To appear in *Lecture Notes on Informatics*
Proceedings of „Die e-Learning Fachtagung Informatik“ (DeLFI), Lübeck 2008.

Der Graphical Learning Modeller: eine intuitive Modellierungsumgebung zur Erstellung IMS Learning Design konformer Lehr-/Lernabläufe

Susanne Neumann[1], Stefan Zander[2], Petra Oberhuemer[1]

[1]Lehrentwicklung, [2]Multimedia Information Systems Group
Universität Wien
[1]Porzellangasse 33a, 1090 Wien, Österreich
[2]Liebiggasse 4/3-4, 1010 Wien, Österreich
susanne.neumann-heyer@univie.ac.at
stefan.zander@univie.ac.at
petra.oberhuemer@univie.ac.at

Abstract: Während der Großteil der E-Learning-Standards Lehr-/Lerninhalte beschreibt und arrangiert, bildet die IMS Learning Design (IMS LD) Spezifikation die Ausnahme, da sie die Lern- und Lehraktivitäten in den Mittelpunkt rückt. Problematisch ist jedoch, dass die weite Verbreitung von IMS LD von der technischen Sprache der Spezifikation behindert wird. In diesem Artikel wird die Modellierungssoftware Graphical Learning Modeller (GLM) vorgestellt, die Lehrende unterstützt, IMS LD konforme Lehr-/Lerneinheiten (Units of Learning) für die virtuelle Lehre zu erstellen. Die Software überbrückt den Graben zwischen technischer Sprache und Lehrendensprache, indem sie grafische Sequenzen von Lehr- und Lernaktivitäten in eine eXtensible Markup Language (XML) Datei übersetzt, die dann von Lernmanagementsystemen interpretiert werden kann. In diesem Artikel werden die wesentlichen Funktionalitäten sowie die zugrunde liegenden technischen Strukturen des GLM beschrieben.

1 Problembeschreibung

Lehr-/Lernabläufe, bei denen alle Beteiligten präsent sind, binden natürlicherweise (Inter)Aktionen dieser Beteiligten ein. E-Learning gestützte Lehr-/Lernabläufe greifen jedoch oft auf ein Modell zurück, das Lernende in einer isolierten Rolle sieht und in dem Lernende lediglich mit Lern*inhalten* agieren [Br07]. Die Entwicklung von E-Learning de facto Standards spiegelt diese Diskrepanz wider, denn sie fokussieren meist auf den Lerninhalt. Die ersten Standards rückten technische, nicht didaktische, Aspekte in den Vordergrund, unter anderem Zugänglichkeit (accessibility), Anpassbarkeit (adaptability), Kompatibilität (interoperability) und Wiederverwendbarkeit (reusability) [ADL06]. Ein Standard, der diese Attribute bedient, ist etwa das Sharable Content Object Reference Model (kurz: SCORM) [ADL06].

Die übermäßige Betonung von Lerninhalten hat in der Vergangenheit zu steifen und wenig hilfreichen Lehrgepflogenheiten geführt [BS07]. Zudem kann ein E-Learning Modell, das lediglich die Anordnung von Lerninhalten reguliert, nicht als didaktisch stimmig bezeichnet werden, da Lerninhalte allein noch keine pädagogische Situation konstituieren [Sl04]. Diese Eindimensionalität beantwortete die IMS Learning Design (IMS LD) Spezifikation mit einer technischen Sprache zur Beschreibung von interaktiven Lehr-/Lernabläufen für mehrere Lernende, die den Fokus auf die Lernaktivität statt auf den Lerninhalt legt [KOA03]. Das Ziel der Spezifikation ist, jede pädagogische Situation beschreiben zu können [KOA03]. Die Vorteile einer standardisierten Beschreibung von Lehr-/Lernsituationen liegen sowohl in der Wiederverwendung pädagogischer Szenarien als auch in der Wiederverwendung von Lerninhalten, denn den Lerninhalten wird eine größere Flexibilität bei der Wiederverwendung im Hinblick auf ihre pädagogischen Einsatzzwecke eingeräumt [Sl04]. Weiterhin bewirkt die standardisierte Beschreibung eine explizite Repräsentation pädagogischen Wissens, worin Aspekte visualisiert werden, die vorher nur angenommen werden konnten [BS07].

Obwohl die Autoren von IMS LD eine Möglichkeit geschaffen haben, vielfältige Interaktionen zwischen mehreren Lernenden und TutorInnen zu beschreiben, ist die bereitgestellte Sprache von der Sprache, die Lehrende in der täglichen Praxis benutzen, entfernt [GB05]. IMS LD benutzt zwar Metaphern aus der Theaterwelt, um die Möglichkeiten der Sprache deutlicher hervor zu heben, aber diese Metaphern sind nur bedingt hilfreich. Zur Verdeutlichung der Komplexität betrachten wir ein Beispiel. IMS LD unterscheidet innerhalb der *Method* (zur zeitlichen Regelung von Aktivitäten und Rollen[1]) parallele Skripts für Lehr-/Lernabläufe (so genannte *Plays*), zwischen denen die in der Lehr-/Lerneinheit (*Unit of Learning*) festgelegten Rollen zu jedem Zeitpunkt hin und zurück wechseln können. Innerhalb dieser Plays befinden sich sequenzielle Abschnitte (*Acts*), anhand derer schrittweise die Lehr-/Lernaktivitäten des nächsten Abschnitts offen gelegt werden. Lehrende können anhand dieser Metaphern nur bedingt die wahre Bedeutung der Möglichkeiten von IMS LD schlussfolgern, da der konzeptionelle Graben von den Metaphern nicht überbrückt werden kann.

