February 2005

Entwicklung von Experimentierumgebungen für den Erwerb von Problemlösefähigkeit

Christian Ullrich
Otto-Friedrich-Universität Bamberg

Otto K. Ferstl
Otto-Friedrich-Universität Bamberg

In: Ferstl, Otto K, u.a. (Hg) 2005. *Wirtschaftsinformatik 2005: eEconomy, eGovernment, eSociety*; 7. Internationale Tagung Wirtschaftsinformatik 2005. Heidelberg: Physica-Verlag

ISBN: 3-7908-1574-8

Entwicklung von Experimentierumgebungen für den Erwerb von Problemlösefähigkeit

Christian Ullrich, Otto K. Ferstl
Otto-Friedrich-Universität Bamberg

Zusammenfassung: Durch den Einsatz computergestützter Lernumgebungen kann die Effektivität und Effizienz von Lehr-/ Lernprozessen erheblich gesteigert werden. Dies gilt vor allem für den Bereich „Anwenden von Wissen in Problemlösesituationen", da computergestützte Lernumgebungen gegenüber herkömmlichen Lernsituationen insbesondere eine bessere Präsentation und Simulation der Problemfälle sowie hohe Interaktionsmöglichkeiten für die Lerner ermöglichen. Der vorliegende Beitrag stellt ein Framework zur Erstellung von qualitativ hochwertigen Experimentierbaukästen vor, die zur Unterstützung des gesamten Lernprozesses in SCORM-fähige Lernumgebungen integriert werden können.

Schlüsselworte: Problemlösefähigkeit, Lernumgebungen, E-Learning, Framework, Komponentenbasierte Software-Entwicklung

1 Einleitung

Lebenslanges Lernen, d.h. Ausbildung und kontinuierliche Weiterbildung, ist für die individuellen Lebens- und Arbeitschancen von wesentlicher Bedeutung. Es umfasst die Aktualisierung und Erweiterung von Faktenwissen ebenso wie die Verbesserung der Problemlösekompetenz. Gegenwärtig unterstützen die Informations- und Kommunikationstechnologien (IuK-Technologien) vor allem die Beschaffung von Faktenwissen, wie der Erfolg des World Wide Web zeigt. Mindestens ein ebenso hoher Unterstützungsbedarf besteht bei Problemlöseprozessen, denn die Problemlösekompetenz ist entscheidend für Berufschancen und beruflichen Erfolg.

Problemlösekompetenz bedeutet zum einen, formale Entscheidungsverfahren zu kennen, und zum anderen, die Fähigkeit zu besitzen, diese in Problemsituationen mit Erfolg anwenden zu können. Um das Anwenden von Wissen in realitätsnahen Problemsituationen zu üben, eignen sich insbesondere computergestützte Simulationsumgebungen. Für einige Anwendungsbereiche ist bereits eine Vielzahl von Simulationswerkzeugen verfügbar. Diese Werkzeuge dienen der Durchführung von Simulationsexperimenten, sind jedoch kaum für die Unterstützung von Aus- und Weiterbildungssituationen geeignet. Oft fehlen notwendige Hilfsleistungen,

wie z.B. Moderation oder eine geeignete Beratung und Betreuung der Lerner. Auch die Integration dieser Werkzeuge in Lernumgebungen, die den gesamten Lernprozess unterstützen, gestaltet sich meist sehr schwierig. Die Hilfsleistungen und die vollständige Integration der Werkzeuge sind aber für eine effektive und effiziente Vermittlung von Problemlösefähigkeit unbedingt erforderlich.

Ziel des Beitrags ist es, ein Framework vorzustellen, das es ermöglicht, Simulationswerkzeuge in den Kontext von Aus- und Weiterbildungssituationen (z.B. Fallstudien) zu integrieren. Das Framework verfügt über Standardkomponenten, mit denen die Hilfsleistungen erbracht werden können, die nicht in den Simulationswerkzeugen enthalten sind. Zusätzlich soll mit Hilfe des Frameworks die Erstellung von problemorientierten Lernsettings erleichtert und die Wiederverwendbarkeit sowie Wartbarkeit der Softwarekomponenten erhöht werden.

2 Erwerb von Problemlösefähigkeit

Die Realisierung und Nutzung computergestützter Lernumgebungen erfordert eine Analyse der zu unterstützenden Lernprozesse. Im vorliegenden Fall wird dazu der Gegenstandsbereich „Erwerb von Problemlösefähigkeit“ anhand seiner Struktur und seines Verhaltens näher betrachtet.

2.1 Problem und Problemlösen

Erster Schritt eines Problemlöseprozesses ist die Abgrenzung des zu betrachtenden Problems. Hierzu wird zwischen einem Realitätsausschnitt als Problembereich und einem Individuum als Problemlöser unterschieden. Ein Problem liegt vor, wenn der Realitätsausschnitt sich in einem Zustand befindet, den das Individuum für nicht wünschenswert hält, es jedoch nicht über Mittel verfügt, den unerwünschten Zustand in einen wünschenswerten Zustand zu überführen [Dörn87, 10]. Ein Problem ist somit durch folgende Merkmale gekennzeichnet:

a. Ausgangspunkt ist ein unbefriedigender Zustand des problemrelevanten Realitätsausschnitts.

b. Angestrebt wird ein erwünschter Zielzustand des Realitätsausschnitts.

c. Lösungsverfahren zur Transformation des Anfangszustandes in den Zielzustand sind nicht durchführbar oder dem Individuum unbekannt.

Eine Differenzierung unterschiedlicher Problembereiche anhand von Struktur- und Verhaltensaspekten sowie eine Differenzierung unterschiedlicher Arten von Ausgangs- und Zielzuständen ermöglichen die Bildung von Problemklassen, die jeweils spezifische Problemlösefähigkeiten erfordern [Fers79, 44ff]:

- **Konstruktionsprobleme:** Ausgangspunkt ist ein System als Problembereich mit beobachtbarem oder postuliertem Verhalten, aber unbekannter Struktur. Erwünscht ist die Ermittlung einer mit dem Verhalten kompatiblen Struktur.
- **Analyseprobleme:** Ausgangspunkt hier ist ein System mit bekannter Struktur, aber unbekannten oder nur teilweise bekannten Verhaltensmerkmalen. Erwünscht ist die Ermittlung ausgewählter Verhaltensmerkmale. Nach der Art der gesuchten Verhaltensmerkmale werden folgende Teilklassen dieser Problemklasse unterschieden. Bei einem **Input-Output-Problem** wird nach der Reaktion eines Systems auf einen vorgegebenen Input gefragt. Im Gegensatz dazu sind bei einem **Output-Input-Problem** Inputs zu bestimmen, die zu einem vorgegebenen Output führen.
- **Entscheidungsprobleme:** Weisen Output-Input-Probleme Freiheitsgrade auf, d.h. kann ein vorgegebener Output von mehreren Inputs aus erreicht werden, ist ggf. eine Auswahl der Inputs anhand einer Bewertung erforderlich. Eine solche Bewertung wird in Form eines Präferenzsystems spezifiziert.
- **Black-Box-Probleme:** Analyseprobleme unterstellen eine bekannte Struktur des Systems. Sind Verhaltensmerkmale bei unbekannter Struktur zu ermitteln, liegen Black-Box-Probleme vor. Sie können wie Analyseprobleme weiter differenziert werden in Input-Output- und Output-Input-Probleme.

