Echtzeitmetamodellierung im Web-Browser

Michael Derntl, Stephan Erdtmann, Petru Nicolaescu, Ralf Klamma, Matthias Jarke

Lehrstuhl Informatik 5 – Informationssysteme und Datenbanken
RWTH Aachen
Ahornstr. 55
52056 Aachen, Deutschland
{derntl, erdtmann, nicolaescu, klamma, jarke}@dbis.rwth-aachen.de

Abstract: Modellierung ist ein integraler Bestandteil von Schaffensprozessen in vielen Disziplinen. Der Modellierungsprozess wird durch vielfältige Tools unterstützt, von denen jedoch die wenigsten eine gemeinsame Modellierung durch mehrere Modellierer ermöglichen und die mittels offener Technologien und Protokolle realisiert sind. Um diese Lücke zu schließen, konzipieren wir in diesem Beitrag ein Framework für Echtzeitmetamodellierung, das als Widget-basierte Anwendung realisiert wird und ausschließlich auf quelloffenen Programmbibliotheken und breit implementierten Web-Technologien basiert. Der Beitrag berichtet über eine vorab durchgeführte Technologiestudie, bei der ein Echtzeitmodellierungstool für eine bestimmte Anwendung realisiert und erfolgreich evaluiert wurde. Das Metamodellierungsframework wurde durch Abstrahierung und Erweiterung der Technologiestudie auf Metamodellebene konzipiert und soll die Verbreitung von Echtzeitkollaborationsfunktionen in Web-Anwendungen vorantreiben.

1 Einleitung

Modelle werden bedeutsam durch soziale Prozesse, bei denen die Perspektiven der Beteiligten einen entscheidenden Einfluss haben [FK98]. Es ist weitgehend bekannt, dass der Erfolg von Informatikprojekten von Analyse- und Entwurfsprozessen abhängig ist, bei denen die Perspektiven möglichst aller Stakeholder berücksichtigt werden sollen. Modellierungstools unterstützen diese sozialen Schaffensprozesse. Die meisten existierenden Modellierungstools unterstützen asynchrone Kollaboration, d.h. die Modellierer schicken sich ihre Entwürfe und Änderungen zu, um sie zu überarbeiten und zu integrieren. Um die mit asynchroner Überarbeitung verbundenen Einschränkungen zurückzudrängen, setzen sich immer mehr Tools durch, die gleichzeitiges gemeinsames Modellieren unterstützen. Diese Idee der gemeinsamen Bearbeitung eines Dokuments mittels Computerunterstützung ist schon seit der „Mutter aller Demos“ [Le94] von Douglas Engelbart vor fast einem halben Jahrhundert bekannt. Die technischen Voraussetzungen, um diese Idee auf breiter Basis mit Informations- und Kommunikationstechnologie umsetzen zu können, wurden jedoch erst seit etwa Mitte der 1990er Jahre mit der Verbreitung des World Wide Web und der Breitbandtechnologie geschaffen. Heute kann man zum Beispiel mit Google Docs, Office365 oder draw.io mit fast beliebig vielen anderen

Benutzern unabhängig von deren physischem Ort gemeinsam und gleichzeitig Texte editieren, Zeichnungen oder Modelle erstellen. Solche technischen Fortschritte bringen viele neue Möglichkeiten zur Kollaboration über das Web, besonders in Disziplinen, in denen enge Zusammenarbeit ein Schlüssel zum Erfolg ist.

Wie eingangs erwähnt, spielt Modellierung für die Informatik eine fundamentale Rolle und gewinnt vor allem als kollaborative Tätigkeit an Bedeutung—siehe z.B. [RKV08]. Aber auch in anderen Bereichen, wie etwa in der Geschäftsprozessmodellierung, ist Kollaboration mittels Computertools unumgängliche Praxis [Ri12]. Es gibt Modellierungstools, die kollaborative Modellierung in Echtzeit ermöglichen, sowohl am Desktop als auch im Web. Diese Tools reichen von sehr anwendungsspezifischen (z.B. das Kanban Tool[1] für Projektmanagement im Team) bis zu völlig methodenfreien Exemplaren (z.B. draw.io, mit dem Symbole aus unterschiedlichen Modellierungssprachen ohne jegliche Einschränkungen kombiniert werden können). Es gibt jedoch, wie der Vergleich existierender Ansätze später zeigen wird, keine Browser-basierten Tools, die mit einem Metamodellierungsansatz die Schaffung und Verwendung beliebiger Modellierungsnotationen mittels Echtzeitkollaboration unterstützen. Ein Framework für solche Tools vorzustellen ist das funktionale Hauptziel dieses Beitrags.

Es existieren Programmbibliotheken von unterschiedlichen Anbietern, mit denen Anwendungsentwickler relativ mühelos Benutzerschnittstellenelemente in ihre Web-Anwendungen einbauen können, die es mehreren Benutzern erlauben, gleichzeitig an einem gemeinsamen Dokument zu arbeiten. Der Begriff „Dokument“ als gemeinsam erstelltes digitales Produkt ist hier im weitesten Sinne gemeint; es kann sich dabei um ein einfaches Textdokument, jedoch auch um ein virtuelles 3D-Modell eines Gebäudes handeln. Die bekannteste dieser Programmierbibliotheken ist das Google Drive Realtime API[2], welches Mitte 2013 veröffentlicht wurde. Während dies sicherlich ein Anschub für die Entwicklung von mehrbenutzerfähigen Web-Anwendungen war, bringen solche Angebote oft das Problem mit sich, dass sie vom Anbieter als Black Box dargeboten werden. Entwickler, die solche Bibliotheken verwenden, übertragen damit alle funktionale—und gegebenenfalls auch inhaltliche—Kontrolle über das Funktionieren ihrer Anwendung an den Anbieter, in diesem Beispiel an Google. Der Anbieter kann jederzeit die Programmierschnittstellen verändern, oder noch problematischer, plötzlich ersatzlos das Angebot einstellen, wie die kürzlich erfolgte ersatzlose Entfernung des Google Reader gezeigt hat. Eine nachhaltige Lösung für Probleme dieser Art besteht darin, quelloffene Programmierbibliotheken mit der Entwicklergemeinschaft zu teilen und die Infrastruktur auf ein Fundament zu stellen, das sich offener Technologien und Standards bedient. Dies ist der nichtfunktionale Eckpfeiler des Frameworks in diesem Beitrag.

Dieser Beitrag ist wie folgt strukturiert. Im zweiten Abschnitt werden Begrifflichkeiten und Konzepte der Metamodellierung dargelegt und die Charakteristika von Echtzeitkollaborationssystemen vorgestellt. Basierend auf diesen Eckpfeilern stellen wir im dritten Abschnitt eine Technologiestudie eines Echtzeitkollaborationssystems für einen konkreten Anwendungskontext vor, um auf Basis der technologischen und benutzerorientierten Implikationen dieser Technologiestudie das System vom konkreten Anwendungskontext

[1] http://kanbantool.com/
[2] https://developers.google.com/drive/realtime/

im vierten Abschnitt zu abstrahieren zu einem Metamodellierungsframework. Diese Abstrahierung ist der konzeptionelle Kern dieses Beitrags. Der fünfte Abschnitt grenzt das vorgestellte Framework von existierenden Ansätzen ab und der letzte Abschnitt fasst den Beitrag zusammen und gibt einen kurzen Ausblick auf zukünftige Forschung.