Die richtige Benutzung der IMS LD Sprache ist nicht trivial, da eine große Menge an technischem Wissen und Programmierfähigkeiten zum Verständnis vonnöten sind [HOZ07]. Software, die momentan zur Verfügung steht, um IMS LD konforme Lehr-/Lernabläufe zu bauen, setzt voraus, dass die BenutzerInnen Syntax und Semantik der IMS LD Spezifikation verstehen. Deshalb wird Software benötigt, welche die Perspektive der Lehrenden unterstützt. In den nächsten Abschnitten wird die Software Graphical Learning Modeller (GLM) vorgestellt. Sie reduziert die Menge des benötigten technischen Wissens, so dass Lehrende in die Lage versetzt werden, IMS LD konforme Lehr-/Lernabläufe zu erstellen.

[1] Lehr-/Lernaktivitäten werden Rollen (d. h. von wem die beschriebene Aktivität ausgeführt wird) zugeordnet.

2 Übersicht zum Graphical Learning Modeller

2.1 Stand der Entwicklung und grundlegende Funktionalitäten

Die IMS LD Spezifikation sieht vor, dass die Entwicklung entsprechender Software in drei Stufen mit steigender Komplexität gegliedert wird: Level A (lineare Lehr-/Lernabläufe), Level B (individualisierte Lehr-/Lernabläufe und Interaktionen) sowie Level C (systemgesteuerte Mitteilungen). Der GLM ist auf der einfachsten von drei Entwicklungsstufen (Level A) abgeschlossen. Implementierungsarbeiten auf der zweiten Entwicklungsstufe (Level B) wurden teilweise fertig gestellt.

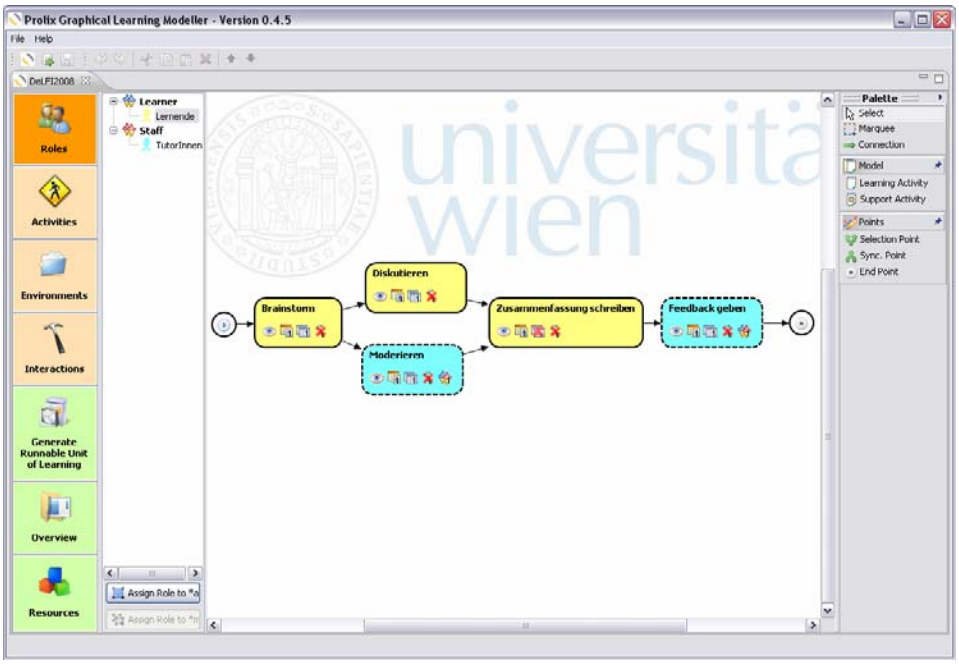

Abbildung 1: Gesamtansicht des GLM

Durch die Verwendung des GLM müssen sich Lehrende kaum mit IMS LD Konzepten auseinander setzen. Die explizite Anwendung von Plays, Acts oder Role-Parts[2] bleibt den Lehrenden erspart, da der GLM diese Konzepte automatisch aus dem grafischen Arbeitsbereich interpretiert.

[2] Über das Konzept des Role-Part wird einer Rolle eindeutig eine Aktivität zugewiesen.

Die Gesamtansicht (siehe Abb. 1) zeigt folgende Bereiche: ein Navigationsmenü am linken Rand, rechts daneben eine Auflistung der bereits angelegten Rollen (alternativ auch der angelegten Aktivitäten, Environments[3] oder Interactions[4]), der grafische Arbeitsbereich in der Mitte, worin die Aktivitäten abgebildet werden, sowie die Palette mit Designfunktionen am rechten Rand. Lehrende können mit dem GLM Lehr-/Lernabläufe erstellen, wenn sie folgendes grundlegendes Prinzip verstehen: Eine Rolle führt eine Aktivität mithilfe von Ressourcen, die in einer Environment gebündelt sind, aus. Mehr Wissen ist auf Level A bei Nutzung des GLM nicht notwendig.