Die gesuchten Problemlösungen sind von einem Individuum in einem schrittweisen Problemlöseprozess zu ermitteln. Dabei wirken die Aktionen des Problemlösers auf den problemrelevanten Realitätsausschnitt ein und führen dort zu Zustandsveränderungen [Edel00, 205]. Um ein Problem erfolgreich bearbeiten zu können, benötigt der Problemlöser sowohl epistemisches als auch heuristisches Wissen [Dörn87, 26ff]. **Epistemisches Wissen** umfasst Faktenwissen sowie das Wissen bezüglich geeigneter Lösungsverfahren. Die Anpassung und Weiterentwicklung des epistemischen Wissens erfolgt in mentalen Prozessen des Problemlösers, sog. Akkomodationsprozessen [Glas97, 117ff]. Wissen über die Gestaltung von Akkomodationsprozessen als wird **heuristisches Wissen** bezeichnet. Es umfasst Operatoren, mit denen die kognitiven Strukturen angepasst werden können. Heuristisches Wissen ist nicht problemspezifisch, sondern es enthält eine Bibliothek von Lösungsverfahren, die zur Generierung und Anpassung von Verfahren zur Lösung konkreter Problemsituationen dienen.

2.2 Problemorientierte Lernsettings

Eine Verbesserung der Problemlösefähigkeit einer Person besteht im Aufbau bzw. in der Weiterentwicklung ihres epistemischen und heuristischen Wissens [Spi+88; Dörn87, 116].

Eine Erweiterung des epistemischen Wissens bezüglich einer Wissensdomäne führt dazu, dass ein Individuum in der Lage ist, Probleme, die sich auf diesen Be-

reich beziehen, effektiver und effizienter zu lösen. Im Gegensatz dazu verbessert eine Weiterentwicklung des heuristischen Wissens die Fähigkeit, neue Lösungsverfahren zu entwickeln oder verfügbare Verfahren zu verbessern. Dadurch wird die allgemeine, bereichsunabhängige Problemlösefähigkeit einer Person gestärkt.

Die Weiterentwicklung des heuristischen Wissens umfasst [And00, 289ff]

- das Üben von mentalen Operatoren, aus denen ein Problemlöseprozess besteht,
- das Verbessern der Fähigkeit, einen Problemlöseprozess zu strukturieren und
- ein Übungstraining, in dem der Lerner sein erworbenes Wissen in unterschiedlichen Problemsituationen anwendet.

Die Vermittlung von Problemlösefähigkeit erfolgt mit Hilfe spezieller Lernsysteme, sog. **problemorientierter Lernsettings**[1]. Diese Lernsettings schulen nicht nur das epistemische und heuristische Wissen unter Verwendung realitätsnaher Problemsituationen, sondern dienen auch zur Vermittlung verschiedener Schlüsselqualifikationen, wie z.B. soziale Kompetenz, Selbstständigkeit und Entscheidungskompetenz.

In der Aus- und Weiterbildung finden die problemorientierten Lernsettings *Projekt*, *Planspiel*, *Rollenspiel*, *Fallstudie* und *Experiment* Anwendung [Mey87, 143ff].

In einem **Projekt** ist meist ein Konstruktionsproblem in Gruppenarbeit zu lösen [Frey98; Gud01, 83ff]. Teil der Projektaufgabe ist häufig die Festlegung der Sach- und Formalziele des Projekts durch die Mitglieder des Projektteams. Auch das Lösungsverfahren zur Durchführung des Projekts ist von den Gruppen selbstständig zu ermitteln und auszuführen. Der Lehrer steht lediglich als Experte beratend zu Seite. Die Dauer von Projekten kann je nach Aufgabenstellung sehr unterschiedlich ausfallen (von wenigen Stunden bis zu mehreren Jahren).

Bei **Planspielen** bilden die Teilnehmer Gruppen und lösen Entscheidungsprobleme in komplexen Situationen, die häufig mit Hilfe dynamischer Modelle simuliert werden [Geu00, 16; BöWo00]. Der Zustand einer Problemsituation ändert sich zum einen durch die Aktionen der Teilnehmer und zum anderen durch die Eigendynamik des Systems. Je nach Aufgabenstellung agieren die Mitspieler in Konkurrenz, in Kooperation oder im Konflikt miteinander. Analog zu Projekten tritt der Lehrer auch hier lediglich als Berater bzw. Experte auf.

In **Rollenspielen** übernehmen Lerner Rollen und lösen spielerisch Entscheidungsprobleme in vorgegebenen Problemsituationen [Bia95, 84ff; Schal01]. Im Gegensatz zu Planspielen enthalten Rollenspiele meist nur geringe formale Ablaufstrukturen in Form von Szenen und Szenenfolgen. Damit gestehen sie den Teilnehmern

[1] Ein Lernsetting ist eine konkrete Ausprägung eines Lernsystems – bestehend aus Lerner und Lernumgebung – über eine fest definierte Zeitspanne [May01, 260].

eine große Handlungsfreiheit zu, so dass diese ihre Rolle in geeigneter Weise ausfüllen können. Die Durchführung von Rollenspielen wird von einer Spielleitung gelenkt. Diese startet, beendet und unterbricht die Szenen, um z.B. Diskussionen oder einen Rollentausch durchzuführen.

Fallstudien konfrontieren Lernende mit praxisrelevanten Entscheidungsproblemen [Kais83, 21; Budd92]. Sie beschreiben meist Problemsituationen, die bereits alle relevanten Fakten zur Lösung des jeweiligen Problems enthalten. Aufgabe der Lernenden ist es, in Einzel- oder Gruppenarbeit die Problemsituationen zu analysieren, Lösungsalternativen zu ermitteln und sich anschließend für eine der Lösungsmöglichkeiten zu entscheiden. Das Durchsetzen der Entscheidungen und der Umgang mit deren Konsequenzen werden in Fallstudien in der Regel nicht betrachtet. Während der Bearbeitung einer Fallstudie übernimmt der Lehrer ausschließlich die Rolle des Experten und des Moderators.

Ein didaktisches **Experiment** umfasst ein Analyseproblem, welches die Lerner auf explorative Art und Weise lösen sollen [Rein96; Schma82, 302f]. Zu Beginn des Experiments entwickeln die Lerner Hypothesen über den Sachverhalt. Anschließend entwerfen und realisieren sie Versuchsanordnungen zur Überprüfung und Revision ihrer Hypothesen. Je nach Art der verfolgten Lernziele kann der Lehrer entweder die Experimentdurchführung fest vorgeben und steuern oder die Lerner lediglich als Experte und Berater unterstützen.

2.3 Invariante Merkmale problemorientierter Lernsettings

Der Ablauf eines problemorientierten Lernsettings besteht aus einem Haupt- und aus fünf Begleitprozessen. Den eigentlichen **Hauptprozess** stellt die Spiel-, Projekt-, oder Experimentdurchführung dar. Obwohl dieser Prozess bei den verschiedenen Arten problemorientierter Lernsettings sehr unterschiedlich ist, lässt sich dennoch eine vom Typ des Lernsettings unabhängige Ablaufstruktur erkennen. So ist der Hauptprozess meist in die Phasen *Vorbereitung*, *Durchführung* und *Abschluss* gegliedert (Abbildung 1).

In der **Vorbereitungsphase** werden Gruppen gebildet, Rollen bzw. Aufgaben übernommen und das Problem definiert. Die **Durchführungsphase** gestaltet sich je nach Typ des Lernsettings unterschiedlich. Invariant ist nur, dass sie beliebig oft und an beliebiger Stelle durch Diskussion, Reflexion und Auswertung von Spielergebnissen und Verhaltensweisen der Teilnehmer unterbrochen werden kann. Die **Schlussphase** ist bei allen fünf Lernsettings nahezu identisch. Zuerst werden die Ergebnisse und Verhaltensweisen der Gruppen ermittelt und anschließend in einer Diskussionsrunde reflektiert. Darüber hinaus werden die notwendigen Aufräumarbeiten und eine Abschlussveranstaltung durchgeführt.