2 Theoretischer Hintergrund

2.1 Konzeptionelle Metamodellierung

Ein konzeptionelles Modell besteht aus Objekten und Beziehungen zwischen diesen Objekten [Ol07], welche den für den Modellierer relevanten System- bzw. Realitätsausschnitt repräsentieren. Für die Erstellung eines Modells wird vom Modellierer die dafür bestimmte Notation einer Modellierungssprache benutzt. Diese wiederum wird durch das Metamodell definiert. Das „Diamantmodell" der Metamodellierung [JKL09] schlägt vor, bei Metamodellierungstechniken die Aspekte Notation (welche Notationen werden benutzt um bestimmte System- und Umgebungsaspekte zu repräsentieren?), Ontologie (welchen Ontologien unterliegen der Systemdomäne?) und Prozess (welche Prozesse bestimmen die Ableitung, Bewertung und Validierung von Modellen) zu unterscheiden, sowie den Zielaspekt, der die Entscheidungen in den ersten drei Aspekten bestimmt. Der Schwerpunkt des vorliegenden Beitrags liegt nach dieser Klassifikation auf der notationsorientierten Metamodellierung, da wir Notationssysteme definieren wollen.

Jede Modellierungssprache hat eine (unter Umständen implizite) Syntaxdefinition, welche die Elemente und ihre Notation beschreibt, sowie eine Semantikdefinition, welche die Bedeutung dieser Elemente beschreibt. Die Syntaxdefinition unterscheidet einen abstrakten und einen konkreten Teil. Der abstrakte Teil definiert die Elemente der Sprache und deren Eigenschaften. Regeln zur Notation und Verwendung bzw. Zusammensetzung dieser Elemente zu Modellen (die Modellierungsmethode) sind im konkreten Teil definiert. Das Metamodell repräsentiert die abstrakte Syntax der Modellierungssprache, d.h. ein Element in einem Modell repräsentiert eine Instanz eines Elements im Metamodell. Da ein Metamodell ebenso ein Modell ist, kann es seinerseits durch ein Meta-Metamodell definiert werden. Durch Fortführung dieses Gedankenexperiments können beliebige Metahierarchien von Modellen erzeugt werden. Ein Beispiel solcher Mehrebenen-Metamodellierungsarchitekturen ist die Meta Object Facility (MOF). Der Metamodellierungsansatz bietet auch die Basis für die Entwicklung von Graph-basierten Bearbeitungstools zur Erzeugung von Diagrammen.

Beim Modellerstellungsprozess unterscheidet man aus Benutzersicht zwei unterschiedliche Bearbeitungsansätze, nämlich freihändiges und strukturiertes Bearbeiten [Mi07]. Freihändiges Bearbeiten bietet dem Modellierer die Möglichkeit, alle verfügbaren Modellelemente beliebig zu kombinieren, beispielsweise in der UML ein Klassensymbol mittels einer Kompositionsbeziehung mit einem Aktivitätssymbol zu verbinden. Dies ermöglicht natürlich die Erstellung syntaktisch inkorrekter Modelle. Demgegenüber erlaubt strukturiertes Bearbeiten dem Modellierer ausschließlich, ein korrektes Modell in ein anderes korrektes Modell zu überführen durch eine Modellierungsoperation.

2.2 Echtzeitkollaborationssysteme

Ein Echtzeitkollaborationssystem ist eine Computeranwendung, die es einer Gruppe von Benutzern ermöglicht, gleichzeitig an einem gemeinsamen Dokument zu arbeiten [EG89, Gr94]. Wir werden in diesem Beitrag den Fokus auf jene Echtzeitkollaborationssysteme legen, die örtlich verteilte Kollaboration ermöglichen und Internetprotokolle als Kommunikationsmedium verwenden. In einem Echtzeitkollaborationssystem wird jede Operation eines Benutzers (z.B. das Einfügen einer Klasse in ein Klassendiagramm) zu allen anderen Benutzern propagiert, sodass dort der Eindruck entsteht, in Echtzeit gemeinsam an einem Modell zu arbeiten. Ein solches System benötigt neben Mechanismen zur Propagierung von Operationen auch solche zur Wahrung der Konsistenz des Modells. Um den Eindruck der Echtzeitkollaboration entstehen zu lassen, sollte ein solches System nur geringe Latenz zwischen Ausführung einer Operation und der Darstellung des Ergebnisses dieser Operation bei allen Benutzern aufweisen. Es wird daher üblicherweise jede Operation zuerst unmittelbar lokal ausgeführt und danach bzw. nebenläufig an die Kollaborateure versendet, etwa mit einer Broadcastnachricht.

Herausforderungen. Die Umsetzung solcher Systeme trifft auf erhebliche Herausforderungen bei der Sicherstellung der Konsistenz der verteilten Modellkopien. Inkonsistenz kann nach [SE98] entweder durch Divergenz oder durch Kausalitätsverletzung entstehen: Ausgehend von der Annahme, dass Operationen bei allen Kollaborateuren in Empfangsreihenfolge ausgeführt werden, tritt Divergenz dann auf, wenn die Reihenfolge nichtkommutativer Operationen an zwei empfangenden Stellen unterschiedlich ist, z.B. wenn während des Propagierens der Operation über das Netzwerk die Reihenfolge vertauscht wird. Die finalen Modellkopien der Kollaborateure sind dann nicht identisch. Kausalitätsverletzung tritt auf, wenn Operationen bei den Kollaborateuren in einer nichtkausalen Reihenfolge ausgeführt werden. Ein Kollaborateur kann hierbei unter Umständen den Effekt einer Operation sehen, bevor er die Ursache dafür gesehen hat.

Konfliktbehandlung. Es gibt zwei grundlegende Kategorien von Möglichkeiten, diese Probleme zu lösen, nämlich Konfliktvermeidung [XZS00] und Konfliktresolution [Su98]. Konfliktvermeidung verhindert, dass Konflikte auftreten, etwa indem mehrere Benutzer nicht gleichzeitig die gleiche Stelle im Dokument bearbeiten dürfen. Dies bietet Vorteile bezüglich Rechen- und Kommunikationsintensität, untergräbt jedoch die Grundidee der Echtzeitkollaboration. Demgegenüber verhindern Ansätze zur Konfliktresolution nicht, dass Konflikte auftreten, sondern sie lösen auftretende bzw. bereits aufgetretene Konflikte durch geeignete Mechanismen auf. Eine der wichtigsten Konfliktresolutionstechniken, die heute in Verwendung sind—z.B. in Google Docs und Apache Wave—ist Operational Transformation (OT) [EG98, Su98]. OT ist eine Technik, welche Operationsparameter anpasst, um Konflikte aufzulösen, z.B. die Verschiebung des Positionsparameters einer Einfügeoperation bei einer „gleichzeitig“ auftretenden Löschoperation. Um dies zu sicherzustellen, bedient sich ein OT-System aus zwei Funktionsgruppen, nämlich aus den Kontrollalgorithmen, die anhand einer gegebenen Operationsmenge entscheiden, welche Operationen eine Transformation benötigen, und den Transformationsfunktionen, welche diese Transformationen anschließend durchführen. Der Großteil der Forschung auf diesem Gebiet widmet sich Konflikten bei gleichzeitiger Textbearbeitung. Texte, die aus einer linearen Sequenz von Zeichen bestehen, sind ein-

fach zu adressieren. Sie sind logisch jedoch anders aufgebaut als Modelle, die multidimensionale logische Abhängigkeiten beinhalten können. Lösungen dafür existieren jedoch, z.B. in [Fa11] oder [SC02].

