Interaktive Lernobjekte in einem mobilen Einsatzumfeld

Peter Bleckmann, René Sprotte
Universität Paderborn, Deutschland
{blepe, renspr}@upb.de

Thorsten Hampel
Heinz Nixdorf Institut, Universität Paderborn, Deutschland
hampel@hni.upb.de

Abstract: Kooperations- und Interaktionaspekte spielen besonders innerhalb mobiler e-Learning Szenerien eine wichtige Rolle. Der Wunsch nach Mobilität verstärkt das Bedürfnis nach innovativen *Computer Supported Cooperative Learning (CSCL)* Anwendungen, stellt jedoch hohe technische Anforderungen an die Architekturen moderner Lernsysteme. Gängige, webbasierte Lösungen können die Anspüche an mobile Lernumgebungen nicht vollständig erfüllen. Mit dem Fokus auf interaktiven Lernobjekten, stellen wir in dieser Arbeit ein Konzept für eine Architektur vor, die sich besonders für den Einsatz in einem mobilen Lernszenario eignet.

1 Einführung

Mobilität gewinnt durch die immer weiter in den Mittelpunkt rückende kooperative bzw. arbeitsteilige Gruppenarbeit im CSCL-Umfeld mehr und mehr an Bedeutung. Mit der Zunahme an Rechenkapazitäten mobiler Geräte steigen die Möglichkeiten diese innerhalb von Lernumgebungen einsetzen zu können, wobei die technischen Anforderungen an die Lernumgebungen jedoch in erheblichem Maße mit dem Wunsch nach Mobilität steigen. Die erhöhten technischen Anforderungen resultieren aus der Notwendigkeit einen hohen Grad an Interaktion für die erfolgreiche Gruppenarbeit in einem mobilen Szenario sicherzustellen und dem Problem, dass Interaktion erschwert wird durch den Einsatz moderner Überungsformen [TN88, NTV94, VT94], unterschiedlichster mobiler Geräte und einer unter Umständen schlechten oder fehlenden Kommunikationinfrastruktur.

Dieses Problem wird besonders deutlich, wenn nicht nur die Interaktion zwischen Teilnehmern betrachtet wird, sondern die Nutzung der Lernobjekte durch einen Benutzer selbst eine gewisse Interaktion erfordert. Diese Art der Lernobjekte werden in diesem Beitrag als *interaktive Lernobjekte* bezeichnet und in Abschnitt 2 genauer definiert. Der Typ und die Eigenschaften dieser Objekte werden im jeweiligen Lernsystem definiert und sind in der Regel proprietär.

Lernsysteme sind typischerweise als webbasierte Client-Server-Systeme konzipiert. Die Benutzerinteraktion erfolgt im Allgemeinen über einen Browser. Diese Architektur erfordert jedoch eine permanente Netzwerkverbindung zwischen Lernsystem und Lernenden, um Interaktion und Kooperation zwischen den Nutzern und Lernobjekten zu ermöglichen.

Da aber eine permanente Netzanbindung zu einem fixen Punkt nicht unbedingt gewährleistet werden kann, macht dieser Umstand den Einsatz solcher Systeme in hoch mobilen Szenarien problematisch. Aus der Sicht einer mobilen Lernanwendung ist es in diesen Systemen somit auch nicht immer möglich, Kooperation zwischen Nutzern zu ermöglichen, da sämtliche Kommunikation über zentrale Server abgewickelt werden muss. Somit ist eine spontane Vernetzung zu Arbeitsgruppen und damit die Kooperation unter Nutzern nicht in jedem Fall möglich.

Hier bedarf es einer Lösung, die zum einen sicherstellt, das relevante Daten in dem System jederzeit, also auch offline, verfügbar sind, und zum anderen die Möglichkeit der spontanen Vernetzung mit anderen Nutzern ermöglicht. Klassische P2P-Architekturen bieten hierfür eine gute Ausgangsbasis, da sie per Definition bereits die spontane Vernetzung einzelner Teilnehmer erlauben. Die gängigen P2P-Architekturen wie z.B. Chord [SMK+01] oder Skipnet [HJS+03] stellen aber dabei nicht die permanente Verfügbarkeit relevanter Daten sicher. Einen Lösungansatz bieten hier verteilte Persistenzen. Beispiele für solche Entwicklungen sind Ozeanstore [KWW+00, REG+03] oder Pastry [RD01, ZBH04]. Diese Systeme garantieren, dass alle Daten innerhalb des P2P-Netzwerkes gesichert zur Verfügung stehen, solange ein Nutzer eine Verbindung zu diesen Netzwerken hat. Sie stellen jedoch keine Mechanismen für die offline Verfügbarkeit von Daten zur Verfügung.