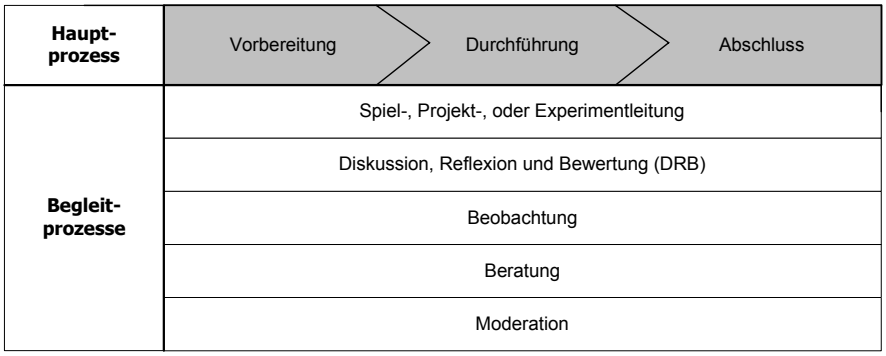

Abbildung 1: Generischer Ablauf problemorientierter Lernsettings

Neben dem Hauptprozess beinhalten problemorientierte Lernsettings auch die **Begleitprozesse**

a. Spiel-, Projekt-, oder Experimentleitung,

b. Diskussion, Reflexion und Bewertung (DRB),

c. Beobachtung,

d. Beratung und

e. Moderation.

Die Begleitprozesse der verschiedenen Lernsettingtypen unterscheiden sich nur geringfügig voneinander. Deshalb können die Begleitprozesse als invariant gegenüber den unterschiedlichen Arten von Lernsettings angesehen werden.

Der Begleitprozess **Spiel-, Projekt-, oder Experimentleitung** umfasst alle notwendigen Aufgaben zur Planung, Steuerung und Kontrolle eines Lernsettings, wie z.B. Aufgabenplanung, Terminplanung und Ressourcenverwaltung.

Die Phase **Diskussion, Reflexion und Bewertung (DRB)** schafft die wesentlichen Voraussetzungen für den Erwerb von Problemlösefähigkeit und Schlüsselqualifikationen [Frey98, 192ff]. Sie ermöglicht es den Lernern, die Geschehnisse während der Durchführungsphase nachträglich aus der Außensicht zu betrachten und zu beurteilen. Erst dadurch können die Lerner ihr epistemisches und heuristisches Wissen in geeigneter Weise anpassen. Für die Durchführung der DRB sind Datenanalysen sowie Beratungs- und Moderationsleistungen erforderlich. Diese werden durch die Begleitprozesse *Beratung*, *Beobachtung* und *Moderation* bereitgestellt.

Die **Beobachtung** protokolliert alle relevanten Merkmale der Spiel-, Projekt-, oder Experimentdurchführung. Die gesammelten Daten (in Form von Verteilungen) und darauf aufbauende Analysen (Varianz, Mittelwert etc.) werden dem Begleit-

prozess *Beratung* und der *Spiel-, Projekt-, oder Experimentdurchführung* zur Verfügung gestellt.

Aufgabe der **Beratung** ist es, die jeweilige Problemsituation sowie das Verhalten des Lerners zu analysieren und darauf aufbauend den Lerner durch konstruktive Hinweise zu einer Lösung des Problems hinzuführen. Für die Analyse der Problemsituation und des Lernerverhaltens ist Datenmaterial erforderlich, welches vom Begleitprozess *Beobachtung* erstellt wird.

Die Strukturen aller problemorientierten Lernsettings weisen untereinander große Ähnlichkeit auf. Ein gemeinsames Strukturmerkmal ist die Trennung zwischen einer Gesamtdurchführung einerseits und einem zentralen Lenkungsobjekt – der Spiel-, Projekt-, oder Experimentleitung – andererseits (Abbildung 2).

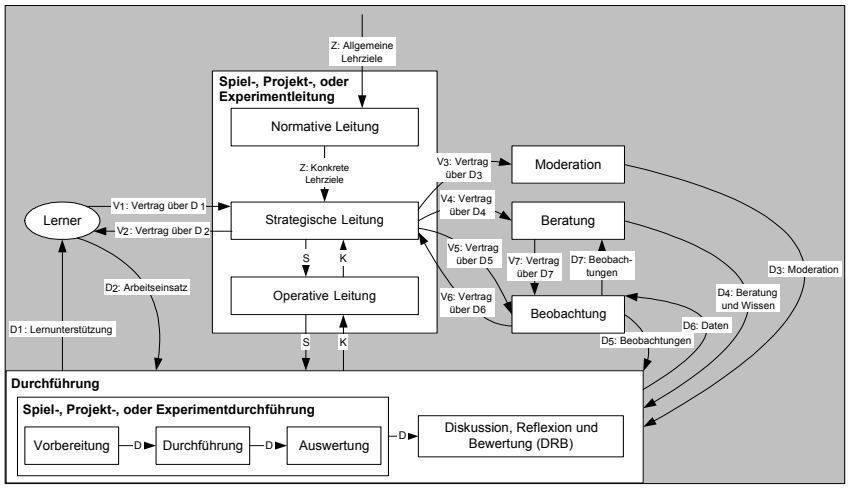

Abbildung 2: Generische Struktur problemorientierter Lernsettings (Interaktionsschema)[2]

Die **Spiel-, Projekt-, oder Experimentleitung** koordiniert den gesamten Austausch von Leistungen innerhalb des Lernsettings. Sie verhandelt mit den Serviceprozessen **Beratung**, **Moderation** und **Beobachtung** hinsichtlich der Abgabe entsprechender Leistungen an die Gesamtdurchführung. Diese kann nach dem Objektprinzip in die Teilobjekte **Spiel-, Projekt-, oder Experimentdurchführung** und **DRB** zerlegt werden. Die Spiel-, Projekt-, oder Experimentdurchführung stellt den Hauptprozess des Lernsettings dar.

Das Lenkungssystem der problemorientierten Lernsettings ist hierarchisch in drei Ebenen strukturiert.[3] Auf der obersten Ebene ermittelt die **normative Leitung** aus allgemeinen Lehrzielen konkrete Lehrziele für das jeweilige Lernsetting und

2 Die Modellierung erfolgt mit Hilfe der SOM-Methodik [FeSi01, 180ff].

3 Vergleiche die Grundstruktur lebensfähiger Systeme in [Beer94, 135ff].

reicht diese an die **strategische Leitung** weiter. Deren Aufgabe ist die Koordination der **Spiel-, Projekt-, oder Experimentdurchführung** mit den **Lernern** und den Begleitprozessen (**Moderation**, **Beratung** und **Beobachtung**). Die **operative Leitung** umfasst die Aufgaben-, Termin-, und Ressourcenplanung für die Spiel-, Projekt-, oder Experimentdurchführung und für die DRB. Darüber hinaus stimmt sie die einzelnen Teilphasen aufeinander ab und überwacht die Ergebnisse derselben.

3 Computergestützte Lernumgebungen

Computergestützte Lernumgebungen nutzen Informations- und Kommunikationstechnologien, um Aufgaben der Lernunterstützung und Kommunikationskanäle zwischen Lerner und Lernumgebung zu automatisieren [Kerr98; Dick00].

Wichtig für die Effektivität und Effizienz von Lehr-/ Lernprozessen ist, dass eine Lernumgebung von den Nutzern als integrierte Einheit wahrgenommen wird. Deshalb sollten computergestützte Lernumgebungen z.B. über eine einheitliche Benutzungsoberfläche, über eine durchgängige didaktische Gestaltung und über ein komponentenübergreifendes Zugangssystem verfügen. Darüber hinaus sollte eine Lernumgebung alle Phasen des Lehr-/ Lernprozesses, d.h. den Erwerb von Wissen, das Üben von Wissen und das Anwenden von Wissen unterstützen [Schm01, 256ff]. Computergestützte Lernumgebungen, die diese Anforderungen erfüllen, werden im Folgenden als integrierte Lernumgebungen bezeichnet.