Auf Level A stehen im GLM folgende fünf Grundfunktionalitäten zur Verfügung:

1. Lehr-/Lernaktivitäten anlegen und anpassen. Aus der Palette am rechten Rand wird entweder eine Learning Activity (dieser können Lernziele zugewiesen werden) oder eine Support Activity (keine Zuweisung von Lernzielen möglich) auf den Arbeitsbereich gezogen. Lehrende können diese instanziierten Aktivitäten benennen, mit einer Beschreibung versehen und bestimmte Ablaufkriterien festlegen (z. B. wie die Aktivität beendet wird). Die Aktivität erscheint als Kästchen im Arbeitsbereich des GLM und ist solange mit weißer Farbe hinterlegt, bis ihr eine Rolle zugewiesen wird. Support Activities sind an der gestrichelten Umrandung zu erkennen.

2. Aktivitäten zu einer Sequenz verbinden. Aktivitäten werden miteinander verbunden, damit eine Sequenz zwischen dem Startpunkt (Play-Tasten-Symbol) und dem Endpunkt (Stop-Tasten-Symbol) ersichtlich wird (vgl. Abb. 1). Hierzu wählen Lehrende die Funktion „Connection“ aus der Palette. Durch Klick auf die vorangehende Aktivität gefolgt vom Klick auf die nachfolgende Aktivität werden diese Aktivitäten miteinander verbunden. Es ist auch möglich, Verbindungen so anzulegen, dass parallele Pfade von Lehr-/Lernaktivitäten entstehen. Der GLM interpretiert dann die Abfolge der Aktivitäten und übersetzt diese in die IMS LD Struktur. Die detaillierte Darstellung der technischen Funktionsweise ist in Abschnitt 3 dargelegt.

3. Rollen anlegen und den Aktivitäten zuweisen. Rollen werden angelegt, um bestimmte Personengruppen nach ihren Funktionen zu unterscheiden. Es erfolgt dabei eine grundlegende Unterscheidung zwischen Learner-Rollen und Staff-Rollen. Per Default ist die Rolle „Learner“ immer vorhanden; weitere Rollen können über das Kontextmenü angelegt werden. Lehrende weisen den Rollen beliebige Farben zu, um sie voneinander zu unterscheiden. Angelegte Rollen werden per Drag&Drop auf die Aktivitäten im Arbeitsbereich gezogen. Die Aktivität nimmt sodann die Farbe der Rolle an, um anzuzeigen, welche Rolle die Aktivität ausübt.

---

[3] „Environments“ beinhalten diejenigen Materialien und Services (z. B. Kommunikationswerkzeuge), die zur Durchführung von Aktivitäten benötigt werden.

[4] In der Ansicht „Interactions“ werden Möglichkeiten geboten, Level B Konzepte in den Lehr-/Lernablauf einzubauen, z. B. zum Verfassen eines Textes zur Laufzeit.

4. Environments anlegen und den Aktivitäten zuweisen. Die Materialien und Kommunikationswerkzeuge (Services), die zur Ausübung der Aktivität gebraucht werden, sind in Environments zusammengefasst. Diese können auf zweierlei Wegen angelegt werden, entweder direkt während der Beschreibung einer Aktivität oder über das Kontextmenü im Instanzenbaum (vgl. Abb. 1 links). In Bezug auf ersteres sind die so angelegten Environments automatisch einer Aktivität zugeordnet; die Environments erscheinen jedoch auch im Instanzenbaum. Beim Anlegen über das Kontextmenü im Instanzenbaum müssen Environments hingegen noch in den Arbeitsbereich, genauer auf eine Aktivität, gezogen werden, um sie der Aktivität zuzuordnen. Für jede Aktivität wird angezeigt, wie viele Environments ihr zugewiesen wurden.

5. Export der Lehr-/Lerneinheit als IMS LD konforme XML-Datei. Über das Navigationsmenü (siehe Abb. 1 links) kann die Funktion „Generate Runnable Unit of Learning" aufgerufen werden. Hier prüft der GLM, ob eine korrekte Abfolge von Lehr-/Lernaktivitäten vorliegt, ob alle Verbindungen akkurat gesetzt wurden und, ob allen Aktivitäten die ausführenden Rollen und Beschreibungen zugeordnet wurden. Bei Erfüllung kann die Lehr-/Lerneinheit exportiert werden. Somit entsteht eine IMS LD konforme XML-Datei, die in verschiedene Lernmanagementsysteme[5] importiert werden kann.

Die erforderlichen Bausteine zur Erstellung einer Lehr-/Lerneinheit sind somit vorhanden. Weitere Funktionalitäten auf Level A beinhalten das Setzen von Steuerungspunkten, z. B. im Fall, dass eine Rolle in der Lehr-/Lerneinheit zwischen verschiedenen Aktivitäten wählen kann (Nutzung des Selection Points; siehe Palette in Abb. 1) oder wenn der Ablauf der Aktivitäten zwischenzeitlich wieder auf eine Linie gebracht also synchronisiert werden soll (Nutzung des Synchronisation Point; siehe Palette in Abb. 1).

