Hochschule für Technik und Wirtschaft Dresden (FH)

Fachbereich Informatik/Mathematik

Diplomarbeit

im Studiengang Medieninformatik

Integration von Spiel-basierten Lernprogrammen in SCORM-konformen Lernumgebungen unter besonderer Beachtung des Multi-User Aspektes

Eingereicht von: Dirk Börner

Eingereicht am: 13. August 2007

Betreuer: Prof. Dr. Teresa Merino,
Hochschule für Technik und Wirtschaft Dresden (FH)

Dipl.-Inf. Mirko Ebert,
Fraunhofer Institut für Graphische Datenverarbeitung Rostock

Integration of game-based learning applications in SCORM conformant learning management systems especially considering the multi-user aspect

Abstract: Game-based learning presents a new way of electronic learning, which sets its focus mainly on the motivation of the learner and transfers knowledge more indirectly. In order for it to be a serious alternative to traditional learning applications, it is important to standardize the learning content included in game-based learning applications and the applications by itself. The most common way to standardize content in the field of e-learning is the Sharable Content Object Reference Model (SCORM). SCORM ensures the reusability and the interoperability of the content. Therefore this thesis explains at first the didactical aspects related to game-based learning and tries afterwards to develop different concepts for the integration of game-based learning applications in SCORM conformant learning management systems. The possible multi-user aspect of game-based learning applications will be especially considered in this context. Furthermore the thesis covers a future prospect regarding the possible enhancement of SCORM. Finally the thesis contains a prototypical implementation of one of the introduced concepts.

Spiel-basiertes Lernen ist eine neue Form des elektronisch unterstützten Lernens. Der Fokus liegt dabei stets auf der Motivation des Lernenden, wobei das zu vermittelnde Wissen eher indirekt transferiert wird. Um als ernsthafte Alternative neben klassischen Lernprogrammen bestehen zu können, ist es wichtig die Lerninhalte Spiel-basierter Lernprogramme und die Lernprogramme selbst zu standardisieren. Ein verbreiteter Standard zur Standardisierung von Inhalten im E-Learning Bereich ist das Sharable Content Object Reference Model (SCORM). SCORM sichert die Wiederverwendbarkeit und Interoperabilität der Lerninhalte. Die vorliegende Arbeit beschreibt zunächst die didaktischen Aspekte des Spiel-basierten Lernens. Anschließend werden verschiedene Konzepte zur Integration Spiel-basierter Lernprogramme in SCORM-konforme Lernumgebungen entwickelt. Der mögliche Multi-User Aspekt Spiel-basierter Lernprogramme wird dabei besonders berücksichtigt. Des Weiteren enthält die Arbeit auch einen Ausblick auf eine zukünftige Erweiterung von SCORM. Abschließend wird eines der vorgestellten Konzepte prototypisch umgesetzt.

Keywords: e-learning, game-based learning, Sharable Content Object Reference Model (SCORM), learning managment system, client-server model, multi-user

ACM CSS:
K. Computing Milieux
K.3 Computers and Education
K.3.1 Computer Uses in Education
Collaborative learning
Distance learning
K.3.m Miscellaneous
Game-based learning

Inhaltsverzeichnis

Abkürzungsverzeichnis..... 7

Einleitung..... 9

Einführung

1 E-Learning..... 12

1.1 Allgemeines..... 12

1.2 Begriffsklärung..... 12

1.2.1 E-Learning..... 12

1.2.2 Lernobjekt..... 12

1.2.3 Lernumgebung..... 13

1.3 Lerntheorien..... 14

1.3.1 Behaviorismus – Lernen durch Erfolg und Belohnung..... 14

1.3.2 Kognitivismus – Verstehen und Einsicht..... 14

1.3.3 Konstruktivismus – Konstruktion von Wissen..... 14

1.3.4 Ausblick..... 15

1.4 Sharable Content Object Reference Model..... 15

1.4.1 Overview..... 16

1.4.2 Content Aggregation Model..... 18

1.4.3 Run-Time Environment..... 20

1.4.4 Sequencing and Navigation..... 21

2 Spiel-basiertes Lernen..... 23

2.1 Allgemeines..... 23

2.2 Begriffsklärung..... 23

2.2.1 Spiel / Computerspiel..... 23

2.2.2 Simulation..... 24

2.2.3 Digitales Lernspiel..... 25

2.3 Didaktische Aspekte..... 26

2.3.1 Motivierende Elemente....26
2.3.2 Lernkonzept....27
2.3.3 Wissenstransfer....29
2.3.4 Interaktionsformen....31
2.4 Kategorien digitaler Lernspiele....32
2.5 Zusammenfassung....34

Konzeption

3 Anforderungsanalyse....36
3.1 Allgemeines....36
3.2 Conformance Requirements....36
3.3 Anforderungen SCORM-konformer Lernumgebungen....37
3.3.1 Allgemeine Anforderungen....37
3.3.2 Spezielle Anforderungen....40
3.4 Anforderungen Spiel-basierter Lernprogramme....43
3.5 Umsetzbarkeit der Anforderungen....48

4 Konzepterstellung....53
4.1 Allgemeines....53
4.2 Aufbau und Verwendung des Spiel-basierten Lernprogramms innerhalb SCORM-konformer Lernumgebungen....53
4.3 Konzept 1: Integration eines Spiel-basierten Lernprogramms in eine SCORM-konforme Lernumgebung....54
4.3.1 Anpassungen....54
4.3.2 Architektur....56
4.3.3 Bewertung....58
4.4 Konzept 2: Integration eines Spiel-basierten Lernprogramms in eine SCORM-konforme Lernumgebung unter Einsatz eines Multi-User-fähigen Spieleservers....58
4.4.1 Anpassung....59
4.4.2 Architektur....60
4.4.3 Bewertung....61

4.5 Ausblick: Erweiterung der SCORM Spezifikation 62
4.5.1 Anpassungen 63
4.5.2 Architektur 64
4.6 Zusammenfassung 65

Umsetzung

5 Prototypische Implementierung 68

5.1 Allgemeines 68
5.2 Verwendete Werkzeuge 68
5.2.1 Adobe Flash 68
5.2.2 SmartFoxServer 69
5.2.3 Reload Editor 71
5.2.4 ADL Sample Run-Time Environment 71
5.3 Integration in die Lernumgebung 72
5.3.1 Erstellung des Content Packages 73
5.3.2 Steuerung des Kurses 75
5.3.2.1 Sequencingstrategie 75
5.3.2.2 Sequencing Control Modes 76
5.3.2.3 Objectives 76
5.3.2.4 Sequencing Rules 77
5.4 Implementierung des Client-Server Modells 78
5.4.1 Spielanwendung 79
5.4.2 Kommunikation mit der Sample RTE 81
5.4.3 Kommunikation mit dem SmartFoxServer 84
5.5 Zusammenfassung 87

Zusammenfassung 89

Literaturverzeichnis 91
Tabellenverzeichnis 94

Abbildungsverzeichnis.....95
Quellcodeverzeichnis..... 96

Anhang

Anhang A – SCORM RTE Data Model Elemente.....98
Anhang B – imsmanifest.xml.....100
Anhang C – APIWrapper.js.....103
Anhang D – Klassenübersicht der SmartFoxServer Client API.....108
Anhang E – config.xml..... 109
Anhang F – CD.....111

Selbstständigkeitserklärung.....112

Abkürzungsverzeichnis

Abkürzung	Bedeutung
ADL	Advanced Distributed Learning Initiative
AICC	Aviation Industry CBT Committee
API	Application Programming Interface
ARIADNE	Alliance of Remote Instructional Authoring & Distribution Networks for Europe
CAM	Content Aggregation Model
CBT	Computer-Based Training
CMI	Computer Managed Instruction
CP	Content Package
CR	Conformance Requirements
DOM	Document Object Model
DTD	Document Type Definition
ECMA	European Computer Manufacturers Association
HTML	Hypertext Markup Language
ID	Identifier
IEEE	Institute of Electrical and Electronics Engineers
IMS	Instructional Management Systems
IP	Internet Protocol
KI	Künstliche Intelligenz
LCMS	Learning Content Management System
LMS	Learning Management System
LOM	Learning Objects Metadata
PIF	Package Interchange File

RTE	Run-Time Environment
SCO	Sharable Content Object
SCOs	Sharable Content Objects
SCORM	Sharable Content Object Reference Model
SN	Sequencing and Navigation
SWF	ShockWave File
TCP	Transmission Control Protocol
URL	Uniform Resource Locator
XML	Extensible Markup Language
XSD	XML Schema Definition

Einleitung

Die größte Kunst ist, den Kindern alles, was sie tun oder lernen sollen, zum Spiel zu machen.

(John Locke, englischer Philosoph und Politiker)

Was John Locke Ende des 17. Jahrhunderts feststellte, regt auch heute noch zum Nachdenken an. Wir befinden uns zwar längst in einer Gesellschaft in der Information und Wissen als wirtschaftliche Güter zählen, ein Patentrezept für die erfolgreiche Vermittlung dieser Güter haben wir bis jetzt allerdings noch nicht gefunden. Dabei beschränkt sich der Prozess des Lernens schon längst nicht mehr nur auf die Kindheit. Das lebenslange Lernen wird heutzutage immer wichtiger, die notwendige Zeit dazu aber auch immer knapper. E-Learning setzt an diesem Punkt an und verbindet bewährte Lernkonzepte mit modernen Kommunikationstechnologien. Die so entstehenden Bildungsangebote, beginnend bei Lernsoftware für Kinder bis zu umfangreichen Schulungsprogrammen, kommen allerdings oft über eine reine Präsentation des zu vermittelnden Wissens nicht hinaus. Darunter leidet vor allem die Motivation und damit die Bereitschaft zu Lernen. Genau diese Bereitschaft ist aber eine der grundlegenden Voraussetzungen für erfolgreiches Lernen.

Einen alternativen Weg bietet das sogenannte Spiel-basierte Lernen. Spiel-basiert bedeutet in diesem Zusammenhang dem Lernenden die Möglichkeit zu geben sein erworbenes Wissen auszuprobieren und so besser zu verarbeiten und schließlich zu begreifen. Die Hauptaufgabe bleibt weiterhin die Vermittlung von Wissen. Ergänzt und begleitet wird dieser Prozess durch spielerische Elemente. Die Motivation des Lernenden soll dadurch stets auf einem hohen Niveau gehalten werden, um so erfolgreiches und vor allem schnelles Lernen zu ermöglichen. Die möglichen Einsatzgebiete sind dabei sehr vielfältig. So ergeben sich durch das Spiel-basierte Lernen beispielsweise neue Perspektiven für die Ergänzung vorhandener Bildungsangebote in der beruflichen Aus- und Weiterbildung.

Ein Kernpunkt des E-Learning ist die Sicherstellung das Lerninhalte wiederverwendbar und auch auf anderen Systemen und Plattformen nutzbar sind. Als entsprechender Standard hat sich inzwischen das *Sharable Content Object Reference Model* (SCORM) der *Advanced Distributed Learning Initiative* (ADL) etabliert. Ebenso wie für klassische Lernprogramme ist

die Konformität zu diesem Standard auch für Spiel-basierte Lernprogramme wünschenswert.

Die Integration von Spiel-basierten Lernprogrammen in vorhandene nach SCORM spezifizierte Lernumgebungen unter besonderer Beachtung des Multi-User Aspektes ist Gegenstand dieser Arbeit. Sie entsteht in Zusammenarbeit mit der Abteilung „Multimediale Kommunikation“ des Fraunhofer Instituts für Graphische Datenverarbeitung in Rostock. Die Abteilung beschäftigt sich im Bereich E-Learning unter anderem mit neuen Konzepten für die Vermittlung von Lerninhalten. Ziel der Arbeit ist es dem Leser das Spiel-basierte Lernen als Alternative zu klassischen elektronisch unterstützten Lernformen näher zu bringen und Lösungsansätze für eine Integration aufzuzeigen.

Der erste Teil der Arbeit (Einführung) erläutert die Grundlagen der Themenbereiche E-Learning und Spiel-basiertes Lernen. Dabei werden im ersten Kapitel wichtige Begriffe geklärt und die SCORM Spezifikation vorgestellt. Das zweite Kapitel beschäftigt sich danach vor allem mit den didaktischen Aspekten und der Klassifizierung von Spiel-basierten Lernprogrammen.

Das dritte Kapitel enthält eine Analyse der Anforderungen der SCORM Spezifikation an die am Lernprozess beteiligten Objekte sowie der Anforderungen Spiel-basierter Lernprogramme. Die Analyse ist die Grundlage für das vierte Kapitel, indem die verschiedenen Möglichkeiten der Integration eines Spiel-basierten Lernprogramms in eine SCORM-konforme Lernumgebung vorgestellt und miteinander verglichen werden. Beide Kapitel bilden zusammen den zweiten Teil der Arbeit (Konzeption) und beachten besonders den möglichen Multi-User Aspekt einiger Spiel-basierter Lernprogramme.

Den dritten und damit letzten Teil der Arbeit (Umsetzung) bildet die prototypische Implementierung, einer der in der Konzeption vorgestellten Lösungsansätze. Als Anwendungsbeispiel wird dabei das Spiel „Drei/Vier/Fünf Gewinnt“ in eine, von der ADL bereitgestellte, beispielhafte Lernumgebung integriert.

Einführung

1 E-Learning

1.1 Allgemeines

In diesem Kapitel wird zunächst erläutert, was E-Learning eigentlich ist und welche Begriffe dem zuzuordnen sind. Nach der Klärung der Begriffe E-Learning, Lernobjekt und Lernumgebung werden die zugrunde liegenden Lerntheorien betrachtet und anschließend die SCORM Spezifikation vorgestellt. Dieses Kapitel bildet zusammen mit dem zweiten Kapitel die theoretische Grundlage für die gesamte Arbeit.

1.2 Begriffsklärung

1.2.1 E-Learning

Der Begriff E-Learning ist ein Kunstwort aus dem Englischen und wird in der Regel als Überbegriff für elektronisch unterstütztes Lernen verwendet. Bei genauerer Betrachtung ist E-Learning nur einer der zwei Bereiche, der sogenannten E-Education. Den anderen Bereich beschreibt der Begriff E-Teaching [Baumgartner02]. Elektronisch unterstütztes Lernen bedeutet in diesem Zusammenhang Software-basiertes Lernen unter Nutzung digitaler Medien und Webtechnologien.

E-Learning entwickelte sich aus dem computergestützten Lernen, auch Computer-Based Training (CBT) genannt. Der Lernende bearbeitet dabei selbstständig, die über den Computer bereitgestellten, Lerninhalte. Der Computer begleitet den Lernprozess und bietet Möglichkeiten zur Interaktion und Wissensüberprüfung. Begünstigt durch die schnelle Ausbreitung des Internets entstanden daraus immer mehr netzgestützte Formen, auch Web Based Training genannt. Beiden Lernformen wurden schließlich durch synchrone und asynchrone Kommunikationsmöglichkeiten ergänzt. E-Learning ist also nicht nur elektronisch unterstütztes Lernen, sondern beinhaltet auch die Kommunikation mit anderen Lernenden und Tutoren.

1.2.2 Lernobjekt

Ein Lernobjekt bildet im Bereich E-Learning die kleinste, inhaltlich sinnvolle, Einheit. Sie kann beispielsweise aus Texten, Bildern oder Animationen zusammengesetzt sein, genauso gut kann aber auch eine einzelne Animation ein eigenständiges Lernobjekt bilden. Die Lernobjekte können dann beliebig zu einer Lerneinheit (z.B. Kurs) zusammengesetzt werden.

Um die Wiederverwendung der Lernobjekte zu ermöglichen, versieht man diese mit Metadaten und spricht dann von sogenannten *Reusable Learning Objects*.

1.2.3 Lernumgebung

Der Begriff Lernumgebung beschreibt „umgangssprachlich die räumlichen, zeitlichen, personellen und instrumentellen Merkmale einer konkreten Situation, in die ein Lernprozess eingebettet ist.“ [Glossar06]. Im E-Learning Bereich steht der Begriff für die Benutzerumgebung zur Organisation und Durchführung des elektronisch unterstützten Lernens und beinhaltet nach [Baumgartner02] die folgenden fünf grundlegenden Funktionsbereiche:

- Präsentation von Inhalten
- Evaluations- und Bewertungshilfen
- Werkzeuge zur Erstellung von Aufgaben und Übungen
- Administration
- Kommunikationswerkzeug

Die technische Umsetzung des eher abstrakten Begriffes der Lernumgebung in Form einer meist auf einem zentralen Server installierten Softwareschnittstelle bezeichnet man als *Learning Management System* (LMS). Hauptbestandteile sind die Präsentation und Administration der gespeicherten Inhalte und Ergebnisse sowie eine Nutzerverwaltung. Das LMS ermöglicht beispielsweise auch das Anlegen von Lernprofilen, um so den Lernenden individuelle Lerninhalte zur Verfügung zu stellen. Um die Wiederverwendbarkeit der Lerninhalte bei der Vielzahl vorhandener LMS zu gewährleisten, wurde für deren Einsatz ein einheitlicher Standard anhand der SCORM Spezifikation definiert. Dieser wird in Abschnitt 1.4 vorgestellt. Inzwischen ist auch der Begriff *Learning Content Management System* (LCMS) weit verbreitet. Diese Systeme kombinieren die Funktionen eines LMS mit denen eines Content Management Systems, welche der Erstellung, Bearbeitung und Verwaltung von Inhalten dienen. LCMS haben also neben den bereits erläuterten Funktionen eines LMS auch die Aufgabe, die aus Lernobjekten zusammengesetzten Lerneinheiten zu organisieren. Auch dabei spielt die Wiederverwendung der einzelnen Lernobjekte eine große Rolle.

Die Bezeichnungen LMS und LCMS werden in der Literatur oft synonym eingesetzt und auch in dieser Arbeit so verwendet.

1.3 Lerntheorien

Entscheidend für den erfolgreichen Einsatz eines E-Learning Angebots ist, neben der technischen Umsetzung, vor allem die didaktische Grundlage. [Meschenmoser02] stellt allerdings fest, „dass es keine universale Theorie des Lernens aller möglichen Verhaltensbereiche gibt, sondern eine Vielzahl sehr unterschiedlicher, sich teilweise ergänzender, aber auch widersprechender Konzepte.“ Zu den Bekanntesten gehören die behavioristische und kognitivistische Lerntheorie sowie die konstruktivistische Erkenntnistheorie. Auch wenn die Schwerpunkte der Arbeit eher die technische Aspekte sind, werden die Lerntheorien in den folgenden Abschnitten kurz vorgestellt.

1.3.1 Behaviorismus – Lernen durch Erfolg und Belohnung

Die älteste Lerntheorie hat ihren Ursprung in den Untersuchungen von I.P. Pawlow zur Konditionierung und zum bedingten Reflex. Spätere Untersuchungen von E.L. Thorndike und B.F. Skinner begründeten die theoretische Grundlage des Behaviorismus und näherten sich immer mehr dem Ziel des automatisierten Lernprozesses an, welcher „Lernen als Verknüpfen von Reiz und Reaktion versteht. Durch Belohnung im Falle des Auftretens der gewünschten Reaktion kann die Auftretenswahrscheinlichkeit erhöht werden, durch Strafe kann sie vermindert werden.“ [Dittler03]. Anwendung findet die Theorie auch noch in vielen interaktiven Lernprogrammen, die ihren Schwerpunkt auf die Ergebniskontrolle legen und dabei individuelle Interessen und Fähigkeiten des Lernenden außer Acht lassen.

1.3.2 Kognitivismus – Verstehen und Einsicht

Anders als bei der behavioristischen Theorie rückten beim Kognitivismus die Lernenden und deren Eigenaktivität zur Verarbeitung von Erfahrungen in den Mittelpunkt. Wichtigstes Anliegen ist „der Aufbau des Verständnisses für ein Problem und damit der Aufbau von Problemlösekompetenz, die es dem Lernenden ermöglicht, sich die Lösung eines Problems selbstständig zu erarbeiten.“ [Dittler03]. Der Kognitivismus schafft damit die Grundlage für ein bewusstes, entdeckendes und kreatives Lernen, welches den Lernenden maßgeblich in den Lernprozess mit einbezieht.

1.3.3 Konstruktivismus – Konstruktion von Wissen

Sowohl in behavioristisch als auch kognitivistisch orientierten Lernumgebungen kann man feststellen, dass der Erfolg der Lernenden unter den gleichen Bedingungen sehr

unterschiedlich ausfällt [Dittler03]. Es reicht demnach nicht aus den Lernenden in den Lernprozess einzubeziehen, sondern der Lernprozess muss direkt auf den Lernenden eingehen. Der Konstruktivismus bezieht aus diesem Grund bereits vorhandenes Wissen des Lernenden stärker in den Lernprozess ein und versteht Lernen als Konstruktion von Wissen auf der Basis des vorhandenen Erfahrungs- und Wissensschatzes [Meschenmoser02]. Der Lernprozess unterliegt keinerlei Richtlinien und wird nur durch den Lernenden selbst gesteuert.

1.3.4 Ausblick

Wichtigstes Anliegen von E-Learning Angeboten ist der Transfer des erworbenen Wissens in alltägliche oder berufliche Anwendungssituationen. Lernumgebungen müssen dafür einerseits die Möglichkeit bieten Wissen zu konstruieren und andererseits auch den Lernprozess gezielt steuern. Erreicht werden kann dieses Ziel am besten durch eine ausgeglichene Kombination der verschiedenen Lerntheorien. [Meschenmoser02] stellt dazu fest:

> Konsequente individualisierte Förderung unter Ausschöpfung der Potenziale interaktiver Medien muss deshalb die kooperative Arbeit am gemeinsamen Lerngegenstand unterstützen, isoliertes (stilles) Lernen ist möglichst zu überwinden und durch kommunikatives, sinnstiftendes und zielorientiertes Lernen zu ersetzen.

Gerade neue E-Learning Konzepte unter Nutzung multimedialer Elemente und neuer Interaktionsformen bieten in diesem Zusammenhang eine viel versprechende Alternative, wie Kapitel 2 zum Spiel-basierten Lernen zeigt.

1.4 Sharable Content Object Reference Model

SCORM ist kein Standard im eigentlichen Sinne, sondern eine Sammlung von Richtlinien und Standards aus dem E-Learning Bereich. Als Referenzmodell zeigt die SCORM Spezifikation Wege zur Problemlösung bei der Erstellung von Lernangeboten auf. Die Spezifikation erläutert die betreffenden Standards und zeigt wie man diese zusammenführt und benutzt.

Verantwortlich für SCORM ist die im November 1997 gegründete ADL Initiative. Die Initiative beteiligt sich an neuen technischen Ideen und Konzepten, integriert und testet die daraus entstehenden Spezifikationen und Standards und schließt so die Lücke zwischen Entwicklungsarbeit und Anerkennung zum Industriestandard. Zu den wichtigsten

Organisationen, Initiativen und Projekten mit denen die ADL Initiative dabei zusammenarbeitet gehören die *Alliance of Remote Instructional Authoring & Distribution Networks for Europe* (ARIADNE), das *Aviation Industry CBT Committee* (AICC), das *Institute of Electrical and Electronics Engineers* (IEEE) *Learning Technology Standards Committee* und das *Instructional Management Systems* (IMS) *Global Learning Consortium*. ADL beschreibt die wichtigsten drei Erfolgskriterien des Referenzmodells. Erstens muss das Modell Richtlinien enthalten, die von den Entwicklern der Lernangebote verstanden und implementiert werden können. Zweitens muss es von einer möglichst großen Menge von Nutzern innerhalb der Zielgruppe (Entwickler von Lernangeboten und Werkzeugen sowie deren Kunden) akzeptiert und benutzt werden. Drittens muss es auch spezifische Instruktionsmodelle der Nutzer abbilden und widerspiegeln können.

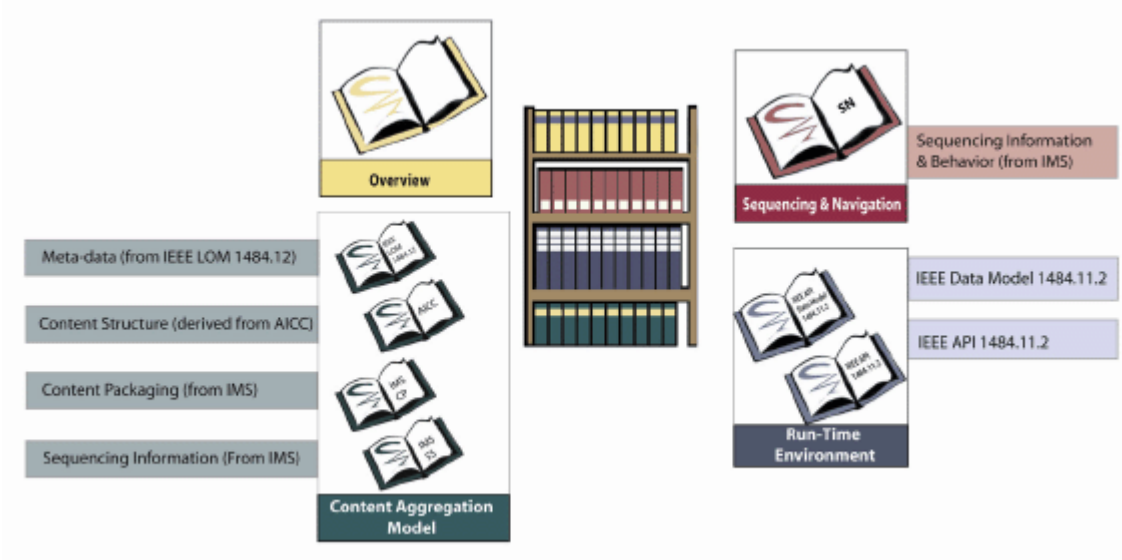

Abbildung 1.1: Bestandteile von SCORM [ADL06a]

Die SCORM Spezifikation hat sich inzwischen als de facto Standard etabliert und liegt aktuell in der Version 2004 3rd Edition vor, welche aus vier Teilen oder Büchern (vgl. Abbildung 1.1) besteht, die im Folgenden vorgestellt werden.

1.4.1 Overview

Der erste Teil der SCORM Spezifikation erläutert zunächst die Geschichte und die Motivation der ADL Initiative. Demnach ist die Vision der Initiative das Anbieten von qualitativ hochwertiger Unterstützung bei der Erstellung und technischen Umsetzung von Lernangeboten, die an individuelle Bedürfnisse angepasst und kostengünstig jederzeit und

überall hin verbreitet werden können. Die Entwicklung von Lernangeboten soll somit beschleunigt und gleichzeitig der Markt für solche Systeme und die entsprechende Software angeregt werden.

Das Dokument führt den Leser in die SCORM Spezifikation ein und erläutert deren Konzepte. Die Basis für alle Konzepte bilden die sogenannten *ADL-ilities*. ADL bezeichnet mit diesem Begriff die als grundlegend angesehenen Eigenschaften für die in Lernangeboten enthaltenen und somit am Lernprozess beteiligten Objekte. Zu den ADL-ilities gehören folgende Eigenschaften [ADL06a]:

- *Accessibility:* Fähigkeit Objekte unabhängig von deren Ort zu finden und weiter zu verwenden.
- *Adaptability:* Anpassbarkeit der Objekte an individuelle und organisatorische Bedürfnisse.
- *Affordability:* Erhöhung der Effizienz und Produktivität des Angebots und Senkung des Zeit- und Kostenaufwandes.
- *Durability:* Fähigkeit technologischen Weiterentwicklungen ohne kostenintensive Umgestaltung, Neukonfiguration oder Umkodierung Stand zu halten.
- *Interoperability:* Nutzung der Objekte auf unterschiedlichen Plattformen und unabhängig von verwendeten Werkzeugen.
- *Reusability:* Möglichkeit die Objekte in verschiedensten Anwendungen und Zusammenhängen zu nutzen.

Die ADL-ilities bilden auch die Grundlage für zukünftige Änderungen und Erweiterungen der SCORM Spezifikation. Zusätzlich, zu diesen Eigenschaften, ist die sogenannte *Web-based assumption* ein grundlegender Bestandteil der Spezifikation. ADL weist damit auf die Chancen zur Erhöhung der Zugänglichkeit und der Wiederverwendbarkeit von Lerninhalten durch die Nutzung von Internettechnologien hin. Aus der Kombination dieser grundlegenden Konzepte ergibt sich schließlich die Leistungsfähigkeit einer Lernumgebung auf Grundlage der SCORM Spezifikation.

Weiterhin enthält das Dokument eine Zusammenfassung der Bestandteile der SCORM 2004 3rd Edition und verdeutlicht deren Verbindung sowie die Änderungen gegenüber den Vorgängerversionen. Am Ende des Dokuments findet man außerdem eine kurze Beschreibung

der SCORM Konformitätsvoraussetzungen (vgl. Abschnitt 3.2). Das vollständige SCORM 2004 3rd Edition Overview Dokument befindet sich auf der, dieser Arbeit, beiliegenden CD.

1.4.2 Content Aggregation Model

Das SCORM *Content Aggregation Model* (CAM) Buch beschäftigt sich mit der Strukturierung und Zusammenstellung von Lerninhalten in wiederverwendbare Pakete und erläutert wie man diese mit Metadaten beschreiben kann. Es zeigt außerdem wie Informationen zur Ablaufsteuerung definiert werden.

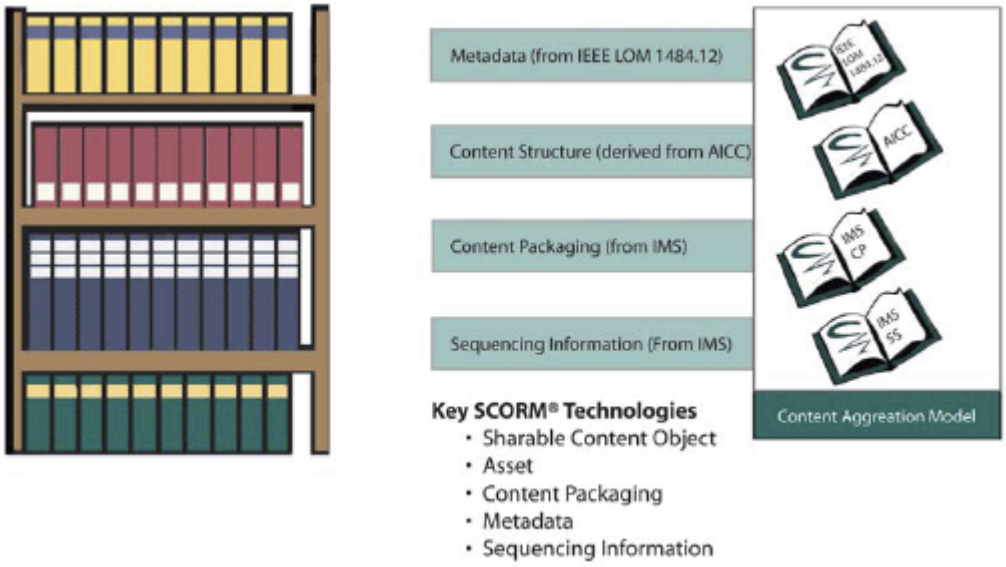

Abbildung 1.2: Das Content Aggregation Model Buch [ADL06a]

Das Modell ermöglicht so das konsistente Speichern, Beschreiben, Packen, Austauschen und Finden von Lerninhalten und ist in vier Teile gegliedert: *Content Model*, *Content Packaging*, *Metadata* und *Sequencing and Presentation*. Eine Übersicht über die verwendeten Standards und Technologien bietet Abbildung 1.2.

