Een systeem voor het assembleren van leerinhoud in een herbruikbaar leerobject

Gevalstudie : het LOMS

Bert MERTENS

promotor :
Prof. Jeanne SCHREURS

Eindverhandeling voorgedragen tot het bekomen van de graad
Handelsingenieur in de beleidsinformatica

universiteit hasselt

UNIVERSITEIT HASSELT

FACULTEIT TOEGEPASTE ECONOMISCHE WETENSCHAPPEN

Een systeem voor het assembleren van leerinhoud in een herbruikbaar leerobject

Gevalstudie: het LOMS

Eindverhandeling voorgedragen tot
het behalen van de graad van Handelsingenieur
door: Bert MERTENS
Promotor: Prof. dr. J. SCHREURS

2007

DANKWOORD

Graag zou ik mijn promotor professor Jeanne Schreurs willen bedanken voor de begeleiding, de raadgevingen en suggesties en de kansen die ze mij gegeven heeft. Ook de heer Maarten Steegmans zou ik willen bedanken voor het geduld en de snelle hulp bij technische problemen.

Verder zou ik iedereen die mij op één of andere manier geholpen heeft bij het uitwerken van dit onderzoek willen bedanken voor de tijd, de steun en/of daadwerkelijk hulp die ze mij gegeven hebben.

Als laatste zou ik u, als lezer van dit eindwerk, willen bedanken voor uw tijd. Ik wens u veel leesgenot.

ABSTRACT

Voor bedrijven en organisaties is het belangrijk om nieuwe kennis efficiënt aan te maken en bestaande kennis gemakkelijk te verspreiden. Dat de computer hierbij een handig hulpmiddel kan zijn, werd reeds in de beginjaren erkend.
Tijdens de jaren '90 creëerde men een theoretisch kader voor dit fenomeen en werd het begrip E-learning geïntroduceerd. E-learning kan in het kort omschreven worden als "het gebruiken van een computer en zijn netwerken om kennis aan te maken, te verspreiden en te beheren".

Om het concept E-learning in de realiteit te brengen was er nood aan systemen die deze taken ondersteunden. Deze systemen kregen de naam leerplatformen. De twee belangrijkste onderdelen waaruit een leerplatform kan bestaan zijn een LMS en een LCMS. Het 'learning management system' begeleidt de leeractiviteit en het 'learning content management system' beheert de leerinhoud. Ontwerpers van leerplatformen streven naar twee belangrijke voordelen, nl. herbruikbaarheid en interoperabiliteit. Herbruikbaarheid houdt in dat stukjes leerinhoud hergebruikt kunnen worden in andere leerinhouden. Met interoperabiliteit wordt bedoeld dat men leerinhoud tussen meerdere mensen en/of systemen kan uitwisselen.

Herbruikbaarheid en interoperabiliteit van een leerplatform wordt bereikt door leerobjecten te implementeren in leerplatformen. De leerinhoud of het oorspronkelijke leerdocument wordt opgesplitst in kleinere delen, waarna men ze apart opslaat. Later worden deze delen gerecombineerd of geassembleerd in leerobjecten en vormen ze dus nieuwe leermodules. Het stukje leerinhoud dat men kan hergebruiken noemt men een leerobject. Een leerobject bezit een eigen hiërarchie, die de relaties tussen de stukjes leerinhoud onderling beschrijft.
Om herbruikbaarheid te realiseren, moeten leerobjecten aan drie eisen voldoen, nl. discoverabiliteit, modulariteit en interoperabiliteit. Met discoverabiliteit bedoelt men dat het systeem het leerobject moet kunnen vinden en begrijpen. Modulariteit beoogt dat elk leerobject op zichzelf kan staan. Het heeft geen externe zaken nodig om te werken. Onder interoperabiliteit op het niveau van een leerobject verstaan we dat de onderdelen van het leerobject met meerdere systemen moet kunnen samenwerken.
Om de herbruikbaarheid en de interoperabiliteit van leerplatformen te verbeteren, kunnen ontwikkelaars standaarden hanteren, bij het ontwerpen van leerplatformen en creëren van leerobjecten, van organisaties zoals Dublin Core Metadata Initiative, AICC, IMS en ADL. Deze standaarden stellen o. a. voor: welke metadata te implementeren, een hiërarchie voor de leerobjecten en een aantal richtlijnen en best practices bij het realiseren van leerplatformen.

Een voorbeeld van een LCMS dat gebaseerd is op leerobjecten technologie, is het LOMS (learning object management system). Het systeem is ontwikkeld in het kader van onderzoek aan de Universiteit Hasselt. Het LOMS laat toe om leerinhoud aan te maken in de vorm van een mindmap (een leerobject). Het systeem werd in 2004 ontwikkeld en voldoet niet meer helemaal aan de eisen van de moderne leerplatformen technologie. De ingegeven leerinhoud kan op dit moment maar door één mindmap gebruikt worden. Ze kan ook niet aangepast worden aan de wensen en noden van de gebruiker, bijvoorbeeld 'Bezoekt hij de mindmap met een PDA of op een desktop?' of 'Is de presentatie van de mindmap aangepast aan zijn manier van leren: auditief, visueel of kinesthetisch?' Momenteel biedt het LOMS hiervoor geen bevredigende oplossing. Deze twee problemen, die het gevolg zijn van beperkte herbruikbaarheid, worden opgelost door het bouwen van een 'assembleer en authoring systeem'. Dit systeem bezit een 3-lagige architectuur en kan werken op de bestaande databank van het LOMS.
De eerste laag van deze architectuur zal de huidige relationele database zijn. Deze bevat de bestaande leerinhoud en regelt de toegang tot deze inhoud. De twee daaropvolgende lagen worden verder in dit eindwerk ontwikkeld.
De tweede laag krijgt de naam 'generic learning object assembling module' en zal het mogelijk maken om op basis van bestaande leerinhouden gestructureerd in de databanklaag, dynamische leerobjecten te creëren. Hiervan worden het ID en de metadata opgeslagen in de eerste laag. Zo zal het mogelijk zijn om de bestaande leerinhoud te hergebruiken in meerdere leerobjecten.
De derde laag zal de presentatie van de leerobjecten regelen. Door verschillende presentatievormen van eenzelfde leerobject toe te laten, kan het LOMS beter voldoen aan de wensen en de situatie van de gebruiker. Deze laag noemen we 'learning object creation module'.

Voordat de 3-lagige architectuur kan worden uitgewerkt, moet het bestaande systeem verder onderzocht en getoetst worden aan de eisen die gesteld worden in de literatuur. We gaan dieper in op de discoverabiliteit, interoperabiliteit, modulariteit en herbruikbaarheid van het bestaande systeem.
De discoverabiliteit wordt geëvalueerd aan de hand van de metadata die het systeem gebruikt. In het huidige LOMS zijn deze zeer uitgebreid en divers. Zo zijn er metadata opgenomen die het systeem moet gebruiken en metadata die de gebruiker kan bewerken.
Op het vlak van interoperabiliteit van het leerplatform is er echter nog ruimte voor verbetering. Het systeem werkt niet volgens een bepaalde gestandaardiseerde architectuur. Ook kunnen LO's niet worden opgeslagen in een file volgens algemene standaarden. De databank (MySQL) is wel vrij verkrijgbaar en toegankelijk voor iedereen. Dit geldt ook voor de leercontent zelf, deze werkt volgens algemene standaarden (pdf, doc, jpg,…).
De modulariteit kan als vrij goed bestempeld worden. De inhoud wordt opgebouwd volgens een vaste hiërarchie en de delen van deze hiërarchie (ALO's, Blocks,…) kunnen onafhankelijk gebruikt worden.
De herbruikbaarheid in het systeem is niet goed, omdat men geen nieuwe leerobjecten (mindmaps) kan maken, gebruik makend van bestaande leerinhoud.

Op basis van de informatie over het bestaande systeem en de 3-lagige architectuur wordt er een UML diagram ontwikkeld voor de systemen van de tweede en de derde laag. Hiervoor wordt beroep gedaan op OOA/D, een methode voor het ontwikkelen van programma's op een object georiënteerde manier. De omgeving van de systemen wordt eerst verkend met behulp van een 'use case' en een domeinmodel. Deze twee tools vertellen ons welke input het systeem krijgt en welke output verwacht wordt. Daarnaast geeft het een eerste idee over welke objecten kunnen opgenomen worden in het UML diagram. Na deze verkennende fase worden met behulp van interactiediagrammen de acties zelf en de manier waarop ze door het systeem afgehandeld worden verder uitgewerkt. Deze worden dan samengevat in een klassendiagram.

Aan de hand van de opgestelde diagrammen worden de twee programma's ontwikkeld. Hiervoor wordt de programmeertaal PHP gekozen. PHP is immers compatibel met andere operating systemen, kan goed samenwerken met internet browsers en ondersteunt de connectiviteit met MySQL zeer goed.
De generic learning object assembling module geeft de gebruiker de mogelijkheid om een nieuw leerobject aan te maken. Zij doet dit door een formulier aan de gebruiker te presenteren. Zo kan hij de onderdelen selecteren die in het leerobject moeten opgenomen worden. Het programma gaat dan een nieuw leerobject aanmaken. Het identificatienummer en de metadata van dit nieuwe object worden opgeslagen in de eerste laag, de huidige relationele database.
Het tweede programma, de learning object creation module vraagt aan de gebruiker welk leerobject hij wil zien en in welke lay-out. Het programma zal dan de gekozen combinatie generen en weergeven aan de gebruiker. Deze twee programma's laten de gebruiker toe om van de bestaande leerinhoud nieuwe leerobjecten aan te maken en op te slaan. Daarnaast komen ze ook tegemoet aan de wensen en de situatie van de gebruiker. We hergebruiken dus bestaande leerinhoud en geven gebruikers beter toegang tot de leerinhoud.

INHOUDSOPGAVE

LIJST VAN FIGUREN

LIJST VAN TABELLEN

1. PROBLEEMSTELLING

1.1 INLEIDING

In onze moderne maatschappij is leren belangrijk. Allerhande organisaties proberen met behulp van ICT het leerproces te bevorderen en te verbeteren. Hiervoor gebruiken ze systemen, die de aangeboden leercontent beheren, nl. leercontent management systemen (LCMS). Naast de LCMS gebruikt men ook learning management systemen (LMS). Zij beheren de leeractiviteit. Met beheren bedoelt men het aanmaken, het sturen en beschikbaar stellen van deze activiteit.
Aan de Universiteit Hasselt wordt in de cursussen beleidsinformatica, voor het beheren en het afstemmen van de leercontent, een LCMS systeem met de naam 'LOMS' gebruikt. LOMS staat voor 'learning object management system'. Dit systeem werd ontwikkeld in het kader van een onderzoek dat professor Schreurs voert en steunt op het principe van leerobjecten. Onder dit principe delen we leerstof op in kleine deeltjes. Deze deeltjes worden opgeslagen en gerecombineerd tot een nieuw stuk leerstof. Het voornaamste doel van het LOMS is het creëren, opslaan en verspreiden van nieuwe leerinhouden.
Het systeem werd ontwikkeld in 2004 toen de 'learning object' technologie nog in haar kinderschoenen stond. Het is daardoor niet aangepast aan de nieuwste ontwikkelingen op het gebied van leerobjecten.
Het LOMS laat de gebruiker toe om een mindmap aan te maken en te verspreiden. Een mindmap is een voorstelling van stukjes leercontent die samen één 'hoofdstuk' leerstof vormen. Een mindmap geeft door de manier van presenteren de relaties tussen de stukjes leerstof visueel weer en ze geeft de gebruiker de mogelijkheid om een bepaald stukje leerstof verder uit te diepen.
Mindmaps hebben op dit moment twee grote beperkingen. Het is onmogelijk om een nieuwe mindmap aan te maken op basis van content die door andere mindmaps gebruikt werd. Bestaande leercontent kan dus niet hergebruikt worden. Mindmaps worden ook weergegeven volgens eenzelfde schema. Er wordt dus geen rekening gehouden met de behoeften/voorkeuren (leerstijl) van de gebruiker. Ook de mogelijkheden van de gebruikte user devices moeten in aanmerking genomen worden. Om zo efficiënt mogelijk informatie te leveren/presenteren moet men op de eigenschappen van het user device inspelen. Mindmaps moeten dus op verschillende manieren kunnen weergegeven worden. Ook dit is een vorm van hergebruik.

1.2 ONDERZOEKSVRAAG

Wanneer men het huidige systeem wil aanpassen aan de nieuwe "vereisten" moet men uitgaan van het bestaande LOMS. Het systeem maakt gebruik van leerobjecten en bevat leerinhouden. In deze thesis verbeter ik de herbruikbaarheid op twee manieren. Enerzijds door vanuit de bestaande leerinhoud meer leerobjecten(LO's) te creëren, telkens voor een andere doelgroep met zijn eigen vereisten en anderzijds door voor de LO's meerdere lay-outs te voorzien.

Een onderzoeksvraag zou kunnen zijn: 'Hoe kan men op basis van bestaande leerinhoud LO's creëren en ervoor zorgen dat dezelfde leerinhoud hergebruikt en aangepast kan worden voor meerdere doelgroepen?

Deze onderzoeksvraag heb ik opgesplitst in een theoretisch en een praktisch gedeelte. Het theoretische gedeelte focust op het verkennen van het kennisdomein. Ik heb bij dit gedeelte volgende deelvragen geformuleerd:

1. Wat is het belang van E-learning en hoe verloopt het?
2. Wat betekenen 'herbruikbaarheid' en 'interoperabiliteit' in het kader van E-learning?
3. Wat zijn leerobjecten en hoe wordt leercontent gestructureerd in leerobjecten?
4. Welke voordelen bieden leerobjecten binnen E-learning systemen en welke voorwaarden moeten vervuld zijn?

Voor het praktische gedeelte heb ik een architectuur van een 'authoring assembly' systeem ontworpen en getest. Voorop staat de mogelijkheid om de content te hergebruiken en aan te passen aan de gebruiker. In het uitwerken van dit deel staat het LOMS centraal. Ik heb dit gedeelte uitgewerkt aan de hand van volgende deelvragen:

1. Hoe moet een LCMS theoretisch gestructureerd zijn om leerinhoud herbruikbaar en aanpasbaar te maken?
2. Hoe werkt het LOMS en voldoet het aan de eisen gesteld in de literatuur en aan de huidige standaarden?
3. Hoe kunnen de OOA/D methodes helpen bij het ontwikkelen van een assembly systeem voor leerobjecten?

1.3 METHODOLOGIE

Ik begin deze thesis met een literatuurstudie, die de deelvragen van het theoretisch gedeelte beantwoordt. Daarna stel ik aan de hand van de opgedane kennis een theoretisch model op dat voldoet aan de eisen van herbruikbaarheid en aanpasbaarheid van leerobjecten. In een volgende stap bestudeer ik het LOMS. Eerst door het systeem zelf onder de loep te nemen en daarna via een interview met de ontwerper ervan. Op basis van de bekomen resultaten ontwerp ik een architectuur aan de hand van de methode beschreven door C. Larman in "UML and Patterns" (2002) die implementeerbaar is in het huidige LOMS. In een laatste fase werk ik deze architectuur uit in een programma. Op basis hiervan evalueer ik of mijn voorgestelde architectuur de gevraagde herbruikbaarheid verbeteringen gerealiseerd heeft.

1.4 INHOUD VAN DE THESIS

Na deze inleiding bekijk ik in het tweede hoofdstuk waarom E-learning belangrijk is en bespreek ik kort de geschiedenis van E-learning. Daarna ga ik in op de begrippen leerplatform, herbruikbaarheid en interoperabiliteit.
In hoofdstuk drie komen de leerobjecten aan bod. De vragen "Wat zijn leerobjecten?" en "Hoe verkrijgen we interoperabiliteit en herbruikbaarheid met leerobjecten?" worden beantwoord. Aansluitend volgt een bespreking van de standaarden ingevoerd om het gebruik van leerobjecten en hun eigenschappen (interoperabiliteit en herbruikbaarheid) te verbeteren. Om dit hoofdstuk af te sluiten bekijk ik welke processen belangrijk zijn bij het creëren van leerobjecten.
In hoofdstuk vier worden de vereisten waaraan de praktische oplossing moet voldoen opgesomd: eerst de vereisten uit de literatuur, daarna de vereisten opgelegd door het bestaande systeem.
Hoofdstuk vijf bevat een uitgewerkte oplossing. Ik leg uit hoe ik tot de voorgestelde softwarearchitectuur kom en bespreek deze kort.
Het laatste hoofdstuk geeft een overzicht van de belangrijkste bevindingen en conclusies die in de loop van dit onderzoek aan het licht kwamen. Daarnaast bevat het aanbevelingen voor verder onderzoek.

2. E-LEARNING

2.1 HET BELANG VAN E-LEARNING

Op 23-24 maart 2000 stelde de Europese Raad zichzelf een doelstelling; namelijk "*de meest concurrerende en dynamische kenniseconomie van de wereld te worden die in staat is tot duurzame economische groei met meer en betere banen en een hechtere sociale samenhang*". (Commissie van de Europese gemeenschappen, 2000)

De nadruk in deze doelstelling ligt op dynamische kenniseconomie. Een economie die competitief wil blijven in deze tijd moet zich concentreren op kennis. Dit betekent zowel het aanmaken van nieuwe kennis als het op een gemakkelijke manier verspreiden van bestaande kennis.

Bedrijven zien het belang van deze doelstelling in. Bolhuis en Simons (1999) stellen vast dat het bedrijfsleven veeleisend is geworden op het gebied van bijscholing voor werknemers.

De meeste werknemers hebben een opleiding genoten bestaande uit een aantal jaar middelbaar en hoger onderwijs. Hier wordt nog altijd volgens het traditionele systeem gewerkt: docenten brengen kennis over op studenten. De studenten verwerken de leerinhouden, en de docenten testen naderhand of de studenten de stof in voldoende mate verwerkt hebben. (Rubens, 2003)

Wanneer werknemers eenmaal van de schoolbanken zijn, moeten zij zich blijvend bijscholen. Het voorgaande traditionele leermodel kan niet meer worden toegepast. Het is te intensief en kost teveel tijd. Deze twee factoren zorgen voor een hoge kost. De industrie verkiest net het tegenovergestelde, namelijk een leerproces dat tijd– en kostenbesparend is. (Bolhuis en Simons, 1999l)

2.2 DE EVOLUTIE VAN E-LEARNING

Vanaf de jaren '60 werden computers gebruikt voor 'Computer Based Instruction' (CBI). CBI definieert men als het maken van een cursus met tekst en geïntegreerde afbeeldingen en dit

met behulp van de computer. Bestanden werden aangemaakt op mainframes en konden door werknemers worden opgevraagd via hun terminal. Voor het maken van deze bestanden had men gespecialiseerde programmeurs nodig. Het verspreiden van kennis op deze manier was door de nood aan hoog opgeleide vaklui zeer duur. (Namahn, 2002)

Tijdens de jaren '70 en '80 kwam de PC op. Bedrijven vervingen hun "domme" terminals door de intelligentere personal computer. Een aantal bedrijven stelde zich tot doel computers voor iedereen toegankelijk te maken. Via innovatieve ontwikkelingen als de GUI en de muis slaagden ze in hun opzet. Iedereen kon nu zelf bestanden aanmaken waarin tekst, geluid en beeld gecombineerd werden.
De industrie bleef niet stilzitten. Ze zocht en vond nieuwe manieren om de computer te laten functioneren als leraar. Een van de belangrijkste ontwikkelingen vond plaats in de luchtvaart. Daar werden specifieke systemen ontwikkeld die de piloten moesten leren hoe te handelen in bepaalde situaties door middel van simulaties. Deze systemen kregen de naam 'computer based traning' (CBT). (Aranda, s.d.)

Tijdens de jaren 90 kwam het gebruik en het leren met behulp van de computer dan echt op gang. Hiervoor waren er twee oorzaken: het World Wide Web en het ontwikkelen van leerplatformen.

De eerste reden waarom het leren via de PC zo een hoge vlucht nam, was het 'Internet' en meer bepaald het 'World Wide Web' (www). Het web is het deel van het Internet dat gebruik maakt van het hypertext transfer protocol (http).
Dit nieuwe protocol bracht twee voordelen met zich mee. Het eerste was onafhankelijkheid. Men kon op elke computer http-pagina's bekijken. Vroeger was men gebonden aan een bepaald platform, bijvoorbeeld Windows. Men kon in de tijd van CBI niet zomaar CBI bestanden uitwisselen. Nu kon men door gebruik te maken van het http CBI pagina's maken die op iedere computer konden bekeken en gelezen worden. Het tweede voordeel was de verspreiding. Vroeger moest men een fysiek medium (diskette, …) hebben dat de bestanden overbracht van de ene computer naar de andere. Door de komst van het Internet kon men zonder een fysieke drager bestanden snel en over grote afstanden uitwisselen.
De tweede reden voor het succes van het leren via de PC was de ontwikkeling van leerplatformen. Softwarebedrijven begonnen systemen te ontwikkelen die hielpen bij het aanmaken, verspreiden en bekijken van leerstof. Een voorbeeld hiervan is het Blackboard platform.

De industrie zag het voordeel van de computer als leermiddel in. Daarnaast verwachtte ze nog nieuwe tijdbesparende mogelijkheden. 'Computer based assessment' (CBA) is hiervan een voorbeeld. Naast de verwachte nieuwe ontwikkelingen besefte men ook dat deze ontwikkelingen

een kader nodig hadden. Men begon namen en definities te verzinnen. Men sprak eerst van teleleren, online leren en web-based leren. Tot men in 1999 de term E-learning gebruikte. Dit in analogie met termen als E-bussines, E-commerce, ...
In zijn nieuwsbrief van 12 oktober 1999 geeft Elliot Masie, E-learning guru, zijn mening over het begrip E-learning. (Rubens, 2003)

> *E-learning Term Catching On: As we looked over recent press releases and the Special Sponsor Showcases at TechLearn '99, the term e-learning seems to be catching on. Developers and vendors are continually struggling with the challenge of what to call the process of combining learning and technology. E-learning, as an extension of e-business and e-commerce, seems to be the latest term to wind up on business plans and presentations. We are even using it at The MASIE Center (…). E-learning is being used to cover a wider set of applications and processes, including web-based training, virtual classrooms, digital collaboration and CBT. This one may stick for a while.*

(Masie, 1999)

Na deze nieuwsbrief spreekt iedereen over E-learning. Het is een echt modewoord dat past in de tijd van de E-commerce, E-business en vele andere woorden beginnend met 'E'. Toch is het opmerkelijk dat tot op heden er nog geen algemeen aanvaarde definitie van het begrip E-learning is opgesteld. (Masie, 1999)

2.3 WAT IS E-LEARNING?

Meerdere auteurs hebben hun eigen definitie van E-learning. In elk van deze definities integreren zij de eigenschappen die volgens hen kenmerkend zijn voor dit leerproces. Hieronder wordt een overzicht gegeven van de meest gangbare definities en wordt aangegeven wat volgens de respectieve auteurs van belang is.

> *"E-Learning" is de verzamelnaam voor het vormgeven van leersituaties (formeel en informeel) met behulp van informatie- en communicatietechnologie (in het bijzonder internettechnologie). E-Learning wordt gebruikt binnen bedrijfsopleidingen en binnen het onderwijs."*

(E-learning.nl, 2007)

> *Een andere definitie van E-Learning (in brede zin) is: elke leervorm die gebruik maakt van een computer netwerk voor distributie, communicatie over en weer en facilitering.*

(Wikipedia, 2006)

> *A simple working definition of the term e-Learning is "learning or training that is prepared, delivered, or managed using a variety of learning technologies and which be deployed either locally or globally."*

(Hodgins et al, 2003)

> *E-Learning is het gebruik van internettechnologie om educatieve content te creëren, te managen, beschikbaar te stellen, te beveiligen, te selecteren en te gebruiken, om gegevens van de lerenden op te slaan en de lerenden te volgen, en om communicatie en samenwerking mogelijk te maken. Het doel is de overdracht en opbouw van kennis en vaardigheden te ondersteunen, uit te breiden en te flexibiliseren.*

(Alberink et al, 2001)

De meeste auteurs definiëren E-learning als het betrekken van ICT in het leerproces. Bij Alberink et al (2001) van het Telematicainstituut gaan ze een stapje verder. Ze verwijzen expliciet naar het gebruik van internettechnologie. Daarnaast splitsen ze het E-learning proces op in verschillende fases. In dit eindwerk wordt de definitie van het Telematicainstituut gehanteerd.

2.4 LEERPLATFORM

2.4.1 Wat is een leerplatform

E-learning is een theoretisch begrip. Om E-learning in de praktijk te brengen zijn er ondersteunende systemen nodig. Deze systemen omvatten hardware, software (programma's en pakketten) en menselijke input. In deze thesis staat de software centraal.

Alberink et al (2001) definiëren een E-learning platform als een platform dat drie taken ondersteunt:

1. Het ontwikkelen van leerstof/toetsen door ontwikkelaar/trainer/docent en presentatie van cursus/leerstof/toetsen door trainer/docent.
2. Het beheer van de communicatie tussen trainer/docent en student/cursist en externen.

3. De organisatie en beheer van de content door trainer/docent; de administratieve organisatie wordt gedaan door een systeembeheer.

Som Naidu (vertaald in OSS in het Onderwijs, 2006) definieert een E-learning platform als een reeks software-instrumenten waarmee allerlei leer- en lesactiviteiten kunnen worden beheerd en vergemakkelijkt. Daarnaast zijn ze zowel kosten- als tijdbesparend, wanneer ze op grote schaal worden ingezet.

In deze thesis wordt een E-Learning platform gedefinieerd naar de definitie van Naidu. Een E-Learning platform wordt dus beschouwd als een software tool waarmee leer- en lesactiviteiten kunnen worden beheerd. In het platform worden drie onderdelen onderscheiden.

Het eerste deel is het LCMS (E-learning content management system). Dit systeem beheert de leerstof. Het geeft de gebruiker de mogelijkheid om de leerstof aan te maken, te gebruiken of te hergebruiken en het slaat deze leerstof op.
Het tweede deel van het E-Learning platform, het LMS (learning management system) spitst zich meer toe op het beheersen van de leerstof en op de interactie tussen het systeem en de gebruiker. Enkele functies van het LMS zijn het beheren van online klasseninteractie, het volgen en het rapporteren van de vooruitgang van leerlingen, het beoordelen van leerresultaten en het rapporteren over prestaties van taken. (OSS in het Onderwijs 2006)
Het derde en laatste onderdeel van een E-learning platform is de koppeling met bestaande programma's. Zo zal een E-learning platform vaak werken met een Word document of een video- of audiofragment. Voor het weergeven van deze bestanden zal het platform beroep doen op andere applicaties zoals Windows Mediaplayer, Internet Explorer of een programma uit de Office Suite.

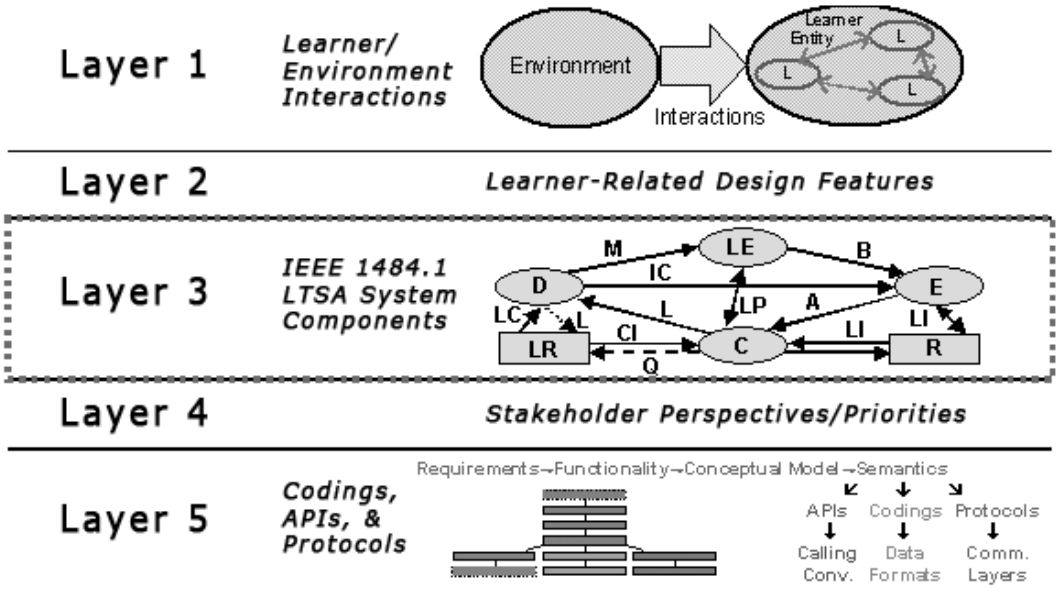
Voor de volledigheid zou ik ook nog willen aanhalen dat Martin Alberink et al (2001) een verschil maken tussen E-Learning platformen en auteursomgevingen. Volgens Alberink kan men met auteursomgevingen ook content aanmaken. Maar deze content heeft niet altijd een educatief doel. Fabrikanten van E-learning platformen maken geen groot onderscheid tussen een auteursomgeving en E-learning platformen.

2.4.2 Voorbeeldarchitectuur van een leerplatform

Om een idee te krijgen van hoe een E-learning platform eruit ziet bespreek ik kort het LTSA model.