Im Rahmen dieses Beitrags beschreiben wir ein Konzept einer P2P-Architektur auf der Basis einer angepassten verteilten Persistenzschicht und ein darauf abgestimmtes Konzept für interaktive Lernobjekte. Beide Teile ermöglichen es, Lernplattformen für hoch mobile Szenarien zu konzipieren, in denen Interaktivität und Kooperation im Vordergrund steht. Unsere Architektur stellt sicher, dass eine effiziente Verteilung der Daten innerhalb des Systems stattfindet, um zu gewährleisten, dass jedem Benutzer zu jeder Zeit die relevanten Daten des Lernsystems zur Verfügung stehen. Dabei berücksichtigt unsere Architektur den Kontext in dem der Nutzer arbeitet um die Menge und den Aufwand der zu verteilenden Daten zu optimieren.

2 Konzept für mobile Interaktive Lernobjekte

Interaktive Lernobjekte bezeichnen in diesem Beitrag Lernobjekte, die festgelegt durch das zugrundeliegende Lernsystem einen gewissen Arbeitsablauf, einen Workflow, für die Bearbeitung durch einen Lernenden definieren. Ein Beispiel dafür kann die Erzeugung, Bearbeitung und Auswertung einer komplexen Aufgabe aus der Mathematik oder dem Maschinenbau sein. Dabei muss zuerst ein Aufgabenobjekt generiert werden, welches durch den Lernenden bearbeitet bzw. beantwortet wird. Anschließend kann eine automatische Auswertung der Antwort, eine Rückmeldung an den Lernenden und ggf. eine Protokollierung im Lernsystem erfolgen. Neben derart komplexen Abläufen können auch einfachere Strukturen wie Multiple-Choice-Aufgaben, Zuordnungsaufgaben oder auch Freitext-Aufgaben abgebildet werden. Ein vorhandener Auswertungsschritt in dem Workflow muss dabei nicht zwangsweise automatisiert, sondern kann auch manuell z.B. durch einen Tutor durchgeführt werden.

In gängigen Client-Server basierten Lernsystemen wird ein solcher Workflow als Bestandteil der Server-Anwendung implementiert. Um eine Verwendbarkeit der Lernobjekte innerhalb eines P2P-Systems zu gewährleisten, ist es jedoch notwendig, neben den Daten des jeweiligen Lernobjekts auch den Workflow innerhalb der Lernobjekte und nicht innerhalb des Lernsystems zu speichern. Nur so können die benötigten Informationen auch in Ad-hoc- oder Offlineszenarien zur Verfügung stehen.

Dabei wollen wir einen Workflow durch ein Flussdiagramm, welches analog zu gängigen standardisierten Business Process Sprachen definiert wird, repräsentieren. Abbildung 1 zeigt ein Beispiel, wie ein solcher Workflow aussehen kann. In diesem Beispiel soll einen gegebene Aufgabe der Form $n^m \ mod \ p$ mittels der Methode des „Repeated Squaring“ gelöst werden, wobei bei einer korrekten Antwort Bonuspunkte vergeben werden.

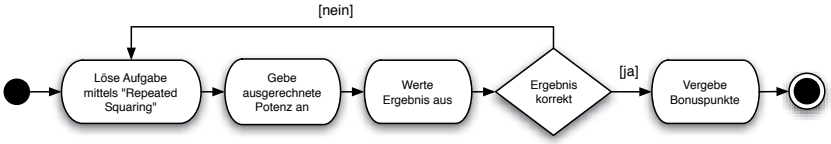
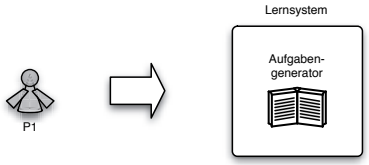


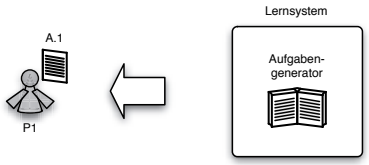
Abbildung 1: Beispiel eines Workflows

Das folgende Beispielszenario zeigt, wie ein Lernsystem unter Verwendung mobiler interaktiver Lernobjekte arbeiten könnte. In diesem Szenario legen wir den Fokus auf die Beschreibung eines möglichen Workflows, den die Objekte durchlaufen, und abstrahieren die verwendete Netzwerkarchitektur. Eine Motivation für den Einsatz interaktiver Objekte auf der Basis einer P2P-Architektur erfolgt im Anschluss.

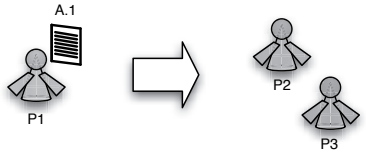
Unser Szenario besteht aus einem Lernsystem, welches Mathematikaufgaben generiert und die Lösung nach der Bearbeitung durch den Lernenden automatisch auswertet. Eine Rückmeldung an den Lernenden erfolgt nach der Auswertung der Ausgabe, die bei falscher Lösung Hilfestellungen und eine Mustertlösung enthält.