3.1 Funktionalität von integrierten Lernumgebungen

Zur Unterstützung des gesamten Lehr-/ Lernprozesses verfügt eine integrierte Lernumgebung über verschiedene Dienste, die sich in die Klassen *Basisdienste*, *Lehrdienste* und *Lernmanagementdienste* gliedern lassen [Fers+00; FeSc01; Schm01].

Die **Basisdienste** *Dialogdienst*, *Navigationsdienst* und *Sitzungsdienst* erbringen Hilfsleistungen für andere Dienste der Lernumgebung. Mit Hilfe des Dialogdienstes wird der wechselseitige Nachrichtenaustausch zwischen Nutzer und Lernumgebung realisiert. Unter Verwendung des Dialogdienstes liefert der Navigationsdienst eine Übersicht über die Funktionalität und über die angebotenen Leistungen der Lernumgebung. Die Steuerung des Ablaufs einer Sitzung erfolgt mit Hilfe des Sitzungsdienstes.

Die **Lehrdienste** *Wissensvermittlungsdienst*, *Übungsdienst* und *Experimentierdienst* stellen den Lernern Lehrleistungen zur Verfügung. Der Wissensvermittlungsdienst dient zur Präsentation von Lerninhalten. Zur Kontrolle des Lernfortschritts bietet der Übungsdienst verschiedene Übungsaufgaben mit entsprechender

Auswertung der Lösungsvorschläge an. Das Anwenden des erworbenen Wissens in unterschiedlichen Problemsituationen erfolgt mit Hilfe des Experimentierdienstes.

Die **Lernmanagementdienste** *Kommunikationsdienst*, *Planungsdienst* und *Verwaltungsdienst* unterstützen die Planung, Steuerung und Kontrolle des gesamten Lehr-/ Lernprozesses. Der Kommunikationsdienst realisiert Kommunikationskanäle zwischen den Nutzern einer Lernumgebung. Zur Unterstützung der Termin-, Aufgaben-, und Ressourcenplanung hält der Planungsdienst entsprechende Werkzeuge bereit. Der Verwaltungsdienst verfügt über Funktionen zur Authentifizierung, Autorisierung und Abrechnung (Billing).

Eine integrierte Lernumgebung besteht aus mehreren Komponenten, die als Teilumgebungen bezeichnet werden. Jede Teilumgebung kombiniert verschiedene Dienste, um eine spezifische Leistung zur Unterstützung des Lehr-/ Lernprozesses zu erbringen. Diejenige Teilumgebung, die problemorientierte Lernsettings für den Erwerb von Problemlösefähigkeit zur Verfügung stellt, wird als Experimentierumgebung bezeichnet. Die Experimentierumgebung unterstützt sowohl den Hauptprozess problemorientierter Lernsettings (Spiel-, Projekt-, oder Experimentdurchführung) als auch die Begleitprozesse *Leitung*, *Beobachtung*, *Beratung* und *Moderation*. Darüber hinaus umfasst die Experimentierumgebung auch Werkzeuge für Autoren und Betreuer, mit denen problemorientierte Lernsettings erstellt, konfiguriert und den Lernern zur Verfügung gestellt werden können.

3.2 Framework für Experimentierumgebungen

Wesentliches Ziel des hier vorgestellten Frameworks für Experimentierumgebungen ist die vollständige Integration problemorientierter Lernsettings in computergestützte Lernumgebungen. Um diese Integration zu ermöglichen, stützt sich das Framework auf das Sharable Content Object Reference Model (SCORM) der Advanced Distributed Learning Initiative (ADL) [ADL04a].

Mit SCORM hat ADL einen Standard geschaffen, der es ermöglicht, Lerninhalte wieder verwendbar, weltweit verfügbar und unabhängig von Soft- und Hardware-Umgebungen zu gestalten. SCORM kapselt Lerninhalte in Form von Lernobjekten, die eine fest definierte Schnittstelle aufweisen. Darüber hinaus können mehrere Lernobjekte zu einer Lerneinheit zusammengefasst werden. Zur Beschreibung der Struktur und des Ablaufs von Lerneinheiten verfügt SCORM über ein geeignetes Meta-Modell, das sog. Content Aggregation Model (CAM) [ADL04b].

Der SCORM-Ansatz sieht vor, Lernobjekte unabhängig von Lernangeboten zu erstellen und erst während der Laufzeit zu Lerneinheiten zusammen zu fassen. Dadurch ist eine weitgehende Entkopplung der Erstellungs-, Bereitstellungs- und Nutzungsphase von Lerninhalten möglich. Dieser Aspekt ist von besonderer Bedeutung, da die Erstellung von hochwertigen Experimentierumgebungen mit er-

heblichem Aufwand verbunden ist und somit meist nicht zeitnah zur Nutzungsphase sowie aus Effizienzgründen nicht ausschließlich für eine einzige Lerneinheit erfolgen kann.

Das Framework für Experimentierumgebungen realisiert die Komponenten der Spiel-, Projekt-, und Experimentdurchführung (vgl. Abbildung 2) in Form von SCORM-kompatiblen Lernobjekten. Dadurch kann die Struktur der Experimentierumgebung durch jedes SCORM-kompatible Lern-Management-System (LMS) visualisiert und zur Navigation genutzt werden. Darüber hinaus ist auch die Speicherung von Experimentierdurchführungen und deren Ergebnissen durch das LMS vorgesehen. Um eine Plattformunabhängigkeit der Experimentierumgebungen zu ermöglichen, erfolgt die Realisierung des Frameworks in der Programmiersprache JAVA.

3.2.1 Konzeption des Frameworks für Experimentierumgebungen

Das Framework besteht aus zwei Teilbereichen: einem Runtime-Bereich und einem Buildtime-Bereich (Abbildung 3). Der **Runtime-Bereich** dient zur Entwicklung von Anwendungssystemen, die die Haupt- und Begleitprozesse problemorientierter Lernsetings während der Durchführungsphase unterstützen. Die Ausführung der Anwendungssysteme kann entweder als Teil einer integrierten Lernumgebung oder als eigenständige Applikation erfolgen.

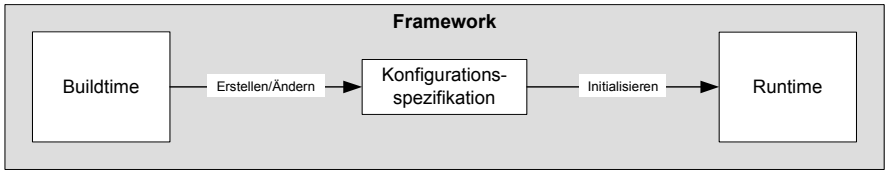

Abbildung 3: Framework für Experimentierumgebungen (Übersicht) [Ull03, 228]

Der **Buildtime-Bereich** des Frameworks bildet die Grundlage für die Entwicklung von Werkzeugen, mit denen die Runtime-Anwendungssysteme parameterisiert, d.h an die Bedürfnisse konkreter Lernsettings angepasst werden können. Parameter, die zur Buildtime spezifiziert werden müssen, sind u.a. das Simulationsmodell, Parameter für die Simulationsexperimente, Beobachtungsvariablen und Auswertungsregeln für automatisierte Rückmeldungen.

Die Kopplung zwischen Runtime- und Buildtime-Anwendungssystemen erfolgt mittels sog. **Konfigurationsspezifikationen**. Diese enthalten alle zur Durchführung eines Lernsettings notwendigen Parameter. Die Werte der Parameter werden mit Hilfe der Buildtime-Werkzeuge festgelegt und anschließend an die Runtime-Anwendungssysteme übermittelt. Dort werden die Parameter zur Instanziierung neuer Lernsettings genutzt.