Auf der Ebene von Level B stehen erweiterte Funktionalitäten zur Verfügung. Diese erlauben eine größere Flexibilität beim Erstellen und Abspielen von IMS LD konformen Lehr-/Lerneinheiten. Die Bereitstellung eines Textfeldes, das erst zur Laufzeit mit Inhalten befüllt wird, etwa wenn Lernende eine Reflexion zu einem in der Lehr-/Lerneinheit gelesenen Artikel schreiben sollen, stellt ein Beispiel dieser Art dar. Vier Funktionalitäten wurden bereits im GLM implementiert (aufrufbar über den Button „Interactions" im Navigationsmenü): das Anlegen von Textfeldern, das Laden von externen Dateien, die Erstellung und Beantwortung von Frage- und Antworttests sowie die Einbindung eines Multiple-Choice-Tests. Wohlgemerkt werden alle diese Funktionen erst zur Laufzeit (nicht zur Designzeit) mit den entsprechenden Inhalten befüllt. Vorarbeiten zur Konzeption von technikfernen Benutzerschnittstellen für Level B Funktionen sind in [HOZ07] beschrieben und konzentrieren sich größtenteils auf die Überwindung des Grabens zwischen programmiertechnischen Konzepten und der natürlichen Sprache von Lehrenden.

---

[5] Systeme, die IMS LD interpretieren können, sind beispielsweise die Erweiterung GRAIL für das Lernmanagementsystem dotLRN [https://gradient.it.uc3m.es/xowiki/main_page] sowie zu Demonstrationszwecken auch der Service-based Learning Design (SLeD) Player, http://sled.open.ac.uk/sledweb/ [Letzter Zugriff: 25.02.2008]

2.2 Bewertung durch NutzerInnen

Im Verlauf der Entwicklung des GLM wurden mehrere Testläufe mit unterschiedlichen Nutzergruppen durchgeführt. Im frühen Entwicklungsstadium wurden Tests mit Hochschullehrenden durchgeführt, die textuell beschriebene Lernszenarien in den GLM übertrugen. Die Lehrenden bewerteten die Software positiv und hoben dabei besonders den Komfort der Drag&Drop Funktionalitäten hervor. Ihre Wünsche bezüglich der Weiterentwicklung bezogen sich vor allem auf die Reduktion technischer Sprache in Dialogen und die Möglichkeit, ihr Learning Design sowohl aus grober Perspektive (Übersicht in großen zeitlichen Abschnitten) als auch detailliert (einzelne Aktivitäten) zu betrachten.

IMS LD-ExpertInnen, hauptsächlich EntwicklerInnen von IMS LD Editoren und Laufzeitumgebungen, repräsentierten die zweite Gruppe von BewerterInnen. Deren Rückmeldung auf strukturierte und offene Fragen gab generell einen positiven Eindruck wider; die offenen Kommentare richteten sich vorrangig auf die technische Interpretation von IMS LD Konzepten im GLM.

Abbildung 2: ISO 9241/110 Bewertung durch projektbeteiligte Unternehmen, nach [Fa08].

Vier projektbeteiligte Unternehmen, die den GLM für die Konfiguration von Lehr-/Lernszenarien in der betrieblichen Weiterbildung benutzen, stellten die dritte Nutzergruppe dar, von der Rückmeldungen eingeholt wurden. Abb. 2 zeigt die durchschnittliche Bewertung für den GLM in Bezug auf die Grundsätze der Dialoggestaltung nach DIN EN ISO 9241/110 Norm [Fa08]. Deutlich positive Werte (>3) erreichte der GLM für Steuerbarkeit und Erwartungskonformität. Den größten Raum an Verbesserung bietet die Fehlertoleranz, was bedeutet, dass NutzerInnen bei fehlerhaften Eingaben schnell und einfach zur Richtigstellung hingeführt werden. In der neuesten Version des GLM sind dahingehende Besserungen bereits implementiert. Neben der durchgängig verfügbaren Undo-Funktion gibt der GLM während der Erstellung des Learning Designs abwählbare Hinweise in Dialogform, wie die Fertigstellung des Designs reibungslos erfolgen kann.

3 Technische Details des Graphical Learning Modeller

3.1 Reload als technische Grundlage des GLM

Die technische Grundlage des GLM bildet der an der University of Bolton entwickelte IMS LD Editor Reload[6] (im weiteren Verlauf kurz als Reload bezeichnet). Der Reload Editor wurde im Rahmen des gleichnamigen Projekts des Joint Information Systems Committee (JISC)[7] Exchange for Learning Programme von Phillip Beauvoir und Paul Sharples entwickelt.

Reload ist eine Eclipse Rich Client Applikation auf Basis der Eclipse Rich Client Plattform (RCP) und wurde vollständig in Java implementiert. Eclipse[8] als technische Grundlage von Reload ist ein vor allem innerhalb der Java-Community bekanntes und weit genutztes Open-Source-Framework zur Entwicklung von nahezu beliebiger Software. Die zugrunde liegende offene und flexible Architektur erlaubt Eclipse und darauf basierenden Programmen die Erweiterung bestehender Funktionalität sowie die Anpassung an bestimmte Applikationsdomänen durch den Einsatz flexibler Plug-ins.