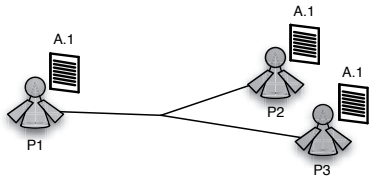
Ein Lernsystem stellt Übungsaufgaben (Lernobjekte) für eine jeweilige Vorlesung oder ein Themengebiet bereit. Lernende greifen mittels einer speziellen Anwendung auf dieses Lernsystem zu.



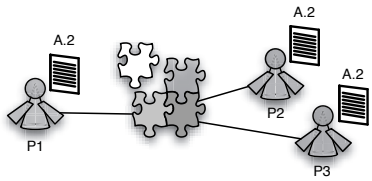
Der Lernende fordert eine bestimmte Aufgabe aus dem Lernsystem an. Dieses generiert mittels eines Aufgabengenerators für den Lernenden, oder dessen Lerngruppe eine Aufgabe. Sie kann ggf. personalisiert sein, so dass jeder Lernende bzw. jede Lerngruppe eine eigene Variante der Aufgabe erhalten.



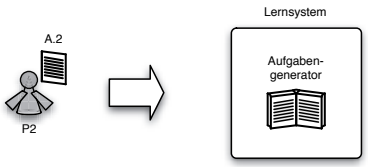
Der Lernende hat die Möglichkeit nach Übertragung der Aufgabe die Verbindung zum Lernsystem zu trennen und sich mit seiner Lerngruppe zu treffen. Die Lerngruppe ist in der Lage die Aufgabe kooperativ zu bearbeiten ohne das die Gruppenmitglieder eine Verbindung zum Lernsystem herstellen müssen.



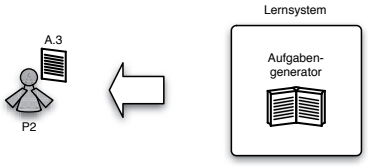
Innerhalb der Lerngruppe synchronisieren sich die relevanten Daten nun selbstständig. So haben alle Mitglieder der Lerngruppe Zugriff auf die Daten und das zu bearbeitende Lernobjekt. Die Grundlage für ein kooperatives Arbeiten ist damit geschaffen.



Die Lerngruppe hat nun die Möglichkeit kooperativ an der Aufgabe zu arbeiten. Durch die Bearbeitung ensteht eine neue Version des Objektes und das Objekt erreicht einen neuen Zustand innerhalb seines Workflows (bearbeitet).



Ausgestattet mit der bearbeiteten Aufgabe kann sich ein Mitglied der Lerngruppe wieder mit dem Lernsystem verbinden. Die neue Version des Lernobjekts wird automatisch an das Lernsystem übertragen. Es erkennt die neue Version und das es bearbeitet wurde und führt eine automatische Auswertung durch. Die Auswertung erzeugt nun wiederum eine neue Version des Lernobjektes und ändert ebenfalls den Zustand des Objektes (abgeschlossen).



Diese neue Version wird wieder an den Lernenden zurück übertragen. Sie beinhaltet Informationen über die Auswertung: War die Lösung korrekt, wieviel Punkte hat der Lernende bzw. die Lerngruppe erziehlt etc.?

Im Folgenden beschreiben wir, wie das vorgestellte Konzept der interaktiven Lernobjekte in ein mobiles Lernumfeld integriert werden kann und warum sich dafür eine P2P-Architektur besonders gut eignet.

In [HKSE03] wurde das didaktische Konzept „Jour Fixe“ vorgestellt, welches traditionelle Lehrmethoden mit Methoden der kooperativen Wissensstrukturierung und Organisation verbindet. Wesentliche Bestandteile sind zum einen ein Vorlesungsbetrieb, in welchem im Wesentlichen vorstrukturierte Inhalte präsentiert werden und zum anderen die Gruppenarbeit, in welcher kleinere Gruppen von Studenten das präsentierte Wissen bearbeiten. In regelmäßigen Abständigen finden so genannte „Jour Fixe“-Termine statt, an denen Grup-

pen ihr erarbeitetes Wissen präsentieren müssen. Im Bereich der gruppeninternen Arbeit beruht die Idee des „Jour Fixe“ in höchsten Maße auf dem Konzept der Selbstorganisation, da es der Gruppe frei überlassen sein soll, wie sie sich zwischen den „Jour Fixe“-Terminen organisiert und arbeitet.