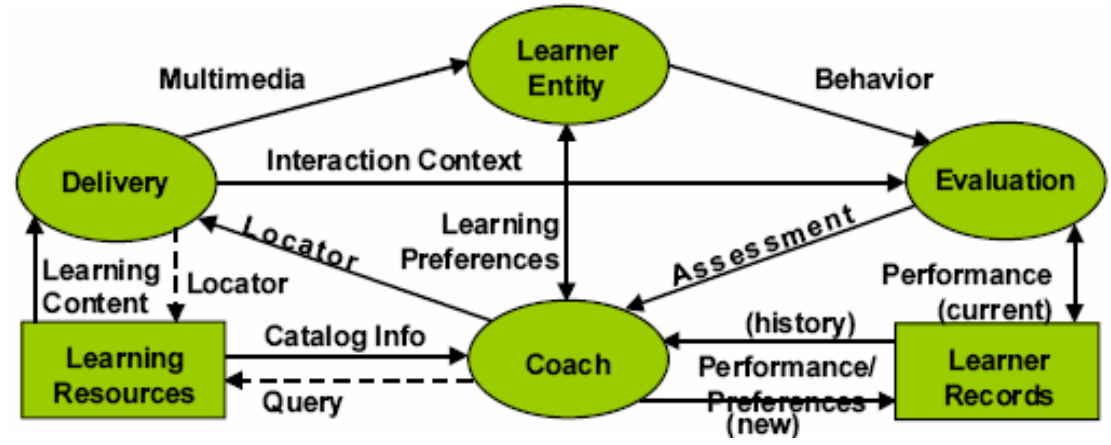
Een E-Learning platform moet verschillende taken uitvoeren. De LTSC (Learning technology systems committee) - een comité van de IEEE (Institute of Electrical and Electronics Engineers),

dat zich bezighoudt met het ontwerpen van architecturen voor E-learning - heeft een voorbeeldarchitectuur ontworpen voor het modelleren van een E-learning platform.
Figuur 1 toont het LTSA (learning technology systems architecture) model. De LTSC stelt hiermee een didactisch, inhoudelijk en technologisch neutraal schema voor. Op deze manier kan iedere maker van een leerplatform zijn instructies, systemen en technologieën implementeren.
Het model bestaat uit vijf lagen. De lagen vertonen een afnemend niveau van abstractie. (Anido-Rifón et al, 2002)



Figuur 1. LTSA architectuur (Anido-Rifón et al, 2002)

Voor ons is vooral de derde laag belangrijk. Layer 3 van deze figuur is daarom uitvergroot in figuur 2.



Figuur 2. Layer 3 (Farance en Schoening, 1998)

Het schema van layer 3 noemt men een procesmodel. Het is opgebouwd uit vijf entiteiten. De ovale entiteiten zijn procescomponenten, de rechthoekige zijn opslagcomponenten. De pijlen stellen relaties tussen de componenten voor. (Riemersma et al, 2002)

De werking van het schema gaat als volgt. Een leerling interageert met het programma. De 'evaluation component' verwerkt dit gedrag en stuurt de bekomen resultaten naar de 'coach'. Een voorbeeld van een resultaat kan een behaalde score op een ondervraging zijn.

De coach en de evaluation component slaan de gevonden resultaten op in de 'learner records component'. Op basis van de opgeslagen resultaten geeft de coach instructies aan de 'delivery component'. De delivery component raadpleegt de 'learning resources' om de volgende - op leergedrag gebaseerde - leereenheden op te halen en in een aangepaste lay-out voor te stellen aan de leerling.

Dit schema geeft een eerste inzicht in wat een E-learning platform allemaal doet en in welke componenten men het kan opdelen. Ook laat het op een interessante manier zien hoe feedback verloopt binnen een E-learning platform. Voor deze thesis is de relatie tussen de learning resources en de delivery component belangrijk. Het schema geeft een idee waar dit proces plaatsvindt binnen een E-Learning platform.

Twee belangrijke begrippen bij het opslaan en gebruiken van data in een E-Learning platform zijn herbruikbaarheid en interoperabiliteit. We bedoelen voornamelijk herbruikbaarheid van de data en de interoperabiliteit van de data tussen systemen.

2.4.3 Interoperabiliteit en herbruikbaarheid in leerplatformen

2.4.3.1 Reusability

Reusability is een term afkomstig uit het jargon van softwareontwikkelaars. De term verwijst naar een stuk code dat opnieuw kan gebruikt worden, al dan niet met een kleine wijziging om het stukje code een nieuwe functionaliteit te geven. (Wikipedia, 2007b)

Reusability heeft meerdere voordelen. Het eerste is efficiëntie. Als men delen kan hergebruiken, kan men veel sneller programmeren. Men moet niet altijd van nul beginnen, maar kan al steunen op bestaande delen.
Een ander pluspunt heeft betrekking op de moeilijkheid en de juistheid bij een aanpassing. Als iets gemaakt is volgens het principe van herbruikbaarheid, moet men bij een eventuele

aanpassing maar één onderdeel aanpassen, omdat alle toepassingen die gebruik maken van veranderde en hergebruikte stukjes code automatisch worden aangepast.
Vooral in de softwareontwikkeling zag men de voordelen van herbruikbaarheid in. Om het principe van herbruikbaarheid zo goed mogelijk te integreren heeft men het 'object georiënteerd' denken ontwikkeld. Hierbij wordt een proces opgesplitst in kleinere delen, die men objecten noemt. Elk object heeft zijn eigen functies en variabelen. In de uiteindelijke toepassing worden dan verschillende objecten gecombineerd. Elk object bezit de voordelen van reusability.

> *Reusability: The flexibility to incorporate instructional components in multiple applications and contexts.*

(ADL, 2004)

Het begrip herbruikbaarheid is dus breder dan alleen maar computer code. In de voorgaande definitie spreekt men over een 'instrucional component' wat we kunnen vertalen naar een stuk leerstof. Dit stuk leerstof moet herbruikbaar zijn in meerdere applicaties en context. Het is deze definitie van herbruikbaarheid die we aan houden in dit eindwerk.

2.4.3.2 Interoperability

Interoperabiliteit definieert Belgif (2005), het Belgische interoperabiliteitskader, als volgt:

> *Interoperabiliteit laat de uitwisseling toe van gegevens en diensten tussen menselijke en technische, en gedeelde systemen. Binnen een ruimer perspectief staat interoperabiliteit voor de uitwisseling van informatie tussen administraties, tussen een administratie en de burger of een onderneming, zonder van hen specifieke inspanningen te verlangen. Afgezien van de ruimtelijke dimensie van interoperabiliteit (de uitwisseling tussen twee systemen), mag men ook de tijdsdimensie niet uit het oog verliezen. Interoperabiliteit beoogt eveneens de duurzaamheid van de gegevens en hun toegankelijkheid in de toekomst.*

(Belgif, 2005)

Bovenstaande definitie is vrij algemeen. De definitie van het IEEE is meer gericht op de informatica.

> *The IEEE defines interoperability as: the ability of two or more systems or components to exchange information and to use the information that has been exchanged.*

(Wikepedia, 2007)

In deze thesis wordt interoperabiliteit gedefinieerd op het niveau van data en computersystemen. Interoperabiliteit betekent hier dat een bepaalde set data en/of instructielijnen kan gelezen en begrepen worden door verschillende systemen op verschillende computers. Een voorbeeld hiervan is de interoperabiliteit tussen een Microsoft Office document en een Open Office document. Zo kan elk document van Microsoft Office gelezen en aangepast worden door Open Office en vice versa.

3. LEEROBJECTEN EN LEERCONTENT

In het vorige hoofdstuk werd vastgesteld dat bij het opslaan en gebruiken van data in een leerplatform herbruikbaarheid en interoperabiliteit van essentieel belang zijn. Om deze principes te kunnen toepassen heeft men leerobjecten bedacht.
Dit hoofdstuk begint met een bespreking van het begrip leerobject. In een volgend onderdeel worden de relaties tussen interoperability en reusability aan de ene kant en leerobjecten aan de andere kant uitgelegd. Aansluitend wordt bekeken hoe deze eigenschappen van leerobjecten gerealiseerd kunnen worden door gebruik te maken van standaarden. Afsluitend bespreek ik hoe men een leerplatform kan gebruiken voor het creëren van leerobjecten. De twee belangrijkste onderdelen van dit proces zijn compositie en decompositie.

3.1 LEEROBJECTEN

3.1.1 Wat zijn leerobjecten?

Om de principes van herbruikbaarheid en interoperabiliteit in leersystemen toe te passen maakt men gebruik van 'learning objects'. De term werd voor het eerst gebruikt door Wayne Hodging (2000 in Liber, 2005) in zijn werk 'Learning Architectures, APIs and Learning Objects'.

Leerobjecten worden vaak afgekort met LO. Vroeger noemde men ze reusable learning objects/herbruikbare leerobjecten (RLO), maar omdat leerobjecten per definitie herbruikbaar zijn heeft men de R laten wegvallen. (Nijnveld en van de Ven, 2003)

Voor het maken van een leerobject deelt men een stuk leerstof op in kleinere delen, die apart worden opgeslagen. De opgeslagen delen kunnen later opnieuw gecombineerd worden tot grotere gehelen. Deze grotere delen kunnen dan door leerlingen gebruikt worden om te leren. (Strijker, 2006)
Voorbeelden van leerobjecten kunnen zijn: een afbeelding, een informatieve tekst, een testvraag, enz.

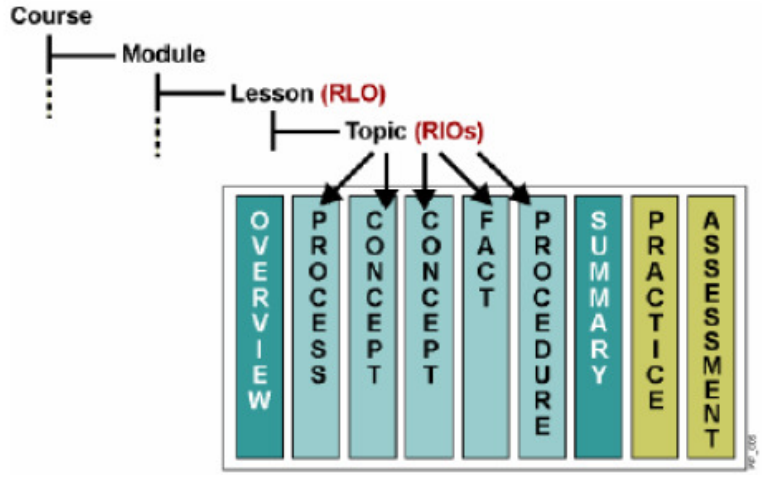
De definitie voor leerobjecten ontwikkeld door het Institute of Electrical and Electronics Engineers (2001, in Nijveld en van de Ven, 2003) wordt wereldwijd als de standaarddefinitie beschouwd. Ze luidt als volgt:

> *A learning object is an entity, digital or non-digital, that can be used, re-used or referenced during technology-supported learning. Examples of learning objects include multimedia content, instructional content, instructional software and software tools [and] in a wider sense ... learning objectives, persons, organizations, or events.*

(IEEE, 2001 in Nijveld en van de Ven, 2003)

Deze definitie is heel algemeen. Een aantal auteurs verklaart dat een leerobject altijd digitaal is. Anderen wijzen op het belang van granulariteit bij de definitie van een leerobject. Onder granulariteit verstaan we de inhoudelijke grootte van het object, bijvoorbeeld: is het object een foto, een heel hoofdstuk, een vraag, enz.

De meeste auteurs maken ook een hiërarchie gebaseerd op granulariteit. Een voorbeeld hiervan is de indeling die men bij Cisco gebruikt (figuur 3).



Figuur 3. De hiërarchische opdeling van Cisco (Nijveld en van de Ven, 2003)

Op het laagste niveau bevinden zich de RIO's. RIO staat voor 'reusable information object'. Dit is het kleinste stukje informatie beschikbaar. Op zich is het enkel informatie. Door het bundelen van RIO's maakt men een 'lesson' en die wordt als een RLO (reuseable learning object) beschouwd. Dit leerobject bevat informatie en relaties tussen de verschillende stukken informatie. Deze stukjes informatie kunnen dan weer verder gebundeld worden tot een 'module'. Een 'module' kan op haar beurt dan weer tot een 'course' gecombineerd worden. Men kan dit vergelijken met een hoofdstuk (= module) en een boek (= course). (Strijker, 2006)

De hiërarchie beschreven in de vorige paragraaf is maar één van de vele mogelijkheden. Elke auteur of ontwerper kan zijn eigen hiërarchie maken. Maar als men aan een bepaalde standaard wil voldoen, moet men met de hiërarchie van die standaard rekening houden.

3.1.2 Herbruikbaarheid van leerobjecten

We weten nu wat een leerobject is, maar hoe helpt het om interoperabiliteit van content tussen systemen en herbruikbaarheid van leercontent te bekomen?

Volgens Friesen (2001) zijn discoverability, modularity en interoperability de belangrijkste eigenschappen van learning objects. Eenmaal deze eigenschappen vervuld zijn, zou volgens de auteur automatisch voldaan worden aan de eis van reusability van de content. Dit gedeelte bespreekt kort deze eigenschappen en bekijkt hoe men de verschillende eigenschappen aan een leerobject kan geven.

Met 'discoverability' bedoelt men dat het systeem het object moet kunnen vinden en begrijpen. Het object moet dus informatie aan het systeem kunnen geven over zichzelf. Dit kan doordat het object metadata bevat. Metadata zijn data over data. Deze metadata moeten op hun beurt door het systeem begrepen worden.

Friesen onderscheidt drie grote categorieën in de metadata. De eerste categorie noemt hij 'general'. Deze categorie bevat algemene informatie over het object, zoals titel, auteur, datum, enz.
Een tweede categorie krijgt het label 'technical'. Deze categorie bevat vooral technische informatie over het object. Eigenschappen als file type, operating system requirements horen hier thuis.
De derde en laatste categorie noemt de auteur 'semantic'. Semantic beschrijft waarover het object gaat. In deze categorie vinden we eigenschappen als de vakken waartoe het object kan behoren, de topics die het behandelt, de moeilijkheidsgraad, enz.

De volgende eigenschap is 'modulariteit'. Met modulariteit bedoelt men, dat het object op zichzelf moet staan, niet sequentieel mag zijn en dat het een eenheid moet vormen. Dit is te vergelijken met de black box benadering uit het object georiënteerd denken. Dit houdt in dat een persoon buiten de maker, het object moet kunnen aanpassen voor eigen gebruik, zonder dat hij kennis moet hebben over de werking van het object.
Friesen (2001) merkt terecht op dat dit een zeer duale eigenschap is. De meeste objecten worden gecreëerd in verschillende omgevingen. Het is dan ook zeer moeilijk om deze objecten achteraf in andere omgevingen (verschillend van hun 'creatieomgeving') te integreren. Als

oplossing hiervoor haalt Friesen het gray box principe aan. Dit idee van Büchi en Weck (1997 in Friesen, 2001) pleit ervoor dat de programmeur een deel van het gebruik en de implementatie overlaat aan de gebruiker.

De derde eigenschap is 'interoperability'. Interoperabiliteit wordt hier gedefinieerd als de mogelijkheid van een object om te werken op verschillende systemen. Dit kan volgens Singh (2000 in Friesen, 2001) alleen als alle betrokkenen met open standaarden werken. Hij wijst er ook op dat bedrijven met commerciële doelstellingen dit kunnen tegenwerken. Als voorbeeld geeft hij Microsoft. Dit bedrijf heeft door het succes van IE 5 ervoor gezorgd dat de nieuwe open protocollen geen kans hadden, omdat het IE 5 alleen de conventionele protocollen ondersteunde.
Open protocollen zijn niet de enige afspraken die men moet maken om interoperabiliteit te bereiken. Men moet ook technische standaarden afspreken. In zijn artikel vergelijkt Friesen (2001) interoperabiliteit met een stekker en een stopcontact. De metadata moeten beschrijven hoe de stekker in het stopcontact past. De standaarden moeten ervoor zorgen dat de stekker in het stopcontact past.

De eigenschap reusability wordt bereikt als aan de voorgaande eigenschappen is voldaan. Als een object opzoekbaar is, geïntegreerd kan worden in een groter geheel en werkt op het systeem wordt het automatisch herbruikbaar.

3.2 DE STANDAARDEN EN HOE ZE HELPEN BIJ HET BEREIKEN VAN INTEROPERABILITEIT EN HERBRUIKBAARHEID?

De vraag die nu nog open blijft, is: "Hoe bereiken we deze eigenschappen in de realiteit?" Een aantal organisaties verricht op dit gebied goed werk. Aan de ene kant proberen ze te zorgen voor zowel standaarden voor metadata als structuren om de herbruikbaarheid te bevorderen. Aan de andere kant leveren ze voorbeeldarchitecturen, richtlijnen en best practises om de interoperabiliteit van het systeem te verbeteren. (Kim en Shih, 2004)
De belangrijkste organisaties zijn Dublin Core Metadata Initiative, AICC, IMS en ADL.

3.2.1 Organisaties en de standaarden waarvoor ze staan

3.2.1.1 Dublin Core Metadata Initiative

Dublin Core Metadata Initiative is de oudste organisatie uit het rijtje. Haar grootste bijdrage is de Dublin Core, een metadata standaard voor webdocumenten. De standaard bestaat uit 15 categorieën. Elke categorie kan tot 10 attributen bevatten. Sinds haar ontstaan heeft de organisatie haar standaard nog niet aangepast. Wel heeft ze de implementering van haar standaard verbeterd. Zo kan men op haar website informatie vinden over hoe de standaard te implementeren in html, xhtml en xml. (Dublin Core, 2007)

3.2.1.2 AICC

AICC staat voor Aviation Industry CBT Committee. Het comité, dat gesticht werd in 1988, ontwikkelt richtlijnen voor de luchtvaartindustrie bij het ontwerpen van CBT (computer based training) systemen voor piloten. Door deze richtlijnen wil het comité bijdragen tot systemen die efficiënt, effectief en interoperatibel zijn. Daarnaast fungeert het ook als een open forum voor discussie over richtlijnen. De richtlijnen van het AICC gaan voornamelijk over welke componenten een systeem moet hebben en hoe ze met elkaar moeten samenwerken. Dit komt later in het stuk over Scorm aan bod (cf. 3.3.1.4 ADL - Scorm). (AICC, 2007)

3.2.1.3 IMS

In 1997 wordt door het National Learning Infrastructure Initiative van het Amerikaanse EDUCAUSE de organisatie IMS (Instructional Management Systems Initiative) opgericht, met als doel het verbeteren van hun leerplatform. Het IMS houdt zich zowel bezig met metadata als met standaarden. (IMS, 2007)

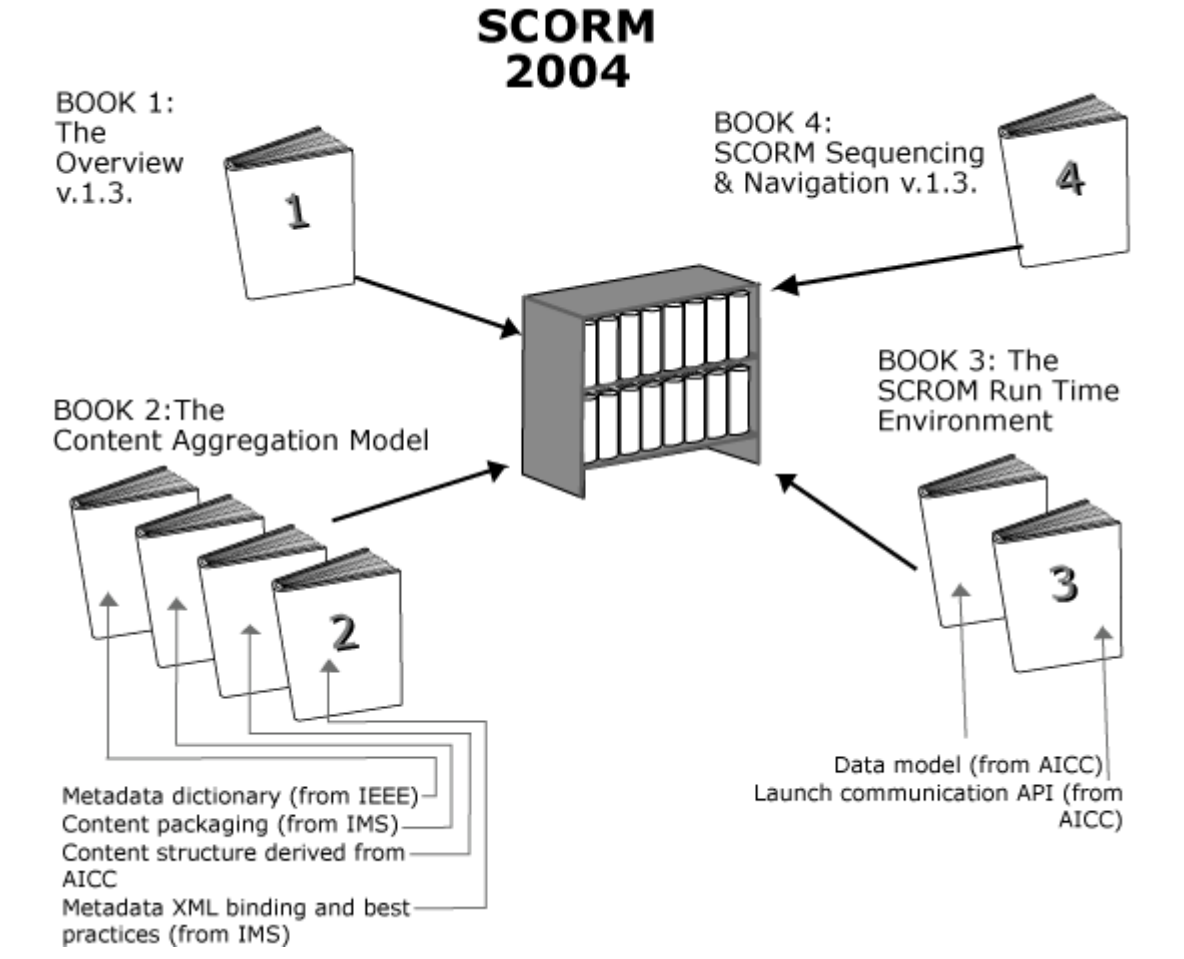
De belangrijkste ontwikkeling is de IEEE LOM, een metastandaard die het IMS samen met de Europese organisatie Ariadne ontwikkeld heeft. Deze metastandaard is gebaseerd op de Dublin Core. Ze telt 86 categorieën. Elke categorie bestaat uit zeven elementen. De producenten die deze standaard gebruikten, vonden dit teveel. Ze vonden dat ze teveel informatie moesten invullen en dat deze niet altijd even nuttig was. Als reactie daarop heeft het IMS een tweede standaard uitgebracht. Deze standaard telt maar 19 categorieën en gaat door het leven onder de naam Best Practise Core Standard. (Friesen, 2001)

Naast de metastandaard heeft het IMS ook standaarden ontwikkeld voor andere delen van hun E-learning platform. Zo is er de ‘Enterprise Interworking Standard’, speciaal ontwikkeld voor het

uitwisselen van informatie over studenten. Een andere bekende standaard is 'Content Packaging', waar men vooral aandacht heeft voor het opslaan van learning content. (IMS, 2000)

3.2.1.4 ADL - Scorm

In 1997 moest het Amerikaanse ministerie van defensie een groot aantal mensen opleiden. Om aan deze opleidingsbehoefte te voldoen, richtte men het ADL (Advanced Distributing Learning) op, een organisatie die een standaard ontwikkelde die ze Scorm (Shareable Content Object Reference Model) noemde. Deze standaard is een overkoepelende standaard, die vaak vergeleken wordt met een boekenkast (figuur 4). In elk "boek" bespreekt Scorm een ander deel van een E-learning platform en wordt er uitgelegd hoe een bestaande standaard kan helpen bij het ontwikkelen van een platform. De categorieën zijn CAM (content aggregation model) RTE (run time environment) en SN (sequencing navigation). (Leteney, 2006)



Figuur 4. Overzicht van de Scorm standaard (Ellis, 2005)

Het eerste boek geeft een algemeen overzicht van de Scorm standaard. Het kan dan ook gebruikt worden als een inleiding tot de wereld van Scorm en learning objects. Het tweede boek bespreekt hoe men een leerobject kan exporteren uit een systeem en hoe men het kan importeren in een ander systeem. Het derde boek bespreekt hoe een SCO (shareable content object) wordt ingelezen en hoe het LMS ermee omgaat. Het vierde en laatste boek geeft meer informatie over hoe men verschillende leerobjecten op elkaar kan laten volgen en hoe men hier eventueel regels kan implementeren. Een regel kan zijn, dat men een bepaalde score op een test moet behalen voor men aan het volgende onderdeel kan beginnen.

De rode draad door al deze boeken is de hiërarchie die Scorm hanteert. De Scorm standaard structureert leerinhouden in kleine objecten. Het laagste niveau bestaat uit 'assets'. Het zijn de bouwstenen van leerobjecten. Het zijn kleine stukjes media die geen speciale betekenis hebben, bijvoorbeeld een stuk tekst, een foto of een geluidsfragment. Een aantal assets samengevoegd vormt een 'shareable content object'. SCO's stellen kleine eenheden van leerstof voor. Een aantal SCO's samen vormt een 'course'. (ADL, 2004)

Alle onderdelen van een course bevatten metadata gebaseerd op de IEEE LOM (IMS) standaard. De courses hebben ook een CSF (content structured format). Dit is een XML bestand waarin staat waaruit het pakketje bestaat en hoe het georganiseerd is. CSF is gebaseerd op de IMS packaging standard. Scorm is op dit moment de leidinggevende standaard voor E-learning platformen. (Alberink et al, 2001)

3.2.2 Kritische beschouwing bij de standaarden

Ondanks al deze initiatieven zijn er nog heel wat problemen op het gebied van E-learning platformen. Scorm heeft zijn best gedaan om een standaard aan de industrie aan te bieden. Toch vindt de industrie dat deze standaard niet volledig voldoet. Men vindt hem te uitgebreid en te complex. Daarnaast worden bestaande standaarden regelmatig vernieuwd. Deze nieuwe standaarden zijn niet altijd compatibel met oudere standaarden.

Een ander probleem is dat Scorm te algemeen is. Afhankelijk van de gebruiker zou men graag andere dingen geïmplementeerd zien. Zo argumenteren Lu en Chen (2006) dat Scorm geen mogelijkheid biedt tot 'access control' of tot een 'delegation system'. Dit komt er eigenlijk op neer dat een auteur niet kan beslissen wie wat mag zien. Zo kan een leraar dus niet uitsluitend *zijn* cursus presenteren voor *zijn* klas. Alles wat hij publiceert zal voor elke gebruiker van het systeem zichtbaar zijn.

Een andere bemerking vinden we bij Hirumi (2005). Deze auteur haalt aan dat de industrie en het onderwijs verschillende standaarden voor kwaliteit van leerobjecten hebben. Hij wijst er dan

ook op dat Scorm op geen enkele manier een oplossing voorziet voor dit probleem. De auteur pleit ervoor dat Scorm een kwaliteitscomponent opneemt in zijn standaard.

Een eerder indirect gevolg wordt aangehaald door Naish (2007). Hij is voorstander van het promoten van 'open source' standaarden en software. Maar hij waarschuwt ervoor dat dit kan leiden tot 'module creep'. Scorm is een open standaard die veel mensen zal aanzetten tot het ontwikkelen van E-learning platformen. Veel van deze E-learning platformen zullen dankzij de open source community, mensen die gratis software ontwikkelen, altijd maar uitgebreider worden. Dit proces zal blijven voortgaan tot er zoveel modules in het platform zitten dat het niet meer duidelijk is welke module wat doet. Daarnaast zal het platform ook heel wat overbodige modules krijgen, omdat er geen centrale autoriteit is, die deze ongewilde ontwikkeling in de hand houdt. Zo een autoriteit zou bijvoorbeeld ADL kunnen zijn.
Op dit moment zijn er twee actieve open source initiatieven, namelijk Moodle en OpenELMS. Het eerste initiatief heeft al last van module creep. Het tweede wordt dankzij Open Elms nu nog in de hand gehouden. Zij ontvangen van iedereen nieuwe toevoegingen. Hieruit selecteren ze de besten en bieden deze aan de gebruikers aan.

3.3 CREATIE VAN LEEROBJECTEN

Een vraag die nog beantwoord moet worden, is: "Hoe gebruikt men een leerplatform dat gebaseerd is op leerobjecten?" In de literatuur wordt het gebruik van zo een platform opgedeeld in twee delen. Het eerste deel noemt decompositie, het tweede compositie.

Decompositie beschrijft hoe leerinhoud in het leerplatform opgeslagen wordt. Dit proces wordt meestal vergeleken met een boek. Zoals een boek uit verschillende delen bestaat, nl. hoofdstukken en paragrafen, zo bestaat leerinhoud ook uit verschillende delen. Leerinhoud wordt dus volgens een bepaalde hiërarchie ingedeeld (cf. 3.2.1 Wat zijn leerobjecten). Dit opdelen wordt decompositie genoemd. Tijdens de decompositie worden de uiteindelijke stukjes leerstof zo klein mogelijk gemaakt. Hoe kleiner de deeltjes, hoe beter herbruikbaar ze zijn. De kleinst mogelijke deeltjes worden op het einde van de decompositie opgeslagen in de database (figuur 5). (Schreurs en Moreau, 2006b)

Figuur 5. Het decompositieproces (Schreurs en Moreau, 2006b)

Weitl et al (2004) delen het decompositieproces nog verder op. Decompositie is voor de auteurs ook het proces van het opdelen van leerinhoud in kleine delen, maar daarna wordt een tweede proces op de opgedeelde leerinhoud toegepast, namelijk 'de-contextualisatie'. Ze definiëren dit proces als het onafhankelijk maken van de opgedeelde delen.