Architekturvarianten. Echtzeitkollaborationssysteme werden entweder mittels einer zentralisierten oder eine replizierten (dezentralen) Architektur realisiert. Die zentralisierte Architektur [SC02] ist eine Client/Server-Architektur, in der alle Clients ihre Operationen an einen Server senden, der für die Dokumentstatusverwaltung und Konfliktresolution zuständig ist. Im Gegensatz dazu folgt die replizierte Architektur den Prinzipien eines Peer-to-Peer-Netzwerks, in dem jeder Knoten eine Kopie des gemeinsamen Modells hält, die lokalen Operationen mittels Broadcast verteilt und eigenständig für die Konfliktresolution zuständig ist. Beide Architekturvarianten haben Vor- und Nachteile, wobei die replizierte Variante den entscheidenden Vorteil hat, dass es keinen „Flaschenhals“ in der Kommunikation gibt.

3 Technologiestudie: SyncLD

Gesamtziel dieses Beitrags ist es, ein Framework für Echtzeitkollaboration an Metamodellen sowie Modellen zu entwerfen. Wie später im Abschnitt 4 beschrieben, können die Modellierungstools dabei aus dem Metamodellierungstool heraus instanziiert werden. Die Basis für das Modellierungstool als auch für das Metamodellierungstool ist ein Diagrammeditor, der die Funktionen eines Echtzeitkollaborationssystems implementiert. Einen solchen Echtzeitdiagrammeditor haben wir als Technologiestudie für einen konkreten Anwendungsfall implementiert um zu ermitteln ob so eine Anwendung für Benutzer sinnvoll und auf Basis existierender offener Technologien machbar ist. Auf Basis der technischen und benutzerbezogenen Erkenntnisse dieser Technologiestudie wurden die funktionalen Anforderungen für das Metamodellierungsframework definiert, das im folgenden Abschnitt 4 beschrieben ist.

3.1 Anwendungskontext

In vielen Disziplinen ist die Verwendung von Kollaborationstools üblich, speziell in Forschung und Entwicklung oder in der Wirtschaft. Es gibt jedoch auch viele Communities, die „immun“ gegen solche Tools zu sein scheinen. Eine solche Community ist die der Lerndesigner. Lerndesigner gestalten Lehr-/Lernprozesse, und obwohl professionelles Lerndesign zumeist ein kollaborativer Prozess ist, haben sich unter den Lerndesignern entsprechende Tools nicht durchgesetzt [DSO11]. Dies gilt speziell für Europa, wo das in den USA stark professionalisierte *instructional design* als Disziplin nicht breit Fuß gefasst hat. Aufgrund langjähriger öffentlicher Förderung ist in Europa jedoch die Forschung im Bereich Lerndesign sehr stark, speziell im Bereich der formalen Beschreibung von Lehr-/Lernarrangements. Die einzige verfügbare umfangreiche formale Spezifikation für Lerndesign ist IMS Learning Design (IMS LD) [IMS03]. Diese Spezifikation erlaubt es Lerndesignern, ihre Lerndesigns als sogenannte Lerneinheiten formal und maschinenlesbar mittels eines XML-Dialektes zu beschreiben. Eine Lerneinheit kann dabei eine ganze Lehrveranstaltung, ein Kurzseminar, oder aber auch eine Lernepisode

von wenigen Minuten sein. Die wichtigsten Konzepte des Metamodells von IMS LD sind in Abbildung 1 dargestellt. Das Metamodell wurde in Anlehnung an eine Theatermetapher mit Konzepten wie Akt (*act*), Rolle (*role*) und Auftritt (*role-part*) versehen.

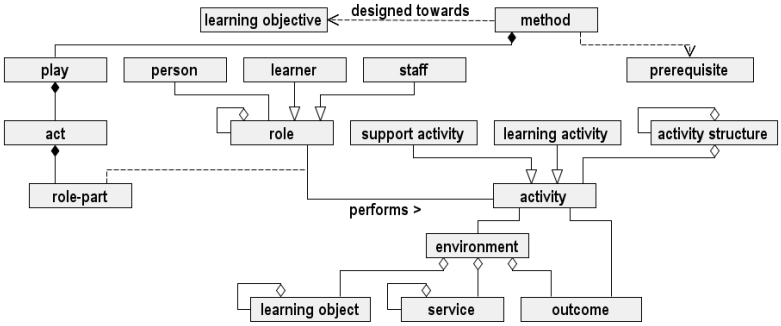

Abbildung 1: Kern des IMS Learning Design Metamodells.

Natürlich schreiben Lerndesigner die XML-Datei, die ihre Lerneinheit beschreibt, nicht per Hand, sondern sie verwenden spezielle Autorentools. Die damit erzeugten Lerneinheiten können dann in beliebigen IMS-LD-kompatiblen Laufzeitumgebungen importiert und ausgeführt werden. Die meisten Autorentools sind Desktopprogramme, es gibt wenige Web-Anwendungen dafür. Obwohl es Autorentools gibt, die Import und Export von Lerndesigns von bzw. zu diversen Onlinerepositorien ermöglichen, gab es bisher kein Autorentool, das Kollaboration in angenäherter Echtzeit unterstützt. Um diese Lücke zu schließen, haben wir ein Autorentool namens SyncLD (steht für „Synchrones Lerndesign“) entwickelt, das Kollaboration mehrerer Lerndesigner an Lerneinheiten im Web-Browser ermöglicht. Die Implementierung erfolgte dabei derart, dass spätere Abstraktion von der konkreten Anwendung IMS LD auf beliebige Metamodelle möglich ist (siehe Abschnitt 4). Die vollständigen technischen Details von SyncLD sind in [NDK13] beschrieben. In den folgenden Unterabschnitten werden wir kurz die grundlegen Konzepte und Ergebnisse in einem Detailierungsgrad beschreiben, der es ermöglicht, die darauffolgende Abstraktion auf die Metamodellierungsebene zu verstehen.