Klassische Lernsysteme, welche üblicherweise in Form einer Client-Server-Architektur realisiert werden, bieten in unseren Augen in dem Bereich der Mobilität nur eine beschränkte Unterstützung für einen hohen Grad an Selbstorganisation. Durch den Einsatz einer lokal gebundenen Infrastruktur, worauf eine Client-Server-Architektur im Regelfall beruht, wird die örtliche und zeitliche Unabhängigkeit stark eingeschränkt. Um diese Limitierung zugunsten eines höheren Grades an Selbstorganisation aufzuheben, wollen wir die klassische Client-Server-Architektur durch eine P2P-Architektur ersetzen, welche neben spontaner Vernetzung auch Möglichkeiten zur Offlinearbeit unterstützt. Diese neuartige Architektur ermöglicht das selbstorganisierte kooperative Arbeiten in einem höchst mobilen Umfeld, ohne dabei auf lokal gebundene Infrastrukturen zurückzugreifen. Kapitel 3 beschreibt detailliert, wie wir uns eine passende Realisierung in Form einer verteilten Persistenzschicht vorstellen.

Zur Integration unserer interaktiven Lernobjekte in obige Architektur ist zu beachten, dass einzelne Lernobjekte nicht mehr nur in einem festgelegten Server existieren, sondern verteilt auf den einzelnen Peers. Bei speziellen Vorgängen im Workflow, wie z.B. die Validierung eines Lernobjektes, kann es vorkommen, dass Lernobjekte in dem P2P-System verlagert werden müssen, da nicht jeder Peer zwingend jeden Schritt im Workflow übernehmen können muss. Diese Funktionalität wird durch die eingeführten Workflow-Mechanismen ermöglicht. Analog zu dem obigen Beispiel kann ein Lernobjekt auf dem Peer eines Lernenden existieren, welches erst die Offline-Arbeit ermöglicht. Entsprechend des Workflows kann das Lernobjekt in dem Netzwerk verlagert werden und so z.B. zu einer Auswertungsinstanz gelangen.

3 Persistenz für mobile Lernszenarien

Um den Bedürfnissen eines mobilen Arbeits- und Lernumfeldes gerecht zu werden, benötigt man ein Verfahren, welches effizient den allzeitigen Zugriff auf notwendige Daten ermöglicht. Da aber nicht garantiert werden kann, dass der physikalische Zugriff jederzeit stattfinden kann, bedarf es eines optimistischen Verfahrens zum Vorhalten der relevanten Daten. Bei der heutigen Menge und Größe der Daten ist es allerdings nahezu unmöglich, ohne Struktur die Menge der relevanten Daten herauszufinden und diese auf effizientem Wege vorzuhalten. Da es für die kooperative Arbeit sinnvoll ist, die verwendeten Daten semantisch zu strukturieren, setzen wir dafür das Konzept der virtuellen Wissensräume [HKS03] (siehe Abbildung 2) ein, welches sich bereits in unseren Systemen wie z.B. *sTeam* [HKS01, HKS02] bewähren konnte. Dieses Konzept ordnet Inhalte in semantischen Gruppierungen, den virtuellen Räumen, welche von Benutzern in Form von Avataren virtuell betreten werden können. Hierdurch wird die kooperative Arbeit mit den im Raum enthaltenen Elementen ermöglicht.

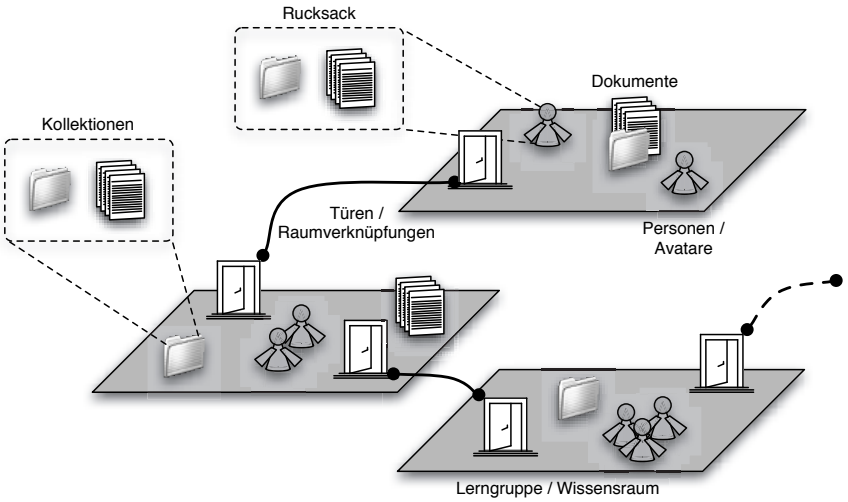


Abbildung 2: Raumkonzept für kooperatives Arbeiten

Der Einsatz dieses Wissensraumkonzeptes in Systemen wie *sTeam* hat gezeigt, dass innerhalb der einzelnen virtuellen Räume nur eine begrenzte Anzahl von Objekten platziert wird. Dieses kann dadurch begründet werden, dass der Inhalt eines Raumes ab einer gewissen Größe nicht mehr sinnvoll angezeigt und mit dem Inhalt nicht mehr vernünftig gearbeitet werden kann. Auf diese Tatsache kann in Bezug auf die Mobilitätsaspekte im Design der verteilten Persistenz zurückgegriffen werden. Da potentiell jedes Datum in einem Raum für einen teilnehmenden Benutzer wichtig sein kann, wir aber wissen, das die Menge dieser Daten begrenzt ist, können wir basierend auf einem geeigneten Synchronisationsmechanismus einem Teilnehmer den vollständigen virtuellen Wissensraum zur Verfügung stellen. Damit ist sichergestellt, dass ein Benutzer jederzeit und unabhängig von Ort oder Netzwerkverbindung mit den Daten arbeiten kann.