Compositie is het tegenovergestelde van decompositie. Tijdens deze fase worden de stukjes leerinhoud weer samengevoegd. Dit samenvoegingproces is iets ingewikkelder dan het op het eerste gezicht lijkt. De leerinhoud wordt niet zonder meer getoond aan de leerling. Het weergeven van de leerinhoud wordt aangepast aan de leerstijl van de leerling. Een leerling leert op drie manieren: visueel (het lezen van teksten en bekijken van grafieken), auditief (het luisteren naar een leerkracht of een geluidsfragment) en kinesthetisch (het aanduiden in teksten, notities nemen tijdens een les). Voor elke leerling is een van deze drie manieren dominant. Een ontwerper van leerobjecten moet proberen de voorstelling van de leerinhoud aan te passen aan de dominante leerstijl van de leerling.
Dit kan hij doen door gebruik te maken van scenario's. Scenario's zijn een onderdeel van het leerplatform die dezelfde leerobjecten voorstellen op verschillende manieren, aangepast aan de dominante leerstijl van de leerling. (Schreurs en Moreau, 2006c)

4. LEERSYSTEEM VOOR DE ONTWIKKELING VAN E-LEARNING MODULES

In het tweede deel van deze thesis wordt de informatie uit de literatuurstudie toegepast in een praktische gevalstudie. De opdracht is een deel van het bestaande LOMS te herontwerpen en zo de herbruikbaarheid van het systeem te verbeteren. Voornamelijk op het vlak van hergebruik van bestaande leerinhoud en het aanpassen van leerobjecten aan de gebruiker.

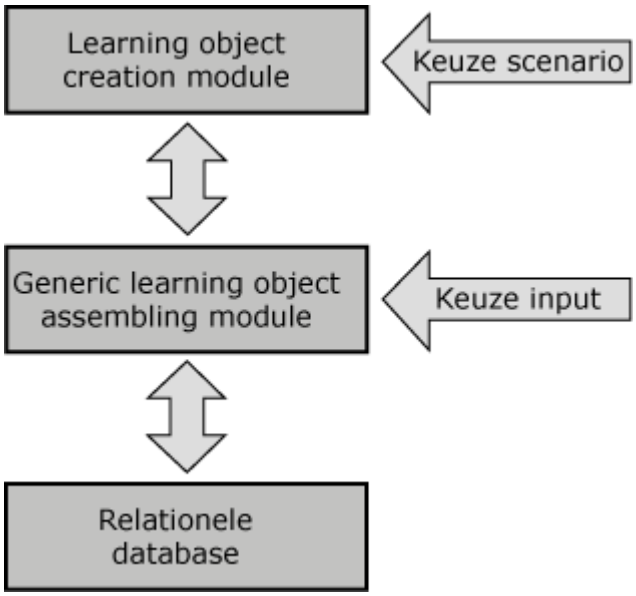
De praktische gevalstudie bestaat uit meerdere delen. Eerst worden de theoretische voorwaarden besproken. Daarna worden de voorwaarden opgelegd door de praktijk onder de loep genomen. Deze voorwaarden zijn een gevolg van tekorten van het bestaande systeem en de manier waarop het gebruikt wordt.

4.1 THEORETISCHE VEREISTEN

De theorie leert ons dat door het gebruik van leerobjecten interoperabiliteit en herbruikbaarheid kunnen verkregen worden. Verder onderzoek (cf. 3.2.2 Leerobjecten, interoperabiliteit en herbruikbaarheid) heeft aangetoond dat discoverabiliteit, interoperabiliteit en modulariteit een belangrijke rol spelen voor het bekomen van herbruikbaarheid in een systeem.

Voor het praktische deel werd, na kennis te hebben verworven tijdens de literatuurstudie, gekozen voor een 3-lagige architectuur. De eerste laag is de bestaande relationele database die aangemaakt wordt in het content management deel van het LOMS. We gaan er hier van uit dat het systeem voor de invoer van content bestaat en los gezien wordt van het authoring assembly systeem dat wij gaan bouwen. De twee daaropvolgende lagen vormen het te ontwerpen systeem. Waarvan de tweede laag de content gaat assembleren tot een leercontentmodule (LO). We noemen dit systeem het ‘generic learning object assembling module’. ‘Generic’ omdat het zich aanpast aan welke content de gebruiker wil opnemen in de nieuwe leermodule. ‘assembling’ beschrijft eerder de taak van het te ontwikkelen systeem. Authoring wordt door Harris (2002) beschreven als het proces waarbij nieuwe leerinhoud (digitale documenten en media) wordt aangemaakt en waarbij deze nieuwe of bestaande leerinhoud wordt samengevoegd (assembled) tot een nieuw leerobject.

Het te ontwikkelen systeem zal alleen de samenvoegingstaak uitvoeren van het authoring proces. Daarom krijgt het de naam 'assembling'.
De derde laag noemen we de 'learning object creation module'. De taak van deze module is het presenteren van het leerobject. Zij zorgt er tevens voor dat bij de presentatie van het leerobject men kan kiezen in welke lay-out het leerobject wordt getoond. Voor haar werking doet ze beroep op de tweede laag. Tijdens het aanmaken van nieuwe leercontent modelleert het 'generic learning object assembling module' de content in een leerobject. Dit modelleerproces kan hergebruikt worden door de derde laag voor het weergeven van de content. Figuur 6 stelt deze drie lagen en de onderlinge interacties schematisch voor.



Figuur 6. Voorstelling 3-lagige model

Deze theoretische architectuur voldoet op twee manieren aan onze voorwaarde, namelijk het verbeteren van de herbruikbaarheid van het systeem.
De eerste manier was de bestaande leerinhoud herbruikbaar te maken. Dit wordt gerealiseerd in de tweede laag. het learning object assembling programma gaat de eerste laag ondervragen en de gebruiker de mogelijkheid geven de content die in deze laag zit samen te voegen tot een nieuw leerobject. Dit leerobject zal de tweede laag dan opslaan in de eerste laag.
De tweede manier was het leerobject aanpassen aan de situatie en eisen van de gebruiker. Hierbij denken we vooral aan zijn persoonlijke leerstijl en aan het systeem waarmee hij het leerobject bekijkt. De derde laag, de 'learning object creation module', heeft als doel een leerobject opgeslagen in de eerste laag weer te geven. Hiervoor kan de gebruiker kiezen uit verschillende lay-outs. De derde laag gaat het leerobject in de lay-out integreren op het moment dat de gebruiker het vraagt. Op deze manier wordt de content hergebruikt in verschillende lay-outs.

Naast deze twee verbeteringen heeft deze 3-lagige architectuur nog een voordeel op het vlak van herbruikbaarheid. De samenwerking tussen laag één en laag twee maakt het eenvoudiger om de content te hergebruiken, want ze laadt de gekozen content in het geheugen. Daarnaast maakt ze het weergeven van de content in de browser gemakkelijker omdat ze het proces van ophalen, relaties leggen en weergeven regelt met het geven van één commando. Men zal de samenwerking van laag één en twee als een basis kunnen gebruiken voor toekomstige programma's.

Doordat we de hiërarchie en de database van het huidige systeem overnemen, behouden we de graad van discoverabiliteit, interoperabiliteit en modulariteit. (cf. 4.2.3 Evaluatie van het LOMS aan de hand van de literatuur)

De graad van discoverabiliteit wordt bepaald door de metadata die het systeem over de leerobjecten bevat. Op dit moment bevat de eerste laag, de relationele database, metadata over de leercontent en de leerobjecten.

Aangezien we de huidige leerobjecten behouden, behouden we ook de interoperabiliteit van de leerobjecten. Die is namelijk afhankelijk van hoe goed de stukjes content (een videofilm, een word document, …) kunnen samenwerken met de verschillende systemen (browers, operating systemen, …).

Vermits de bestaande granulariteit en de bijbehorende hiërarchie behouden blijven, behouden we ook de graad van modulariteit die deze hiërarchie bezit. De tweede laag bouwt deze hiërarchie op volgens het schema aangereikt door de eerste laag.

4.2 DE PRAKTISCHE VEREISTEN OPGELEGD DOOR HET LOMS

4.2.1 Het analyseren van LOMS

In dit onderdeel worden de vereisten van het huidige systeem beschreven. Eerst wordt het bestaande leerplatform van de UHasselt, het LOMS geanalyseerd. Hierna wordt het systeem getoetst aan de voorwaarden gesteld in de literatuur. Vervolgens worden de vereisten en bedenkingen opgesomd die tijdens het interview met de ontwerper van het systeem, professor Schreurs aan het licht kwamen.

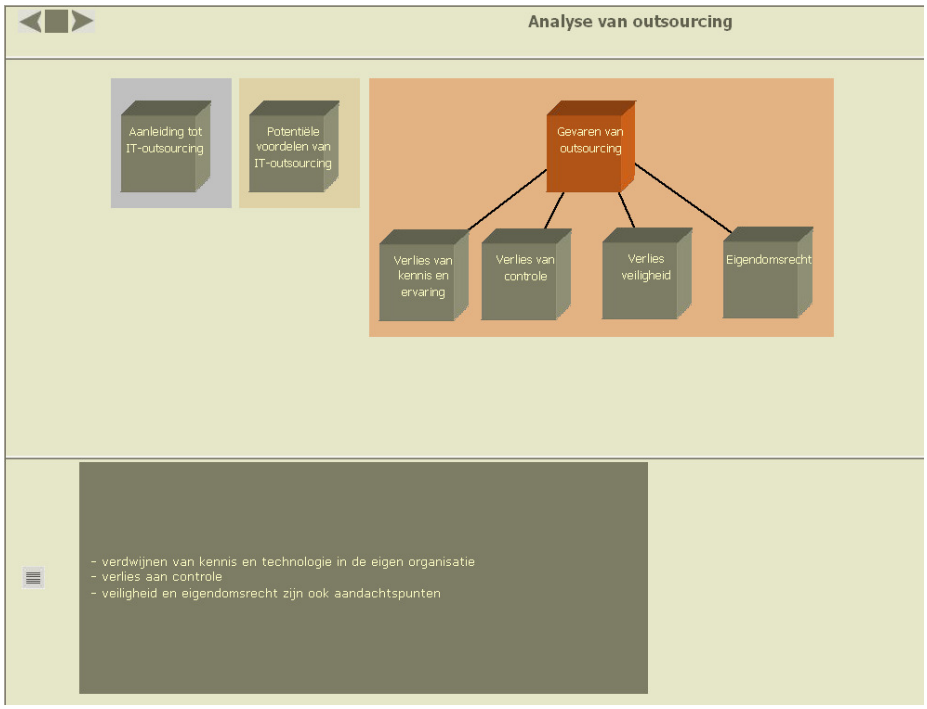
4.2.2 Het bestaande LOMS

De hiërarchie en de granulariteit van het LOMS worden eerst besproken, daarna komen de authoring omgeving en de technische werking aan de beurt. De metadata die door het LOMS worden bijgehouden sluiten dit onderdeel af.

Granulariteit

Het bestaande leerplatform is opgebouwd volgens het principe van leerobjecten. Het kleinste deel van het bestaande LOMS is een ALO of een 'atomic learning object'. Dit kan een tekst, een afbeelding of een audiobestand zijn. Deze ALO's staan op zichzelf. Ze hebben geen enkele onderlinge relatie. Een trapje hoger staan de 'blocks'. Blocks bestaan uit een aantal ALO's die samen een geheel vormen. Deze blocks zijn de bouwstenen voor de volgende trap in de hiërarchie: de LO zelf die hier gepresenteerd wordt als een 'mindmap'. Een mindmap is een verzameling van een aantal blocks, die samen een geheel vormen. De mindmap toont ook hoe deze blocks een geheel vormen door de onderlinge relaties visueel voor te stellen. (Schreurs en Moreau, 2006)

Tussen het niveau van blocks en mindmap is er nog een niveau. Het niveau van 'units'. Een aantal blocks kan tot een unit behoren. Units kunnen op hun beurt dan weer tot een mindmap behoren. De units bevatten de blocks en stellen ook voor hoe deze onderling gerelateerd zijn. Zo kan men in figuur 7 zien dat het geselecteerde block verder onderverdeeld kan worden in vier andere blocks. Men kan ook een visueel verschil zien tussen units doordat ze een andere achtergrondkleur hebben. Hierdoor geeft men visueel aan dat ze van het grotere thema van de mindmap een apart deel uitmaken.



Figuur 7. Mindmap 'Analyse van outsourcing'

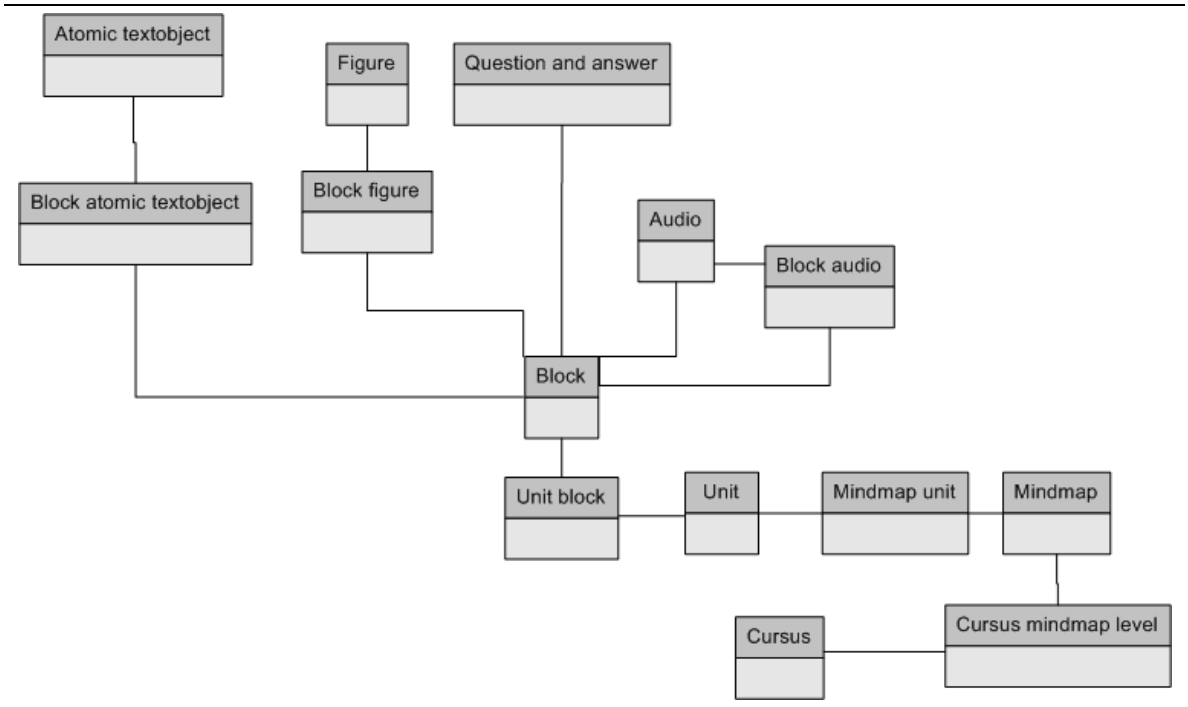
Eenmaal een mindmap gemaakt, kan men nog een stapje hoger gaan in de hiërarchie. Het volgende niveau is het 'cursus' niveau: meerdere mindmaps gebundeld tot een cursus.

E_schemas

Centraal in dit hele proces staat nu het systeem 'E_schemas'. Dit systeem heeft twee belangrijke functies. De eerste is de authoring functie op basis van de content in de onderliggende database, de tweede is het presenteren van mindmaps.
Het 'authoring' gedeelte laat de gebruiker toe zijn eigen mindmap aan te maken. Dit alles gebeurt via een web gebaseerd formulier. Men begint met het aanmaken van een block. Met behulp van het formulier geeft men gegevens over het aan te maken block in, bijvoorbeeld een titel, tot welke unit het behoort, of het een sub-block is of welke sequentie het block heeft. Eenmaal men het block heeft gespecificeerd, gaat men de ALO's (waaruit het block is opgebouwd) aanmaken. Als men alle benodigde blocks heeft aangemaakt kan men aan de mindmap beginnen. Doordat men bij het aanmaken van de blocks al gespecificeerd heeft tot welke unit ze behoren en hoe ze zich onderling verhouden (opeenvolging en/of het sub-blocks zijn) is het aanmaken van de mindmap vooral beperkt tot het invullen van metadata. Opmerkelijk hierbij is dat de datastructuur wordt bepaald tijdens de authoring. Hierdoor is de mindmap gebonden aan de onderliggende leercontent. De toegang tot de content verloopt via het programma E_schemas. Dit is de reden waarom deze authoring omgeving het niet toelaat bestaande mindmaps aan te passen of nieuwe te maken op basis van de bestaande content.

Indien men toch een mindmap wil aanpassen, moet men dit doen in de database of moet men een nieuwe mindmap aanmaken. Als men dan een fout maakt bij het opbouwen van de mindmap moet men opnieuw beginnen. Hierdoor bevat de database heel wat "mislukte" mindmaps.

Eenmaal de gebruiker de blocks en de mindmap heeft aangemaakt, gaat het systeem deze opslaan. Het grootste deel wordt opgeslagen in een database, de overige bestanden worden op de server opgeslagen.



Figuur 8. Vereenvoudigd overzicht van de database

Figuur 8 geeft weer hoe een mindmap en haar onderdelen opgeslagen worden in de database. De kleinste delen, de ALO's worden in aparte tabellen opgeslagen afhankelijk van hun type (atomic textobject, figure, question and answer of audio). Deze tabellen bevatten soms een link (figure, audio) of soms het ALO (atomic textobject, question and answer). Naast deze informatie bevat elk van deze tabellen een groot deel metadata over het ALO. De relatie tussen de ALO's en een block is 'meer' op 'één'. Dit houdt in dat bij elk block er één of meerdere ALO's horen, maar in elk block komt één ALO maximaal één keer voor. Om dit te bereiken zonder dataredundantie in de block tabel heeft men koppeltabellen (block atomic textobject, block figure, block audio) gecreëerd. Door deze koppeltabellen kan men meerdere ALO's toewijzen aan een block zonder de metadata van een block onnodig te herhalen. Als we het schema verder doorlopen zien we dat bij de overgang van block naar unit, van unit naar mindmap, en van mindmap naar cursus van hetzelfde systeem van koppeltabellen gebruik wordt gemaakt.

Eenmaal alles goed in de database is opgeslagen, kan E_schemas de mindmap aan de hand van de inhoud die tijdens het authoring proces in de database werd opgeslagen voorstellen aan de gebruiker.
Doordat de twee onderdelen van E_schemas de database als buffer gebruiken kunnen we stellen dat ze onafhankelijk zijn van elkaar.

Metadata

In het bestaande systeem komen we metadata tegen op alle niveaus van de hiërarchie. Naast deze hiërarchische opsplitsing is ook een meer beschrijvende opsplitsing mogelijk. Zo kan men metadata opsplitsen in twee categorieën: de technical oriented metadata en de learning oriented metadata. Enkele voorbeelden van technical oriented metadata zijn 'date of creation' en 'last adjustment'. Mogelijke voorbeelden van learning oriented metadata zijn 'learning level' en 'learning area'. (Schreurs en Moreau, 2006)

Op het laagste niveau in de hiërarchie bevinden zich de ALO's. Deze hebben het grootste deel van de metadata gemeenschappelijk (tabel 1). Alle learning oriented metadata (Title en Description) zijn gelijk voor alle types van ALO's. Het grootste deel van de technical oriented metadata komt ook overeen. De enige categorieën die kunnen verschillen zijn 'Format' en 'Type' (Id_type, Id_type_audio). Deze categorieën bevatten een parameter, in dit geval een getal, dat aan het systeem doorgeeft welk type het ALO is.
De reden van opsplitsing in Format en Type is niet helemaal duidelijk.

Tabel 1
Overzicht metadata ALO's

Atomic_textobjectsField	**Audio**	**Figure**
Creation_date	Creation_date	Creation_date
Format		Format
Copyright	Copyright	Copyright
Title	Title	Title
Description	Description	Description
Id_type	Id_type_audio	
Text	audio	Figure

De blocks zijn het niveau boven de ALO's. De blocks bezitten op hun beurt weer veel metadata (tabel 2). Deze hebben grotendeels dezelfde structuur als bij de ALO's. We onderscheiden twee categorieën: learning oriented metadata (Title en Description) en technical oriented metadata (Format en Id_qa).

Tabel 2

Overzicht metadata block

Block
Id_audio
Creation_date
Format
Copyright
Title
Description
Id_qa

Een trapje hoger dan het block niveau bevindt zich het unit niveau. De metadata die bij een unit horen, hebben een dubbele functie. Enerzijds beschrijven ze de unit, anderzijds bevatten ze de lay-out van de blocks die tot deze unit behoren. Tabel 3 toont ons deze metadata. We zien weer de gewone categorieën en ook enkele interessante nieuwe. Zo zijn Language, Learning_level en Learning_area belangrijke learning oriented metadata. Als men een gedeelte wil hergebruiken is het belangrijk om te weten in welke taal het stukje leerstof zich bevindt, hoe moeilijk het is en bij welk vak het nu precies aansluit.

Men kan zich de vraag stellen of het nuttig zou zijn dit op een vroeger niveau (block of ALO) op te nemen.

Tabel 3

Overzicht metadata unit

Unit	**Unit layout**
Title	Id_block
Learning_level	Sequence_block
Learning_area	Childof_id_block
Description	Position
Creation_date	
Last_adjustment	
Elapsed_time	
Language	
Format	
Copyright	

Een aantal units samen vormt een mindmap. De metadata voor de mindmap worden in drie delen opgedeeld. De mindmap, de summary van de mindmap en een lay-out component (tabel 4). De metadata van de mindmap zijn dezelfde als die van een unit, wanneer de kolom ‘Topic’

buiten beschouwing wordt gelaten. Deze kolom wordt gebruikt om aan te geven bij welk topic een mindmap hoort volgens de bibliotheekclassificatie.

Tabel 4
Overzicht metadata mindmap

Mindmap	**Summary_mindmap**	**Mindmap layout**
Title	Creation_date	Id_unit
Description	Format	Position
Creation_date	Copyright	Sequence_unit
Last_adjustment	Title	
Format	Description	
Copyright	Id_audio	
Topic	id_typeIS	
Learning_level		
Learning_area		
Language		
Omschrijving		

Cursus is de laatste component in de hiërarchie (tabel 5). Het valt op dat een aantal learning oriented metadata elementen is weggevallen ten opzichte van een mindmap. Zo zijn er geen velden meer voorzien voor 'Copyright' en 'Topic'.

Tabel 5
Overzicht metadata cursus

Cursus
Title
Description
Learningdomain
Learninglevel
Language
Creation_date
Aantalhoofd
Aantallevel
Aantalsub
Einde

Algemeen kunnen we besluiten dat in het bestaande systeem op verschillende niveaus metadata worden voorzien, zowel voor de gebruiker (Title, Description, Copyright,…) als voor het systeem

intern (Creation_date, Last_adjustment, …). Heel wat categorieën komen terug op verschillende niveaus. Hierdoor kunnen de metadata gedeeltelijk overgeërfd worden. (Schreurs en Moreau, 2006)

4.2.3 Evaluatie van het LOMS aan de hand van de literatuur

De literatuurstudie in het eerste deel heeft aangetoond dat de leerobjecten binnen een systeem dat herbruikbaarheid beoogt aan drie eigenschappen moeten voldoen: discoverabiliteit, interoperabiliteit en modulariteit. In dit onderdeel wordt het huidige systeem op deze drie eigenschappen beoordeeld en, indien nodig, worden corrigerende maatregelen voor een nieuw verbeterd LOMS voorgesteld.

De discoverabiliteit in het LOMS is redelijk goed. Door het implementeren van metadata weet zowel het systeem als de gebruiker waar het over gaat. Daarnaast is er ook een goede opdeling gemaakt voor metadata op de verschillende hiërarchische niveaus. Zo heeft elke ALO, elk block, elke unit, elke mindmap en elke cursus zijn eigen metadata. Deze metadata omvatten zowel de technische als de inhoudelijke kant.
Het bestaande systeem voldoet echter niet aan de huidige metadata standaard. Bij zijn ontwikkeling voldeed het wel aan deze standaard, maar sindsdien voorziet de standaard meer metadata elementen.

Alle tekst- en mediabestanden die tot een atomisch leerobject behoren, zijn van een veel voorkomend type. De samenwerking met de browser is dan ook gegarandeerd. De overige informatie van de ALO's zit opgeborgen in een MySQL database en is dus eenvoudig te bereiken met SQL. Er kan gesteld worden dat de interoperabiliteit van de mindmaps goed is.
Het bestaande systeem is niet ontworpen volgens een gestandaardiseerde architectuur. Het systeem maakt wel gebruik van algemene protocollen en systemen, die de interoperabiliteit steunen. De protocollen en basissystemen (de gebruikte programmeertaal en databasesoftware) zijn immers verkrijgbaar en bruikbaar voor iedereen. Zo draait het systeem op een Mysql server en is het geprogrammeerd in Asp.
Ook voorziet het LOMS geen mogelijkheid tot het opslaan van een LO in een file volgens algemene standaarden, zodat het niet mogelijk is om een file zonder meer 'leesbaar' naar een andere gebruiker te sturen.

De modulariteit handelt over de manier waarop het leerobject wordt aangemaakt, opgebouwd en opgeslagen (ALO's, blocks, units, mindmaps). De opbouw van de mindmaps in dit systeem gebeurt volgens een vaste hiërarchie: ALO's, blocks, units en mindmaps. Alle onderdelen worden opgeslagen in de database waar ze gemakkelijk te bereiken zijn en van waaruit ze hergebruikt kunnen worden.

De mindmaps zijn onderling volledig onafhankelijk van elkaar, ze zijn niet sequentieel en vormen een eenheid op zichzelf (de leerstof die een mindmap bevat is een "af" verhaal). De modulariteit van de mindmaps kan dus ook als goed bestempeld worden.

Hoewel de drie voorwaarden voor hergebruik van leerobjecten voldaan zijn is het niet mogelijk om leercontent en leerobjecten te hergebruiken in het bestaande systeem E_schemas.
We kunnen wel besluiten dat we de bestaande relationele database kunnen gebruiken als eerste laag in onze architectuur en dat we hier geen correcties op moeten doorvoeren. Zij voldoet namelijk aan alle voorwaarden die herbruikbaarheid vooropstelt.

4.2.4 Informatie over het LOMS uit het interview

Na het bestuderen van de literatuur en het systeem had ik een gesprek met professor Schreurs. Tijdens dit interview werden vooral de gebruikerservaring en de tekorten van het systeem onder de loep genomen. Ook waren er bijkomende bedenkingen over de vereisten waaraan het nieuwe systeem zou moeten voldoen.

Een eerste belangrijke vereiste is, dat het nieuwe systeem met de database van het oude systeem moet blijven werken. Als er tabellen worden bijgevoegd aan de huidige database moeten deze compatibel zijn met het oude systeem. Bestaande tabellen kunnen niet aangepast worden aangezien er geen data bestaan voor eventuele nieuwe 'kolommen'.
Onze 3-lagige architectuur voldoet hier vermits de eerste laag de huidige relationele database is.

Een tweede belangrijk punt behandelt de uitwisselbaarheid van de mindmaps. Op dit moment kan men slechts moeizaam een mindmap aan iemand doorgeven. Hier moet een oplossing voor gezocht worden en wel op twee manieren.
Een eerste manier is door elke mindmap een uniek adres geven. Op dit moment gebeurt dit nog niet, omdat mindmaps nu nog dynamisch gegenereerd worden. Als een leerling een mindmap wil bekijken activeert hij een programma. Het programma bevindt zich op de url: http://lspserver.luc.ac.be/eschemapers/framemindmap1.asp. Na de activatie genereert het programma HTML pagina's die de mindmap bevatten. Door dit proces hebben de verschillende mindmaps geen verschillende url's. Toch zijn deze verschillende url's nodig. Zo zal men, wanneer men een mindmap voorziet in een cursusportaal deze hieraan willen linken. Op dit moment moet eerst een statische pagina voorzien worden waaruit de dynamische mindmap aangeroepen wordt.
Het toevoegen van een mindmap identificatienummer aan de URL lost dit probleem op. Het identificatienummer zal voorgesteld worden door een variabele. De nieuwe url zal er dan bijvoorbeeld als volgt uitzien:

'http://lspserver.luc.ac.be/eschemapers/framemindmap1.php?id=< identificatienummer>'

De tweede manier zou het off-line bekijken van een mindmap eenvoudiger moeten maken. Op dit moment kan men moeizaam een mindmap exporteren en bijvoorbeeld doormailen. Als men nu een mindmap off-line wil bekijken moet men de bronbestanden bezitten en een Mysql en ASP server installeren.
Dit is eigenlijk een dubbel probleem. Het eerste is dat men mindmaps niet gemakkelijk off-line kan bekijken.
Er zou een manier moeten ontwikkeld worden om dynamische ASP pagina's van een mindmap om te zetten naar statische HTML pagina's. Hierdoor kan de mindmap bekeken worden zonder een server te installeren.
Het tweede probleem is dat men mindmaps niet kan archiveren tot één bestand en uitwisselen.
Men zou een methode moeten zoeken om de queries van de database en de nodige files om te zetten in een 'pif' (package interchange file). Een pif is een file speciaal ontwikkeld voor het archiveren van alle commando's en files van een mindmap in één bestand. Omdat er standaardregels zijn opgesteld voor deze file kan elk systeem dat aan deze regels voldoet, een pif bestand inlezen en gebruiken.