3.2.2 Konzeption des Runtime-Frameworks für Experimentierumgebungen

Grundlage für die Realisierung des Runtime-Frameworks bildet die generische Struktur problemorientierter Lernsettings (Abbildung 2). Anhand dieser Struktur wird festgelegt, welche Aufgaben (teil-)automatisierbar sind und von der Experimentierumgebung unterstützt werden sollen. Da die generische Struktur problemorientierter Lernsettings sehr abstrakt ist, kann der Automatisierungsgrad vieler Aufgaben lediglich in Form eines Intervalls angegeben werden. Darüber hinaus hängt der Automatisierungsgrad mancher Aufgaben auch von der jeweiligen Instanz des Lernsettings ab. So ist es z.B. in bestimmten Situationen aus didaktischen Gründen sinnvoll, Beratungsaufgaben personell statt maschinell gestützt durchzuführen. Deshalb wird das Framework für Experimentierumgebungen flexibel gegenüber dem Automatisierungsgrad dieser Aufgaben gestaltet.

Die Aufgaben der **Spiel-, Projekt-, und Experimentleitung** sind teilautomatisierbar. Sie werden jedoch gemäß dem SCORM-Standard von Lern-Management-Systemen unterstützt [ADL04a, 1-23ff] und sind somit nicht Bestandteil der Experimentierumgebung.

Das Objekt **Moderation** ist ebenfalls teilautomatisierbar. Die Unterstützung von Moderationsaufgaben ist Bestandteil des Kommunikationsdienstes einer Lernumgebung. Deshalb werden diese Aufgaben innerhalb des Frameworks für Experimentierumgebungen ebenfalls nicht betrachtet.

Die Aufgaben des Objekts **Beobachtung** sind automatisierbar. Die Beobachtung zeichnet während der Durchführungsphase die relevanten Zustände des Modellsystems auf. Das Ergebnis der Aufzeichnungen liegt in Form von Verteilungen vor, die jeweils die Ausprägungen einer Modellkomponente in Abhängigkeit der Zeit erfassen. Für das Objekt *Beobachtung* ist lediglich das dynamische Modell der Problemsituation und nicht die Durchführungsform des Lernsettings von Bedeutung. Deshalb umfasst es unabhängig vom Typ des Lernsettings stets die gleichen Aufgaben.

Der Umfang, in welchem die Aufgaben des Objekts **Beratung** automatisierbar sind, hängt von den Eigenschaften des jeweiligen Lernsettings ab. Aus diesem Grund bietet das Framework die Möglichkeit, die Beratung entweder personell mit Hilfe des Kommunikationsdienstes der Lernumgebung (E-Mail, Chat, Videokonferenz) oder maschinell unter Verwendung von Auswertungsregeln durchzuführen. Eine Auswertungsregel für die automatisierte Auswertung besteht aus einer Bedingung und einer Rückmeldung. Die Bedingung wird in Form eines regulären Ausdrucks spezifiziert, der während der Durchführung des Lernsettings gegenüber den Aufzeichnungen des Objektes *Beobachtung* getestet wird. Ist die Bedingung erfüllt, so erfolgt die Ausgabe der korrespondierenden Rückmeldung auf der Benutzungsoberfläche.

Das Objekt **Diskussion, Reflexion und Bewertung (DRB)** ist unter Verwendung geeigneter Groupware-Funktionen des Kommunikationsdienstes zu einem kleinen Teil automatisierbar.

Die Aufgaben der Teilobjekte der **Spiel-, Projekt-, oder Experimentdurchführung** (Vorbereitung, Durchführung und Auswertung) sind je nach Art des Lernsettings sehr unterschiedlich gestaltet. Deshalb lassen sich auch kaum allgemeine Aussagen über deren Automatisierbarkeit treffen. Basiert die Durchführungsphase z.B. auf einem Simulationsmodell, so kann meist die Initialisierung des Modells und die Auswertung der Simulationsexperimente (teil-)automatisiert erfolgen.

Grundsätzlich soll in die Spiel-, Projekt-, oder Experimentdurchführung jedes beliebige (Simulations-)Werkzeug und Untersuchungsobjekt integriert werden können. Deshalb betrachtet das Framework für Experimentierumgebung die Durchführungsphase lediglich aus der Außensicht. Das Framework definiert eine Schnittstelle mit dem Namen *SimulationControl*, die den Nachrichtenaustausch zwischen der Durchführungskomponente und der Experimentierumgebung festlegt. Da die Schnittstelle ausschließlich das Verhalten der Durchführungskomponente beschreibt, kann die Komponente eine beliebige Struktur aufweisen.

Mit Hilfe der Schnittstelle ist es möglich, beliebige Anwendungssysteme, die über eine Programmierschnittstelle verfügen, in die Experimentierumgebung einzubinden. Die Integration erfolgt mittels sog. Wrapper, welche unter Verwendung der Funktionen des jeweiligen Anwendungssystems die Schnittstelle *SimulationControl* implementieren.

3.2.3 Realisierung des Runtime-Frameworks für Experimentierumgebungen

Das Runtime-Framework für Experimentierumgebungen ist zweidimensional strukturiert. In der ersten Dimension ist das Framework nach Objekttypen gegliedert. Gemäß dem SOM-Ansatz werden hierbei folgende Objekttypen unterschieden [FeSi01, 180ff]:

- **Dialogobjekte (IO)** realisieren die Mensch-Computer- oder die Computer-Computer-Kommunikation.
- **Vorgangsobjekte (VO)** beschreiben das Zusammenwirken konzeptueller Objekte bei einem Vorgang.
- **Konzeptuelle Objekte (KO)** repräsentieren die Daten einer Anwendung.

Die zweite Dimension des Runtime-Frameworks für Experimentierumgebungen ist nach den Komponenten problemorientierter Lernsettings (Beobachtung, Beratung und Durchführung) gegliedert.

Zur besseren Wartbarkeit und Wiederverwendbarkeit sieht das Framework vor, zusammengehörige Teile des Anwendungssystems zu sog. Containern zu aggre-

gieren. Jeder Container umfasst ein bis beliebig viele Objekte gleichen Typs. Je nach Art der enthaltenen Objekttypen wird zwischen IO-Container (Dialogobjekte), VO-Container (Vorgangsobjekte) und KO-Container (konzeptuelle Objekte) unterschieden. Container stellen Spezialisierungen ihrer jeweiligen Komponentenobjekte dar, so dass Container beliebig oft rekursiv ineinander geschachtelt werden können. Auf diese Art und Weise lassen sich Softwarekomponenten entwickeln, die sich nach dem Baukastenprinzip sehr leicht zu Anwendungssystemen zusammenbauen lassen.

Die generische Struktur problemorientierter Lernsettings, d.h. die Aufteilung der Spiel-, Projekt-, oder Experiementdurchführung in die Komponenten *Durchführung*, *Beobachtung* und *Beratung*, findet sich auf der obersten Container-Ebene des Frameworks wieder (Abbildung 4). So enthält jeder VO-Container vom Typ Experimentierumgebungsvorgänge (ExpEnvProcedures) einen VO-Container des Typs Expertenvorgänge (ExpertProcedures) und einen des Typs Beobachtervorgänge (ObserverProcedures). Gleiches gilt für die Ebene der KO-Container. Jedem KO-Container vom Typ Experimentierumgebungskonzepte (ExpEnvConcepts) sind der KO-Container eines Experten (ExpertConcepts) und eines Beobachters (ObserverConcepts) zugeordnet.