3.1 Verteilte Daten-Synchronisation

Änderungen an bzw. in einem virtuellen Wissensraum sollten möglichst effizient an alle betroffenen Personen, welche zu dem jeweiligen Zeitpunkt verfügbar sind, propagiert werden. Im Gegensatz zu unterschiedlichen Point-To-Point Verfahren, welche die Synchronisation in einem Schritt auf zwei Teilnehmer beschränken, möchten wir eine Synchronisation benutzen, welche auf einem Verzeichnis basiert, welches Informationen über die Versionierung der in dem jeweiligen Wissensraum enthaltenen Objekte speichert (siehe Abbildung 3). Dieses ermöglicht mehreren Teilnehmern gleichzeitig ihre lokalen Versionen zu bewerten und zu entscheiden, ob diese verteilt oder erneuert werden müssen. Eine Konfliktauflösung kann dabei nach den in [TTP+95] vorgestellten Verfahren geschehen.

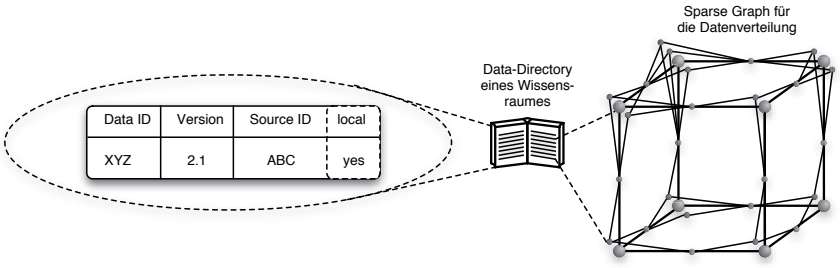


Abbildung 3: Verteilte Daten-Synchronisations

Zwischen den einzelnen Teilnehmern eines Wissensraumes soll ein so genannter Sparse Graph [EKM03] aufgebaut werden, welcher für die Verteilung von Objekten im Wissensraum genutzt wird. Durch seinen beschränkten Knotengrad, bei trotzdem kleinem Durchmesser, eignet sich diese Klasse von Graphen besonders gut für eine balancierte Verteilung von Daten. Neue Objekte sollen dabei über ein Pull-Verfahren verteilt werden, welches bedeutet, dass ein Knoten über die Verfügbarkeit eines neuen Datums informiert wird und dieses daraufhin anfordert. Neue Versionen eines Objektes werden stets von den Nachbarn geladen. Ist laut Verzeichnis eine neue Version verfügbar, aber keiner der Nachbarn verfügt darüber, so ist diese Version noch sehr neu im Wissensraum und befindet sich in der Verbreitung. Hierdurch bekommt der lokale Knoten diese neue Version automatisch.

3.2 Synchronisationsschemata

Gerade in mobilen Einsatzszenarien können sehr heterogene Netzwerke bezüglich der eingesetzten Geräte bzw. Geräteklassen entstehen. In so einem Umfeld kann man nicht mehr von jedem Gerät bzw. der jeweiligen Anwendung erwarten, dass es den vollständigen Inhalt eines Content-Bereiches synchronisieren kann. Aus diesem Grund möchten wir in unserem System die Möglichkeit bieten, dass Geräte bzw. Anwendungen mit unterschiedlichen Synchronisationsschemata an der Persistenz teilhaben können. Wir unterscheiden hierbei folgende drei Klassen von Synchronisationsschemata:

- **Synchronizing Nodes** verfügen über die nötigen Ressourcen um eine vollständige Synchronisation der Daten eines Wissensraumes durchzuführen. Aus diesen Systemen wird der Sparse Graph zur Datenverteilung gebaut
- **Selective Nodes** verfügen lediglich über einen begrenzten Speicher, welcher nur die teilweise Synchronisation der Daten eines Wissensraumes erlaubt. Daten können zum Beispiel nach der Häufigkeit des Zugriffes bewertet und die jeweils am höchsten bewerteten Daten synchronisiert werden. Ein Selective Node wird nicht in den Sparse Graphen mit aufgenommen, sondern über eine Hash-Funktion einer festen Zahl von Synchronizing Nodes zugeordnet, von welchen er Daten erhalten kann