Een derde punt ter verbetering is de mogelijkheid om eenzelfde mindmap op verschillende manieren te presenteren. Zo heeft men de mogelijkheid om scenario's in te bouwen in het bestaande systeem.
Dit probleem sluit aan bij de reeds gestelde specificatie voor het nieuwe systeem dat het weergeven van data aanpasbaar moet zijn aan de dominante leerstijl en de situatie van de leerling.

Een laatste punt handelde over de implementatie van nieuwe file extensies. Zoals opgemerkt in de bespreking van de database kan het systeem maar samenwerken met een zeker aantal soorten bestanden.

5. VOORSTEL TOT VERBETERING VAN HET LOMS: HET GEWENSTE E-LEARNING SYSTEEM

In hoofdstuk 4 werd een 3-lagige architectuur voorgesteld die de herbruikbaarheid van de leerinhoud van het bestaande systeem verbetert en scenario's mogelijk maakt. Onderzoek van het huidige systeem heeft ons geleerd hoe het werkt, welke data het opslaat en wat de knelpunten zijn. Op basis van deze informatie wordt in de volgende onderdelen beschreven hoe we de 3-lagige architectuur kunnen integreren in het LOMS rekening houdend met de beperkingen opgelegd door de ontwerper.
Eerst wordt de tweede laag ontworpen aan de hand van 'object oriented analysis and design' (OOA/D). Daarna wordt het ontworpen programma besproken. De derde laag wordt met dezelfde methode ontworpen. Afsluitend wordt ook deze oplossing besproken.

5.1 ONTWIKKELING VAN DE TWEEDE LAAG: 'GENERIC LEARNING OBJECT ASSEMBLING MODULE'

De tweede laag van de nieuwe architectuur heeft twee functies. Ze geeft de gebruiker de mogelijkheid om nieuwe leerobjecten aan te maken en levert het leerobject aan de derde laag.

De nieuwe leerobjecten worden aangemaakt op basis van de bestaande leerobjecten. Een gebruiker kan een nieuw leerobject aanmaken door een bestaand leerobject te selecteren. Na deze selectie krijgt hij een overzicht van de blocks die tot dit leerobject behoren. Aan de hand van dit overzicht selecteert de gebruiker welke blocks hij tot het nieuwe leerobject wil laten behoren. Door middel van deze keuzes gaat het programma een nieuw leerobject aanmaken en opslaan in de bestaande database. Voor de duidelijkheid hernoemen we de database tabel 'mindmap' naar 'LO' en de tabel 'mindmap unit' naar 'LO unit'. (zie bijlage 1)

Tijdens het voorstellen en opslaan van het nieuwe leerobject moet het programma het leerobject door middel van zijn relaties modelleren in het werkgeheugen van de computer. Eenmaal dit virtuele leerobject gecreëerd is, kan men op basis van korte commando's de inhoud uit dit leerobject opvragen. Het zijn deze functies van het programma waarop de derde laag beroep gaat doen bij het presenteren van het leerobject.

5.1.1 Systeemontwerp

In dit deel wordt het idee van de 'learning object assembling module' uitgewerkt tot een module die in het LOMS geïmplementeerd kan worden. Het eindresultaat van dit onderdeel zal een UML schema zijn. UML staat voor 'unified modelling language' en is een manier om softwareprogramma's theoretisch voor te stellen.
Hierbij wordt gebruik gemaakt van de methode van C. Larman (2002) beschreven in "UML and Patterns". Deze auteur werkte een methode uit voor het omzetten van een probleem in een softwareschema gebaseerd op OOA/D (object oriented design and analysis).

OOA/D bestaat uit twee delen, een 'analyse' gedeelte en een 'ontwerp' gedeelte. Beide delen concentreren zich op het denken in en ontwikkelen van objecten. Zo worden in de analyse de omgeving en de taken die het systeem moet vervullen weergegeven in objecten en relaties tussen de objecten. In het design gedeelte worden de objecten en hun relaties, die gecreëerd werden in het eerste deel, verder verfijnd. Ze krijgen attributen en taken. Deze verfijnde objecten worden dan in een laatste stap omgezet in objecten die softwareontwikkelaars kunnen gebruiken bij het schrijven van software. In de methode van Larman wordt het voorgaande principe samengebracht in UML diagrammen en schema's. (Larman, 2002)

5.1.1.1 Use case

Een eerste stap in de methode van Larman is het ontwikkelen van een 'use case'. Een use case wordt gedefinieerd als een schema dat aangeeft wat de functies van het systeem moeten zijn. Voor de use cases wordt gebruik gemaakt van de template die Larman ons ter beschikking stelt.

In het eerste gedeelte (primary actor, stakeholders and their interests) verklaar ik voor wie de software, en dus ook de use case wordt opgesteld. Daarna bekijk ik de voorwaarden die altijd vervuld moeten zijn; dat zijn de 'preconditions'. De 'success condition' vertelt ons aan welke voorwaarden het systeem op het einde moet voldoen. Het 'main success scenario' geeft een overzicht van een normale interactie met het systeem. Aanvullend op dit main success scenario sluit ik deze use case af met de alternatieve gevallen.

Use case name: Aanmaken leerobject
Primary actor: Maker leerobject

Stakeholders and their interests:
- Maker van het LOMS:

        - Hij wil dat zijn leerobject wordt weergegeven zoals hij het gemaakt heeft.
        - Hij wil kunnen selecteren welke content tot zijn leerobject behoort.
    - Beheerder van het LOMS:
        - Hij wil dat het systeem eenvoudig te onderhouden is.

Preconditions:

De content bestaat in de database en de maker is gemachtigd om de content te bekijken en een leerobject aan te maken.

Success condition: De maker kan het leerobject aanmaken en opslaan.

Main success scenario:

1. De leerobject maker komt op het "zoek content" scherm.
2. De leerobject maker geeft aan het systeem aan welke content hij wil gebruiken om een leerobject aan te maken.
3. Het systeem geeft een confirmatie dat de content bestaat.
4. Het systeem vraagt aan de database de onderdelen van de content op.
5. Het systeem benoemt de onderdelen en geeft ze weer.
6. De leerobject maker geeft aan welke onderdelen hij in zijn leerobject wil.
7. Het systeem slaat het leerobject van de maker op.
8. De leerobject maker geeft aan terug naar het zoekmenu te willen.
9. Het systeem geeft het "zoek content" scherm.

Alternative cases:

Op elk moment.

Een foutmelding treedt op. De foutmelding wordt aan de gebruiker getoond. De leerobject maker wordt naar het "zoek content" scherm gebracht.

2 De content wordt niet in de database gevonden.

Het systeem geeft de boodschap: "De gevraagde content <naam> werd niet gevonden.", daarna brengt het de leerobject maker terug naar het "zoek content" scherm.

Use case name: Opvragen leerobject
Primary actor: LOMS gebruiker/leerling

Stakeholders and their interests:

- LOMS gebruiker/leerling:
  - Hij wil het door hem gekozen leerobject bekijken.
  - Hij wil dat dit snel laadt.
  - Hij wil het juiste leerobject zien.
  - Hij wil een onderdeel (block) van het leerobject meer gedetailleerd kunnen bekijken.
- Maker van het LOMS:
  - Hij wil dat zijn leerobject wordt weergegeven zoals hij het gemaakt heeft.
- Beheerder van het LOMS:
  - Hij wil dat het systeem eenvoudig te onderhouden is.

Preconditions:

Het leerobject bestaat in de database en de gebruiker is gemachtigd om het leerobject te bekijken.

Success condition:

De gebruiker kan het leerobject bekijken met verschillende lay-outs.

Main success scenario:

1. De LOMS gebruiker/leerling komt op het "zoek leerobject" scherm.
2. De LOMS gebruiker/leerling geeft aan het systeem aan welke leerobject hij wil bekijken.
3. Het systeem geeft een confirmatie dat het leerobject bestaat.
4. Het systeem vraagt aan de database de onderdelen van het leerobject op.
5. De LOMS gebruiker/leerling navigeert door het leerobject door het selecteren van blocks.
6. Het systeem geeft de inhoud van een geselecteerd block weer.
7. De LOMS gebruiker/leerling geeft aan terug naar het zoekmenu te willen.
8. Het systeem geeft het "zoek leerobject" scherm.

Alternative cases:

Op elk moment.

Een foutmelding treedt op. De foutmelding wordt aan de gebruiker getoond. De LOMS gebruiker/leerling wordt naar het "zoek leerobject" scherm gebracht.

2 Het leerobject wordt niet in de database gevonden.

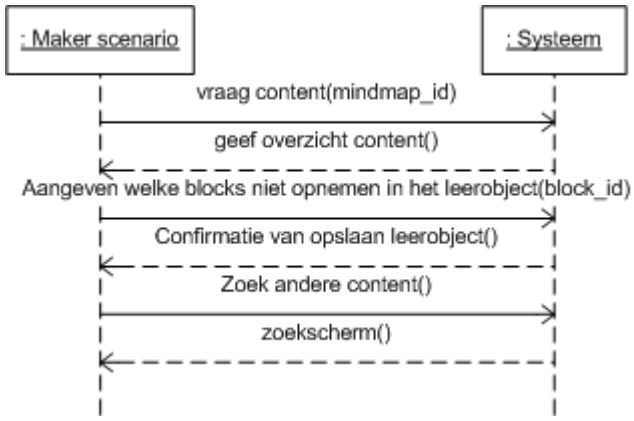
Het systeem geeft de boodschap: "Het gevraagde leerobject <naam> werd niet gevonden.", daarna brengt het de LOMS gebruiker/leerling terug naar het "zoek leerobject" scherm.

6 Het geselecteerde block bestaat niet.

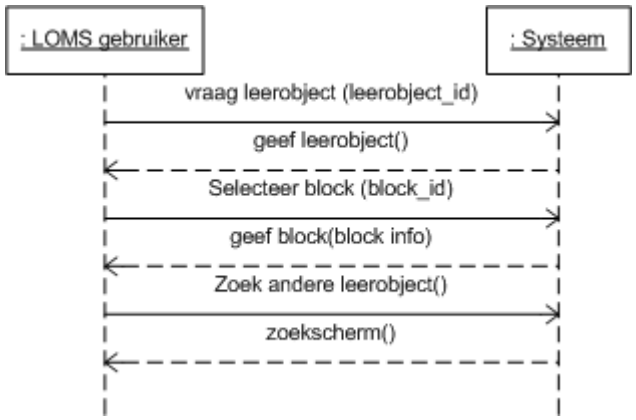
Het systeem geeft de boodschap: "Het gevraagde block <naam> bestaat niet.", daarna brengt het de LOMS gebruiker/leerling terug naar de indexpagina van het geselecteerde leerobject.

5.1.1.2 System sequence diagram

Eenmaal de use case opgesteld is, kan een 'system sequence diagram' gemaakt worden. Een SSD is een afbeelding die een bepaalde use case toont door de interacties tussen de gebruiker en het systeem weer te geven.



Figuur 9. System sequence diagram voor 'aanmaken leerobject'



Figuur 10. System sequence diagram voor 'opvragen leerobject'

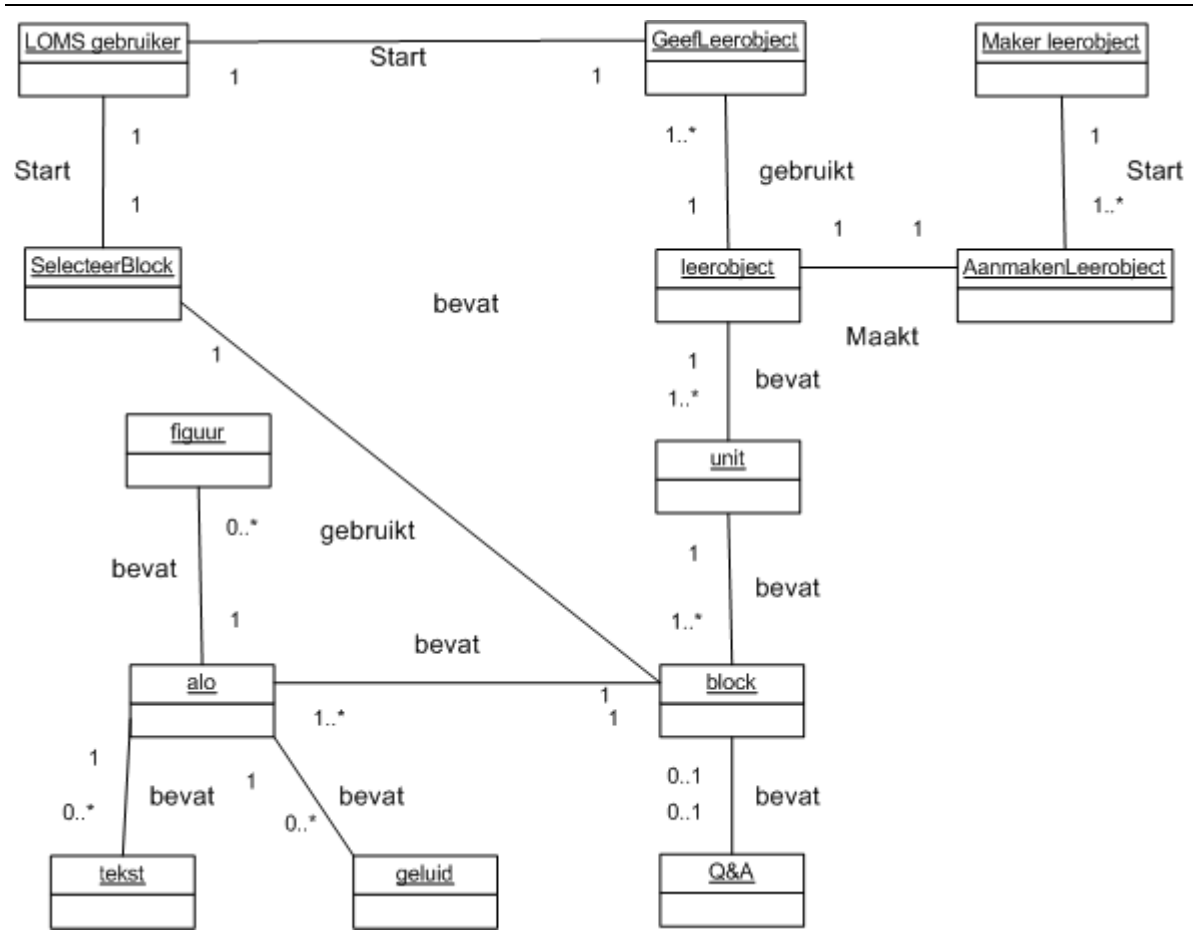
5.1.1.3 Domeinmodel

Het maken van de use cases en de SSD's wordt bij Larman gevolgd door het 'domeinmodel'. Dit is een visuele representatie van domeinobjecten of conceptuele klassen. Deze representatie focust op de objecten zelf en op de relaties tussen de objecten. Het opbouwen van dit model doet Larman in twee stappen. Eerst worden de domeinobjecten geïdentificeerd. Dit gebeurt aan de hand van de use case en een lijst van veel voorkomende conceptuele klassen.

Tabel 6
De domeinobjecten

Lijst van veel voorkomende klassen		**Use case**
Reële objecten	*Processen*	-
Leerobject	AanmakenLeerobject	LOMS gebruiker
Unit	SelecteerBlock	Maker Leerobject
Block	GeefLeerobject	
ALO		

In een volgende stap worden relaties tussen deze objecten gelegd. De naam van elke relatie geeft op twee manieren weer hoe de objecten zich onderling verhouden. De eerste manier wordt uitgedrukt in de naam zelf en geeft aan of het object het andere object start, gebruikt of bevat. De tweede manier drukt de numerieke verhouding uit: "Hoeveel instanties van een object bestaan er ten opzichte van een ander object?" Dit wordt ook 'multipliciteit' genoemd en wordt aangeduid door de cijfers naast de objecten.



Figuur 11. Domeinmodel met relaties

5.1.1.4 Sequentie- en collaboratiediagrammen

Larman beschrijft de volgende stap van het ontwikkelingsproces als het 'selecteren en uitwerken van de acties die de omgeving van het systeem verwacht'.

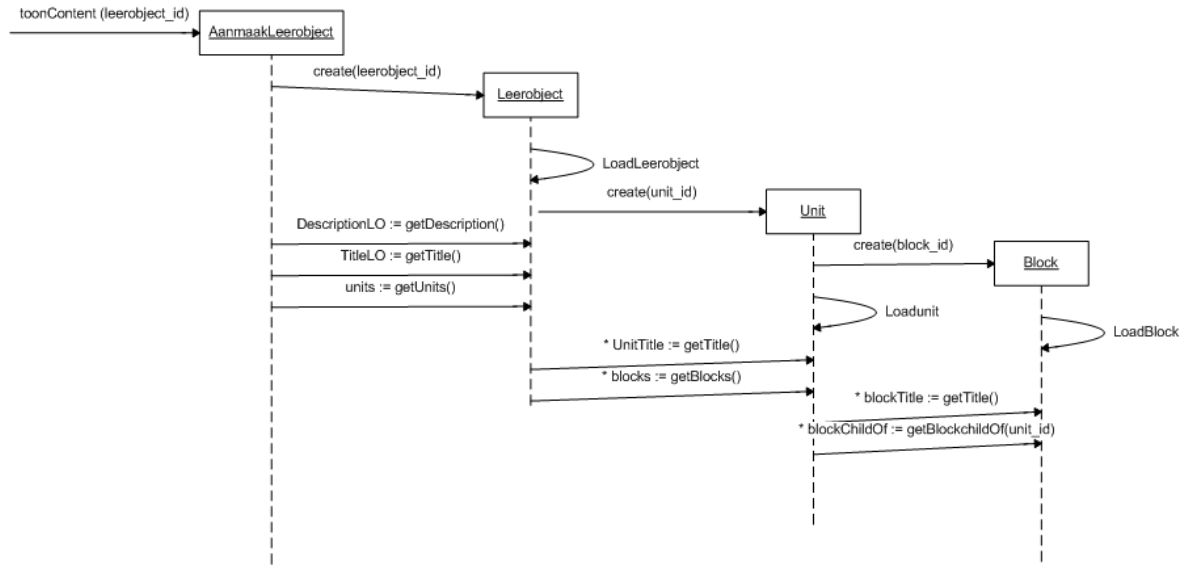
Die acties worden eerst geselecteerd. Hiervoor moet er vastgesteld worden, *wat* de omgeving nu *exact* van het systeem verlangt. De use case en het SSD diagram worden hiervoor gehanteerd. Larman stelt hier voor om de return acties in het SSD diagram te bekijken. Deze zijn:'geef leerobject, 'geef block', 'geef overzicht content' en 'geef confirmatie opslaan leerobject'. Deze vier acties worden verder verkend in de use case en de vragen: 'Wat verwacht de gebruiker van de actie?' en 'Welke input voorziet hij?' worden beantwoord. Wanneer de systeemacties duidelijk zijn, krijgen ze een actieve naam. In dit geval: 'geef leerobject', 'selecteer block' , 'toon content' en 'maak leerobject'.

Vervolgens wordt uitgewerkt hoe het systeem de acties intern afhandelt. Ook voor deze stap staan er hulpmiddelen ter beschikking. Ze worden interactiediagrammen genoemd. Interactiediagrammen zijn schema's die weergeven welke onderdelen van een systeem actief worden, welke informatie ze nodig hebben, welke boodschappen ze naar de andere onderdelen van het systeem sturen en hoe ze dit doen. Larman geeft twee soorten interactiediagrammen: het 'sequence diagram' en het 'collaboration diagram'.
De eerste soort focust vooral op de volgorde van de acties die het systeem intern uitvoert en op welk deel van het interne systeem de actie bezit. De tweede soort richt zich vooral op de wijze waarop de onderdelen deze informatie naar elkaar communiceren.

Toon content

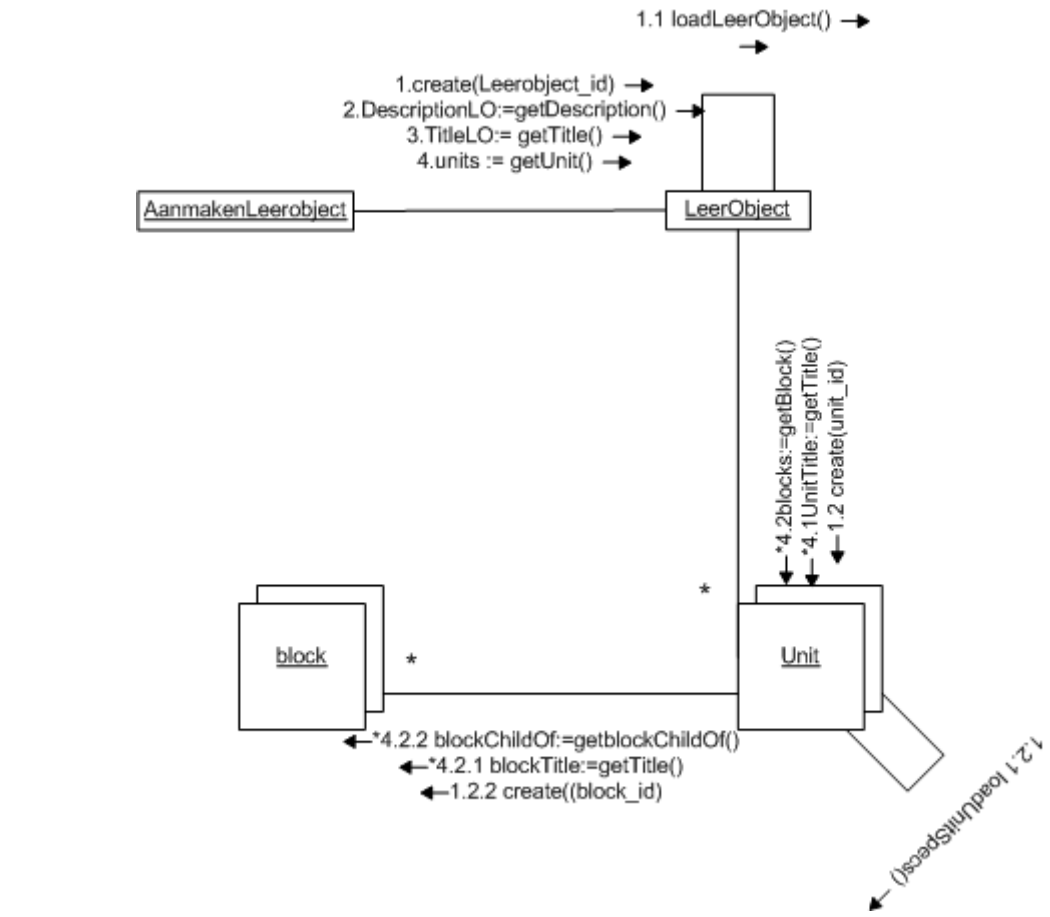
"Toon content" is de eerste actie die wordt uitgewerkt. In deze actie stelt het systeem de blocks en units voor van het gekozen leerobject aan de gebruiker. Het voorstellen gebeurt aan de hand van een formulier met checkboxes voor iedere block naam (figuur 21). De gebruiker vinkt de checkboxes af van de blocks die hij wil behouden in het nieuwe leerobject, vult de metadata van het nieuwe object in en verstuurt het formulier.
Figuur 12 toont het sequentiediagram van "Toon content". De pijl die van links komt stelt de input van de gebruiker voor. De user geeft het identificatienummer mee van de content die hij wil bewerken. Aan de hand van dit nummer wordt een object 'AanmakenLeerobject' gecreëerd. 'AanmakenLeerobject' doet hier dienst als 'controller klasse'. Een controller klasse is een object dat men gebruikt voor het initiëren van de gebruikersinput. Het object 'AanmakenLeerobject' creëert nu een object 'Leerobject'. Voor de creatie van 'Leerobject' geeft 'AanmakenLeerobject' het leerobject identificatienummer mee. Aan de hand van dit nummer haalt het object 'Leerobject' de nodige gegevens uit de database. Bij de opgevraagde gegevens zitten de identificatienummers van de units. Aan de hand van deze nummers worden de objecten 'Unit' aangemaakt. Deze objecten worden volgens het principe "by creator" door het object 'Leerobject' aangemaakt. Dit principe kan toegepast worden, omdat units tot een leerobject behoren. De 'Unit' objecten vragen op hun beurt de nodige data uit de database op en slaan ze op. Elke 'Unit' heeft nu een lijst van block nummers die tot de unit behoren. Volgens hetzelfde principe als bij de overgang 'Leerobject' – 'Unit' maken de 'Unit' objecten nu de objecten 'Block' aan. De 'Blocks' raadplegen op hun beurt de database en halen de nodige data op. De structuur is nu helemaal aangemaakt.
In de volgende stap gaan we de informatie, die we aan de gebruiker willen laten zien, uit de structuur halen. Het object 'AanmakenLeerobject' gaat hiervoor de titel en de beschrijving (description) van het 'Leerobject' opvragen. Eenmaal deze gegevens ontvangen, vraagt het object 'AanmakenLeerobject' de objecten 'Unit' op van 'Leerobject'. De ontvangen 'Unit' objecten doorlopen dan volgende lus. Van elk 'Unit' object worden de titel en de objecten 'Block', die tot de unit behoren, opgevraagd. Via een geneste lus wordt ook van ieder object 'Block' de titel opgevraagd en of het om een sub-block gaat van een andere block. Al de

verzamelde informatie uit deze structuur wordt weergegeven aan de gebruiker. We sorteren de informatie bij het weergeven, zodat de blocks van een bepaalde unit onder de titel van deze unit staan. Daarnaast worden de blocks die een sub-block zijn onder hun ouder weergegeven en met een kleine insprong. Ook wordt er een checkbox voorzien voor elk block. Zo kan de gebruiker aangeven of hij de block wil opnemen in het nieuwe leerobject.



Figuur 12. Sequentiediagram "Toon content"

Na het creëren van het sequentiediagram voor de actie "Toon content", wordt een collaboratiediagram gemaakt. Dit doen we omdat collaboratiediagrammen de vertakkingen van de berichtgeving duidelijker weergeven.



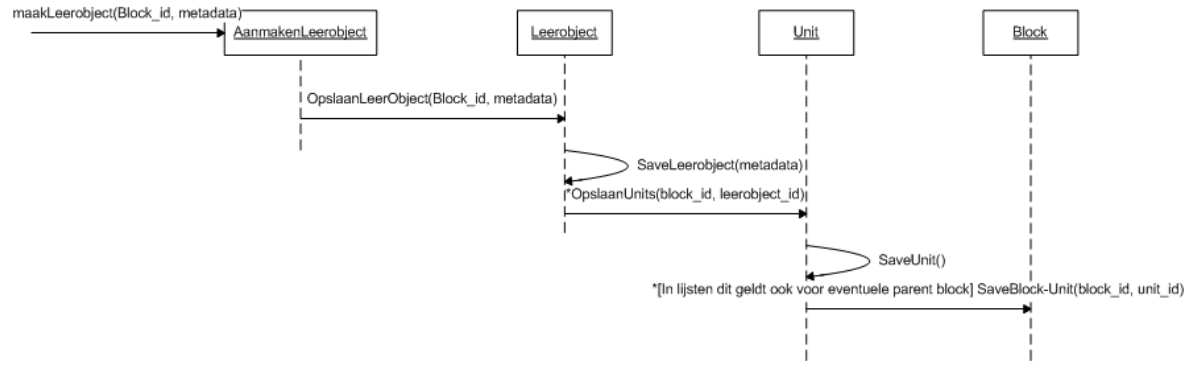
Figuur 13. Collaboratiediagram "Toon content"

Maak leerobject

De tweede actie is "Maak leerobject" en werkt verder op de objecten aangemaakt in "Toon content". In deze actie gaat het programma aan de hand van de gekozen blocks en de ingevulde metadata het nieuwe leerobject opslaan in de database.