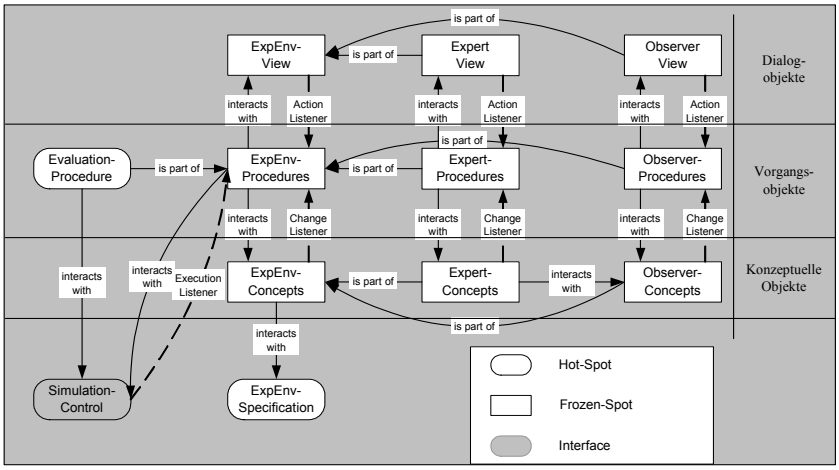

Abbildung 4: Oberste Aggregationsebene des Runtime-Frameworks für Experimentierumgebungen [angelehnt an Ull03, 237]

Für die Auswertung von Spiel-, Projekt-, oder Experimentdurchführungen ist ein Standardauswertungsverfahren vorgesehen. Dieses Verfahren basiert wie oben beschrieben auf einer regelbasierten Auswertung von Beobachtungsergebnissen und kann je nach Bedarf durch Spezialisierung des Vorgangsobjekttyps *EvaluationProcedure* angepasst werden.

3.2.4 Buildtime-Framework für Experimentierumgebungen

Das Buildtime-Framework für Experimentierumgebungenen dient als Basis für die Erstellung von Werkzeugen, mit denen die Spiel-, Projekt-, oder Experimentdurchführung parametrisiert werden kann. Die Parametrisierung erfolgt mit Hilfe von Konfigurationsspezifikationen, die in Form von XML-Dateien vorliegen. In der Regel wird für jeden Typ von Spiel-, Projekt-, oder Experimentdurchführung eine spezielle Konfigurationsspezifikation benötigt. Aus diesem Grund ist für jeden Durchführungstyp ein eigenes Werkzeug zur Bearbeitung der entsprechenden Konfigurationsspezifikation notwendig.

Aus Sicht der Buildtime-Werkzeuge stellen die Konfigurationsspezifikationen konzeptuelle Objekte dar. Die Vorgangsobjekte der Buildtime-Werkzeuge (Bearbeiten, Laden, Speichern etc.) operieren auf den Konfigurationsspezifikationen und führen dort zu Zustandsveränderungen. Jedes Werkzeug besitzt für die Bearbeitung verschiedene Eingabefelder sowie mehrere Dialoge und Hilfsfunktionen, die speziell auf die Bearbeitung des korrespondierenden Typs von Konfigurationsspezifikation ausgerichtet sind.

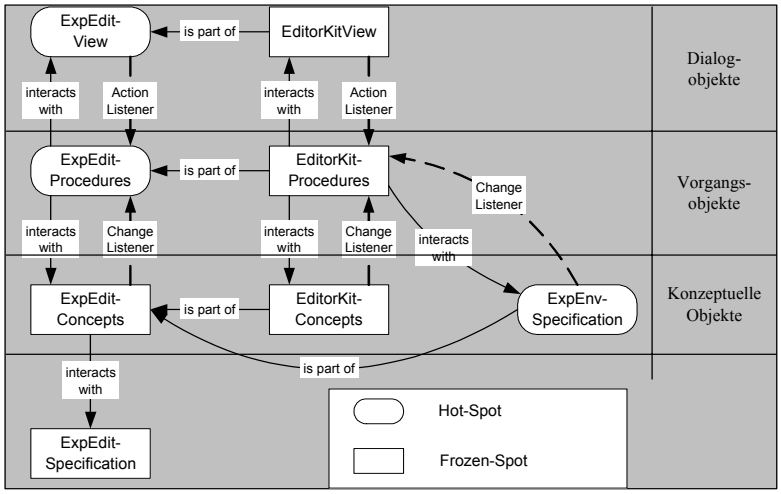

Abbildung 5: Oberste Aggregationsebene des Buildtime-Frameworks für Experimentierumgebungen [angelehnt an Ull03, 246]

Das Buildtime-Framework für Experimentierumgebung ist analog zum Runtime-Framework in die Dimensionen *Objekttypen* und *Komponenten* gegliedert (Abbildung 5). Die oberste Ebene des Frameworks besteht aus dem IO-, VO-, und KO-Container des Buildtime-Werkzeugs (ExpEditView, ExpEditProcedures und ExpEditConcepts). Die zu bearbeitende Konfigurationsspezifikation (ExpEnvSpecification) ist als konzeptuelles Objekt Bestandteil des KO-Containers. Der VO-Container dient zur Instanziierung und Parameterisierung von sog. EditorKits. Al-

le EditorKits sind ebenfalls in IO-, VO- und KO-Container unterteilt und stellen dem Benutzer Funktionen zur Bearbeitung der Konfigurationsspezifikation zur Verfügung. Sie generieren für jedes Attribut einer Konfigurationsspezifikation einen geeigneten Editor (z.B. Eingabefelder für Strings oder Listen für Arrays). Anschließend fassen die EditorKits alle Editoren zu einem Formular zusammen und präsentieren es auf der Benutzungsoberfläche. Alle Informationen darüber, welche Editoren für welche Attribute und mit welchen Parametern zu instanziieren sind, stellt der KO-Container des jeweiligen EditorKits (EditorKitConcepts) zur Verfügung. Diese Informationen lassen sich je nach Bedarf mittels Parameter anpassen.

Jede Konfigurationsspezifikation ist in drei Teilbereiche gegliedert:

a. **Eigenschaften der Durchführung:** Dieser Teilbereich enthält diejenigen Parameter, die an die Spiel-, Projekt-, oder Experiementdurchführungen mit Hilfe der Schnittstelle *SimulationControl* übermittelt werden.

b. **Spezifikation der Beobachtung:** Hier sind die Systemzustände angegeben, deren Änderungen während der Durchführungsphase protokolliert werden sollen.

c. **Spezifikation der Beratung und Auswertung:** Die Spezifikation umfasst die Beratungs- und Auswertungsregeln zur Generierung automatisierter Rückmeldungen an die Lerner während und nach der Durchführungsphase.

3.3 Benutzungsoberfläche der Experimentierumgebung

Die Benutzungsoberflächen des Runtime- und des Buildtime-Bereichs der Experimentierumgebung sind sehr unterschiedlich gestaltet. Der Runtime-Bereich ist in der Regel in eine SCORM-Lerneinheit (Kurs, Seminar etc.) integriert und wird mit dessen Inhalt vernetzt dargestellt [FeUl04]. Im Gegensatz dazu ist der Buildtime-Bereich meist eine eigenständige Java-Applikation, die ausschließlich von den Autoren oder Betreuern der Lerneinheiten und nicht von den Lernern genutzt wird.

3.3.1 Runtime-Benutzungsoberfläche der Experimentierumgebung

Die Runtime-Benutzungsoberfläche ist in zwei Bereiche aufgeteilt (Abbildung 6). Im oberen Bereich wird die Benutzungsoberfläche des (Simulations-)Werkzeugs dargestellt (Methode *getContentPane()* innerhalb der Schnittstelle *SimulationControl*). Darunter befindet sich der Bereich der Bedienelemente, mit welchen der Nutzer den Ablauf der Durchführungsphase steuern kann. Es stehen Funktionen zum Starten, Stoppen und Zurücksetzen der Durchführungsphase zur Verfügung. Darüber hinaus kann sich der Nutzer aufgezeichnete Experimentdurchführungen wieder anzeigen lassen. Mittels eines Schiebereglers ist es hierbei möglich, den zeitlichen Ablauf beliebig zu beeinflussen.