3.2 Systembeschreibung

SyncLD wurde als Widget-basierte Anwendung konzipiert. Ein Widget ist eine Anwendung, die einen wohldefinierten und typischerweise limitierten Funktionsumfang bietet, und die mit anderen Widgets zu einer komplexeren Anwendung zusammengefügt werden kann [Go11]. In SyncLD wurde dieser Benutzerschnittstellenansatz verwendet, da er einerseits plattformunabhängig in allen gängigen Web-Browsern lauffähig ist, und neben dem SyncLD-Widget weitere Widgets, welche die Kollaboration unterstützen, eingebunden werden können (z.B. Chat- oder Videokonferenz-Widgets).

Das IMS LD Metamodell wurde aktivitätsbasiert in SyncLD umgesetzt, d.h. den Kern des Modells einer Lerneinheit bildet eine Folge von Aktivitäten, die visuell ähnlich einem UML-Aktivitätsdiagramm dargestellt werden, und die dann mittels unterschiedli-

cher Registerkarten mit den anderen Elementen der Lerneinheit verknüpft werden können. Ein Beispiel dafür in SyncLD ist in Abbildung 2 zu sehen. Benutzer 1 (überlagertes Browserfenster oben) modelliert die Aktivitäten, während der Benutzer 2 (unteres Browserfenster) gleichzeitig die Eigenschaften einer oder mehrerer dieser Aktivitäten ausfüllt. Durch diese Aufgabentrennung bei Echtzeitkollaboration können Modellierungsschritte, die sonst sequenziell durchgeführt werden müssen, parallel von mehreren Benutzern gleichzeitig erledigt werden, was den Modellierungsprozess effizienter und unter Umständen durch zusätzliche Kommunikation auch effektiver macht.

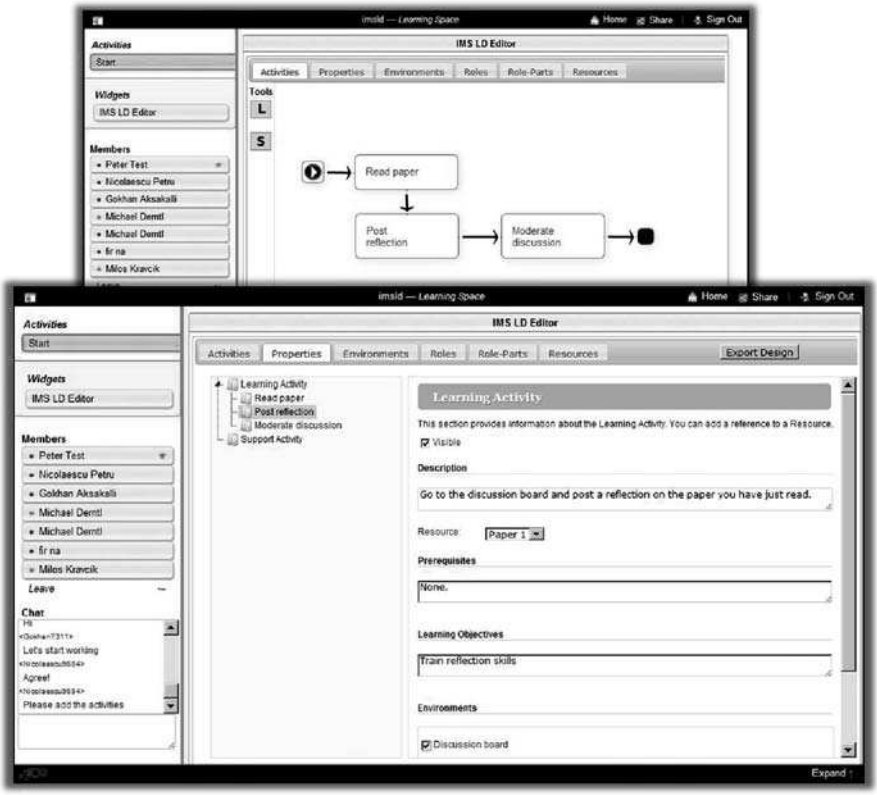

Abbildung 2: Browserfenster zweier kollaborierender Benutzer

Das Fundament der technischen Infrastruktur von SyncLD bildet das quelloffene ROLE SDK[3], eine Programmsammlung für Widget-basierte Web-Anwendungen. Das ROLE SDK wurde deshalb ausgewählt, weil es eine Menge von Konzepten und Technologien bietet, die für SyncLD vorteilhaft sind, vor allem das Konzept der Widget-Spaces. Eine Instanz eines Widget-Space bietet einen Container zum Darstellen von mehreren Widgets und ergänzt diesen um Schnittstellen für Kommunikation zwischen Widgets, Benutzerverwaltung, –authentifizierung und –autorisierung, sowie Datenverwaltung. Eine Anwendung, in der Funktionalität auf Widgets aufgeteilt ist, und in der mehrere Benut-

[3] http://sourceforge.net/projects/role-project/

zer über unterschiedliche Instanzen dieser Widgets zusammenarbeiten, erfordert Kommunikation zwischen den Widgets—die sogenannte Widget-zu-Widget-Kommunikation [Go11]. Diese umfasst einerseits lokale browserinterne Kommunikation zwischen den verschiedenen Widgets der Benutzeroberfläche, andererseits die globale browserübergreifende Kommunikation zwischen Widgets unterschiedlicher Benutzer. Ein lokales Widget agiert dabei als Gateway für den globalen Datenaustausch, indem eingehende Nachrichten über lokale Widget-zu-Widget-Kommunikation an die entsprechenden Widgets weitergeleitet werden. Das ROLE SDK beinhaltet eine Implementierung dieser Widget-zu-Widget-Kommunikationsmechanismen auf Basis von HTML5 Web Messaging [W3C11] und des Extensible Messaging and Presence Protocol (XMPP) [Sa11]. Es ermöglicht Benutzerauthentifizierung und –autorisierung mittels OpenID und OAuth.

Für SyncLD wurde das quelloffene OpenCoWeb Operational Transformation (OT) Engine API[4] mit dem ROLE SDK integriert, um dezentral Konfliktresolution zu betreiben. Wenn ein Benutzer einer Modellierungssitzung beitritt, wird jeweils der Status der gemeinsamen Sitzung repliziert. Die Clients propagieren mittels Broadcast deren Modellierungsoperationen an alle anderen Benutzer und die Konfliktresolution mittels OT wird auf Empfängerseite auf dem dortigen Sitzungsreplikat durchgeführt. Eine Modellierungsoperation ist hier jede Änderung des Modells, sei es z.B. durch Einfügen einer Aktivität, durch Auswahl einer Option in einer Dropdown-Box, oder durch Tippen bzw. Löschen eines Buchstabens in einem Textfeld.

3.3 Evaluierung

SyncLD wurde mit Partnern des Projects METIS[5], das integrierte Lerndesignumgebungen entwickelt und im EU-Programm für lebenslanges Lernen mitfinanziert wird, evaluiert. In fünf Sitzungen mit jeweils vier zusammenarbeitenden IMS LD Autoren, wurden—teils durch eine nebenläufige Audiokonferenz unterstützt—die Funktionen des Tools nach einem gegebenen Evaluierungsprotokoll getestet. Die detaillierten Evaluierungsergebnisse sind in [NDK13] zusammengefasst. Hier wollen wir jene Ergebnisse hervorheben, die für das Metamodellierungsframework in Abschnitt 4 relevant sind.