Content Model

Das Content Model beschreibt die grundlegenden Objekte von SCORM und erläutert wie man mit deren Hilfe komplexe Lerninhalte zusammensetzt. Zu den Objekten gehören auf der einen Seite *Assets* und *Sharable Content Objects* (SCOs), bei denen es sich um Ressourcen handelt. Auf der anderen Seite werden auch sogenannte *Activities*, die *Content Organization* und die *Content Aggregation* beschrieben, die der Strukturierung der Lerninhalte dienen [ADL06a]. Abbildung 1.3 verdeutlicht die Zusammenhänge.

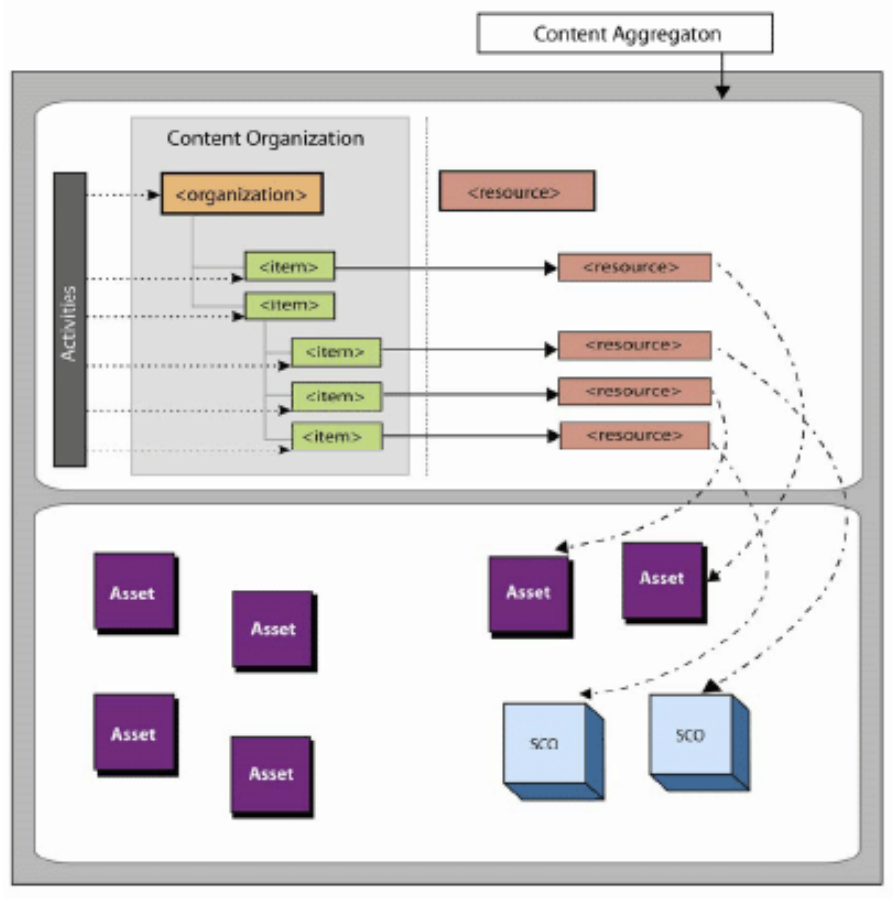

Abbildung 1.3: Content Model [ADL06a]

Bei einem Asset kann es sich um Audio, Video, Text, Bilder oder eine andere webfähige Ressource handeln. Assets können auch wieder zu einem neuen Asset zusammengesetzt werden. Ein *Sharable Content Object* (SCO) setzt sich aus Assets zusammen und bildet eine abgeschlossene wiederverwendbare Ressource, die direkt aufgerufen wird und im Unterschied zum Asset auch mit dem LMS kommunizieren kann. Dazu muss das SCO als Voraussetzung Methoden zum Intialisieren und Beenden der Kommunikation unterstützen, die durch die Laufzeitumgebung (vgl. Abschnitt 1.4.3) bereitgestellt werden.

Activities sind vorstrukturierte Lerneinheiten, die dem Lernenden Ressourcen zur Verfügung stellen oder weitere Lerneinheiten enthalten. Somit lassen sich beispielsweise hierarchische Strukturelemente, wie Kurse, Kapitel oder Module, realisieren. Activities, die wiederum aus Activities bestehen, bezeichnet man als Cluster. Abgebildet werden die Activities durch die Content Organization, während Content Aggregation einerseits den Prozess der Zusammenstellung der Inhalte und andererseits die Zusammenstellung an sich bezeichnet.

Content Packaging

Das Content Packaging bezeichnet den Prozess zum Austausch der Lerninhalte zwischen verschiedenen Systemen. Alle physischen Dateien und eine Manifest Datei werden dabei in eine standardisierte Struktur gepackt. Dieses *Content Package* (CP) bildet eine abgeschlossene systemunabhängige Lerneinheit. Das Manifest wird in Form eines *Extensible Markup Language* (XML) Dokuments im Wurzelverzeichnis des Package abgelegt. Es enthält alle notwendigen Informationen zur Struktur und zu den Ressourcen aufgeteilt in vier Abschnitte: Metadata, Organizations, Ressources und (sub)Manifest(s). Die komprimierte Archivedatei, die alle Bestandteile des Packages enthält, bezeichnet man als *Package Interchange File* (PIF). Zur Sicherung der Interoperabilität empfiehlt ADL die Verwendung des Formats PKZip v2.04g.

Metadata

Metadaten dienen der Beschreibung der einzelnen Komponenten des Content Models und basieren auf den IEEE Standards 1484.12.1-2002 Learning Object Metadata und 1484.12.3 Standard for Extensible Markup Language Binding for Learning Object Metadata Data Model. Dabei stehen 64 standardisierte Elemente zur Beschreibung der Komponenten zur Verfügung.

Sequencing and Presentation

Dieser Teil beschreibt die Möglichkeiten Regeln zur Steuerung des Ablaufs und zur Präsentation der Inhalte anzulegen und im Manifest umzusetzen. Die Sequencing Informationen werden in der Manifest Datei abgelegt und den entsprechenden Activities zugeordnet. Das vollständige SCORM 2004 3rd Edition Content Aggregation Model Dokument befindet sich auf der, dieser Arbeit, beiliegenden CD.

1.4.3 Run-Time Environment

Das Hauptziel der SCORM Spezifikation ist die Wiederverwendbarkeit der Lerninhalte unabhängig vom verwendeten LMS zu sichern (vgl. Abschnitt 1.4.1). Die Voraussetzungen um dieses Ziel zu erreichen, sind die Verwendung einheitlicher Methoden um Lerninhalte zu laden und zu verwalten sowie eine einheitliche Kommunikation zwischen LMS und den Lernobjekten, basierend auf einem gemeinsamen Datenmodell.

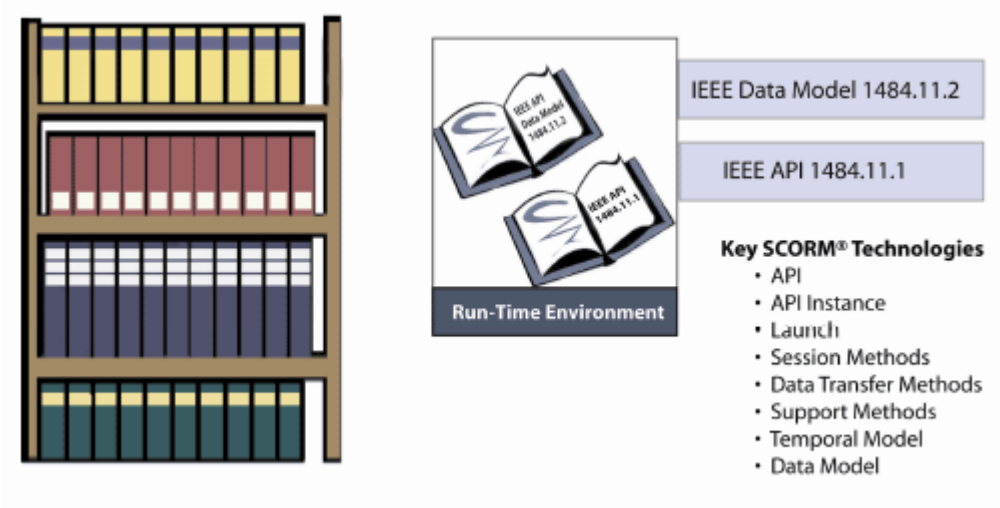

Abbildung 1.4: Das Run-Time Environment Buch [ADL06a]

Die Umsetzung dieser Voraussetzungen wird in dem SCORM *Run-Time Environment* (RTE) Buch behandelt und gliedert sich in drei Teile: *Launch*, *Application Programming Interface* (API) und *Data Model* [ADL06a]. Launch bezeichnet den Prozess zum Laden der webbasierten Lernobjekte. Die API bildet die Schnittstelle für die Kommunikation mit dem LMS und Data Model definiert die dazu notwendigen Elemente. Eine Übersicht über die verwendeten Standards und Technologien bietet Abbildung 1.4. Das vollständige SCORM 2004 3rd Edition Run-Time Environment Dokument befindet sich auf der, dieser Arbeit, beiliegenden CD.

1.4.4 Sequencing and Navigation

Das SCORM *Sequencing and Navigation* (SN) Buch definiert Methoden zur Steuerung der Lerninhalte. Es beschreibt die zur Laufzeit notwendige Funktionalität des LMS und der Lernobjekte. Eine Übersicht über die verwendeten Standards und Technologien bietet Abbildung 1.5.

Der Sequencingteil des Buches basiert auf der IMS *Simple Sequencing* Spezifikation und beschreibt die Umsetzung und Erweiterung dieser Spezifikation in SCORM. Benötigt wird dazu eine festgelegte Struktur der Activities (*Activity Tree*), eine entsprechende Strategie (*Sequencing Definition Model*) und ein definiertes Verhalten (*Sequencing Behaviors*). Das Sequencing bezieht sich immer nur auf Activities. Dadurch sind die Ressourcen unabhängig vom Sequencing wiederverwendbar. Das LMS muss die in den Activities verwendeten Ressourcen laden und ist somit verantwortlich für die Einhaltung des definierten Ablaufs.

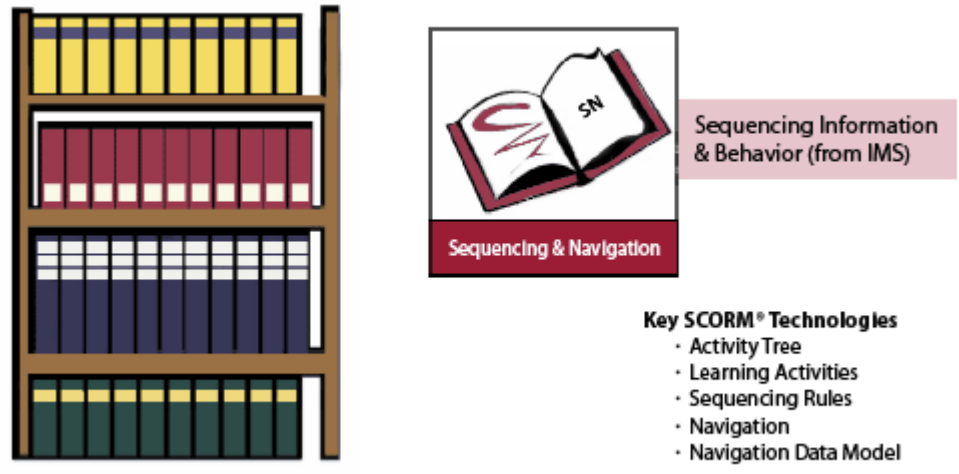

Abbildung 1.5: Das Sequencing and Navigation Buch [ADL06a]

Der Navigationteil beschreibt die Möglichkeiten zur Steuerung der Activities gemäß den Nutzereingaben und der zuvor festgelegten Ablaufsteuerung. Das sogenannte *Navigation Model*, definiert die Menge von Ereignissen, die durch das LMS oder das Lernobjekt selbst ausgelöst werden können. Dazu gehört beispielsweise ein Startereignis, welches den Anfangspunkt der Activity anfordert. Generell werden von SCORM keinerlei Anforderungen an das Aussehen der Nutzerschnittstelle und die Navigation gestellt, sondern lediglich Empfehlungen ausgesprochen. Das vollständige SCORM 2004 3rd Edition Sequencing and Navigation Dokument befindet sich auf der, dieser Arbeit, beiliegenden CD.

2 Spiel-basiertes Lernen

2.1 Allgemeines

Wie Spiel-basiertes Lernen charakterisiert werden kann, soll in diesem Kapitel näher untersucht werden. Den Schwerpunkt des Kapitels bilden, nach einer Klärung der Begriffe Spiel, Computerspiel, digitales Lernspiel und Simulation, die didaktischen Aspekte des Spiel-basierten Lernens, vor allem im Hinblick auf motivierende Elemente, Interaktion, Lernkonzepte und Wissenstransfer. Dieses Kapitel bildet zusammen mit dem ersten Kapitel die theoretische Grundlage für die gesamte Arbeit.

2.2 Begriffsklärung

2.2.1 Spiel / Computerspiel

Der Begriff Spiel bezeichnet im Allgemeinen eine Tätigkeit, die nicht zweckgebunden sondern lediglich zum persönlichen Vergnügen ausgeführt wird. Spiele sind nach [Meier03] gekennzeichnet durch eine dem Spiel zugrunde liegende Spielidee, die den Handlungsrahmen vorgibt und für den Spielspaß bzw. die Motivation verantwortlich ist, sowie ein festgelegtes Regelwerk, welches einem oder mehreren Spielern ermöglicht den Ablauf des Spiels zu steuern. Aspekte, die bei Spielen einen wichtigen Stellenwert haben, sind nach [Prensky01] festgelegte Ziele, Interaktion, Problemlösung (Konfliktbewältigung, Wettkampf, Herausforderung, Gegner) sowie die daraus resultierenden Ergebnisse und eine entsprechende Rückmeldung. [Ho06] führen des Weiteren die Elemente Fantasie und Unterhaltung als wichtige Elemente eines Spiels ein und erläutern das Prinzip der simulierten Sicherheit. Damit ist in diesem Zusammenhang die Möglichkeit des Spielers, innerhalb des Spiels frei und ohne Konsequenzen zu handeln, gemeint.

Werden Spiele als Computerprogramme umgesetzt, spricht man von Computerspielen. Das eigentliche Spielgeschehen wird auf dem Bildschirm dargestellt und der Spieler hat die Möglichkeit sich, durch die ihm zur Verfügung stehenden Eingabegeräte (Tastatur, Maus, Joystick), aktiv daran zu beteiligen [Müsgens00]. Die dabei verwendeten Endgeräte können entweder Computer, Spielkonsolen oder auch Mobiltelefone sein.

Gefördert durch die zunehmende Verbreitung von Computern und deren rasanter technischer Entwicklung, konnten Computerspiele in den letzten Jahren vor allem bei Kindern und

Jugendlichen, aber auch bei Erwachsenen klassische Spiele an Faszination überbieten. [Prensky01] führt unter anderem folgende Gründe an: Computerspiele bieten dem Spieler schon allein durch die besseren graphischen Fähigkeiten eine größere Bandbreite möglicher Szenarien, die so bei klassischen Spielen nicht möglich wären. Zudem können Computerspiele wesentlich komplexere Handlungen und Inhalte umsetzen. Auch die Anpassungs- und Nutzungsmöglichkeiten werden beispielsweise durch einstellbare Schwierigkeitsgrade sowie die Option jederzeit gegen andere Spieler oder den Computer selbst zu spielen, unabhängig von Zeit und Ort, erhöht. Man unterscheidet im Allgemeinen acht Genres bei Computerspielen: Action, Abenteuer, Geschicklichkeit, Kampf, Rollenspiel, Simulation, Sport und Strategie.

2.2.2 Simulation

Spiele, insbesondere Computerspiele, sind eng mit Simulationen verbunden oder bestehen ganz und gar aus ihnen. Deshalb fällt es schwer die Begrifflichkeiten zu trennen. Nach [Dittler96] ist das zentrale Gestaltungselement einer Simulation stets eine gewisse Nähe zur Realität. Sie setzt Ausschnitte daraus modellhaft um und ermöglicht, die Gesetzmäßigkeiten innerhalb des Ausschnitts zu kontrollieren und gezielt zu beeinflussen. Der Computer eröffnet hier nahezu unbegrenzte Möglichkeiten. Er dient einerseits zur Darstellung und andererseits zur Interaktion mit dem Modell. So werden in Computerspielen simulierte Spielwelten nicht nur wahrnehmbar, sondern auch für Handlungen des Spielers zugänglich [Fromme01].

Eigenständige Simulationen oder Simulationen als Teil von Spielen werden auch in der Bildung eingesetzt. „Simulationen können in vielen Bereichen sehr erfolgreich zu Lernzwecken eingesetzt werden. Sie unterstützen eine aktive entdeckende und selbstgesteuerte Vorgehensweise und können bei guter Gestaltung motivierend wirken“ [Blumstengel98]. Der Lernende kann somit sein erlerntes Wissen anwenden, und beobachten wie sich sein Handeln in dem vorliegenden Modell auswirkt. Eine Reflexion des Modellcharakters durch den Lernenden ist dennoch sehr wichtig, denn Simulationen bilden die Realität fast immer reduziert auf einige Aspekte ab.

Eine mögliche Klassifizierung nehmen [Diener06] vor. Sie unterscheiden Simulationen anhand ihres Einsatzgebietes und des Interaktivitätsgrades in Charaktersimulation, interaktive Trainingssimulation, Umgebungssimulation und demonstrative Simulation. In der praktischen Anwendung kommt meist eine Mischung der Ansätze zum Einsatz.

2.2.3 Digitales Lernspiel

Betrachtet man nun Spiele und klassische Lernprogramme für den Computer, so scheint die Verbindung beider zu einem Lernspiel recht schwierig. Schon die Zielsetzung unterscheidet sich nach [Dittler96] deutlich. Während Computerspiele auf spannende Unterhaltung in einem in sich geschlossenen digitalen Erlebnisraum ohne Verbindung zur Außenwelt abzielen, sollen Lernprogramme Wissensinhalte in konkrete Anwendungssituationen transferieren.

Unabhängig von der Art des Spiels ist es allerdings nicht von der Hand zu weisen, dass beim Spielen gewisse Lerneffekte festgestellt werden können. Betrachtet man noch einmal die Definition von Spielen (vgl. Abschnitt 2.2.1), ist dieser Umstand wohl eher unbeabsichtigt bzw. teilweise sogar unerwünscht und steht außerdem für den Spieler nicht im Vordergrund. Nichtsdestotrotz lassen sich diese Effekte auch für das Vermitteln und Festigen von Wissen und Fähigkeiten und damit zum Lernen nutzen. Lernspiele sind also Tätigkeiten „[...], deren Inhalte, Struktur und Ablauf in pädagogischer Absicht und auf der Grundlage didaktischer Prinzipien gestaltet sind, die zugleich aber zentrale Merkmale von Spielen enthalten“ [Meier03]. Unterhaltung und Bildung werden so miteinander verbunden und das Spiel als Methode des Lernens eingesetzt. „Durch spielerische Elemente können Inhalte veranschaulicht, Wissen unterhaltsam vermittelt und der Erwerb von Fähigkeiten und Fertigkeiten motivierend unterstützt werden.“ [Meschenmoser02].

Hinterfragen sollte man inwiefern sich Lerninhalte überhaupt für die Umsetzung als Lernspiel eignen und in welchem Verhältnis die aufgewendete Zeit zum gewünschten Lernerfolg steht. So ist der Einsatz von Lernspielen „oft erst sinnvoll, wenn bereits Grundlagenwissen vorliegt, um komplexe Zusammenhänge verstehen und steuern zu können.“ [Blumstengel98]. Daneben ist für das Lösen der Probleme innerhalb des Spiels vor allem logisches Denken und planvolles Handeln gefragt [Meschenmoser02]. Vorgefertigte Standardlösungen für die Umsetzung digitaler Lernspiele gibt es leider nicht. Berücksichtigen sollte man auf jeden Fall die Vorstellungen und Wünsche der Zielgruppe, die Besonderheiten des jeweiligen Fachgebietes sowie die Nutzungsmöglichkeiten der vorhandenen Ressourcen und einsetzbaren Technologien.

Der Begriff digitales Lernspiel wird in der Literatur oft synonym für ein Spiel-basiertes Lernprogramm oder im Zusammenhang mit den englischen Begriffe *Game-based learning* bzw. *Edutainment* eingesetzt und auch in dieser Arbeit so verwendet.

2.3 Didaktische Aspekte

Die Anlehnung digitaler Lernspiele an das didaktische Design von Computerspielen hat grundlegenden Einfluss auf den gesamten Lernprozess. Es bleibt also zu klären, welche Elemente in Computerspielen motivieren, welche Lernkonzepte digitalen Lernspielen zugrunde liegen, welche Wissenstransfers stattfinden und welche Möglichkeiten der Interaktion bestehen.

2.3.1 Motivierende Elemente

Computerspiele unterscheiden sich von anderen Medien in ihrer technischen, gestalterischen und inhaltlichen Umsetzung. Dabei werden die neusten Technologien benutzt, um den Spielern eine realitätsnahe Spielwelt zu bieten, in der sie ohne Konsequenzen handeln können. Viele der Eigenschaften von Computerspielen, die Spieler am meisten ansprechen, dargestellt in Tabelle 2.1, lassen sich auch auf digitale Lernspiele übertragen. Sie bilden so den Grundstein für einen erfolgreichen Einsatz und eine hohe Motivation, die in diesem Zusammenhang „als wesentlicher Aspekt zur Aktivierung und Aufrechterhaltung des Lernprozesses [...] verstanden“ [Diener06] werden. Ist dieser Grundstein gelegt, unterstützt die Motivation nach [Becta01] die Ausdauer und das Vergnügen beim Lernen sowie das eigenständige Arbeiten und selbstständige Lösen von Problemen.

Tabelle 2.1: Eigenschaften von Computerspielen [Becta01]

technisch	erzählerisch	persönlich
Graphik	Neuheit	Logik
Klang	Handlung	Gedächtnis
Interaktivität	Neugier	Reflexe
	Komplexität	Mathematische Fähigkeiten
	Fantasie	Herausforderung
		Problemlösung
		Veranschaulichung

Eine weitergehende Unterscheidung der motivierenden Elemente eines Spiels treffen [Ho06]. Dabei werden die Elemente Herausforderung, Neugier, Kontrolle und Fantasie zur individuellen Motivation für jeden einzelnen Spieler. Die Elemente Wettkampf, Kooperation und Wahrnehmung dagegen dienen der zwischenmenschlichen Motivation aller Spieler. Ist

der Spieler schließlich motiviert, so besteht das Hauptziel des Spielers darin, alle Herausforderungen zu meistern und das Spiel letztendlich zu gewinnen. Die Art und Weise wie das Spiel diesen Prozess umsetzt und die aufkommenden Gefühle von Macht, Herrschaft und Kontrolle [Fritz06], die sich währenddessen einstellen, sind die eigentlichen unterhaltenden Faktoren und bilden die Hauptmotivation für den Spieler, das Spiel weiter zu spielen.

Vergisst der Spieler sich dabei selbst und taucht völlig in das Spiel ein, verschmelzen Handlung und Bewusstsein und man spricht vom sogenannten Flow-Erleben [Dittler96]. Dieser fließende Zustand des Handelns vermindert die Distanz zum Spiel deutlich und sorgt so für die große Faszination und die hohe intrinsische Motivation bei Computerspielen, wie [Becta01] verdeutlicht. Der Anreiz wird nicht durch externe Belohnungen gegeben, sondern liegt im Spieler selbst, gesteuert durch kognitive und affektive Prozesse. Hier liegt die Chance für Lernprogramme, denn eine „hohe Motivation des Lernenden ist [...] extrinsisch schwer zu erreichen“ [Diener06].

Einen weiteren Ansatz beschreiben [Fehr06]. Die Autoren sehen die Ursache für die hohe Motivation bei Computerspielen in dem Prozess der strukturellen Kopplung, der Spiel und Spieler verbindet. Computerspiele sind danach vor allem gekennzeichnet durch die drei Elemente: Präsentation, Inhalt und Regeln. Die richtige Kombination dieser Elemente ermöglicht erst die faszinierende Wirkung des Spiels, wobei durch den Spieler an jedes einzelne Element konkrete Erwartungen geknüpft sind. Stimmen in einem Bereich diese Erwartungen mit der Umsetzung überein, spricht man von einer strukturellen Kopplung in diesem Bereich, welche die Motivation des Spielers erhöht.

Ein weiterer wichtiger Aspekt zur Aufrechterhaltung der Motivation ist das Design der Benutzerschnittstelle. „Digitale Lernspiele sollten durch eine hohe Benutzerfreundlichkeit, einfache Navigation, einem einheitlichen Prinzipien folgendem Bildschirmdesign, eine klare Informationspräsentation und durch eine gefällige Ästhetik überzeugen.“ [Meier03].

2.3.2 Lernkonzept

Klassische Lernprogramme sollen in ihrer didaktischen Funktion Wissen bereitstellen und vermitteln, Möglichkeiten zum Anwenden und Üben bieten und die entsprechenden Resultate beurteilen. Nach [Dittler96] erfüllen die meisten Lernprogramme nur eine dieser Funktionen und können so unterschieden werden in Informationsprogramme, Übungsprogramme und

Rückmeldeprogramme. Sogenannte konstruktivistisch orientierte Lernprogramme (vgl. Abschnitt 1.3.3) hingegen erfüllen alle drei didaktischen Funktionen. Lernen wird „als aktive, vom Lernenden selbstständig durchzuführende Tätigkeit“ angesehen, wobei dieser „in einem kreativen Prozess sein Wissen aus den angebotenen Informationen“ [Glossar06] konstruiert und sich anschließend mit diesem Konstrukt und verschiedenen Problemsituationen auseinandersetzt. Zur Förderung der Vermittlung anwendbaren Wissens leitet [Dittler96] zusammenfassend folgende Möglichkeiten aus dem konstruktivistischen Ansatz ab:

> Lernen als *aktiven, konstruktiven Prozess* zu gestalten, den Wissenserwerb *selbst steuern* zu können, Wissen in der *aktiven Auseinandersetzung* mit einem Problem erwerben zu können, Wissen in einer *authentischen Problemsituation* erwerben zu können (Authentizität), das erworbene Wissen in *verschiedenen Problemsituationen* anwenden zu können (multiple Kontexte, multiple Perspektiven) und das neu erworbene Wissen in einer *sozialen Gruppe* überprüfen zu können.

In der Verbindung dieser Möglichkeiten mit den motivierenden technischen, gestalterischen und inhaltlichen Elementen von Computerspielen, vorgestellt in Abschnitt 2.3.1, sieht er die Grundlage für eine „konstruktivistisch orientierte, spielerisch gestaltete multi-mediale Lernumgebung“ [Dittler96]. Gerade digitale Lernspiele haben das Potential eine solche authentische Lernumgebung zur Verfügung zu stellen. Sie beziehen den Anwender in das Spielgeschehen ein und ermöglichen ihm Inhalte selbstständig zu erkunden, mit seinem konstruierten Wissen in Simulationsumgebungen zu experimentieren und das Lernen, Arbeiten und Spielen in einem sozialen Kontext zu verknüpfen.

Abbildung 2.1 verdeutlicht noch einmal die Unterschiede zu klassischen Lernprogrammen. Zum Vergleich wird hierbei ein Lernquiz und die virtuelle Lernwelt einer Lern- bzw. Spielegemeinschaft herangezogen. In der virtuellen Lernwelt werden Lernziele nicht klar definiert. Die geringe oder fehlende Strukturierung der Lerninhalte ermöglicht dem Lernenden die vollständige Kontrolle und Steuerung des Lernprozesses. Die Lerninhalte kann er sich selbstständig oder, durch die Integration kooperativer Lernbestandteile, innerhalb einer Lerngemeinschaft aneignen. Fehler werden nicht vermieden, sondern dienen dem Erfahrungslernen. Der Lehrer übernimmt hierbei keine klassisch korrigierende Rolle, sondern unterstützt und berät den Lernenden auf seinem Weg.

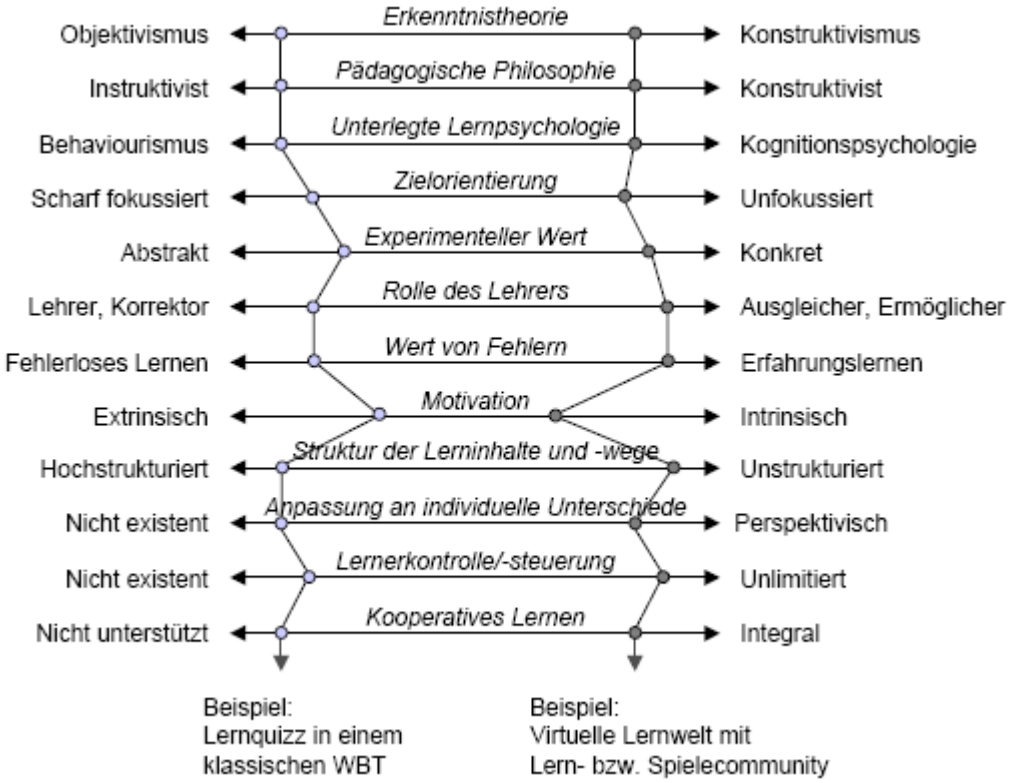

Abbildung 2.1: Gestaltung interaktiver Lernsysteme [Meier03]

2.3.3 Wissenstransfer

Einer der zentralen Aspekte beim Einsatz digitaler Lernspiele ist der Transfer, also die Übertragung oder Überführung, von erworbenem Wissen in den Alltag oder den Beruf. Man teilt den Wissenstransfer grob in intramondiale (innerhalb einer Welt) und intermondiale (von einer Welt in eine andere) Transfers. Zu beachten ist bei beiden die mit dem Transfer einhergehende Transformation. Dazu erläutern [Diener06]:

> Der Transfer von Wissen, Handlungsstrategien oder auch einzelnen Handlungsabläufen und Reaktionen in die Alltagswelt unterliegt immer einer Transformation. Das Wissen und die Strategien werden der Alltagswelt angepasst, sie werden aus dem Kontext des virtuellen Spiels in den Kontext der Alltagswelt übertragen und unterliegen anschließend den Gesetzmäßigkeiten und Bedingungen der Alltagswelt.