Eine vorab durchgeführte Analyse verfügbarer Editoren zeigte, dass Reload als frei zugängliches Open-Source-Projekt, das alle drei Level der IMS LD Spezifikation abbildet, hervorragend als technische Grundlage zur Entwicklung des GLM herangezogen werden kann. Die Notwendigkeit zur Entwicklung eines eigenen IMS LD Editors ergab sich vor allem durch Reloads geteilte Ansicht der Design Elemente (getrennte Ansichten für Rollen, Aktivitäten, Method etc.). Dies erschwert das Erstellen von Lehr-/Lernabläufen und verhindert die gesamtheitliche Sicht auf die Lehr-/Lerneinheit. Zeitliche Abläufe sowie logische Zusammenhänge innerhalb einer Lehr-/Lerneinheit lassen sich somit nur schwer erschließen.

Vor der Entwicklung des GLM wurde der Reload Editor eingehend hinsichtlich der Wiederverwendbarkeit und Erweiterung bestehender Komponenten untersucht. Die in Reload enthaltenen Datenmodell-Klassen sowie Teile der Business-Logik wurden mittels eigens implementierter Wrapper-Klassen gekapselt und mit GLM-spezifischen Daten und Logik erweitert. Hierzu wurde eine zusätzliche funktionale Schicht oberhalb der Reload-Business-Logik implementiert, die über eine eigene Business-Logik und ein eigenes Datenmodell für die visuelle Darstellung von Elementen auf dem Arbeitsbereich verfügt (vgl. Abb. 3). Diese funktionale Schicht steuert die Verarbeitung der Benutzer-Interaktionen mit den Arbeitsbereichelementen und enthält die zur grafischen Anzeige der Arbeitsbereichselemente notwendigen Daten.

---

[6] Reusable Learning Object Authoring and Delivery
[7] http://www.jisc.ac.uk/ [Letzter Zugriff: 25.02.2008]
[8] http://www.eclipse.org/ [Letzter Zugriff: 25.02.2008]

3.2 Das Graphical Editing Framework als Grundlage zur Modellierung von Lehr-/Lernabläufen im GLM

Um die Erstellung IMS LD konformer Lehr-/Lernabläufe zu vereinfachen, wurde ein grafisches Framework in den GLM eingebunden, auf dessen Grundlage zeitliche Abläufe und logische Zusammenhänge grafisch modelliert werden können. Damit wird eine ganzheitliche Sicht auf die Lehr-/Lernabläufe geboten, die Änderungen an den erstellten Abläufen direkt wiedergibt.

Eclipse ermöglicht den Einsatz zweier grafischer Frameworks, des Graphical Editing Framework (GEF) und des Graphical Modelling Framework (GMF). Die Entscheidung, das GEF als grafisches Framework innerhalb des GLM einzusetzen, resultierte aus einer zuvor durchgeführten Evaluation der für die Eclipse Plattform existierenden Modellierungsframeworks. Die Evaluation zeigte, dass das GMF weniger gut für das Vorhaben geeignet ist, da es eher auf die Erstellung von UML-Modellen ausgelegt ist. Ferner erfordert das GMF, dass das zugrunde liegende Datenmodell konform zum Eclipse Modelling Framework (EMF) sein muss. Dies hätte zur Folge, dass das Reload-Datenmodell in ein entsprechendes EMF-Datenmodell transformiert werden müsste, was – aus unserer Sicht – keinen erkennbaren Mehrwert lieferte. Im Gegensatz dazu unterliegt das GEF keiner solchen Restriktion, da es die Verwendung beliebiger Datenmodelle erlaubt.

Neben der Neutralität hinsichtlich der Anwendungsdomäne liegt ein weiterer Vorteil von GEF in der Möglichkeit, eine Vielzahl grafischer Anwendungen und Repräsentationen erstellen zu können. Zu den Funktionalitäten zählen die Bereitstellung einer Werkzeugpalette zur Erstellung eines grafischen Modells und - in Bezug auf die Elemente im grafischen Arbeitsbereich - die visuelle Komposition sowie die Verknüpfung mittels Drag&Drop und das Wiederholen und Rückgängigmachen von Manipulationen.

Eines der zentralen Konzepte, auf dem das GEF basiert, ist das Model-View-Controller (MVC) Konzept. Es erlaubt die Erweiterung und Verbesserung bestehender Funktionalität, wobei das zugrunde liegende Datenmodell unangetastet bleibt. Die Aufteilung in Domain-Modell (Model), visuelle Repräsentation (View) und Verarbeitungslogik (Controller) bietet Flexibilität hinsichtlich der Erweiterung des Funktionsumfanges des GLM, ohne eine Anpassung des zugrunde liegenden Reload-Datenmodells zu bewirken.

GEF und auf Eclipse RCP basierende Anwendungen verwenden für die Anzeige grafischer Elemente das Standard Widget Toolkit (SWT) in Verbindung mit JFace, das als Schnittstelle für den Zugriff und die Anzeige nativer GUI-Elemente des zugrunde liegenden Betriebssystems fungiert. Die Nutzung von SWT und JFace ist in der logischen Architektur des GLM in Abb. 3 dargestellt.