Die Evaluierung hatte eine technische und eine benutzerorientierte Komponente. Die technische Systemevaluierung wurde durchgeführt, um zu prüfen, ob die Implementierung der Echtzeitfunktionen und vor allem der Konfliktlösung durch OT funktionieren. Das Ergebnis war, dass alle replizierten Kopien der Modelle kongruent waren. Die benutzerorientierte Evaluierung zeigte einerseits, dass die Teilnehmer die Web-basierte Autorenumgebung gegenüber einer Desktopanwendung bevorzugen und dass die Echtzeitkollaboration als sehr nützliche Funktion empfunden wurde. Die Evaluierung zeigte aber auch durch Mehrfachnennung in den offenen Kommentaren der Teilnehmer, dass die Kollaboration mit SyncLD zwei wesentliche Probleme hat. Erstens war den Modellierern nicht immer klar, woran die anderen Modellierer im Moment arbeiten, was auch durch einen zusätzlichen Kommunikationskanal wie einer Audiokonferenz nicht vollständig kompensiert werden konnte. Zweitens war den Benutzern nicht klar, wie Sie den

[4] https://github.com/opencoweb/coweb-jsoe
[5] http://www.metis-project.org/

Autorenprozess mittels der Benutzerschnittstelle von SyncLD sinnvoll in einzelne Bearbeitungsschritte auftrennen sollten. In den Modellierungssitzungen hat ein Benutzer jeweils die Moderatorenrolle übernommen um die anderen Benutzer anzuleiten und die Sitzung zu steuern. Die Summe aus positiven und kritischen Rückmeldungen während dieses Evaluierungsprozesses war die Ausgangsmotivation für die Konzeption des im nächsten Abschnitt vorgestellten Metamodellierungsframeworks. Die positiven Aspekte sollten bei der Abstrahierung von der konkreten Anwendung (IMS LD Modelle) beibehalten werden, um zugleich die kritisierten Aspekte durch geeignete Mechanismen im Framework auszubessern.

4 Framework für Echtzeitmetamodellierung

Kern des Frameworks in diesem Abschnitt ist ein Modellierungstool, das es ermöglicht, sich selbst mittels eines Metamodellierungsansatzes zu instanziieren um Modelle der definierten Sprachen gemeinsam in Echtzeit bearbeiten zu können. In den Unterabschnitten behandeln wir Anforderungen, Architektur und Implementierung des Frameworks.

4.1 Anforderungen

Die Anforderungen betreffen die zwei Hauptkomponenten des hier vorgestellten Frameworks. Einerseits das Kollaborationstool, das kollaborative Modellierung nahe Echtzeit ermöglichen soll, und den Kollaborationstoolgenerator, eine Instanz des Kollaborationstools, der verwendet wird, um ein Kollaborationstool für eine bestimmte Modellierungssprache zu erzeugen. Wie in Abbildung 3 illustriert, lassen sich mit den beiden Hauptkomponenten des Frameworks die Rollen des Modellierungsprozesses assoziieren, welche die jeweiligen Komponenten benutzen. Im ersten Schritt generieren Metamodellierer nach Spezifizierung der abstrakten und konkreten Syntax einer Modellierungssprache ein Kollaborationstool für diese Sprache, das die Modellierer dann zur Bearbeitung von Modellen dieser Sprache verwenden können. Für das Kollaborationstool wurden folgende Anforderungen definiert:

- **Modellierung**: Ein Modellierer kann ein Graph-basiertes Diagramm als visuelle Repräsentation einer definierten Modellierungssprache erstellen. Er kann Knoten und Kanten hinzufügen, verschieben, löschen, sowie deren Eigenschaften bearbeiten.
- **Echtzeitkollaboration**: Mehrere Modellierer können zusammen an einem gemeinsamen Modell arbeiten unabhängig von ihrem physischen Ort. Die Änderungen jedes Benutzers werden dabei zu den anderen Benutzern propagiert, Konflikte aufgelöst, und visuell dargestellt.
- **Änderungshistorie**: Eine Historie aller Änderungen wird gepflegt und allen Benutzern dargestellt. Benutzer können Undo- und Redo-Funktionalität abrufen.
- **Awareness**: Um die gemeinsame Modellierungssitzung besser moderieren zu können, sollen die Benutzer *Awareness* (ein Begriff aus der CSCW-Forschung [DB92]) über die teilnehmenden Modellierer und deren aktuelle Aktionen erfahren. Entsprechende Information soll jederzeit sichtbar dargestellt werden.

- **Modellexport**: Das Modell soll jederzeit, sofern es die Syntax der Modellierungssprache nicht verletzt, in maschinenlesbarer Form exportiert werden können. Für den Prototypen wird sich diese Anforderung auf einen Bildexport beschränken.

Der Kollaborationstoolgenerator ist eine Instanz des Kollaborationstools mit vorgegebener Modellierungssprache. Er erbt damit dessen funktionale Anforderungen und erweitert diese wie folgt:

- **Metamodelldefinition**: Metamodellierer sollen das Metamodell einer abstrakten Syntax einer visuellen Sprache, für die eine Instanz des Kollaborationstools erzeugt werden soll, mittels eines UML-Klassendiagramms definieren können.
- **Konkrete Syntaxdefinition**: Die visuelle Erscheinung der Knoten und Kanten der Modellierungssprache soll entweder durch eine einfache Formbeschreibungssprache oder durch Auswahl vordefinierter Symbole definiert werden können.
- **Kollaborationstoolgenerator**: Metamodellierer können jederzeit, sofern das definierte Metamodell in einem korrekten Zustand ist, die Generierung eines Kollaborationstools für dieses Metamodell anstoßen.

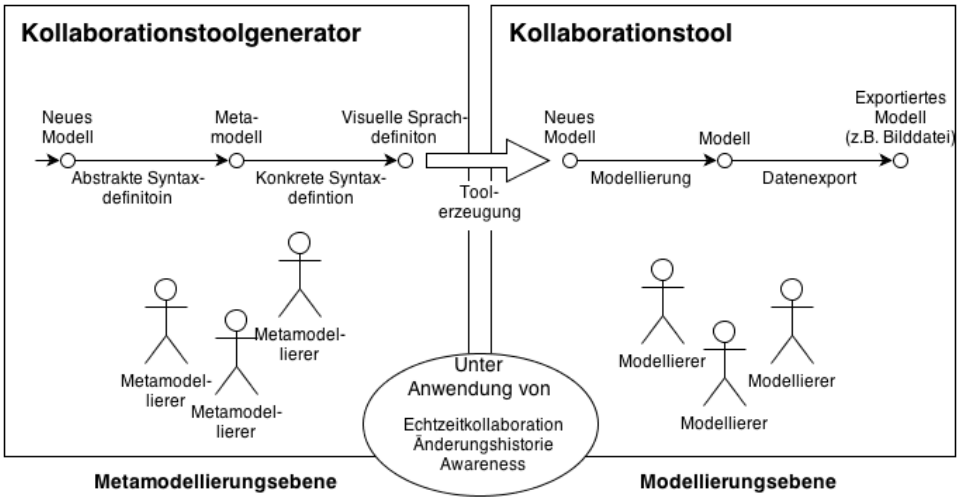

Abbildung 3: Anforderungen eingebettet im Metamodellierungsprozess.