Ein Modell zur Untersuchung dieser Transformationen veranschaulicht [Fritz06] anhand von fünf Transferebenen, die sich in ihrem Abstraktionsgrad unterscheiden:

- *Fact-Ebene:* Beruhend auf eigenen authentischen Erfahrungen wird auf dieser Ebene konkretes inhaltliches Wissen ergänzt durch Informationen aus Medien aller Art (z.B. Bücher, Bilder, Filme), die stets kritisch geprüft werden müssen, übertragen.

- *Skript-Ebene:* Als Skripte bezeichnet man Ereignisabläufe mit vorgegebenen Verhaltensweisen für bestimmte Situationen (z.B. Kinobesuch). Sie bilden den Handlungsrahmen und können auch auf andere ähnliche Situationen angewendet werden.
- *Print-Ebene:* Einfache Handlungsmuster ohne konkreten inhaltlichen und sozialen Bezug (z.B. in die Hände klatschen) nennt man Print. Solche Elemente sind sehr oft Teil eines Skriptes, stehen aber nur für sich selbst und verweisen nicht auf eine Bedeutung.
- *metaphorische Ebene:* Hergeleitet von dem Begriff „Metapher" gehören zu dieser Ebene symbolisch-funktionale Übertragungen strukturell ähnlicher Handlungsmuster.
- *dynamische Ebene:* Losgelöst von inhaltlichen Bezügen und konkreten Handlungsimpulsen entsteht eine grundlegende Handlungsorientierung, reduziert auf Grundmuster des Handelns.

Beginnend bei der inhaltlich noch sehr konkreten Fact-Ebene steigt der Grad der Abstraktion von Ebene zu Ebene und der übertragene Inhalt entfernt sich vom Gegenständlichen. Neben der Einteilung in Ebenen ist eine Unterscheidung verschiedener Formen des Transfers möglich. [Müsgens00] unterscheidet folgende Formen:

> Beim emotionalen Transfer werden verschiedene Emotionen, wie Ärger, Angst, Anspannung, [...] transferiert. [...] Der instrumentell-handlungsorientierte Transfer beschreibt die größtenteils spielerischen Übertragungen [...]. Beim zeitlichen Transfer wird die Zeitstrukturierung der realen Welt in der virtuellen Welt aufgehoben [...] Bei realitätsstrukturierenden Transfers werden Erfahrungen im Spiel auf die Einschätzung und Bewertung der realen Welt bezogen.

[Fritz06] ergänzt noch weitere Formen des Transfers: den problemlösenden Transfer, den ethisch-moralischen Transfer, den assoziativen Transfer, den informationellen Transfer und auf das Gedächtnis oder die Fantasietätigkeit bezogene Transfers. Einige der Transferformen konnten bis jetzt allerdings noch nicht genauer untersucht und somit auch nicht belegt werden.

Für den Anwendungsbereich digitaler Lernspiele gilt also: Besonders die instrumentell-handlungsorientierten, realitätsstrukturierenden und informationellen Transfers auf Fact, Skript und Print Ebene sind entscheidend für die Vermittlung anwendbaren Wissens in den

Alltag oder Beruf und müssen dementsprechend näher untersucht und beachtet werden. Das daraus entstehende Potenzial beschreibt [Meschenmoser02] folgendermaßen:

> „Lernspiele können die Entwicklung und Anwendung komplexer Strategien (logisches, vernetztes, algorithmisches Denken) fördern und positive Wirkungen auf Wahrnehmungsfähigkeit, Formerfassung, Raumvorstellung, Konzentration und Reaktionsvermögen, Geschicklichkeit, Feinmotorik, Ausdauer, Belastbarkeit, Übersicht, Merkfähigkeit, Selbstständigkeit haben."

2.3.4 Interaktionsformen

Die Interaktion, also das wechselseitige Handeln des Spielers mit dem Spiel, ist ein wesentliches Kennzeichen von Computerspielen und ein entscheidender Aspekt für deren Anziehungskraft (vgl. Abschnitt 2.2.1). Beeinflusst nur der Spieler selbst das Spielgeschehen, so spricht man von Single-User Spielen. Der Computer berechnet dabei alle Reaktionen auf die Handlungen des Spielers. In Multi-User Spielen dagegen beteiligen sich mehrere Spieler am Spielgeschehen. Über vernetzte Computer besteht die Möglichkeit gegeneinander aber auch miteinander zu spielen, und so die eigenen Spielfähigkeiten direkt zu vergleichen. Verfügt man über einen Internetanschluss, kann man auch über einen Spieleserver mit anderen Spielern spielen. Die Begriffe „Single-User" und „Multi-User" werden oft synonym für Einzel- bzw. Mehrspieler oder die englischen Begriffe „Singleplayer" bzw. „Multiplayer" eingesetzt und auch in dieser Arbeit so verwendet.

Aus didaktischer Sicht ist vor allem die Multi-User Variante ein Ansatz für weitere Überlegungen, denn gerade für konstruktivistische Lernumgebungen ist das kooperative Lernen ein wichtiges Kriterium (vgl. Abschnitt 2.3.2). Kooperatives Lernen meint dabei entweder das gemeinsame Bearbeiten eines bestimmten Lerngegenstandes oder das Annähern an ein bestimmtes Lernergebnis durch kommunikativen Austausch [Breuer02]. Die Vorteile solcher Lerngemeinschaften erläutert [Wiebe01]:

> Wird der Wissensbildungsprozess gemeinsam geplant und durchgeführt, erhöht sich seine Effektivität. [...] Die kooperative Arbeit an spezifischen Aufgaben zwingt die Lernenden dazu, ihre Vorschläge den anderen zu erklären. Dadurch werden die z.T. unbewussten Vorstellungen konkretisiert und mit widersprechenden Alternativen verglichen. Dieser Prozess verbessert das Verständnis und die Lösungsmöglichkeiten der bearbeiteten Probleme. Unterstützung erfahren die Lernenden durch gegenseitige Kommentierung ihrer Vorstellungen. Positive Rückmeldungen verbessern die Motivation und [...] die Qualität der Wissensentwicklung.

Kooperatives Lernen erhöht nach [Breuer02] den Lernerfolg, verbessert die Lernleistung, fördert soziales Verhalten in der Gruppe und steigert sowohl die Motivation als auch die Selbsteinschätzung. Die Nutzung unterschiedlicher Interaktionsmöglichkeiten, insbesondere der Multi-User und Online-Computerspiele, sollte demnach auch für digitale Lernspiele stärker in Betracht gezogen werden. Das Thema der vorliegenden Arbeit wird auch aus diesem Grund unter Beachtung des Multi-User Aspektes bearbeitet.

2.4 Kategorien digitaler Lernspiele

Um die richtige Auswahl für ein konkretes Einsatzgebiet zu treffen, ist es notwendig digitale Lernspiele zu kategorisieren. Aus der Vielzahl möglicher Aspekte zur Kategorisierung erläutert [Prensky01] folgende Kriterien, die seiner Meinung nach schon beim Erstellen digitaler Lernspiele wichtige Entscheidungshilfen darstellen. Dabei betrachtet er zunächst die inhaltlichen Kriterien, die sich mit der Verknüpfung bzw. der Integration der Inhalte in das Spiel beschäftigen:

- *Intrinsische oder extrinsische Spiele* unterscheiden sich anhand des Zusammenhangs zwischen Inhalt und Spiel. Bei intrinsischen Spielen (z.B.: Flugsimulation) ist der Inhalt fester Bestandteil des Spiels, während bei extrinsischen Spielen (z.B.: Quiz) dieser Zusammenhang nur gering oder gar nicht vorhanden ist.
- Die Unterscheidung in g*ebundene oder aus Vorlagen erstellte Spiele* stellt die tatsächliche Integration des Lerninhalts in das Spiel und die Wiederverwendbarkeit in den Mittelpunkt. Gebundene Spiele sind speziell für einen festen Lerninhalt entworfen. Die Bindung kann allerdings unterschiedlich stark sein. Aus Vorlagen erstellte Spiele ermöglichen den Lerninhalt unabhängig vom Spiel jederzeit zu ändern, da diese erst zur Laufzeit in das Spiel geladen werden.

Weiterhin beschäftigt er sich mit didaktischen Kriterien zur Vermittlung der Inhalte:

- Ein wichtiger Aspekt für das Spiel-basierte Lernen ist die Frage inwieweit dem Spieler eine R*eflexion* des Spiels ermöglicht wird. Dabei ist darauf zu achten das richtige Verhältnis, bzw. das richtige Tempo zu finden. Zu wenig Zeit zum Reflektieren verringert den Lerneffekt, während zu viel Reflexion schnell für Langeweile sorgt.
- *Erzählerische oder Reaktionsspiele* unterscheiden sich in ihrem Spielansatz. Erzählerische Spiele basieren auf sehr komplexen Hintergrundinformationen und

bieten dem Spieler, neben einem größeren emotionalen Faktor, die Möglichkeit sich an der Geschichte zu orientieren. Reaktionsspiele, wie Quiz, verlangen dagegen schnelle Reaktionen des Spielers und verzichten auf unnötige Hintergrundinformationen.

Abschließend erläutert [Prensky01] einige technische bzw. gestalterische Kriterien:

- *Single-User oder Multi-User Spiele* verwenden als Kriterium die Fähigkeit eines Spiels, nur einen oder auch mehrere Spieler gleichzeitig über ein Netzwerk oder das Internet in das Spielgeschehen einzubeziehen (vgl. Abschnitt 2.3.4).
- *Echtzeit- oder Runden-basierte Spiele* unterscheiden sich in der Art und Weise, ob das Spiel den oder die Spieler synchron oder asynchron spielen lässt. Runden-basierte Spiele eignen sich vor allem dann, wenn Spieler aus Zeitgründen nicht gleichzeitig spielen können oder für einzelne Spielschritte mehr Zeit benötigen. Echtzeit-Spiele dagegen sind zum Beispiel für das Training im Team geeignet.
- *Sitzungs-basierte Spiele* werden von einem oder mehreren Spielern initiiert und existieren nur solange diese Spieler auch spielen, während *persistente Spiele* unabhängig von den Spielern immer weiter laufen. So kann der Spieler seine Fähigkeiten und Kenntnisse immer weiter ausbauen. Zu den persistenten Spielen gehören beispielsweise die sehr beliebten *Massively Multiplayer Online Games*, bei denen sehr viele Spieler gleichzeitig in einer virtuellen Welt über das Internet spielen.
- *Video- oder Animations-basierte Spiele* verwenden unterschiedliche Formen der graphischen Gestaltung. Realistische Videosequenzen stehen schnelleren und vielseitigeren Animationen gegenüber.

Eine andere Möglichkeit der Kategorisierung zeigen [Meier03], dargestellt in Tabelle 2.2. Dabei sind die Kriterien in erster Linie das Vorhandensein bzw. die Sichtbarkeit von Lernzielen, die vermittelbaren Inhalte und Kompetenzen sowie die Motivation. Das Spektrum reicht von eingebetteten spielerischen Elementen zur Wissensvermittlung über Lernspiele mit einem ausgeglichenen Verhältnis von Lernen und Unterhaltung bis zu reinen Unterhaltungsspielen, die auch zur Motivation eingesetzt werden können.

Tabelle 2.2: Typen digitaler Lernspiele, nach [Meier03]

	Motivation	Lernziele	vermittelbare Inhalte/Kompetenzen
Spielerische Elemente in Online Kursen (z.B. Quiz)	Wissensüberprüfung; unmittelbare Rückmeldung	klar definierte Lernziele (Aufgabe); didaktisch orientierter Aufbau	wissensorientierte Inhalte
Plan-/Rollenspiele	Lernkonzept basiert auf einer Simulation	klar definierte Lernziele; didaktisch orientierter Aufbau	Handlungskompetenz; systematische Zusammenhänge
Lern-/Spielwelten	Vermittlung von Inhalten in einem spielerischen Kontext	wenig vorstrukturiertes und entdeckendes Lernen	Orientierungsverhalten; wissensorientierte Inhalte
Abenteuerspiele	Bestehen eines Abenteuers; Identifikation mit einer Rolle	Integration von Spielhandlung und Didaktik	wissensorientierte Inhalte
Sonstige Spiele	Belohnung, zur Entspannung und zur Erhöhung der Motivation	unbemerktes, nicht geplantes Lernen	Kognitive und sensomotorische Fertigkeiten; Medienkompetenz

2.5 Zusammenfassung

Spiel-basiertes Lernen ermöglicht eine andere Sichtweise auf elektronische Lernformen. Bisher fehlte vor allem der jüngeren Zielgruppe die nötige Motivation, elektronisch unterstützt zu lernen, doch gerade die Motivation ist der entscheidende Aspekt für eine erfolgreiche Vermittlung von Lerninhalten. Wie aber kann man die Motivation erhöhen? Die entscheidende Anregung geben Computerspiele. Kein anderes Medium bietet eine ähnlich motivierende interaktive Unterhaltungsform und nutzt dafür alle technischen, gestalterischen und inhaltlichen Möglichkeiten des Computers aus. Der Lernende wird zum Spieler und im Vordergrund stehen nicht mehr die Lerninhalte, sondern das Spiel. Für den so motivierten Spieler wird Lernen zum Vergnügen und zum positiven Nebeneffekt des Spiels.

Die Umsetzung eines solchen digitalen Lernspiels gestaltet sich allerdings noch sehr schwierig, denn ein generelles Rezept ist noch nicht gefunden. Zu unterschiedlich sind die Erwartungen und Wünsche der Nutzer. Den richtigen Kompromiss, aus Spiel- und Lerninhalten für die jeweilige Zielgruppe, zu finden garantiert schließlich den Erfolg.

Konzeption

3 Anforderungsanalyse

3.1 Allgemeines

Zur Vorbereitung möglicher Integrationskonzepte soll in diesem Kapitel analysiert werden, welche Anforderungen die, bei einer Integration eines Spiel-basierten Lernprogramms in eine SCORM-konforme Lernumgebungen, beteiligten Objekte stellen. Dazu werden zunächst allgemeine und spezielle Anforderungen SCORM-konformer Lernumgebungen und anschließend die Anforderungen Spiel-basierter Lernprogramme betrachtet. Des Weiteren wird die Umsetzbarkeit dieser Anforderungen untersucht.

Die allgemeinen Anforderungen SCORM-konformer Lernumgebungen ergeben sich hauptsächlich aus den didaktischen und technischen Konzepten der SCORM Spezifikation. Die speziellen Anforderungen SCORM-konformer Lernumgebungen ergeben sich aus den in der SCORM Spezifikation enthaltenen Voraussetzungen für die Integration von Lerninhalten. Die im folgenden Abschnitt erläuterten SCORM *Conformance Requirements* (CR) bilden dabei die Grundlage. Die Anforderungen Spiel-basierter Lernprogramme resultieren aus den didaktischen Aspekten des Spiel-basierten Lernens (vgl. Abschnitt 2.3) und sind abgeleitet aus einer Untersuchung zu den typischen Bestandteilen digitaler Lernspiele. Die Anforderungen werden in einem Anforderungskatalog Spiel-basierter Lernprogramme zusammengefasst.

3.2 Conformance Requirements

Die SCORM Bücher, vorgestellt in Abschnitt 1.4, erläutern alle technischen Details der Spezifikation. Eine Zusammenfassung der Anforderungen, die am Lernprozess beteiligte Objekte erfüllen müssen, um SCORM-Konformität zu erreichen, stellt ADL mit dem SCORM Conformance Requirements Dokument zur Verfügung. Weiterhin bietet ADL dafür einen Selbsttest an, die SCORM *Conformance Test Suite*. Absolvieren die Objekte diesen Test erfolgreich, können sie sich als „SCORM 2004 3rd Edition Conformant“ bezeichnen, da sie alle Anforderungen erfüllen. Wird der Test von einem ADL Certification Testing Center durchgeführt, können sich die Objekte sogar als „SCORM 2004 3rd Edition Certified“ bezeichnen [ADL06b]. Technisch gibt es zwischen den verschiedenen Bezeichnungen allerdings keinen Unterschied.

Je nach Objekt müssen unterschiedliche Anforderungen erfüllt werden. In den nachfolgenden

Abschnitten werden die Voraussetzungen für die einzelnen Objekte (LMS, Content Package, SCO) beschrieben. Einen Überblick, welches Objekt welche Anforderungen erfüllen muss, bieten dabei die Conformance Requirements Matrices (vgl. Tabellen 3.1, 3.2 und 3.3). Das vollständige SCORM 2004 3rd Edition Conformance Requirements Dokument befindet sich auf der, dieser Arbeit, beiliegenden CD.

3.3 Anforderungen SCORM-konformer Lernumgebungen

3.3.1 Allgemeine Anforderungen

Das elektronisch unterstützte Lernen beinhaltet heutzutage wesentlich mehr, als die reine Präsentation von Inhalten und Medien zu einem bestimmten Thema. Nach [Ho06] werden die Angebote vor allem durch die Vollständigkeit der Informationen und die Interaktionsmöglichkeiten charakterisiert. So kommt zu einer umfassenden Darstellung eines Themas auch die Möglichkeit der Nutzer miteinander zu kommunizieren. Neben diesen Grundbausteinen führen [Ho06] als weiteres Kennzeichen den nicht linearen und unmittelbaren Zugang zu den Informationen über eine einfache und komfortable Benutzerschnittstelle an. Die so entstehende Benutzerumgebung bezeichnet man im Allgemeinen als „Lernumgebung“ (vgl. Abschnitt 1.2.3). Eine genaue Definition dieses Begriffes und Details zur technischer Umsetzung finden sich in der SCORM Spezifikation nicht. Die Spezifikation konzentriert sich mehr auf die Schnittstelle zwischen Lernobjekten und der Lernumgebung und weniger auf spezifische Details und Fähigkeiten der Implementation, welche den Entwicklern überlassen werden. Dies hat zur Folge, dass sich die unterschiedlichen Implementationen stark voneinander unterscheiden, stets unter Einhaltung der Interoperabilität.

Aus den SCORM Grundlagen, vorgestellt in Abschnitt 1.4, ergeben sich dennoch fundamentale Fähigkeiten einer Lernumgebung. Dazu gehören nach [ADL06a]:

- das Laden von Inhalten, die mit Werkzeugen unterschiedlicher Anbieter erstellt wurden, sowie der Austausch von Daten mit diesen Inhalten
- die Fähigkeit von Lernumgebungen, unterschiedlicher Anbieter, den selben Inhalt zu laden und Daten mit diesen Inhalten während der Laufzeit auszutauschen
- die Fähigkeit mehrerer Lernumgebungen auf eine gemeinsame Quelle von ausführbaren Inhalten zuzugreifen und diese zu laden

Die Schlüsselfunktion einer Lernumgebung ist somit die Steuerung der gesamten Lernerfahrung des Lernenden. Die Lernumgebung entscheidet welche Lerninhalte, wann an den Lernenden ausgeliefert werden und verfolgt sowohl die Durchführung als auch den Fortschritt des gesamten Lernprozesses.

LMS Conformance Requirements

Tabelle 3.1: LMS Conformance Matrix [ADL06b]

LMS Conformance Matrix
Conformance Label: **LMS SCORM 2004 3rd Edition Conformant** The LMS shall adhere to the conformance requirements defined for the following Conformance Categories: **LMS Run-Time Environment Version 1.0 (LMS RTE 1.0)** **LMS Content Aggregation Model Version 1.0 (LMS CAM 1.0)** **LMS Sequencing and Navigation Version 1.0 (LMS SN 1.0)**
Conformance Category: **LMS RTE 1.0** *Requirements Summary:* - The LMS shall be able to launch an Asset, and - The LMS shall be able to launch a known SCORM 2004 3rd Edition conformant Sharable Content Object (SCO), and - The LMS shall provide and expose an API Instance as a Document Object Model (DOM) object that correctly implements all of the API methods, and - The LMS shall correctly implement support for all SCORM 2004 3rd Edition Run-Time Environment Data Model Elements, and - The LMS shall correctly implement support for all SCORM 2004 3rd Edition Navigation Data Model Elements.
Conformance Category: **LMS CAM 1.0** *Requirements Summary:* - The LMS shall be able to "import" and process a known SCORM 2004 3rd Edition conformant SCORM Content Aggregation Application Profile Content Package, and - The LMS shall correctly initialize SCORM 2004 3rd Edition Run-Time Environment Data Model elements based on information supplied in a Content Package Manifest
Conformance Category: **LMS SN 1.0** *Requirements Summary:* - The LMS shall correctly implement all of the sequencing behaviors defined by the pseudocode included in the SCORM 2004 3rd Edition Sequencing and Navigation (SN) Version 1.0, and - The LMS shall correctly implement support for all SCORM 2004 3rd Edition Navigation Data Model Elements, and - The LMS shall correctly implement support for User Interface requirements.

Für die technische Umsetzung einer Lernumgebung, dem LMS, regelt SCORM die genaue Anforderungen zur Sicherung der Konformität, zusammengefasst in den *LMS Conformance Requirements*. Einen Überblick über die Anforderungen bietet Tabelle 3.1, welche die *LMS Conformance Matrix* enthält. Danach muss ein LMS die Spezifikationen der SCORM RTE, des SCORM CAM und der SCORM SN umsetzen, um sich als „LMS SCORM 2004 3rd Edition Conformant“ zu bezeichnen.

Grundlegende Anforderung ist die Fähigkeit des LMS ein Content Package in Form eines PIF zu importieren und darin verwendete Ressourcen (SCOs oder Assets) zu laden. Weiterhin müssen die SCORM RTE API und das zugehörige Data Model sowie die Sequencing Behaviors und das Navigation Data Model in dem LMS, unter Beachtung des SCORM CAM, implementiert sein. Die kompletten Anforderungen befinden sich im zweiten Kapitel des SCORM CR Dokuments, welches in die folgenden Abschnitte unterteilt ist [ADL06b]:

- *Launch Conformance Requirements* behandelt die Anforderungen an das LMS zum Laden der Ressourcen. Entscheidend sind hier die festgelegte Content Organization sowie eventuell im Content Package Manifest vorhandene Informationen zum Sequencing.
- *API Implementation Conformance Requirements* regelt die Anforderungen an die im LMS implementierte API Instanz, basierend auf der SCORM RTE.
- *Run-Time Environment Data Model Conformance Requirements* listet alle Data Model Elemente auf, die eine LMS implementieren muss, um zur Laufzeit mit einem SCO über die API zu kommunizieren.
- *Run-Time Environment Data Model Data Type Conformance Requirements* listet zusätzlich alle verfügbaren Datentypen der Data Model Elemente auf, basierend auf den Run-Time Environment Data Model Data Types.
- *Navigation Data Model Conformance Requirements* listet alle Navigation Data Model Elemente auf die ein LMS implementieren muss, um mit einer SCO Navigation umzugehen.
- *Sequencing Conformance Requirements* enthält alle Testfälle zum Überprüfen der Anforderungen des Sequencing an das LMS. Wie das LMS das Sequencing implementiert, ist nicht vorgegeben. Die Ausgangsbasis bildet das SCORM SN.

- *User Interface Conformance Requirements* beschreibt die minimalen User Interface Schnittstellen, die ein LMS dem Nutzer zur Verfügung stellen sollte, damit dieser die Inhalte abrufen kann. Gleichzeitig wird festgelegt welche Schnittstellen zu bestimmten Zeitpunkten nicht zur Verfügung stehen sollten, um Fehlermeldungen zu vermeiden.

3.3.2 Spezielle Anforderungen

Bei der Integration in eine SCORM-konforme Lernumgebung wird das Spiel-basierte Lernprogramm genauso wie andere Lerninhalte behandelt. Die technischen Anforderungen, die SCORM an Lerninhalte stellt, ergeben sich aus den in SCORM festgelegten Anforderungen für Content Packages und SCOs, zusammengefasst in den *Content Package Conformance Requirements* bzw. den *SCO Conformance Requirements.*

Content Package Conformance Requirements

Ein Content Package (vgl. Abschnitt 1.4.2) muss die Spezifikationen der SCORM RTE und des SCORM CAM umsetzen, um sich als „CP SCORM 2004 3rd Edition Conformant“ zu bezeichnen. Einen Überblick über die Anforderungen bietet Tabelle 3.2, welche die *Content Package Conformance Matrix* enthält. Die kompletten Anforderungen befinden sich im dritten Kapitel des SCORM CR Dokuments. Das Kapitel ist in die folgenden Abschnitte unterteilt [ADL06b]:

- *Content Package Conformance Requirements* beschreibt welche festen Anforderungen jedes Content Package erfüllen muss. Beispielsweise muss jedes Content Package ein Manifest und mindestens ein SCO oder Asset enthalten.
- *Content Aggregation Package Manifest Conformance Requirements* regelt die Anforderungen an das Content Aggregation Package Manifest.
- *Sequencing Extensions Conformance Requirements* definiert die möglichen Sequencing Elemente, die in einem Content Package Manifest verwendet werden können, basierend auf dem SCORM SN Buch.
- *Navigation and Presentation Conformance Requirements* definiert die möglichen Navigation Elemente, die in einem Content Package Manifest verwendet werden können, basierend auf dem SCORM SN Buch.
- *Resource Package Manifest Conformance Requirements* beschreibt die Anforderungen an das Manifest beim Erstellen eines Resource Packages.

Tabelle 3.2: Content Package Conformance Matrix [ADL06b]

Content Package Conformance Matrix
Conformance Label: **CP SCORM 2004 3rd Edition Conformant** The Content Package shall adhere to the conformance requirements defined for the following Conformance Categories: **Content Package Content Aggregation Model Version 1.0 (CP CAM 1.0)** **Content Package Run-Time Environment Version 1.0 (CP RTE 1.0)**
Conformance Category: **CP CAM 1.0** *Requirements Summary:* - The Content Package shall conform to the requirements defined for the Content Package, and - If the Content Package is a SCORM 2004 3rd Edition Content Aggregation Package Application Profile, then the Manifest shall conform to the Content Aggregation Package Application Profile Manifest requirements, and - If the Content Package is a SCORM 2004 3rd Edition Content Aggregation Package Application Profile and the Manifest contains SCORM Sequencing information, then the sequencing extensions in the Manifest shall conform to the SCORM 2004 3rd Edition Sequencing Extension requirements, and - If the Content Package is a SCORM 2004 3rd Edition Content Aggregation Package Application Profile and the Manifest contains SCORM Navigation/Presentation information, then the navigation/presentation extensions in the Manifest shall conform to the SCORM 2004 3rd Edition Navigation/Presentation Extension requirements, and - If the Content Package is a SCORM 2004 3rd Edition Resource Package Application Profile, then the Manifest shall conform to the Resource Package Application Profile Manifest requirements, and - If the Content Package Manifest contains metadata, then the metadata shall be well-formed and valid according to the respective Controlling Document (e.g., XSD, DTD).
Conformance Category: **CP RTE 1.0** *Requirements Summary:* - The Content Package shall contain at least one Sharable Content Object (SCO) resource or Asset resource, and - All SCO resources identified in the Manifest shall conform to the SCO conformance requirements.

SCO Conformance Requirements

Ein SCO (vgl. Abschnitt 1.4.2) muss die Spezifikationen der SCORM RTE umsetzen, um sich als „SCO SCORM 2004 3rd Edition Conformant“ zu bezeichnen. Grundlegende Anforderung an das SCO ist die Fähigkeit, die vom LMS zur Verfügung gestellte API Instanz zu suchen und zu finden. Des Weiteren müssen die Methoden zum Initialisieren und Beenden

der Kommunikation mit dem LMS implementiert werden. Alle anderen SCORM API Methoden sind zwar optional, aber in der Spezifikation definiert und müssen demnach korrekt verwendet werden. Die SCO Conformance Requirements garantieren allerdings nicht, dass das SCO inhaltlich und technisch vollständig fehlerfrei ist.

Tabelle 3.3: SCO Conformance Matrix [ADL06b]

SCO Conformance Matrix
Conformance Label: **SCO SCORM 2004 3rd Edition Conformant** The SCO shall adhere to the conformance requirements defined for the following Conformance Categories: **SCO Run-Time Environment Version 1.0 (SCO RTE 1.0)**
Conformance Category: **SCO RTE 1.0** *Requirements Summary:* - The SCO shall search for and find an API Instance named API_1484_11 as a Document Object Model (DOM) object as defined in the SCORM 3rd Edition Run-Time Environment Version 1.0, and - The SCO shall successfully invoke, at a minimum, the `Initialize()` and `Terminate()` API methods, and - If used, the SCO shall successfully invoke the Data Transfer Methods, and - If used, the SCO shall successfully invoke the Support Methods, and - If using the Data Transfer Methods, the SCO shall ensure that all SCORM 2004 3rd Edtion Run-Time Environment Data Model elements used in the method calls adhere to the requirements of those elements.

Einen Überblick über die Anforderungen bietet Tabelle 3.3, welche die *SCO Conformance Matrix* enthält. Die kompletten Anforderungen befinden sich im vierten Kapitel des SCORM CR Dokuments. Das Kapitel ist in die folgenden Abschnitte unterteilt [ADL06b]:

- *Launch Conformance Requirements* beschreibt die Anforderungen zum Finden der API Instanz des LMS und die korrekte Nutzung der Methoden zum Initialisieren und Beenden der Kommunikation mit dem LMS.
- *API Conformance Requirements* listet alle nutzbaren API Methoden und die damit verbundenen Anforderungen an das SCO auf.
- *Run-Time Environment Data Model Conformance Requirements* definiert die Data Model Elemente zum Datenaustausch mit dem LMS. Der Datenaustausch ist optional und kann auch über die definierten Elemente hinausgehen.

- *Run-Time Environment Data Model Data Type Conformance Requirements* beschäftigt sich mit den verfügbaren Datentypen der Data Model Elemente, basierend auf den Run-Time Environment Data Model Data Types.
- Run-Time Navigation Data Model Conformance Requirements definiert die Navigation der Data Model Elemente, die optional genutzt werden können.

3.4 Anforderungen Spiel-basierter Lernprogramme

Spiel-basierte Lernprogramme können in ihrer jeweiligen Ausprägung sehr unterschiedliche Komponenten beinhalten. Um daraus Anforderungen zu formulieren, müssen diese Komponenten zunächst untersucht werden. [Burgos06] betrachteten dazu eine Vielzahl von Spielen und einschlägigen Magazinen und extrahierten daraus didaktische und technische Komponenten von digitalen Lernspielen, die in den Tabellen 3.4 und 3.5 aufgelistet sind.