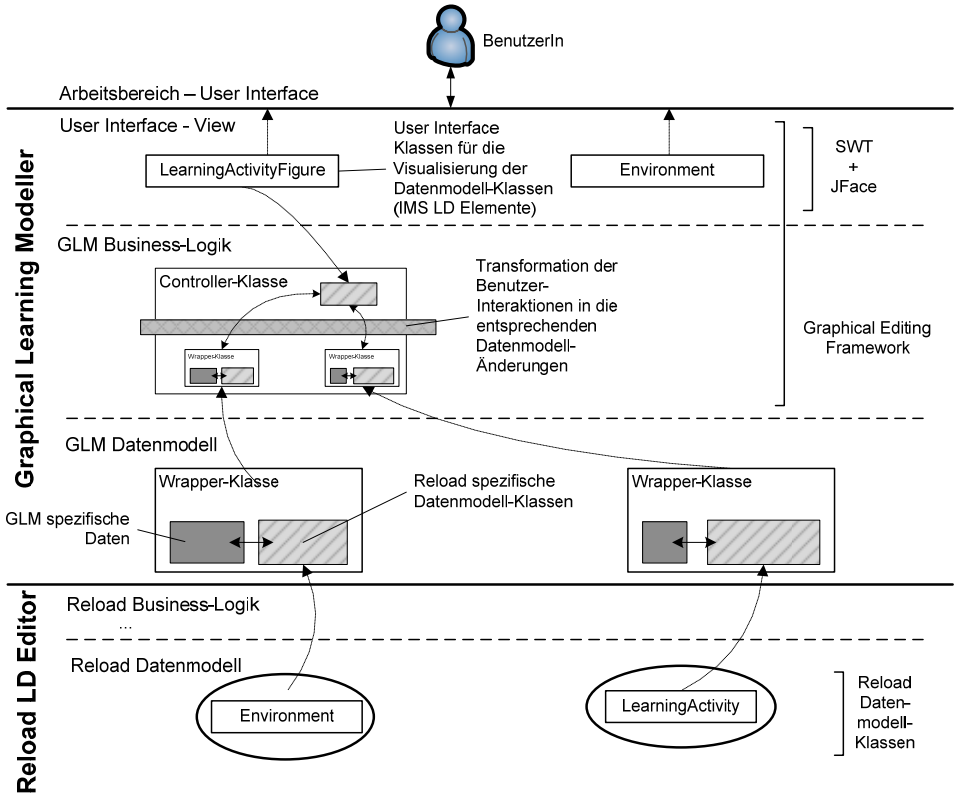

Abbildung 3: Logische Architektur des GLM

Die Transformation des Reload-Datenmodells in GEF-konforme Strukturen, d. h. Klassen, wurde mittels Wrapper-Klassen realisiert. So existieren Wrapper-Klassen für das Datenmodell als auch für Teile der in Reload implementierten Business Logik, beispielsweise für den Export IMS LD konformer Lehr-/Lernabläufe. Die Verwendung von GEF erfordert, dass diese Klassen konform zum verwendeten GEF Framework sein müssen, um den vollen Funktionsumfang (Drag&Drop, Redo&Undo etc.) bereitzustellen.

3.3 Algorithmen zur Erstellung IMS LD konformer Lehr-/Lernabläufe

Die Möglichkeit der intuitiven Modellierung im Arbeitsbereich erfordert unter anderem die logische Auswertung der grafisch erstellten Lehr-/Lernabläufe. Eine der Herausforderungen lag hierbei in der automatischen Erkennung und Generierung von Acts, da der GLM (und nicht die Lehrende) entscheidet, welche Aktivitäten in einem Act zusammengefasst werden können. Hierzu wurde ein Regelwerk ausgearbeitet, auf dessen Grundlage Algorithmen zur automatischen Act-Erkennung und Act-Strukturierung implementiert wurden. Diese Algorithmen untersuchen den modellierten Lehr-/Lernablauf auf Widerspruchsfreiheit, Endlosschleifen sowie unerreichbare Pfade und überführen ihn intern in ein IMS LD konformes Format.

Als Beispiel betrachten wir Abb. 1. Zu sehen sind fünf Aktivitäten, die von zwei Rollen ausgeführt werden. Nach der ersten Aktivität „Brainstorm“ laufen zwei parallele Aktivitäten ab: Die Lernenden diskutieren, während die Tutorin moderiert. Danach schreiben die Lernenden eine Zusammenfassung, und die Tutorin gibt in einer nächsten Aktivität auf die geschriebenen Zusammenfassungen Feedback. Zur Interpretation dieses Lehr-/Lernablaufs nutzt der GLM folgende Regeln aus dem Regelwerk:

Regel 1: Wenn mehrere Verbindungen eine Aktivität verlassen und in zwei oder mehrere Aktivitäten mit anderen Rollen münden, und wenn es keine weiteren parallelen Aktivitätenstränge gibt, dann eröffne einen neuen Act. Diese Regel bezieht sich auf die Verbindung zwischen der Ausgangsaktivität „Brainstorm“ und den Folgeaktivitäten „Diskutieren“ und „Moderieren“.