4.2 Systemarchitektur

Die schematisch in Abbildung 4 dargestellte Architektur des vorgestellten Frameworks ist unterteilt in eine Metamodellierungsebene (links im Bild) und eine Modellierungsebene (rechts im Bild). Konzeptionell unterscheiden sich die beiden Ebenen nur geringfügig, da der Kollaborationstoolgenerator auf der Metaebene selbst eine Instanz des Kollaborationstools ist. Daher wird im Folgenden zuerst die Umsetzung der Modellierungsebene erläutert, die analog für die Metamodellierungsebene gilt. Anschließend wird auf Besonderheiten der Metamodellierungsebene eingegangen.

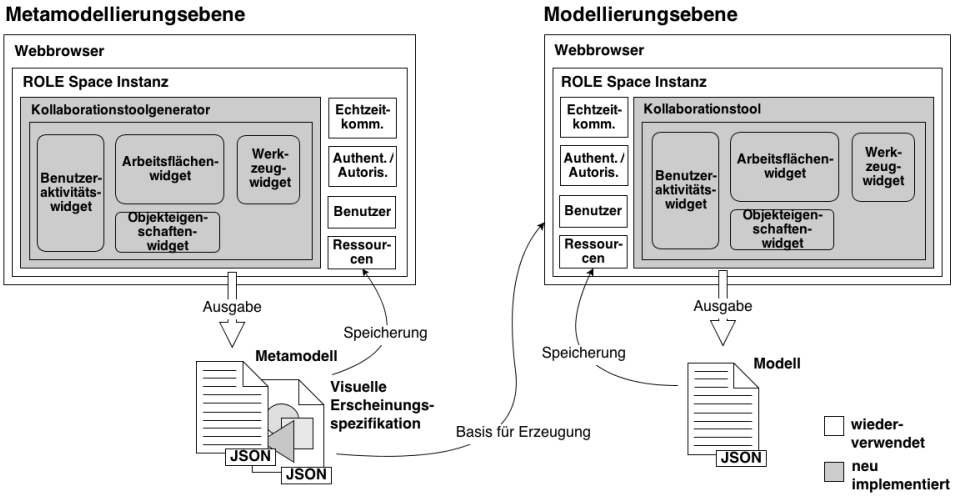

Abbildung 4: Architektur des Echtzeitmetamodellierungsframeworks

Die Architektur fußt grundlegend auf Technologien, die in SyncLD eingesetzt sind und im Abschnitt 3.2 beschrieben sind. In der Symbolik der Abbildung werden weiß gefüllte Symbole „wiederverwendet“, d.h. sie werden von SyncLD übernommen. Die grau gefüllten Symbole werden darauf aufsetzend für das Framework neu implementiert. Das Kollaborationstool lässt sich als Web- und Widget-basierte Anwendung plattformunabhängig ohne Installations- oder Wartungsaufwand verwenden, was auch die Einbindung und Wiederverwendung des Tools in verschiedene Umgebungen ermöglicht. Anders als bei SyncLD, das ebenfalls Widget-basiert realisiert wurde, werden hier die verschiedenen Komponenten der Benutzeroberfläche über mehrere Widgets verteilt, sodass zum einen der Benutzer diese individuell anpassen kann und zum anderen verschiedene Ansichten des Modells parallel betrachtet werden können (siehe Abbildung 5). Neben den üblichen Elementen eines graphischen Editors wie der Arbeitsfläche und einem Werkzeugpanel, erlaubt ein Objekteigenschaftenwidget die Manipulation der Attribute verschiedener Komponenten des in Bearbeitung stehenden Modells. Eingehend auf Benutzerwünsche während der SyncLD-Evaluierung sorgt ein Awarenesswidget für bessere Orientierung, indem es die ausgeführten Aktionen aller Kollaborateure darstellt.

In dem Framework liefert die Implementierung der globalen Widget-zu-Widget-Kommunikation im ROLE SDK so wie in SyncLD die Basis für Propagierung der lokalen Änderungen am Modell an die Kollaborateure. Für diesen Zweck wird wie in SyncLD eine replizierte Architektur mit Konfliktresolution basierend auf Operational Transformation (OT; s. Abschnitt 2.2) verwendet. Die Gefahr eines Flaschenhalses, die bei einer zentralisierten Architektur besteht, wird somit vermieden. Weiterhin kann durch die Methode der Konfliktresolution jeder Benutzer jederzeit jeden Teil des Modells uneingeschränkt bearbeiten, so wie man es von Einzelbenutzeranwendungen kennt. Der Aspekt der größtmöglichen Freiheit im Bearbeitungsprozess wird auch bei der Wahl des Bearbeitungsschemas verfolgt. Eine freihändige Bearbeitungsmöglichkeit des Modells

bildet die Grundlage. Es kann darauf aufbauend strukturiertes Bearbeiten realisiert werden durch situationsabhängige Einschränkung der erlaubten Modellierungsoperationen.

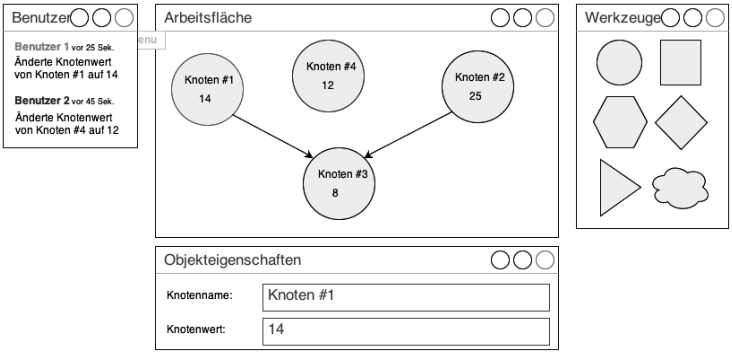

Abbildung 5: Mockup der Benutzerschnittstelle des Kollaborationstools.

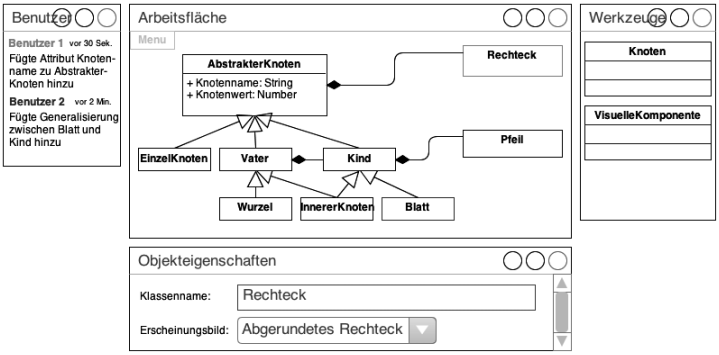

Abbildung 6: Mockup der Benutzerschnittstelle des Kollaborationstoolgenerators.