- **On-demand Nodes** sind eine eingeschränkte Version der Selective Nodes. Sie verfügen lediglich über die Ressourcen, um neben dem Verzeichnis auf nur einem Datum aus dem jeweiligen Wissensraum zu arbeiten. Dieses wird immer bei Bedarf von den Nachbarn geladen. Die Integration in die Persistenz geschieht analog zu den Selective Nodes

Diese Struktur der Persistenz ermöglicht einen sehr flexiblen Einsatz bezüglich der Anwendungsstruktur. Sowohl klassische Client-Server-Strukturen, reine P2P-Lösungen als auch Mischformen lassen sich einfach realisieren. Für ein Client-Server-System werden der Server als Synchronizing Node und die Clients als Selective bzw. On-demand Nodes realisiert. Für eine reine P2P-Anwendung werden alle Systeme als Synchronizing Nodes realisiert.

Als Mischform beider Systeme kann man eine in der Größe variable Serverfarm mit automatischem Load-Balancing betrachten, in welcher die Server als Synchronizing Nodes und die Clients als Selective bzw. On-demand Nodes realisiert werden. Durch die Synchronisation arbeiten die Server automatisch mit dem gleichen Datenbestand und durch die Hash-Funktionen und der Pull-Verfahren bei der Datenverteilung entsteht eine gleichmäßige Auslastung der Serverfarm. Das Hinzufügen oder Entfernen von Servern resultiert automatisch in einer Neuverteilung der Clients, welches mit sehr geringem Aufwand die Dimensionierung der Serverfarm an eine sich ändernde Anzahl von Clients erlaubt.

4 Interaktive Lernobjekte innerhalb einer verteilten Persistenz

Das vorgestellte Konzept einer Persistenz-Schicht eignet sich besonders für den Entwurf mobiler Anwendungen, so auch für moderne Lernszenarien. Allerdings unterscheiden sich die interaktiven Lernobjekte, wie sie in diesem Beitrag vorgestellt werden, von den statischen Objekten, welche üblicherweise in der Persistenz gespeichert werden. Die Interaktiven Lernobjekte müssen von jedem Lernenden oder jeder Lerngruppe individuell bearbeitet werden. Wird das Interaktive Lernobjekt innerhalb der Persistenz durch ein konkretes Objekt repräsentiert, kann die individuelle Bearbeitung nicht durch die Modifikation eines Lernobjektes geschehen. Die Bearbeitung durch eine oder mehrere Personen bzw. Gruppen würde zwangsweise zu Konflikten führen.

So bedarf es eines Konzeptes, um den interaktiven, kooperativen Teil der Lernobjekte in der Persistenz abbilden zu können. Zu diesem Zweck benutzen wir die Idee eines Template Generator [ESS92], welcher einen Bauplan für eine Aufgabe definiert. Diese einzelnen Aufgaben-Templates werden von dem Lernsystem generiert und als Objekte innerhalb eines Wissensraumes abgelegt. Dazu muss das Lernsystem ein Mitglied in dem entsprechenden Wissensraum, also ein Teilnehmer an der Persistenz sein. Diese Templates werden wie jedes andere Objekt in dem Wissensraum verteilt. Jeder Benutzer (oder Gruppe), mit Zugriff auf die jeweilige Aufgabe, kann sich aus dem zugehörigen Template eine konkrete, je nach Bedarf personalisierte, Aufgabe ableiten, welche als eigenständiges Objekt in der Persistenz abgelegt wird. Zugriff auf dieses Objekt erhalten ausschließlich der jeweilige Benutzer (Gruppe) und das Lernsystem bzw. die Korrektur- und Bewertungsinstanzen.

Diese können sowohl durch ein Software-System als auch durch eine spezielle Person oder Gruppe repräsentiert werden.

Auf diesem Wege ist es in unserem anfangs vorgestellten Szenario möglich, in entsprechenden Wissensräumen Templates für Übungsaufgaben zu hinterlegen. Eine Besonderheit an an diesem System ist, dass die Generierung einer Aufgabe aus einem Template kein trivialer Vorgang sein muss und den Einsatz einer speziellen Software-Komponente erfodert erfordert, welche nicht jedem Peer zur Verfügung steht. Daher ist die Aufgabengenerierung nur mit einer Verbindung zu dem Lernsystem möglich.

Da nun jeder Lernende über seine eigene Kopie einer zu bearbeitenden Aufgabe verfügt, können diesem Objekt beliebige, personenbezogene Informationen hinzugefügt werden. Das bietet die Möglichkeit, die Aufgabe durch eine persönliche Lösung zu beantworten. Dieser Vorgang generiert eine neue Version des jeweiligen Objektes, welche wieder automatisch über den Synchronisationsmechanismus zu der Auswertungsinstanz übertragen wird. Diese erkennt die neue Version automatisch und ist so in der Lage, sie zu bewerten.