Tabelle 3.4: Didaktische Komponenten von Lernspielen [Burgos06]

Bestandteil	Bemerkung
Eine oder mehrere Lösungen	Es gibt nur eine Möglichkeit oder mehrere durchführbare Lösungen, um das Spiel erfolgreich abzuschließen.
Offene oder geschlossene Lösung	Die Lösung wird intern aus einer Auswahl an Lösungen ausgewählt oder der Spieler erzeugt seine eigene Lösung, die so nicht in der Anwendung vorhanden war.
Eigenständige oder gemeinschaftliche Lösung	Wie viele Spieler werden zur Lösung des Spiels benötigt?
Gemeinschaftliche oder wettbewerbliche Ausführung	Muss zur Lösung des Spiels gemeinschaftlich gearbeitet werden oder wird ein Wettbewerb unter den Spielern initiiert?
Dynamische Rückmeldung	Eine Einschätzung/Bewertung der Aktionen des Spielers ist vorhanden und dient zum Konditionieren des Spielablaufs.
Adaptives Lernen	Es werden individuelle Inhalte und Einschätzungen abhängig vom Profil des Spielers und seiner Art und Weise des Spielens verwendet.
Stufenweises oder isoliertes/abgetrenntes Lernen	Werden innerhalb des Lernprozesses stufenweise Lernebenen oder einzelne abgetrennte Lernmodule verwendet?

Bei der Auswahl der Komponenten muss beachtet werden, dass einige Komponenten auf anderen Komponenten aufbauen oder von diesen abhängig sind. Die Umsetzung einer Komponente impliziert daher meist die Umsetzung einer ganzen Reihe anderer Komponenten.

So stehen beispielsweise die Komponenten „Lokale oder verteilte Ausführung“ und „Einzel- oder Mehrspieler“ in Tabelle 3.5 in direktem Zusammenhang.

Tabelle 3.5: Technische Komponenten von Lernspielen [Burgos06]

Bestandteil	Bemerkung
Einzel- oder Mehrspieler	Anzahl der Spieler zur gleichen Zeit.
Unterstützung von Nutzergruppen	Gemeinsames Benutzen von Ressourcen, Teilen von Zielen und Kommunizieren.
Lokale oder verteilte Ausführung	Notwendigkeit einer Netzwerkverbindung oder einer Internetverbindung.
Synchrone oder asynchrone Kommunikation	Möglichkeit der Kommunikation zur Laufzeit.
Nutzung von Multimediaelementen	Integration und Nutzung von Multimediaelementen.
Graphische Unterstützung	Anteil der graphischen Unterstützung aus Sicht der Performance
2D- oder 3D-Graphik bzw. Vektor- oder Bitmap-basierte Bilder	Verwendete Grafikformate zur graphischen Unterstützung.
Einsatz einer Spiel-Engine zur Laufzeit	Ist bereits alles vordefiniert oder führt eine eigene Spiel-Engine zur Laufzeit Berechnungen (z.B.: Grafik, KI, Physik, Zustand, Steuerung) für das Spiel durch?
Statische oder dynamische Eingabefelder	Änderungsmöglichkeit an Eingabefeldern.
Individuell anpassbare Eigenschaften oder statisches Profil	Änderungsmöglichkeit an spezifischen Kennzeichen des Spiels oder des Spielers zur Laufzeit.
Verwendung (Lesen, Speichern) von externen Dateien	Lesen und speichern von relevanten Daten des Spielers oder des Spiels.
Kommunikation mit anderen Anwendungen und Werkzeugen	Austausch von Informationen während des Spiels.
Bezug/Interaktion zur Realität notwendig	Steht das Spiel allein oder wird ein Bezug oder eine Interaktion mit der realen Welt notwendig.

Aus den in Abschnitt 2.3 erläuterten didaktischen Aspekten und den in den Tabellen 3.4 und 3.5 aufgeführten Komponenten lassen sich nun die folgenden Anforderungen Spiel-basierter Lernprogramme ableiten:

- *Unterstützung kooperativer Lernformen:* Diese Anforderung beschreibt vor allem die Möglichkeit den Lernprozess in einer Gruppe gemeinschaftlich zu durchlaufen und im Rahmen eines Wettbewerbs oder durch Kooperation eine gemeinsame Lösung zu

erarbeiten. Das Lernspiel sollte also in einer verteilten Umgebung ausgeführt werden können, welche die Erstellung von Nutzergruppen und das Teilen gemeinsamer Ressourcen ermöglicht. Innerhalb des Lernspiels sollte für den Nutzer ein entsprechender Multi-User Modus bereitstehen, auf dem die Interaktion der Lernenden untereinander während des Spiels basiert. Ein weiterer wichtigen Bestandteil ist die Möglichkeit mit anderen Nutzern zu kommunizieren (synchron und asynchron).

- *Strukturierung des Lernprozesses:* Bei dieser Anforderung können zwei Strukturen des Lernprozesses unterschieden werden. Einerseits das stufenweise Lernen mit aufeinander aufbauenden Lernebenen und andererseits das abgetrennte Lernen in voneinander unabhängigen Lerneinheiten. Die Lernebenen bzw. die Lerneinheiten entsprechen dabei im Spielaufbau am ehesten einzelnen Levels oder Szenarien des Spiels. Die Möglichkeit den Lernprozess zu strukturieren bildet auch eine mögliche Ausgangsbasis für die nächste Anforderung, die dynamische Anpassung des Lernprozesses.
- *Anpassung des Lernprozesses:* Die Fähigkeit, sich an veränderte Bedingungen anzupassen, ist Kern dieser Anforderung. So können dem Nutzer, beispielsweise statt einer, auch mehrere Lösungen des Lernspiels angeboten werden. Erst durch die individuelle Gestaltung seines eigenen Lernprozesses bestimmt er dabei den endgültigen Lösungsweg. Eine Möglichkeit wäre auch, gar keine Lösung vorzugeben und den Nutzer seine eigene Lösung kreieren zu lassen. Technisch sind mit solchen dynamischen Prozessen viele Berechnungen verbunden. Der Einsatz einer Spiel-Engine ermöglicht in diesem Zusammenhang die Berechnung verschiedener Parameter zur Laufzeit, in komplexen Spielwelten auch in Echtzeit.
- *Erstellung individueller Lernprofile:* Als Grundlage dieser Anforderung müssen für jeden Nutzer bestimmte Eigenschaften und Optionen festgelegt, und wenn möglich auch zur Laufzeit angepasst, werden können. Die Lernprofile ermöglichen die Anpassung des Lernprozesses an individuelle Bedürfnisse und Fähigkeiten sowie die Bewertung des Spielers und stehen somit in direktem Zusammenhang mit der vorhergehenden und der nächsten Anforderung.
- *Bewertung des Nutzers:* Um dem Nutzer die Selbsteinschätzung in Bezug auf den Fortschritt innerhalb des Lernprozesses und im Vergleich zu anderen Lernenden zu

ermöglichen, ist es notwendig, ihm kontinuierlich und nicht nur nach erfolgreichem Abschluss des Lernspiels eine Rückmeldung zu geben. Diese Einschätzung bzw. Bewertung dient dabei auch zur Aufrechterhaltung der Motivation sowie der Konditionierung des Nutzers.

- *Erweiterung der Kommunikationsmöglichkeiten:* Diese Anforderung beschäftigt sich mit den Kommunikationsmöglichkeiten der Lernobjekte und somit auch der Nutzer untereinander. Sie steht damit in direktem Zusammenhang mit der Anforderung zur Unterstützung kooperativer Lernformen. Des Weiteren könnten Spiele-basierte Lernprogramme in ihrer technischen Umsetzung auch erweiterte Kommunikationsmöglichkeiten mit anderen Anwendungen und Werkzeugen sowie die Verwendung externer Dateien voraussetzen. Beispielsweise könnten so das Profil des Spielers, oder spezifische Zustände des Spiels, auf einem zentralen Server gespeichert werden.
- *Nutzung multimedialer Elemente:* Wie bei klassischen Lernprogrammen, ist für den Erfolg eines Spiel-basierten Lernprogramms neben einer gelungenen graphischen Umsetzung auch der Einsatz multimedialer Elemente entscheidend. Gerade Lernspiele bieten in diesem Zusammenhang vielfältige Integrationsmöglichkeiten, die über die klassische Verwendung multimedialer Elemente innerhalb von Lernobjekten hinausgehen. Denkbar ist hier beispielsweise auch der Einsatz komplexer interaktiver virtueller Lernwelten.

Auch bei der Auswahl und Formulierung dieser Anforderungen muss beachtet werden, dass wiederum einige Anforderungen aufeinander aufbauen oder voneinander abhängig sind. In diesem Sinne lassen sich folgende fünf Gruppen aus den erläuterten Anforderungen zusammenstellen:

- Unterstützung kooperativer Lernformen
- Strukturierung & Anpassung des Lernprozesses
- Erstellung individueller Lernprofile & Bewertung des Nutzers
- Erweiterung der Kommunikationsmöglichkeiten
- Nutzung multimedialer Elemente

Als Grundlage für die Bewertung der Integrationskonzepte in Kapitel 4, müssen diese Anforderungsgruppen noch genauer spezifiziert werden. Der in Tabelle 3.6 beschriebene Anforderungskatalog Spiel-basierter Lernprogramme listet die Anforderungsgruppen noch einmal auf und ordnet den Gruppen konkret formulierte Anforderungen, entsprechend der obigen Erläuterungen, zu.

Tabelle 3.6: Anforderungskatalog Spiel-basierter Lernprogramme

Unterstützung kooperativer Lernformen	
	Gemeinschaftlicher Lernprozess in einer verteilten Umgebung
	Einrichtung von Gruppen, die gemeinsame Ressourcen teilen
	Möglichkeiten zur Kommunikation der Nutzer untereinander
Strukturierung & Anpassung des Lernprozesses	
	Strukturierung entsprechend des Spielaufbaus
	Dynamische Gestaltung des Lernprozesses (individuell/innerhalb einer Gruppe)
	Datenmodell zur Strukturierung und Steuerung des Spiels
Erstellung individueller Lernprofile & Bewertung des Nutzers	
	Individuelles anpassbares Lernprofil
	Kontinuierliche Bewertung und Rückmeldung (individuell/innerhalb einer Gruppe)
	Datenmodell für ein Lernprofil sowie zur Bewertung und Rückmeldung
Erweiterung der Kommunikationsmöglichkeiten	
	Direkte Kommunikation der Lernobjekte (SCOs) untereinander
	Schnittstellen zu spezialisierten Servern und/oder lokalen Anwendungen
	Datenmodell zum Informationsaustausch zwischen den Lernobjekten
Nutzung multimedialer Elemente	
	Verknüpfung und Nutzung multimedialer Elemente
	Möglichkeit der Einbindung interaktiver virtueller Lernwelten

Die meisten Anforderungen basieren auf der Verwendung eines einheitlichen Datenmodells. Das Vorhandensein eines solchen Datenmodells wurde deshalb in den Anforderungskatalog einbezogen und den betreffenden Anforderungsgruppen zugeordnet. Bei den beiden Anforderungen dynamische Gestaltung des Lernprozesses und kontinuierliche Bewertung und Rückmeldung wurde außerdem zwischen einer Erfüllung der Anforderung für einen einzelnen Nutzer und innerhalb einer Gruppe unterschieden.

3.5 Umsetzbarkeit der Anforderungen

Trotz der sowohl technischen als auch didaktischen Unterschiede sind Spiel-basierte Lernprogramme vor allem im Aufbau klassischen Lernprogrammen sehr ähnlich und eigenen sich prinzipiell auch für die Integration in eine SCORM-konforme Lernumgebung. Eine SCORM-konforme Lernumgebung ist dabei charakterisiert durch die Umsetzung der Anforderungen der LMS Conformance Requirements (vgl. Abschnitt 3.3.1).

Tabelle 3.7: Erfüllbare Anforderungen (SCORM) eines Spiel-basierten Lernprogramms ohne Multi-User Aspekt

✓ - erfüllte Anforderung / ✗ - nicht erfüllte Anforderung	
Unterstützung kooperativer Lernformen	
✗	Gemeinschaftlicher Lernprozess in einer verteilten Umgebung
✓	Einrichtung von Gruppen, die gemeinsame Ressourcen teilen
✓	Möglichkeiten zur Kommunikation der Nutzer untereinander
Strukturierung & Anpassung des Lernprozesses	
✓	Strukturierung entsprechend des Spielaufbaus
✓ / ✗	Dynamische Gestaltung des Lernprozesses (individuell/innerhalb einer Gruppe)
✓	Datenmodell zur Strukturierung und Steuerung des Spiels
Erstellung individueller Lernprofile & Bewertung des Nutzers	
✓	Individuelles anpassbares Lernprofil
✓ / ✗	Kontinuierliche Bewertung und Rückmeldung (individuell/innerhalb einer Gruppe)
✓	Datenmodell für ein Lernprofil sowie zur Bewertung und Rückmeldung
Erweiterung der Kommunikationsmöglichkeiten	
✗	Direkte Kommunikation der Lernobjekte (SCOs) untereinander
✗	Schnittstellen zu spezialisierten Servern und/oder lokalen Anwendungen
✓	Datenmodell zum Informationsaustausch zwischen den Lernobjekten
Nutzung multimedialer Elemente	
✓	Verknüpfung und Nutzung multimedialer Elemente
✓	Möglichkeit der Einbindung interaktiver virtueller Lernwelten

Als einfachste Möglichkeit könnte man die Spiel-basierten Lernprogramme wie klassische Lernprogramme in ein solches LMS integrieren. Entsprechende Anpassungen müssen dabei lediglich an dem Spiel-basierten Lernprogramm selbst vorgenommen werden, um die Anforderungen der Conformance Requirements (vgl. Abschnitt 3.3.2) vollständig umzusetzen

und damit SCORM Konformität zu erreichen. Zusätzlich sollten möglichst viele der in Abschnitt 3.4 erläuterten Anforderungen Spiel-basierter Lernprogramme umgesetzt werden. Anhand des erarbeiteten Anforderungskataloges soll deshalb untersucht werden, welche der enthaltenen Anforderungen eine SCORM-konforme Lernumgebung bereits ohne zusätzliche Erweiterungen umsetzen kann und welche nicht. Tabelle 3.7 listet die Anforderungen des Anforderungskataloges noch einmal auf und stellt die Erfüllung der entsprechenden Anforderungen eines Spiel-basierten Lernprogramms, zunächst ohne Multi-User Aspekt, durch die SCORM Spezifikation grafisch dar. Tabelle 3.8 listet die Anforderungen ein zweites Mal auf und stellt die Erfüllung der entsprechenden Anforderungen eines Spiel-basierten Lernprogramms, diesmal mit Multi-User Aspekt, durch die SCORM Spezifikation grafisch dar.

Tabelle 3.8: Erfüllbare Anforderungen (SCORM) eines Spiel-basierten Lernprogramms mit Multi-User Aspekt

✅ - erfüllte Anforderung / ❌ - nicht erfüllte Anforderung	
Unterstützung kooperativer Lernformen	
❌	Gemeinschaftlicher Lernprozess in einer verteilten Umgebung
❌	Einrichtung von Gruppen, die gemeinsame Ressourcen teilen
❌	Möglichkeiten zur Kommunikation der Nutzer untereinander
Strukturierung & Anpassung des Lernprozesses	
✅	Strukturierung entsprechend des Spielaufbaus
✅ / ❌	Dynamische Gestaltung des Lernprozesses (individuell/innerhalb einer Gruppe)
❌	Datenmodell zur Strukturierung und Steuerung des Spiels
Erstellung individueller Lernprofile & Bewertung des Nutzers	
✅	Individuelles anpassbares Lernprofil
✅ / ❌	Kontinuierliche Bewertung und Rückmeldung (individuell/innerhalb einer Gruppe)
❌	Datenmodell für ein Lernprofil sowie zur Bewertung und Rückmeldung
Erweiterung der Kommunikationsmöglichkeiten	
❌	Direkte Kommunikation der Lernobjekte (SCOs) untereinander
❌	Schnittstellen zu spezialisierten Servern und/oder lokalen Anwendungen
❌	Datenmodell zum Informationsaustausch zwischen den Lernobjekten
Nutzung multimedialer Elemente	
✅	Verknüpfung und Nutzung multimedialer Elemente
✅	Möglichkeit der Einbindung interaktiver virtueller Lernwelten

Wie aus den Tabelle zu entnehmen ist, können viele der Anforderungen Spiel-basierter Lernprogramme durchaus von SCORM-konformen Lernumgebungen erfüllt werden. Vor allem die Strukturierung entsprechend des Spielaufbaus, die dynamische Gestaltung des individuellen Lernprozesses, das individuell anpassbare Lernprofil sowie die kontinuierliche individuelle Bewertung und Rückmeldung können mit Hilfe der SCORM Spezifikation sehr gut realisiert werden. Jedoch ist die Beurteilung, ob eine Anforderung erfüllt werden kann oder nicht, in einigen Fällen stark abhängig von der Beachtung des Multi-User Aspektes. Beim direkten Vergleich der Tabellen 3.7 und 3.8 wird klar, welche Auswirkungen der Multi-User Aspekt Spiel-basierter Lernprogramme auf die Erfüllung der Anforderungen durch die SCORM Spezifikation hat. Dieser Umstand lässt sich recht einfach durch die fehlende Unterstützung des Multi-User Aspektes innerhalb der SCORM Spezifikation begründen.

Trotz der klar definierten Vorgaben innerhalb der Spezifikation und der damit einhergehenden Grenzen des technisch Machbaren, ist die Beurteilung ob eine Anforderung erfüllt werden kann oder nicht, nicht immer ganz eindeutig zu treffen. Beispielsweise wird in vielen Lernumgebungen die Einrichtung von Gruppen und das Teilen gemeinsamer Ressourcen zwar unterstützt, eine Kooperation der Gruppe innerhalb eines Lernprozesses kann aber nicht umgesetzt werden. Ähnlich verhält es sich mit den Möglichkeiten zur Kommunikation der Nutzer untereinander. Prinzipiell bieten Lernumgebungen den Nutzern mehr oder weniger umfangreiche synchrone und asynchrone Kommunikationsmöglichkeiten. Innerhalb des Lernprozesses sind diese dann aber meist nur eingeschränkt oder gar nicht verfügbar. Das gilt vor allem für synchrone Kommunikationsmöglichkeiten. Ein weiteres Beispiel ist das SCORM Data Model, welches zur Strukturierung und Steuerung des Spiels, Beschreibung eines Lernprofils, zur Bewertung und Rückmeldung und zum Informationsaustausch zwischen den Lernobjekten genutzt werden kann. Das Datenmodell ist zwar für die Verwendung innerhalb eines klassisches Lernprogramms im Sinne der SCORM Spezifikation sehr gut geeignet, Elemente zur Unterstützung und Beschreibung kooperativer Lernformen sind aber nicht vorhanden. Dazu würde beispielsweise ein umfangreicheres Nutzerprofil gehören, welches den einzelnen Nutzer ausführlich beschreibt und somit innerhalb der Gruppe deutlicher hervorhebt. In Bezug auf das Thema dieser Arbeit und der damit einhergehenden besonderen Beachtung des Multi-User Aspektes, wurden die Anforderungen in den erläuterten Fällen innerhalb der Übersicht in Tabelle 3.8 aus den angeführten Gründen schließlich als nicht erfüllbar beurteilt, um die fehlende Unterstützung kooperativer

Lernformen zu verdeutlichen. Tabelle 3.9 fasst die technischen Möglichkeiten der SCORM Spezifikation in diesem Kontext zusammen und erläutert die dabei auftretenden Probleme.

Tabelle 3.9: Umsetzung der Anforderungen in SCORM

Unterstützung kooperativer Lernformen
• Kooperative Lernformen (Multi-User Aspekte) werden nicht unterstützt. Lerninhalte werden für jeden einzelnen Nutzer ausgeliefert, denn die SCORM Spezifikation ist ausgerichtet auf den prozeduralen Lernprozess eines eigenständig lernenden Nutzers. • Das SCORM RTE Data Model ist dementsprechend nur auf einzelne Nutzer ausgelegt und enthält keine Elemente zur Beschreibung kooperativer Lernformen.
Strukturierung & Anpassung des Lernprozesses
• Eine Strukturierung des Lernprozesses lässt sich mit Hilfe des SCORM SN unter Beachtung des SCORM CAM umsetzen. Grundlage dafür ist das SCORM RTE Data Model und das SCORM SN Tracking Model. • Da innerhalb des SCORM RTE Data Model keine Elemente zur Beschreibung kooperativer Lernformen existieren, lässt sich mit dem SCORM SN die Auswahl und Auslieferung von Lerninhalten innerhalb einer kooperativen Lernform nur sehr schwer umsetzen.
Erstellung individueller Lernprofile & Bewertung des Nutzers
• Das SCORM RTE Data Model enthält Elemente zur Beschreibung von Zuständen, Punkteständen, Interaktionen und Zielen der Lerninhalte. Daraus lassen sich sowohl Lernprofile als auch Modelle zur Bewertung und Rückmeldung entwickeln. • Kooperatives Lernen erfordert aber auch den direkten Vergleich zu anderen Nutzern. Diese Anforderung kann mit dem bestehenden SCORM RTE Data Model nicht umgesetzt werden, da beispielsweise Elemente zum Anlegen eines umfangreichen Nutzerprofils fehlen.
Erweiterung der Kommunikationsmöglichkeiten
• Die SCORM API ermöglicht die Kommunikation zwischen Lernobjekten (SCO) und LMS. Indirekt können so auch SCOs untereinander Daten austauschen. • Ein LMS hat allerdings keine Möglichkeit mit einem Lernobjekt zu kommunizieren, es kann nur auf entsprechende Anfragen reagieren. Die Möglichkeit zur Kommunikation zwischen Lernobjekten innerhalb einer kooperativen Lernform existiert nicht. • Schnittstellen zu anderen Anwendungen werden zur Sicherung der Wiederverwendbarkeit und der Interoperabilität nicht unterstützt. • Grundsätzlich werden SCORM-konforme Lerninhalte als Webseiten in einem Browser dargestellt. Die Ausführung von Lernobjekten außerhalb eines Browsers, etwa als eigenständige Desktop Anwendung, ist nicht vorgesehen.
Nutzung multimedialer Elemente
• Das SCORM Content Model definiert die eingesetzten Medienelemente als Assets und beschreibt sie mit Metadaten. Diese Assets können dann in Lernobjekte eingebunden werden. • Die multimedialen Grenzen sind vorgegeben durch die Ausführbarkeit der entsprechenden Elemente in einem Browser.

Zusammenfassend lässt sich somit sagen, dass die SCORM Spezifikation einerseits den überwiegenden Teil der Anforderungen Spiel-basierter Lernprogramme ohne einen Multi-User Aspekt erfüllen kann (vgl. Tabelle 3.7). Andererseits aber nur einen geringen Teil der Anforderungen Spiel-basierter Lernprogramme mit einem Multi-User Aspekt erfüllt (vgl. Tabelle 3.8). Die Umsetzbarkeit der meisten Anforderungen ist in diesem Fall nicht gewährleistet. Einige Anforderungen wie die Strukturierung des Lernprozesses oder die Nutzung multimedialer Elemente können allerdings in jedem Fall durch die SCORM Spezifikation umgesetzt werden. Zu beachten ist in diesem Zusammenhang auch, dass ein Spiel-basiertes Lernprogramm nicht zwingend einen Multi-User Aspekt unterstützen muss. Im Sinne der erläuterten didaktischen Aspekte des Spiel-basierten Lernens (vgl. Abschnitt 2.3) sollte man die Umsetzung des Multi-User Aspektes aber stets in Betracht ziehen.

4 Konzepterstellung

4.1 Allgemeines

Aufbauend auf der vorangegangenen Anforderungsanalyse werden in den folgenden Abschnitten der mögliche Aufbau und die Verwendung eines Spiel-basierten Lernprogramms innerhalb einer SCORM-konformen Lernumgebung sowie konkrete Konzepte zur Integration erläutert. Dazu gehört einerseits die Integration eines Spiel-basierten Lernprogramms in eine SCORM-konforme Lernumgebung und andererseits die Integration eines Spiel-basierten Lernprogramms in eine SCORM-konforme Lernumgebung unter Einsatz eines Multi-User-fähigen Spieleservers. Abschließend wird die Erweiterung der SCORM Spezifikation als Ausblick auf die zukünftige Entwicklung aufgeführt. Ausgangspunkt für alle Konzepte ist die SCORM Spezifikation mit der Kernaussage zur Sicherung der Wiederverwendbarkeit von Lerninhalten, basierend auf Interoperabilitätsstandards.

4.2 Aufbau und Verwendung des Spiel-basierten Lernprogramms innerhalb SCORM-konformer Lernumgebungen

Unabhängig von der Art und Weise der Integration, gilt es zunächst zu klären wie ein Spiel-basiertes Lernprogramm aufgebaut seien kann. Der Aufbau ist maßgeblich von Umfang, Komplexität und Verwendung des Lernprogramms abhängig. Ein Spiel-basiertes Lernprogramm kann einerseits Teil eines klassischen Lernprogramms sein und andererseits auch völlig eigenständig als abgeschlossene Lerneinheit agieren. Je nachdem besteht das Lernprogramm im Sinne von SCORM also entweder aus einem SCO oder auch mehreren SCOs, die als Lernobjekte einem Content Package zugeordnet sind, oder es bildet als Content Package eine eigenständige Lerneinheit.

Für die Strukturierung des Lernprozesses definiert die Spezifikation selbst schon einen Ansatz, zu finden im SCORM CAM (vgl. Abschnitt 1.4.2) und dem SCORM SN Dokument (vgl. Abschnitt 1.4.4). Im Sinne des CAM würden alle Bestandteile eines Spiel-basierten Lernprogramms in einem Content Package enthalten sein. Die dazugehörige Manifest Datei würde neben den einzelnen Bestandteilen auch deren Strukturierung in eine oder mehrere Lernaktivitäten beschreiben. Als weiterer Bestandteil des CAM, ist das SCO definiert als Sammlung von Ressourcen und repräsentiert ein selbständig ausführbares Lernobjekt. Innerhalb eines Spiel-basierten Lernprogramms könnte so ein SCO einen Level oder ein

Szenario des Spiels definieren und jeweils mit eigenen Parametern geladen werden, wobei die Ressourcen innerhalb des Content Packages von den SCOs gemeinsam genutzt werden könnten. Die Navigation zwischen einzelnen SCOs wird durch die Sequencing Regeln, beschrieben im SCORM SN, bestimmt. Die Regeln legen fest in welcher Reihenfolge die Lerninhalte an den Lernenden ausgeliefert werden sollen. Zusätzlich kann dabei auch der Lernfluss und die Verzweigung der Lerninhalte beachtet werden. In Spiel-basierten Lernprogrammen könnten die Sequencing Regeln somit bestimmen wann ein Level bzw. Szenario geladen werden soll und welche Bedingungen dafür erfüllt werden müssen.

4.3 Konzept 1: Integration eines Spiel-basierten Lernprogramms in eine SCORM-konforme Lernumgebung

Ein mögliches Konzept zur Nutzung Spiel-basierter Lernprogramme ist die Integration in eine vorhandene SCORM-konforme Lernumgebung. Dabei wird vorausgesetzt das diese Lernumgebung die in Abschnitt 3.3.1 erläuterten LMS Conformance Requirements erfüllt und somit SCORM-konform ist. Gegenüber der SCORM-konformen Lernumgebung verhält sich das Spiel-basierte Lernprogramm in diesem Fall wie eine klassische Lerneinheit oder ein Lernobjekt. Eine Anpassung bzw. Erweiterung der SCORM-Spezifikation ist also nicht notwendig.

4.3.1 Anpassungen

Um SCORM Konformität zu erreichen, muss das Lernprogramm die entsprechenden Conformance Requirements (vgl. Abschnitt 3.3.2) umsetzen. Dabei ist entscheidend, wie das Lernprogramm aufgebaut ist bzw. verwendet werden soll (vgl. Abschnitt 4.2). Entsprechend muss das Lernprogramm entweder die Content Package Conformance Requirements und die SCO Conformance Requirements, oder auch nur die SCO Conformance Requirements, umsetzen. Im Folgenden werden die aus den Conformance Requirements resultierenden Anpassungen für Spiel-basierte Lernprogramme erläutert. Zunächst in Tabelle 4.1 für den Fall, dass das Spiel sich in Form eines SCO in ein klassisches Lernprogramm einfügt oder ein SCO einem einzelnen Level bzw. Szenario des Spiels entspricht. Die weiteren notwendigen Anpassungen für den Fall, dass das Spiel-basierte Lernprogramm als eigenständige Lerneinheit in Form eines Content Package vorliegt, erläutert Tabelle 4.2.

Tabelle 4.1: Anpassung Spiel-basierter Lernprogramme (SCO)

Launch	
	Nachdem das SCO durch das LMS geladen wurde, muss es die API Instanz des LMS finden.
	Außerdem müssen mindestens die Methoden `Initialize()` und `Terminate()` zum Initiieren und Beenden der Kommunikation mit der API aufgerufen werden können.
API	
	Laut Spezifikation können SCOs mit einem LMS kommunizieren. Die Methoden müssen bei Verwendung entsprechend der jeweiligen Anforderungen umgesetzt werden. Die Methode `GetValue()` dient beispielsweise zum Auslesen von Data Model Elementen.
	Zur Nutzung der API Methoden wird außerdem festgelegt, dass das SCO eine Scriptsprache auf Basis des ECMAScript-Standards (JavaScript) verwenden sollte.
Data Model	
	Ein SCO muss prinzipiell keine Daten mit einem LMS austauschen. Findet dennoch ein Datenaustausch statt, so sind die Data Model Elemente, deren Verwendung sowie die zugeordneten Datentypen genau spezifiziert.
	Zu den Data Model Elementen gehören beispielsweise Elemente, die den Lernenden anhand einer ID `cmi.learner_id` identifizieren oder die Punktzahl `cmi.score` übermitteln.

Tabelle 4.2: Anpassung Spiel-basierter Lernprogramme (Content Package)

Content Package	
	Ein Content Package muss neben einem Manifest unter dem Namen „imsmanifest.xml“ in seinem Wurzelverzeichnis auch mindestens ein SCO oder ein Asset enthalten.
	Das PIF, in welches das Content Package gepackt wird, muss im Archivformat PKZIP Version 2.04g vorliegen.
Manifest	
	Das Content Package Manifest selbst sowie die enthaltenen Sequencing und Navigation Informationen müssen gültig im Sinne der SCORM Spezifikation sein. Beispielsweise muss das Manifest aus genau einem Wurzelelement `<manifest>` bestehen, welches die Unterelemente `<metadata>`, `<organizations>`, `<resources>` enthält und optional auch weitere Submanifeste und Erweiterungselemente enthalten kann.
	Für Manifeste von Lernressourcen, wie SCOs und Assets, gelten ähnliche Vorschriften.
Sequencing und Navigation	
	Sollen Sequencing Strategien angewendet werden, so sind die entsprechenden Informationen innerhalb eines `<sequencing>` Elementes abzulegen. Das `<sequencing>` Element ist wiederum ein Unterelement der Elemente `<item>`, `<organization>` oder `<sequencingCollection>`.
	Sollen Navigation Strategien angewendet werden, so sind die entsprechenden Informationen innerhalb eines `<item>` Elementes abzulegen.