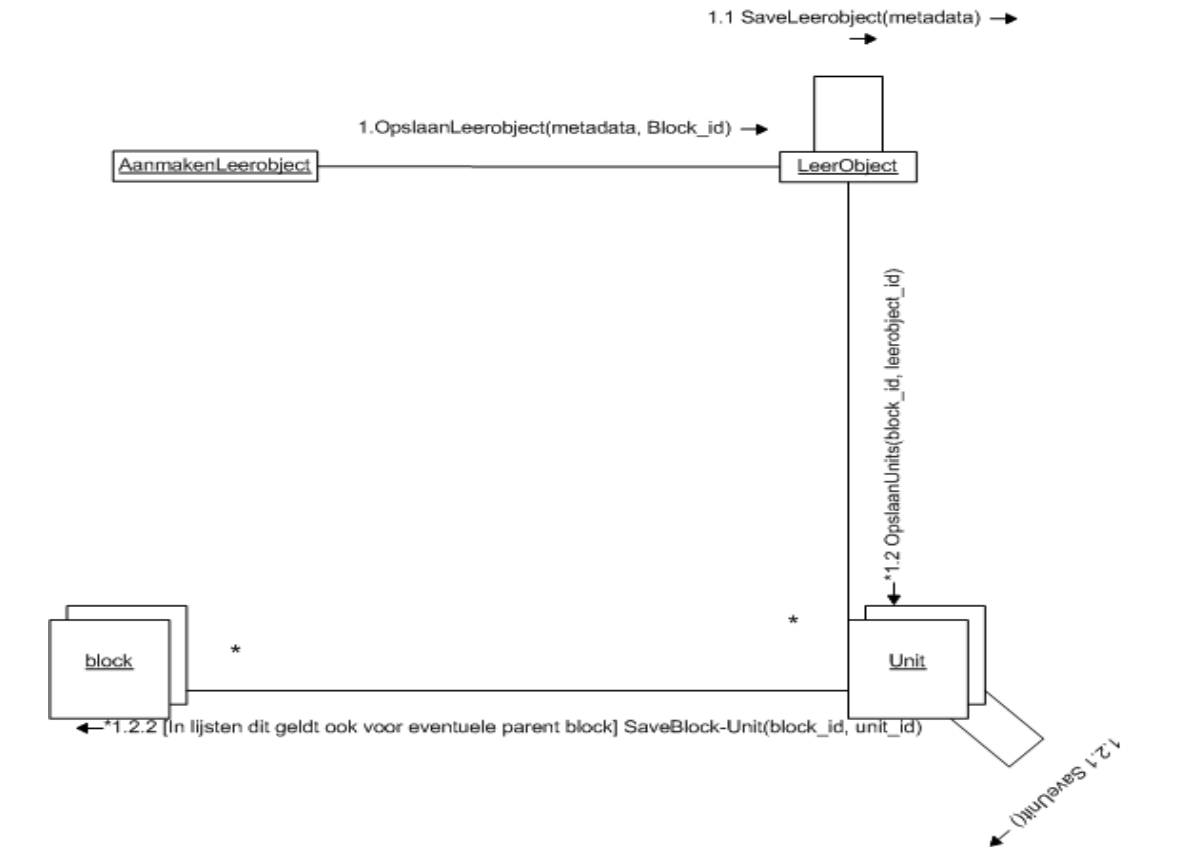
"Maak leerobject" begint met gebruikersinput die wordt gegeven aan de controller klasse, 'AanmakenLeerobject'. De input bestaat uit een lijst van blokken die de gebruiker in het leerobject wil zien opgenomen worden en de metadata (titel, description, topic_id, learning level en learning area) voor het nieuwe leerobject. De lijst en metadata worden doorgestuurd naar het object 'Leerobject'. Op basis van de metadata schrijft 'Leerobject' een nieuwe record in de tabel LO (Bijlage 1), dit wordt gedaan door de functie 'SaveLeerobject'. De lijst van blocks geeft het 'Leerobject' daarna door aan de 'Unit' objecten via de functie 'OpslaanUnits'. Na het uitvoeren van 'OpslaanUnits' is het de beurt aan de functie 'SaveUnit'. Deze functie voegt in de tabel Unit voor elke 'Unit' een regel toe met de inhoud van de actieve 'Unit'. Daarna voegt 'SaveUnit' per unit een regel in de koppeltabel Unit LO in. Deze regel bevat het identificatienummer van de unit en het leerobject. Zo legt de functie de relatie tussen het

leerobject en zijn units. Om deze actie af te ronden wordt de functie 'SaveBlock-Unit' uitgevoerd. Deze functie wordt voorafgegaan door een conditie. Als het identificatienummer van het object 'Block' en zijn eventuele parent 'Block' opgenomen zijn in de lijst van gekozen blocks, voegt de functie 'SaveBlock-Unit' een regel toe in de tabel 'Unit block'. Indien het nummer niet is opgenomen in de lijst, dan voegt ze geen regel toe in de tabel.



Figuur 14. Sequentiediagram "Maak leerobject"

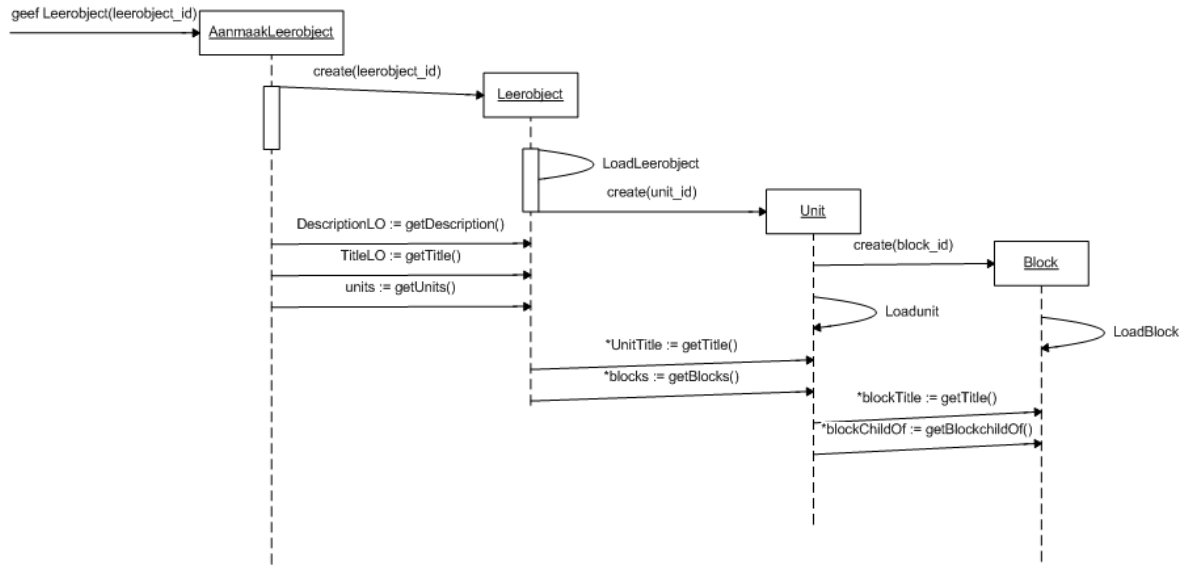
Ook na dit sequentiediagram maken we een collaboratiediagram van de actie op.



Figuur 15. Collaboratiediagram "Maak leerobject"

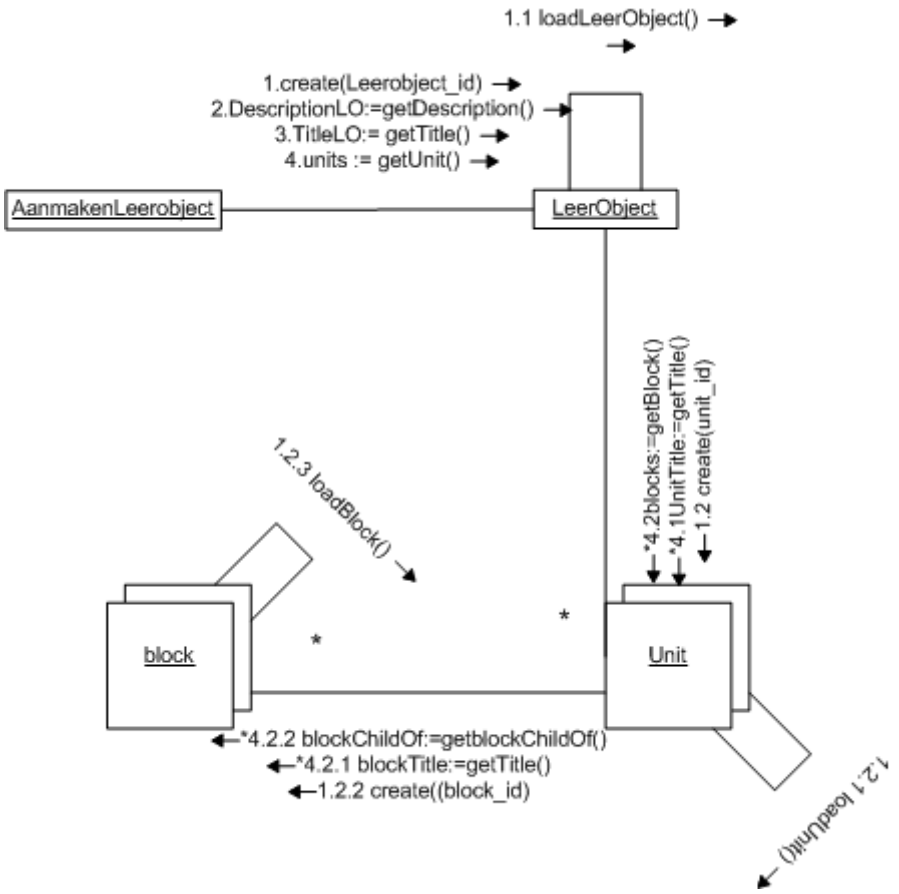
Geef leerobject

De volgende actie "Geef leerobject" loopt analoog met "Toon content". Figuur 16 toont het sequentiediagram van deze actie. Het grote verschil is dat bij "Toon content" de informatie in een formulier werd weergegeven waar de gebruiker blocks kon kiezen. Bij deze actie wordt de informatie zonder formulier weergegeven. De gebruiker kan klikken op de blocks om hun inhoud te zien.



Figuur 16. Sequentiediagram "Geef leerobject"

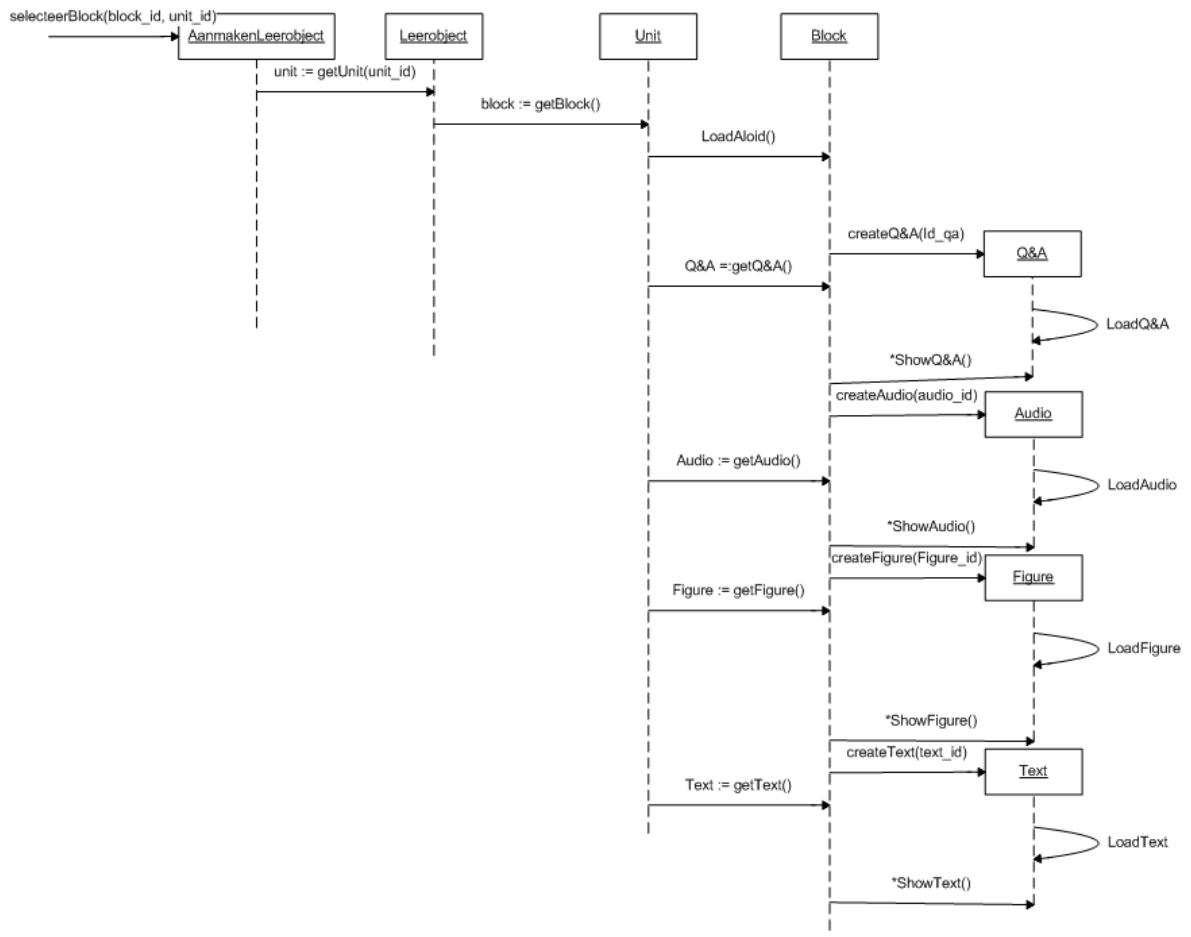
Na het creëren van het sequentiediagram voor de actie "Geef leerobject", wordt een collaboratiediagram gemaakt voor "Geef leerobject".



Figuur 17. Collaboratiediagram "Geef leerobject"

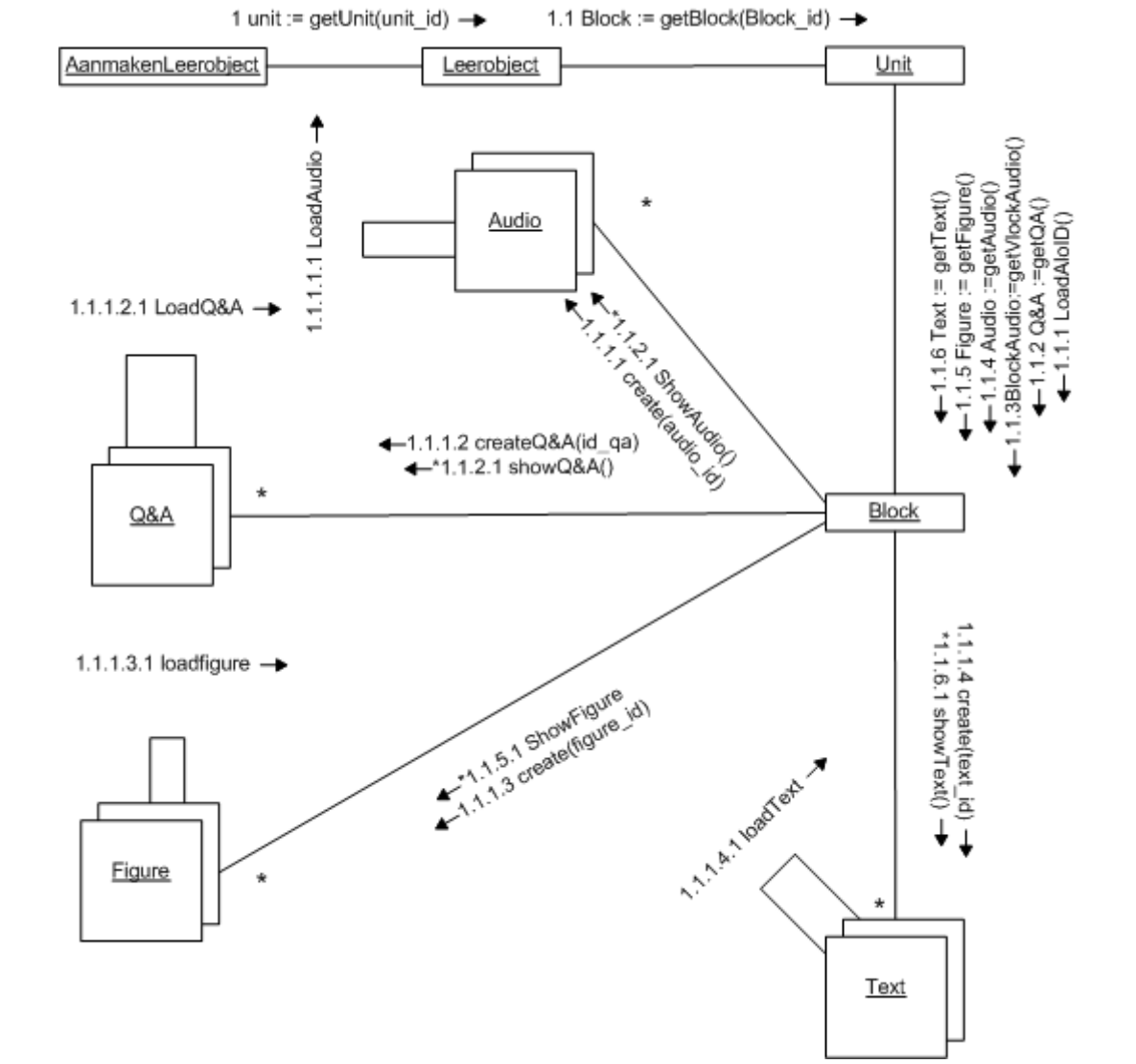
Selecteer block

"Selecteer block" is de vierde actie. Deze actie geeft de inhoud van het block weer. De inhoud wordt zonder lay-out weergegeven. Ze wordt gelinkt aan de actie "Geef leerobject". Zo kan men bij het bekijken van en leerobject de inhoud van de blocks bekijken.
Het vertrekpunt van "Selecteer block" is weer de controller klasse 'AanmakenLeerobject'. Dit object weet, dankzij het block identificatienummer en het unit identificatienummer, welk block de gebruiker wil zien. Via de functie 'getUnit' vraagt ze de 'Unit' op die het 'Block' bevat. Aan deze 'Unit' vraagt 'AanmakenLeerobject' het juiste object 'Block' op. Het gekozen 'Block' vraagt de identificatienummers van de ALO's, die aan hem gerelateerd zijn, op uit de database. Op basis hiervan gaat het object 'Block' verschillende 'ALO' objecten (Q&A, Audio, Figure, Text) aanmaken en dit volgens het principe "by creator". Eenmaal de 'ALO' objecten gecreëerd zijn, raadplegen ze de database voor het invullen van hun data. Al deze 'ALO' objecten bezitten een 'show' functie (showQ&A, showAudio, showfigure en showText). Om de actie "Selecteer block" te beëindigen gaat het object 'Block' aan elke 'ALO' vragen zijn 'show' functie uit te voeren. Hierdoor wordt aan de gebruiker de volledige inhoud van de block weergegeven.



Figuur 18. Sequentiediagram "Selecteer block"

Zoals bij alle andere acties wordt na het opmaken van het sequentiediagram - voor de volledigheid - een collaboratiediagram opgesteld.



Figuur 19. Collaboratiediagram "Selecteer block"

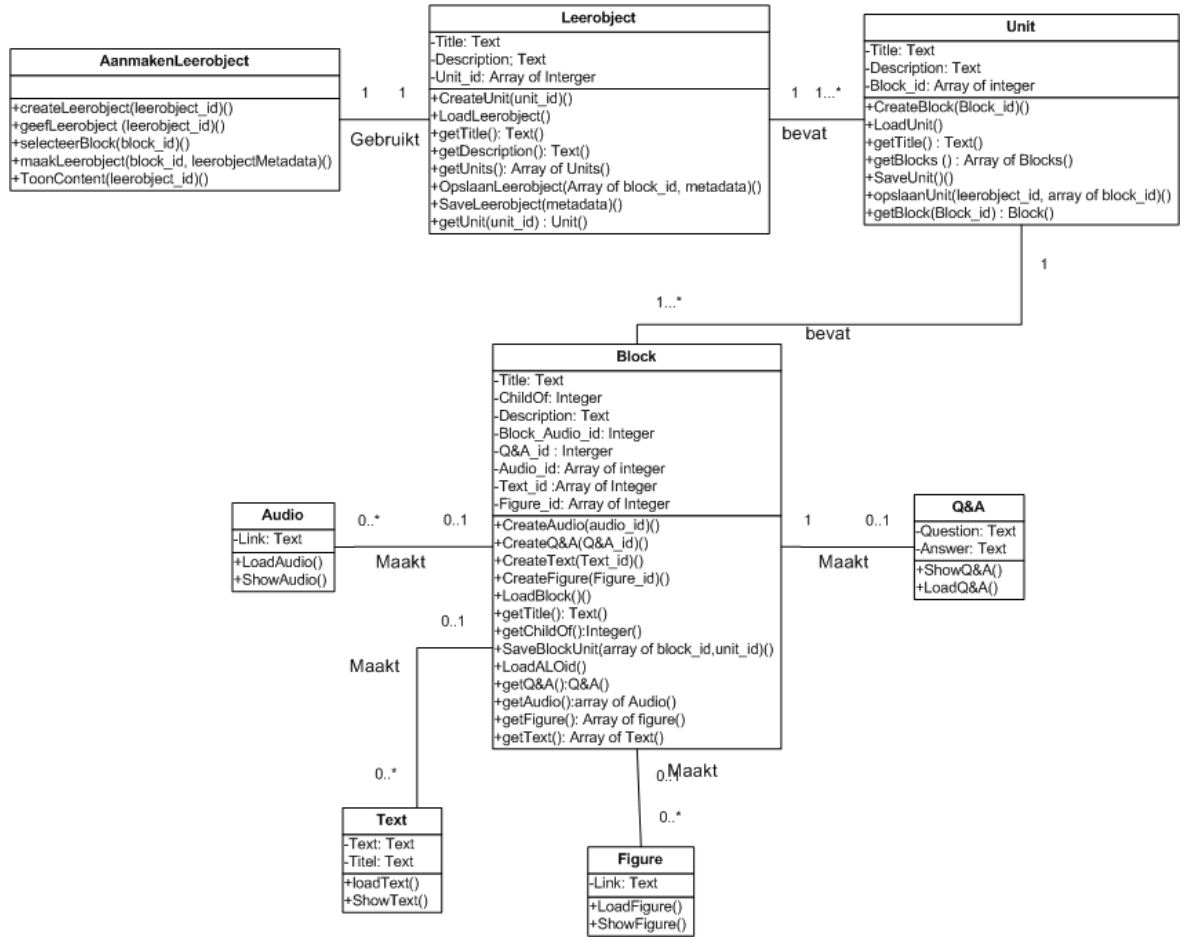
5.1.1.5 Klassendiagram

De laatste stap in het ontwerpen van de 'learning object assembling module' is het opstellen van het klassendiagram. Het klassendiagram (figuur 20) is een synthese van de zes voorgaande diagrammen.

De kaders (AanMakenLeerobject, Leerobject, Unit, Block, ALO, Audio, Figure, Q&A, Text, BlockMap en Layout) stellen de klassen van het systeem voor. Deze klassen zijn onderverdeeld in drie delen. Het bovenste kader stelt de naam van de klasse voor, het kader eronder stelt de variabelen (ook wel attributen genoemd) van de klasse voor. Het laatste kadertje stelt de methodes van de klassen met hun parameters voor. De verbindingslijnen tussen de klassen

geven hun onderlinge relaties weer. De cijfers aan de uiteinden van de verbindingslijnen stellen de multipliciteit voor.

Dit diagram vertelt de programmeur hoe hij het systeem moet programmeren. Alle klassen, hun variabelen en hun methodes zijn benoemd en toebedeeld. Toch is er wat vrijheid bij het programmeren van een systeem op basis van een UML klassendiagram. Meerdere programmeurs kunnen dus verschillende programma's schrijven die hetzelfde doen en op hetzelfde diagram gebaseerd zijn.



Figuur 20. Het klassendiagram van Generic authoring assembly system

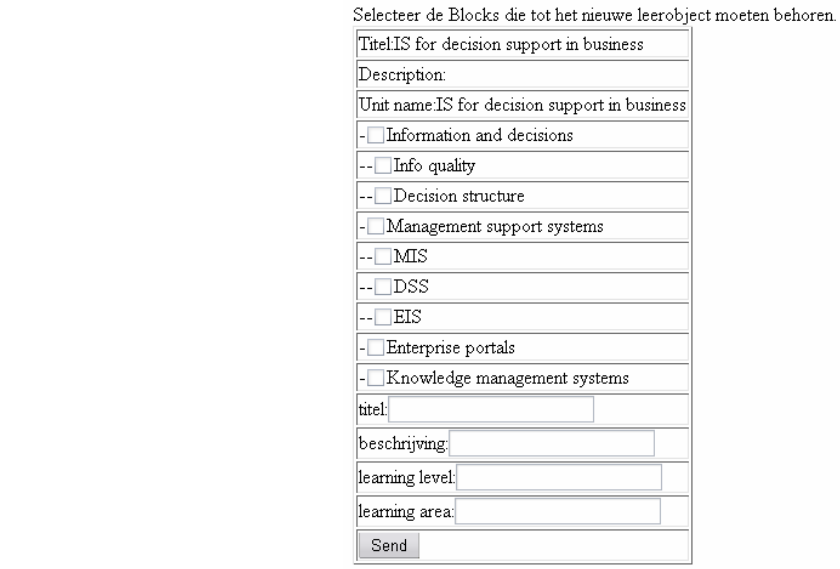
5.1.2 Het programma

Voor het schrijven van de 'learning object assembling module' was er de keuze tussen de talen PHP en ASP. Beide talen zijn ontwikkeld voor een goede samenwerking met web browsers; een noodzaak vermits dit programma zijn volledige output moet weergeven via web browsers.

PHP is het meest geschikt in deze context omwille van de 'connectiviteit' tussen de database en PHP. Die is beter dan de connectiviteit tussen de database en ASP. PHP werkt ook beter samen met de verschillende platformen. ASP werd voornamelijk ontwikkeld met het oog op een Microsoft Windows platform, terwijl PHP compatibel is met elk platform (Linux, Unix, Windows, OS X). De laatste reden om PHP te verkiezen boven ASP is de kostprijs. PHP wordt gratis aangeboden terwijl men voor ASP moet betalen. (Pires, 2005)

Alle interacties tussen de 'learning object assembling module' en de relationele database gebeuren via SQL commando's. Om de database en het programma te connecteren moeten er enkele variabelen ingevuld worden in de 'connect.php' file.

De werking van het programma is relatief eenvoudig. De gebruiker kiest een bestaand leerobject. Eenmaal de keuze gemaakt is, geeft het programma weer uit welke blocks dit leerobject bestaat. Aan de hand van checkboxen geeft de gebruiker aan welke blocks hij wil opslaan in het leerobject (figuur 21).



Figuur 21. Screenshot "keuze van de blocks"

Op basis van de afgevinkte blocks maakt het programma een lijst op van blocks die worden opgenomen in het nieuwe leerobject. Vervolgens wordt het nieuwe leerobject opgeslagen in de bestaande database. Het is dit leerobject dat later door de derde laag gebruikt zal worden bij het voorstellen van de leerinhoud in verschillende lay-outs. In bijlage 2 is de programmacode van deze module opgenomen.

We slagen erin de vooropgestelde herbruikbaarheid te behalen. Nieuwe leerobjecten worden aangemaakt op basis van bestaande leerobjecten en de nieuwe leerobjecten zijn op hun beurt ook weer herbruikbaar. Dit komt omdat ook de nieuwe leerobjecten en hun content voldoen aan de drie eisen: discoverabiliteit, modulariteit en interoperabiliteit.
Omdat we metadata toevoegen aan het nieuwe leerobject zijn ze discoveratibel. Tijdens het aanmaken en het opslaan van het nieuwe leerobject behouden we de relaties van de leercontent onderling. Het object is dus nog altijd modulair opgebouwd. Ook wordt bestaande content hergebruikt en de interoperabiliteit van de content wordt gewaarborgd.

In het prototype worden alleen op het niveau van leerobject nieuwe metadata toegevoegd. Op de andere niveaus worden de bestaande metadata behouden. Hierdoor moet de gebruiker niet enorm veel metadata ingeven bij de creatie van een nieuw leerobject, maar kan hij toch de belangrijkste verschillen met andere leerobjecten meegeven. Een nadeel hiervan is dat we redundante data in onze database bekomen. Meerdere records zullen grotendeels dezelfde data bevatten.

5.2 ONTWIKKELING VAN DE DERDE LAAG: 'LEARNING OBJECT CREATION MODULE'

De derde laag heeft als taak het leerobject, dat werd aangemaakt in de tweede laag voor te stellen aan de gebruiker. Dit moet gebeuren in een lay-out gekozen door de gebruiker. Voor deze derde laag zijn de samenwerking met laag twee en de toegankelijkheid tot de verschillende lay-outs belangrijk.

5.2.1 Systeemontwerp

Het ontwerp van de derde laag verloopt grotendeels op dezelfde manier als dat van de tweede laag. Er wordt weer een beroep gedaan op de methode van Larman voor het creëren van een klassendiagram. Voor deze laag moet maar één use case opgesteld worden, omdat een gebruiker het programma maar op één manier kan gebruiken, nl. het opvragen van een leerobject. De volgende use case werd opgesteld.

5.2.1.1 Use Case

Use case name: Lay-out leerobject
Primary actor: LOMS gebruiker/leerling

Stakeholders and their interests:

- LOMS gebruiker/leerling:
  - Hij wil het door hem gekozen leerobject in een door hem gekozen lay-out bekijken.
  - Hij wil dat dit snel laadt.
  - Hij wil de juiste lay-out zien.
  - Hij wil een onderdeel (block) van het leerobject meer gedetailleerd kunnen bekijken in de gekozen lay-out.
- Maker van de LOMS:
  - Hij wil dat zijn leerobject in een lay-out wordt weergegeven zoals hij het gemaakt heeft.
- Beheerder van het LOMS:
  - Hij wil dat het systeem eenvoudig te onderhouden is.

Preconditions:

Het leerobject bestaat in de database, de lay-out template bestaat en de gebruiker is gemachtigd om het leerobject te bekijken.

Success condition:

De gebruiker kan het leerobject bekijken met verschillende lay-outs.

