otras.

Para poder correr la herramienta es necesario crear un "Java Project" en eclipse. Teniendo eclipse abierto se da clic en "File" ->"New"-> "Java Project", Figura 20.

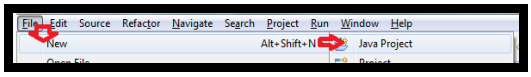


Figura 20.

Se coloca un nombre al proyecto en "Project name" y se da clic en Finish, Figura 21.

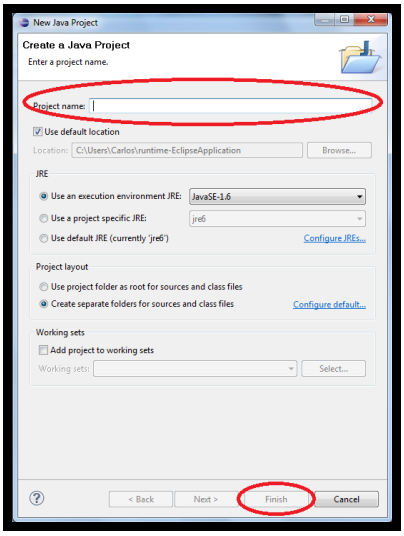


Figura 21.

Esto creara un nuevo proyecto en eclipse. En el "Package Explorer", se vería como en la Figura 22.

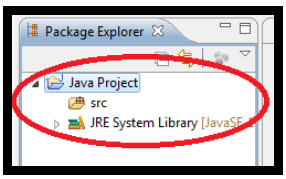


Figura 22.

Luego se da clic derecho sobre el "Java Project" creado y se escoge la opción "New" -> "Example", Figura 23.

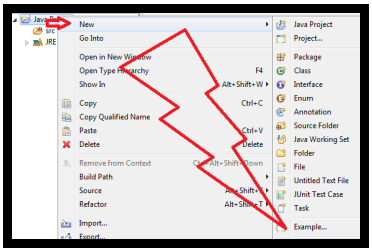


Figura 23.

Apareceré una nueva venta, en ella se seleccionara “Modellms Diagram” que corresponde a la herramienta de modelado para módulos de plataformas LMS, Figura 24.

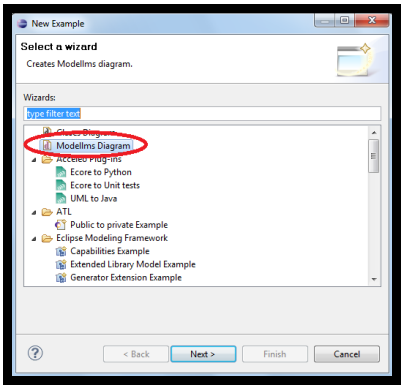


Figura 24.

Clic sobre el botón “next”, apareceré una nueva ventana en donde se pone el nombre del fichero a crear en “File name” y se da “Finish”, Figura 25.

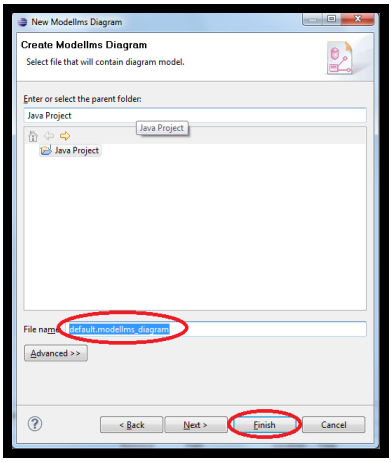


Figura 25.

Así se ha creado un nuevo proyecto en la herramienta, esta se compone de un “package explorer”, una paleta de herramientas “Palette” y un área de trabajo, Figura 26.

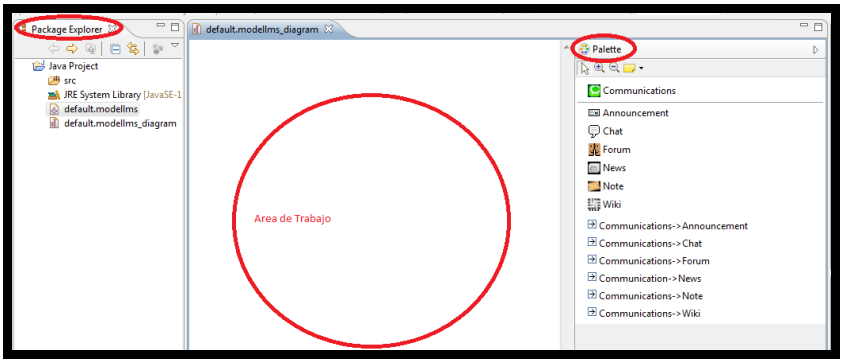


Figura 26.

El funcionamiento de la herramienta es muy sencillo, la paleta de herramientas "Palette" básicamente tiene dos tipos de herramientas a seleccionar Nodos y Conectores, los primeros corresponden a los módulos que tiene un LMS en el área de Comunicaciones y los otros son las conexiones que existen entre esos nodos, Figura 27.

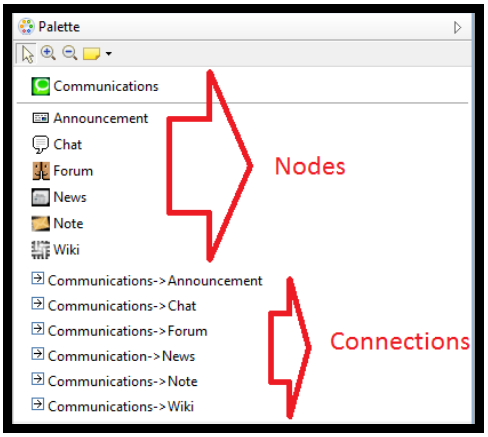


Figura 27.

Para crear los módulos basta con arrastrar los nodos de la "Pallete" al área de trabajo, rellenar los campos y conectarlos respetando las siguientes reglas:

- Un cursos solo puede tener un modulo de "Communications".
- El modulo de "Communications" tiene cero o muchos: "Announcements", "Chat", "Forum", "News", "Note", y "Wiki".

La herramienta valida los siguientes casos:

- Solo permite un modulo de "Communications".
- Los enlaces solo pueden corresponder entre el nodo "Communications" y su respectiva herramienta, por ejemplo "Communications -> Chat ", solo sirve para conectar el nodo "Communications" con el nodo "Chat", la herramienta no permite utilizarlo en otro caso.

- Cuando se genere el código a desplegar, aquellos nodos que estén sueltos (sin conexión) no serán tenidos en cuenta para la creación del curso.

Es muy importante considerar que los elementos modelados con esta herramienta se desplegaran en un solo tema del curso, con lo cual es substancial considerar los nombre de los campos en cada elemento (Nodo o Modulo) como genéricos, para no tener que cambiar el modelo cada vez que se despliegue sobre el curso.

A continuación se muestra la creación de los módulos para un tema de un curso, Figura 28.

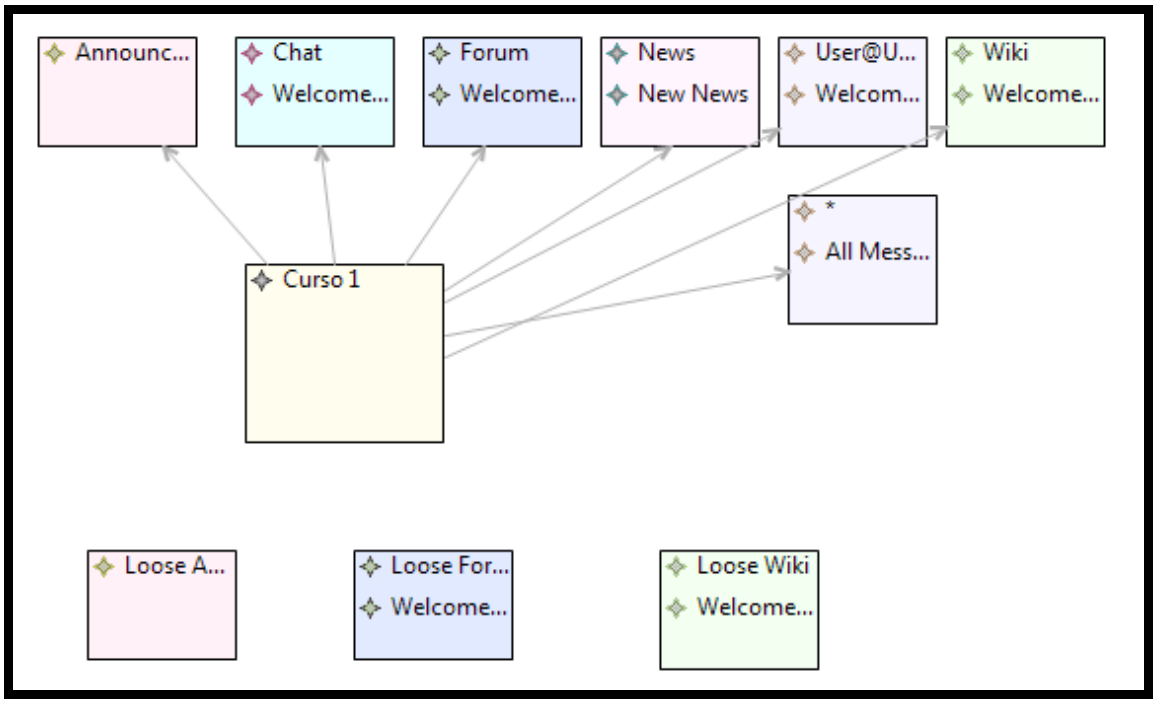


Figura 28.

En la Figura 28 se pueden visualizar los módulos que están conectados con sus respectivas relación y los que no tiene conexión, estos últimos no serán desplegados en el tema del curso.

Observaciones a tener en cuenta en el modelado:

- Si se desea enviar una “Note” a todos los inscritos en un curso, en el campo “Send To” debe ir un *.
- Si se desea enviar una “Note” a un solo inscrito en un curso, en el campo “Send To” debe ir un el correo completo con el que la persona se inscribió en la plataforma virtual. Si este correo no corresponde con el de la persona el mensaje no será enviado.
- Recuerde diligenciar todos los campos en los nodos, pues aquellos campos en los cuales el usuario no ponga información serán tomados en blanco y se procesaran erradamente.

- **Todos los campos aparecen con información por defecto pero esta información solo es de carácter orientativo, No implica información en los nodos.**

Guarde todos los cambios y envié al administrador de la aplicación los ficheros generados, para este caso de ejemplo “default.modellms” y “default.modellms_diagram”.

Una vez reciba respuesta del Administrador, ingrese a moodle con su perfil de profesor y cambie a modo de edición. Ubíquese en cada uno de los temas y despliegue el ComboBox que dice “Add an activity”, seleccione la opción [[modulename]], Figura 29.

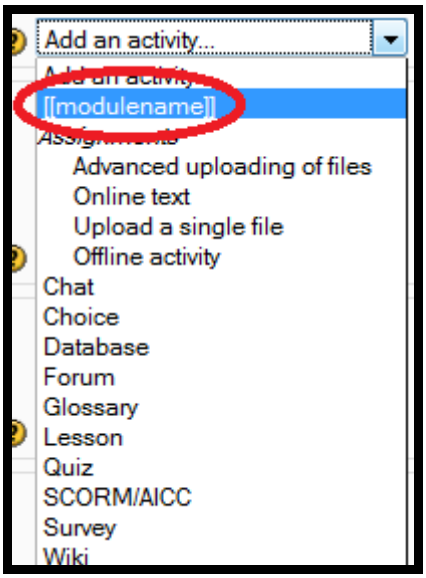


Figura 29.

Una vez seleccionada esta opción los elementos modelados serán desplegados en la plataforma, Figura 30 Y 31.

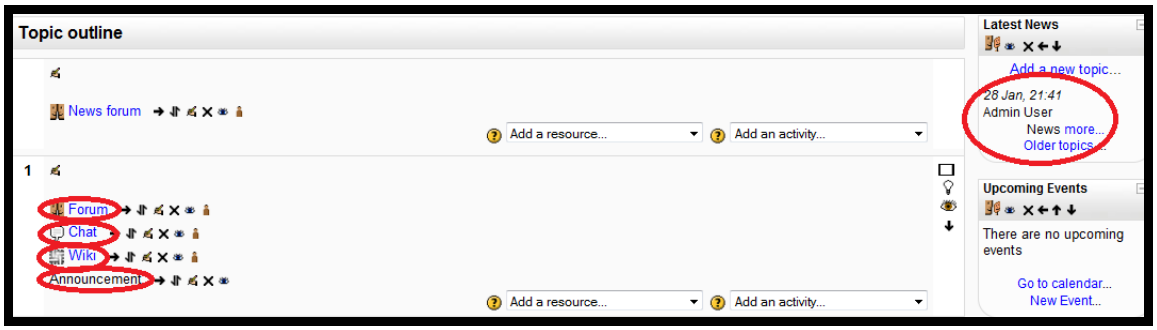


Figura 30.

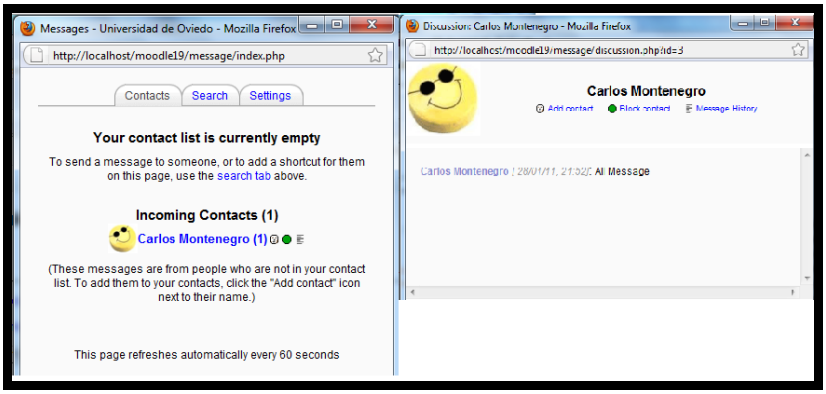

Messages - Universidad de Oviedo - Mozilla Firefox
http://localhost/moodle19/message/index.php
Contacts
Search
Settings
Your contact list is currently empty
To send a message to someone, or to add a shortcut for them on this page, use the search tab above.
Incoming Contacts (1)
Carlos Montenegro (1)
(These messages are from people who are not in your contact list. To add them to your contacts, click the "Add contact" icon next to their name.)
This page refreshes automatically every 60 seconds
Carlos Montenegro
Add contact
Block contact
Message History
Carlos Montenegro : 28/01/11, 21:32: Al Message

Figura 31.

Guía para el Administrador de la herramienta que realiza un modelado específico de dominio para la construcción de módulos de LMS independientes de la plataforma y moodle.

Una vez reciba los ficheros enviado por el profesor, para este caso de ejemplo "default.modellms" y "default.modellms_diagram":

1. Cambie la extensión del fichero "default.modellms" a "default.ecore".
2. Abra eclipse y aplique las transformaciones MOFScrip que tiene el fichero "modellmsTransformationMoodle.m2t".
3. Esta transformación generan dos ficheros "lib.php" y "mod_form.php" cópielas y reemplace las anteriores en la ruta en donde este el modulo "plantilladsl" previamente creado, que debe estar en "*path_moodle**mod**plantilladsl*".
4. Abra Moodle en modo Administrador y compile la plataforma nuevamente, para ello ingrese al navegador y digite "*path_moodle/admin/index.php*".

Con estos pasos ya el profesor podrá ver el modulo con todo lo que el creo en moodle y lo podrá desplegar.

Guía pruebas sobre Atutor

Carlos Enrique Montenegro Marín

Guía de Pruebas para una herramienta que realiza un modelado específico de dominio para la construcción de módulos de LMS independientes de la plataforma.

Objetivo: Probar el rendimiento de utilizar la herramienta que realiza un modelado específico de dominio para la construcción de módulos de LMS independientes de la plataforma, contra la creación de estos módulos manualmente. Las pruebas evaluaran rendimiento y esfuerzo del profesor en la creación de módulos para la plataforma.

Como MDA plantea la creación de aplicaciones partiendo desde un dominio específico hasta poder desplegar el modelo creado en una plataforma especifica. El despliegue se realizará bajo la plataforma ATutor 2.0.2.

Partiendo del caso hipotético; en el cual un profesor desea impartir una asignatura apoyándose en una plataforma virtual y bajo las siguientes consideraciones:

- ATutor vincula automáticamente por cada curso un foro, un chat, una sección de anuncios y una wiki.
- Por cada uno de los temas se desea agregar información genérica sobre la disponibilidad del nuevo tema en el foro, el chat, un anuncio, en la wiki y una noticia.
- También se desea enviar una nota a un estudiante en particular y luego a todos los inscritos en el curso, informado que “un nuevo tema está disponible en la plataforma virtual”.

La medición de tiempos se realizara para los dos ambientes y se tomara:

- Tiempo en la adición de información por cada herramienta.
- Tiempo en la adición de información en todas las herramientas que conforman un tema.
- Tiempo total en la adición de información en las herramientas de los cinco temas.

La medición de esfuerzo mide la cantidad de ocasiones que el usuario tuvo que seleccionar o ingresar algún tipo de dato al sistema, y se realizara para:

- Esfuerzo en la adición de información por cada herramienta.
- Esfuerzo en la adición de información en todas las herramientas que conforman un tema.
- Esfuerzo total en la adición de información en las herramientas de los cinco temas.

Para poder comparar los tiempos es necesario que en la herramienta DSL, se cree el modulo y a continuación su enlace, **NO** crear todos los nodos y luego todos los enlaces.

Como la prueba busca medir el tiempo y nivel de esfuerzo en adicionar la información a cada modulo, se le proporcionaran los datos al usuario para ingresar:

Chat	Name	Chat
	Text	Welcome to Chat
Forum	Name	Forum
	Text	Welcome to Forum
Wiki	Name	Wki
	Text	Welcome to Wiki
Announcement	Text	New topic
News	Name	New topic
	Text	New topic available
Notes student1@uniovi.es	Send to	student1@uniovi.es
	Text	New topic available
Notes All Users	Text	New topic available

Guía para el profesor en la adición de información en las herramientas de comunicación de ATutor.