4.3.2 Architektur

Die technische Umsetzung dieses Konzepts gestaltet sich einfach und entspricht der Integration klassischer Lerninhalte in SCORM-konformen Lernumgebungen. Das Spiel-basierte Lernprogramm wird somit, je nach Aufbau (vgl. Abschnitt 4.2), wie ein Content Package bzw. ein SCO behandelt und entsprechend integriert. Abbildung 4.1 veranschaulicht die zugrunde liegende Architektur.

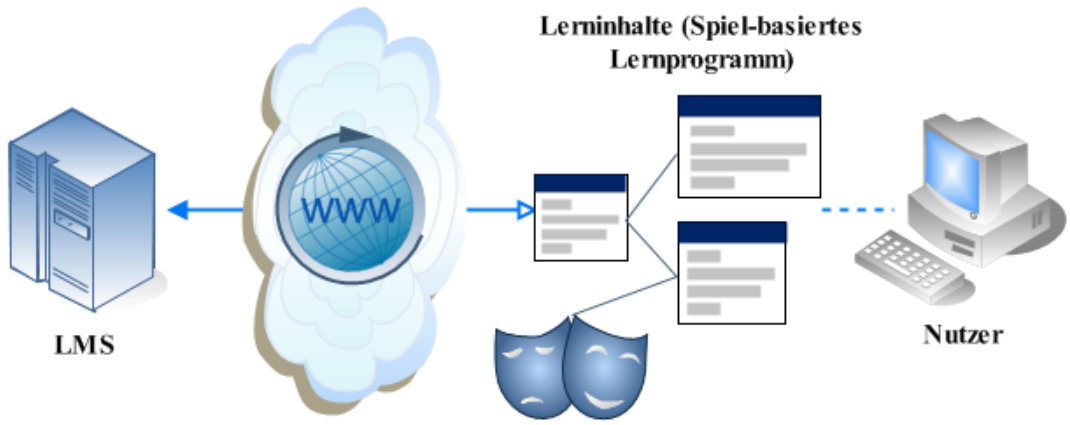

Abbildung 4.1: Integration eines Spiel-basierten Lernprogramms in eine SCORM-konforme Lernumgebung

Innerhalb dieser Architektur steuert das LMS den Lernprozess und kann Informationen über den Lernfortschritt und eventuelle Bewertungen des Nutzers sammeln. Die Kommunikation findet dabei ausschließlich zwischen SCO und LMS, basierend auf der SCORM Spezifikation, statt. Das LMS ist auf Informationen durch die SCOs angewiesen und kann selbst keinerlei Informationen von den Lernobjekten beziehen. Die verfügbaren Daten basieren auf den Elementen des SCORM RTE Data Model und beschränken sich überwiegend auf Informationen zum aktuellen Lernprozess. Obwohl dabei keine speziellen Elemente zur Steuerung Spiel-basierter Lernprogramme oder zum Setzen von Startparametern für einzelne Levels vorgesehen sind, eignen sich doch einzelne Elemente durchaus auch zur Verwendung für Spiel-basierte Lernprogramme. Tabelle 4.3 listet einige Elemente auf und erläutert beispielhaft deren mögliche Verwendung. Eine Übersicht über alle verfügbaren Elemente des SCORM RTE Data Model befindet sich in Anhang A und im vierten Kapitel des SCORM RTE Dokuments. Innerhalb des SCORM RTE Dokuments wird die Implementation der Elemente außerdem detailliert beschrieben.

Tabelle 4.3: Verwendung der SCORM RTE Data Model Elemente für Spiel-basierte Lernprogramme

Data Model Element	Beschreibung und mögliche Verwendung
Completion Status	Gibt an, ob der Lernende das SCO vollständig abgeschlossen hat.
	Steht das SCO innerhalb eines Spiels beispielsweise für einen Level, kann über dieses Element festgehalten werden, dass der jeweilige Level bereits vollständig abgeschlossen ist.
Interactions	Wird zum Ablegen von Informationen, die Interaktionen des Lernenden mit dem SCO betreffen, benutzt mit der Absicht, Interaktionen des Lernenden zu bewerten und einzuschätzen.
	Interaktionen des Spielers innerhalb des Spiels, wie beispielsweise die Lösung eines vorgegebenen Rätsels, kann mit Hilfe dieses Elementes für eine spätere Auswertung festgehalten werden.
Launch Data	Enthält SCO spezifische Daten, die ein SCO zur Initialisierung nutzen kann, festgelegt innerhalb des Content Package Manifests.
	Bei der Erstellung eines Kurses mit einem eingebundenen Spiel, kann über dieses Element beispielsweise festgelegt werden, in welchem Schwierigkeitsgrad das Spiel innerhalb des Kurses ausgeführt wird.
Location	Repräsentiert eine bestimmte Stelle innerhalb des SCO.
	Das Element kann verwendet werden um nach einer Unterbrechung an eine bestimmte Stelle (Level) des Spiels zurückzukehren.
Maximum Time Allowed	Gibt die maximale Zeitspanne zur Benutzung eines SCO durch den Lernenden innerhalb eines Versuchs an.
	Kann verwendet werden, um festzuhalten, in welcher Zeit ein Spieler das Spiel oder den Level absolvieren muss.
Objectives	Spezifiziert beliebige Lern- oder Leistungszielsetzungen des SCO.
	Objectives können sehr vielfältig und flexibel zum Verfolgen von Zielsetzungen innerhalb eines Spiels genutzt werden.
Success Status	Gibt an, ob der Lernende das SCO erfolgreich abgeschlossen hat.
	Kann wie das Element Completion Status verwendet werden. Mit dem Unterschied, dass damit beschrieben werden kann, ob der Level auch erfolgreich abgeschlossen wurde.

Die SCORM Spezifikation enthält für viele SCORM RTE Data Model Elemente nur sehr wenige Vorgaben bezüglich deren Verwendung. So wird Entwicklern von Lerninhalten eine recht flexible Nutzung der Elemente ermöglicht, wie Tabelle 4.3 in Bezug auf Spiel-basierte Lernprogramme zeigt. Die Übersicht zeigt aber auch, dass die Elemente des SCORM RTE Data Model überwiegend nur für die Strukturierung und die anschließende Bewertung des

Lernprozesses benutzt werden können. Für Spiele unter Umständen relevante Informationen, wie das Erscheinungsbild des Spielcharakters, lassen sich dagegen nur schwer mit den vorhandenen Elementen umsetzen.

4.3.3 Bewertung

Das Konzept der Integration eines Spiel-basierten Lernprogramms in eine SCORM-konforme Lernumgebung unter Nutzung der SCORM Spezifikation ist eine leicht zu realisierende Lösung, die ohne Erweiterungen der Spezifikation umgesetzt werden kann. Das Konzept erfüllt alle Anforderungen aus Abschnitt 3.3 und ist damit SCORM-konform. Zur Übersicht, welche Anforderungen des Anforderungskataloges aus Abschnitt 3.4 das Konzept erfüllt, sei auf Tabelle 3.7 in Abschnitt 3.5 verwiesen. Aufgrund des fehlenden Multi-User Aspektes des integrierten Spiel-basierten Lernprogramms, kann die Tabelle auch zur Veranschaulichung für das vorliegende Konzept verwendet werden.

Zusammenfassend lässt sich sagen, dass das Konzept im Sinne der didaktischen Aspekte des Spiel-basierten Lernens nur bedingt empfohlen werden kann, da es eine kooperative Lernform nicht unterstützt. Bei der Integration eines Spiel-basierten Lernprogramms ohne Multi-User Aspekte in eine SCORM-konforme Lernumgebung, erfüllt das Konzept dagegen alle Anforderungen.

4.4 Konzept 2: Integration eines Spiel-basierten Lernprogramms in eine SCORM-konforme Lernumgebung unter Einsatz eines Multi-User-fähigen Spieleservers

Das bereits erläuterte Konzept ermöglicht die Integration Spiel-basierter Lernprogramme in SCORM-konforme Lernumgebungen. Die erweiterten Anforderungen im Sinne des Anforderungskataloges aus Abschnitt 3.4, vor allem die Unterstützung einer kooperativen Lernform, können damit, wie bereits erläutert, nicht vollständig umgesetzt werden. Ein mögliches Konzept zur Lösung dieser Problematik ist der Einsatz eines zusätzlichen Multi-User Servers der das LMS bei der Erfüllung der erweiterten Anforderungen unterstützt.

Betrachtet man beispielsweise Computerspiele unter diesem Gesichtspunkt, so basieren die meisten auf einer angepassten Spiele-Engine, welche für die Multi-User Unterstützung oder auch komplexe Echtzeitberechnungen verwendet wird. Eine Spiele-Engine in Form eines Servers wird im Allgemeinen als Spieleserver bezeichnet und auch in dieser Arbeit so verwendet. In der Literatur findet man auch die Bezeichnung dedizierter Server.

4.4.1 Anpassung

Neben den Anpassungen zur Erreichung der SCORM-Konformität, erläutert in Abschnitt 4.3.1, muss das Spiel-basierte Lernprogramm bei diesem Konzept zusätzlich auch für den Umgang mit dem Spieleserver angepasst werden. Der Verlauf der Kommunikation läuft dabei prinzipiell wie folgt ab. Der Spieleserver stellt verschiedene Dienste zur Verfügung, die das Spiel-basierte Lernprogramm als Client nutzen kann. Dazu muss es eine Anfrage für einen bestimmten Dienst an den Server senden. Dieser erfüllt den Dienst indem er die nachgefragten Daten oder eine Fehlermeldung zurück liefert.

Der Spieleserver reagiert somit über ein klassisches Client-Server-Modell auf die Interaktion des Spielers mit dem Spiel. Diese Art der Kommunikation zwischen Client und Server nennt man auch Client-Push. Gerade für Multi-User Anwendungen ist eine Erweiterung des klassischen Modells, das sogenannte Server-Push, sehr interessant. Dabei liefert der Server auch ohne eine Anfrage des Clients Daten an diesen aus. So wird der Client zum passiven Bestandteil des Modells und kann beispielsweise umgehend über aktualisierte Daten auf dem Server informiert werden. Damit entfällt das regelmäßige Anfragen des Clients. Vor allem Echtzeitanwendungen profitieren von dieser Erweiterung, bzw. lassen sich so überhaupt erst realisieren. Tabelle 4.4 charakterisiert Client und Server als Bestandteile dieses erweiterten Client-Server-Modells.

Tabelle 4.4: Charakterisierung der Bestandteile eines erweiterten Client-Server-Modells

Server	
	Passiver Bestandteil des Client-Server-Modells, der mit Zuständen arbeiten kann.
	Wartet auf Anfragen eines Clients und beantwortet diese bzw. stellt einen entsprechenden Dienst zur Verfügung. Erweiterung: Auslieferung von Daten auch ohne Anfrage eines Clients (Server-Push).
	Akzeptiert Verbindungen mit einer großen Anzahl Clients.
	Keine direkte Interaktion mit den Nutzern.
Client	
	Aktiver Bestandteil des Client-Server-Modells.
	Initiiert Anfragen und wartet auf die entsprechende Antwort des Servers (Client-Push). Erweiterung: Empfang von Daten ohne vorherige Anfrage an den Server.
	Meist Verbindungsaufbau mit einem Server. Ein Verbindungsaufbau mit mehreren Servern ist aber ebenfalls möglich.
	Interaktion mit dem Nutzer durch eine graphische Benutzerschnittstelle.

4.4.2 Architektur

Die technische Umsetzung dieses Konzepts gestaltet sich komplexer als die Vorhergehende, basiert aber ebenfalls auf der Integration klassischer Lerninhalte in SCORM-konforme Lernumgebungen. Die Architektur, vorgestellt in Abschnitt 4.3.2, wird durch einen zusätzliche Spieleserver erweitert. Jeder Nutzer interagiert gleichzeitig mit dem LMS und dem Spieleserver. Die Kommunikation mit dem Spieleserver kann sehr flexibel gestaltet werden. Somit ermöglicht der Spieleserver unter anderem die Multi-User Unterstützung, welche innerhalb der SCORM Spezifikation nicht realisierbar ist. Abbildung 4.2 veranschaulicht die zugrunde liegende Architektur.

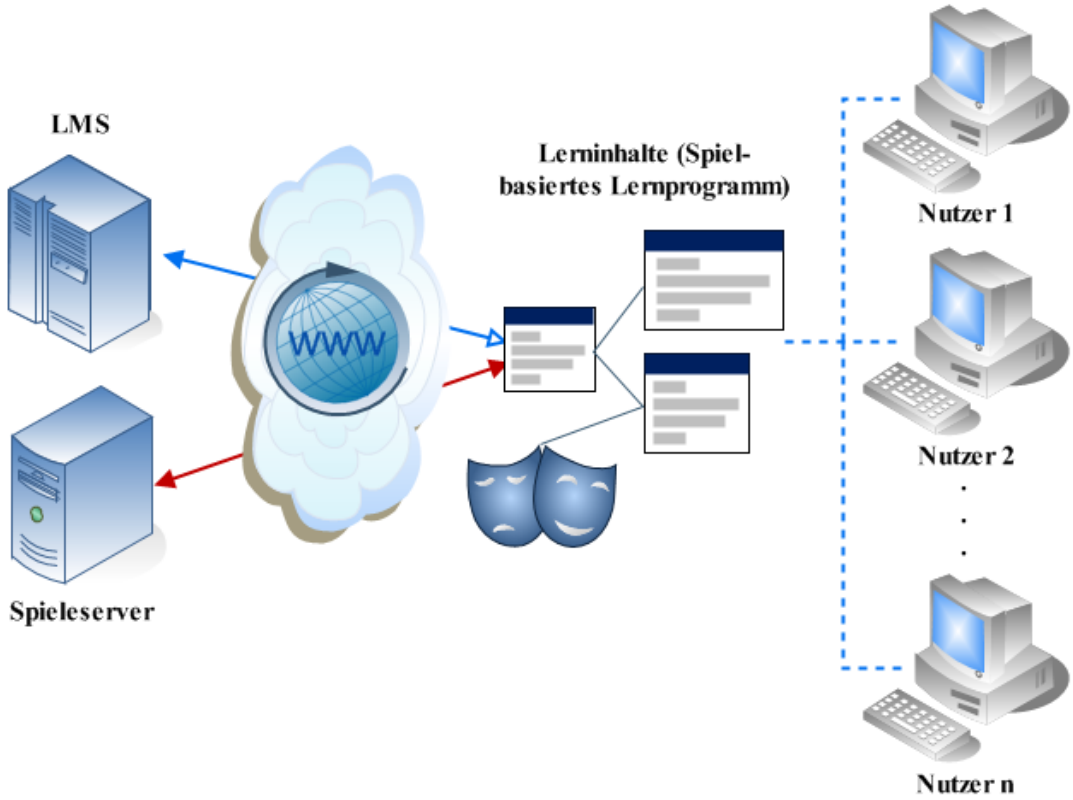

Abbildung 4.2: Integration eines Spiel-basierten Lernprogramms in eine SCORM-konforme Lernumgebung unter Einsatz eines Multi-User-fähigen Spieleservers

Der Spieleserver wird parallel zum LMS ausgeführt und kommuniziert mit dem Spiel-basierten Lernprogramm, über das bereits erläuterte erweiterte Client-Server-Modell, mit dem Nachteil das für die Kommunikation kein standardisiertes Datenmodell existiert. Eine Einflussnahme des LMS auf den Spieleserver ist aufgrund der SCORM-Spezifikation nicht möglich. Ebenso kann der Spieleserver nicht mit dem LMS kommunizieren. Die funktionale Schnittstelle bildet also das Spiel-basierte Lernprogramm in Form des Clients sowohl für das

LMS als auch für den Spieleserver. Dabei existiert weder eine entsprechend standardisierte Schnittstelle mit dem Spieleserver noch ein standardisiertes Datenmodell für die Kommunikation. Im Umgang mit dem LMS verhält sich das Spiel-basierte Lernprogramm entsprechend der SCORM Spezifikation, je nach Aufbau (vgl. Abschnitt 4.2), wie ein Content Package oder wie ein bzw. mehrere SCOs. Die Kommunikation mit dem LMS basiert wiederum auf dem SCORM RTE Data Model, mit den selben Einschränkungen wie in Abschnitt 4.3.2 erläutert.

4.4.3 Bewertung

Das Konzept der Integration eines Spiel-basierten Lernprogramms in eine SCORM-konforme Lernumgebung unter Einsatz eines Multi-User-fähigen Spieleservers, ist im Vergleich zum ersten Konzept schwieriger zu realisieren. Eine Erweiterung der SCORM Spezifikation zur Umsetzung des Konzepts ist ebenfalls nicht nötig. Allerdings müssen die zusätzlichen Funktionalitäten, wie die Unterstützung des Multi-User Aspektes, durch den Spieleserver implementiert werden. Der größte Vorteil liegt dabei in der Kombination der Möglichkeiten von Spieleserver und LMS. So können beispielsweise die in SCORM vorhandenen Data Model Elemente sowie die Möglichkeiten des Sequencing zur Strukturierung und Bewertung bzw. Steuerung des Lernprozesses genutzt werden.

Das Konzept erfüllt alle Anforderungen aus Abschnitt 3.3 und ist somit, was die Integration der Lerninhalte in die Lernumgebung angeht, zunächst SCORM-konform. Eine Übersicht welche Anforderungen aus Abschnitt 3.4 das Konzept erfüllt, bietet Tabelle 4.5 basierend auf dem erarbeiteten Anforderungskatalog. Der Vergleich mit Tabelle 3.8 in Abschnitt 3.5 zeigt die Vorteile dieses Integrationskonzepts für ein Spiel-basiertes Lernprogramm mit Multi-User Aspekt gegenüber einer Integration in eine SCORM-konforme Lernumgebung ohne Einsatz eines Multi-User-fähigen Spieleservers. Das Konzept ermöglicht einen gemeinschaftlich gestalteten Lernprozess, innerhalb dessen Gruppen eingerichtet werden können, die auf gemeinsame Ressourcen zugreifen. Eine Bewertung und Rückmeldung für den Einzelnen in Bezug auf die Gruppe kann so auch realisiert werden. Des Weiteren können die Nutzer innerhalb des Lernprozesses miteinander kommunizieren.

Tabelle 4.5: Erfüllbare Anforderungen (Konzept 2) eines Spiel-basierten Lernprogramms mit Multi-User Aspekt

✓ - erfüllte Anforderung / ✗ - nicht erfüllte Anforderung	
Unterstützung kooperativer Lernformen	
✓	Gemeinschaftlicher Lernprozess in einer verteilten Umgebung
✓	Einrichtung von Gruppen, die gemeinsame Ressourcen teilen
✓	Möglichkeiten zur Kommunikation der Nutzer untereinander
Strukturierung & Anpassung des Lernprozesses	
✓	Strukturierung entsprechend des Spielaufbaus
✓ / ✓	Dynamische Gestaltung des Lernprozesses (individuell/innerhalb einer Gruppe)
✗	Datenmodell zur Strukturierung und Steuerung des Spiels
Erstellung individueller Lernprofile & Bewertung des Nutzers	
✓	Individuelles anpassbares Nutzer- und Lernprofil
✓ / ✓	Kontinuierliche Bewertung und Rückmeldung (individuell/innerhalb einer Gruppe)
✗	Datenmodell für ein Lernprofil sowie zur Bewertung und Rückmeldung
Erweiterung der Kommunikationsmöglichkeiten	
✓	Direkte Kommunikation der Lernobjekte (SCOs) untereinander
✓	Schnittstellen zu spezialisierten Servern und/oder lokalen Anwendungen
✗	Datenmodell zum Informationsaustausch zwischen den Lernobjekten
Nutzung multimedialer Elemente	
✓	Verknüpfung und erweiterte Nutzung multimedialer Elemente
✓	Möglichkeit der Einbindung interaktiver virtueller Lernwelten

Die bereits erläuterten Unzulänglichkeiten des SCORM RTE Data Model in Bezug auf kooperative Lernformen (vgl. Abschnitte 3.5 und 4.3.2) können also ausgeglichen werden. Die Schnittstelle für den Umgang mit dem Spieleserver sowie das verwendete Datenmodell sind aber nicht standardisiert. Zusammenfassend lässt sich sagen, dass das Konzept im Sinne der didaktischen Aspekte des Spiel-basierten Lernens zwar empfohlen werden kann, eine Sicherung der Wiederverwendbarkeit und Interoperabilität der Lerninhalte allerdings nicht gewährleistet ist. Die vollständige SCORM-Konformität wird so nicht erreicht.

4.5 Ausblick: Erweiterung der SCORM Spezifikation

Die Basis, für die beiden in diesem Kapitel vorgestellten Konzepte, ist die SCORM Spezifikation. Neben der bereits erläuterten Sicherung der Wiederverwendbarkeit

(Reusability) und Interoperabilität (Interoperability), werden als Basis der Spezifikation weitere sogenannte ADL-ilities (vgl. Abschnitt 1.4.1) für die am Lernprozess beteiligten Objekte formuliert. Dazu gehören die ortsunabhängige Zugänglichkeit (Accessibility) und die Anpassbarkeit an individuelle Bedürfnisse (Adaptability) genauso, wie die Möglichkeit der effizienten Produktion (Affordability) und die technologische Beständigkeit (Durability). Diese Basis bildet auch die Grundlage für eventuelle Erweiterungen der Spezifikation. Beide Konzepte kommen zunächst ohne eine solche Erweiterung aus. Das zweite Konzept (vgl. Abschnitt 4.4) nutzt allerdings einen zusätzlichen Multi-User-fähigen Spieleserver und umgeht so die Beschränkungen der Spezifikation im Hinblick auf kooperative Lernformen. Die Sicherung der Wiederverwendbarkeit und Interoperabilität der Lerninhalte ist damit nicht mehr gewährleistet. Das Konzept stellt dementsprechend keine endgültige Lösung, sondern lediglich einen Zwischenschritt zu einer Erweiterung der SCORM Spezifikation hinsichtlich kooperativer Lernformen, speziell für das Spiel-basierten Lernen, dar.

4.5.1 Anpassungen

Die technologischen Ziele der ADL Initiative beinhalten bereits eine geplante Erweiterung der SCORM Spezifikation zur Integration von Spiel-basierten Lernprogrammen unter Beachtung des Multi-User Aspektes (vgl. [ADL06c]). Demnach sollen die Voraussetzungen für eine effektive Implementation geschaffen werden. Wie genau diese Implementation aussehen soll, ist bisher allerdings nicht bekannt. In der aktuellen Spezifikation SCORM 2004 3rd Edition gibt es zur Gestaltung gemeinsamer Lernprozesse in verteilten Umgebungen bisher überhaupt keinen Ansatz. Zu den Voraussetzungen kooperativer Lernformen gehört hauptsächlich ein erweitertes Datenmodell. Benötigt werden beispielsweise Elemente zur Strukturierung und Steuerung, Beschreibung eines Lernprofils, Bewertung und Rückmeldung und zum Informationsaustausch zwischen den Lernobjekten innerhalb einer solchen kooperativen Lernform.

Eine Möglichkeit wäre die Bereitstellung eines alternativen Datenmodells. Prinzipiell gibt es keine Beschränkung der Spezifikation bezüglich der Nutzung und Definition anderer Datenmodelle. So basiert das in SCORM verwendete RTE Data Model auf dem IEEE 1484.11.1 Data Model for Content Object Communication Standard und schreibt vor, dass alle verwendeten Elemente das Kürzel `cmi` zur Kennzeichnung benutzen. Ein gültiges Element ist also beispielsweise `cmi.success_status`. Es ist vorgesehen, dass alternative

Datenmodelle ein anderes Kürzel zur Kennzeichnung oder auch eine andere Notation benutzen. Ein Datenmodell zur Unterstützung kooperative Lernformen kann also durchaus definiert werden. Verwendung finden könnte das Datenmodell aber erst, als standardisierter Bestandteil der Spezifikation. Nur so kann schließlich die Wiederverwendbarkeit und Interoperabilität der Lerninhalte gesichert werden. Wie ein solches Datenmodell aussehen kann, soll an dieser Stelle nur grob skizziert werden. Umgesetzt werden kann beispielsweise eine Art Raumkonzept, wie es einige kollaborative Systeme verwenden. Dabei definieren virtuelle Räume gemeinsame Bereiche zum kooperativen Lernen, wobei eine hierarchische Schachtelung der Räume möglich ist. Einzelne Nutzergruppen können diesen Räumen zugeordnet werden und so an einem gemeinschaftlichen Lernprozesse teilnehmen.

Eine andere Möglichkeit ist die Erweiterung des bestehenden SCORM RTE Data Model. Eine solche Erweiterung ist, nach Angaben von ADL, vorerst nicht vorgesehen [ADL06a]. So reagiert ein SCORM-konformes LMS bei der Verwendung nicht definierter Elemente mit einer entsprechenden Fehlermeldung. Im gleichen Zusammenhang bekräftigt ADL allerdings auch, dass man an einer Erweiterung des Datenmodells und Möglichkeiten zur Nutzung und Definition anderer Datenmodelle arbeitet.

4.5.2 Architektur

Bezüglich der technischen Umsetzung einer Erweiterung der SCORM Spezifikation sind verschiedene Möglichkeiten denkbar. Einerseits könnten alle benötigten Funktionalitäten in die SCORM RTE integriert und damit Bestandteil eines LMS werden. Die Spezifikation würde so aber mit der Zeit einen kaum noch überschaubaren Umfang annehmen, da ADL nicht nur Erweiterungen hinsichtlich des Spiel-basierten Lernens plant. Beispielsweise führen [LSAL03] als mögliche Erweiterungen für die Spezifikation unter anderem auch das Mobile Lernen, den Einsatz von Simulationen, Kompetenzbasiertes Lernen sowie Intelligente Tutorielle Systeme an.

Sinnvoller in diesem Zusammenhang wäre der Einsatz einer spezialisierten Zwischenschicht, die alle benötigten Funktionalitäten für das Spiel-basierte Lernen enthält und über standardisierte Schnittstellen mit dem LMS und den Lernobjekten kommuniziert. Die Zwischenschicht könnte für die Lernobjekte einerseits als Spieleserver und andererseits als eine Art Abbild des LMS agieren. Dabei stellt sie den Lernobjekten die entsprechenden Funktionalitäten zur Verfügung und kommuniziert gleichzeitig, basierend auf den

Spezifikationen der SCORM RTE, mit dem LMS. Abbildung 4.3 veranschaulicht die zugrunde liegende Architektur.

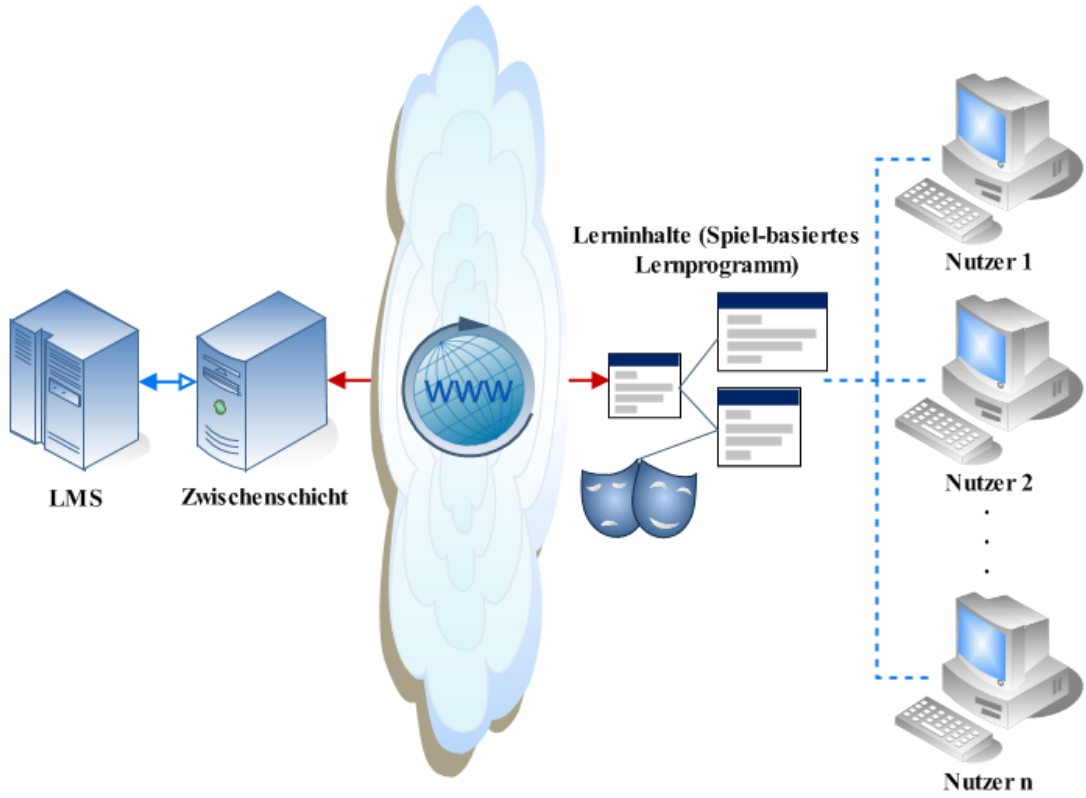

Abbildung 4.3: Einsatz einer Zwischenschicht

Der Vorteil einer solchen Architektur liegt in der Flexibilität dieser Zwischenschicht. Auch die anderen angeführten Erweiterungen abseits des Spiel-basierten Lernens könnten so integriert werden, wobei die ADL-ilities, hauptsächlich die Sicherung der Wiederverwendbarkeit und Interoperabilität der Lerninhalte, gewährleistet wären.

4.6 Zusammenfassung

Die beiden vorgestellten Konzepte bauen aufeinander auf und bilden einen Zwischenschritt zur Erweiterung der SCORM Spezifikation. Das erste Konzept integriert ein Spiel-basiertes Lernprogramm in eine SCORM-konforme Lernumgebung. Die größte Nachteil dabei ist die fehlende Unterstützung des Multi-User Aspektes und dem damit verbundenen gemeinschaftlichen Lernprozess. Im Sinne des Spiel-basierten Lernens ist dieses Konzept demzufolge keine endgültige Lösung, sondern bildet lediglich einen Kompromiss zur Integration Spiel-basierter Lernprogramme unter Beachtung der SCORM Spezifikation.

Das zweite Konzept integriert ein Spiel-basiertes Lernprogramm in eine SCORM-konforme Lernumgebung unter Einsatz eines Multi-User-fähigen Spieleservers. Ein Nachteil dieses Konzepts ist die fehlende Standardisierung von Spieleserversystemen. Modelle zur Sicherung der Interoperabilität und Wiederverwendbarkeit von Spiel-basierten Lerninhalten unter Einsatz eines Spieleservers existieren bisher nicht. Das Spiel-basierte Lernprogramm kann somit nicht ohne vorherige Anpassung an den speziellen Spieleserver in eine SCORM-konforme Lernumgebung integriert werden. Dieses Konzept ist also im Sinne der SCORM Spezifikation keine endgültige Lösung. Es beschreibt allerdings mögliche Ansätze zur Erweiterung der Spezifikation, vor allem die Möglichkeit der Unterstützung kooperativer Lernformen. Im Sinne des Spiel-basierten Lernens erfüllt dieses Konzept die Anforderungen eines Spiel-basierten Lernprogramms, entsprechend der Anforderungsanalyse in Kapitel 3, besser als das Konzept ohne Spieleserver. Eine Erweiterung der SCORM Spezifikation setzt es dabei ebenfalls nicht voraus.