Für unser Szenario bedeutet dies, dass jede Lerngruppe eigenständig ihre Aufgabe bearbeitet und durch eine kurze Verbindung zum Lernsystem bewerten lassen kann. Auf diesem Wege ist es z.B. möglich, im Falle einer falschen Lösung problembezogene Hilfsanweisungen in dem Lernobjekt zu hinterlegen oder die Aufgabe als richtig gelöst zu markieren. In beiden Fällen würde wiederum eine neue Version des Objektes erzeugt werden, welche durch die Synchronisation automatisch zu der jeweiligen Lerngruppe zurück gelangt.

Aufgrund der Tatsache, dass Änderungen durch den Synchronisationsmechanismus automatisch zu dem Lernsystem bzw. der Auswertungsinstanz übertragen werden, ist eine Erhebung der erbrachten Leistungen sowie des Lernstandes durchführbar. In unserem konkreten Einsatz zur Unterstützung des Lehrbetriebes ist es dadurch möglich, die Leistung von Studenten zu bewerten und ein System zur Vergabe von Bonuspunkten für korrekt bearbeitete Übungen einzuführen.

Die bisher beschriebenen Eigenschaften der Persistenz im Zusammenspiel mit den interaktiven Lernobjekten ermöglichen, wie in Kapitel 3.2 beschrieben, sowohl den Einsatz angelehnt an ein klassisches Client-Server-System, als auch als P2P-Lösung. Um allerdings den gewünschten Grad an Interaktionsmöglichkeiten einer Gruppe bzgl. eines Lernobjektes zu erreichen, welches im Besonderen fordert, dass die Gruppe ohne Verbindung zu dem Lernsystem arbeiten kann, ist die Realisierung als P2P-System zwingend. Ein Lernender, repräsentiert durch einen Synchronizing Node, holt sich am Lernsystem, welches ebenfalls die Rolle eines Synchronizing Nodes übernimmt, die Aufgabe für seine Gruppe ab. Die dafür benötigte Netzwerkverbindung kann nach der Erstellung der Aufgabe und anschließender automatischer Synchronisation zu dem Lernenden getrennt werden (siehe Abbildung 4).

Dieses Vorgehen ermöglicht es dem Lernenden, sich mit seiner Lerngruppe an einem beliebigen Ort zu treffen, an den keine besonderen Anforderungen bzgl. der Kommunikationsinfrastruktur gestellt werden. Jedes Mitglied der Lerngruppe wird ebenfalls als Synchronizing Node repräsentiert. Innerhalb der Lerngruppe wird ein spontanes Netzwerk aufgebaut, welches es der Persistenz ermöglicht, die Aufgabe an alle Gruppenteilnehmer zu verteilen. Nun können die Gruppenmitglieder kooperativ die Aufgabe bearbeiten. Ist

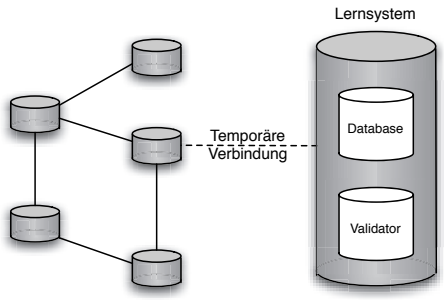


Abbildung 4: Lernsystem eingebettet in einem P2P-Umfeld

die Bearbeitung abgeschlossen, kann ein Mitglied die gelösten Aufgaben zum Lernsystem zurückbringen, indem es eine Verbindung zu dem Netzwerk aufbaut, in welchem das Lernsystem läuft. Wie bereits erwähnt, wird durch den Synchronisationsmechanismus eine automatische Bewertung durch das Lernsystem ermöglicht.

5 Zusammenfassung und Ausblick

In diesem Beitrag haben wir ein Konzept für eine Architektur eines Lernsystemes vorgestellt, welches sich besonders für den Einsatz in einem mobilen Lernszenario eignet. Eine P2P-basierte verteile Persistenz stellt in dieser Archtiktur sicher, dass die Lernenden jederzeit Zugriff auf relevante Daten des Lernsystems haben. Zusätzlich wurde der Fokus auf die speziellen Eigenschaften interaktiver Lernobjekte gelenkt und diese beim Entwurf der Architektur berücksichtigt. Hierbei wurden die Konzepte zur Datenverteilung und für interaktive Lernobjekte in der Metapher des virtuellen Wissensraumes verschmolzen. Basierend auf dieser Architektur lässt sich somit eine neue Generation von kooperativen und interaktiven Lernsystemen für mobile Einsatzumfelder realisieren.

In unserer zukünftigen Arbeit wollen wir untersuchen, ob sich auf Basis der vorgestellten Architektur ein generisches Modell für beliebige Arten von interaktiven Objekten realisieren läßt. Der Fokus der Untersuchungen soll auf Objekten liegen die komplexe mehrstufige, ggf. asynchrone Abläufe definieren und wie sich diese Abläufe in einem mobilen Einsatzumfeld auswirken. Hier sollen besonders Fragen der Repräsentation der Arbeitsabläufe in der Anwendung im Hinblick auf die Optimierung der kooperrativen Arbeit im Mittelpunkt stehen.