1. Ingresar a la plataforma con su usuario y contraseña. Ver figura 1.

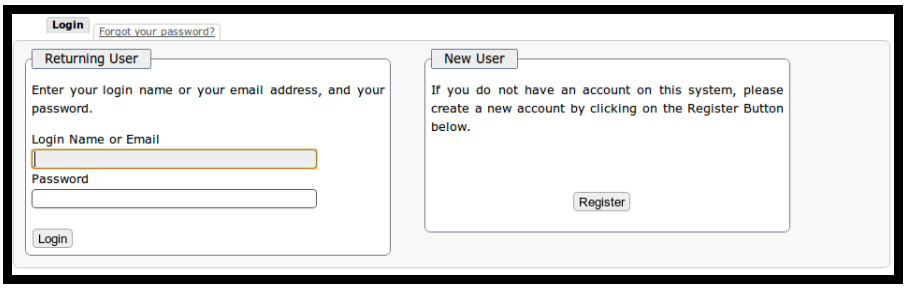


Figura 1.

2. Seleccione el curso a trabajar, ver figura 2.

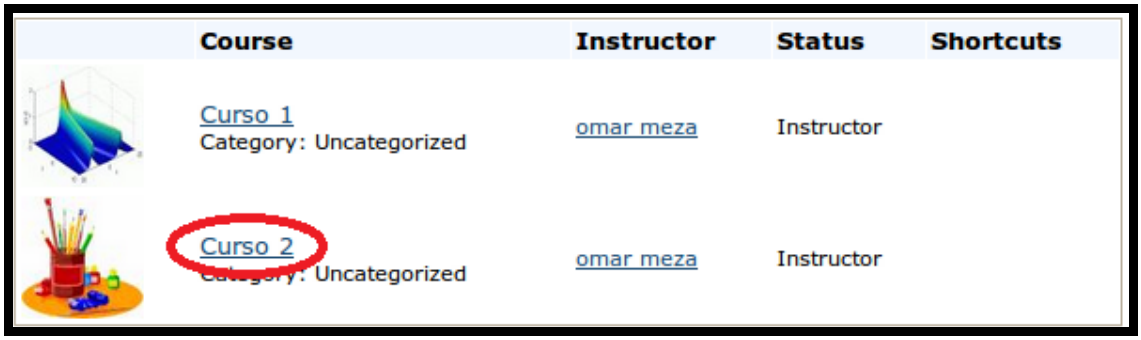


Figura 2.

3. Una vez ingresado al curso, selecciona la pestaña “Manage” para ingresar a las herramientas del mismo.

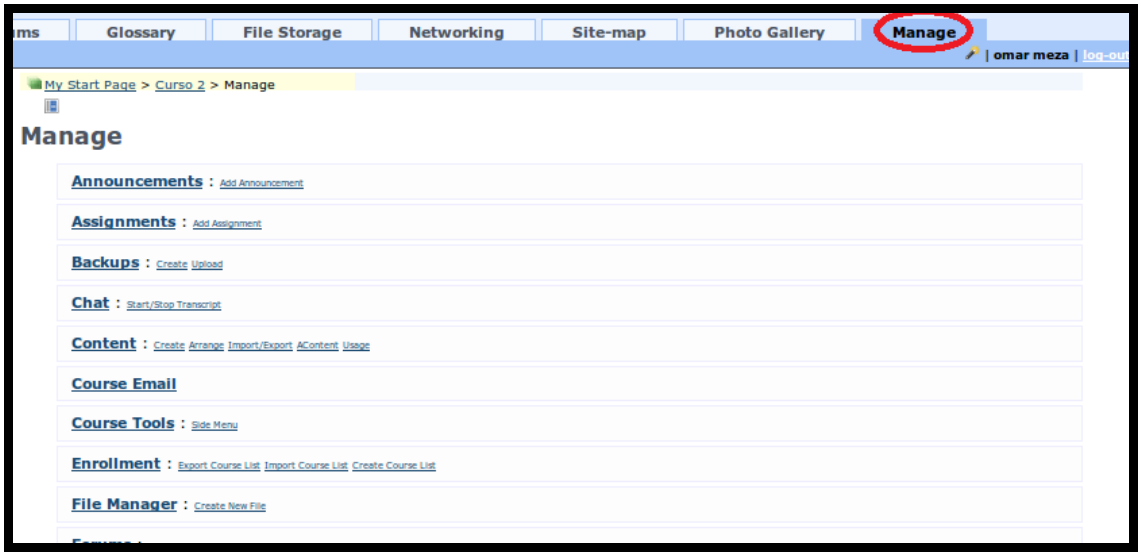


Figura 3.

4. Adición de información.

Para poder agregar la información usted debe seleccionar la herramienta que desea, como se mostró en la figura anterior. A continuación se muestra una tabla comparativa de opciones para ingresar los datos requeridos:

Concepto	Herramienta ATutor	Operación
Anuncio	Announcements	Add Announcements
Noticia	Assignments	Add Assignment
Foro	Forums	Create Forum
Chat	Chat	Enter the Chat
Wiki	Wiki	EditThisPage
Note Todos	Course Email	Send
Note	MyContacts	1. Search People 2. Send Message

a. Al seleccionar ***Announcement*** aparecerá una vista con los anuncios que tiene el curso. Para agregar un anuncio es necesario regresar a la pestaña *"Manage"* como ya se mencionó, se hace clic en la opción *"Add Announcement"* ingresando el título y descripción como se muestra en la Figura 4, una vez concluido este proceso el anuncio se verá reflejado en la página principal del curso, véase Fig. 5.

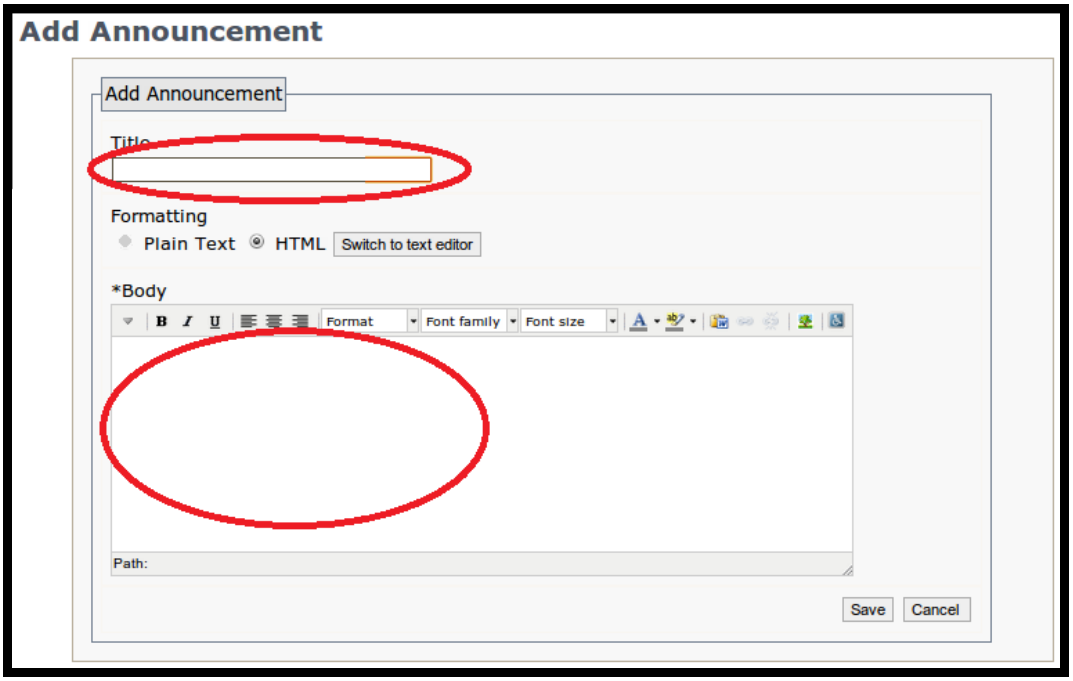


Figura 4.

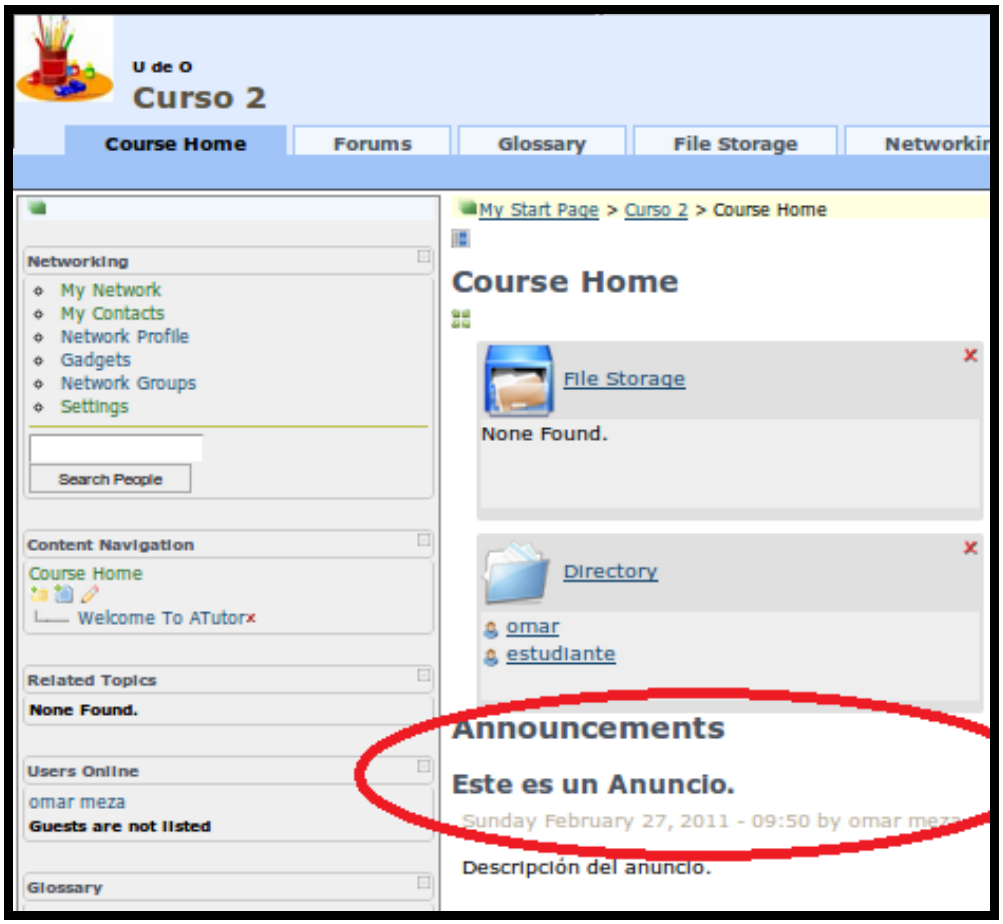


Figura 5.

b. Al seleccionar ***Assignments*** aparecerá una vista con las noticias para los alumnos pertenecientes al curso. Para agregar una noticia es necesario regresar a la pestaña *"Manage", clic en "Add Assignment" e igresar el título de la noticia, especificar si es para todos los alumnos o solo grupos específicos, en este caso se aplica para todos los alumnos, véase Figura 6.*

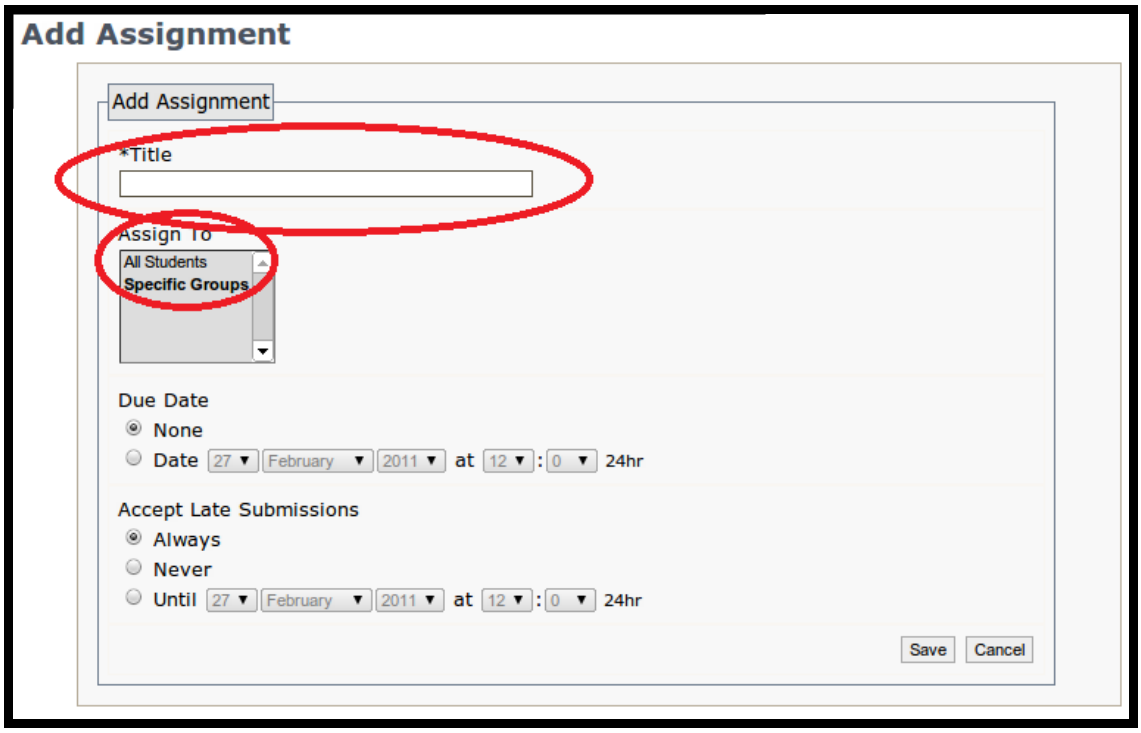


Figura 6.

c. Al seleccionar ***Forums*** aparecerá una vista con los foros para los alumnos pertenecientes al curso. Para agregar un nuevo foro es necesario regresar a la pestaña *"Manage", clic en "Create Forum"* e ingresar los datos correspondientes de título & descripción, véase Figura7. Al finalizar el proceso enviará un mensaje de *"Action completed succesfully"* mostrando los foros del curso.

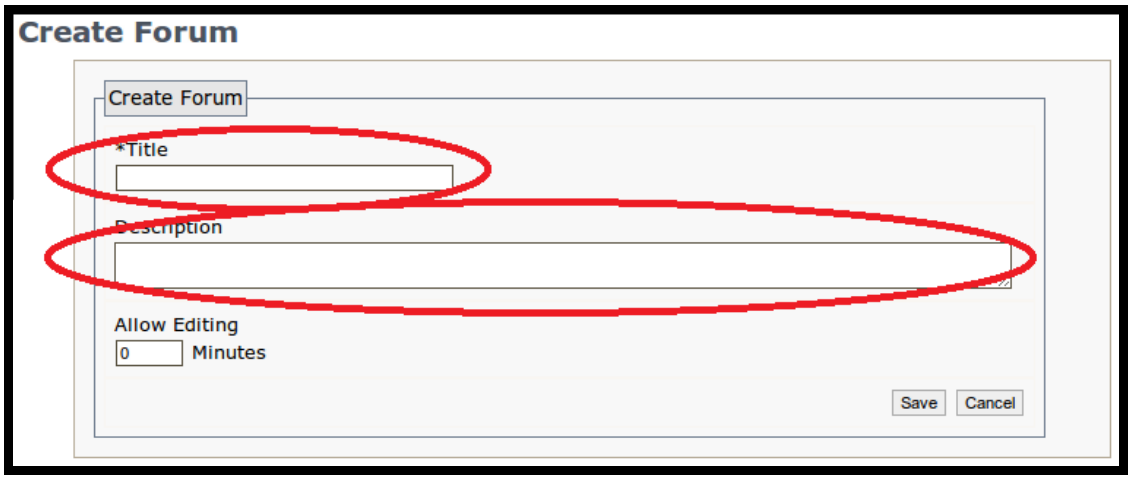


Figura 7.

d. Para ingresar al **Chat** se requiere ir a la página principal del curso "Course Home" dar clic en Chat y en "Enter the Chat", véase Figura 8.

Figura 8.

e. Al seleccionar ***Wiki*** aparecerá una página con opciones para modificar la actual, crear nuevas páginas, ayuda, actualización, entre otros. En este caso se toma en cuenta que el wiki es un módulo de terceros de nombre Ewiki, que ha sido incorporado a la plataforma. Para continuar con la creación del Wiki es necesario dar clic en "EditThisPage", modificarlo con nuestro texto y guardar. Al completar el proceso se verán reflejados los cambios en la página de wiki, véase Figura 9.

Figura 9.

f. Al seleccionar ***Course Email*** aparecerá una ventana para ingresar los datos a enviar, el subjet & el body. En este caso enviaremos el email a todos los enrolados, de no estar seleccionada la casilla “Enrolled” es necesario seleccionarla, véase Figura 10.