Main success scenario:

1. De LOMS gebruiker/leerling selecteert een lay-out.
2. Het systeem geeft het overzicht van het leerobject weer in de gekozen lay-out.
3. De LOMS gebruiker/leerling navigeert door het leerobject door het selecteren van blocks.
4. Het systeem toont de inhoud van een geselecteerd block in de gekozen lay-out.
5. De LOMS gebruiker/leerling geeft aan terug naar het zoekmenu te willen.
6. Het systeem geeft het “zoek leerobject” scherm.

Alternative cases:

Op elk moment.

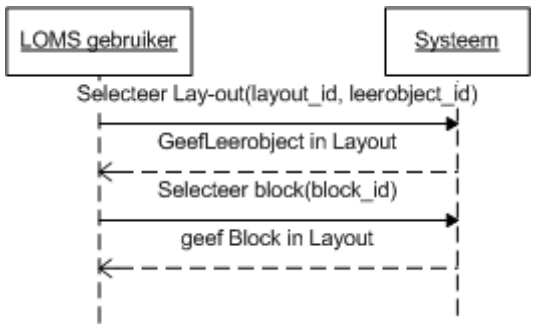
Een foutmelding treedt op. De foutmelding wordt aan de gebruiker getoond. De LOMS gebruiker/leerling wordt naar het "zoek leerobject" scherm gebracht.

6 Het geselecteerde block bestaat niet.

Het systeem geeft de boodschap: "Het gevraagde block <naam> bestaat niet." Daarna brengt het de LOMS gebruiker/leerling terug naar de indexpagina van het geselecteerde leerobject.

5.2.1.2 SSD

Aan de hand van deze korte use case wordt een SSD diagram opgesteld. De use case bevat twee acties van de gebruiker en twee antwoorden van het systeem, nl. stap één tot en met vier. Deze antwoorden en acties worden telkens in één woord omschreven in het SSD. De nodige in- en output informatie voor de respectievelijke acties en antwoorden worden opgenomen tussen haakjes.



Figuur 22. Sequence diagram voor E-course module

5.2.1.3 Domeinmodel

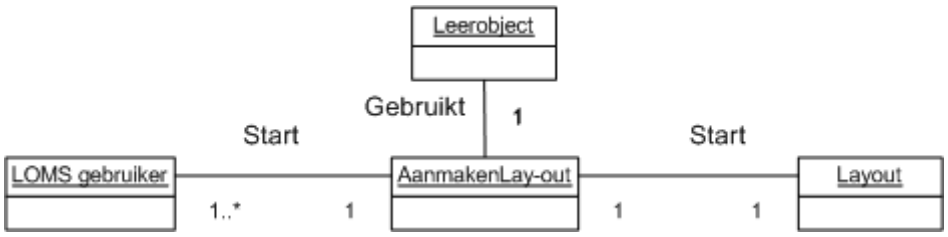
Voor het maken van het domeinmodel worden de use case en de lijst van veel voorkomende klassen geraadpleegd. Een aantal objecten, aangemaakt en uitgewerkt in de tweede laag, wordt opnieuw gebruikt. Er wordt nog een klasse 'Lay-out' toegevoegd, die wordt ingedeeld bij de reële objecten. Het proces 'AanmakenLay-out' wordt toegevoegd om de klasse 'Lay-out' te controleren.

'LOMS gebruiker' wordt als laatste klasse toegevoegd aan het domeinmodel. Deze klasse wordt toegevoegd om de user input in het systeem te simuleren.

Tabel 7
Domeinmodel

Lijst van veel voorkomende klassen		**Use case**
Reële objecten	*Processen*	-
Lay-out Leerobject	AanmakenLay-out	LOMS gebruiker

Op basis van tabel 7 wordt het domeinmodel met relaties gemaakt (figuur 23). De relaties worden weer weergegeven door multipliciteit en door een aanduiding (start, gebruikt, enz.) die aangeeft hoe de objecten met elkaar gerelateerd zijn.



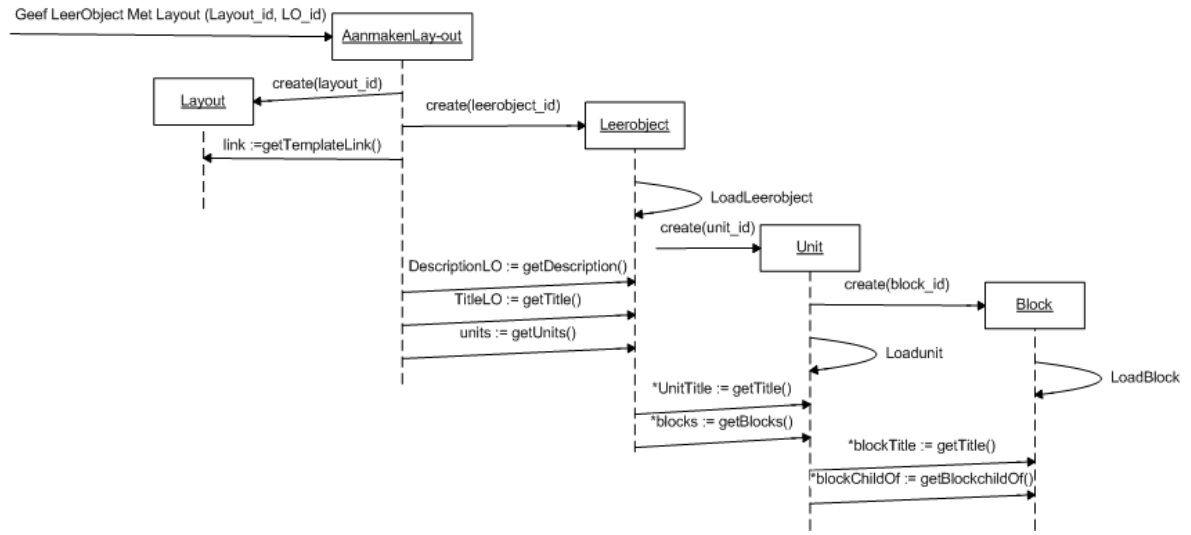
Figuur 23. Domeinmodel met relaties

5.2.1.4 Sequentie- en collaboratiediagrammen

Uit het SDD kunnen twee acties afgeleid worden: "Geef leerobject in lay-out" en "Geef block in lay-out". Deze acties worden verder uitgewerkt in sequentie- en collaboratiediagrammen.

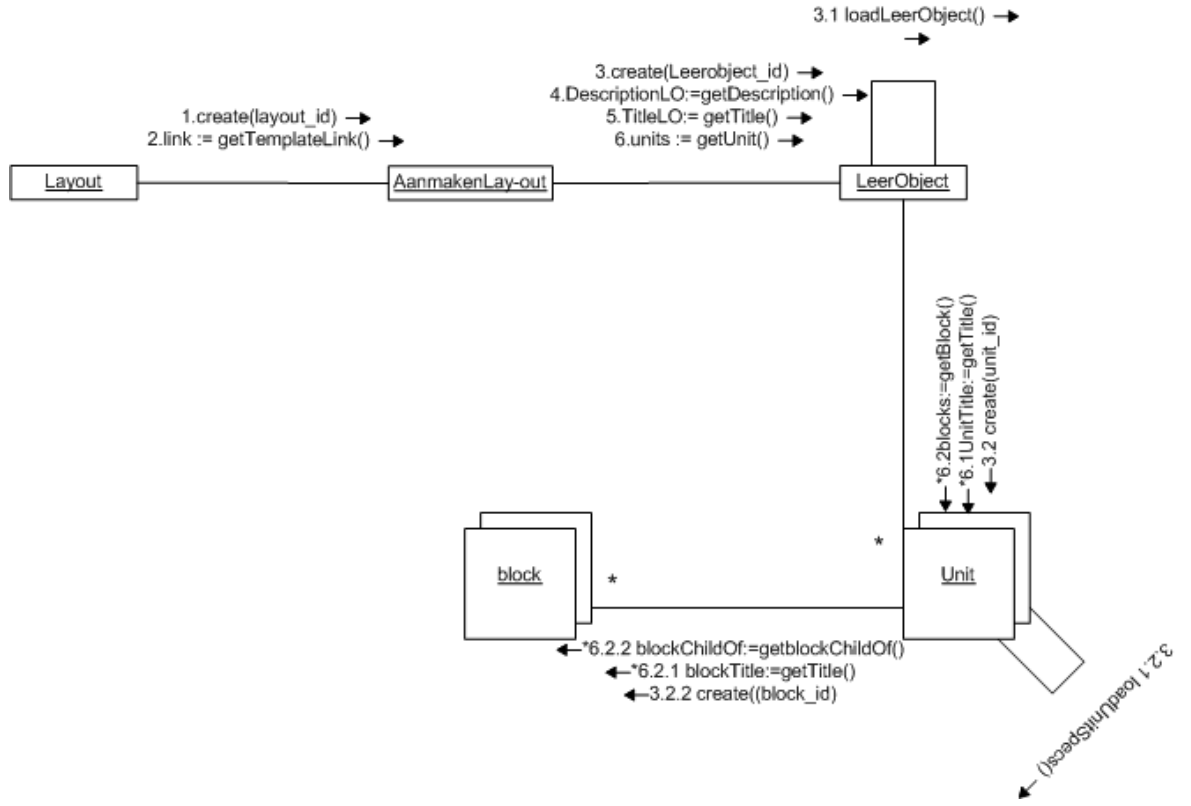
Geef leerobject in lay-out

Deze actie stelt het leerobject voor aan de gebruiker. De voorstelling bestaat uit een overzicht van de blocks, hun onderlinge relaties en de algemene informatie van het leerobject.
Aan de hand van de user input wordt een object 'AanmakenLay-out' aangemaakt. Dit object krijgt van de gebruiker een lay-out identificatienummer en een leerobject identificatienummer. Met behulp van deze twee nummers creëert het object 'AanmakenLay-out' twee nieuwe objecten, nl. 'Layout' en 'Leerobject'. Het object 'Layout' bezit de locatie van de template. Via de functie 'getTemplateLink' vraagt het object 'AanmakenLay-out' de locatie van het template op. Het tweede aangemaakte object, 'Leerobject', gebruiken we om de structuur aan te maken en de informatie uit de structuur te halen. Analoog naar "Geef leerobject" en "Toon content" wordt de informatie uit het leerobject gehaald. Het object 'AanmakenLay-out' brengt in een laatste stap de inhoud en het template samen.



Figuur 24. Sequentiediagram "Geef leerobject in lay-out"

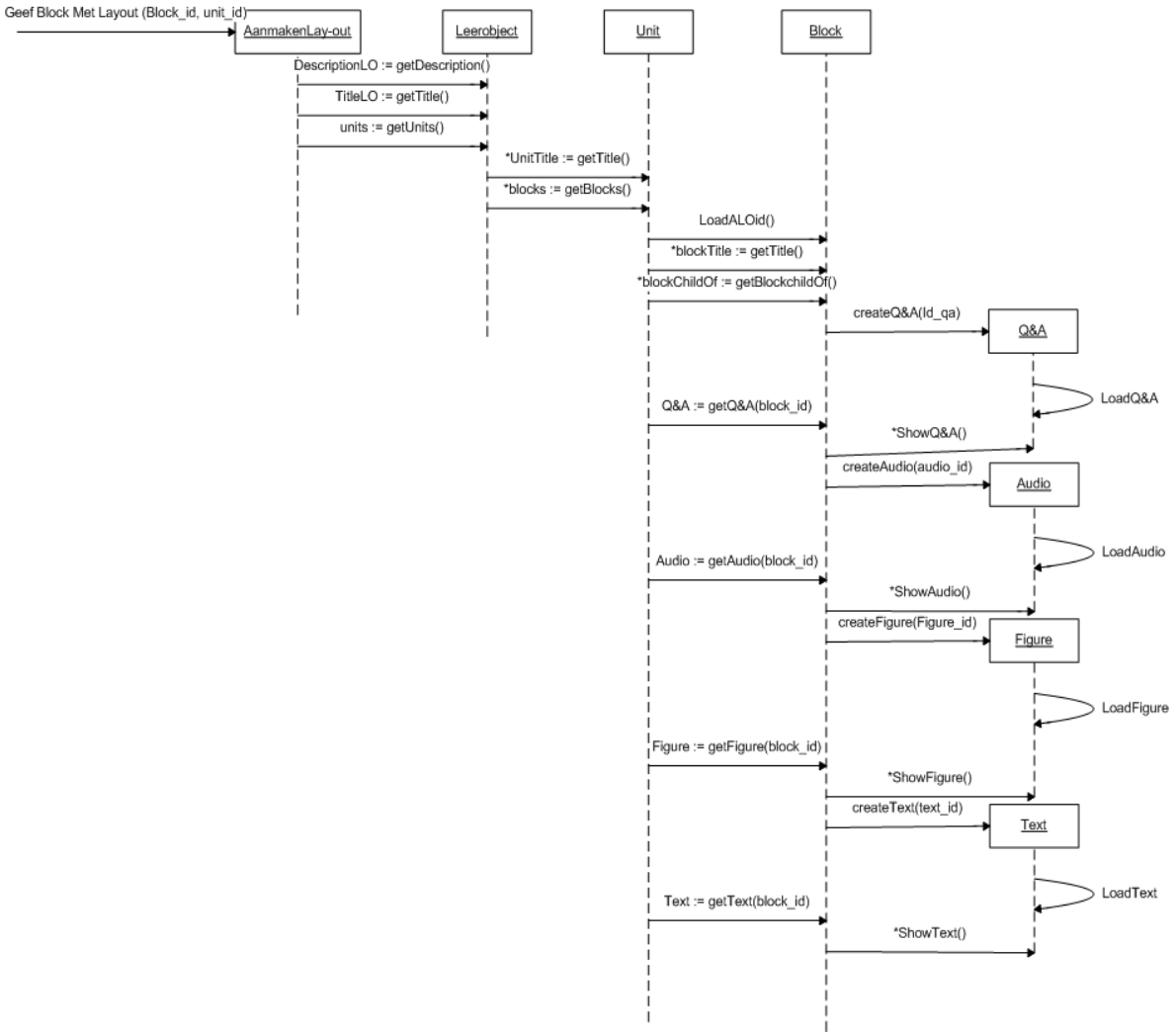
Na sequentiediagram wordt een collaboratiediagram opgesteld. Dit collaboratiediagram verduidelijkt de volgorde en de aftakkingen van de berichtgeving voor de actie "Geef leerobject in lay-out".



Figuur 25. Collaboratiediagram "Geef leerobject in lay-out"

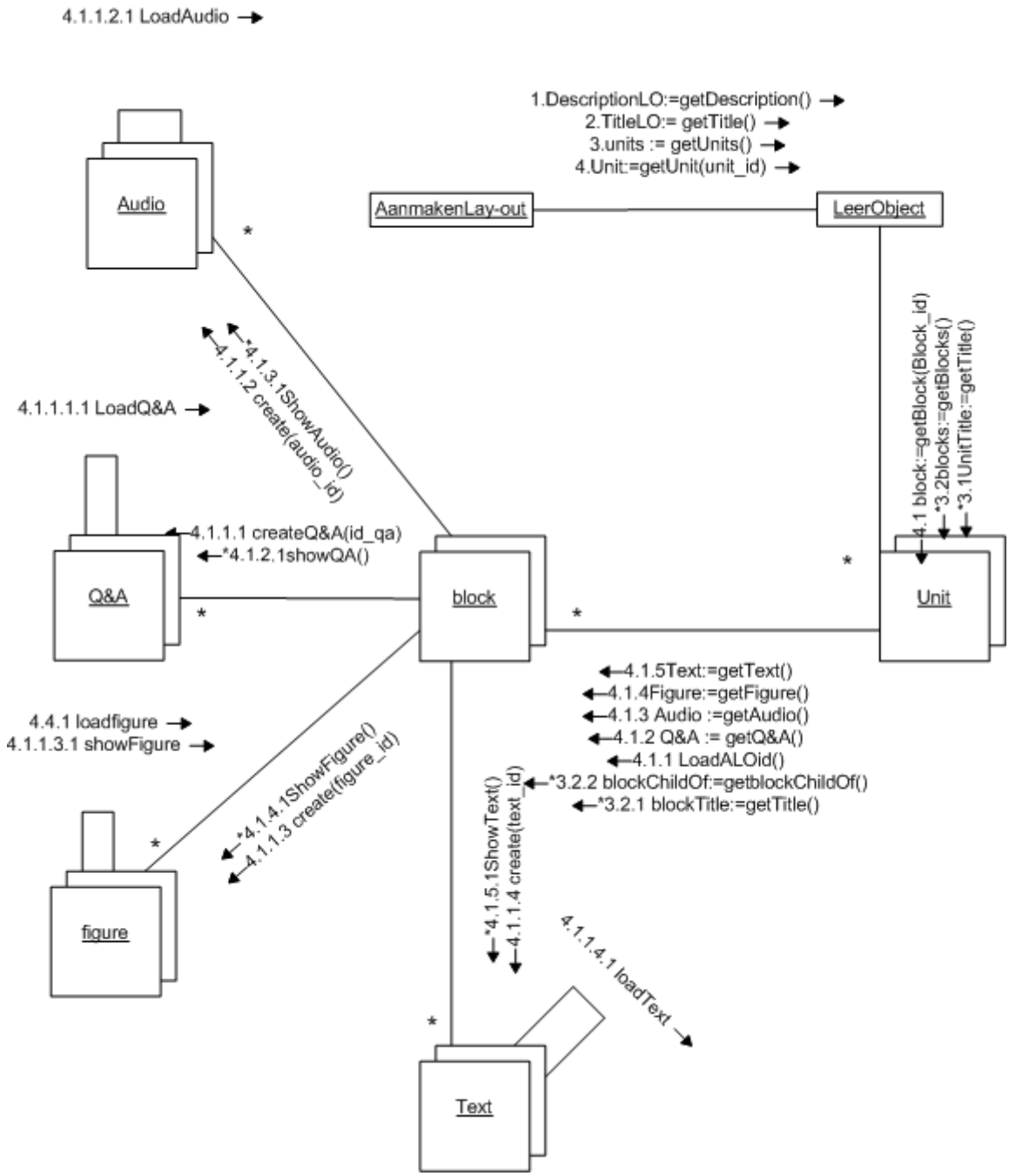
Geef block in lay-out

De actie "Geef block in lay-out" presenteert een overzicht van de blocks, hun onderlinge relaties en de inhoud van het geselecteerde block in de gekozen lay-out.
De gebruiker geeft het block identificatienummer aan het object 'AanmakenLay-out'. Dit nummer geeft aan welk block de gebruiker graag zou bekijken. Het ophalen van de informatie uit de structuur kunnen we in twee delen opdelen, namelijk het ophalen van de informatie omtrent het leerobject en het ophalen van de inhoud van het gekozen block. Het opvragen van de informatie betreffende het leerobject verloopt analoog aan "Geef leerobject" Eerste vraagt 'AanmakenLay-out' de titel en de beschrijving van het 'Leerobject' op. Daarna worden de objecten 'Unit' opgevraagd die tot het 'leerobject' behoren. Deze 'Unit' objecten worden doorlopen met behulp van een lus. Per 'Unit' wordt de titel en de objecten 'Block', die tot de unit behoren, opgevraagd. De 'Block' objecten worden dan aan de hand van een geneste lus doorlopen. Tijdens dit doorlopen wordt de titel van elk object 'Block' opgevraagd en of het al dan niet een sub-block is. Deze informatie wordt doorgegeven aan het object 'AanmakenLay-out'. Dit object gaat deze informatie samenvoegen met het gekozen template en zo een overzicht maken hoe de blocks van het leerobject gestructureerd zijn. Het opvragen van het tweede deel van informatie loopt analoog aan 'Selecteer block'.
Het object 'AanmakenLay-out' vraagt de 'Unit' waartoe het object behoord op via de functie 'getUnit'. Aan deze 'Unit' vraagt 'AanmakenLay-out' het gekozen 'Block' op. Het gekozen object leest via de functie 'loadALOId' de identificatienummers in van de ALO's die aan hem gerelateerd zijn. Daarna maakt 'Block' de nodige 'ALO' objecten aan. Deze aangemaakte objecten raadplegen de database voor het invullen van hun data. Via de functies getAudio, getText, getFigure en getQ&A kan 'Block' de inhoud van de 'ALO's in de juiste lay-out voegen.



Figuur 26. Sequentiediagram "Geef block in lay-out"

Het laatste collaboratiediagram toont ons duidelijk de volgorde van de berichten en hun aftakking.

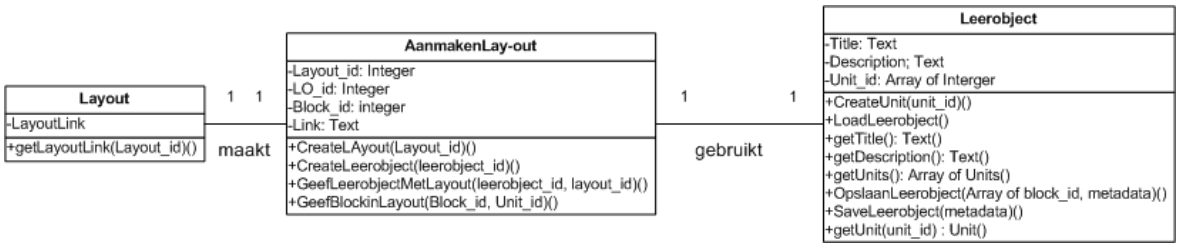


Figuur 27. Collaboratiediagram "Geef block in lay-out"

5.2.1.5 Klassendiagram

Op basis van voorgaande diagrammen wordt weer een klassendiagram samengesteld. Een aantal functies en objecten zijn hetzelfde als bij de tweede laag. We gaan deze dan ook

hergebruiken voor de derde laag. In het klassendiagram komt dit naar voor door het object 'Leerobject' op te nemen.

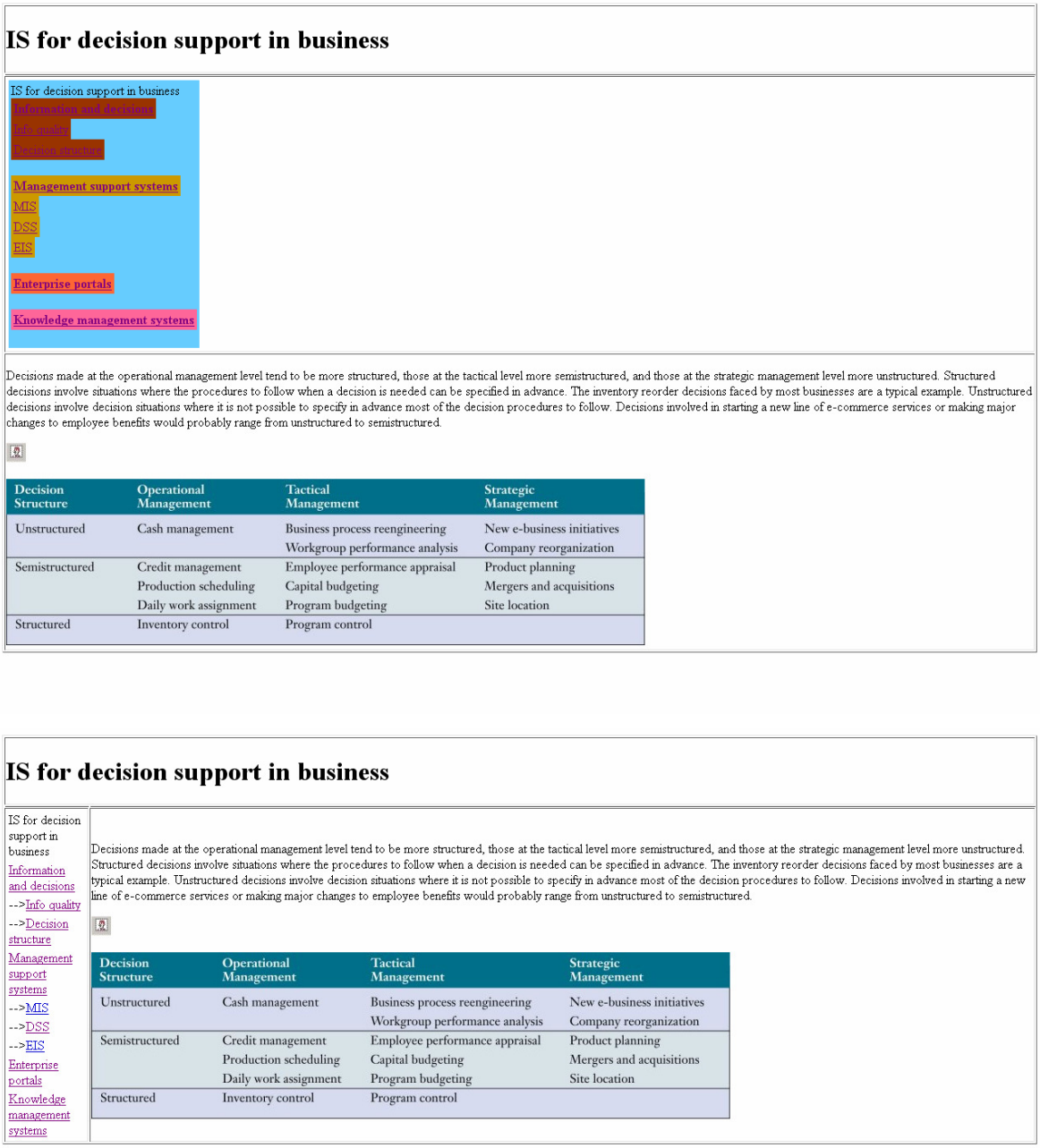


Figuur 28. Klassendiagram E-course module

5.2.2 Programma

Ook voor de learning object creation module is PHP te verkiezen als programmeertaal. Bovenop de reeds aangehaalde redenen (cf. 5.1.1 Systeemontwerp) komt als argument het belang van samenwerking tussen de tweede en derde laag.
Voor een goede werking moet het programma 'derde laag' commando's kunnen geven aan de 'tweede laag'. De tweede laag moet deze commando's begrijpen en erop kunnen reageren. Dit is belangrijk omdat de tweede laag de leerinhoud voor de derde laag levert.

Het programma toont de gebruiker eerst een keuzemenu met een overzicht van alle beschikbare lay-outs. Uit dit menu selecteert hij de lay-out die gebruikt moet worden om de leerinhoud te laten zien. Na deze selectie wordt de gebruiker naar het "zoek leerobject" scherm gebracht. Hier klikt hij op het gewenste leerobject. Nu kan het programma het gekozen leerobject weergeven in de geselecteerde lay-out. Volgende schermen tonen enkele voorbeelden van lay-outs (figuur 29). In bijlage 3 is de programmacode van deze module opgenomen.



Figuur 29. Screenshot verschillende lay-outs

Het programma geeft de mogelijkheid om bestaande leerobjecten te hergebruiken in een nieuwe lay-out en zo beter te voldoen aan de wensen en eisen van de gebruiker. Bij het uitwerken van dit programma kwamen twee moeilijkheden aan het licht. Het eerste heeft betrekking op het opslaan van de templates, het tweede heeft te maken met de communicatie tussen de tweede en derde laag.

Een moeilijkheid bij het schrijven van dit programma was de manier waarop het adres van de lay-out templates werd opgeslagen. Er zijn twee mogelijke oplossingen, nl. in de database kan een nieuwe tabel gemaakt worden of de adressen kunnen "hard-coded" opgenomen worden in het programma.
Als een nieuwe tabel aangemaakt wordt, creëren we flexibiliteit om nieuwe lay-outs op te nemen in het systeem. Men kan namelijk gemakkelijk aan een database tabel records en dus ook het adres van lay-outs toevoegen of eruit verwijderen. Het probleem is dat deze nieuwe database tabel niet past binnen het huidige database schema, er zijn geen relaties mogelijk met al bestaande tabellen.
Als de templates hard-coded worden opgenomen in het programma, hoeven we ons geen zorgen te maken over het database schema. Het nadeel is wel dat het veel moeilijker is om nieuwe templates toe te voegen of te verwijderen.
In dit ontwerp werd ervoor gekozen om de adressen van de templates hard-coded op te nemen in het programma. Vermits het hier om een prototype ging en maar enkele templates aangemaakt werden, lag deze keuze voor de hand.

De tweede moeilijkheid heeft te maken met de samenwerking tussen de tweede en de derde laag. In het theoretisch model krijgt de tweede laag de verantwoordelijkheid voor het opbouwen van het leerobject in het geheugen. De derde laag gebruikt dit opgebouwde leerobject voor de weergave. Bij het uitwerken van de derde laag werd duidelijk hoe nauw verbonden deze twee lagen zijn. De derde laag doet namelijk beroep op functies van objecten ontwikkeld in de tweede laag. Aangezien beide programma's in dezelfde programmeertaal zijn geschreven was dit bij het uitwerken van het prototype niet echt een probleem. Toch moet er rekening gehouden worden met het feit dat bij een echte implementatie van het systeem er gekozen kan worden voor verschillende talen die niet zo gemakkelijk kunnen samenwerken.

6. BESLUIT

Op het einde van dit onderzoek som ik nog enkele interessante en belangrijke vaststellingen op uit de literatuurstudie, het realiteitsonderzoek en de uitgewerkte oplossingen.