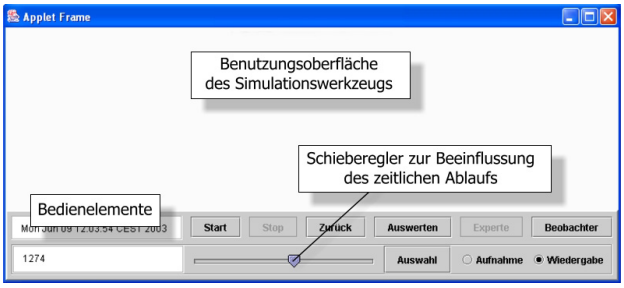

Abbildung 6: Runtime-Benutzungsoberfläche der Experimentierumgebung

Während der Durchführungsphase und während der Wiedergabe hat der Nutzer die Möglichkeit, sich die Werte der aufgezeichneten Systemzustände anzeigen oder sich vom virtuellen Experten beraten zu lassen. Letzterer liefert in Abhängigkeit des aktuellen Zustands des (Simulations-)Modells hilfreiche Hinweise und konstruktive Lösungsvorschläge. Nach Abschluss der Durchführungsphase erfolgt eine regelbasierte Auswertung der Ergebnisse, die dem Nutzer in Form von Text oder HTML-Seiten präsentiert wird.

Alle Aufzeichnungen der Durchführungsphase werden an das Lern-Management-System übermittelt. Dort werden sie gespeichert und stehen anschließend dem jeweiligen Lerner und – falls erwünscht – auch seinem Betreuer zur Verfügung. Ist Letzteres der Fall, so kann der Betreuer die aufgezeichnete Durchführungsphase abspielen lassen und das Verhalten des Lerners auf Basis der protokollierten Systemzustände analysieren. Dadurch ist er in der Lage, dem Lerner detaillierte Rückmeldungen bezüglich der Durchführungsphase z.B. mit Hilfe des Kommunikationsdienstes der Lernumgebung (E-Mail, Chat, Videokonferenz etc.) zu geben.

3.3.2 Buildtime-Benutzungsoberfläche der Experimentierumgebung

Die Bearbeitung der Konfigurationsspezifikation für die Durchführungsphase erfolgt mit Hilfe von Editoren. Da die Konfigurationsspezifikationen je nach Typ der Spiel-, Projekt-, oder Experimentdurchführung unterschiedlich sind, unterscheiden sich auch die Editoren zu deren Bearbeitung. Dennoch sind alle Editoren gemäß der allgemeinen Struktur von Konfigurationsspezifikationen in die Teilbereiche *Eigenschaften der Durchführung*, *Beobachtung*, *Beratung* und *Auswertung* gegliedert (Abbildung 7).

Die Bereiche *Beobachtung*, *Beratung* und *Auswertung* sind vom Typ der Spiel-, Projekt-, oder Experimentdurchführung unabhängig, falls das Standardverfahren zur Auswertung verwendet wird. Im Gegensatz dazu ist der Bereich *Eigenschaften der Durchführung* nicht allgemein spezifizierbar und muss unter Verwendung geeigneter Editorkomponenten (EditorKit) an den jeweiligen Typ der Durchführung angepasst werden.

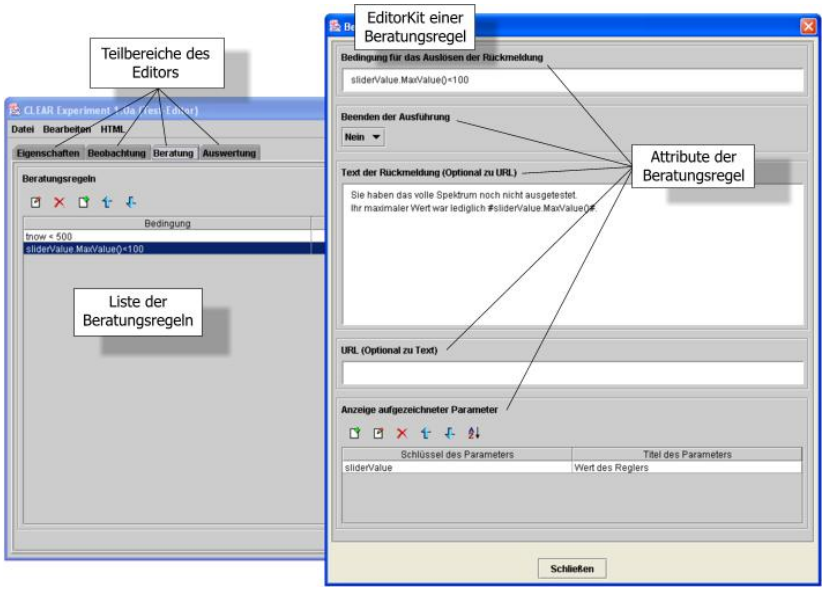

Abbildung 7: Buildtime-Benutzungsoberfläche der Experimentierumgebung

Allgemeine Editorfunktionen, wie z.B. das Laden und Speichern von Dateien, werden vom Framework zur Verfügung gestellt. Darüber hinaus bietet das Framework auch die Möglichkeit, Funktionen zum Ausführen und zum Testen der bearbeiteten Spiel-, Projekt-, oder Experimentdurchführungen zu realisieren.

4 Zusammenfassung und Ausblick

Das vorgestellte Framework bildet die Grundlage für die Entwicklung von Experimentierumgebungen, die sowohl den Hauptprozess problemorienterter Lernsettings (Spiel-, Projekt-, oder Experimentdurchführung) als auch die Begleitprozesse (Leitung, Beobachtung, Beratung und Moderation) unterstützen und in jede SCORM-fähige Lernumgebung integriert werden können. Darüber hinaus ist es Ziel des Frameworks, die Wiederverwendbarkeit und Wartbarkeit der Softwarekomponenten zu erleichtern und die Integration beliebiger Werkzeuge und Simulationsmodelle in die Experimentierumgebung zu ermöglichen. Aus Gründen der Plattformunabhängigkeit ist das Framework in der Programmiersprache JAVA realisiert.

Zur Reduzierung des Aufwands für die Erstellung von Experimentierumgebungen ist es sinnvoll, auf Basis des Frameworks folgende Standardimplementierungen für die Schnittstelle *SimulationControl* zu entwickeln:

- **Standardimplementierungen für verschiedene Simulationsformen:** Für diskrete und kontinuierliche Simulationsumgebungen kann jeweils eine Standardimplementierung zur Verfügung gestellt werden. Die Simulationsumgebungen sollten hierbei mit beliebigen Modelltypen (analytisch, wissensbasiert und konnektionistisch)[4] gekoppelt werden können. Dies könnte mit Hilfe von entsprechenden Standardimplementierungen für die einzelnen Modelltypen unterstützt werden.
- **Standardimplementierungen für verschiedene Simulationswerkzeuge:** Jedes beliebige Simulationswerkzeug, das über eine Programmierschnittstelle verfügt, kann mit Hilfe einer geeigneten Wrapper-Implementierung in die Experimentierumgebung integriert werden. Der Erstellungsaufwand einer Experimentierumgebung verringert sich erheblichen, wenn für die zu integrierenden Werkzeuge bereits entsprechende Wrapper-Implementierungen verfügbar sind.