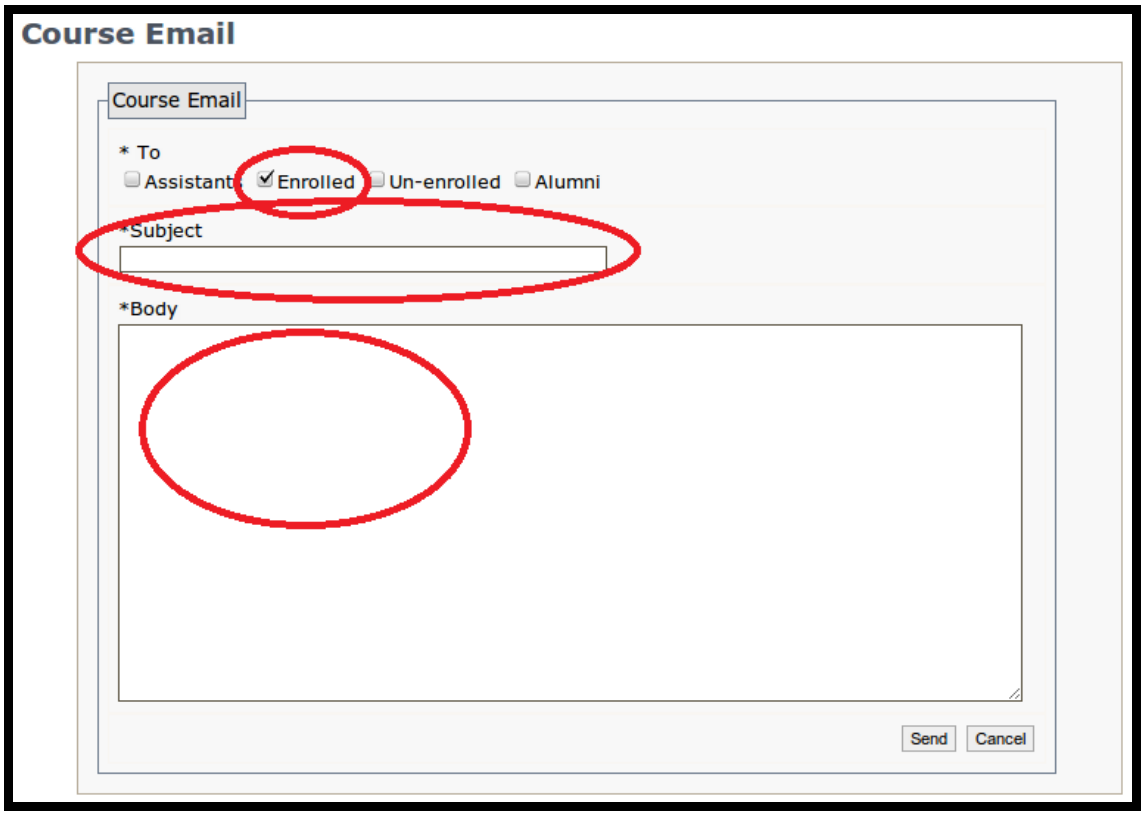


Figura 10.

g. Para enviar un ***Email*** a un solo estudiante es necesario ir a la página principal del curso “Course Home”, buscarlo en “My Contacts”, clic en la opción “Send Message”, rellenar el formulario y enviarlo, véase Figura 11.

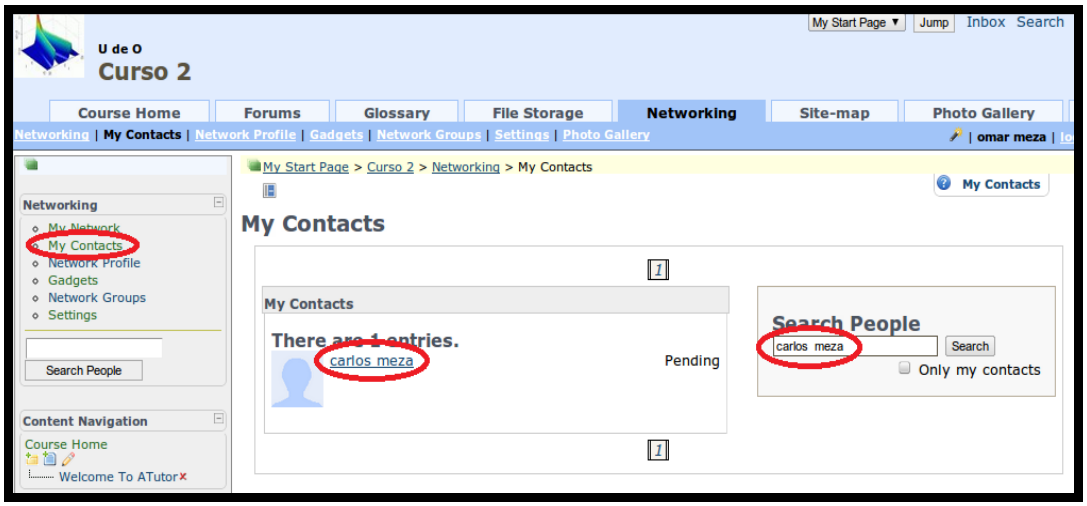


Figura 11.

Figura 12.

<u>Guía para el profesor en la creación de módulos de comunicación en la herramienta que realiza un modelado específico de dominio para la construcción de módulos de LMS independientes de la plataforma.</u>

Esta herramienta esta creada bajo entorno eclipse y funciona como un plugin para él, en la construcción de la herramientas se utilizaron conceptos y tecnologías como: MDE, EMF, Ecore, GMF, MOFScript, entre otras.

Para poder correr la herramienta es necesario crear un "Java Project" en eclipse. Teniendo eclipse abierto se da clic en "File" ->"New"-> "Java Project", Figura 13.

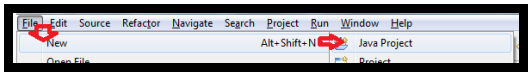


Figura 13.

Se coloca un nombre al proyecto en "Project name" y se da clic en Finish, Figura 14.

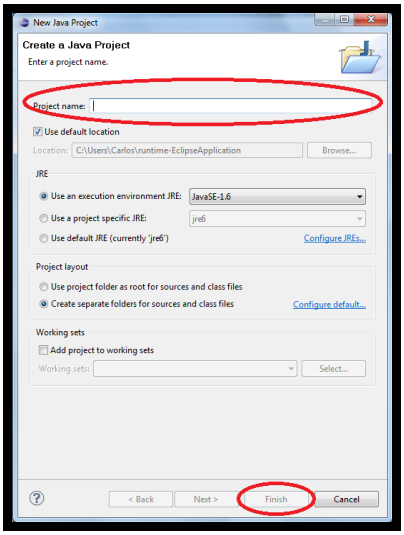


Figura 14.

Esto creara un nuevo proyecto en eclipse. En el “Package Explorer”, se vería como en la Figura 15.

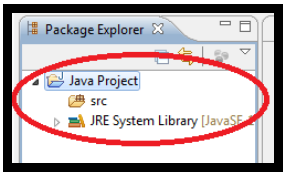


Figura 15.

Luego se da clic derecho sobre el “Java Project” creado y se escoge la opción “New” -> “Example”, Figura 16.

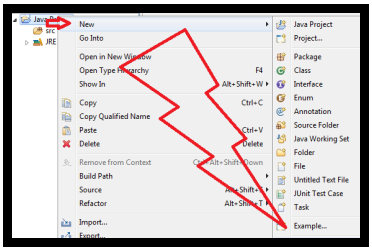


Figura 16.

Apareceré una nueva venta, en ella se seleccionara “Modellms Diagram” que corresponde a la herramienta de modelado para módulos de plataformas LMS, Figura17.

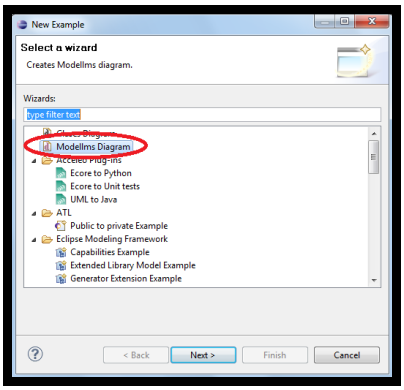


Figura 17.

Clic sobre el botón “next”, apareceré una nueva ventana en donde se pone el nombre del fichero a crear en “File name” y se da “Finish”, Figura 18.

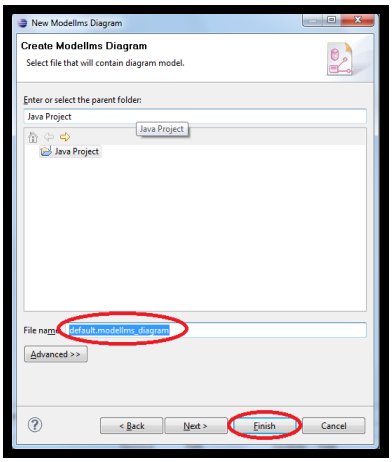


Figura 18.

Así se ha creado un nuevo proyecto en la herramienta, esta se compone de un “package explorer”, una paleta de herramientas “Palette” y un área de trabajo, Figura 19.

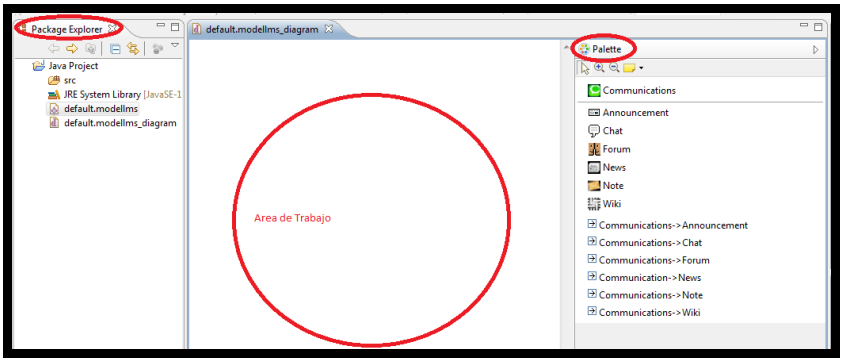


Figura 19.

El funcionamiento de la herramienta es muy sencillo, la paleta de herramientas “Palette” básicamente tiene dos tipos de herramientas a seleccionar Nodos y Conectores, los primeros

corresponden a los módulos que tiene un LMS en el área de Comunicaciones y los otros son las conexiones que existen entre esos nodos, Figura 20.

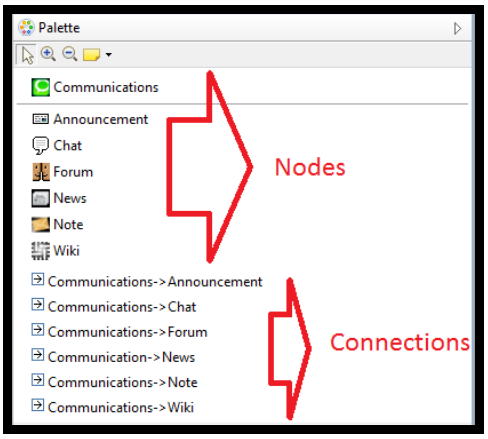


Figura 20.

Para crear los modulos basta con arrastrar los nodos de la “Pallete” al área de trabajo, rellenar los campos y conectarlos respetando las siguientes reglas:

- Un cursos solo puede tener un modulo de “Communications”.
- El modulo de “Communications” tiene cero o muchos: “Announcements”, “Chat”, “Forum”, “News”, “Note”, y “Wiki”.

La herramienta valida los siguientes casos:

- Solo permite un modulo de “Communications”.
- Los enlaces solo pueden corresponder entre el nodo “Communications” y su respectiva herramienta, por ejemplo “Communications -> Chat “, solo sirve para conectar el nodo “Communications” con el nodo “Chat”, la herramienta no permite utilizarlo en otro caso.
- Cuando se genere el código a desplegar, aquellos nodos que estén sueltos (sin conexión) no serán tenidos en cuenta para la creación del curso.

Es muy importante considerar que los elementos modelados con esta herramienta se desplegaran en un solo tema del curso, con lo cual es substancial considerar los nombre de los campos en cada elemento (Nodo o Modulo) como genéricos, para no tener que cambiar el modelo cada vez que se despliegue sobre el curso.

A continuación se muestra la creación de los módulos para un tema de un curso, Figura 21.

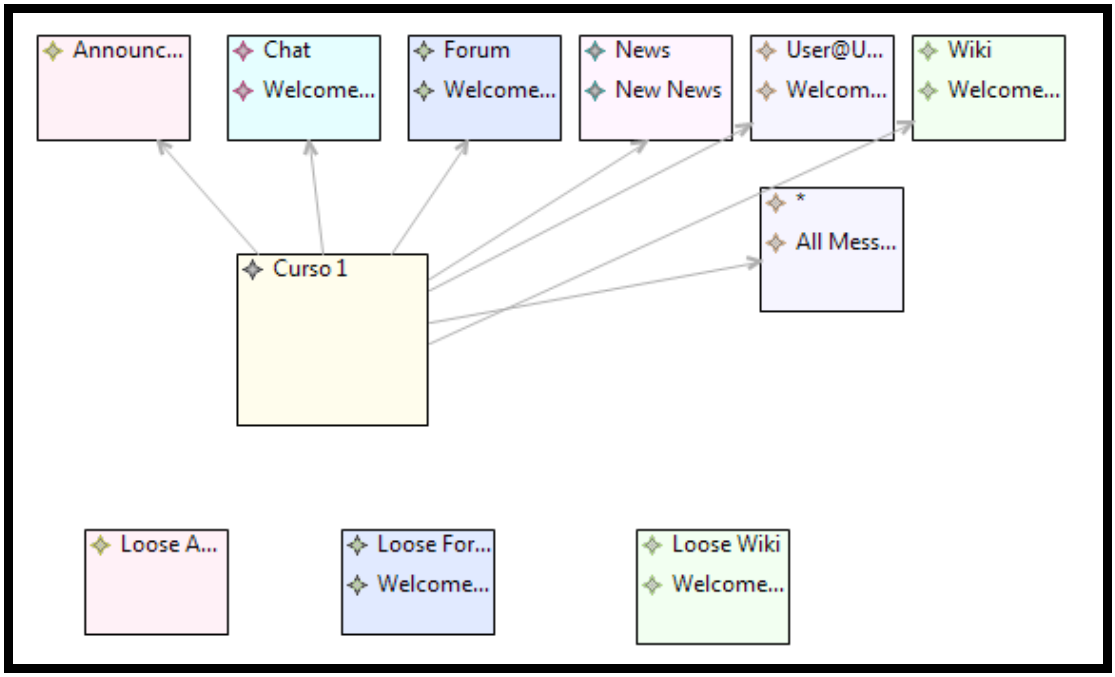


Figura 21.

En la Figura 21 se pueden visualizar los módulos que están conectados con sus respectivas relación y los que no tiene conexión, estos últimos no serán desplegados en el tema del curso.

Observaciones a tener en cuenta en el modelado:

- Si se desea enviar una “Note” a todos los inscritos en un curso, en el campo “Send To” debe ir un *.
- Si se desea enviar una “Note” a un solo inscrito en un curso, en el campo “Send To” debe ir un el correo completo con el que la persona se inscribió en la plataforma virtual. Si este correo no corresponde con el de la persona el mensaje no será enviado.
- Recuerde diligenciar todos los campos en los nodos, pues aquellos campos en los cuales el usuario no ponga información serán tomados en blanco y se procesaran erradamente.
- **Todos los campos aparecen con información por defecto pero esta información solo es de carácter orientativo, No implica información en los nodos.**

Guarde todos los cambios y envié al administrador de la aplicación los ficheros generados, para este caso de ejemplo “default.modellms” y “default.modellms_diagram”.

Una vez reciba respuesta del Administrador, ingrese a Atutor con su perfil de profesor, luego ingrese al curso en el cual desea desplegar los módulos creados con la herramientas DSL, después ingresar a la pestaña “Manager” buscar en la serie de herramientas la opción llamada ***“Plantilladsl”,*** para desplegar los módulos sencillamente de clic sobre esta nueva opción y los módulos se crearan acorde al modelo hecho con la herramienta DSL. Figuras 22.

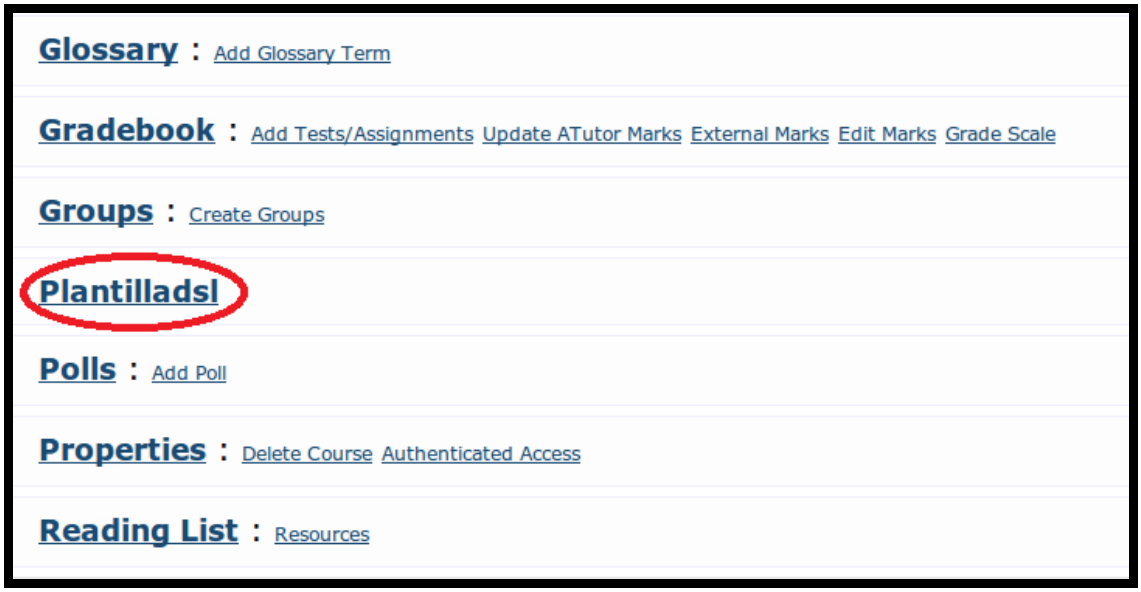

Glossary : Add Glossary Term
Gradebook : Add Tests/Assignments Update ATutor Marks External Marks Edit Marks Grade Scale
Groups : Create Groups
Plantilladsl
Polls : Add Poll
Properties : Delete Course Authenticated Access
Reading List : Resources

Figura 22.

Guía para el Administrador de la herramienta que realiza un modelado específico de dominio para la construcción de módulos de LMS independientes de la plataforma y atutor.

Una vez el Administrador reciba los ficheros enviado por el profesor, para este caso de ejemplo "default.modellms" y "default.modellms_diagram":

1. Cambie la extensión del fichero "default.modellms" a "default.ecore".
2. Abra eclipse y aplique las transformaciones MOFScrip que tiene el fichero "modellmsTransformationATutor.m2t".
3. Esta transformación generará siete ficheros, es requerido ingresar las librerías php siguientes: "class.phpmailer.php" y "class.smtp.php" a la carpeta "lib" de forma manual. Al terminar el proceso anterior comprimir la carpeta plantilladsl.zip con la siguiente estructura:

- \index.php
- \module.php
- \module.sql
- \module.xml
- \module_install.php
- \module_uninstall.php
- \lib\dsllib.lib.php
- \lib\class.phpmailer.php
- \lib\class.smtp.php

4. Abra atutor en modo Administrador y cargue el nuevo modulo, para ello valla a la opción "Modules", "Install Modules", seleccionar el fichero zip que se creó en el paso anterior e instalar, cumplir con los pasos requeridos para la instalación según el sistema operativo que se cuente, las instrucciones son vistas en el wizard de instalación de la plataforma, véase Figura 23.