Diese Erweiterung zur Unterstützung kooperativer Lernformen wird in diesem Kapitel zwar als Ausblick ausgeführt, ist aber sowohl technisch als auch zeitlich im Rahmen dieser Arbeit nicht realisierbar. Im nächsten Kapitel wird deshalb, das Konzept zur Integration eines Spiel-basierten Lernprogramms in eine SCORM-konforme Lernumgebung unter Einsatz eines Multi-User-fähigen Spieleservers prototypisch implementiert.

Umsetzung

5 Prototypische Implementierung

5.1 Allgemeines

In Kapitel 4 wurden verschiedene Konzepte zur Integration Spiel-basierter Lernprogramme in SCORM-konforme Lernumgebungen untersucht. Aufbauend auf diesen Untersuchungen soll in diesem Kapitel das Konzept zur Integration eines Spiel-basierten Lernprogramms in eine SCORM-konforme Lernumgebung unter Einsatz eines Multi-User-fähigen Spieleservers (vgl. Abschnitt 4.4) prototypisch implementiert werden. Der Schwerpunkt liegt dabei auf den technischen Aspekten der Integration. Konkretes Ziel der prototypischen Implementierung ist die Erstellung eines Kurses für eine SCORM-konforme Lernumgebung. Als Bestandteil dieses Kurses implementiert eine interaktive Anwendung das Spiel „Drei/Vier/Fünf Gewinnt“ mit drei unterschiedlichen Schwierigkeitsgraden (Levels). Diese Anwendung ermöglicht die Interaktion und Kommunikation mit anderen Nutzern des Kurses, unabhängig von der jeweils verwendeten Lernumgebung. Die Lernumgebung ist im Sinne der SCORM Spezifikation verantwortlich für die Ablaufsteuerung der Anwendung und die Verwaltung der Nutzerdaten.

Nach einer Erläuterung der verwendeten Werkzeuge, beschäftigen sich die nachfolgenden Abschnitte mit der Integration in die Lernumgebung und der Implementation des Client-Server Modells.

5.2 Verwendete Werkzeuge

Für die Implementierung wurden ein Werkzeug zur Erstellung und Bearbeitung der Lerninhalte, ein Editor zur Erstellung von SCORM Content Packages, ein Server zur Entwicklung von Multi-User Anwendungen und Spielen sowie eine Beispiel-Implementierung der SCORM RTE verwendet. Die verwendeten Werkzeuge werden in den folgenden Abschnitten näher erläutert.

5.2.1 Adobe Flash

Flash ist eine integrierte Entwicklungsumgebung der Firma Adobe zur Erstellung multimedialer Webinhalte. Die Anwendung unterstützt die Arbeit mit Audio-, Video- und Animationselementen und hat sich im Bereich des E-Learning vor allem als Autorenwerkzeug für Lerninhalte etabliert. Die erzeugten Inhalte werden über den Flash-Player angezeigt. Dazu müssen die vorhandenen Quelldateien zuvor in SWF-Dateien kompiliert werden. Der für viele

Plattformen verfügbare Flash-Player kann sowohl als eigenständige Anwendung oder aber als Browsers Plug-in ausgeführt werden. Ein wichtiger Bestandteil von Flash ist die Scriptsprache ActionScript. Aufbauend auf dem ECMAScript Standard umfasst ActionScript die nötige Funktionalität und Flexibilität für dynamische und interaktive Anwendungen.

Im Rahmen dieser Arbeit wurde Adobe Flash in der Version Professional 8 verwendet. Es existiert aber bereits eine aktuellere Version CS3 Professional. Diese ist als 30 Tage Testversion auf den Seiten von Adobe [Adobe07] erhältlich und befindet sich zusätzlich auf der, dieser Arbeit, beiliegenden CD.

5.2.2 SmartFoxServer

Der SmartFoxServer der Firma gotoAndPlay() ist eine plattformunabhänige Multi-User-Server-Software, die in Java implementiert ist und auch einen vollständigen Webserver enthält. Entwickelt wurde der SmartFoxServer zur schnellen Realisierung von Flash-basierten Multi-User Anwendungen und Spielen.

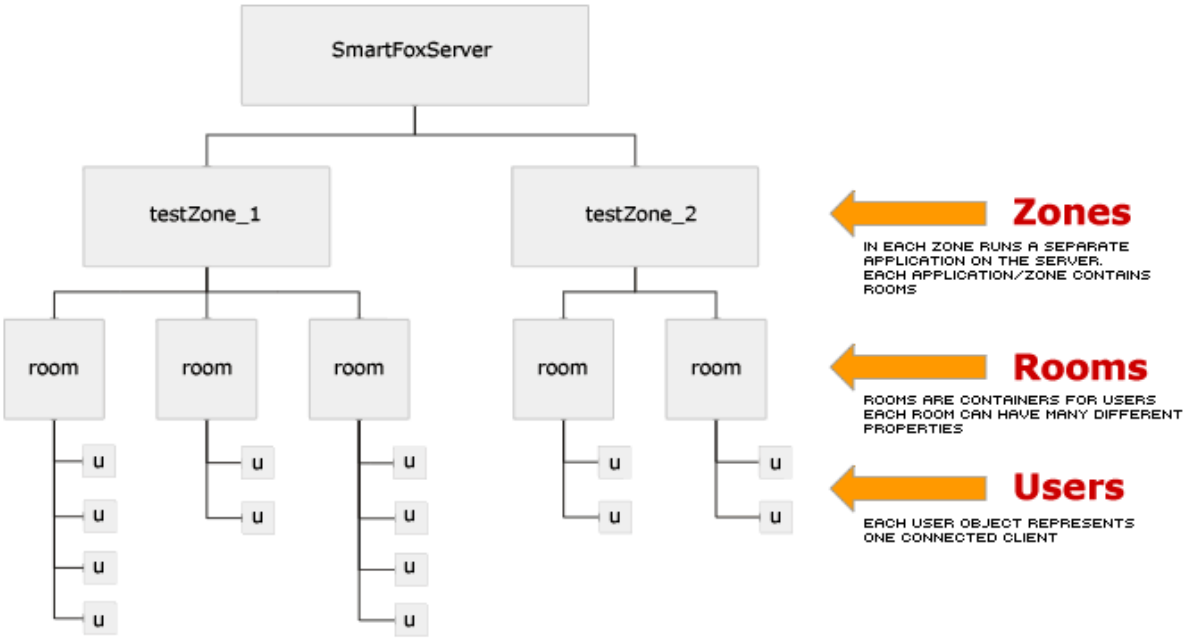

Abbildung 5.1: Datenstruktur des SmartFoxServer [SFS07]

Die Voraussetzung für die Unterstützung des Multi-User Aspektes durch den SmartFoxServer ist seine zugrunde liegende Datenstruktur (vgl. Abbildung 5.1). Ein Raum (Room) ist verantwortlich für die Verbindung und die Kommunikation der Nutzer (User) untereinander und eine entsprechende Ereignisbehandlung. Jeder Raum ist einer Zone (Zone) zugeordnet,

die stellvertretend für eine Anwendung auf dem Server steht. Eine Zone bildet somit einen Container für Räume, während ein Raum, wiederum als Container, Nutzer enthält. Die Nutzer können zwischen den Räumen einer Zone wechseln. Konfiguriert wird der Server über eine serverseitige Konfigurationsdatei, die neben allgemeinen Einstellungen des Servers auch die Einrichtung von Zonen und untergeordneten Räumen ermöglicht. Es existieren aber auch Methoden zur Erstellung von Räumen zur Laufzeit. Bei komplexeren Anwendungen kann der Server außerdem durch Erweiterungen in verschiedenen Programmiersprachen serverseitig ergänzt werden. Die benötigten Funktionen, zur Verbindung und Kommunikation des Clients mit dem Server, stellt schließlich eine Client API bereit. Je nach Bedarf enthält der SmartFoxServer eine Client API für ActionScript 1.0, 2.0 oder 3.0.

Im Rahmen dieser Arbeit wurde die aktuelle Version 1.5.5 des SmartFoxServer Basic verwendet. Diese ist erhältlich auf den entsprechenden Seiten der Firma gotoAndPlay() [SFS07] und befindet sich zusätzlich auf der, dieser Arbeit, beiliegenden CD. Diese kostenlose Version ist auf maximal 20 Verbindungen beschränkt. Die in der prototypischen Implementation verwendete Konfigurationsdatei befindet sich ebenfalls auf der CD und in Anhang E.

Beispielanwendung

Im Rahmen eines Tutorials für den SmartFoxServer wird die Erstellung der Multi-User Beispielanwendung „SmartFoxTris“ erläutert. Das Tutorial, zu finden unter dem Namen „Board Game“, ist Teil der Dokumentation für den SmartFoxServer.

Das Spiel läuft wie folgt ab: Die Anwendung startet zunächst mit einem Anmeldebildschirm. Die Spieler melden sich mit ihrem Namen an. Nach der Anmeldung gelangen sie in einen gemeinsamen Raum. Dort haben sie die Möglichkeit mit anderen Spielern zu kommunizieren oder einen eigenen Spielraum anzulegen. Das eigentliche Spiel findet in diesem abgeschlossenen für zwei Spieler zugänglichen Spielraum statt. Die Spieler können auf einem aus 3x3 Feldern bestehenden Spielfeld, jeweils pro Runde, ein Feld markieren. Der Spieler der zuerst drei Felder waagerecht nebeneinander, senkrecht untereinander oder diagonal markiert hat, gewinnt das Spiel. Auch während des Spiels ist es möglich mit anderen Spielern zu kommunizieren.

5.2.3 Reload Editor

Der Reload Editor ist eine Java-basierte Anwendung des Reload Projektes. Reload steht für *Reusable eLearning Object Authoring & Delivery*. Das Projekt beschäftigt sich mit der Entwicklung von Werkzeugen, basierend auf den aktuellen Standards und Spezifikationen des E-Learning. Der Editor unterstützt den Anwender bei der Erstellung und Beschreibung von Content Packages. Dazu wird aus vorhandenen Lerninhalten eine Content Organization mit Angaben zum Sequencing und Metadaten erstellt und anschließend entsprechend der SCORM Spezifikation ein Content Package generiert.

Im Rahmen dieser Arbeit wurde die aktuelle Version 2.5.4 des Reload Editors verwendet. Diese ist kostenlos erhältlich auf den Seiten des Reload Projektes [REL07] und befindet sich zusätzlich auf der, dieser Arbeit, beiliegenden CD. Die Version unterstützt erstmals auch die erweiterten Konzepte der SCORM 2004 3rd Edition Spezifikation.

5.2.4 ADL Sample Run-Time Environment

Die von ADL zur Verfügung gestellte SCORM 2004 3rd Edition Sample RTE implementiert beispielhaft die in der SCORM Spezifikation beschriebenen Konzepte, speziell die SCORM RTE (API und Data Model) sowie das SCORM SN. Dem Entwickler wird durch die Sample RTE ermöglicht, seine SCORM-konformen Lerninhalte auch ohne vorhandenes LMS zu testen. Dabei unterstützt diese alle grundlegenden Funktionen eines LMS, wie das Importieren und Ausliefern von SCORM Content Aggregation Content Packages, die Kommunikation zwischen Lernobjekten und LMS, die vollständige Implementierung des SCORM RTE Data Model und das Sequencing, bzw. die Navigation, von Lerninhalten. Zusätzlich bietet die Sample RTE Funktionen zur Verwaltung der Kurse und Nutzer.

Realisiert ist die Sample RTE als webbasierte Client-Server Anwendung. Die Server Komponente ist in Java unter Einsatz von Java Servlets implementiert. Die Servlets antworten auf Anfragen der Client Komponente und sind verantwortlich für die Persistenz der Data Model Elemente und die Umsetzung des Sequencing and Navigation. Die Client Komponente besteht aus einer Benutzerschnittstelle, implementiert in HTML und JavaScript, und einer Instanz der Sample RTE API, implementiert als Java Applet. Bei Zugriff auf die Sample RTE wird das Applet beim Client ausgeführt und ermöglicht die Kommunikation zur Server Komponente. Abbildung 5.2 verdeutlicht die Funktionsweise, wobei die Sample RTE sich wie das LMS verhält.

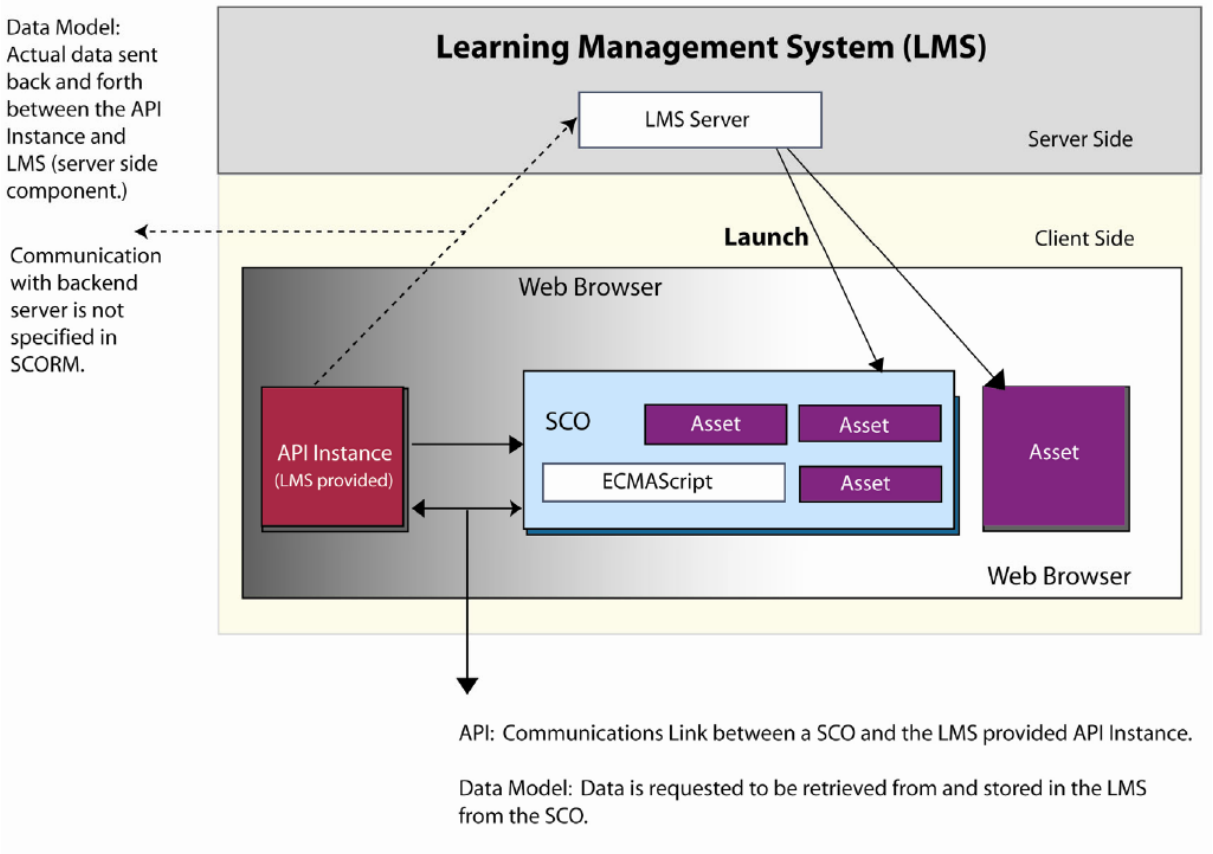

Abbildung 5.2: Funktionsweise der SCORM Sample RTE [ADL06a]

Im Rahmen dieser Arbeit wurde die aktuelle Version 1.0.1 der SCORM 2004 3rd Edition Sample RTE verwendet. Diese ist erhältlich auf den Seiten der ADL Initiative [ADL07] und befindet sich zusätzlich auf der, dieser Arbeit, beiliegenden CD.

5.3 Integration in die Lernumgebung

Zur Verwendung innerhalb einer SCORM-konformen Lernumgebung müssen die Lerninhalte als Content Package (vgl. Abschnitt 1.4.2) in Form eines PIF vorliegen. Neben den physischen Dateien enthält ein Content Package auch ein Manifest zur Beschreibung und Organisation des Inhalts. Das verwendete Content Package befindet sich auf der, dieser Arbeit, beiliegenden CD. Die Manifest Datei befindet sich außerdem in Anhang B.

In den folgenden Abschnitten wird die Erstellung des Kurses erläutert. Dabei wird zum Einen die Erstellung des Content Packages und zum Anderen die Steuerung des Kurses beschrieben.

5.3.1 Erstellung des Content Packages

Das Content Package wurde mit Hilfe des Reload Editors aus der Vorlage *ADL SCORM 2004 Package* erstellt. Abbildung 5.3 zeigt das Manifest des Content Packages im Reload Editor.

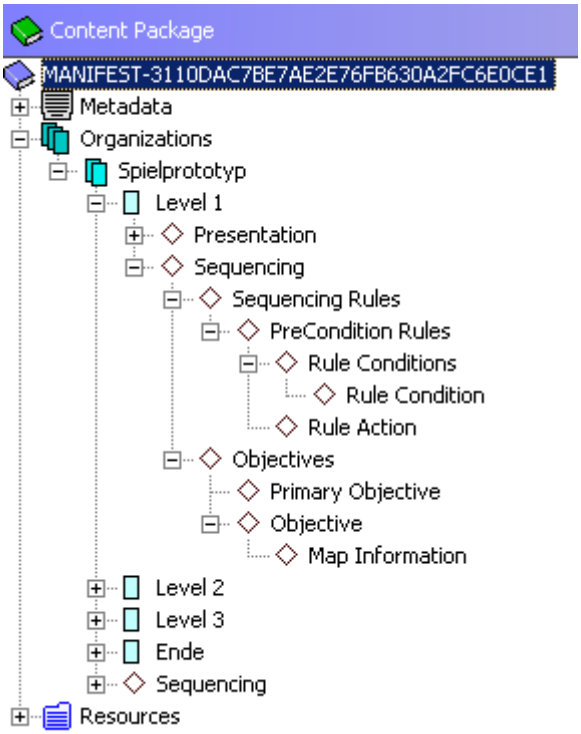

Abbildung 5.3: Content Package im Reload Editor

Der Vorlage wurden zunächst die benötigten physischen Dateien als Ressourcen hinzugefügt. Listing 5.1 zeigt den Abschnitt `<resources>` der Manifest Datei.

Listing 5.1: Abschnitt `<resources>` der Manifest Datei

```
<resources>
  <resource identifier="RES1" adlcp:scormType="sco" type="webcontent"
            href="bin/game.html">
    <file href="bin/game.html" />
    <file href="bin/game.swf" />
    <file href="scripts/APIWrapper.js" />
  </resource>
  <resource identifier="RES2" adlcp:scormType="sco" type="webcontent"
            href="bin/complete.html">
    <file href="bin/complete.html" />
    <file href="bin/complete.jpg" />
    <file href="scripts/APIWrapper.js" />
  </resource>
</resources>
```

Die Ressourcen bestehen aus einer Webseite sowie einer kompilierten Adobe Flash Datei bzw. einer Bilddatei. Die Ressourcen beinhalten zusätzlich eine JavaScript Datei und sind als SCO definiert, da Daten zum LMS übertragen werden sollen. Die erste Ressource repräsentiert die eigentliche Spielanwendung. Die genaue Funktionsweise und der Aufbau der Anwendung werden ausführlich in Abschnitt 5.4.1 erläutert. Die zweite Ressource dient als Hinweis für den Nutzer, wenn alle Levels des Spiels bereits erfolgreich absolviert wurden.

Listing 5.2: Abschnitt `<organizations>` der Manifest Datei

```
<organizations default="ORG">
  <organization identifier="ORG" structure="hierarchical">
    <title>Spielprototyp</title>
    <item identifier="ITEM-LEVEL1" isvisible="true" identifierref="RES1">
      <title>Level 1</title>
    </item>
    <item identifier="ITEM-LEVEL2" isvisible="true" identifierref="RES1">
      <title>Level 2</title>
    </item>
    <item identifier="ITEM-LEVEL3" isvisible="true" identifierref="RES1">
      <title>Level 3</title>
    </item>
    <item identifier="ITEM-ENDE" isvisible="true" identifierref="RES2">
      <title>Ende</title>
    </item>
  </organization>
</organizations>
```

Nach dem Einfügen der Ressourcen wurde die Strukturierung des Content Packages festgelegt. Listing 5.2 zeigt den Abschnitt `<organizations>` der Manifest Datei. Das Element `<organization>` bildet den gesamten Kurs, bzw. das Spiel, als Activity ab. Dem untergeordnet sind die Activities „Level 1“, „Level 2“, „Level 3“ und „Ende“ als `<item>` Elemente mit den entsprechend zugeordneten Ressourcen. Die ersten drei Activities stehen für die drei möglichen Levels des Spiels und nutzen die erste Ressource. Die vierte Activity beinhaltet schließlich den Hinweis und nutzt dementsprechend die zweite Ressource.

Auf die Beschreibung der einzelnen Elemente des Content Packages durch Metadaten wurde größtenteils verzichtet, da diese für die Funktionsweise der prototypischen Implementierung nicht von Bedeutung sind.

5.3.2 Steuerung des Kurses

Die Möglichkeiten zur Steuerung eines Kurses sind im SCORM SN Dokument erläutert (vgl. Abschnitt 1.4.4). Zur technischen Umsetzung ist dabei vor allem das Sequencing Definition Model von Bedeutung. Mit den darin enthaltenen Elementen lässt sich eine genaue Ablaufsteuerung definieren. Auch die Sequencing Informationen werden in der Manifest Datei des Content Packages abgelegt. Das Sequencing bezieht sich dabei immer nur auf Activities. Die Angaben zum Sequencing, beschrieben durch das Element `<imsss:sequencing>`, können jedem `<item>` Element und dem `<organization>` Element untergeordnet werden. Diesem Element kann dann wiederum ein Element des Sequencing Definition Model untergeordnet werden. Das Präfix `imss` steht dabei für einen Namensraum und deutet auf die Herkunft der Elemente aus der IMS Simple Sequencing Spezifikation.

Die Steuerung des Kurses wurde zunächst als Sequencingstrategie ausformuliert. Diese wurde dann mit Hilfe der Elemente S*equencing Control Modes*, *Objectives* und *Sequencing Rules* des Sequencing Definition Model umgesetzt. Eine ausführliche Erläuterung des gesamten Sequencing Definition Models befindet sich im dritten Abschnitt des SCORM 2004 3rd Edition Sequencing and Navigation Dokuments. Die umgesetzte Sequencingstrategie hat auch Einfluss auf die Spielanwendung selbst. Die Anwendung nutzt die in Abschnitt 5.4.2 beschriebenen Möglichkeiten und ist so dank der Sequencinginformationen in der Lage herauszufinden, welcher Level gerade angezeigt werden soll.

5.3.2.1 Sequencingstrategie

Wie in der als Vorlage verwendeten Beispielanwendung (vgl. Abschnitt 5.2.2) erscheint beim Starten des Kurses direkt der Anmeldebildschirm für das Spiel „Drei/Vier/Fünf Gewinnt“. Der Nutzer wird je nach Level mit anderen Nutzern gleichen Levels verbunden. Dabei besteht keine Möglichkeit über das Menü der Lernumgebung einen bestimmten Level auszuwählen oder sich mit den Navigationselementen der Lernumgebung durch die Levels zu bewegen. Lediglich das Navigationselement zum Abbrechen des Kurses steht zur Verfügung. In den nächsten Level des Spiels gelangt der Nutzer nur, wenn er ein Spiel gewinnt. In diesem Fall gelangt er sofort wieder zum Anmeldebildschirm für den nächsten Level. Wurde auch im dritten und somit letzten Level ein Spiel gewonnen, bekommt der Nutzer den Hinweis, dass er alle Levels erfolgreich absolviert hat. Der Kurs ist damit vollständig abgearbeitet und

bestanden. Schafft der Nutzer es nicht alle Levels des Spiels zu absolvieren oder wird der Kurs vorher abgebrochen, gelangt er beim erneuten Starten des Kurses wiederum direkt zum Anmeldebildschirm für den jeweiligen Level.

5.3.2.2 Sequencing Control Modes

Sequencing Control Modes können die Anzeige der Navigationselemente der Lernumgebung beeinflussen. Sie können jeder Activity zugeordnet werden. Listing 5.3 zeigt die verwendeten Sequencing Control Modes in der Manifest Datei.

Listing 5.3: Sequencing Control Modes

```
<organizations default="ORG">
  <organization identifier="ORG" structure="hierarchical">
    <title>Spielprototyp</title>
    <imsss:sequencing>
      <imsss:controlMode choice="false" choiceExit="false" flow="true"
                         forwardOnly="true"/>
    </imsss:sequencing>
  </organization>
</organizations>
```

Das Element zur Beschreibung der Sequencing Control Modes in der Manifest Datei ist `<imsss:controlMode>`, welches in diesem Beispiel dem Element `<organization>` untergeordnet ist. Damit gelten die Angaben für den gesamten Kurs. Das Element enthält selbst keine Unterelemente sondern nur optionale Attribute. In diesem Beispiel verhindern die Attribute `choice="false"` und `choiceExit="false"` die Anzeige der Navigationselemente der Lernumgebung, indem sie der Lernumgebung mitteilen, dass der Nutzer die einzelnen Activities nicht frei auswählen kann. Das Attribut `flow="true"` legt fest, dass die vorhandenen Activities durchlaufen werden können. Allerdings nur vorwärts, da das Attribut `forwardOnly="true"` gesetzt wurde.

5.3.2.3 Objectives

Mit Hilfe von Objectives kann man Statuswerte innerhalb einer Activity speichern. Man unterscheidet globale und lokale Objectives, je nachdem ob die gespeicherten Werte für alle Activities zugänglich sind oder nicht. Globale Objectives eignen sich somit sehr gut zum Festhalten des Lernerfolgs, oder der bereits absolvierten Levels, innerhalb des Spiels. Listing 5.4 zeigt die Verwendung von Objectives innerhalb der ersten Activity in der Manifest Datei.

Listing 5.4: Objectives

```
<item identifier="ITEM-LEVEL1" isvisible="true" identifierref="RES1">
  <title>Level 1</title>
  <imsss:sequencing>
   <imsss:objectives>
     <imsss:primaryObjective />
     <imsss:objective objectiveID="level_1">
       <imsss:mapInfo readSatisfiedStatus="true"
                      targetObjectiveID="global_level_1"
                      writeSatisfiedStatus="true" />
     </imsss:objective>
   </imsss:objectives>
  </imsss:sequencing>
</item>
```

Die entsprechenden Elemente zur Beschreibung der Objectives sind dem Element `<imsss:objectives>` untergeordnet. Im vorliegenden Beispiel wird durch das Element `<imsss:objective objectiveID="level_1">` zunächst das lokale Objective `level_1` innerhalb der Activity angelegt. Anschließend wird über das Attribut `readSatisfiedStatus="true"` festgelegt, dass der Status des globalen Objective `global_level_1` auf das lokale Objective `level_1` übertragen wird. Das ist notwendig, da die im nächsten Abschnitt erläuterten Sequencing Rules nur Objectives der jeweiligen Activity verwenden können. Das Attribut `writeSatisfiedStatus="true"` legt weiterhin fest, dass der Status des lokalen Objective `level_1` nach dem Beenden der Activity wiederum auf das globale Objective `global_level_1` übertragen wird. Die Spielanwendung kann den Status des lokalen Objectives abfragen und auch setzten (vgl. Abschnitt 5.4.2), wodurch das entsprechende globale Objective für den Level beeinflusst wird. Die Auswertung des lokalen Objective übernehmen die für die jeweilige Activity festgelegten Sequencing Rules, erläutert im nächsten Abschnitt.

5.3.2.4 Sequencing Rules

Für jede Activity können eine oder mehrere Sequencing Rules definiert werden. Damit kann man beispielsweise festlegen, wann Activities übersprungen oder verlassen werden sollen. Das entsprechende Element in der Manifest Datei ist `<imsss:sequencingRules>`. Listing 5.5 zeigt dessen Anwendung innerhalb der ersten Activity.

Listing 5.5: Sequencing Rules

```
<item identifier="ITEM-LEVEL1" isvisible="true" identifierref="RES1">
  <title>Level 1</title>
  <imsss:sequencing>
    <imsss:sequencingRules>
      <imsss:preConditionRule>
        <imsss:ruleConditions>
          <imsss:ruleCondition condition="satisfied"
                               referencedObjective="level_1"/>
        </imsss:ruleConditions>
        <imsss:ruleAction action="skip"/>
      </imsss:preConditionRule>
    </imsss:sequencingRules>
  </imsss:sequencing>
</item>
```

In diesem Beispiel wird durch das Element `<imsss:preConditionRule>` eine Regel erstellt, die vor der Ermittlung der nächsten auszuführenden Activity ausgewertet wird. Die Bedingung `<imsss:ruleCondition condition="satisfied" referenced Objective="level_1"/>` besagt, dass die folgende Aktion ausgeführt werden soll, wenn das lokale Objective `level_1` den Status `satisfied` besitzt. Das Element `<imsss:ruleAction action="skip" />` enthält die auszuführende Aktion. Die Aktion `skip` überspringt die aktuelle Activity bei der Ermittlung der nächsten auszuführenden Activity. Diese Sequencing Rule verhindert somit, dass ein bereits absolvierter Level noch einmal ausgeführt wird.

5.4 Implementierung des Client-Server Modells

Das Client-Server Modell der prototypischen Implementierung besteht grundlegend aus drei Teilen. Serverseitig werden der SmartFoxServer als Spieleserver und die Sample RTE als LMS ausgeführt. Auf der Clientseite wird eine Spielanwendung, in Form eines SCO, in einem Browser ausgeführt. Die Anwendung fungiert nach dem in Abschnitt 4.4 erarbeiteten Konzept als funktionale Schnittstelle zwischen Spieleserver und LMS. Die beiden Server sind dabei völlig unabhängig voneinander ausführbar. So kann die Sample RTE durch jede SCORM-konforme Lernumgebung ersetzt werden. Aufgrund der fehlenden standardisierten Schnittstelle bei der Verwendung eines Spieleservers ist die Verwendung des SmartFoxServer für diese prototypische Implementierung dagegen unabdingbar. Bei der Verwendung eines anderen Spieleservers müsste die Anwendung entsprechend angepasst werden.