Auf der Metamodellierungsebene wird die visuelle Modellierungssprache, für die ein Kollaborationstool erzeugt werden soll, definiert. Diese Definition umfasst die Beschreibung der abstrakten und konkreten Syntax. Erstere wird durch ein Metamodell beschrieben, das kollaborativ von mehreren Benutzern erstellt werden kann. Im Detail wird dafür eine in [Mi07] beschriebene Repräsentation des Metamodells, d.h. der Objekte und Beziehungen der Modellierungssprache, als UML-Klassendiagramm verwendet. Für die Spezifikation der konkreten Syntax des Sprache bietet der Kollaborationstoolgenerator eine zusätzliche Komponente für die Festlegung der visuellen Erscheinung der Objekte und Beziehungen des Notation mit Hilfe einer Formbeschreibungssprache wie SVG oder als Auswahl aus vordefinierten Formen ermöglicht (s. Abbildung 6).

4.3 Implementierung

Das Modellierungsframework wird clientseitig auf Basis der aktuellen Web-Technologien HTML5, CSS3 und JavaScript umgesetzt und orientiert sich an der Code-Basis des SyncLD Tools. Wie bei SyncLD basiert auch die Infrastruktur des Modellie-

rungsframeworks auf dem ROLE SDK bzw. insbesondere auf dem Konzept der Widget-Spaces. Als Operational Transformation Bibliothek wird die bereits beim SyncLD-Tool verwendete OpenCoWeb OT JavaScript Engine verwendet. Für die Benutzerauthentifizierung wird das vom ROLE SDK bereitgestellte entsprechende auf OAuth basierende Modul verwendet. Als Datenspeicher für die von den Benutzern erstellten Modelle sowie für den mit dem Kollaborationstoolgenerator erzeugten Quellcode wird der über eine REST API zugreifbare Datenspeicher des ROLE SDK verwendet. Als Datenformat für die erzeugten Modelle dient dabei die JavaScript Objektnotation (JSON). Die Implementierung des Frameworks ist noch nicht abgeschlossen; es sollten jedoch durch die vorangegangene SyncLD-Technologiestudie in einem realen Anwendungsbereich und der starken Anlehnung der Implementierung an diese Technologiestudie (vgl. Abbildung 4) bei der Fertigstellung keine unüberwindbaren Probleme auftreten.

5 Abgrenzung verwandter Ansätze

Wie in Tabelle 1 ersichtlich, wurden in den letzten Jahren diverse Ansätze und Tools vorgeschlagen, die vergleichbare Eigenschaften im Schnittstellenbereich Modellierung und Echtzeitkollaborationssysteme haben, wie das von uns vorgeschlagene Framework. Die Spalten in der Tabelle entsprechen jenen aus dem Hauptziel (siehe die Einleitung des Beitrags) und dem theoretischen Hintergrund (siehe Abschnitt 2) abgeleiteten erwünschten Charakteristika.

Tabelle 1: Vergleich bestehender Ansätze im Hinblick auf die Zielcharakteristika des Frameworks.

Tool / Framework	Plattform / Sprache	Domänen-neutral	Metamodel-lierung	Echtzeitkol-laboration	Awareness	Freihandbe-arbeitung	Strukturierte Bearbeitung	Browser-basiert	Quelloffen
MetaEdit+ [TPK07]	Smalltalk	X	X			X	X		
ADOxx [FK13]	Gecko / C++	X	X			X	X		
DiaMeta [Mi07]	Java / EMF	X	X			X	X		X
[GBR12]	Java / EMF	X	X	X	X	X			
Tiger [Eh05]	Java / EMF	X					X		X
AToM³ [LVA04]	Python	X	X			X	X		X
GenGED [BEW04]	Java	X				X	X		X
SyncLD [NDK13]	ROLE SDK			X		X		X	X
Vorgestelltes Framework	**ROLE SDK**	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**

MetaEdit+ [TPK07] ist ein Tool, das die Definition einer Modellierungssprache ermöglicht und darauf aufbauend ein Bearbeitungstool für Modelle der definierten Sprache generiert. Domänenspezifische Sprachen können dabei graphisch als Metamodell definiert werden ohne jegliche Programmierkenntnisse. Das Bearbeitungstool ermöglicht das Generieren von Programmcode und Dokumentation aus den Modellen sowie die Simula-

tion des Modellverhaltens. Das Tool wurde in Smalltalk implementiert und ermöglicht den Benutzern sowohl freihändiges als auch strukturiertes Bearbeiten.

Eine der kommerziell erfolgreichsten Metamodellierungsplattformen ist ADOxx [FK13], das die Definition von beliebigen konzeptionellen Modellierungssprachen ermöglicht und ein Bearbeitungstool zur Verfügung stellt. Während MetaEdit+ eher auf den Softwareengineering-Prozess getrimmt ist, bietet ADOxx erweiterte Möglichkeiten etwa für Simulation und Evaluation von Geschäftsprozessen. Die ADOxx Plattform basiert auf der Gecko UI Engine und unterstützt ebenso freihändiges wie strukturiertes Bearbeiten.

DiaMeta [Mi07] ist ein plattformunabhängiges Java-basiertes Framework für die Erzeugung von Diagrammeditoren für visuelle Sprachen. Es nutzt Metamodelle zur Spezifikation der Modellierungssprachen, wobei die abstrakte Syntax als Klassendiagram definiert wird. Das Framework basiert auf dem Eclipse Modeling Framework (EMF). Ebenfalls auf dem EMF bzw. seiner Modellierungsarchitektur Ecore basierend stellt Gallardo [GBR12] ein weiteres Tool zur Generierung von domänenunabhängigen Modellierungswerkzeugen vor. Im Gegensatz zu DiaMeta erlaubt es auch Echtzeitkollaboration.

Einen unterschiedlichen Ansatz zur Definition der Modellierungssprache wählt das Tiger Project [Eh05]. Es bietet ein graphisches Tool zur Spezifikation einer Sprache durch Syntaxgrammatiken. Als Benutzerinteraktionsschema verfolgt es die Methode des strukturierten Bearbeitens, wobei anwendbare gültige Operationen aus den Regeln der zu Grunde liegenden Graphgrammatik abgeleitet werden.

Das AToM³ Metamodellierungsframework [LVA04] erlaubt die Definition von Modellen auf Basis von unterschiedlichen Formalismen sowie die Transformation von Modellen in Modelle gleicher oder unterschiedlicher Formalismen. Basierend auf der Spezifikation eines Formalismus kann ein graphisches Tool generiert werden, das die visuelle Bearbeitung von Modellen dieses Formalismus ermöglicht. GenGED [BEW04] gestattet ebenfalls die graphische Definition von visuellen Sprachen zur Generierung von graphischen Editoren für die jeweilig zugrunde liegende Sprache. Modelle können sowohl frei als auch strukturiert bearbeitet werden. Die Sprachspezifikation erfolgt mit Hilfe von Grammatikregeln. Erstellte Modelle lassen sich mit dem Tool simulieren.