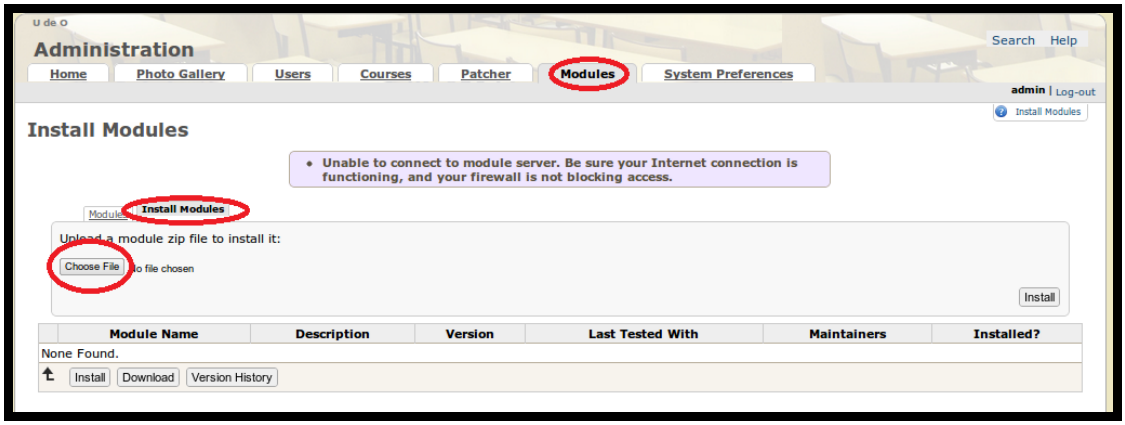


Figura 23.

5. Al finalizar la instalación solo queda habilitar el módulo, para realizarlo solo se selecciona el módulo y se da clic en el botón “Enable”, para desinstalarlo solo clic en “Unisntall”, véase Figura 24.

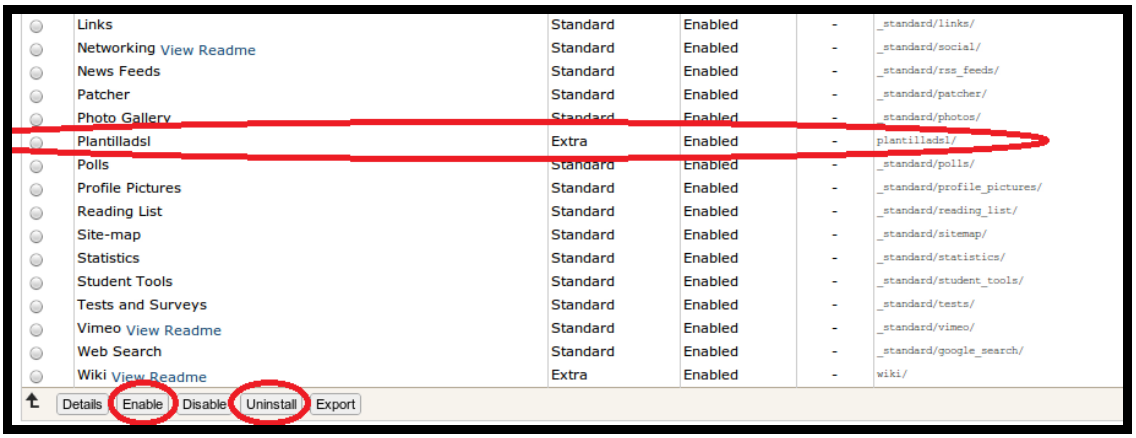


Figura 24.

Con estos pasos ya el profesor podrá ver el modulo con todo lo que el creo en atutor y lo podrá desplegar.

Guía pruebas sobre Claroline

Carlos Enrique Montenegro Marín

Guía de Pruebas para una herramienta que realiza un modelado específico de dominio para la construcción de módulos de LMS independientes de la plataforma.

Objetivo: Probar el rendimiento de utilizar la herramienta que realiza un modelado específico de dominio para la construcción de módulos de LMS independientes de la plataforma, contra la creación de estos módulos manualmente. Las pruebas evaluaran rendimiento y esfuerzo del profesor en la creación de módulos para la plataforma.

Como MDA plantea la creación de aplicaciones partiendo desde un dominio específico hasta poder desplegar el modelo creado en una plataforma especifica. El despliegue se realizara bajo la plataforma LMS Claroline 1.8.

Partiendo del caso hipotético; en el cual un profesor desea impartir una asignatura apoyándose en una plataforma virtual y bajo las siguientes consideraciones:

- Claroline vincula automáticamente por cada curso un foro, un chat, una sección de anuncios y una wiki.
- Por cada uno de los temas se desea agregar información genérica sobre la disponibilidad del nuevo tema en el foro, el chat, un anuncio, en la wiki, y una noticia.
- También se desea enviar una nota a un estudiante en particular y luego a todos los inscritos en el curso, informado que "un nuevo tema está disponible en la plataforma virtual".

La medición de tiempos se realizara para los dos ambientes y se tomara:

- Tiempo en la adición de información por cada herramienta.
- Tiempo en la adición de información en todas las herramientas que conforman un tema.
- Tiempo total en la adición de información en las herramientas de los cinco temas.

La medición de esfuerzo mide la cantidad de ocasiones que el usuario tuvo que seleccionar o ingresar algún tipo de dato al sistema, y se realizara para:

- Esfuerzo en la adición de información por cada herramienta.
- Esfuerzo en la adición de información en todas las herramientas que conforman un tema.
- Esfuerzo total en la adición de información en las herramientas de los cinco temas.

Para poder comparar los tiempos es necesario que en la herramienta DSL, se cree el modulo y a continuación su enlace, **NO** crear todos los nodos y luego todos los enlaces.

Como la prueba busca medir el tiempo y nivel de esfuerzo en adicionar la información a cada modulo, se le proporcionaran los datos al usuario para ingresar:

Chat	Name	Chat
	Text	Welcome to Chat
Forum	Name	Forum
	Text	Welcome to Forum
Wiki	Name	Wki
	Text	Welcome to Wiki
Announcement	Text	New topic
News	Name	New topic
	Text	New topic available
Notes student1@uniovi.es	Send to	student1@uniovi.es
	Text	New topic available
Notes All Users	Text	New topic available

<u>Guía para el profesor en la adición de información en las herramientas de comunicación de moodle.</u>

8. Ingresar a la plataforma con su usuario y contraseña.
9. Seleccione el curso a trabajar, ver figura 2.

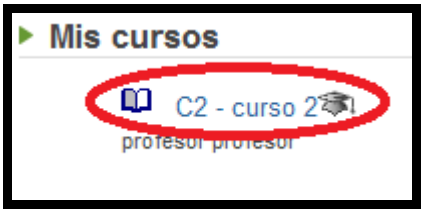


Figura 2.

10. Existen dos modos de vista, usted debe seleccionar el modo de vista "Responsable de curso", ya que esta le permite editar la información, ver figura 2.

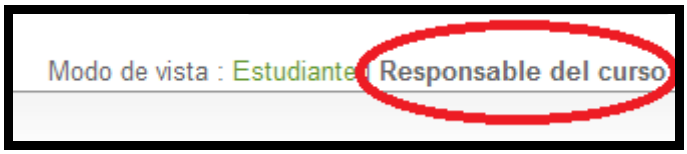


Figura 2.

11. Adición de información.

Para poder agregar la información usted debe seleccionar el modulo al cual desea agregar la información; *Anuncio* para Announcement y Note, *Foro* para Forum, *Debate* para Chat, Wiki para wiki o Añadir texto para News, Figura 3.

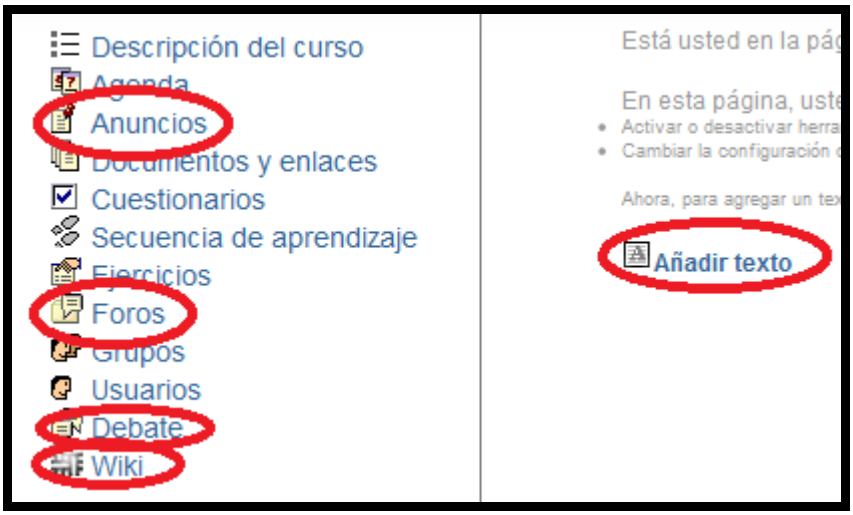


Figura 3.

a. Si usted selecciono *Anuncio*, aparecerá una vista con los anuncios que tiene el curso, allí pude seleccionar: *Añadir un anuncio* para publicar un nuevo anuncio (Announcement) *o Mensajes a usuarios seleccionados* para enviar un mensaje (Note) a los usuarios inscritos en el curso. Figura 4.

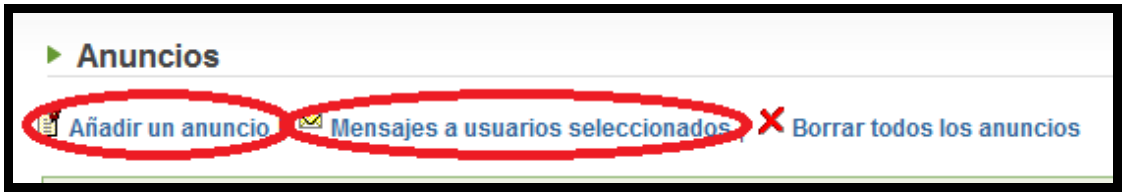


Figura 4.

i. Si usted selecciono *Añadir un anuncio,* aparecerá un formulario como el de la Figura 5, en este formulario se deben diligenciar los campos *Titulo* y *Contenido*, finalmente para publicar el anuncio se da clic sobre el botón *validar*, esto lo regresara nuevamente a la vista de Anuncios y vera su anuncio publicado en la plataforma, Figura 6.

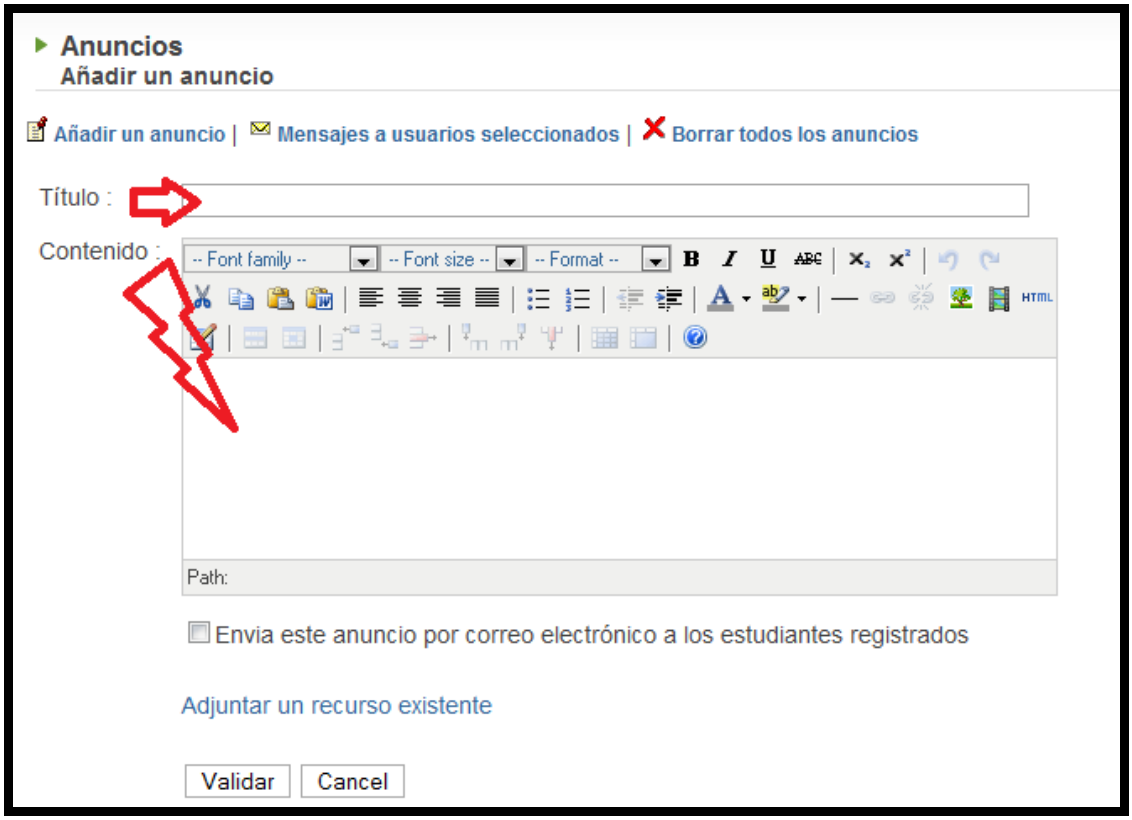


Figura 5.

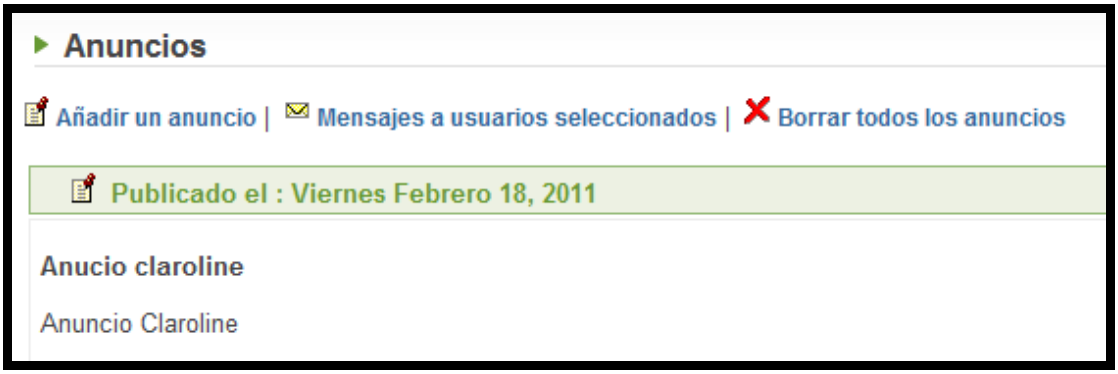


Figura 6.

ii. Si usted selecciono *Mensajes a usuarios seleccionados*, aparecerá una vista como la de la Figura 7, en esta vista se deben seleccionar los usuarios a los cuales se les va a enviar el mensaje de la *Lista de usuarios* y pasarlos a la lista de *Usuario seleccionado,* en el campo *Anuncio* se escribirá el mensaje a enviar y finalmente se da clic sobre el botón *Asunto* para enviar el mensaje, como se muestra en la Figura 7. (Observación: El mensaje (Note) es enviado al correo de la persona seleccionada, por este motivo el servidor de correo saliente SMTP debe estar configurado en la plataforma). Una vez el mensaje es enviado aparecerá un mensaje de confirmación como lo muestra la figura 8.

Figura 7.

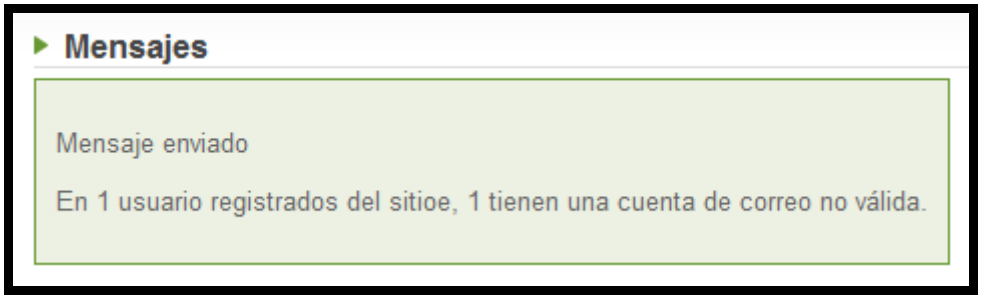


Figura 8.

b. Si usted selecciono *Foro*, aparecerá una vista con los foros que tiene el curso, para crear un nuevo foro, debe dar clic sobre *Crear foro* como se muestra en la figura 9, aparecerá un nuevo formulario a diligenciar con el *Nombre* y la *Descripción* del foro, Figura 10. Se llenan estos datos y se da clic en *Validar* y lo

regresara nuevamente a la vista de foros con un mensaje que confirma la creación del foro.

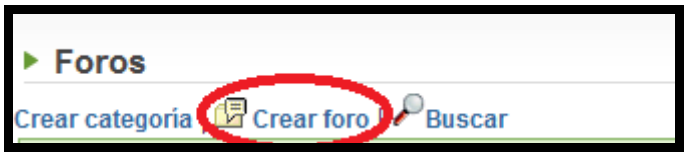


Figura 9.