Een leerplatform gebaseerd op leerobjecten biedt voordelen op het vlak van het hergebruiken van stukjes leerinhoud. Deze herbruikbaarheid steunt op drie andere principes, nl. discoverabiliteit, modulariteit en interoperabiliteit. Discoverabiliteit kan verbeterd worden door het toevoegen van metadata. Modulariteit wordt bevorderd door de leerinhoud te structuren naar het grey box principe en interoperabiliteit wordt verbeterd door het gebruik van standaarden en open protocollen.
De Scorm standaard is op dit moment de belangrijkste standaard voor leerobjecten en leerplatformen. Het is een overkoepelende standaard die bestaande standaarden samenbrengt in één kader.
De bouwstenen van leerplatformen zijn leerobjecten. Bij het aanmaken van deze objecten is de 'compositie' uiterst belangrijk vermits dit proces toelaat om de voorstelling van het leerobject aan te passen aan de wensen en behoeften van de gebruiker.

Bij het bestuderen van het LOMS werd vastgesteld dat het bestaande systeem een oudere implementatie van de leerobjecten technologie is. Toch voldoet de het systeem nog grotendeels aan de eisen van de literatuur: de uitgebreide metadata ondersteunen de eis van discoverabiliteit, de gekozen hiërarchie ondersteunt de eis van modulariteit en door gebruik te maken van open standaarden voldoet het aan de eis van interoperabiliteit. De voorwaarden voor herbruikbaarheid zijn dus vervuld. Toch laat het bestaande systeem geen hergebruik van de content toe.

De herbruikbaarheid van het LOMS kan significant verbeterd worden, door het gebruik van een 3-lagige architectuur voor de module die de LO's aanmaakt. De eerste laag is de bestaande relationele database. De tweede laag creëert nieuwe leerobjecten op basis van bestaande leerobjecten. Deze laag leest een bestaand leerobject in en stelt het voor aan de gebruiker. De gebruiker kan het aanpassen aan zijn behoeften en opslaan. Het leerobject wordt dan als een nieuw leerobject opgeslagen in de bestaande database.
De derde laag gebruikt de leerobjecten opgeslagen in de eerste laag en gevormd in de tweede laag. Ze gaat deze objecten tonen aan de gebruiker. Tijdens deze weergave wordt de

presentatie van het leerobject aangepast aan de gebruiker (zijn leerstijl) en aan zijn situatie (de eisen van het gebruikte user device).
Een laatste verbetering ontstaat door de samenwerking tussen de eerste en tweede laag. De tweede laag geeft de mogelijkheid om op basis van simpele commando's leerobjecten op te halen, relaties te leggen en te gebruiken. Deze functionaliteit kan door toekomstige programma's gebruikt worden.

In de ontwikkelingsfase kwamen volgende punten aan het licht:

De nieuw gecreëerde leerobjecten voldoen aan de voorwaarden van herbruikbaarheid. Ze krijgen tijdens het aanmaken nieuwe en bestaande metadata mee. Ze worden opgeslagen in een modulaire structuur waarin hun relaties behouden blijven. Doordat bestaande content gebruikt wordt, blijft de interoperabiliteit gewaarborgd.
Een moeilijkheid bij het ontwikkelen van de derde laag was de plaats waar het adres van de lay-out templates opgeslagen wordt. Op dit moment zijn er twee mogelijke oplossingen. We kunnen het adres opnemen in onze code of een database tabel aanmaken waarin het adres opgeslagen wordt. Beide oplossingen hebben voor- en nadelen. Zo zal het hard-coden van de adressen het database schema logisch en intact houden, maar het zal minder gemakkelijk zijn om nieuwe templates aan het systeem toe te voegen. Een extra datatabel zal flexibiliteit geven voor het toevoegen en verwijderen van templates maar de tabel zal geen relatie hebben met andere tabellen.

Ter afsluiting nog enkele suggesties/vragen voor verder onderzoek:

In deze thesis werd de herbruikbaarheid van het LOMS verbeterd door het invoeren van een nieuwe architectuur. Deze architectuur kan uitgebreid worden om de interoperabiliteit van het systeem te verbeteren. Hiervoor zal men moeten onderzoeken hoe men een systeem kan ontwikkelen dat de relaties en de content van de leerobjecten ingelezen in de tweede laag kan omzetten naar één bestand gemaakt volgens algemeen aanvaarde standaarden.

In deze thesis werd PHP gekozen als programmeertaal voor nieuwe lagen. Doordat beide lagen in dezelfde taal geschreven zijn, kunnen ze goed met elkaar communiceren. Als men een ander systeem wil laten gebruik maken van de tweede laag of men beslist om de bestaande lagen te ontwikkelen in verschillende talen zal men een manier moeten vinden om deze twee lagen met elkaar te laten samenwerken.

Zoals vermeld is het bestaande LOMS al enkele jaren oud. Sinds zijn ontstaan zijn er nieuwe standaarden gezet. De metadata voldoen niet aan de huidige Scorm standaard. Het zou dan ook interessant om uit te zoeken welke elementen moeten toegevoegd of weggelaten worden om aan deze standaard te voldoen.

REFERENTIES

ADL (2004) *Sharable Content Object Reference Model (SCORM)® 2004 3rd Edition Version 1.0* (online) (12/4/2007). Beschikbaar op <URL:http://www.adlnet.gov/scorm/index.aspx>.

AICC (2007) (online) (15/03/2007). Beschikbaar op <URL:http://www.aicc.org>.

Alberink, M., Annokkee, G., Brussee, R., Grootveld, M., de Poot, H., Strating, Verwijs, C., Veenstra, M. en Swaak, J. (Ed.) (2001) *State-of-the-art-e-learning'*, Project Raport, Enschede, Telematica Instituut (online) (03/12/2006). Beschikbaar op <URL:https://doc.telin.nl/dscgi/ds.py/Get/File-16308/State-of-the-art_e-learning.pdf >.

Anido-Rifón, L.E., Santos-Gago, J.M., Ródríguez-Estévez, J.S., Caeiro-Rodríguez, M., Fernández-Iglesias, M.J. en Llamas-Nistal, M. (2002) 'A Step ahead in E-learning Standardization: Building Learning Systems from Reusable and Interoperable Software Components', *Paper gepresenteerd op de International World Wide Web Conference*, Honolulu, Hawaii, 7 mei – 11 mei. Beschikbaar op <URL:http://www2002.org/CDROM/alternate/136/>

Aranda, N. (s.d.) 'A Brief History of Computer Based Training', *Ezinearticles*, (online) (25/11/2006). Beschikbaar op <URL:http://ezinearticles.com/?A-Brief-History-of-Computer-Based-Training&id=287273>.

Belgif (2005) 'Interoperabiliteit' (online) (15/02/2007). Beschikbaar op <URL:http://www.belgif.be/index.php/Interoperabiliteit>.

Bolhuis, S.M. en Simons, P.R.J. (1999) *Leren en werken: opleiden en leren*, Deventer, Kluwer, p. 246. Beschikbaar op <URL:http://books.google.com/books?hl=en&lr=&id=5lI2GONlnU8C&oi=fnd&pg=PA4&dq=Bolhuis+en+Simons+(1999&ots=GNybQ7CNhK&sig=UFh7Uq6VulzudcF3kFayTao9G2U>

Büchi, M. en Weck, W. (1997) in Friesen, N. (2001) 'What are Educational Objects?', *Interactive Learning Environments*, 9:3, p. 219-230.

Commissie van de Europese gemeenschappen (2000) *E-learning – het onderwijs van morgen uitdenken* (online) (09/12/2006). Beschikbaar op <URL:http://ec.europa.eu/education/programmes/elearning/comnl.pdf>.

Dublin Core (2007) (online) (20/02/2007). Beschikbaar op <URL:http://dublincore.org/index.shtml>.

E-learning.nl (2007) 'Wat is e-learning?' (online) (03/11/2006). Beschikbaar op <URL:http://www.e-learning.nl>.

Ellis, R.K. (2005) 'E-Learning Standards Update' (online) (03/11/2007). Beschikbaar op <URL:http://www.learningcircuits.org/2005/jul2005/ellis.htm>.

Farance, F. en Schoening, J. (1998) 'PAPI Specification: Learning Technology: Public and Private Information' (online) (07/03/2007). Beschikbaar op <URL:http://edutool.com/papi/papi-500.html>.

Friesen, N. (2001) 'What are Educational Objects?', *Interactive Learning Environments,* 9:3, p. 219-230.

Harris, J. (2002) 'An Introduction to Authoring Tools' (online) (06/03/2007). Beschikbaar op <URL:http://www.learningcircuits.org/2002/mar2002/harris.html>.

Hirumi, A. (2005) 'In Search of Quality: An Analysis of e-Learning Guidelines and Specifications', *Quarterly Review of Distance Education*, 6:4, p. 309-330.

Hodgins, W. (2000) in Liber, O. (2005) 'Learning Objects: Conditions for Viability', *Journal of Computer Assisted Learning*, 21:5, p. 366-373

Hodgins, W., Tomás R.G., Brown, J., Dodds, P., Christensen, M., Miller, B., et al (2003) *Making Sense of Learning Specifications & Standards: A Decision Maker's Guide to their Adoption,* Saragota Springs NY, The MASIE Center e-Learning. Beschikbaar op <URL:http://www.masie.com/standards/s3_2nd_edition.pdf>.

IEEE (2001) (online). Beschikbaar op <URL:http://ltsc.ieee.org/wg12/s_p.html>, in Nijveld, B. en van de Ven, M.J.J.M. (2003) *Learning Objects iets voor de EUR?*, Rotterdam, OECR. Beschikbaar op <URL:http://www.eur.nl/icto/rapportages/learningobjects.pdf>.

IMS (2000) 'Defining the Internet architecture for distance learning' (online) (08/03/2007). Beschikbaar op <URL:http://www.cni.org/tfms/2000b.fall/handout/IMS-E-Learning-TWason.pdf>.

IMS (2007) 'Background' (online) (08/03/2007). Beschikbaar op <URL:http://www.imsglobal.org/background.html >.

Kim, W. en Shih, T.K. (2004) 'On Reusability and Interoperability for Distance Learning', *Journal of Object Technology*, 3:8, p. 27-34. Beschikbaar op <URL:http://www.jot.fm/issues/issue_2004_09/column3>.

Larman, C. (2002) *Applying UML and Patterns An Introduction to Object-Oriented Analysis and Design and Iterative Development*, Upper Saddle River, Prentice Hall.

Leteney, F. (2006) 'Sussing the standard: taking a look at SCORM's four books', *E-learning age*, 5:9, p.17.

Lu, E.J.-L. en Chen, Y.-H. (2006) 'Design of a delegable SCORM conformant learning management system', *Journal of Computer Assisted Learning*, 22:6, p. 423-436.

Masie, E. (1999) 'Learning trends by Elliott Masie' (online) (13/11/2006). Beschikbaar op <URL:http://trends.masie.com/archives/1999/10/index.html>.

Naish, R. (2007) 'Insearch of the holy grail', *E-learning age*, 6:2, p. 8-9.

Namahn (2002) *E-learning A research notes* (online) (25/11/2006). Beschikbaar op <URL:http://www.namahn.com/resources/documents/note-e-learning.pdf>.

Nijveld, B. en van de Ven, M.J.J.M. (2003) *Learning Objects iets voor de EUR?*, Rotterdam, OECR. Beschikbaar op <URL:http://www.eur.nl/icto/rapportages/learningobjects.pdf>.

OSS in het Onderwijs (2006) *Inleiding E-learning* (online) (15/03/2007). Beschikbaar op <URL:http://www.ossinhetonderwijs.nl/modules/CmodsDownload/upload/downloads_van_OSS_in_het_Onderwijs/Inleiding_e-learning.pdf>. Vertaling van: Naidu, S. (2003) *E-Learning: A Guidebook of Principles, Procedures and Practices*, New Delhi, Commonwealth Educational Media Center for Asia (CEMCA), and the Commonwealth of Learning.

Pires, H. (2005) 'ASP vs. PHP' (online) (//). Beschikbaar op <URL:http://www.webpronews.com/expertarticles/2005/12/22/asp-vs-php>.

Riemersma, J., Veerman A., Pennings, L. en Hoving, D. (2002) *E-learning: het vervagen van grenzen*, Onderwijsraad (online) (25/11/2006). Beschikbaar op <URL:http://www.onderwijsraad.nl/uploads/pdf/elearning_het_vervagen_van_grenzen.pdf>.

Rubens, W. (2003) 'Omzien in verwondering: De (prille) geschiedenis van e-learning', *in* Rubens, W., Tjepkema, S., Poell, R., Wagenaar, S. en Dekker, H. (red.), *E-learning: meerwaarde of meer van hetzelfde?*, HRD Thema, 4:3, Deventer, Kluwer, p.9-17. Beschikbaar op <URL:http://www.te-learning.nl/omzieninverwondering.pdf>.

Schreurs, J. en Moreau, R. (2006) 'Converting digital learning content into learning objects', *Proceedings of the International Conf. Knowledge Acquisition and management KAM2006,* Wroclaw, Poland, 18 mei - 20 mei, p. 227.

Schreurs, J. en Moreau, R. (2006b) 'Converting learning content to learning objects (LO) and atomic LO's', *Paper gepresenteerd op de IADIS Virtual Multi Conference on Computer Science and Information Systems, 15 mei - 19 mei*. Beschikbaar op <URL:http://www.iadis.org/Multi2006/Papers/15/F018_EL.pdf>.

Schreurs, J. en Moreau, R. (2006c) 'Learning objects (LO) aligning different learning styles', *Paper gepresenteerd op de ICEL 2006 The International Conference on e-learning*, University of Quebec at Montreal, Canada, 22 juni - 23 juni.

Singh, H. (2000) in Friesen, N. (2001) 'What are Educational Objects?', *Interactive Learning Environments,* 9:3, p. 219-230.

Strijker, A. (2006) *Het maken van leerobjecten binnen de Digitale Universiteit*, Digitale Universiteit 2006, (online) (18/3/2007). Beschikbaar op <URL: http://www.du.nl/digiuni//download/temp/MakenLeerobjecteDUhandboek.pdf>.

Valacich, J.S., George, J.F. en Hoffer, J.A. (2004) *Systeemanalyse en systeemontwerp*, Amsterdam, Pearson Education Benelux.

Weitl, F., Kammerl, R. en Göstl M (2004) 'Context Aware Reuse of Learning Resources', *Proceedings of ED-MEDIA 2004 World Conference on Educational Multimedia, Hypermedia & Telecommunications*, Lugano, Switzerland, 2004.

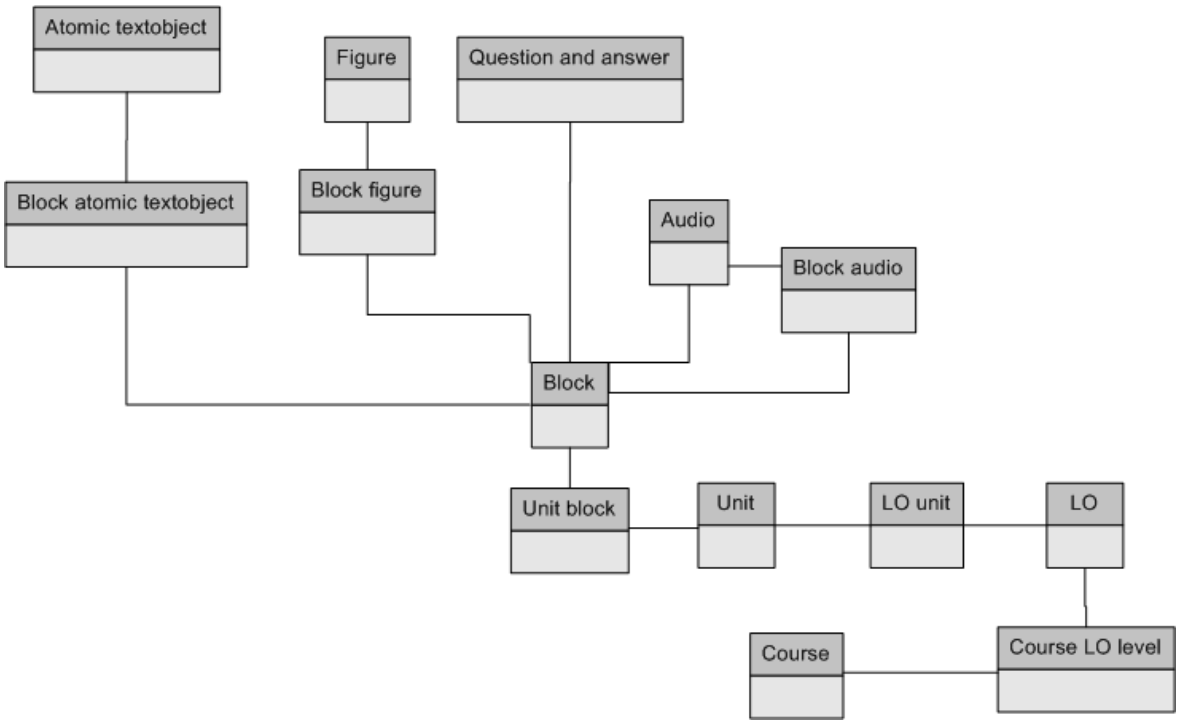
Wikipedia (2006) 'E-learning' (online) (01/11/2006). Beschikbaar op <URL:http://nl.wikipedia.org/wiki/E-learning>.

Wikipedia (2007) 'Interoperability' (online) (12/02/2007). Beschikbaar op <http://en.wikipedia.org/wiki/Interoperability>.

Wikipedia (2007b) 'Reusability' (online) (12/02/2007). Beschikbaar op <http://en.wikipedia.org/wiki/Reusability.

BIJLAGEN

Bijlage 1. Overzicht nieuwe database



Bijlage 2. Code van het generic authoring assembling systeem

```
<?PHP

    include('Classes/aamakenLeerobject.php');
    include('Classes/AanmakenLayout.php');
    $id = $_GET['id'];
    $action =$_GET['action'];
    $layout_id = $_GET['layout_id'];

    if($action == "toon")
    {
        $my_class=new aanmakenLeerobject();
        $my_class->ToonContent($id);

    }
    if($action == "view")
    {
        $my_class=new aanmakenLeerobject();
        $my_class->geefLeerobject($id);

    }
    if($action == "save")
    {
        $block = $_POST['block'];
        $object_id = $id;
        $titel= $_POST['titel'];
        $beschrijving= $_POST['beschrijving'];
        $level= $_POST['level'];
        $area= $_POST['area'];
        $my_class=new aanmakenLeerobject();
```

```
        $my_class->maakLeerobject($block, $object_id, $titel, $beschrijving, $level, $area);

    }
    if($action == "block")
    {
        $block_id = $_GET['block_id'];
        $unit_id = $_GET['unit_id'];
        $my_class=new aanmakenLeerobject();
        $my_class->selecteerblock($block_id, $unit_id, $id);

    }
    if($action == "layoutblock")
    {

        $my_layout = new AanmakenLayout();
        $block_id = $_GET['block_id'];
        $unit_id = $_GET['unit_id'];
        $my_layout->GeefBlockinLayout($id, $layout_id, $block_id, $unit_id);
    }
    if($action == "layoutview")
    {

        $my_layout = new AanmakenLayout();
        $my_layout->GeefLeerobjectMetLayout($id, $layout_id);

    }
?>
```

```
<?PHP
    include('include/pagestart.php');
    include('Classes/Figure.php');
    include('Classes/Text.php');
    include('Classes/Audio.php');
    include('Classes/Block.php');
    include('Classes/unit.php');
    include('Classes/QA.php');
    include('Classes/leerobject.php');

class aanmakenLeerobject
{

    function geefLeerobject($leerobject_id)
    {
        //vervangt de functie create leerobject
        $my_class=new Leerobject($leerobject_id);

        ?>
        <table border=1>
        <tr><td>Titel:<? echo($my_class->getTitle()); ?></tr></td>
        <tr><td>Description:<? echo($my_class->getDescription()); ?></tr></td>
        <? foreach( $my_class->getUnits() as $key => $value)
        {

            echo ("<tr><td>Unit name:" . $value->getTitle()."</tr></td>" );
            $unit = $value;

            $unit_id = $key;
            if(count($unit->getBlocks()) > 0)
```

```
            {
            foreach( $unit->getBlocks() as $key => $value)
            {

                    if($value->getChildOf($unit_id) == 0)
                    {


                        echo ("<tr><td><a href='index.php?action=block&block_id=" . $key . "&unit_id=" . $unit_id ."&id=".
                            $leerobject_id ."'>"  . $value->getTitle() . "</a></tr></td>");
                        foreach( $unit->getBlocks() as $key2 => $value2)
                        {
                            if($value2->getChildOf($unit_id) == $key)
                            {
                            echo ("<tr><td>--><a href='index.php?action=block&block_id=" . $key2 . "&unit_id=" . $unit_id ."&id=".
                                $leerobject_id ."'>" . $value2->getTitle() . "</a></tr></td>");

                            }
                        }
                    }
            }
            }
        }
         ?>
        </table>
        <?
    }
    function selecteerblock($block_id, $unit_id, $leerobject_id)
    {
        $my_class=new Leerobject($leerobject_id);
```

```
        $unit = $my_class->getUnit($unit_id);
        $block = $unit->getBlock($block_id);
        $block->LoadALOid();

        $block->ShowAll();



    }
    function ToonContent($leerobject_id)
    {
        //vervangt de functie create leerobject
        $my_class=new Leerobject($leerobject_id);

        ?>
        Selecteer de Blocks die tot het nieuwe leerobject moeten behoren.
        <form name="select_blocks" action="index.php?id=<? echo($leerobject_id); ?>&action=save" method="POST">

        <table border=1>
        <tr><td>Titel:<? echo($my_class->getTitle()); ?></tr></td>
        <tr><td>Description:<? echo($my_class->getDescription()); ?></tr></td>
        <? foreach( $my_class->getUnits() as $key => $value)
        {

            echo ("<tr><td>Unit name:" . $value->getTitle()."</tr></td>" );
            $unit = $value;

            $unit_id = $key;
            if(count($unit->getBlocks()) > 0)
            {
            foreach( $unit->getBlocks() as $key => $value)
```

```
        {

                if($value->getChildOf($unit_id) == 0)
                {


                    echo ("<tr><td>-<input type='checkbox' name='block[]' value=" . $key .">" . $value->getTitle() . "</tr></
                        td>");
                    foreach( $unit->getBlocks() as $key2 => $value2)
                    {
                        if($value2->getChildOf($unit_id) == $key)
                        {
                        echo ("<tr><td>--<input type='checkbox' name='block[]' value=" . $key2 .">" . $value2->getTitle() . "<
                            /tr></td>");

                        }
                    }
                }
        }
        }



}
    ?><tr><td>titel:<input type="text" name="titel" size="25" value=""></tr></td><?
    ?><tr><td>beschrijving:<input type="text" name="beschrijving" size="25" value=""></tr></td><?
    ?><tr><td>learning level:<input type="text" name="level" size="25" value=""></tr></td><?
    ?><tr><td>learning area:<input type="text" name="area" size="25" value=""></tr></td><?
 ?>
<tr><td><input type="submit" value="Send"></tr></td>
```

```
        </table>
        </form>
        <br>
        <?
    }
    function maakLeerobject($block, $object_id, $titel, $beschrijving, $level, $area)
    {
            $my_class=new Leerobject($object_id);
            $my_class->opslaanLeerobject($block ,$titel, $beschrijving, $level, $area, $object_id);

    }

}
?>
```

```
<?PHP

class Leerobject
{

    var $title;
    var $description;
    var $array_unit;

    function Leerobject ($leerobject_id)
    {

            $this->LoadLeerobject ($leerobject_id);
    }

    function LoadLeerobject ($leerobject_id)
    {
                //Laad de gegevens van het leerobject in.
                $this->kolom = 'Title';
                $this->tabel = 'mindmap';
                $this->id_type = 'Id_mindmap';
                $this->title= getOneResult($leerobject_id, $this->kolom ,$this->tabel, $this->id_type);
                $this->kolom = 'Description';
                $this->tabel = 'mindmap';
                $this->id_type = 'Id_mindmap';
                $this->description= getOneResult($leerobject_id, $this->kolom ,$this->tabel, $this->id_type);
                //maak array van units
                $kolom = 'Id_unit';
                $tabel = 'mindmap_unit';
                $id_type = 'Id_mindmap';
```

```
            $result = getArrayResult($leerobject_id, $kolom ,$tabel, $id_type);

            if(mysql_num_rows($result) > 0){
                while ($row = mysql_fetch_assoc($result)) {

                    $naam = $row[$kolom];
                    //dit is de create unit functie
                    $this->array_unit[$naam] = new Unit($row[$kolom]);
                }
            }
    }
    function getDescription ()
    {
        return $this->description;
    }
    function getTitle ()
    {
        return $this->title;
    }
    function getUnits()
    {
        return  $this->array_unit;
    }
    function opslaanLeerobject($block ,$titel, $beschrijving, $level, $area, $object_id)
    {

        $new_leerobjectID = $this->saveLeerobject($titel, $beschrijving, $level, $area, $object_id);
        foreach( $this->array_unit as $key => $value)
                {
                    $value->OpslaanUnits($block, $new_leerobjectID, $key);
```

```
            }
            echo '<a href=index.php?action=view&id='.$new_leerobjectID.'>klik voor de nieuwe mindmap</a>';
    }
    function saveLeerobject($titel, $beschrijving, $level, $area, $object_id)
    {

        $new_leerobjectID;

            $sql = "select * from mindmap where Id_mindmap = ". $object_id . ";";

                $result = mysql_query($sql);

                if (!$result) {
                    $message  = 'Invalid query: ' . mysql_error() . "\n";
                    $message .= 'Whole query: ' . $query;
                    die($message);
                    }

                while ($row = mysql_fetch_assoc($result)) {
                        $sql ="insert into mindmap (Title, Description, Creation_date, Last_adjustment, Format, Copyright,
                            id_Topic, Learning_level,   Learning_area, Language)  VALUES ('" . $titel. "' ,'" . $beschrijving . "'
                            ,'" . $row['Creation_date'] . "','" .$row['Last_adjustment'] . "','" .$row['Format'] . "','" .$row['
                            Copyright'] . "','".$row['id_Topic'] . "','" . $level . "','" . $row['Learning_area'] . "','". $area .
                            "')";

                $result = mysql_query($sql);

                $sql = "SELECT max( Id_mindmap ) AS blub FROM mindmap";
                $result = mysql_query($sql);
                    while ($row = mysql_fetch_assoc($result))
```

```
                    { $new_leerobjectID = $row['blub'];}

                }

        return $new_leerobjectID;

    }
    function getUnit($unit_id)
    {
    return  $this->array_unit[$unit_id];
    }
}

?>
```

```
<?PHP

class Unit {
    var $title;
    var $description;
    var $Block_Array;
    var $unit_id;
    function Unit ($unit_id)
    {
        $this->LoadUnit($unit_id);
    }
    function getBlocks()
    {
        return  $this->Block_Array;
    }
        function getBlock($block_id)
    {
        return  $this->Block_Array[$block_id];
    }
    function getTitle()
    {
        return($this->title);
    }
    function LoadUnit($unit_id)
    {
            $this->unit_id = $unit_id;
            $this->kolom = 'Title';
            $this->tabel = 'unit';
            $this->id_type = 'Id_unit';
            $this->title= getOneResult($unit_id, $this->kolom ,$this->tabel, $this->id_type);
```

```
            $this->kolom = 'Description';
            $this->tabel = 'unit';
            $this->id_type = 'Id_unit';

            $this->description= getOneResult($unit_id, $this->kolom ,$this->tabel, $this->id_type);

                $kolom = 'Id_block';
                $tabel = 'unit_block';
                $id_type = 'Id_unit';
                $result = getArrayResult($unit_id, $kolom ,$tabel, $id_type);

                if(mysql_num_rows($result) > 0){
                    while ($row = mysql_fetch_assoc($result)) {
                        $naam = $row[$kolom];
                        $this->Block_Array[$naam] = new Block($naam);

                    }
                }



    }
    function OpslaanUnits($block, $newObject_id, $unit_array_id)
    {


        $new_unitID = $this->saveUnits($blocks, $newObject_id, $unit_array_id);
        foreach( $this->Block_Array as $key => $value)
        {
```

```
            if($value->getChildOf($this->unit_id) != 0)
            {
                if(in_array($value->getChildOf($this->unit_id), $block))
                {
                    if(in_array($key, $block))
                    {
                    $value->SaveBlockUnit($this->unit_id, $new_unitID);
                    }
                }

            }
            else
            {
                    if(in_array($key, $block))
                    {
                    $value->SaveBlockUnit($this->unit_id, $new_unitID);
                    }
            }
        }

    }
    function saveUnit($blocks, $newObject_id, $unit_array_id)
    {
        $sql_unit = "select * from unit where Id_unit = ".  $unit_array_id;
                    $result_unit = mysql_query($sql_unit);
                    if (!$result_unit) {
                    $message_unit  = 'Invalid query: ' . mysql_error() . "\n";
                    $message_unit .= 'Whole query: ' . $query;
                    die($message_unit);
                    }
```

```
while ($row_unit = mysql_fetch_assoc($result_unit))
{
   $sql ="insert into unit (Title, Learning_level,  Learning_area,  Description, Creation_date,
       Last_adjustment, Elapsed_time, Language, Format, Copyright)".
    "VALUES ('" . $row_unit['Title']. "' ,'" . $row_unit['Learning_level'] . "','" . $row_unit['Learning_area'
        ] . "','" .$row_unit['Description'] .
     "','" .$row_unit['Creation_date'] . "','" .$row_unit['Last_adjustment'] . "','".$row_unit['Elapsed_time']
         . "','" . $row_unit['Language'] . "','" . $row_unit['Format'] . "','". $row_unit['Copyright'] . "')";


    $result_unit2 = mysql_query($sql);
    if (!$result_unit2) {
    $message_unit2  = 'Invalid query: ' . mysql_error() . "\n";
    $message_unit2 .= 'Whole query: ' . $query;
    die($message_unit2);
    }

    $sql_max = "SELECT max( Id_unit ) AS blub2 FROM unit";
    $result_max = mysql_query($sql_max);
    while ($row_max = mysql_fetch_assoc($result_max))
    { $new_unitID = $row_max['blub2'];}

}
//kopeltabel invullen
$sql_unit_mm ="insert into mindmap_unit (Id_mindmap, Id_unit, Position, Sequence_unit) Values ('" .
    $newObject_id. "','" . $new_unitID . "', '' ,'0')";
$result_unit_mm = mysql_query($sql_unit_mm);
if (!$result_unit_mm) {
$message_unit_mm  = 'Invalid query: ' . mysql_error() . "\n";
```

```
                    $message_unit_mm .= 'Whole query: ' . $sql_unit_mm;
                    die($message_unit_mm);
                    }
                    return $new_unitID;

    }
}
?>
```

```
<?PHP
class Block
{
    var $title;
    var $childof;
    var $description;
    var $Block_audio_id;
    var $qa_id;
    var $audio_id;
    var $Text_id;
    var $Figure_id;
    var $Block_id;

    function Block($Block_id)
            {
                  $this->loadBlock($Block_id);
            }
    function loadBlock($Block_id)
    {
                $this->Block_id = $Block_id;

                $this->kolom = 'Title';
                $this->tabel = 'block';
                $this->id_type = 'Id_block';
                $this->title = getOneResult($Block_id, $this->kolom ,$this->tabel, $this->id_type);



    }
    function getTitle()
    {
```

```
        return($this->title);
    }
    function getChildOf($unit_id)
    {
    $this->kolom = 'childof_id_block';
    $this->tabel = 'unit_block';
    $id_type1 = 'Id_block';
    $id_type2 = 'Id_unit';
    $subblock = getOneResultTwoVar($this->Block_id, $unit_id, $this->kolom ,$this->tabel, $id_type1, $id_type2);
    return $subblock;
    }
    function SaveBlockUnit($unit_id, $new_unitID)
    {
            $sql = "select * from unit_block where Id_unit = ". $unit_id .  " and Id_block = " .$this->Block_id;
            echo($sql);
            $result = mysql_query($sql);
            if (!$result) {
            $result  = 'Invalid query: ' . mysql_error() . "\n";
            $result .= 'Whole query: ' . $sql;
            die($result);
            }
            while ($row = mysql_fetch_assoc($result))
            {
            $sql_block ="insert into unit_block (Id_unit, Id_block, Sequence_block, Position, childof_id_block)  VALUES ('" .
                $new_unitID . "' ,'" . $this->Block_id . "','" . $row_unit_block_new['Sequence_block'] . "','','" .$this->
                getChildOf($unit_id) . "')";


            $result_block = mysql_query($sql_block);
```

```
        }
    }
    function LoadALOid ()
    {


            $this->kolom = 'id_audio';
            $this->tabel = 'block';
            $this->id_type = 'Id_block';
            $id_audio = getOneResult($this->Block_id, $this->kolom ,$this->tabel, $this->id_type);


            if($id_audio != 0)
            {

                $this->CreateAudio($id_audio);

            }
            $this->kolom = 'id_qa';
            $this->tabel = 'block';
            $this->id_type = 'Id_block';
            $id_qa = getOneResult($this->Block_id, $this->kolom ,$this->tabel, $this->id_type);
            if($id_qa != null)
            {

                $this->CreateQA($id_qa);

            }
```

```
            $this->CreateAloText($this->Block_id);
            $this->CreateAloFigure($this->Block_id);
            $this->CreateAloAudio($this->Block_id);

    }
    function CreateQA ($id_qa)
    {

        $this->QA = new QA ($id_qa);

    }
    function CreateAudio ($id_audio)
    {

        $this->audio = new Audio ($id_audio);

    }
    function CreateAloText($Block_id)
    {
            $kolom = 'Id_atomic_textobjects';
            $tabel = 'block_atomic_textobjects';
            $id_type = 'Id_block';
            $result = getArrayResult($Block_id, $kolom ,$tabel, $id_type);

            if(mysql_num_rows($result) > 0){
                while ($row = mysql_fetch_assoc($result)) {
                    //echo ($row[$kolom]);
                    $naam = $row[$kolom];
                    $this->Alo_text[$naam] = new Text($row[$kolom]);
                }
```

```
            }

    }
    function CreateAloFigure($Block_id)
    {
            $kolom = 'Id_figure';
            $tabel = 'block_figure';
            $id_type = 'Id_block';
            $result = getArrayResult($Block_id, $kolom ,$tabel, $id_type);

            if(mysql_num_rows($result) > 0){
                while ($row = mysql_fetch_assoc($result)) {

                    //echo ($row[$kolom]);
                    $naam = $row[$kolom];
                    $this->Alo_figure[$naam] = new Figure($row[$kolom]);
                }
            }

    }
    function CreateAloAudio($Block_id)
    {
            $kolom = 'id_audio';
            $tabel = 'block_audio';
            $id_type = 'Id_block';
            $result = getArrayResult($Block_id, $kolom ,$tabel, $id_type);

            if(mysql_num_rows($result) > 0){
                while ($row = mysql_fetch_assoc($result)) {
```

```
                    //echo ($row[$kolom]);
                    $naam = $row[$kolom];
                    $this->Alo_audio[$naam] = new Audio($row[$kolom]);
                }
            }

    }
    function ShowAll()
    {
        echo "dit block bevat" .count($this->Alo_text). " Text objecten , " .count($this->Alo_figure). " Figuur objecten , " .
            count($this->Alo_audio). " Audio objecten.";
        echo "<HR>";
        if(count($this->Alo_text) > 0)
            {
                foreach( $this->Alo_text as $key => $value)
                {
                    echo "<br> Alo_text " . $key . "<br>";
                    $value->showText();
                }
            }
            echo "<HR>";
        if(count($this->Alo_figure) > 0)
            {
                foreach( $this->Alo_figure as $key => $value)
                    {
                        echo  "<br> Alo_figure " . $key . "<br>";
                        $value->showFigure();
                    }
            }
            echo "<HR>";
```

```
        if(count($this->Alo_audio) > 0)
            {
                foreach( $this->Alo_audio as $key => $value)
                {
                    echo "<br> Alo_sound " . $key . "<br>";
                    $value->showAudio();
                }
            }
    }

}

?>
```

```
<?PHP

Class Audio {
            var $kolom;
            var $tabel;
            var $id_type;
            var $link;
        function Audio($Audio_id)
        {
             $this->kolom = 'Audio';
             $this->tabel = 'audio';
             $this->id_type = 'Id_audio';
             $this->link = getOneResult($Audio_id, $this->kolom ,$this->tabel, $this->id_type);

        }
        function showAudio()
        {

            echo("<EMBED src='" . $this->link . "' autostart=false hidden=false>");

        }
}

?>
```

```
<?PHP

class Figure {
            var $kolom;
            var $tabel;
            var $id_type;
            var $link;

        function Figure($Figure_id)
        {
            $this->kolom = 'Figure';
            $this->tabel = 'figure';
            $this->id_type = 'Id_figure';
            $this->link = getOneResult($Figure_id, $this->kolom ,$this->tabel, $this->id_type);

        }
        function showFigure()
        {

            echo("<img src='" . $this->link . "' alt='Cannot show'/>");

        }
}
?>
```

```php
<?PHP

Class QA {
            var $question;
            var $answer;
            var $kolom;
            var $tabel;
            var $id_type;
        function QA($qa_id)
        {

            $this->kolom = 'vraag';
            $this->tabel = 'qa';
            $this->id_type = 'id_qa';
            $this->question= getOneResult($qa_id, $this->kolom ,$this->tabel, $this->id_type);
            $this->kolom = 'antwoord';
            $this->tabel = 'qa';
            $this->id_type = 'id_qa';
            $this->answer= getOneResult($qa_id, $this->kolom ,$this->tabel, $this->id_type);
        }
        function showQA()
        {
            echo($this->question);
            echo ("<br>");
            echo($this->answer);
        }
}

?>
```

```
<?PHP

Class Text {


            var $text;
            var $kolom;
            var $tabel;
            var $id_type;
            var $type;
        function Text($Text_id)
        {


            $this->kolom = 'Text';
            $this->tabel = 'atomic_textobjects';
            $this->id_type = 'Id_atomic_textobjects';
            $this->text= getOneResult($Text_id, $this->kolom ,$this->tabel, $this->id_type);
            $this->kolom = 'Id_type';
            $this->tabel = 'atomic_textobjects';
            $this->id_type = 'Id_atomic_textobjects';
            $this->type= getOneResult($Text_id, $this->kolom ,$this->tabel, $this->id_type);


        }
        function showText()
        {
            if($this->type == 2)
            {
            echo("<a href=".$this->text.">click here to read the full text</a>");
```

```
            }
            else
            {
            echo($this->text);
            }
        }

}

?>
```

```
<?PHP
function getOneResultTwoVar($id1, $id2, $kolom ,$tabel, $id_type1, $id_type2)
{
$sql = "select " . $kolom . " from " . $tabel . " where " . $id_type1 . " = " . $id1 . " and " . $id_type2 . " = " . $id2 .";";

    $result = mysql_query($sql);

        if (!$result) {
        $message  = 'Invalid query: ' . mysql_error() . "\n";
        $message .= 'Whole query: ' . $query;
        die($message);
        }

        while ($row = mysql_fetch_assoc($result)) {
        $dbresult = $row[$kolom];
        }
        return $dbresult;


};

function getOneResult ($id, $kolom ,$tabel, $id_type) {

    $sql = "select " . $kolom . " from " . $tabel . " where " . $id_type . " = " . $id . ";";

    $result = mysql_query($sql);

        if (!$result) {
        $message  = 'Invalid query: ' . mysql_error() . "\n";
        $message .= 'Whole query: ' . $query;
```

```
        die($message);
        }

        while ($row = mysql_fetch_assoc($result)) {
        $dbresult = $row[$kolom];
        }
        return $dbresult;

};
function getArrayResult ($id, $kolom ,$tabel, $id_type) {

    $sql = "select " . $kolom . " from " . $tabel . " where " . $id_type . " = " . $id . ";";

    $result = mysql_query($sql);

        if (!$result) {
        $message  = 'Invalid query: ' . mysql_error() . "\n";
        $message .= 'Whole query: ' . $query;
        die($message);
        }
        return $result;

};
?>
```

```
<?php
$dbhost = 'localhost';
$dbuser = 'root';
$dbpass = '';
$dbname = 'eschema';

$conn = mysql_connect($dbhost, $dbuser, $dbpass) or die
                    ('Error connecting to mysql');

mysql_select_db($dbname);
?>
```

```
<?PHP
include ('connect.php');
include ('function.php');
?>
```