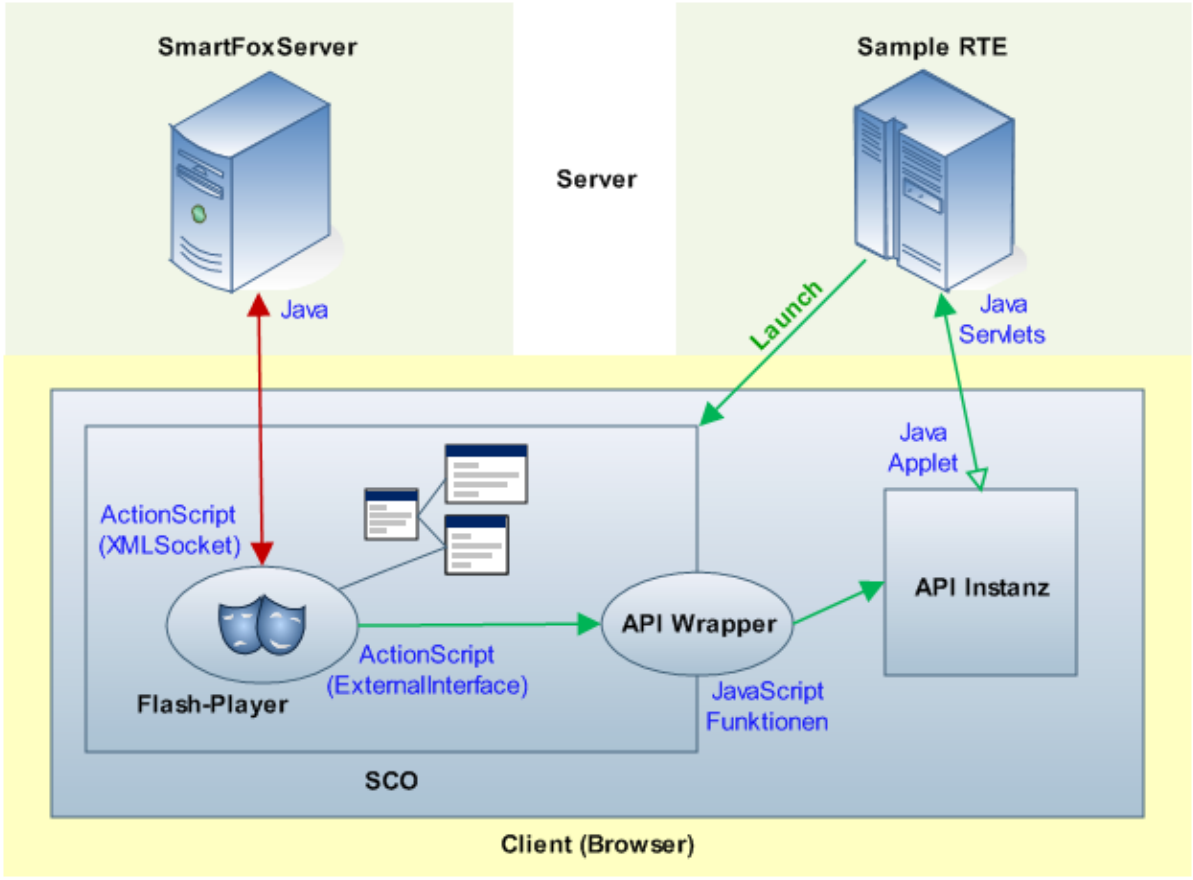

Abbildung 5.4: Implementiertes Client-Server Modell

Abbildung 5.4 zeigt eine schematische Darstellung des implementierten Client-Server Modells. Die Kommunikation des Clients mit der Sample RTE ist in der Abbildung durch die grünen Pfeile dargestellt. Die Kommunikation mit dem SmartFoxServer ist durch den roten Pfeil dargestellt. Beide Möglichkeiten der Kommunikation sowie die Spielanwendung selbst werden in den nachfolgenden Abschnitten erläutert.

5.4.1 Spielanwendung

Die für die prototypische Implementierung eingesetzte Anwendung basiert auf der in Abschnitt 5.2.2 beschriebenen Beispielanwendung. Für die Integration in eine SCORM-konforme Lernumgebung wurde die Beispielanwendung allerdings etwas angepasst. Der Spieler gelangt beim Start der Anwendung wiederum zu einem Anmeldebildschirm. Sein in der Lernumgebung hinterlegter Name ist bereits eingetragen. Der Spieler sieht zudem in welchem Level des Spiels er sich befindet und kann die Anwendung starten. Entsprechend seiner bereits absolvierten Levels gelangt der Spieler in einen Chatraum. Dort hat er die

Möglichkeit mit anderen Spielern gleichen Levels zu kommunizieren oder einen eigenen Spielraum anzulegen. Abbildung 5.5 zeigt den Chatraum der Spielanwendung.

Abbildung 5.5: Chatraum der Spielanwendung

Ein Spielraum ist immer nur für maximal zwei Spieler zugänglich. Der Aufbau des Spielfeldes richtet sich nach dem Level der Spieler. Insgesamt gibt es drei Levels des Spiels. Ausgangspunkt ist ein Spielfeld aus 3x3 Feldern. Bei jedem weiteren Level kommt in jeder Spalte und Zeile des Spielfeldes ein Feld hinzu (Level 1: 3x3 Felder, Level 2: 4x4 Felder, Level 3: 5x5 Felder). Wie in der Beispielanwendung können die Spieler auf dem Spielfeld jeweils pro Runde ein Feld markieren. Der Spieler, der zuerst drei, vier bzw. fünf Felder waagerecht nebeneinander, senkrecht untereinander oder diagonal markiert hat, gewinnt.

Auch während des Spiels ist es möglich, mit anderen Spielern zu kommunizieren. Gewinnt ein Spieler das Spiel, wird das Ergebnis in der Lernumgebung abgelegt und der Spieler kommt automatisch ins nächste Level. Hat er bereits alle Level des Spiels erfolgreich absolviert, bekommt er einen entsprechenden Hinweis. Abbildung 5.6 zeigt den Spielraum der Spielanwendung.

Abbildung 5.6: Spielraum der Spielanwendung

Die Anpassungen der Beispielanwendung wurden mit Adobe Flash implementiert. Die daraus entstandene Anwendung ist in kompilierter Form in eine HTML-Webseite eingebettet. Diese Webseite bildet innerhalb des Content Packages ein SCO (vgl. Abschnitt 5.3). Gemäß der SCORM Spezifikation wird das SCO von der Sample RTE geladen und in einem Browser angezeigt. Ausgeführt wird die Anwendung vom Flash-Player in Form eines Browsers Plug-ins. Die, für die Umsetzung des Spiels und die Kommunikation mit den Servern, notwendige Funktionalität wurde durch den Einsatz von ActionScript implementiert. Neben der eingebetteten Anwendung wurden zusätzlich verschiedene JavaScript Funktionen in die Webseite eingebunden. Die Anwendung nutzt diese Funktionen zur Kommunikation mit der Sample RTE über die API Instanz, in dem sie die in die Webseite eingebundenen Funktionen mit Hilfe einer ActionScript Klasse aufruft. In den nächsten Abschnitten wird die Umsetzung der Kommunikation mit der Sample RTE bzw. dem SmartFoxServer ausführlich erläutert.

5.4.2 Kommunikation mit der Sample RTE

Die Schnittstelle für die Kommunikation von Lerninhalten mit dem LMS ist die API. Jedes LMS, so auch die Sample RTE, muss zur Laufzeit eine Instanz der API zur Verfügung stellen. Die einzigen Objekte des Content Models, die über diese Instanz mit einem LMS

kommunizieren können, sind SCOs (vgl. Abschnitt 1.4.2). Die Lerninhalte werden in einen Browser dargestellt, wobei die SCOs durch den Aufruf von ECMAScript-konformen Funktionen (JavaScript) mit der API Instanz kommunizieren können. Die im Folgenden verwendeten JavaScript Funktionen sind, in der von ADL zur Verfügung gestellten Datei „APIWrapper.js“, zusammengefasst und wurden in die Webseite, welche die Spielanwendung umgibt, eingebunden. Die Zusammenstellung bietet dem Nutzer eine übersichtliche und korrekt implementierte Schnittstelle für den Umgang mit der API Instanz. Die Datei befindet sich in Anhang C und innerhalb des Content Packages auf der, dieser Arbeit, beiliegenden CD. Listing 5.6 zeigt als Beispiel die Funktion `initializeCommunication()`. Diese Funktion initialisiert die Kommunikation zwischen SCO und LMS durch Ausführen der API Methode `api.Initialize("")`. Eine Übersicht über alle Funktionen der API befindet sich im dritten Kapitel des SCORM RTE Dokuments.

Listing 5.6: JavaScript Funktion `initializeCommunication()`

```
function initializeCommunication()
{
   var api = getAPIHandle();

   if ( api == null )
   {
      return "false";
   }
   else
   {
      var result = api.Initialize("");

      if ( result != "true" )
      {
         var errCode = retrieveLastErrorCode();

         displayErrorInfo( errCode );

         // may want to do some error handling
      }
   }
   return result;
}
```

Zur vollständigen Umsetzung der gewünschten Funktionalität muss die im Flash-Player ausgeführte Anwendung auch mit dem LMS kommunizieren können. Die Anwendung muss also in der Lage sein die bereits erwähnten JavaScript Funktionen aufzurufen. Dies wird durch die in ActionScript verfügbare Klasse `ExternalInterface` ermöglicht. Die Klasse

ermöglicht die Kommunikation zwischen ActionScript und den eingebetteten Funktionen der Webseite in beide Richtungen. Diese Vorgehensweise wird von Adobe ausdrücklich für die Kommunikation zwischen ActionScript und JavaScript empfohlen. Listing 5.7 enthält alle verwendeten ActionScript Funktionsaufrufe der Spielanwendung.

Listing 5.7: ActionScript Funktionsaufrufe der Spielanwendung

```
import flash.external.ExternalInterface;

ExternalInterface.call("initializeCommunication");


ExternalInterface.call("retrieveDataValue","cmi.learner_name");
ExternalInterface.call("retrieveDataValue","cmi.objectives.0.id");


ExternalInterface.call("storeDataValue","cmi.objectives.0.success_status",
                       "passed");
ExternalInterface.call("storeDataValue","cmi.success_status","passed");
ExternalInterface.call("storeDataValue","cmi.completion_status",
                       "completed");


ExternalInterface.call("storeDataValue","adl.nav.request","continue")
ExternalInterface.call("terminateCommunication")
```

Bevor die Kommunikation aufgebaut werden kann, muss zunächst die API Instanz des LMS gefunden werden. Dazu wird nach dem Importieren der Klasse `ExternalInterface` die bereits erläuterte Funktion `initializeCommunication()` aufgerufen. Die aufgerufenen Funktionen `retrieveDataValue()` und `storeDataValue()` ermöglichen das Abfragen und Ablegen von Werten der Data Model, bzw. Navigation Data Model, Elemente des LMS. Dazu werden als Parameter der Name des Elementes bzw. zusätzlich der zu übermittelnde Wert an die Funktion übergeben. Zu Beginn werden die Werte der Elemente `cmi.learner_name` und `cmi.objectives.0.id` abgefragt. So wird der im LMS hinterlegte Name des Lernenden sowie der aktuellen Level des Spiels ermittelt. Gewinnt der Spieler das Spiel, wird das Element `cmi.objectives.0.success_status` des Objectives für den aktuellen Level auf `passed` gesetzt und damit dem LMS signalisiert, dass der Lernende den Level erfolgreich absolviert hat (vgl. Abschnitt 5.3.2). Des Weiteren wird der Wert des Elementes `cmi.success_status` auf `passed` und der Wert des Elementes `cmi.completion_status` auf `completed` gesetzt, was dem LMS signalisiert, dass der Lernende das SCO erfolgreich und vollständig abgeschlossen hat.

Schließlich wird, durch das Ablegen des Wertes `continue` in das Navigation Data Model Element `adl.nav.request`, das LMS angewiesen nach dem Beenden der Kommunikation mit dem SCO zur nächsten Activity und damit zum nächsten Level des Spiels zu springen. Die Funktion `terminateCommunication()` beendet schließlich die Kommunikation.

5.4.3 Kommunikation mit dem SmartFoxServer

Der SmartFoxServer, in der Rolle des Spieleservers, ist verantwortlich für die Verbindung der Spielanwendungen untereinander. Dazu muss die Anwendung mit dem Spieleserver kommunizieren können. Ein Konzept zur Client-Server Kommunikation eines, im Flash-Player ausgeführten Clients, ist die Verbindung mit dem Server über Sockets, einer standardisierten Schnittstelle zwischen einer Anwendung und einem Netzwerk. Die Klasse `XMLSocket` implementiert in Adobe Flash clientseitige Sockets. Um die Klasse verwenden zu können, müssen auf dem Server entsprechende Dienste zur automatischen Verarbeitung des, von der Klasse verwendeten, Protokolls ausgeführt werden. Beschrieben ist dieses Protokoll in der in [Adobe07] enthaltenen Hilfe. Danach werden über eine TCP/IP-Streaming-Socket-Verbindung XML-Nachrichten gesendet. Jede XML-Nachricht ist dabei ein vollständiges XML-Dokument. Über eine solche `XMLSocket` Verbindung kann eine unbegrenzte Anzahl von XML-Nachrichten gesendet und empfangen werden.

Abgeleitet von der Klasse `XMLSocket`, ist die Klasse `SmartFoxClient` die Voraussetzung für den Umgang mit dem SmartFoxServer. Die Klasse ist Teil der im SmartFoxServer enthaltenen Client API für ActionScript. Diese wird in Adobe Flash als Erweiterung installiert und kann anschließend verwendet werden. Neben der `SmartFoxClient` Klasse enthält die API noch die Klassen `Room` und `User` zur Verwaltung der Eigenschaften von Räumen und Nutzern. Eine Übersicht über die in der SmartFoxServer Client API für ActionScript enthaltenen Klassen sowie deren öffentliche Eigenschaften und Methoden ist in Anhang D enthalten.

Listing 5.8: Funktion zur Ereignisbehandlung

```
smartFox.onPrivateMessage = function(msg:String, sender:User)
{
        trace("A private message from " + sender.getName())
        trace(msg)
}
```

Der SmartFoxServer arbeitet ereignisbasiert. Das Ausführen einer Methode löst in der Regel ein oder mehrere Ereignisse aus, die dann durch den Client behandelt werden. Die Methode `sendPrivateMessage` der `SmartFoxClient` Klasse löst beispielsweise das Ereignis `onPrivateMessage` aus. Listing 5.8 zeigt eine mögliche Funktion zur Behandlung dieses Ereignisses durch den Client. Erst diese Vorgehensweise ermöglicht schließlich die Interaktion des SmartFoxServers mit der Spielanwendung und der Anwendungen untereinander.

Damit eine Interaktion überhaupt möglich ist, sind prinzipiell drei Schritte nötig. Zuerst wird versucht eine Verbindung zwischen Client und Server aufzubauen. Wurde diese Verbindung erfolgreich aufgebaut, muss sich der Client im zweiten Schritt auf dem Server in einer Zone anmelden. Abschließend betritt der Client im dritten Schritt einen Raum innerhalb der gewählten Zone. Listing 5.9 zeigt zunächst den Aufbau einer Verbindung zwischen Spielanwendung und SmartFoxServer. Um den Umgang mit der Klasse `SmartFoxClient` zu vereinfachen, werden die zum SmartFoxServer zugehörigen Klassen importiert. Anschließend wird ein neues `SmartFoxClient` Objekt angelegt und eine Funktion für die Ereignisprozedur `onConnection` des Objekts definiert. Die eigentliche Verbindung stellt dann die Methode `connect` her.

Listing 5.9: Aufbau einer Verbindung zum SmartFoxServer

```
import it.gotoandplay.smartfoxserver.*

var ip:String     = "127.0.0.1"
var port:Number   = 9339

var smartfox:SmartFoxClient = new SmartFoxClient()
smartfox.onConnection = handleConnection

System.security.loadPolicyFile("xmlsocket://127.0.0.1:9339")
smartfox.connect(ip, port)

function handleConnection(success:Boolean)
{
      if (success)
      {
            //connected
      }
      else
      {
            //not connected
      }
}
```

Zur Identifizierung des Servers genügen dessen IP-Adresse und eine Portnummer. Die Methode `System.security.loadPolicyFile` lädt eine Richtliniendatei von der angegebenen URL. Der Flash-Player verwendet die Richtliniendatei unter anderem als Berechtigungsmechanismus für den Umgang mit externen Servern. Nur so kann die Spielanwendung eine `XMLSocket` Verbindung mit dem SmartFoxServer über die Methode `connect` herstellen. Diese Richtliniendatei wird vom SmartFoxServer automatisch zur Verfügung gestellt. Wurde die Methode `connect` ausgeführt, wird das `onConnection` Ereignis ausgelöst und die zuvor definierte Funktion aufgerufen. Über den boolschen Parameter `success` der Funktion wird ermittelt ob die Verbindung erfolgreich war.

Listing 5.10: Anmeldung auf dem SmartFoxServer

```
smartfox.login("zone", "username", "password")

smartfox.onLogin = function(resObj:Object)
{
	if (resObj.success)
	{
		//login succesfull
	}
	else
	{
		//login failed
	}
}
```

Listing 5.10 zeigt das Anmelden auf dem SmartFoxServer. Die Methode `login` des `SmartFoxClient` Objekts realisiert die Anmeldung des Clients am Server. Als Parameter werden der Methode die Namen der Zone und des Benutzers sowie optional ein Passwort übergeben. Jeder Client kann sich nur in einer Zone des Servers anmelden. Der Benutzername muss innerhalb dieser Zone eindeutig sein. Wurde die Methode `login` ausgeführt, wird das `onLogin` Ereignis ausgelöst und die entsprechende Funktion aufgerufen. Über die Eigenschaft `success` des übergebenen Parameters `resObj` kann man wiederum ermitteln ob die Anmeldung erfolgreich war. Nach einer erfolgreichen Anmeldung wird intern die Methode `getRoomList` des `SmartFoxClient` Objekts ausgeführt, welche eine Übersicht über sämtliche Räume der Zone liefert und das Ereignis `onRoomListUpdate` auslöst. Dieses Ereignis kann dann durch den Client behandelt werden. Listing 5.11 zeigt die Funktionen zur Raumübersicht und zum Betreten eines Raumes innerhalb einer Zone.

Listing 5.11: Raumübersicht und Betreten eines Raumes innerhalb einer Zone

```
smartfox.onRoomListUpdate = function(roomList:Object)
{
	for (var i:String in roomList)
	{
		var room:Room = roomList[i]

		trace(room.getName())
	}
	this.joinRoom("roomname", "password")
}

smartfox.onJoinRoom = function(roomObj:Room)
{
	var userList:Object = roomObj.getUserList()

	for (var i:String in userList)
	{
		var user:User = userList[i]
		trace(user.getName())
	}
}
```

Der übergebene Parameter `roomList` enthält alle vorhandenen Instanzen der Klasse `Room` innerhalb der Zone. Über diesen Parameter können die einzelnen Methoden der Klasse aufgerufen werden. Schließlich betritt der Client über die Methode `joinRoom` des `SmartFoxClient` Objekts einen Raum, wodurch das Ereignis `onJoinRoom` ausgelöst wird. Der hier übergebene Parameter `roomObj` enthält die Instanz des betretenen Raumes und ermöglicht so den Zugriff auf die Eigenschaften und Methoden des Raumes. Die Methode `getUserList` listet alle Instanzen der Klasse `User` auf, die sich in einem Raum befinden. Über das zurückgegebene Objekt `userList` können dann wiederum die einzelnen Methoden der Klasse `User` aufgerufen werden. Damit sind `onRoomListUpdate` und `onJoinRoom` die wichtigsten Ereignisse des SmartFoxServers, da sie die Datenstruktur (vgl. Abschnitt 5.2.2) innerhalb des `SmartFoxClient` Objekts abbilden. Nachdem der Client nun den Raum betreten hat, ist er in der Lage mit allen anderen Clients innerhalb des Raumes zu interagieren. Diese Interaktion basiert wiederum auf der Ausführung von Methoden und dem Behandeln der dadurch ausgelösten Ereignisse.

5.5 Zusammenfassung

Die erläuterte prototypische Implementierung setzt das, in Abschnitt 4.4 vorgestellte, Konzept zur Integration eines Spiel-basierten Lernprogramms in eine SCORM-konforme

Lernumgebung unter Einsatz eines Multi-User-fähigen Spieleservers um. Dabei ist ein Kurs für eine SCORM-konforme Lernumgebung entstanden, der die Spielanwendung „Drei/Vier/Fünf Gewinnt“ enthält, welche die Funktionalitäten eines Spieleservers nutzt.

Die Kursstruktur wurde mit dem Reload Editor realisiert und die Spielanwendung mit Hilfe der integrierten Entwicklungsumgebung Adobe Flash bearbeitet. Als Spieleserver diente der SmartFoxServer der Firma gotoAndPlay() und als beispielhafte Lernumgebung die SCORM Sample RTE. Zur Umsetzung des Konzepts musste einerseits der Kurs in die Lernumgebung integriert und andererseits das Client-Server Modell implementiert werden. Für die Integration in die Lernumgebung wurde zunächst ein Content Package, basierend auf dem SCORM CAM (vgl. Abschnitt 1.4.2), erstellt. Durch die Verwendung der SCORM SN Elemente zur Ablaufsteuerung (vgl. Abschnitt 1.4.4) konnte anschließend die Steuerung des Kurses, gemäß der vorher festgelegten Sequencingstrategie, umgesetzt werden. Die Implementierung des Client-Server Modells gliederte sich in zwei Teile. Während die Kommunikation zwischen Spielanwendung und Lernumgebung durch die SCORM RTE (vgl. Abschnitt 1.4.3) spezifiziert ist und dementsprechend umgesetzt werden konnte, erfolgt die Kommunikation der Spielanwendung mit dem Spieleserver nicht über eine ebenso standardisierte Schnittstelle. Die Wiederverwendbarkeit und Interoperabilität des Kurses ist somit nur unter Einsatz des SmartFoxServer als Spieleserver gesichert. Das zugrunde liegende Konzept für die Integration ist also umgesetzt, eine vollständige Konformität, bezüglich der SCORM Spezifikation und ihren Zielen, ist dessen ungeachtet nicht gegeben.

Zusammenfassung

Die vorliegende Arbeit beschäftigte sich mit dem Thema Spiel-basiertes Lernen, als Teilaspekt des E-Learning und alternative Form zur Vermittlung von Lerninhalten. Den Einstieg in die Thematik bildete eine Einführung zum E-Learning. Nach einer Begriffsklärung folgte eine Erläuterung der SCORM Spezifikation, als Möglichkeit zur Standardisierung von Lerninhalten. Des Weiteren wurden die didaktischen Aspekte des Spiel-basierten Lernens beschrieben und eine entsprechende Klassifizierung vorgenommen.

Die Arbeit zeigt, dass Spiel-basierte Lernprogramme in didaktischer Hinsicht für bestimmte Lerninhalte besser geeignet sind als klassische Lernprogramme. Sie vereinen spielerische Elemente mit didaktischen Aspekten und schaffen so ein motivierendes, spielerisch gestaltetes multimediales Lernangebot. Der Lernende wird dabei stärker in den Lernprozess einbezogen, kann Lerninhalte selbstständig erarbeiten und mit seinem erworbenen Wissen gefahrlos experimentieren. Der mögliche Multi-User Aspekt einiger Spiel-basierter Lernprogramme erlaubt des Weiteren die Kooperation der Lernenden, was die Motivation des Einzelnen und damit die Effektivität des Lernprozesses zusätzlich erhöht. Die so entstehenden Lernangebote verknüpfen Lernen und Spielen in einem sozialen Kontext.

Hauptgegenstand der Arbeit war die Untersuchung möglicher Integrationskonzepte Spiel-basierter Lernprogramme in vorhandene SCORM-konforme Lernumgebungen. Bei den Untersuchungen wurde besonders der mögliche Multi-User Aspekt einiger Lernprogramme berücksichtigt. Als Vorbereitung für die Erstellung der konkreten Integrationskonzepte diente die Analyse der Anforderungen SCORM-konformer Lernumgebungen und Spiel-basierter Lernprogramme. Zunächst wurden die Möglichkeiten zum Aufbau und zur Verwendung der Lernprogramme betrachtet und im Anschluss zwei Konzepte zur Integration, bezüglich der notwendigen Anpassungen und der zugrunde liegenden Architektur, erläutert und bewertet. Als Ausblick wurde eine Erweiterung der SCORM Spezifikation diskutiert. Den Abschluss der Arbeit bildete schließlich die prototypische Implementierung des Konzepts zur Integration eines Spiel-basierten Lernprogramms in eine SCORM-konforme Lernumgebung unter Einsatz eines Multi-User-fähigen Spieleservers.

Die Bewertung der Konzepte fällt unterschiedlich aus. Innerhalb der SCORM Spezifikation können Spiel-basierte Lernprogramme ohne Multi-User Aspekt wie klassische Lerninhalte

integriert werden. Das SCORM RTE Data Model, verbunden mit den Möglichkeiten des Sequencing, lässt ihnen dabei einen sehr flexiblen Gestaltungsspielraum, wie das erste vorgestellte Konzept zeigt. Das zweite Konzept integriert ein Spiel-basiertes Lernprogramm mit Multi-User Aspekt und setzt, aufgrund der nicht vorhanden Unterstützung kooperativer Lernformen durch die SCORM Spezifikation, einen zusätzlichen Spieleserver ein. Die daraus resultierende Implementierung ist jedoch nicht vollständig SCORM-konform. Die Einbindung der Lerninhalte in die Lernumgebung ist zwar standardkonform, die Sicherung der Wiederverwendbarkeit und Interoperabilität der Lerninhalte ist aber durch den Einsatz des Spieleservers nicht gewährleistet, da eine standardisierte Schnittstelle für den Umgang mit einem Spieleserver nicht existiert. Die prototypische Implementierung des Konzepts liefert dennoch wichtige Ansatzpunkte im Sinne einer Erweiterung der SCORM Spezifikation zur Unterstützung eines möglichen Multi-User Aspektes Spiel-basierter Lernprogramme.

Eine solche Erweiterung wäre auch für andere kooperative Lernformen, abseits des Spiel-basierten Lernens, interessant und würde die Attraktivität vieler bestehender und zukünftiger Lernangebote steigern. Zur Zeit gibt es ungeachtet dessen keine Entwicklungen in dieser Richtung seitens der ADL Initiative. Die Entwicklung einer standardisierten Lösung für die Integration Spiel-basierter Lernprogramme und anderer alternativer Lernformen innerhalb der SCORM Spezifikation, ohne Beachtung eines möglichen Multi-User Aspektes, wird durch die ADL Initiative dagegen voran getrieben. Konkrete Ergebnisse liegen aber speziell für das Spiel-basierte Lernen auch hier noch nicht vor. ADL beschreibt aber bereits signifikante Komponenten [ADL06c], die es in diesem Kontext zu realisieren gilt. Dazu gehört die Entwicklung von Spielactivities mit formalen Regeln, die den Spielern die Möglichkeit zum spielenden Lernen geben und die erzielten Spiel- und Lernergebnisse festhalten. Des Weiteren eine Spielhandlung, die eng an die Activities geknüpft ist und Zusammenhänge erläutert. Und schließlich eine Art Simulation, die dem Spieler den Raum zum Lernen zur Verfügung stellt, um in die Handlung und damit die Activities einzutauchen. Der Weg den die ADL Initiative einschlägt entspricht somit durchaus den Erkenntnissen dieser Arbeit. Die Realisierung bleibt allerdings Gegenstand weiterer Untersuchungen.