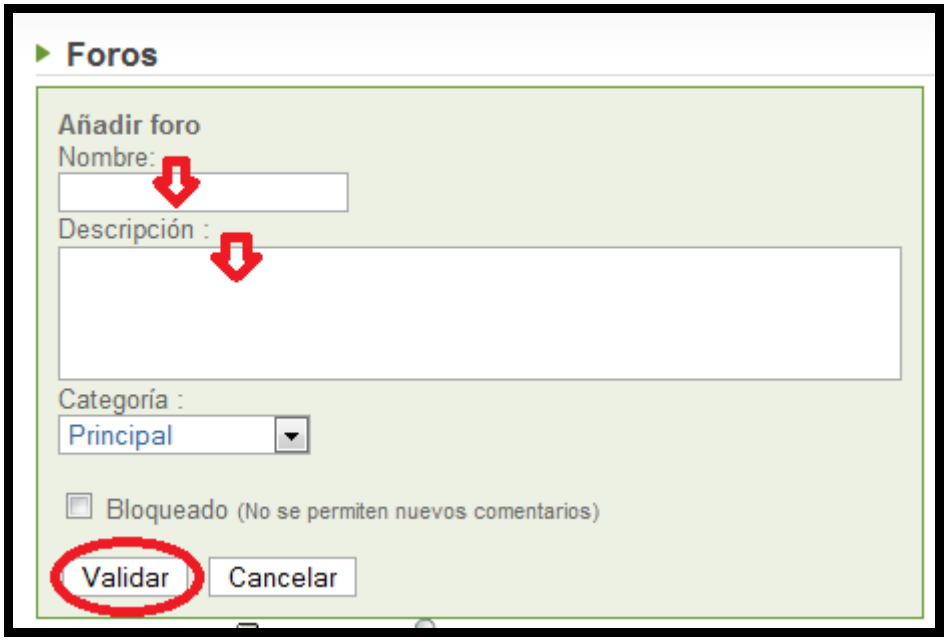


Figure 10.

c. Si usted selecciono *Debate (chat)*, aparecerá una vista que contiene la sala de debate del curso (chat), con los aportes que los usuarios han realizado, para ingresar un nuevo aporte, se debe escribir en el campo de texto ubicado en la parte inferior y dar clic en >> para enviar el mesnaje al chat, como se muestra en la figura 11. Los aportes se mostraran en cuadro de texto central de la sala de debates, indicando la fecha, hora, nombre del usuario y texto que se agrego, como se muestra en la figura 12.

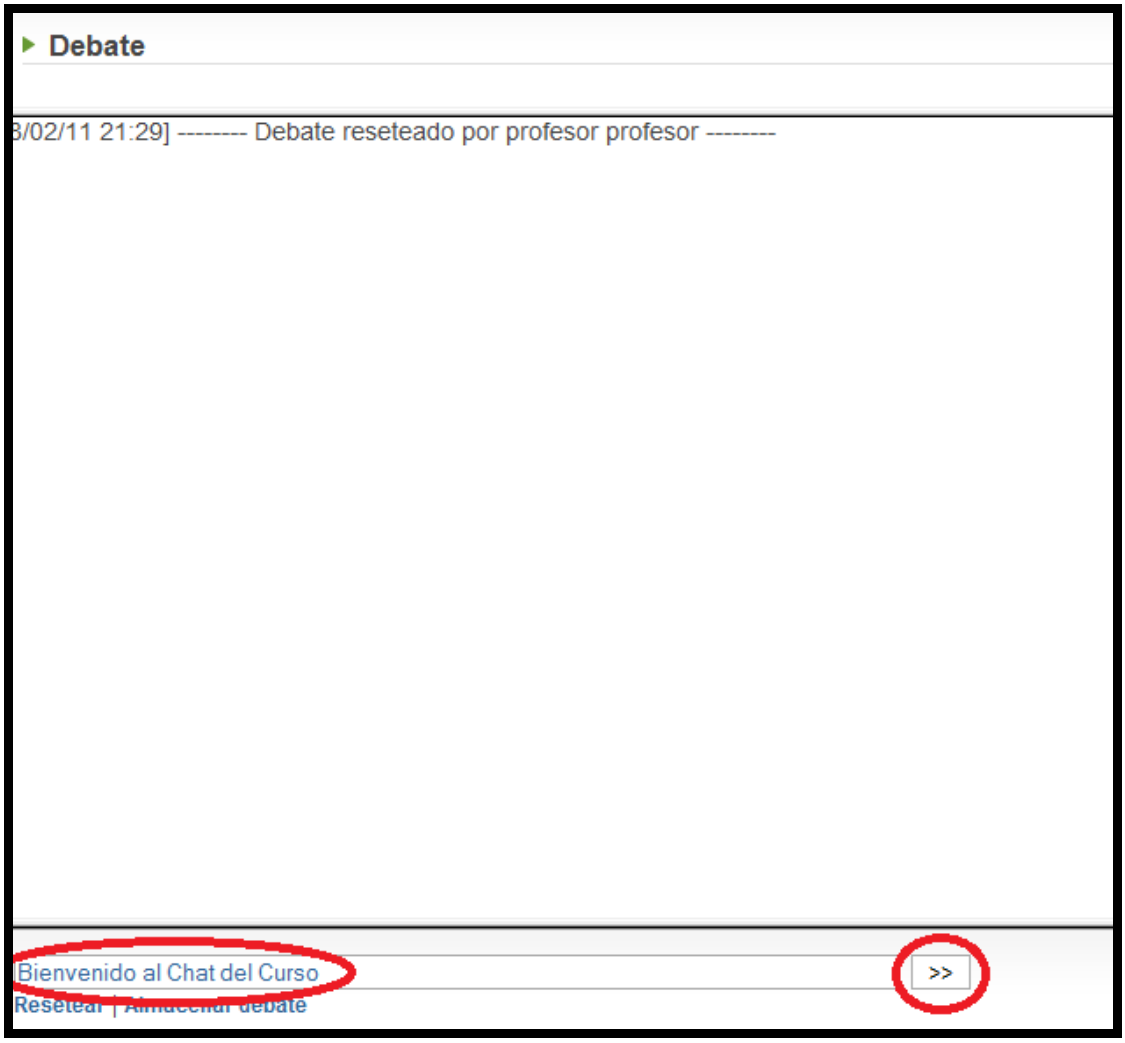


Figura 11.

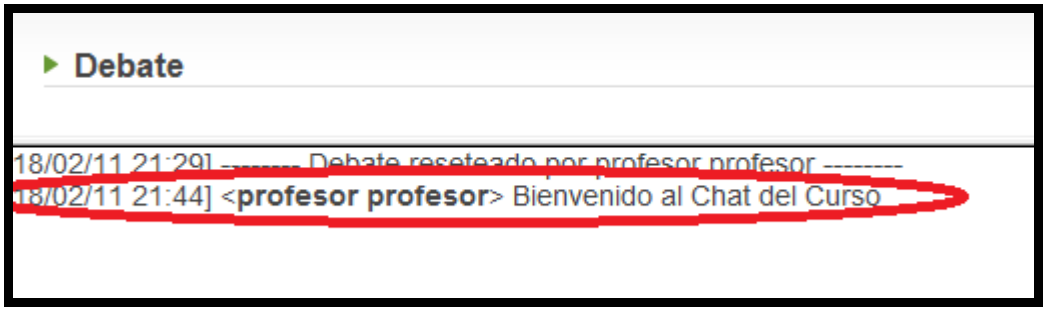


Figura 12.

d. Si usted selecciono *Wiki*, aparecerá una vista con las Wikis que tiene el curso, para crear una nueva Wiki, debe dar clic sobre *Crear un nuevo Wiki* como se muestra en la figura 13, aparecerá un nuevo formulario a diligenciar con el *Titulo* y la *Descripción* de la Wiki, Figura 14. Se llenan estos datos y se da clic

en *Validar,* esto lo regresara nuevamente a la vista de Wiki y podrá visualizar la nueva Wiki creada.

Figura 13.

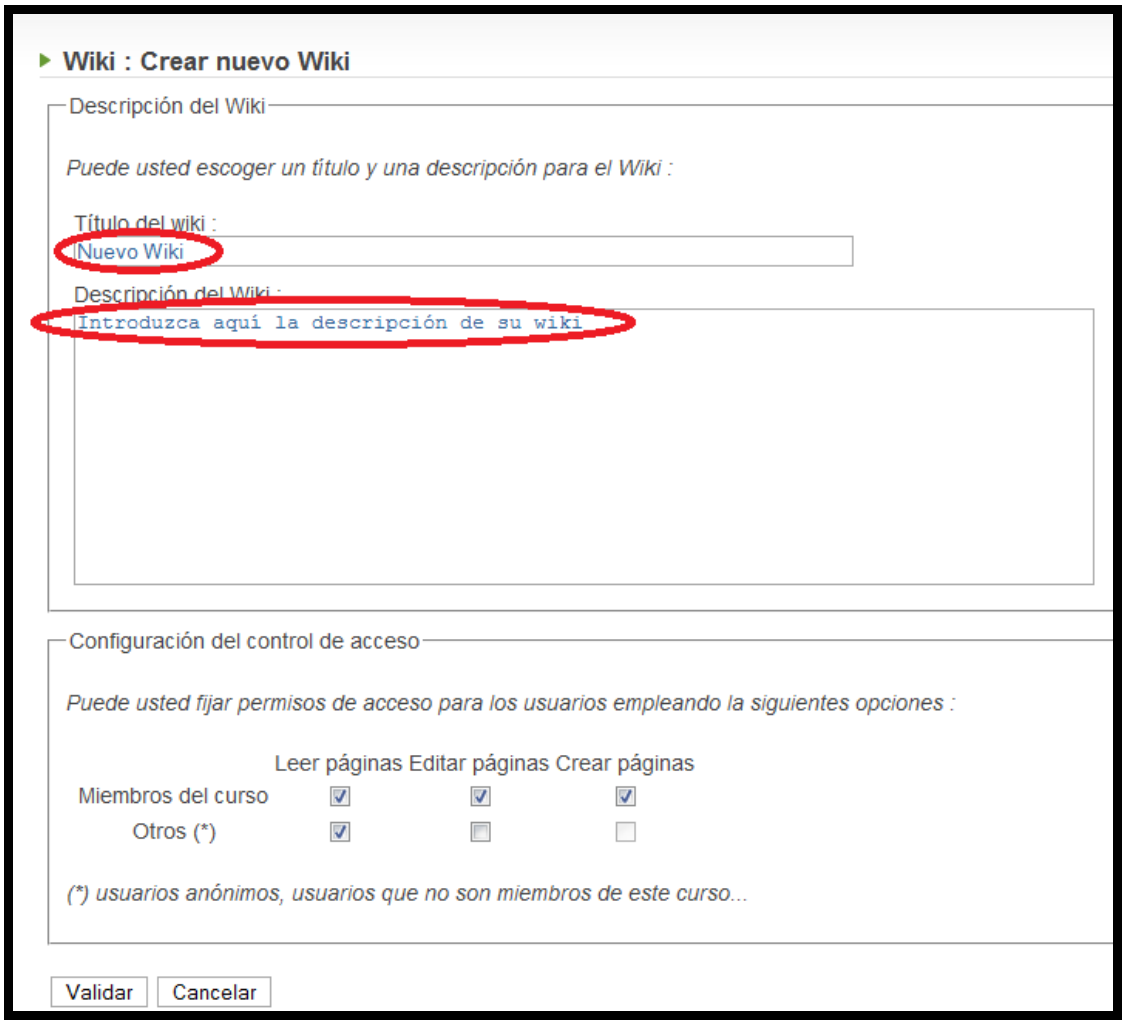


Figura 14.

e. Si usted selecciono *Anadir texto (news)*, aparecerá una vista con un editor, para ingresar el texto de la noticia (news), como lo muestra la figura 15. Para crear una nueva news, debe ingresar el texto y dar clic sobre *Validar,* esto lo regresara nuevamente a la vista de principal del curso y podrá visualizar la nueva noticia creada, Figura 26.

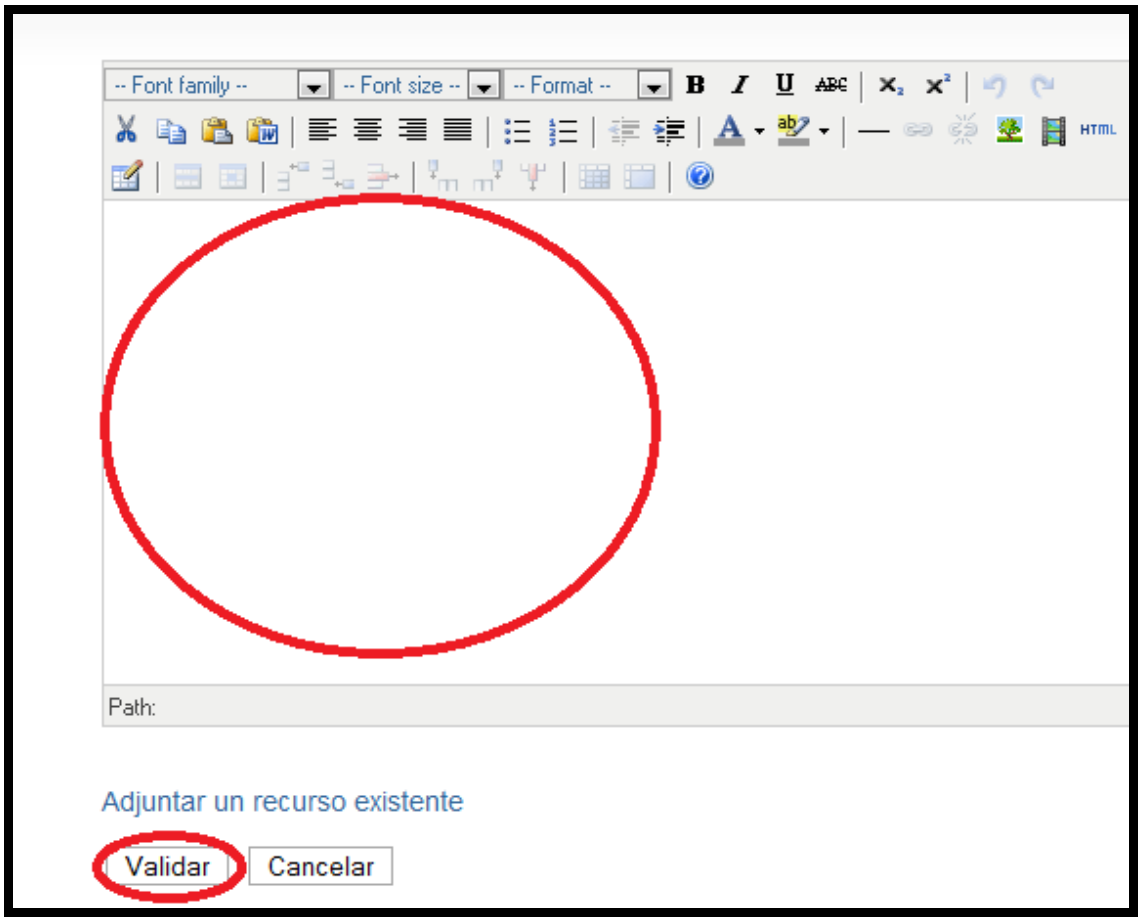


Figura 15.

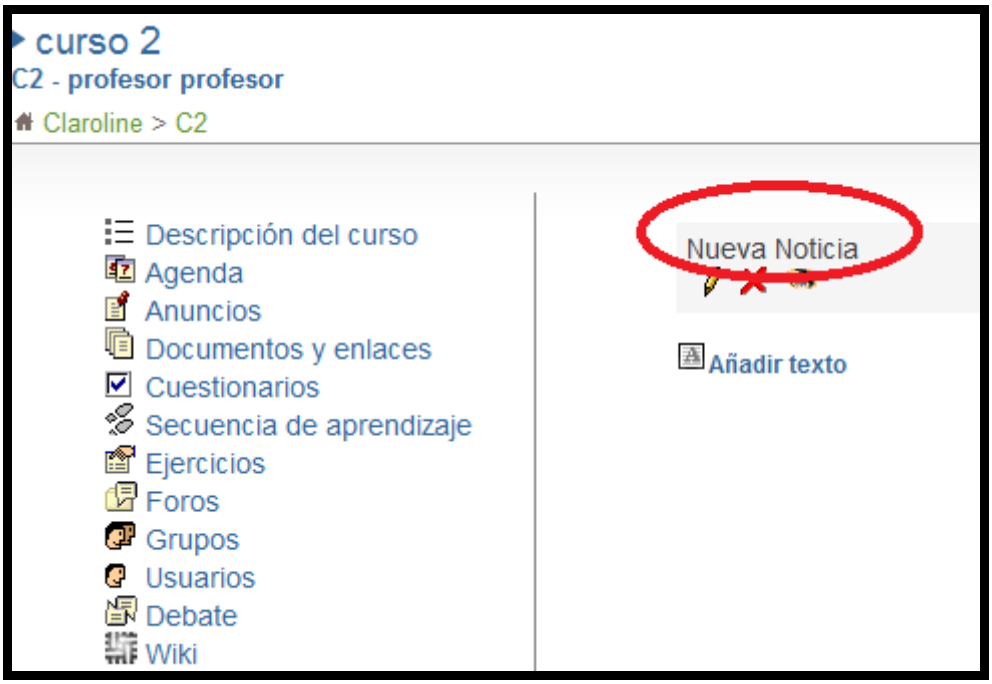


Figura 16.

Guía para el profesor en la creación de módulos de comunicación en la herramienta que realiza un modelado específico de dominio para la construcción de módulos de LMS independientes de la plataforma.

Esta herramienta esta creada bajo entorno eclipse y funciona como un plugin para él, en la construcción de la herramientas se utilizaron conceptos y tecnologías como: MDE, EMF, Ecore, GMF, MOFScript, entre otras.

Para poder correr la herramienta es necesario crear un "Java Project" en eclipse. Teniendo eclipse abierto se da clic en "File" ->"New"-> "Java Project", Figura 17.

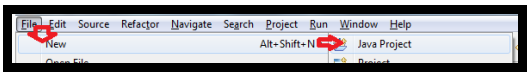


Figura 17.

Se coloca un nombre al proyecto en "Project name" y se da clic en Finish, Figura 18.

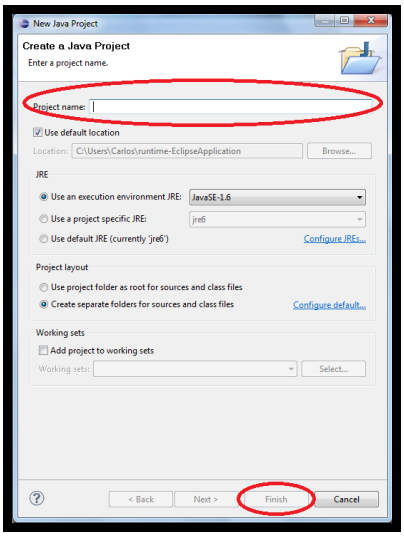


Figura 18.