Regel 2: Wenn mehr als eine Verbindung in eine Aktivität mündet, prüfe ob es weitere parallele Aktivitäten oder Aktivitätenstränge gibt, die andere Rollen einbinden, und die länger dauern als diese Aktivität (d. h. deren Verbindungen in eine spätere Aktivität münden). Falls dies nicht *der Fall ist, eröffne einen neuen Act hier. Falls dies der Fall ist, gehe zu den Level B Regeln.* Diese Regel bezieht sich auf die Verbindung zwischen den Ausgangsaktivitäten „Diskutieren“ und „Moderieren“ sowie der Folgeaktivität „Zusammenfassung schreiben“. Im hier angeführten Beispiel kommt die einfache Variante ohne Nutzung von Level B Konzepten zur Anwendung.

Regel 3: Wenn eine Verbindung eine Aktivität verlässt und in eine Aktivität mündet, die einer anderen Rolle zugeordnet ist, eröffne einen neuen Act. Diese Regel bezieht sich auf die Verbindung zwischen den Aktivitäten „Zusammenfassung schreiben“ und „Feedback geben“.

Aufgrund der eben genannten und weiterer, hier nicht angeführten, Regeln wurden Algorithmen entwickelt, die den grafischen Arbeitsbereich zur Interpretation nach IMS LD auslesen. Dies erfolgt nach einem Schema, das in den folgenden Absätzen beschrieben wird, beispielhaft dargestellt am Lehr-/Lernablauf in Abb. 1.

Wird z. B. eine Lehr-/Lerneinheit exportiert, so liest der Algorithmus alle im grafischen Arbeitsbereich vorhandenen Elemente (z. B. Aktivitäten, Selection Points) aus und überführt sie in eine interne Struktur. Dieser Struktur liegt ein eigenes Modell zur Interpretation zugrunde, weil die grafischen Daten für die Erzeugung einer IMS LD konformen Lehr-/Lerneinheit ungeeignet sind. Nach der Überführung prüft der Algorithmus zuerst, welche Elemente der internen Modellstruktur direkt vom Startpunkt aus referenziert werden; im Beispiel in Abb. 1 ist dies nur die Aktivität „Brainstorm“. Der Algorithmus erzeugt einen ersten Act und weist all jene Elemente, die vom Startpunkt aus referenziert werden, diesem Act zu. Danach prüft der Algorithmus, ob die Liste der Elemente, die vom Startpunkt referenziert wurden, mehr als ein Element enthält. In diesem Fall enthält die Liste nur ein Element, die Aktivität „Brainstorm“, also enthält der erste Act nur die Aktivität „Brainstorm“.

Danach baut der Algorithmus eine Nachfolgeliste auf, indem er alle Elemente, die dem Element „Brainstorm“ unmittelbar folgen, ausliest. Im Beispiel sind das die Aktivitäten „Diskutieren“ und „Moderieren“. Der Algorithmus erzeugt einen neuen Act und legt alle gefundenen Elemente hinein. Da mehr als ein Element in der Liste ist, sucht der Algorithmus in diesem Fall nach möglichen „Endpunkten“, um festzustellen, ob noch weitere Elemente in diesen Act eingebunden werden müssen. Die Aktivitäten, die am Anfang der erzeugten Liste stehen, bilden Startpunkte für die Prüfung (hier: „Diskutieren“ und „Moderieren“). Der Algorithmus testet nun den Lehr-/Lernablauf auf mögliche Endpunkte zur Beendigung des Acts. Der Endpunkt ist dann der richtige Endpunkt für den Act, wenn von diesem Endpunkt aus alle Startpunkte (d. h. Aktivitäten) durch die im Arbeitsbereich gesetzten Verbindungen erreicht werden[9]. Im Beispiel bildet das Element „Zusammenfassung schreiben“ den Endpunkt zur Bestimmung des laufenden Acts. Der Algorithmus weist alle Elemente, die von den Startpunkten aus vor dem Endpunkt liegen, diesem Act zu, also die Elemente „Diskutieren“ und „Moderieren“.

Der identifizierte Endpunkt bildet gleichzeitig den Startpunkt für den neuen Act. Von diesem Startpunkt wird wiederum eine neue Liste erstellt, und der eben beschriebene Prüfprozess wird wiederholt. Somit werden die Aktivitäten „Zusammenfassung schreiben“ und „Feedback geben“ jeweils in eigene Acts verwiesen. Bei komplexeren Lehr-/Lernabläufen als dem in Abb. 1 dargestellten, z. B. wenn mehrere Aktivitäten hintereinander von derselben Rolle ausgeführt werden, prüft der Algorithmus zusätzlich, ob Acts zusammengelegt werden und durch andere IMS LD Konzepte wie Activity Structures[10] besser abgebildet werden können.