Bijlage 3. Code van de Learning object creation module

```
<?

    include('Classes/Layout.php');

class AanmakenLayout
{
    function AanmakenLayout ()
    { }
    function GeefLeerobjectMetLayout($leerobject_id, $layout_id)
    {
        //vervangt de functie create leerobject
        $soort = "leerobject";
        $my_class=new Leerobject($leerobject_id);
        $my_layout = new Layout();

        include($my_layout->getTemplateLink($layout_id));
    }
    function GeefBlockinLayout($leerobject_id, $layout_id, $block_id, $unit_id)
    {
        //vervangt de functie create leerobject
        $soort = "block";
        $my_class=new Leerobject($leerobject_id);
        $my_layout = new Layout();
        $unit = $my_class->getUnit($unit_id);
        $block = $unit->getBlock($block_id);
        $block->LoadALOid();
        include($my_layout->getTemplateLink($layout_id));
    }
}
?>
```

```
<?

class Layout
{
    var $template;
    function Layout ()
    {
        //ophalen layouts, we simuleren het in een array
        $this->template[0] = "templates/tree.html";
        $this->template[1] = "templates/mm.html";

    }
    function getTemplateLink($layout_id)
    {
        return $this->template[$layout_id];
    }

}

?>
```

```
<?PHP

class Leerobject
{

    var $title;
    var $description;
    var $array_unit;

    function Leerobject ($leerobject_id)
    {

            $this->LoadLeerobject ($leerobject_id);
    }

    function LoadLeerobject ($leerobject_id)
    {
                //Laad de gegevens van het leerobject in.
                $this->kolom = 'Title';
                $this->tabel = 'mindmap';
                $this->id_type = 'Id_mindmap';
                $this->title= getOneResult($leerobject_id, $this->kolom ,$this->tabel, $this->id_type);
                $this->kolom = 'Description';
                $this->tabel = 'mindmap';
                $this->id_type = 'Id_mindmap';
                $this->description= getOneResult($leerobject_id, $this->kolom ,$this->tabel, $this->id_type);
                //maak array van units
                $kolom = 'Id_unit';
                $tabel = 'mindmap_unit';
                $id_type = 'Id_mindmap';
```

```
            $result = getArrayResult($leerobject_id, $kolom ,$tabel, $id_type);

            if(mysql_num_rows($result) > 0){
                while ($row = mysql_fetch_assoc($result)) {

                    $naam = $row[$kolom];
                    //dit is de create unit functie
                    $this->array_unit[$naam] = new Unit($row[$kolom]);
                }
            }
    }
    function getDescription ()
    {
        return $this->description;
    }
    function getTitle ()
    {
        return $this->title;
    }
    function getUnits()
    {
        return  $this->array_unit;
    }
    function opslaanLeerobject($block ,$titel, $beschrijving, $level, $area, $object_id)
    {

        $new_leerobjectID = $this->saveLeerobject($titel, $beschrijving, $level, $area, $object_id);
        foreach( $this->array_unit as $key => $value)
                {
                    $value->OpslaanUnits($block, $new_leerobjectID, $key);
```

```
            }
            echo '<a href=index.php?action=view&id='.$new_leerobjectID.'>klik voor de nieuwe mindmap</a>';
    }
    function saveLeerobject($titel, $beschrijving, $level, $area, $object_id)
    {

        $new_leerobjectID;

            $sql = "select * from mindmap where Id_mindmap = ". $object_id . ";";

                $result = mysql_query($sql);

                if (!$result) {
                    $message  = 'Invalid query: ' . mysql_error() . "\n";
                    $message .= 'Whole query: ' . $query;
                    die($message);
                    }

                while ($row = mysql_fetch_assoc($result)) {
                        $sql ="insert into mindmap (Title, Description, Creation_date, Last_adjustment, Format, Copyright,
                            id_Topic, Learning_level,   Learning_area, Language)  VALUES ('" . $titel. "' ,'" . $beschrijving . "'
                            ,'" . $row['Creation_date'] . "','" .$row['Last_adjustment'] . "','" .$row['Format'] . "','" .$row['
                            Copyright'] . "','".$row['id_Topic'] . "','" . $level . "','" . $row['Learning_area'] . "','". $area .
                            "')";

                $result = mysql_query($sql);

                $sql = "SELECT max( Id_mindmap ) AS blub FROM mindmap";
                $result = mysql_query($sql);
                    while ($row = mysql_fetch_assoc($result))
```

```
                    { $new_leerobjectID = $row['blub'];}

                }

        return $new_leerobjectID;

    }
    function getUnit($unit_id)
    {
    return  $this->array_unit[$unit_id];
    }
}

?>
```

```
<?

if($soort == "leerobject"){
    ?>

    <html>
<head>
<title><? echo($my_class->getTitle()); ?></title>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1"></head>
<body>
    <table border="1" align="center">

    <tr><td colspan=""><h1><? echo($my_class->getTitle()); ?></h1></td><tr>

    <tr><td>
        <table><tr>
        <?
        $color[0]= "66CCFF";
        $color[1]= "993300";
        $color[2]= "CC9900";
        $color[3]= "FF6633";
        $color[4]= "FF6699";
        $color[5]= "660000";
        $x = -1;
        foreach( $my_class->getUnits() as $key => $value)
        {
            $x = $x + 1;
            echo ("<td><table bgcolor='" .$color[$x] . "'><tr><td>" . $value->getTitle()."<br>" );
            $unit = $value;
```

```
$unit_id = $key;
if(count($unit->getBlocks()) > 0)
{
    $y = $x;
foreach( $unit->getBlocks() as $key => $value)
{

        if($value->getChildOf($unit_id) == 0)
        {
            $y = $y + 1;

            echo ("<table bgcolor='" .$color[$y] . "'><tr><td><a href='index.php?action=layoutblock&block_id=" . $key
                .  "&unit_id=" . $unit_id ."&layout_id=1&id=". $leerobject_id ."'><b>"  . $value->getTitle() . "</b></
                a></td></tr></table>");


            foreach( $unit->getBlocks() as $key2 => $value2)
            {
                if($value2->getChildOf($unit_id) == $key)
                {
                echo ("<table bgcolor='" .$color[$y] . "'><tr><td><a href='index.php?action=layoutblock&block_id=" . $
                    key2 .  "&unit_id=" . $unit_id ."&layout_id=1&id=". $leerobject_id ."'>" . $value2->getTitle() . "
                    </td></tr></table></a>");

                }
            }
            echo("<br>");

        }
}
```

```
            echo("</td>");
            }

            echo("</tr></td></table></td>");
        }

         ?>
        </tr></table>
    </td></tr><tr><td><? echo($my_class->getTitle()); ?></td></tr>
    </table>
    </body>
</html>
    <?
    }
    if($soort == "block"){
    ?>
    <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN""http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title><? echo($my_class->getTitle()); ?></title>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1">
<script language="javascript">
<!--
//from http://www.webmasterworld.com/forum91/441.htm
var state = 'none';

function showhide(layer_ref) {

if (state == 'block') {
state = 'none';
```

```
}
else {
state = 'block';
}
if (document.all) { //IS IE 4 or 5 (or 6 beta)
eval( "document.all." + layer_ref + ".style.display = state");
}
if (document.layers) { //IS NETSCAPE 4 or below
document.layers[layer_ref].display = state;
}
if (document.getElementById &&!document.all) {
hza = document.getElementById(layer_ref);
hza.style.display = state;
}
}
//-->
</script>
</head>
<body>

    <table border="1" align="center">

    <tr><td colspan=""><h1><? echo($my_class->getTitle()); ?></h1></td><tr>

    <tr><td>
        <table><tr>
        <?
        $color[0]= "66CCFF";
        $color[1]= "993300";
        $color[2]= "CC9900";
```

```
        $color[3]= "FF6633";
        $color[4]= "FF6699";
        $color[5]= "660000";
        $x = -1;
        foreach( $my_class->getUnits() as $key => $value)
        {
            $x = $x + 1;
            echo ("<td><table bgcolor='" .$color[$x] . "'><tr><td>" . $value->getTitle()."<br>" );
            $unit = $value;

            $unit_id = $key;
            if(count($unit->getBlocks()) > 0)
            {
                $y = $x;
            foreach( $unit->getBlocks() as $key => $value)
            {

                    if($value->getChildOf($unit_id) == 0)
                    {
                        $y = $y + 1;

                        echo ("<table bgcolor='" .$color[$y] . "'><tr><td><a href='index.php?action=layoutblock&block_id=" . $key
                            .  "&unit_id=" . $unit_id ."&layout_id=1&id=". $leerobject_id ."'><b>"  . $value->getTitle() . "</b></
                            a></td></tr></table>");


                        foreach( $unit->getBlocks() as $key2 => $value2)
                        {
                            if($value2->getChildOf($unit_id) == $key)
                            {
```

```
                        echo ("<table bgcolor='" .$color[$y] . "'><tr><td><a href='index.php?action=layoutblock&block_id=" . $
                            key2 .  "&unit_id=" . $unit_id ."&layout_id=1&id=". $leerobject_id ."'>" . $value2->getTitle() . "
                            </td></tr></table></a>");

                        }
                    }
                    echo("<br>");

                }
            }
            echo("</td>");
            }

            echo("</tr></td></table></td>");
        }

         ?>
        </tr></table>
    </td></tr><tr><td>
    <?
    //audio van het block
    if (isset($block->audio))
    {
        ?><p><a href="#" onclick="showhide('audio');"><img src='pics/audio_icon.gif' /></a></p>
        <div id="audio" style="display: none;"><? $block->audio->showAudio(); ?></div><?
    $block->audio->showAudio();
    }
    //Q&A van het block
    if (isset($block->QA))
    {
```

```
    $block->QA->showQA();
    }

    //gewone blocks
    if(count($block->Alo_text) > 0)
            {
                $eerstetekst = true;
                foreach( $block->Alo_text as $key => $value)
                {

                    if($eerstetekst)
                    {

                    ?>
                        <p><? $value->showText(); ?></p>
                    <?


                        $eerstetekst = false;

                    }
                    else
                    {
                    $blub = "div" . $key;
                    ?>
                        <p><a href="#" onclick="showhide('<? echo $blub ?>');"><img src='pics/ft.jpg' /></a></p>
                        <div id="<? echo $blub ?>" style="display: none;"><? $value->showText(); ?></div>
                    <?

                    }
```

```
                }
            }
    if(count($block->Alo_figure) > 0)
            {

                foreach( $block->Alo_figure as $key => $value)
                {

                    $blub = "div" . $key;
                    ?>
                        <p><a href="#" onclick="showhide('<? echo $blub ?>');"><img src='pics/figuur.jpg' /></a></p>
                        <div id="<? echo $blub ?>" style="display: none;"><? $value->showFigure(); ?></div>
                    <?
                }
            }
    if(count($block->Alo_audio) > 0)
            {

                foreach( $block->Alo_audio as $key => $value)
                {

                    $blub = "div" . $key;
                    ?>
                        <p><a href="#" onclick="showhide('<? echo $blub ?>');"><img src='pics/audio_icon.gif' /></a></p>
                        <div id="<? echo $blub ?>" style="display: none;"><? $value->showAudio(); ?></div>
                    <?
                }
            }
```

```
        ?>

</td></tr>
    </table>
    </body>
</html>
    <? } ?>
```

```
<?

if($soort == "leerobject"){
    ?>

    <html>
<head>
<title><? echo($my_class->getTitle()); ?></title>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1"></head>
<body>
    <table border="1" align="center">

    <tr><td colspan="2"><h1><? echo($my_class->getTitle()); ?></h1></td><tr>

    <tr><td>
        <table>
        <? foreach( $my_class->getUnits() as $key => $value)
        {

            echo ("<tr><td>" . $value->getTitle()."</tr></td>" );
            $unit = $value;

            $unit_id = $key;
            if(count($unit->getBlocks()) > 0)
            {
            foreach( $unit->getBlocks() as $key => $value)
            {

                    if($value->getChildOf($unit_id) == 0)
                    {
```

```
                        echo ("<tr><td><a href='index.php?action=layoutblock&block_id=" . $key .  "&unit_id=" . $unit_id ."&
                            layout_id=0&id=". $leerobject_id ."'>"  . $value->getTitle() . "</a></tr></td>");
                        foreach( $unit->getBlocks() as $key2 => $value2)
                        {
                            if($value2->getChildOf($unit_id) == $key)
                            {
                            echo ("<tr><td>--><a href='index.php?action=layoutblock&block_id=" . $key2 .  "&unit_id=" . $unit_id .
                                "&layout_id=0&id=". $leerobject_id ."'>" . $value2->getTitle() . "</a></tr></td>");

                            }
                        }
                    }
            }
            }
        }

         ?>
        </table>
    </td><td><? echo($my_class->getTitle()); ?></td></tr>
    </table>
    </body>
</html>
    <?
    }
    if($soort == "block"){
    ?>
    <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN""http://www.w3.org/TR/html4/strict.dtd">
<html>
```

```
<head>
<title><? echo($my_class->getTitle()); ?></title>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1">
<script language="javascript">
<!--

var state = 'none';

function showhide(layer_ref) {

if (state == 'block') {
state = 'none';
}
else {
state = 'block';
}
if (document.all) { //IS IE 4 or 5 (or 6 beta)
eval( "document.all." + layer_ref + ".style.display = state");
}
if (document.layers) { //IS NETSCAPE 4 or below
document.layers[layer_ref].display = state;
}
if (document.getElementById &&!document.all) {
hza = document.getElementById(layer_ref);
hza.style.display = state;
}
}
//-->
</script>
</head>
```

```
<body>

    <table border="1" align="center">

    <tr><td colspan="2"><h1><? echo($my_class->getTitle()); ?></h1></td><tr>

    <tr><td>
        <table>
        <? foreach( $my_class->getUnits() as $key => $value)
        {

            echo ("<tr><td>" . $value->getTitle()."</tr></td>" );
            $unit = $value;

            $unit_id = $key;
            if(count($unit->getBlocks()) > 0)
            {
            foreach( $unit->getBlocks() as $key => $value)
            {

                    if($value->getChildOf($unit_id) == 0)
                    {


                        echo ("<tr><td><a href='index.php?action=layoutblock&block_id=" . $key .  "&unit_id=" . $unit_id ."&
                            layout_id=0&id=". $leerobject_id ."'>"  . $value->getTitle() . "</a></tr></td>");
                        foreach( $unit->getBlocks() as $key2 => $value2)
                        {
                            if($value2->getChildOf($unit_id) == $key)
                            {
```

```
                        echo ("<tr><td>--><a href='index.php?action=layoutblock&block_id=" . $key2 .  "&unit_id=" . $unit_id .
                            "&layout_id=0&id=". $leerobject_id ."'>" . $value2->getTitle() . "</a></tr></td>");

                        }
                    }
                }
            }
            }
        }

         ?>
        </table>
    </td><td>
    <?
    //audio van het block
    if (isset($block->audio))
    {
        ?><p><a href="#" onclick="showhide('audio');"><img src='pics/audio_icon.gif' /></a></p>
        <div id="audio" style="display: none;"><? $block->audio->showAudio(); ?></div><?
    $block->audio->showAudio();
    }
    //Q&A van het block
    if (isset($block->QA))
    {
    $block->QA->showQA();
    }

    //gewone blocks
    if(count($block->Alo_text) > 0)
            {
```

```
                $eerstetekst = true;
                foreach( $block->Alo_text as $key => $value)
                {

                    if($eerstetekst)
                    {

                    ?>
                        <p><? $value->showText(); ?></p>
                    <?

                        $eerstetekst = false;

                    }
                    else
                    {
                    $blub = "div" . $key;
                    ?>
                        <p><a href="#" onclick="showhide('<? echo $blub ?>');"><img src='pics/ft.jpg' /></a></p>
                        <div id="<? echo $blub ?>" style="display: none;"><? $value->showText(); ?></div>
                    <?

                    }
                }
            }
    if(count($block->Alo_figure) > 0)
            {

                foreach( $block->Alo_figure as $key => $value)
                {
```

```
                    $blub = "div" . $key;
                    ?>
                        <p><a href="#" onclick="showhide('<? echo $blub ?>');"><img src='pics/figuur.jpg' /></a></p>
                        <div id="<? echo $blub ?>" style="display: none;"><? $value->showFigure(); ?></div>
                    <?
                }
            }
    if(count($block->Alo_audio) > 0)
            {

                foreach( $block->Alo_audio as $key => $value)
                {

                    $blub = "div" . $key;
                    ?>
                        <p><a href="#" onclick="showhide('<? echo $blub ?>');"><img src='pics/audio_icon.gif' /></a></p>
                        <div id="<? echo $blub ?>" style="display: none;"><? $value->showAudio(); ?></div>
                    <?
                }
            }


        ?>

    </td></tr>
    </table>
</body>
</html>
    <? } ?>
```

Auteursrechterlijke overeenkomst

Opdat de Universiteit Hasselt uw eindverhandeling wereldwijd kan reproduceren, vertalen en distribueren is uw akkoord voor deze overeenkomst noodzakelijk. Gelieve de tijd te nemen om deze overeenkomst door te nemen, de gevraagde informatie in te vullen (en de overeenkomst te ondertekenen en af te geven).

Ik/wij verlenen het wereldwijde auteursrecht voor de ingediende eindverhandeling:
Een systeem voor het assembleren van leerinhoud in een herbruikbaar leerobject. Gevalstudie : het LOMS
Richting: **Handelsingenieur in de beleidsinformatica** Jaar: **2007**
in alle mogelijke mediaformaten, - bestaande en in de toekomst te ontwikkelen - , aan de Universiteit Hasselt.

Niet tegenstaand deze toekenning van het auteursrecht aan de Universiteit Hasselt behoud ik als auteur het recht om de eindverhandeling, - in zijn geheel of gedeeltelijk -, vrij te reproduceren, (her)publiceren of distribueren zonder de toelating te moeten verkrijgen van de Universiteit Hasselt.

Ik bevestig dat de eindverhandeling mijn origineel werk is, en dat ik het recht heb om de rechten te verlenen die in deze overeenkomst worden beschreven. Ik verklaar tevens dat de eindverhandeling, naar mijn weten, het auteursrecht van anderen niet overtreedt.

Ik verklaar tevens dat ik voor het materiaal in de eindverhandeling dat beschermd wordt door het auteursrecht, de nodige toelatingen heb verkregen zodat ik deze ook aan de Universiteit Hasselt kan overdragen en dat dit duidelijk in de tekst en inhoud van de eindverhandeling werd genotificeerd.

Universiteit Hasselt zal mij als auteur(s) van de eindverhandeling identificeren en zal geen wijzigingen aanbrengen aan de eindverhandeling, uitgezonderd deze toegelaten door deze overeenkomst.

Ik ga akkoord,

Bert MERTENS

Datum: **24.08.2007**