Esto creara un nuevo proyecto en eclipse. En el "Package Explorer", se vería como en la Figura 19.

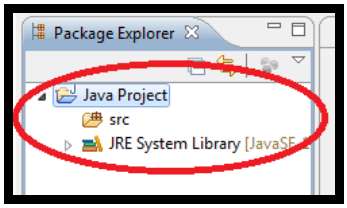


Figura 19.

Luego se da clic derecho sobre el "Java Project" creado y se escoge la opción "New" -> "Example", Figura 20.

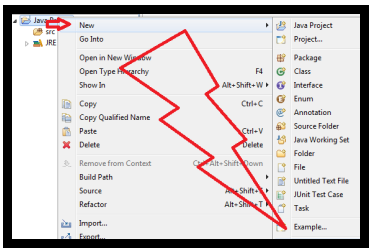


Figura 20.

Apareceré una nueva venta, en ella se seleccionara "Modellms Diagram" que corresponde a la herramienta de modelado para módulos de plataformas LMS, Figura 21.

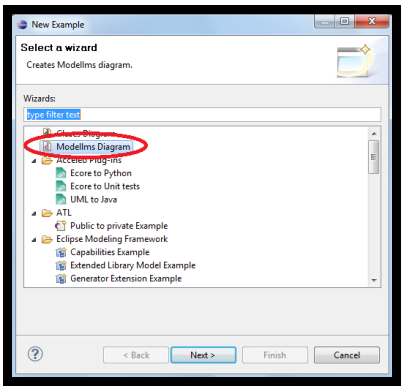


Figura 21.

Clic sobre el botón "next", apareceré una nueva ventana en donde se pone el nombre del fichero a crear en "File name" y se da "Finish", Figura 22.

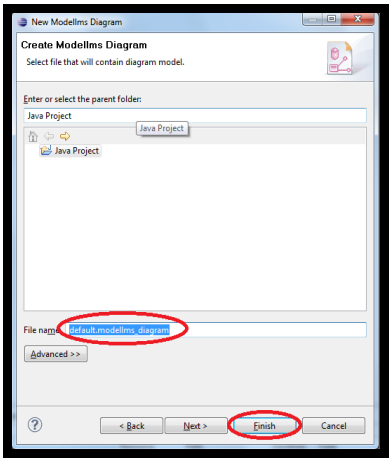


Figura 22.

Así se ha creado un nuevo proyecto en la herramienta, esta se compone de un "package explorer", una paleta de herramientas "Palette" y un área de trabajo, Figura 23.

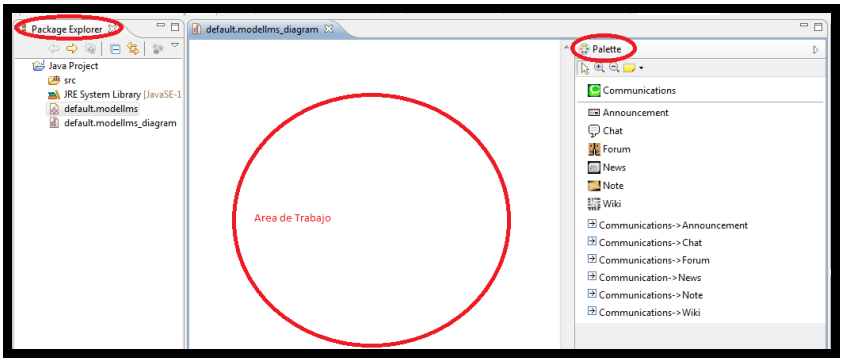


Figura 23.

El funcionamiento de la herramienta es muy sencillo, la paleta de herramientas "Palette" básicamente tiene dos tipos de herramientas a seleccionar Nodos y Conectores, los primeros corresponden a los módulos que tiene un LMS en el área de Comunicaciones y los otros son las conexiones que existen entre esos nodos, Figura 23.

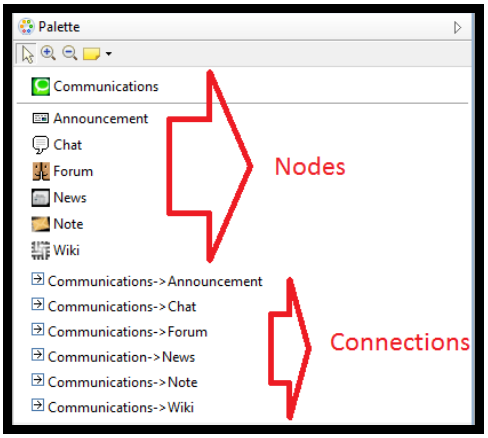


Figura 23.

Para crear los modulos basta con arrastrar los nodos de la “Pallete” al área de trabajo, rellenar los campos y conectarlos respetando las siguientes reglas:

- Un cursos solo puede tener un modulo de “Communications”.
- El modulo de “Communications” tiene cero o muchos: “Announcements”, “Chat”, “Forum”, “News”, “Note”, y “Wiki”.

La herramienta valida los siguientes casos:

- Solo permite un modulo de “Communications”.
- Los enlaces solo pueden corresponder entre el nodo “Communications” y su respectiva herramienta, por ejemplo “Communications -> Chat “, solo sirve para conectar el nodo “Communications” con el nodo “Chat”, la herramienta no permite utilizarlo en otro caso.
- Cuando se genere el código a desplegar, aquellos nodos que estén sueltos (sin conexión) no serán tenidos en cuenta para la creación del curso.

Es muy importante considerar que los elementos modelados con esta herramienta se desplegaran en un solo tema del curso, con lo cual es substancial considerar los nombre de los campos en cada elemento (Nodo o Modulo) como genéricos, para no tener que cambiar el modelo cada vez que se despliegue sobre el curso.

A continuación se muestra la creación de los módulos para un tema de un curso, Figura 24.

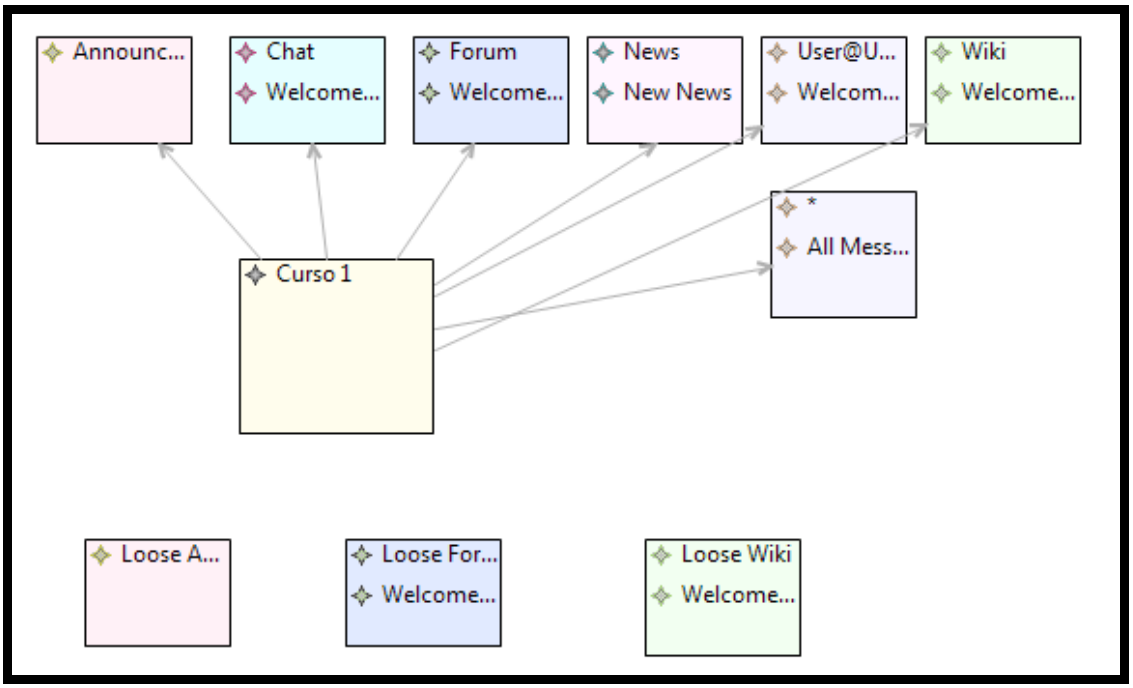


Figura 24.

En la Figura 24 se pueden visualizar los módulos que están conectados con sus respectivas relación y los que no tiene conexión, estos últimos no serán desplegados en el tema del curso.

Observaciones a tener en cuenta en el modelado:

- Si se desea enviar una “Note” a todos los inscritos en un curso, en el campo “Send To” debe ir un *.
- Si se desea enviar una “Note” a un solo inscrito en un curso, en el campo “Send To” debe ir un el correo completo con el que la persona se inscribió en la plataforma virtual. Si este correo no corresponde con el de la persona el mensaje no será enviado.
- Recuerde diligenciar todos los campos en los nodos, pues aquellos campos en los cuales el usuario no ponga información serán tomados en blanco y se procesaran erradamente.
- **Todos los campos aparecen con información por defecto pero esta información solo es de carácter orientativo, No implica información en los nodos.**

Guarde todos los cambios y envié al administrador de la aplicación los ficheros generados, para este caso de ejemplo “default.modellms” y “default.modellms_diagram”.

Una vez reciba respuesta del Administrador, ingrese a claroline con su perfil de profesor, luego ingrese al curso en el cual desea desplegar los módulos creados con la herramientas DSL, deberá aparecer una nueva opción en los módulos llamada *“Plantilla DSL para Claroline”,* para

desplegar los módulos sencillamente de clic sobre esta nueva opción y los módulos se crearan acorde al modelo hecho con la herramienta DSL. Figuras 25.

Figura 25.

Una vez seleccionada esta opción los elementos modelados serán desplegados en la plataforma, Figura 26.

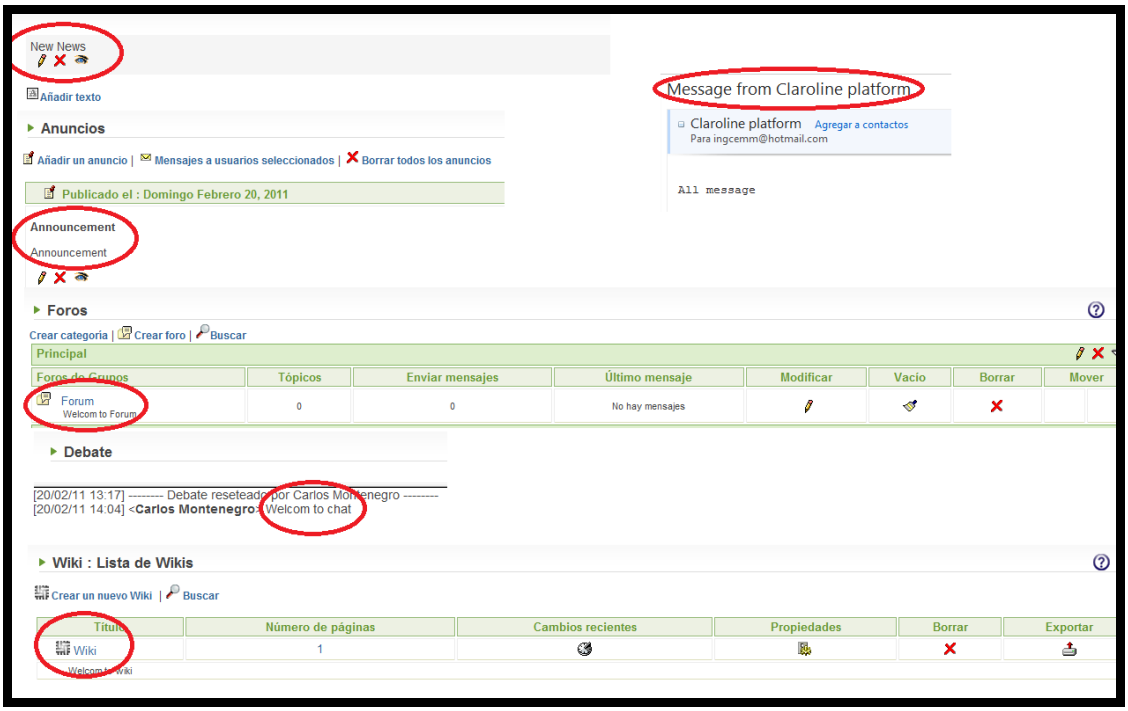


Figura 26.

Guía para el Administrador de la herramienta que realiza un modelado específico de dominio para la construcción de módulos de LMS independientes de la plataforma y claroline.

Una vez el Administrador reciba los ficheros enviado por el profesor, para este caso de ejemplo "default.modellms" y "default.modellms_diagram":

1. Cambie la extensión del fichero "default.modellms" a "default.ecore".
2. Abra eclipse y aplique las transformaciones MOFScrip que tiene el fichero "modellmsTransformationClaroline.m2t".
3. Esta transformación generan dos ficheros "entry.php" y "plantillasdsl.lib.php", comprímalas en un archivo que se llamara "UOPDSL2.zip" con la siguiente estructura:
    - \entry.php
    - \manifest.xml
    - \lib\plantillasdsl.lib.php
4. Abra Claroline en modo Administrador y cargue el nuevo modulo, para ello valla a la opción "Administración de la plataforma" del menú, como lo muestra la figura27.

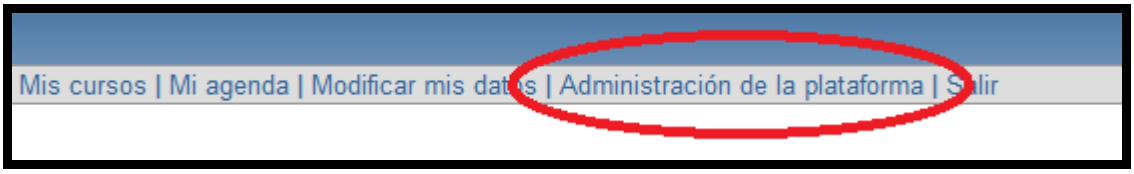


Figura 27.

5. En el nuevo menú seleccione "Módulos", figura 28.

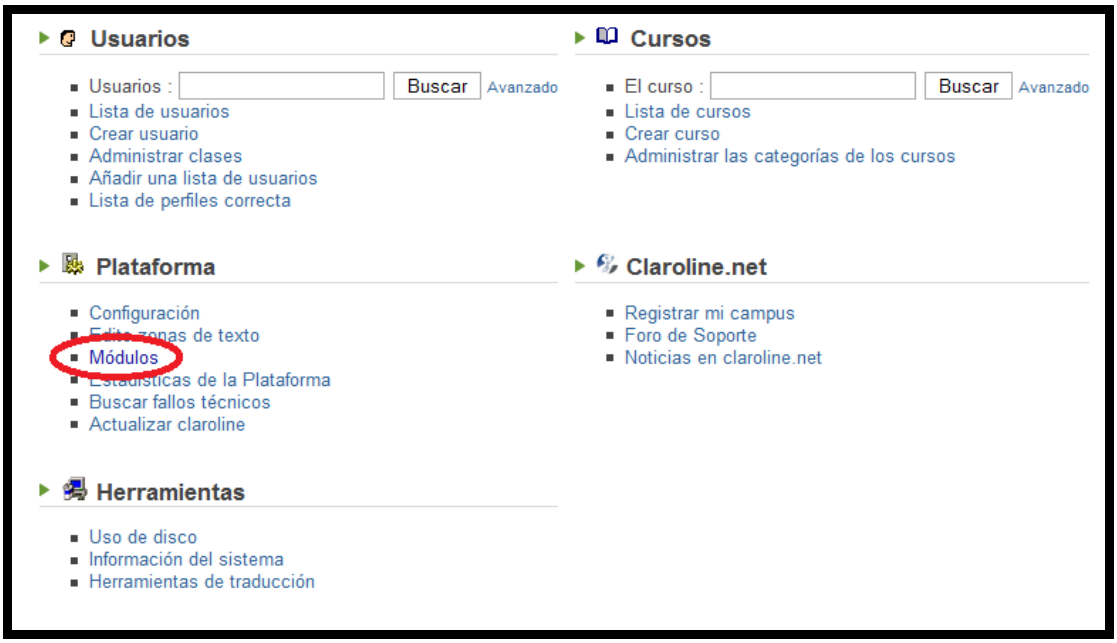


Figura 28.

6. En esta nueva ventana de clic sobre el vinculo “Instale un módulo”, luego en el botón examinar busque el archivo comprimido “UOPDSL2.zip” ábralo, de clic sobre el botón “Instale un módulo”, figura 29.

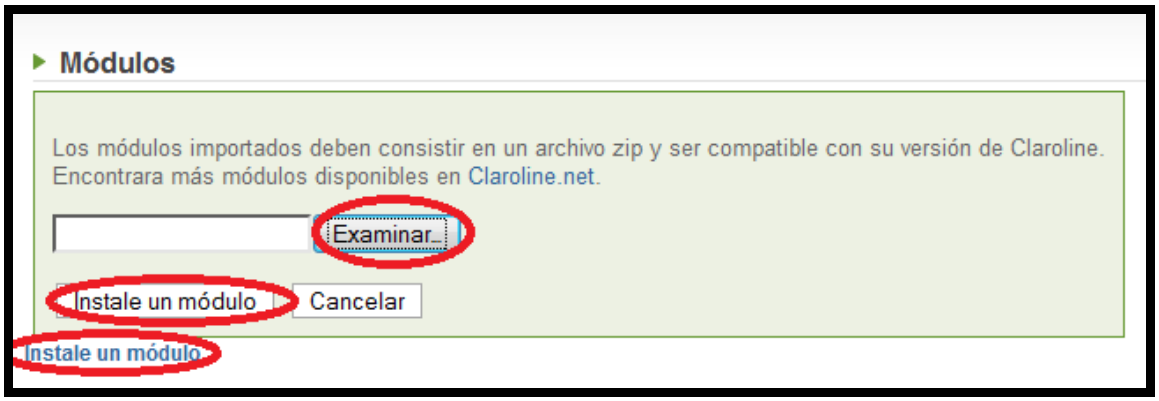


Figura 29.

Si la instalación ha sido exitosa aparecerá un mensaje informado que el modulo ha sido instalado y este se desplegara en el listado de modulos disponibles, para que el profesor lo pueda visualizar debe activar el modulo para ello de clic sobre la opción activado del modulo , Figura 30.