Literatur

[EKM03] R. Elsässer, R. Královiˆ und B. Monien. Sparse topologies with small spectrum size. *Theor. Comput. Sci.*, 307(3):549–565, 2003.

[ESS92] H.-D. Ehrich, G. Saake und A. Sernadas. Concepts of Object-Orientation. In R. Studer, Hrsg., *Informationssysteme und Künstliche Intelligenz: Modellierung, Proc. 2. Workshop, Ulm, Germany*, Jgg. 303, Seiten 1–19, Berlin, 1992. Springer-Verlag.

[HJS+03] N. HARVEY, M. JONES, S. SAROIU, M. THEIMER und A. WOLMAN. Skipnet: A scalable overlay network with practical locality properties, 2003.

[HKS01] Thorsten Hampel und Reinhard Keil-Slawik. sTeam - Designing an integrative infrastructure for Web-based computer-supported cooperative learning. In *Proceedings of the 10th international conference on World Wide Web (WWW10)*, Seiten 76–85, 2001.

[HKS02] Thorsten Hampel und Rheinhard Keil-Slawik. sTeam: Structuring Information in a Team - Distributed Knowledge Management in Cooperative Learning Environments. *ACM Journal of Educational Resources in Computing*, 1(2):1–27, 2002.

[HKS03] Thorsten Hampel und Reinhard Keil-Slawik. Experience With Teaching and Learning in Cooperative Knowledge Areas. In *Proceedings of The Twelfth International World Wide Web Conference. (Hrsg. Hencsey, G., White, B.) Budapest (Ungarn)*, Seiten 76–85, May 2003.

[HKSE03] Thorsten Hampel, Reinhard Keil-Slawik und Bernd Eßmann. Jour Fixe - We Are Structuring Knowledge Collaborative - Structuring of Semantic Spaces as a Didactic Concept and New Form of Cooperative Knowledge Organization. In Allison Rossett, Hrsg., *E-Learn 2003*, Proceedings of E-Learn 2003, Seiten 225–232. AACE, AACE Press, 2003.

[KWW+00] John Kubiatowicz, Westley Weimer, Chris Wells, Ben Zhao, David Bindel, Yan Chen, Steven Czerwinski, Patrick Eaton, Dennis Geels, Ramakrishan Gummadi, Sean Rhea und Hakim Weatherspoon. OceanStore:An Architecture for Global-Scale Persistent Storage. *ACM SIGPLAN Notices*, 35(11):190–201, 2000.

[NTV94] A. Nevin, J. Thousand und R Villa. Introduction to creative cooperative group lesson plans. In *J. Thousand, R. Villa, and A. Nevin (Eds.), Creativity and collaborative learning: A practical guide to empowering students and teachers*, Seiten 131–225. Baltimore: Paul H. Brookes Publishing, 1994.

[RD01] Antony Rowstron und Peter Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, Seiten 329–350, November 2001.

[REG+03] S. Rhea, P. Eaton, D. Geels, H. Weatherspoon, B. Zhao und J. Kubiatowicz. Pond: the OceanStore Prototype. In *Proceedings of the Second USENIX Conference on File and Storage Technologies (FAST '03)*, San Francisco, California, 2003.

[SMK+01] Ion Stoica, Robert Morris, David Karger, Frans Kaashoek und Hari Balakrishnan. Chord: A Scalable Peer-To-Peer Lookup Service for Internet Applications. In *Technical Report TR-819, MIT, March 2001*, Seiten 149–160, 2001.

[TN88] J. Thousand und A. Nevin. Cooperative learning and special education. In *T. Husen and T.N. Postlethwaite (Eds.), International Encyclopedia of Education*, Seiten 273–286. Oxford, England: Pergamen Press, 1988.

[TTP+95] D. B. Terry, M. M. Theimer, Karin Petersen, A. J. Demers, M. J. Spreitzer und C. H. Hauser. Managing update conflicts in Bayou, a weakly connected replicated storage system. In *Proceedings of the 15th ACM Symposium on Operating Systems Principles*, Seiten 172–182, Copper Mountain, Colorado, United States, 1995. ACM Press, New York, NY, USA.

[VT94] R. Villa und J. Thousand. One divided by two: Redefining the role of a cooperative learning teaching team. In *J. Thousand, R. Villa, and A. Nevin (Eds.), Creativity and collaborative learning: A practical guide to empowering students and teachers*, Seiten 79–101. Baltimore: Paul H. Brookes Publishing, 1994.

[ZBH04] Rongmei Zhang, Ali R. Butt und Y. Charlie Hu. Topology-Aware Peer-to-Peer On-Demand Streaming, Oktober 2004.