Literaturverzeichnis

ADL06a ADL Technical Team: *Sharable Content Object Reference Model (SCORM) 2004 3rd Edition Documentation Suite*.
URL: http://www.adlnet.gov/downloads/files/311.cfm
[Stand: 10/20/2006]

ADL06b ADL Technical Team: *Sharable Content Object Reference Model (SCORM) 2004 3rd Edition Conformance Requirements Version 1.0*.
URL: http://www.adlnet.gov/downloads/files/301.cfm
[Stand: 10/20/2006]

ADL06c ADL Technical Team: *Technologies > Gaming*.
URL: http://www.adlnet.gov/technologies/gaming/
[Stand: 01/30/2007]

ADL07 ADL Technical Team: *SCORM 2004 3rd Edition Sample Run-Time Environment Version 1.0.1*.
URL: http://www.adlnet.gov/downloads/downloadpage.aspx?ID=280
[Stand: 07/15/2007]

Adobe07 Adobe Systems Incorporated: *Flash CS3 Professionel*.
URL: http://www.adobe.com/de/products/flash/
[Stand: 07/01/2007]

Baumgartner02 Baumgartner, Peter/Häferle, Hartmut/Maier-Häferle, Kornelia: *E-Learning Praxishandbuch - Auswahl von Lernplattformen*.
Innsbruck: Studienverlag, 2002

Becta01 British Educational Communications and Technology Agency: *Computer Games in Education project: Aspects*.
URL: http://partners.becta.org.uk/index.php?section=rh&rid=13588
[Stand: 07/15/2007]

Blumstengel98 Blumstengel, Astrid: *Entwicklung hypermedialer Lernsysteme*.
Paderborn: Wiss. Verlag Berlin, 1998

Breuer02 Breuer, Jens: „Kooperative Lernformen beim E-Learning einsetzen“.
In: Hohenstein, Andreas/Wilbers, Karl (Hrsg.): *Handbuch E-Learning - Expertenwissen aus Wissenschaft und Praxis - Band 4*.
Köln: Fachverlag Deutscher Wirtschaftsdienst, 2002

Burgos06 Burgos, Daniel/Tattersall, Colin/Koper, Rob: *Can IMS learning design be used to model computer-based educational games?*.
URL: http://dspace.ou.nl/bitstream/1820/673/1/BINARIA_Issue5_burgos_et_al.pdf
[Stand: 01/30/2007]

Diener06 Diener, Holger/Malo, Steffen/Martens, Alke: „Game-Based Learning - Spiel, Simulation und Lernen“. In: Diener, Holger/Malo, Steffen/ Martens, Alke/Urban, Bodo (Hrsg.): *Game Based Learning – Beiträge des Preconference Workshop der 3. Deutschen e-Learning Fachtagung Informatik (DeLFI 2005)*.
Rostock: ITB Verlag Stuttgart, 2006

Dittler03 Dittler, Ullrich: „Computer-Based-Training zur Vermittlung von Hardskills“. In: Dittler, Ullrich (Hrsg.): *E-Learning – Einsatzkonzepte und Erfolgsfaktoren des Lernens mit interaktiven Medien*.
München: Oldenburg Wissenschaftsverlag, 2003

Dittler96 Dittler, Ullrich: *Von Computerspielen zu Lernprogrammen – empirische Befunde für die Förderung computergestützten Lernens*.
Frankfurt am Main: Peter Lang Verlag, 1996

Fehr06 Fritz, Jürgen/Fehr, Wolfgang: „Computerspiele als Fortsetzung des Alltags - Wie sich Spielwelten und Lebenswelten verschränken“. In: Fritz, Jürgen/Fehr, Wolfgang (Hrsg.): *Handbuch Medien: Computerspiele - Virtuelle Spiel- und Lernwelten*.
URL: http://www.medienpaedagogik-online.de/cs/00785/
[Stand: 10/02/2006]

Fritz06 Fritz, Jürgen: „Wie virtuelle Welten wirken - Über die Struktur von Transfers aus der medialen in die reale Welt“. In: Fritz, Jürgen/Fehr, Wolfgang (Hrsg.): *Handbuch Medien: Computerspiele – Virtuelle Spiel- und Lernwelten*.
URL: http://www.medienpaedagogik-online.de/cs/00800/
[Stand: 10/02/2006]

Fromme01 Fromme, Johannes/Meder, Norbert: „Computerspiel und Bildung: Zur theoretischen Einführung“. In: Fromme, Johannes/Meder, Norbert (Hrsg.): *Bildung und Computerspiele - zum kreativen Umgang mit elektronischen Bildschirmspielen*.
Opladen: Leske + Budrich, 2001

Glossar06 T-Systems: *Global Learning Glossar - Fachausdrücke des Telelernens und relevante Internetbegriffe*.
URL: http://www.global-learning.de/g-learn/cgibin/gl_userpage.cgi?StructuredContent=ml0801
[Stand: 10/02/2006]

Ho06 Ho, Pei-Chi/Chung, Szu-Ming/Tsai, Ming-Hsin: „A Case Study of Game Design for E-Learning“. In: Pan,Zhigeng/Aylett, Ruth/Diener, Holger/Jin, Xiaogang/Göbel, Stefan/Li, Li (Hrsg.): *Technologies for E-Learning and Digital Entertainment - First International Conference, Edutainment 2006 Hangzhou, China, April 2006 Proceedings*.
Heidelberg: Springer Verlag, 2006

LSAL03 Learning Systems Architecture Lab: *Technical Evolution of SCORM*. URL: http://lsal.org/lsal/expertise/projects/scorm/scormevolution/reportv1p02/report-v1p02.pdf [Stand: 01/30/2007]

Meier03 Meier, Christoph/Seufert, Sabine: „Game-based Learning – Erfahrungen mit und Perspektiven für digitale Lernspiele in der betrieblichen Bildung". In: Hohenstein, Andreas (Hrsg.): *Handbuch E-Learning - Expertenwissen aus Wissenschaft Praxis*. Köln: Fachverlag Deutscher Wirtschaftsdienst, 2003

Meschenmoser02 Meschenmoser, Helmut: „Lernen mit Multimedia und Internet". In: Bönsch, Manfred/Kaiser, Astrid (Hrsg.): *Basiswissen Pädagogik - Unterichtskonzepte und -techniken Band 5*. Hohengehren: Schneider Verlag, 2002

Müsgens00 Müsgens, Martin: „Die Wirkung von Bildschirmspielen auf Kinder im Alter von 6 bis 11 Jahren. Ein empirischer Feldversuch". In: Bühl, Achim (Hrsg.): *Cyberkids - Empirische Untersuchungen zur Wirkung von Bildschirmspielen*. Münster: LIT Verlag, 2000

Prensky01 Prensky, Marc: *Digital Game-Based Learning*. New York: McGraw-Hill Education, 2001

REL07 Reload Project: *Reload Editor Version 2.5.4*. URL: http://www.reload.ac.uk/editor.html [Stand: 04/05/2007]

SFS07 gotoAndPlay(): *SmartFoxServer Pro Version 1.5.5*. URL: http://www.smartfoxserver.com/products/pro.php [Stand: 04/05/2007]

Wiebe01 Wiebe, Gerrit: „Spielerisches Lernen in Multi-User-Dungeons". In: Fromme, Johannes/Meder, Norbert (Hrsg.): *Bildung und Computerspiele - zum kreativen Umgang mit elektronischen Bildschirmspielen*. Opladen: Leske + Budrich, 2001

Tabellenverzeichnis

Tabelle 2.1: Eigenschaften von Computerspielen [Becta01].....26

Tabelle 2.2: Typen digitaler Lernspiele, nach [Meier03]..... 34

Tabelle 3.1: LMS Conformance Matrix [ADL06b].....38

Tabelle 3.2: Content Package Conformance Matrix [ADL06b]..... 41

Tabelle 3.3: SCO Conformance Matrix [ADL06b]..... 42

Tabelle 3.4: Didaktische Komponenten von Lernspielen [Burgos06].....43

Tabelle 3.5: Technische Komponenten von Lernspielen [Burgos06] 44

Tabelle 3.6: Anforderungskatalog Spiel-basierter Lernprogramme 47

Tabelle 3.7: Erfüllbare Anforderungen (SCORM) eines Spiel-basierten Lernprogramms ohne Multi-User Aspekt 48

Tabelle 3.8: Erfüllbare Anforderungen (SCORM) eines Spiel-basierten Lernprogramms mit Multi-User Aspekt 49

Tabelle 3.9: Umsetzung der Anforderungen in SCORM.....51

Tabelle 4.1: Anpassung Spiel-basierter Lernprogramme (SCO) 55

Tabelle 4.2: Anpassung Spiel-basierter Lernprogramme (Content Package).....55

Tabelle 4.3: Verwendung der SCORM RTE Data Model Elemente für Spiel-basierte Lernprogramme 57

Tabelle 4.4: Charakterisierung der Bestandteile eines erweiterten Client-Server-Modells 59

Tabelle 4.5: Erfüllbare Anforderungen (Konzept 2) eines Spiel-basierten Lernprogramms mit Multi-User Aspekt 62

Abbildungsverzeichnis

Abbildung 1.1: Bestandteile von SCORM [ADL06a]..... 16

Abbildung 1.2: Das Content Aggregation Model Buch [ADL06a]..... 18

Abbildung 1.3: Content Model [ADL06a]..... 19

Abbildung 1.4: Das Run-Time Environment Buch [ADL06a]..... 21

Abbildung 1.5: Das Sequencing and Navigation Buch [ADL06a]..... 22

Abbildung 2.1: Gestaltung interaktiver Lernsysteme [Meier03]..... 29

Abbildung 4.1: Integration eines Spiel-basierten Lernprogramms in eine SCORM-konforme Lernumgebung..... 56

Abbildung 4.2: Integration eines Spiel-basierten Lernprogramms in eine SCORM-konforme Lernumgebung unter Einsatz eines Multi-User-fähigen Spieleservers..... 60

Abbildung 4.3: Einsatz einer Zwischenschicht..... 65

Abbildung 5.1: Datenstruktur des SmartFoxServer [SFS07]..... 69

Abbildung 5.2: Funktionsweise der SCORM Sample RTE [ADL06a]..... 72

Abbildung 5.3: Content Package im Reload Editor..... 73

Abbildung 5.4: Implementiertes Client-Server Modell..... 79

Abbildung 5.5: Chatraum der Spielanwendung..... 80

Abbildung 5.6: Spielraum der Spielanwendung..... 81

Quellcodeverzeichnis

Listing 5.1: Abschnitt <resources> der Manifest Datei 73

Listing 5.2: Abschnitt <organizations> der Manifest Datei 74

Listing 5.3: Sequencing Control Modes 76

Listing 5.4: Objectives 77

Listing 5.5: Sequencing Rules 78

Listing 5.6: JavaScript Funktion initializeCommunication() 82

Listing 5.7: ActionScript Funktionsaufrufe der Spielanwendung 83

Listing 5.8: Funktion zur Ereignisbehandlung 84

Listing 5.9: Aufbau einer Verbindung zum SmartFoxServer 85

Listing 5.10: Anmeldung auf dem SmartFoxServer 86

Listing 5.11: Raumübersicht und Betreten eines Raumes innerhalb einer Zone 87

Anhang

Anhang A – SCORM RTE Data Model Elemente

Tabelle Anhang A: SCORM RTE Data Model Elements [ADL06a]

Data Model Element	Dot-Notation Binding	Description
Comments From Learner	cmi.comments_from_learner	Contains text from the learner.
Comments From LMS	cmi.comments_from_lms	Contains comments and annotations intended to be made available to the learner.
Completion Status	cmi.completion_status	Indicates whether the learner has completed the SCO.
Completion Threshold	cmi.completion_threshold	Identifies a value against which the measure of the progress the learner has made toward completing the SCO can be compared to determine whether the SCO should be considered completed.
Credit	cmi.credit	Indicates whether the learner will be credited for performance in this SCO.
Entry	cmi.entry	Contains information that asserts whether the learner has previously accessed the SCO.
Exit	cmi.exit	Indicates how or why the learner left the SCO.
Interactions	cmi.interactions	Defines information pertaining to an interaction for the purpose of measurement or assessment.
Launch Data	cmi.launch_data	Provides data specific to a SCO that the SCO can use for initialization.
Learner ID	cmi.learner_id	Identifies the learner on behalf of whom the SCO instance was launched.
Learner Name	cmi.learner_name	Represents the name of the learner.
Learner Preference	cmi.learner_preference	Specifies learner preferences associated with the learner’s use of the SCO.
Location	cmi.location	Represents a location in the SCO.
Maximum Time Allowed	cmi.max_time_allowed	Indicates the amount of accumulated time the learner is allowed to use a SCO in the learner attempt.
Mode	cmi.mode	Identifies the modes in which the SCO may be presented to the learner.

Data Model Element	Dot-Notation Binding	Description
Objectives	cmi.objectives	Specifies learning or performance objectives associated with a SCO.
Progress Measure	cmi.progress_measure	Identifies a measure of the progress the learner has made toward completing the SCO.
Scaled Passing Score	cmi.scaled_passing_score	Identifies the scaled passing score for a SCO.
Score	cmi.score	Identifies the learner's score for the SCO.
Session Time	cmi.session_time	Identifies the amount of time that the learner has spent in the current learner session for the SCO.
Success Status	cmi.success_status	Indicates whether the learner has mastered the SCO.
Suspend Data	cmi.suspend_data	Provides information that may be created by a SCO as a result of a learner accessing or interacting with the SCO.
Time Limit Action	cmi.time_limit_action	Indicates what the SCO should do when the maximum time allowed is exceeded.
Total Time	cmi.total_time	Identifies the sum of all of the learner's learner session times accumulated in the current learner attempt prior to the current learner session.

Anhang B – imsmanifest.xml

Listing Anhang B: imsmanifest.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<manifest xmlns="http://www.imsglobal.org/xsd/imscp_v1p1"
          xmlns:imsmd="http://ltsc.ieee.org/xsd/LOM"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xmlns:adlcp="http://www.adlnet.org/xsd/adlcp_v1p3"
          xmlns:imsss="http://www.imsglobal.org/xsd/imsss"
          xmlns:adlseq="http://www.adlnet.org/xsd/adlseq_v1p3"
          xmlns:adlnav="http://www.adlnet.org/xsd/adlnav_v1p3"
          identifier="MANIFEST-3110DAC7BE7AE2E76FB630A2FC6E0CE1"
          xsi:schemaLocation="http://www.imsglobal.org/xsd/imscp_v1p1
          imscp_v1p1.xsd http://ltsc.ieee.org/xsd/LOM lom.xsd
          http://www.adlnet.org/xsd/adlcp_v1p3 adlcp_v1p3.xsd
          http://www.imsglobal.org/xsd/imsss imsss_v1p0.xsd
          http://www.adlnet.org/xsd/adlseq_v1p3 adlseq_v1p3.xsd
          http://www.adlnet.org/xsd/adlnav_v1p3 adlnav_v1p3.xsd">
 <metadata>
  <schema>ADL SCORM</schema>
  <schemaversion>2004 3rd Edition</schemaversion>
  <imsmd:lom>
   <imsmd:technical>
    <imsmd:format>text/html</imsmd:format>
    <imsmd:format>application/x-shockwave-flash</imsmd:format>
   </imsmd:technical>
  </imsmd:lom>
 </metadata>
 <organizations default="ORG">
  <organization identifier="ORG" structure="hierarchical">
    <title>Spielprototyp</title>
    <item identifier="ITEM-LEVEL1" isvisible="true" identifierref="RES1">
     <title>Level 1</title>
     <adlnav:presentation>
      <adlnav:navigationInterface>
       <adlnav:hideLMSUI>suspendAll</adlnav:hideLMSUI>
       <adlnav:hideLMSUI>continue</adlnav:hideLMSUI>
      </adlnav:navigationInterface>
     </adlnav:presentation>
     <imsss:sequencing>
      <imsss:sequencingRules>
       <imsss:preConditionRule>
        <imsss:ruleConditions>
         <imsss:ruleCondition condition="satisfied" operator="noOp"
                              referencedObjective="level_1" />
        </imsss:ruleConditions>
        <imsss:ruleAction action="skip" />
       </imsss:preConditionRule>
      </imsss:sequencingRules>
      <imsss:objectives>
       <imsss:primaryObjective />
       <imsss:objective objectiveID="level_1">
        <imsss:mapInfo readSatisfiedStatus="true"
                       targetObjectiveID="global_level_1"
                       writeSatisfiedStatus="true" />
       </imsss:objective>
```

```
      </imsss:objectives>
     </imsss:sequencing>
    </item>
    <item identifier="ITEM-LEVEL2" isvisible="true" identifierref="RES1">
     <title>Level 2</title>
     <adlnav:presentation>
      <adlnav:navigationInterface>
       <adlnav:hideLMSUI>suspendAll</adlnav:hideLMSUI>
       <adlnav:hideLMSUI>continue</adlnav:hideLMSUI>
      </adlnav:navigationInterface>
     </adlnav:presentation>
     <imsss:sequencing>
      <imsss:sequencingRules>
       <imsss:preConditionRule>
        <imsss:ruleConditions>
         <imsss:ruleCondition condition="satisfied" operator="noOp"
                              referencedObjective="level_2" />
        </imsss:ruleConditions>
        <imsss:ruleAction action="skip" />
       </imsss:preConditionRule>
      </imsss:sequencingRules>
      <imsss:objectives>
       <imsss:primaryObjective />
       <imsss:objective objectiveID="level_2">
        <imsss:mapInfo readSatisfiedStatus="true"
                       targetObjectiveID="global_level_2"
                       writeSatisfiedStatus="true" />
       </imsss:objective>
      </imsss:objectives>
     </imsss:sequencing>
    </item>
    <item identifier="ITEM-LEVEL3" isvisible="true" identifierref="RES1">
     <title>Level 3</title>
     <adlnav:presentation>
      <adlnav:navigationInterface>
       <adlnav:hideLMSUI>suspendAll</adlnav:hideLMSUI>
       <adlnav:hideLMSUI>continue</adlnav:hideLMSUI>
      </adlnav:navigationInterface>
     </adlnav:presentation>
     <imsss:sequencing>
      <imsss:sequencingRules>
       <imsss:preConditionRule>
        <imsss:ruleConditions>
         <imsss:ruleCondition condition="satisfied" operator="noOp"
                              referencedObjective="level_3" />
        </imsss:ruleConditions>
        <imsss:ruleAction action="skip" />
       </imsss:preConditionRule>
      </imsss:sequencingRules>
      <imsss:objectives>
       <imsss:primaryObjective />
       <imsss:objective objectiveID="level_3">
        <imsss:mapInfo readSatisfiedStatus="true"
                       targetObjectiveID="global_level_3"
                       writeSatisfiedStatus="true" />
       </imsss:objective>
      </imsss:objectives>
     </imsss:sequencing>
```

```
    </item>
    <item identifier="ITEM-ENDE" identifierref="RES2" isvisible="true">
     <title>Ende</title>
     <adlnav:presentation>
      <adlnav:navigationInterface>
       <adlnav:hideLMSUI>suspendAll</adlnav:hideLMSUI>
       <adlnav:hideLMSUI>exitAll</adlnav:hideLMSUI>
      </adlnav:navigationInterface>
     </adlnav:presentation>
     <imsss:sequencing>
      <imsss:sequencingRules>
       <imsss:preConditionRule>
        <imsss:ruleConditions>
         <imsss:ruleCondition condition="satisfied" operator="not"
                              referencedObjective="ende" />
        </imsss:ruleConditions>
        <imsss:ruleAction action="skip" />
       </imsss:preConditionRule>
      </imsss:sequencingRules>
      <imsss:objectives>
       <imsss:primaryObjective />
       <imsss:objective objectiveID="ende">
        <imsss:mapInfo readSatisfiedStatus="true"
                       targetObjectiveID="global_level_3" />
       </imsss:objective>
      </imsss:objectives>
     </imsss:sequencing>
    </item>
    <imsss:sequencing>
     <imsss:controlMode choice="false" choiceExit="false" flow="true"
                        forwardOnly="true" />
    </imsss:sequencing>
  </organization>
 </organizations>
 <resources>
  <resource identifier="RES1" adlcp:scormType="sco" type="webcontent"
            href="bin/game.html">
   <file href="bin/game.html" />
   <file href="bin/game.swf" />
   <file href="scripts/APIWrapper.js" />
  </resource>
  <resource identifier="RES2" adlcp:scormType="sco" type="webcontent"
            href="bin/complete.html">
   <file href="bin/complete.jpg" />
   <file href="bin/complete.html" />
  </resource>
 </resources>
</manifest>
```

Anhang C – APIWrapper.js

Die Datei wird von der ADL Initiative bereitgestellt. Die eingefügten Kommentare zu den Funktionen sind in dem Listing nicht mit aufgeführt.

Listing Anhang C: APIWrapper.js

```
var apiHandle = null;
var findAPITries = 0;
var noAPIFound = "false";

var terminated = "false";
var _debug = false;


function findAPI( win )
{
   while ( (win.API_1484_11 == null) &&
           (win.parent != null) &&
           (win.parent != win) )
   {
      findAPITries++;

      if ( findAPITries > 500 )
      {
         alert( "Error finding API -- too deeply nested." );
         return null;
      }

      win = win.parent;
   }

   return win.API_1484_11;
}

function getAPI()
{
   var theAPI = findAPI( window );

   if ( (theAPI == null) &&
        (window.opener != null) &&
        (typeof(window.opener) != "undefined") )
   {
      theAPI = findAPI( window.opener );
   }

   if (theAPI == null)
   {
      alert( "Unable to locate the LMS's API Implementation.\n" +
             "Communication with the LMS will not occur." );

      noAPIFound = "true";
   }

   return theAPI
}
```

```
function getAPIHandle()
{
   if ( apiHandle == null )
   {
      if ( noAPIFound == "false" )
      {
         apiHandle = getAPI();
      }
   }

   return apiHandle;
}

function initializeCommunication()
{
   var api = getAPIHandle();

   if ( api == null )
   {
      return "false";
   }
   else
   {
      var result = api.Initialize("");

      if ( result != "true" )
      {
         var errCode = retrieveLastErrorCode();

         displayErrorInfo( errCode );

         // may want to do some error handling
      }
   }

   return result;
}

function terminateCommunication()
{
   var api = getAPIHandle();

   if ( api == null )
   {
      return "false";
   }
   else
   {
      // call Terminate only if it was not previously called
      if ( terminated != "true" )
      {
         // call the Terminate function that should be implemented by
         // the API
         var result = api.Terminate("");

         if ( result != "true" )
         {
            var errCode = retrieveLastErrorCode();
```

```
            displayErrorInfo( errCode );

            // may want to do some error handling
         }
         else  // terminate was successful
         {
            terminated = "true";
         }
      }
   }

   return result;
}

function retrieveDataValue( name )
{
   // do not call a set after finish was called
   if ( terminated != "true" )
   {
      var api = getAPIHandle();

      if ( api == null )
      {
         return "";
      }
      else
      {
         var value = api.GetValue( name );

         var errCode = api.GetLastError();

         if ( errCode != "0" )
         {
            var errCode = retrieveLastErrorCode();

            displayErrorInfo( errCode );
         }
         else
         {
            return value;
         }
      }
   }

   return;
}

function storeDataValue( name, value )
{
   // do not call a set after finish was called
   if ( terminated != "true" )
   {
      var api = getAPIHandle();

      if ( api == null )
      {
         return;
```

```
      }
      else
      {
         var result = api.SetValue( name, value );

         if ( result != "true" )
         {
            var errCode = retrieveLastErrorCode();

            displayErrorInfo( errCode );

            // may want to do some error handling
         }

         return result;
      }
   }

   return;
}

function retrieveLastErrorCode()
{
   // It is permitted to call GetLastError() after Terminate()

   var api = getAPIHandle();

   if ( api == null )
   {
      return "";
   }
   else
   {
      return api.GetLastError();
   }
}

function retrieveErrorInfo( errCode )
{
   // It is permitted to call GetLastError() after Terminate()

   var api = getAPIHandle();

   if ( api == null )
   {
      return "";
   }
   else
   {

      return api.GetErrorString( errCode );
   }
}

function retrieveDiagnosticInfo( error )
{
   // It is permitted to call GetLastError() after Terminate()
```

```
   var api = getAPIHandle();

   if ( api == null )
   {
      return "";
   }
   else
   {
      return api.GetDiagnostic( error );
   }
}

function persistData()
{
   // do not call a set after Terminate() was called
   if ( terminated != "true" )
   {
      var api = getAPIHandle();

      if ( api == null )
      {
         return "";
      }
      else
      {
         return api.Commit();
      }
   }
   else
   {
      return "";
   }
}

function displayErrorInfo( errCode )
{
   if ( _debug )
   {
      var errString = retrieveErrorInfo( errCode );
      var errDiagnostic = retrieveDiagnosticInfo( errCode );

      alert( "ERROR: " + errCode + " - " + errString + "\n" +
             "DIAGNOSTIC: " + errDiagnostic );
   }
}
```

Anhang D – Klassenübersicht der SmartFoxServer Client API

Die SmartFoxServer Client API enthält die drei Hauptklassen `SmartFoxClient`, `Room` und `User`.

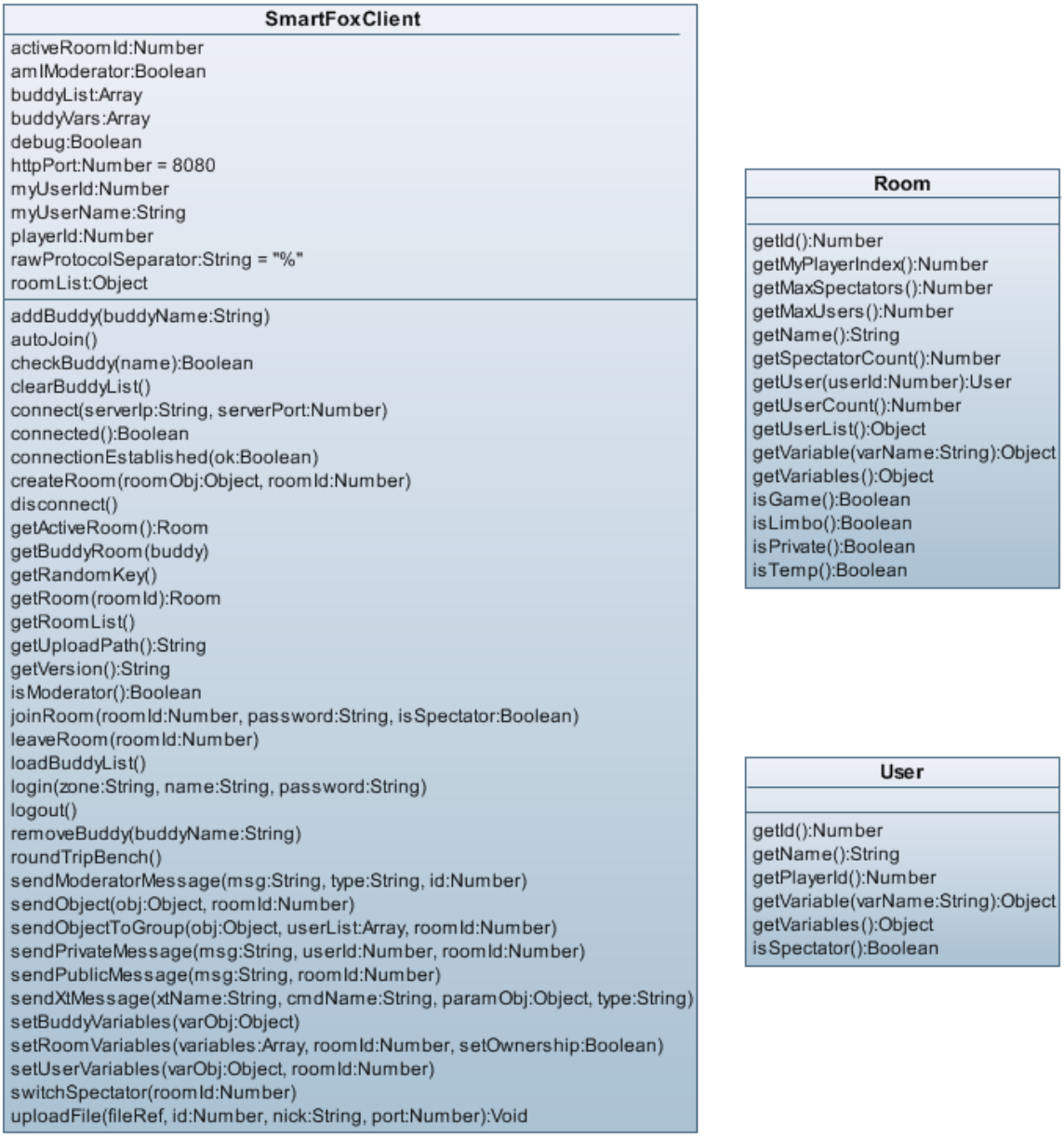

Abbildung Anhang D: Übersicht über die Klassen der SmartFoxServer Client API

Anhang E – config.xml

Die Datei wird zur Konfiguration des SmartFoxServer benötigt. Ohne die Definition der benötigten Datenstruktur innerhalb des `<zones>` Elementes, ist die prototypische Implementierung nicht ausführbar.

Listing Anhang E: config.xml

```
<SmartFoxConfig>

 <ServerSetup>

  <ServerIP>127.0.0.1</ServerIP>
  <ServerPort>9339</ServerPort>

  <AutoSendPolicyFile>true</AutoSendPolicyFile>
  <MaxUserIdleTime>300</MaxUserIdleTime>

  <!-- Server Variables limits (-1 = unlimited) -->
  <MaxRoomVars>-1</MaxRoomVars>
  <MaxUserVars>-1</MaxUserVars>

  <AntiFlood active="false">
    <MinMsgTime tolerance="5">1000</MinMsgTime>
    <MaxRepeatedMessages>3</MaxRepeatedMessages>
    <WarningsBeforeKick>2</WarningsBeforeKick>
    <WarningMessage><![CDATA[...)]]></WarningMessage>
    <KickMessage><![CDATA[...]]></KickMessage>
    <BanMessage><![CDATA[...]]></BanMessage>
    <BanAfter timeSpan="1">3</BanAfter>
  </AntiFlood>

  <BadWordsFilter active="false">
    <FilterMode>filter</FilterMode> <!-- REMOVE or FILTER -->
    <StripCharacters><![CDATA[,.;:_!$%&/#*-+]]></StripCharacters>
    <Warnings>true</Warnings>
    <FilterRoomNames>true</FilterRoomNames>
    <FilterUserNames>true</FilterUserNames>
    <WarningsBeforeKick>3</WarningsBeforeKick>
    <WarningMessage><![CDATA[...]]></WarningMessage>
    <KickMessage><![CDATA[...]]></KickMessage>
    <BanMessage><![CDATA[...]]></BanMessage>
    <BanAfter timeSpan="1">3</BanAfter>
    <BadWordsList></BadWordsList>
  </BadWordsFilter>

  <BanCleaning>auto</BanCleaning>
  <BanDuration>1800</BanDuration> <!-- 30 min -->
  <BannedLoginMessage>You have been banned!</BannedLoginMessage>

  <OutQueueThreads>1</OutQueueThreads>
  <ExtHandlerThreads>1</ExtHandlerThreads>
  <MaxWriterQueue>50</MaxWriterQueue>
  <MaxIncomingQueue>8000</MaxIncomingQueue>
  <DeadChannelsPolicy>strict</DeadChannelsPolicy>
```

```
  <MaxMsgLen>4096</MaxMsgLen>
  <LogMaxSize>5000000</LogMaxSize>
  <LogMaxFiles>5</LogMaxFiles>

  <FileLoggingLevel>WARNING</FileLoggingLevel>
  <ConsoleLoggingLevel>INFO</ConsoleLoggingLevel>
  <AdminLogin>sfs_admin</AdminLogin>
  <AdminPassword>sfs_admin</AdminPassword>

  <AdminAllowedAddresses>
   <AllowedAddress>*.*.*.*</AllowedAddress>
  </AdminAllowedAddresses>

  <IpFilter>5</IpFilter>

  <!-- Enable / Disable remote zone info -->
  <EnableZoneInfo>true</EnableZoneInfo>

 </ServerSetup>

 <Zones>
  <Zone name="level_1" emptyNames="true">
   <Rooms>
    <Room name="Lobby" maxUsers="50" isPrivate="false" isTemp="false"
          autoJoin="true" />
   </Rooms>
  </Zone>
  <Zone name="level_2" emptyNames="true">
   <Rooms>
    <Room name="Lobby" maxUsers="50" isPrivate="false" isTemp="false"
          autoJoin="true" />
   </Rooms>
  </Zone>
  <Zone name="level_3" emptyNames="true">
   <Rooms>
    <Room name="Lobby" maxUsers="50" isPrivate="false" isTemp="false"
          autoJoin="true" />
   </Rooms>
  </Zone>
 </Zones>

</SmartFoxConfig>
```

Anhang F – CD

Auf der CD befinden sich, neben dieser Arbeit, der entwickelte Prototyp als Content Package und PIF sowie die dazugehörigen Quelldateien. Auf der CD befinden sich des Weiteren die Anwendungen, die für die Erstellung des Prototyps verwendet wurden, und elektronische Dokumente, die im Rahmen dieser Arbeit benutzt wurden.

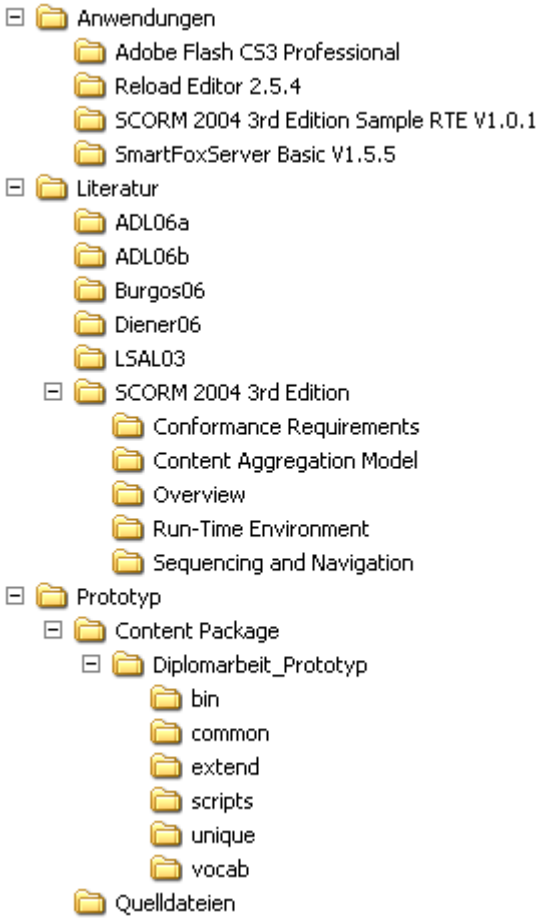

Abbildung Anhang F: CD Inhalt

Selbstständigkeitserklärung

Ich versichere, dass ich die Diplomarbeit ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Dresden, den 13. August 2007 .....

Dirk Börner