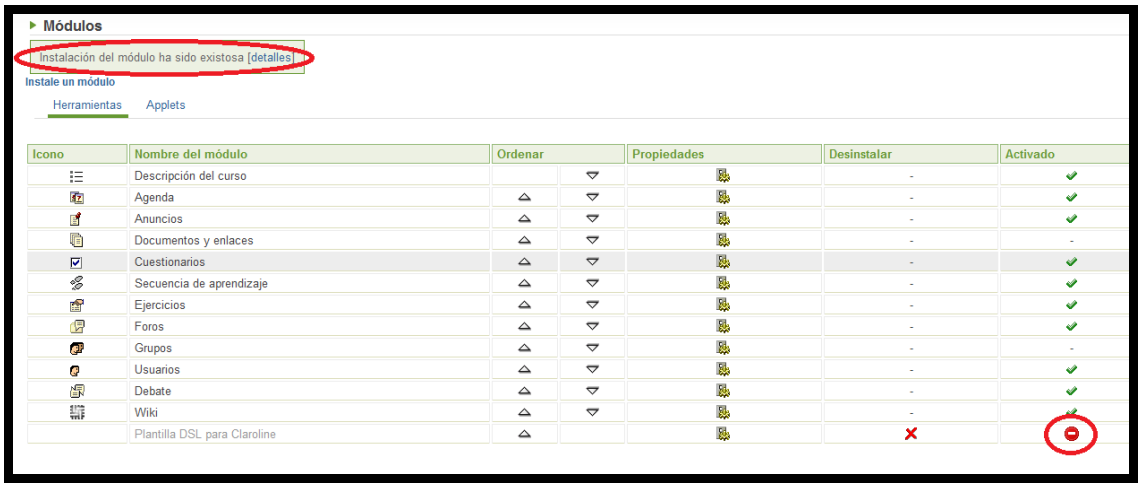


Figura 30.

Con estos pasos ya el profesor podrá ver el modulo con todo lo que el creo en claroline y lo podrá desplegar.

9 Acronimos

ATUTOR: Es un sistema de gestión del aprendizaje (LMS) Web abierto

.JAR: Java Archive

.ZIP : Formato de almacenamiento con compresión de datos

Acceleo: Tecnología para convertir modelos a texto

ACS: Architecture Community System

ADL: Advanced Distributed Learning

AICC: Aviation Industry CBT Comitee

AOL: American On Line

ATL: Atlas Tranformation Language

ATRC: Adaptive Technology Resource Centre

Blackboard: Es un sistema de gestión del aprendizaje (LMS), de uso comercial

B-learning: Blended-learning

BSI: British Standards

CAM: Contenido del modelo de agregación

CBT: Computer Based Training

CIM: Modelo Independiente de la Computación

CiteSeer: Es un motor de búsqueda público y biblioteca digital enfocado en publicaciones académicas y científicas.

CLAROLINE: Es un sistema de gestión del aprendizaje (LMS) Web abierto

CMaptools: Es una herramienta para confeccionar esquemas conceptuales

CMS: Content Management System

DAM + OIL: Unificación de los lenguajes DAML y OIL

DAML : DARPA Agent Markup Language

DARPA: Defense Advanced Research Projects Agency)

Dialnet: Portal de difusión de la producción científica hispana

D-Learning: Distance learning

DOTLRN: Es un sistema de gestión del aprendizaje (LMS) Web abierto

DSDM: Desarrollo de Software Dirigido por Modelos

DSL: Domain Specific Language

DSM: Modelado de Dominio Especifico

EBNF: Extended Backus–Naur Form

EBNF MS: Extended Backus–Naur Form Model Space

Ebsco: Es una base de datos que ofrece textos completos, índices y publicaciones periódicas.

Eclipse: Es un entorno de desarrollo integrado de código abierto multiplataforma

ECore: Eclipse Core

E-Learning: Electronic Learning

EMF/GEF: Eclipse Graphical Editing Framework

EMF: Eclipse Modeling Framework

e-working: Electronic working

FAQ: Frequently asked questions

FreeMind: Herramienta de software libre que permite la elaboración de mapas mentales o conceptuales

FSF: Free Software Foundation

GEF: Graphical Editing Framework

GMF Tooling: Graphical Modeling Framework Tooling

GMF: Graphical Modeling Framework

GMP: Graphical Modeling Project

GNU: GNU No es Unix

Google Schoolar: Buscador de Google especializado en artículos de revistas científicas

GOPRR: Grafo, Objeto, Propiedad, Relación, y Rol

GPL: General Purpose Language

HTML: HyperText Markup Language

IBM: International Business Machines

IBT: Internet Based Training

IEEE: Institute of Electrical and Electronics Engineers

IHMC: Institute for Human and Machine Cognition

ILIAS: Sistema de gestión para la enseñanza (LMS), desarrollado en código abierto

IMS 1.1.3: Versión 1.1.3 Instructional Management Systems

IMS CP: Especificación Content Packaging del Instructional Management Systems

IMS LD: Especificación Learning Design del Instructional Management Systems

IMS QTI 1.2: Especificación Question and Test Interoperability specification del Instructional Management Systems version 1.2

IMS QTI 2.0: Especificación Question and Test Interoperability specification del Instructional Management Systems version 2.0

IMS: Instructional Management Systems

IMS-CP: Especificación Content Packaging del Instructional Management Systems

IMS-LRMD: Especificación Learning Resource Meta Data del Instructional Management Systems

IMS-MD: Especificación Model Data del Instructional Management Systems

IMS-QTI: Question and Test Interoperability specification del Instructional Management Systems

IPTV: Internet Protocol Television

ISI: ISI Web of Knowledge

ISM: Implementation Specific Model

ISO IEC 9126: Estándar internacional para la evaluación del Software

ISO: International Organization for Standardization

Jet: Java Emitter Templates

JOIN: Proyecto europeo para evaluar la calidad de las plataformas de tele-enseñanza (LMS) de software libre

KIF: Knowledge Interchange Formalism

KiwiDSM: Herramienta DSL para la construcción de módulos independientes de la plataforma para LMS

KiwiDSM v1.0: Herramienta DSL para la construcción de módulos independientes de la plataforma para LMS versión 1.0

KiwiDSM v2.0: Herramienta DSL para la construcción de módulos independientes de la plataforma para LMS versión 2.0

LAMS: Learning Activity Management System

-

LD: Learning Design

LM: Language Modeling

LMML: Learning Material Markup Language (LMML)

LMS: Learning management system

LMS-RTE3: especificaciones Runtime Environment

LOM: Learning Object Metadata

Loom: Knowledge Representation Language

LTSC: Learning Technology Estándards Committee

M2M: Model to Model

M2T: Model to Text

MDA: Model Driven Architecture

MDE: Model-driven engineering

MIT: Massachusetts Institute of Technology

M-Learning: Mobile Learning

MOF MS: Meta-Object Facility Model Space

MOF: Meta-Object Facility

MOFScript: Lenguaje para transformaciones de Modelo a texto

MOODLE: Module Object-Oriented Dynamic Learning Environment

MS: Modeling Space

MySQL: sistema de gestión de base de datos relacional libre

ODM: e Ontology Definition Metamodel

OIL: Ontology Inference Layer or Ontology Interchange Language

OLAT: Online Learning And Trainin

OMG: Object Management Group

Ontolingua: lenguaje basado en KIF

OpenACS: Open Architecture Community System

OperationalQVT: Operationa Query/View/Transformation

Oracle: U n sistema de gestión de base de datos objeto-relacional

OVA: Objetos Virtuales de Aprendizajes

OWL: Ontology Web Language

OWLViz: Plugin de Protege para visualizar grafos

PHP: Personal Home Page Tools

PIF: Package Interchange File

PIM: Modelo Independiente de la Plataforma

PostgreSQL: Es un sistema de gestión de base de datos relacional orientada a objetos y libre

Proquest: Es un recurso de colecciones electrónicas

Protege: Framework para la construcción de ontologías.

PSM: Modelo Especifico de la Plataforma

QTI: Question and Test Interoperability

QVT: Query View Transformation

RDF: Resource Description Framework

RDF Schema: Esquema de Resource Description Framework

RDFS: RDF Schema

ReLOAD: Reusable eLearning Object Authoring & Delivery

RLO: Reusable Learning Object

RTE: Run-Time Environment

SCO: Sharable Content Objects

SCORM 1.2: paquetes de contenido

SCORM: Shareable Content Object Reference Model

SE: Standart Edition

SHOE: Lenguaje para describir Ontologías

SIG: Open-Source Software for Education in Europe

SN: Sequencing and Navigation

SQL: Structured Query Language

TIC's: Tecnologías de la Información y las Comunicaciones

T-learning: Televisión Learning

TML/Netquest: Tutorial Markup Language/ Netquest

TS: Technical Spaces

UML: Unified Modeling Language

UNESCO: Organización de las Naciones Unidas

UNSPSC : United Nations Standard Products and Services Code

UoL: Unidades de Aprendizaje

VHDL: VHSIC hardware description language

W3C: World Wide Web Consortium

WBT: Web Based Training

WCAG 1.0: Web Content Accessibility Guidelines 1.0

WebCT: Web Course Tools

XFML: eXtensive Faceted Markup Language

XMI: XML Metadata Interchange

XML: Extensible Markup Language

XML Schemas: Esquema Extensible Markup Language

XMLDB: XML Databases

XOL: XML-based ontology-exchange language

Xpand: Code Generation

XSD: XML Schema Definition

ZAKAI: Es un sistema de gestión del aprendizaje (LMS) Web abierto basado en Java.

10 Referencias

ADVANCED DISTRIBUTED LEARNING. 2004. *SCORM 2004 Sharable Content Object Reference Model* [Online]. Available: http://www.adlnet.org/Pages/Default.aspx [Accessed April 2010].

ÁLVAREZ, G. L. A., ESPINOZA, P. D. P. & BUCAREY, A. S. G. 2006. Empaquetamiento y Visualización de Objetos de Aprendizaje SCORM en LMSs de Código Abierto. Valdivia: Universidad Austral de Chile.

ALVAREZ, M. 2010. *Diseño y construcción de lenguajes especificos de dominío asistido por Ontologías.* Master, Universidad de Oviedo.

AMORIM, R. R., LAMA, M., SÁNCHEZ, E., RIERA, A. & VILA, X. A. 2006. A Learning Design Ontology based on the IMS Specification. *Educational Technology & Society,* 9**,** 38-57.

ANASTASAKIS, K., BORDBAR, B., GEORG, G. & RAY, I. 2007. UML2Alloy: A Challenging Model Transformation. *In:* ENGELS, G., OPDYKE, B., SCHMIDT, D. & WEIL, F. (eds.) *Model Driven Engineering Languages and Systems.* Springer Berlin / Heidelberg.

ASUNCI, N, G., MEZ, P. & REZ 2002. Ontology Specification Languages for the Semantic Web. *In:* OSCAR, C. (ed.) *IEEE Intelligent Systems.*

ATKINSON, C. & KUHNE, T. 2003. Model-driven development: a metamodeling foundation. *Software, IEEE,* 20**,** 36-41.

ATUTOR. 2010. *Atutor Learning Management Tools* [Online]. Available: http://atutor.ca/development/ [Accessed Feb 2011].

AUSUBEL, D. P., NOVAK, J. D. & H., H. 1978. *Educational Psychology: A Cognitive View (2a edición),* New York:.

ÁVAREZ, V. 2010. *Voice Interative Classroom, a service-oriented software architecture to enable cross-platform multi-chanel access to Internet-based learning.* University of Oviedo.

BECHHOFER, S., GOBLE, C. & HORROCKS, I. 2001. DAML+OIL is not enough. In Proceedings of the First Semantic Web Working Symposium (SWWS'01).

BERNERS-LEE, T. 2005. Random reflections on ISWC, and busting some myths and a few challenges. *In:* BIOPAX (ed.). CSAIL, MIT.

BIZONOVA, Z. & RANC, D. Year. Model Driven LMS Platform Integration. *In:* Telecommunications, 2007. AICT 2007. The Third Advanced International Conference on, 2007. 25-25.

BIZONOVA, Z. & RANC, D. Year. Interoperability and Reuse Between Systems in eLearning. *In:* LUCA, J. & WEIPPL, E. R., eds. World Conference on Educational Multimedia, Hypermedia and Telecommunications 2008, 2008 Vienna, Austria. AACE, 1700-1705.

BIZOŇOVÁ, Z., RANC, D. & DROZDOVÁ, M. Year. Model Driven E-Learning Platform Integration. *In:* MAILLET, K., KLOBUCAR, T., GILLET, D. & KLAMMA, R., eds. 2nd European Conference on Technology Enhanced Learning EC-TEL PROLEARN 2007 Doctoral Consortium, 2007 Crete, Greece. CEUR-WS.org, 8-14.

BONEU, J. 2007. Plataformas abiertas de e-learning para el soporte de contenidos educativos abiertos. *Revista de Universidad y Sociedad del Conocimiento (RUSC), Universidad Oberta de Catalunya,* 4**,** 36 - 47.

BOTICARIO, J. G. & SANTOS, O. C. 2007. An Open IMS-Based User Modelling Approach for Developing Adaptive Learning Management Systems. *Journal of Interactive Media in Education*.

BRICKLEY, D. 1996. *Towards an open question-interchange framework* [Online]. Available: http://www.ilrt.bris.ac.uk/netquest/about/lang/motivation.html [Accessed Jun 2010].

BRINE, J., WILSON, I. & ROY, D. 2007. *Using moodle and other software tools in EFL courses in a Japanese IT university,* Los Alamitos, Ieee Computer Soc.

BUDINSKY, F., STEINBERG, D., PATERNOSTRO, M. & MERKS, E. 2009. *EMF: Eclipse Modeling Framework*, Addison-Wesley.

BURGOS D. 2006. *Estudio de la estructura y del comportamiento de las comunidades virtuales de aprendizaje no formal sobre estandarización del e-learning.* Universidad Europea de Madrid.

BURGOS, D., TATTERSALL, C. & KOPER, R. Year. Utilización de estándares en el aprendizaje virtual. *In*, 2005. University Complutense of Madrid.

BYRD, T. A., COSSICK, K. L. & ZMUD, R. W. 1992. A synthesis of research on requirements analysis and knowledge acquisition techniques. *MIS Q.,* 16**,** 117-138.

CANAS, A. J., CARFF, R., HILL, G., CARVALHO, M., ARGUEDAS, M., ESKRIDGE, T. C., LOTT, J. & CARVAJAL, R. 2005. Concept maps: Integrating knowledge and information visualization. *In:* TERGAN, S. O. & KELLER, T. (eds.) *Knowledge and Information Visualization: Searching for Synergies.* Berlin: Springer-Verlag Berlin.

CANAS, A. J., VALERIO, A., LALINDE-PULIDO, J., CARVALHO, M. & ARGUEDAS, M. 2003. Using WordNet for word sense disambiguation to support concept map construction. *In:* NASCIMENTO, M. A., DEMOURA, E. S. & OLIVEIRA, A. L. (eds.) *String Processing and Information Retrieval, Proceedings.* Berlin: Springer-Verlag Berlin.

CENT UNIVERSITAT JAUME I. 2004. *MoodleMoot Spain 2004* [Online]. Castelló. Available: http://cent.uji.es/pub/node/245 [Accessed April 2010].

CIOCOIU, M., NAU, D. S. & GRUNINGER, M. 2001. Ontologies for Integrating Engineering Applications. *Journal of Computing and Information Science in Engineering,* 1**,** 12-22.

CMAPTOOLS, I. 2010. *IHMC CmapTools* [Online]. Available: http://cmap.ihmc.us/ [Accessed October 2010].

CODE, T. U. N. S. P. A. S. 2010. *The United Nations Standard Products and Services Code UNSPSC* [Online]. Available: http://www.unspsc.org/ [Accessed Nov 2010].

COMMONWEALTH OF LEARNING (COL). 2003. *COL LMS Open Source* [Online]. Vancouver, Canada, . Available: http://www.col.org/Pages/default.aspx [Accessed 2010].

CONSORCIO CLAROLINE. 2008. *Claroline* [Online]. Available: http://www.claroline.net [Accessed October 2010].

CONSORTIUM, G. L. 2003a. *IMS Simple Sequence. Version 1.0 Final Specification* [Online]. Available: http://www.imsglobal.org/simplesequencing/index.html [Accessed April 2010].

CONSORTIUM, G. L. 2010. *IMS Global* [Online]. Available: http://www.imsglobal.org [Accessed April 2010].

CONSORTIUM, I. G. L. 2003b. *Learning Design Specification* [Online]. Available: http://www.imsglobal.org/learningdesign/ [Accessed Jun 2010].

CHANDRASEKARAN, B., JOSEPHSON, J. R. & BENJAMINS, V. R. 1999. What Are Ontologies, and Why Do We Need Them? *IEEE Intelligent Systems,* 14**,** 20-26.

CHIEN, L. R., BUEHRE, D. J. & IEEE 2008. A Visual Lambda-Calculator Using Typed Mind-Maps. *In:* XIE, Y., LI, W. & ZHOU, J. (eds.) *Iccee 2008: Proceedings of the 2008 International Conference on Computer and Electrical Engineering.* Los Alamitos: Ieee Computer Soc.

DEVEDZI\, V. & \#263 2002. Understanding ontological engineering. *Commun. ACM,* 45**,** 136-144.

DÍAZ-ANTÓN, G. & PÉREZ, M. 2005. Hacia una ontología sobre LMS. *In:* COLIMA, U. D. (ed.) *Hacia una ontología sobre LMS.* Colima, México: VII Jornadas Internacionales de las Ciencias Computacionales. Universidad de Colima.

DÍAZ-ANTÓN, G. & PÉREZ, M. A. 2006. TOWARDS AN ONTOLOGY OF LMS A Conceptual Framework. *In:* MANOLOPOULOS, Y. & FILIPE, J. (eds.) *8th International Conference on Enterprise Information Systems.* Paphos - Cyprus.

DODERO, J. M., DEL VAL, Á. M. & TORRES, J. 2010. An extensible approach to visually editing adaptive learning activities and designs based on services. *Journal of Visual Languages & Computing,* 21**,** 332-346.