Nachdem diese Algorithmen endgültig durchlaufen wurden, findet eine Transformation aller in der internen Struktur festgestellten Elemente in Reload konforme Datenstrukturen statt. Dies beinhaltet hauptsächlich das Schreiben der Method. Für das Beispiel in Abb. 1 erzeugt Reload nach der Übertragung innerhalb der Method ein Play, vier Acts und fünf Role-Parts (für jede Aktivität, die von einer Rolle ausgeführt wird). Der Reload eigene Speicherprozess erzeugt daraus beim Export ein IMS Content Package[11] mit der zugehörigen imsmanifest.xml-Datei, die die IMS LD konforme Beschreibung des Lehr-/Lernprozesses enthält.

4 Fazit und Ausblick

Die IMS LD Spezifikation bietet die Möglichkeit, komplexe Lehr-/Lernabläufe abzubilden. Ihre weitestgehend technische Sprache bleibt jedoch für den Großteil der Lehrenden unzugänglich. Die Modellierungsumgebung GLM reduziert das Ausmaß des benötigten technischen Wissens, sodass Lehrende in die Lage versetzt werden, auf intuitive Art IMS LD konforme Lehr-/Lernabläufe zu erstellen.

---

[9] Der Lehr-/Lernablauf in Abb. 1 bietet nur ein geringes Maß an Komplexität, so dass die mehrfache Bestimmung und Verwerfung von Endpunkten nicht deutlich veranschaulicht werden kann.
[10] Die Activity Structure ermöglicht es, Aktivitäten sequenziell oder alternativ, also zur Auswahl, anzuordnen.
[11] IMS Content Packaging ist ein Standard zur Strukturierung von Daten http://www.imsglobal.org/content/packaging/cpv1p1p4/imscp_infov1p1p4.html [Letzter Zugriff: 25.02.2008]

Lehrenden wird somit ein Werkzeug zur Darstellung und Dokumentation ihrer pädagogischen Erfahrungen an die Hand gegeben. Die einheitliche, auf den Konzepten der IMS LD Spezifikation basierte Darstellung von Lehr-/Lernabläufen ermöglicht erstmals eine Vergleichbarkeit pädagogischer Umsetzungen und kann die weitere Verwendung bestehender Szenarien durch Dritte anstoßen. Da EntwicklerInnen von Lernmanagementsystemen vermehrt die IMS LD Spezifikation implementieren, kann zukünftig der Designprozess von Lehr-/Lernabläufen unabhängig von der Verfügbarkeit eines Lernmanagementsystems bei gleichzeitiger Sicherstellung der Abspielbarkeit durchgeführt werden.

In Zukunft wird der Zugang zu den erweiterten Konzepten auf Level B noch verbessert, da Level B den größeren Teil der Spezifikation und seiner bereitgestellten Möglichkeiten ausmacht. Hierfür werden geeignete Visualisierungs- und Benutzungskonzepte geprüft.

Danksagung

Dieser Artikel entstand im Rahmen des Forschungs- und Entwicklungsprojekts PROLIX, welches als Integrated Project im sechsten EU-Rahmenprogramm mit Schwerpunkt „Information Society Technologies“ gefördert wird. Die in dem Artikel erwähnte Eigenentwicklung GLM erfolgte an der Universität Wien Multimedia Information Systems Group durch Philipp Prenner und Stefan Zander unter Leitung von Wolfgang Klas.

Literaturverzeichnis

[ADL06] Advanced Distributed Learning (ADL): Sharable Content Object Reference Model (SCORM®) 2004 3rd Edition Overview. Alexandria, VA, 2006.

[BS07] Beetham, H.; Sharpe, R.: An introduction to rethinking pedagogy for a digital age. In (Beetham, H.; Sharpe, R., Hrsg.): Rethinking Pedagogy for a Digital Age. Designing and delivering e-learning. Routledge, London, New York, 2007; S. 1-10

[Br07] Britain, S.: Learning design systems: Current and future developments. In (Beetham, H.; Sharpe, R., Hrsg.): Rethinking Pedagogy for a Digital Age. Designing and delivering e-learning. Routledge, London, New York, 2007; S. 103-114

[Fa08] Faltin, N.: D9.3 PROLIX prototype version – pilot evaluation. Aufgerufen am 9. Mai , 2008 im World Wide Web unter http://www.prolixproject.org/index.php?option=com_remository&Itemid=45&func=fileinfo&id=107 (Registrierung notwendig), 2008.

[GB05] Griffiths, D.; Blat, J.: The Role of Teachers in Editing and Authoring Units of Learning Using IMS Learning Design. Advanced Technology for Learning, 2(4), 2005.

[HOZ07] Heyer, S.; Oberhuemer, P.; Zander, S.; Prenner, P.: Making Sense of IMS Learning Design Level B: from specification to intuitive modeling software. In (Duval, E.; Klamma, R.; Wolpers, W., Hrsg.): Lecture Notes in Computer Science 4753. Springer, Berlin, Heidelberg, 2007; S. 86-100

[KOA03] Koper, R.; Olivier, B.; Anderson, T., Hrsg.: IMS Learning Design Information Model. IMS Global Learning Consortium, 2003.

[Sl04] Sloep, P.: The Language of Flexible Reuse; Reuse, Portability and Interoperability of Learning Content or Why an Educational Modelling Language. In (McGreal, R., Hrsg.): Online Education Using Learning Objects. Routledge/Falmer, London, 2004; S. 128-137