Die Qualität des Frameworks für Experimentierumgebungen kann anhand zweier Kriterien bestimmt werden:

1. **Effizienz:** Aufwand der notwendig ist, um auf Basis des Frameworks qualitativ hochwertige Experimentierumgebungen zu erstellen.
2. **Effektivität:** Qualität der Experimentierumgebungen, die mit Hilfe des Frameworks erstellt werden können.

Die Qualitätskriterien sind nur auf Basis von bereits realisierten Experimentierumgebungen messbar. Zur Zeit werden an der Universität Bamberg für die Lernumgebung *Grundkurs Wirtschaftinformatik (ILU-GKWI)*[5] verschiedene Experimentierwerkzeuge entwickelt. Realisiert wird z.B. im Rahmen von Diplomarbeiten ein System-Dynamics-Werkzeug zur Modellierung und Analyse des Verhaltens komplexer Systeme.

Die ILU-GKWI wird begleitend zur Präsenzlehre an der Universität Bamberg und an der Virtuellen Hochschule Bayern (vhb) eingesetzt. Sie ist als Selbstlernumgebung konzipiert und umfasst neben einer Experimentierumgebung sechs weitere Lernräume zum expositorischen und explorativen Lernen. Während des Einsatzes der ILU-GKWI erfolgt eine kontinuierliche Evaluation derselben. Dieser Rahmen wird die Grundlage für die Evaluation des Frameworks und der darauf basierenden Experimentierumgebungen bilden.

---

4 Die unterschiedlichen Arten von Modelltypen sind in [FeSi01, 96f] beschrieben.

5 Eine Demoversion der Lernumgebung findet sich unter der Adresse http://www.iaws.wiai.uni-bamberg.de/forschung/projekte/lernsoft/projekte/gkwi/ilugkwi.html

Literatur

[ADL04a] Dodds P. (ADL): SCORM 2004 Overview. http://www.adlnet.org/screens/shares/dsp_displayfile.cfm?fileid=992, 2004, Abruf am 18.06.2004

[ADL04b] Dodds P. (ADL): SCORM Content Aggregation Model Version 1.3. http://www.adlnet.org/screens/shares/dsp_displayfile.cfm?fileid=994, 2004, Abruf am 18.06.2004

[And00] Anderson J. R.: Kognitive Psychologie. 5. Auflage, Spektrum, Heidelberg und Berlin, 2000

[Bia95] Bialy H. Volk-von: Pädagogisches Rollenspiel – Erleben, um zu verstehen. In: Buddrus V. (Hrsg.): Humanistische Pädagogik. Klinkhardt, Bad Heilbrunn, 1995, S. 133-151

[Beer94] Beer S.: Brain of the firm. 2. Auflage, John Wiley & Sons, Chichester u.a., 1994

[BöWo00] Böhret C., Wordelmann P.: Planspiele als Methode der Fortbildung und Entscheidungshilfe: Das computergestützte Planspiel TAU. In: Herz D., Blätte A. (Hrsg.): Simulation und Planspiel in den Sozialwissenschaften. LIT, Münster, 2000, S. 208-229

[Budd92] Buddensiek W.: Entscheidungstraining im Methodenverbund – Didaktische Begründung für die Verbindung von Fallstudie und Simulationsspiel. In: Keim H. (Hrsg.): Planspiel, Rollenspiel, Fallstudie: zur Praxis und Theorie lernaktiver Methoden. Wirtschaftsverlag, Köln, 1992, S. 9-24

[Dick00] Dick E.: Multimediale Lernprogramme und telematische Lernarrangements. BW, Nürnberg, 2000

[Dörn87] Dörner D.: Problemlösen als Informationsverarbeitung. 3. Auflage, Kohlhammer, Stuttgart u.a, 1987

[Edel00] Edelmann W.: Lernpsychologie. 6. Auflage, Beltz, Weinheim, 2000

[FeSc01] Ferstl O. K., Schmitz K.: Integrierte Lernumgebungen für virtuelle Hochschulen. In: Wirtschaftsinformatik, 43 (2001) 1, S. 12-22

[FeSi01] Ferstl O.K., Sinz E. J.: Grundlagen der Wirtschaftsinformatik. 4. Auflage, Oldenbourg, München und Wien, 2001

[FeUl04] Ferstl O.K., Ullrich C.: Eine MultiView-Benutzungsoberfläche für integrierte Lernumgebungen. In: Engels G., Seehusen S.: DeLFI2004: Die 2. e-Learning Fachtagung Informatik. Köllen Druck+Verlag, Bonn, 2004, S. 103-114

[Fers+00] Ferstl O. K., Hahn K., Schmitz K., Ullrich C.: Funktionen und Architektur einer Internet-Lernumgebung für individuelles und kooperatives Lernen. In: Uellner S., Wulf V.: Vernetztes Lernen mit digitalen Medien. Physica-Verlag, Heidelberg, 2000, S. 53-68

[Fers79] Ferstl O.K.: Konstruktion und Analyse von Simulationsmodellen. Hain, Königstein/Ts, 1979

[Frey98] Frey K.: Die Projektmethode. 8. Auflage, Beltz, Weinheim und Basel, 1998

[Geu00] Geuting M.: Soziale Simulation und Planspiel in pädagogischer Perspektive. In: Herz D., Blätte A. (Hrsg.): Simulation und Planspiel in den Sozialwissenschaften: Eine Bestandsaufnahme der internationalen Diskussion. LIT, Münster, 2000, S. 15-61

[Glas97] Glasersfeld E. von: Radikaler Konstruktivismus. 1. Auflage, Suhrkamp, Frankfurt am Main, 1997

[Gud01] Gudjons H.: Handlungsorientiert Lehren und Lernen. 6. Auflage, Klinkhardt, Bad Heilbrunn, 2001

[Kais83] Kaiser F. J.: Grundlagen der Fallstudiendidaktik - Historische Entwicklung - Theoretische Grundlagen - Unterrichtliche Praxis. In: Kaiser F. J. (Hrsg.): Die Fallstudie. Klinkhardt, Bad Heilbrunn, 1983, S. 9-34

[Kerr98] Kerres M.: Multimediale und telemediale Lernumgebungen: Konzeption und Entwicklung. 2. Auflage, Oldenbourg, München u.a., 1998

[May01] Mayer T.: I-Learning statt E-Learning. Dissertation an der Friedrich-Alexander-Universität, Erlangen-Nürnberg, 2001

[Mey87] Meyer H.: Unterrichtsmethoden. Cornelsen Scriptor, Frankfurt am Main, 1987

[Rein96] Reinhold P.: Offenes Experimentieren und Physiklernen. IPN, Kiel, 1996

[Schal01] Schaller R.: Das große Rollenspielbuch. Beltz, Weinheim und Basel, 2001

[Schma82] Schmayl W.: Das Experiment im Technikunterricht. Didaktischer Dienst Franzbecker, Bad Salzdetfurth, 1982

[Schm01] Schmitz K.: Virtualisierung von wirtschaftswissenschaftlichen Lehr- und Lernsituationen: Konzeption eines Application Framework. 1. Auflage, DUV, Wiesbaden, 2001

[Spi+88] Spiro R.J., Coulson R.L., Feltovich P.J., Anderson D.K.: Cognitive flexibility theory: Advanced knowledge acquisition in ill-structured domains. In: Cognitive Science Society (Hrsg.): Tenth Annual Conference of the Cognitive Science Society. Erlbaum, Hillsdale, 1988, S. 375-383

[Ull03] Ullrich C.: Erwerb von Problemlösefähigkeit mit Hilfe von Lernumgebungen: Konzeption und Implementierung eines Frameworks. Dissertation an der Fakultät Wirtschaftsinformatik und Angewandte Informatik der Otto-Friedrich-Universität, Bamberg, 2003