DZEMYDIENE, D., TANKELEVICIENE, L. & DZEMYDA, I. 2007. AN APPROACH FOR MANAGING EDUCATIONAL RESOURCES IN AN ADAPTIVE E-LEARNING SYSTEM. *International Journal of Information and Communication Technology Education***,** 200.

EDUTECH. 2000. *Edutech* [Online]. University of Fribourg. Available: http://www.edutech.ch/ [Accessed April 2010].

EDUTOOLS. 2010. *CMS: Product Comparison System* [Online]. Available: http://www.edutools.info/item_list.jsp?pj=4

http://www.edutools.info/compare.jsp?pj=4&i=624,560,562,616,621 [Accessed October 2010].

FENSEL, D., VAN HARMELEN, F., HORROCKS, I., MCGUINNESS, D. L. & PATEL-SCHNEIDER, P. F. 2001. OIL: an ontology infrastructure for the Semantic Web. *Intelligent Systems, IEEE,* 16**,** 38-45.

FOUNDATION, T. E. 2007. *bug in map generation of the GMF tutorial* [Online]. Available: https://bugs.eclipse.org/bugs/show_bug.cgi?id=189410 [Accessed January 2011].

FOUNDATION, T. E. 2010a. *ATL* [Online]. Available: http://www.eclipse.org/atl/ [Accessed Dec 2010].

FOUNDATION, T. E. 2010b. *GMF Tutorial* [Online]. Available: http://wiki.eclipse.org/GMF_Tutorial [Accessed Dec 2010].

FOUNDATION, T. E. 2010c. *Graphical Modeling Project (GMP)* [Online]. Available: http://www.eclipse.org/modeling/gmp/ [Accessed Dec 2010].

FOUNDATION, T. E. 2010d. *M2T-JET* [Online]. Available: http://wiki.eclipse.org/M2T-JET [Accessed Dec 2010].

FOUNDATION, T. E. 2010e. *MOFScript* [Online]. Available: http://www.eclipse.org/gmt/mofscript/ [Accessed Dec 2010].

FOUNDATION, T. E. 2010f. *XPand* [Online]. Available: http://wiki.eclipse.org/Xpand [Accessed Dec 2010].

FREEMAN, S. & PRYCE, N. 2006. Evolving an embedded domain-specific language in Java. *Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications.* Portland, Oregon, USA: ACM.

FREEMIND. 2010. *Freemind* [Online]. Available: http://freemind.sourceforge.net/wiki/index.php/Main_Page [Accessed 2010].

GAMMA, E., HELM, R., JOHNSON, R. & VLISSIDES, J. 2003. *Patrones de diseño. Elementos de software orientado a objetos reutilizable,* Madrid, 2003.

GARCÍA-DÍAZ, V., TOLOSA, J., G-BUSTELO, B., PALACIOS-GONZÁLEZ, E., SANJUAN-MARTÍNEZ, Ó. & CRESPO, R. 2009. TALISMAN MDE Framework: An Architecture for Intelligent Model-Driven Engineering. *In:* OMATU, S., ROCHA, M., BRAVO, J., FERNÁNDEZ, F., CORCHADO, E., BUSTILLO, A. & CORCHADO, J. (eds.) *Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living.* Springer Berlin / Heidelberg.

GARCIA-DIAZ, V., TOLOSA, J. B., G-BUSTELO, B. C. P., PALACIOS-GONZALEZ, E., SANJUAN-MARTINEZ, O. & CRESPO, R. G. 2009. TALISMAN MDE Framework: An Architecture for Intelligent Model-Driven Engineering. *In:* OMATU, S., ROCHA, M. P., BRAVO, J., FERNANDEZ, F.,

CORCHADO, E., BUSTILLO, A. & CORCHADO, J. M. (eds.) *Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living, Pt Ii, Proceedings.* Berlin: Springer-Verlag Berlin.

GARCÍA-MAGARIÑO, I., FUENTES-FERNÁNDEZ, R. & GÓMEZ-SANZ, J. J. 2009. Guideline for the definition of EMF metamodels using an Entity-Relationship approach. *Information and Software Technology,* 51**,** 1217-1230.

GARCÍA, A. 2010. *CADAM: Construcción de Aplicaciones Web, para el diseño de asignaturas basada en modelos.* Ingeniería técnica en informática, Universidad de Oviedo.

GARCÍA DÍAZ, V. & CUEVA LOVELLE, J. M. 2010. Ingeniería Dirigida por Modelos. Oviedo.

GEEKNET, I. 2010. *SourceForge* [Online]. Available: http://sourceforge.net/ [Accessed April 2010].

GENESERETH, M. R. & KETCHPEL, S. P. 1994. Software agents. *Commun. ACM,* 37**,** 48-ff.

GLOBAL LEARNING CONSORTIUM. 2001. *IMS Meta Data. Version 1.3 Final* [Online]. Available: http://www.imsglobal.org/metadata/index.html [Accessed April 2010].

GLOBAL LEARNING CONSORTIUM. 2005. *Content Package. Version 1.2. Public Draft Specification* [Online]. Available: http://www.imsglobal.org/content/packaging/index.html [Accessed April 2010].

GONZÀLEZ, A. 2009. *Guìa de apoyo para el uso moodle 1.9.4 Usuario Desarrollador.* Universidad de Oviedo.

GROB, H. L., BENSBERG, F. & DEWANTO, B. L. 2010. eleed - Model Driven Architecture (MDA): Integration and Model Reuse for Open Source eLearning Platforms.

GROUP, O. M. 2007. MOF 2.0/XMI Mapping, Version 2.1.1. Object Management Group.

GUIZZARDI, G. 2005. *Ontological foundations for structural conceptual models.*

GUTIERREZ S. 2007. *Secuenciamiento de actividades educativas orientado a la reutilización y la auto-organización en tutoría inteligente.* Universidad Carlos III de Madrid.

HEFLIN, J. & HENDLER, J. 2001. A portrait of the Semantic Web in action. *Intelligent Systems, IEEE,* 16**,** 54-59.

HEFLIN, J., HENDLER, J. & LUKE, S. 1999. SHOE: A Knowledge Representation Language for Internet Applications.

HEIYANTHUDUWAGE, S. R. & KARUNARATNE, D. D. 2006. A Learner Oriented Ontology of Metadata to Improve Effectiveness of Learning Management Systems. *International Journal of the Computer, the internet and management,* 14.

HENDERSON-SELLERS, B. 2011. Bridging metamodels and ontologies in software engineering. *Journal of Systems and Software,* 84**,** 301-313.

HENDLER, J. 2001. Agents and the Semantic Web. *Intelligent Systems, IEEE,* 16**,** 30-37.

HOLZINGER, A., KLEINBERGER, T. & MÜLLER, P. Year. Multimedia Learning Systems based on IEEE Learning Object Metadata (LOM). *In:* MONTGOMERIE, C. & VITELI, J., eds. World Conference on Educational Multimedia, Hypermedia and Telecommunications 2001, 2001 Norfolk, VA. AACE, 772-777.

IEEE 2002. Draft Standard for Learning Object Metadata. New York: Institute of Electrical and Electronics Engineers, Inc.

IVORRA, R. 2009. *Tutorial: Creaciòn de un mòdulo actividad. Moodle (1.9.3).*

JENSEN, K. 1991. Coloured petri nets: A high level language for system design and analysis. *In:* ROZENBERG, G. (ed.) *Advances in Petri Nets 1990.* Springer Berlin / Heidelberg.

JOUAULT, F., ALLILAIRE, F., BÉZIVIN, J. & KURTEV, I. 2008. ATL: A model transformation tool. *Science of Computer Programming,* 72**,** 31-39.

JOUAULT, F. & BÉZIVIN, J. 2006. KM3: A DSL for Metamodel Specification. *In:* GORRIERI, R. & WEHRHEIM, H. (eds.) *Formal Methods for Open Object-Based Distributed Systems.* Springer Berlin / Heidelberg.

KARP, P. D., CHAUDHRI, V. K. & THOMERE, J. F. 1999. XOL: An XML-Based Ontology Exchange Language. 333 Ravenswood Ave., Menlo Park, CA 94025: AI Center, SRI International.

KLEPPE, A., WARMER, J. & BAST, W. 2003. *MDA Explained: The Model Driven Architecture™: Practice and Promise* Addison Wesley.

LASSILA, O. & SWICK, R. 1998. Resource Description Framework (RDF) model and syntax specification. The World Wide Web Consortium.

LLORENTE-CEJUDO, M. C. 2007. Moodle como entorno virtual de formación al alcance de todos. *Comunicar,* 28**,** 197-202. .

MACGREGOR, R. M. 1991. Inside the LOOM description classifier. *SIGART Bull.,* 2**,** 88-92.

MÁRQUEZ, V. J. M. 2007. *Estado del arte del eLearning. Ideas para la definición de una plataforma universal.* Universidad de Sevilla.

MCGREAL, R. 2004. Learning Objects: A Practical Definition. *International Journal of Instructional Technology and Distance Learning* 1.

MCILRAITH, S. A., SON, T. C. & HONGLEI, Z. 2001. Semantic Web services. *Intelligent Systems, IEEE,* 16**,** 46-53.

MICROSYSTEMS, S. 1997. Java Code Conventions. *Java Code Conventions.* Sun Microsystems: Sun Microsystems.

MICHAEL, K. 2002. Interoperable Community Platforms and Identity Management in the University Domain.

MONTAÑEZ, S. 2008. *MODELO DE PORTAL PARA EL ACCESO A UNA GRID DE GEOSENSORES.* Universidad Distrital.

MOODLE. 2011. *Moodle* [Online]. Available: http://moodle.org/ [Accessed Feb 2011].

MORENO, N. & ROMERO, J. R. 2005a. A MDA-based framework for building interoperable e-learning platforms. *In:* MÉNDEZ-VILAS, A., GONZÁLEZ-PEREIRA, B., J.A.M., J. M. G. A., GONZÁLEZ & FORMATEX, E. (eds.) *Recent Research Developments in Learning Technologies (2005).* Badajoz, Spain (2005).

MORENO, N. & ROMERO, J. R. 2005b. Recent Research Developments in Learning Technologies (2005). *In:* MÉNDEZ-VILAS, A., GONZÁLEZ-PEREIRA, B., GONZÁLEZ, J. M. & GONZÁLEZ, J. A. M. (eds.) *A MDA-based framework for building interoperable e-learning platforms.* Badajoz, Spain: FORMATEX, Badajoz, Spain (2005).

MOTIK, B., PATEL-SCHNEIDER, P. F., PARSIA, B., BOCK, C., FOKOUE, A., HAASE, P., HOEKSTRA, R., HORROCKS, I., RUTTENBERG, A., SATTLER, U. & SMITH, M. 2008. OWL 2 Web Ontology Language: structural specification and functional-style syntax. W3C.

MUÑOZ-MERINO, P. J., KLOOS, C. D. & NARANJO, J. F. 2009. Enabling interoperability for LMS educational services. *Computer Standards & Interfaces,* 31**,** 484-498.

NATALYA, F. N. & DEBORAH, L. M. 2005. Ontology Development 101: A Guide to Creating Your First Ontology. Stanford,.

NETSCAPE. 2002. *DMOZ Open Directory Project* [Online]. Available: http://www.dmoz.org/ [Accessed Nov 2010].

NOVAK, J. D. & GOWIN, D. B. 1984. *Learning How to Learn,* New York, Cambridge University Press.

OBEO. 2010. *Acceleo* [Online]. Available: http://www.eclipse.org/acceleo/

http://www.acceleo.org/pages/home/en [Accessed Dec 2010].

OBJECT MANAGEMENT GROUP, I. 2011. *Documents associated with Meta Object Facility (MOF) 2.0 Query/View/Transformation, v1.0* [Online]. Available: http://www.omg.org/spec/QVT/1.0/ [Accessed Feb 2011].

OFFICE., D. S. I. E. 2000. *The DARPA Agent Markup Language Homepage* [Online]. Available: http://www.ai.sri.com/daml/ontologies/ [Accessed Nov 2010].

OLDEVIK, J. 2009. *MOFScript User Guide Version 0.8 (MOFScript v 1.3.6).*

OMG 1999. OMG Unified Modeling Language Specification, Version 1.3.

OTHERS, I. C. A. 2006. *Package org.eclipse.emf.ecore* [Online]. Available: http://download.eclipse.org/modeling/emf/emf/javadoc/2.6.0/org/eclipse/emf/ecore/package-summary.html [Accessed Dec 2010].

PARDILLO, J. & CACHERO, C. 2010. Domain-specific language modelling with UML profiles by decoupling abstract and concrete syntaxes. *Journal of Systems and Software,* 83**,** 2591-2606.

PATIL, R. S., FIKES, R. E., PATEL-SCHNEIDER, P. F., MCKAY, D., FININ, T., GRUBER, T. & NECHES, R. 1998. The DARPA knowledge sharing effort: progress report. *In:* MICHAEL, N. H. & MUNINDAR, P. S. (eds.) *Readings in agents.* Morgan Kaufmann Publishers Inc.

PBWORKS. 2007. *Estándares eLearning* [Online]. Available: http://internet-educativa.pbworks.com/Est%C3%A1ndares-eLearning [Accessed October 2010].

PERNIU, L., RASNOVEANU, C. & NICOLAE, G. (eds.) 2006. *ELMSET PROJECT*, BULGARIA: TECHNICAL UNIVERSITY OF SOFIA, BULGARIA.

PETERSON, J. 1981. *Petri Net Theory and the Modeling of Systems*, Prentice Hall PTR.

POLANSKY, D. 2010. *FreeMind - free mind mapping software* [Online]. Available: http://freemind.sourceforge.net/wiki/index.php/Main_Page [Accessed October 2010].

RIUS, À., SICILIA, M. Á. & GARCÍA, E. 2007. Justificación y Descripción del Dominio de Conocimiento de una Ontología para la Formalización y Automatización de Escenarios Educativos. *In:* BENITO, M. & ROMO, J. (eds.) *El IV Simposio Pluridisciplinar sobre Diseño, Evaluación y Desarrollo de Contenidos Educativos Reutilizables (SPDECE07).* Bilvao, spain.

RODRIGUEZ-ARTACHO, M., VERDEJO, M. E., MAYORGA, J. J. & CALERO, M. Y. Year. Using a high-level language to describe and create Web-based learning scenarios. *In:* Frontiers in Education Conference, 1999. FIE '99. 29th Annual, 1999 1999. 13A2/1-13A2/6 vol.2.

RODRIGUEZ, M. A., BARRIOS, J. D. & SCHULTZ, E. S. 2009. *THE USE OF AN INNOVATION CLASSROOM A Perspective in the Introduction of ICT in Elementary Schools,* Setubal, Insticc-Inst Syst Technologies Information Control & Communication.

ROMERO, M. & DE LARA, J. 2006. VALIDACIÓN Y VERIFICACIÓN DE INTERFACES DE USUARIO EN EL ÁMBITO DEL DESARROLLO BASADO EN MODELOS. *In:* BOTELLA, J. R.-P. (ed.) *XV Jornadas de Ingeniería del Software y Bases de Datos JISBD 2006.* CIMNE, Barcelona.

ROY, D. & IEEE 2008. *Using Concept Maps for Information Conceptualization and Schematization in Technical Reading and Writing Courses: A Case Study for Computer Science Majors in Japan,* New York, Ieee.

SIGOSSEE, O. S. F. E. I. E. G. 2010. *Evaluación de las plataformas LMS* [Online]. 2007. Available: http://www.guidance-research.org/sigossee/join/sp/ [Accessed October 2010].

SLAVIN, R. E. 1995. *Cooperative Learning: Theory, Research and Practice,* Boston.

SRIMATHI, H. 2010. Knowledge Representation of LMS using Ontology. *International Journal of Computer Applications,* Volume 6– No.3.

STANFORD CENTER FOR BIOMEDICAL INFORMATICS RESEARCH. 2010. *Protege Ontology Library* [Online]. Available: http://protegewiki.stanford.edu/wiki/Protege_Ontology_Library [Accessed Nov 2010].

STEPHEN, J. M., KENDALL, S., AXEL, U. & DIRK, W. 2004. *MDA Distilled: Principles of Model-Driven Architecture* Addison Wesley.

TOLVANEN, J. P., MARTTIIN, P. & SMOLANDER, K. Year. An integrated model for information systems modeling. *In:* System Sciences, 1993, Proceeding of the Twenty-Sixth Hawaii International Conference on, 5-8 Jan 1993 1993. 470-479 vol.3.

UMBC. 2007. *Swoogle* [Online]. Available: http://swoogle.umbc.edu/ [Accessed Nov 2010].

UNESCO. 2005. *Free & Open Source Software Portal* [Online]. Available: http://www.unesco-ci.org/cgi-bin/portals/foss/page.cgi?d=1 [Accessed April 2010].

UNIVERSITY, S. 2005. *Ontolingua* [Online]. 2005. Available: http://ksl.stanford.edu/software/ontolingua/ [Accessed Nov 2010].

VALIATHAN P. 2002. *Designing a Blended Learning Solution,* [Online]. Available: http://www.learningcircuits.org/2002/aug2002/valiathan.html [Accessed April 2010].

VÉLEZ R, J. B. 2007. *Arquitectura para la Integración de las Dimensiones de Adaptación en un Sistema Hipermedia Adaptativo.* Fabregat G. Ramón Proyecto de Investigación, Universitat de Girona.

VERBERT, K. & DUVAL, E. Year. Towards a global architecture for learning objects: a comparative analysis of learning object content models. *In:* ED-MEDIA 2004 World Conference on Educational Multimedia, Hypermedia and Telecommunications location:Lugano, Switzerland date:June 21-26, 2004, 2004.

W3C 2004. RDF Vocabulary Description Language 1.0: RDF Schema. Dan Brickley, W3C

WILEY D & EDWARDS K 2002. Online self-organizing social systems: The decentralized future of online learning. *Quarterly Review of Distance Education.*

WOUTERS, C. 2005. *A Formalization and Application of Ontology Extraction.* PhD Thesis, La Trobe University.

YAMADA, S., HISHITANI, J. & OSAKI, S. 1993. Software-reliability growth with a Weibull test-effort: a model and application. *Reliability, IEEE Transactions on,* 42**,** 100-106.

ZDUN, U. 2010. A DSL toolkit for deferring architectural decisions in DSL-based software design. *Information and Software Technology,* 52**,** 733-748.