Wie der Vergleich in Tabelle 1 zeigt, ist das in dem Beitrag vorgestellte Framework für Echtzeitmetamodellierung durch eine Kombination aus Eigenschaften alleingestellt. Dies begründet sich die Unterstützung von Echtzeitkollaboration, Awareness über aktuell durchgeführte Modellierungsoperationen anderer Benutzer, sowie die Lauffähigkeit im Web-Browser ohne zusätzliche lokal installierte Programme (Spalte „Browser-basiert“).

6 Zusammenfassung

Dieser Beitrag stellte ein Framework vor, welches es ermöglicht, Echtzeitkollaborationstools für beliebige konzeptionelle Modellierungssprachen zu generieren und als Widget-basierte Anwendung anzubieten. Modellierer und Metamodellierer können somit ohne Softwareinstallation oder -wartung gemeinsam und gleichzeitig im Web-Browser an

Modellen und Metamodellen arbeiten. Wie der Vergleich bestehender Ansätze in diesem Forschungsbereich zeigt, sind die Alleinstellungsmerkmale dieses Frameworks, dass es Echtzeitkollaboration im Web-Browser an Modellen und Metamodellen auf Basis offener Web-Technologien, die in allen gängigen Browsern implementiert sind, ermöglicht und dabei vollständig mit quelloffenen Programmbibliotheken sowie mit Web-Technologien, die in allen gängigen Browsern implementiert sind, realisiert ist.

Das Framework basiert auf einer vorangehenden technologischen Machbarkeitsstudie, bei der ein Echtzeitmodellierungstool für eine Lerndesigner-Community implementiert und erfolgreich evaluiert wurde. Die detailliert im Beitrag vorgestellte technische Infrastruktur für die Echtzeitkommunikation und die dafür erforderliche Konfliktresolution, sowie die aus der Evaluation gewonnen Erkenntnisse dieser Technologiestudie bilden das gefestigte Fundament des Frameworks.

Das langfristige Ziel dieser Forschung ist es, den einfachen Einbau von Echtzeitkollaborationsfunktionen in allen Web-Anwendungen, bei denen dies sinnvoll ist, auf Basis offener, interoperabler Technologien und Protokolle zu ermöglichen, ohne dabei auf Black-Box-Lösungen kommerzieller Anbieter angewiesen zu sein.

Danksagungen

Diese Arbeit wurde mit Unterstützung der Europäischen Kommission finanziert durch das Programm für lebenslanges Lernen im Projekt „METIS“ (531262-LLP-2012-ES-KA3-KA3MP – http://metis-project.org), sowie durch das 7. Rahmenprogramm im Projekt „Learning Layers“ (318209 – http://learning-layers.eu). Die Verantwortung für den Inhalt dieses Beitrags tragen allein die Verfasser; die Kommission haftet nicht für die weitere Verwendung der darin enthaltenen Angaben.

Literaturverzeichnis

[BEW04] Bardohl, R.; Ermel, C.; Weinhold, I: GenGED–a visual definition tool for visual modeling environments. In Pfaltz, J. L.; Nagl, M.; Böhlen, B. (Hrsg.): Applications of Graph Transformations with Industrial Relevance. Springer, Berlin, 2004; S. 413–419.

[DSO11] Derntl, M.; Neumann, S.; Oberhuemer, P.: Opportunities and challenges of formal instructional modeling for web-based learning. In: Proc. ICWL 2011, LNCS vol. 7048. Springer, Berlin, 2011; S. 253–262.

[DB92] Dourish, P.; Bellotti, V.: Awareness and coordination in shared workspaces. In: Proc. 1992 ACM Conf. on Computer-supported Cooperative Work. ACM, New York, 1992; S. 107–114.

[Eh05] Ehrig, K. et al.: Generation of visual editors as eclipse plug-ins. In: Proc. 20th IEEE/ACM Int. Conf. on Automated Software Engineering. ACM, New York, 2005; S. 134–143.

[EG89] Ellis, C. A.; Gibbs, S. J.: Concurrency Control in Groupware Systems. In: Proc. ACM SIGMOD Int. Conf. on Management of Data. ACM, New York, 1989; S. 399–407.

[Fa11] Fatima, Z. et al.: Group editor using graphical operational transformation. In: Proc. 5th Nat. Conf. on Computing for Nation Development. IndiaCOM 2011.

[FK13] Fill, H.-G.; Karagiannis, D.: On the conceptualisation of modelling methods using the ADOxx meta modelling platform. In: Enterprise Modelling and Information Systems Architectures, 8(1), 2013.

[FK98] Floyd, C.; Klischewski, R.: Modellierung - ein Handgriff zur Wirklichkeit. Zur sozialen Konstruktion und Wirksamkeit von Informatik-Modellen. In: Pohl, K.; Schürr, A.; Vossen, G. (Hrsg.): Modellierung '98. CEUR-WS.org, 1998.

[GBR12] Gallardo, J.; Bravo, C.; Redondo, M. A.: A model-driven development method for collaborative modeling tools. Journal of Network and Computer Applications, 35(3), 2012; S. 1086–1105.

[Gr94] Grudin, J: Computer-supported cooperative work: History and focus. Computer, 27(5), 1994; S. 19–26.

[Go11] Govaerts, S. et al.: Towards Responsive Open Learning Environments: The ROLE Interoperability Framework. In: Proc. EC-TEL 2011. Springer, Berlin, 2011; S. 125-138.

[IMS03] IMS Global Learning Consortium: Learning Design Specification. 2003. http://www.imsglobal.org/learningdesign/

[JKL09] Jarke, M.; Klamma, R.; Lyytinen, K.: Metamodeling. In: Jeusfeld, M. A.; Jarke, M.; Mylopoulos, J. (Hrsg.): Metamodeling for Method Engineering. MIT Press, 2009; S. 43-88.

[Le94] Levy, S.: Insanely Great: The Life and Times of Macintosh, the Computer that Changed Everything. Penguin Books, New York, 1994.

[LVA04] de Lara, J.; Vangheluwe, H.; Alfonseca, M.: Meta-modelling and graph grammars for multi-paradigm modelling in AToM³. Software and Systems Modeling, 3(3), 2004; S. 194–209.

[Mi07] Minas, M.: Generating meta-model-based freehand editors. Electronic Communications of the EASST, 1, 2007.

[NDK13] Nicolaescu, P.; Derntl, M.; Klamma, R.: Browser-Based Collaborative Modeling in Near Real-Time. In: Proc. IEEE CollaborateCom 2013. IEEE, Los Alamitos, 2013.

[Ol07] Olivé, A.: Conceptual modeling of information systems. Springer, 2007.

[Ri12] Rittgen, P.: The role of editor in collaborative modeling. In: Proc. SAC 2012. ACM, New York, 2013; S. 1674-1679.

[RKV08] Renger, M.; Kolfschoten, G. L.; de Vreede, G.-J.: Challenges in Collaborative Modeling: A Literature Review. Lecture Notes in Business Information Processing, Volume 10, 2008; S. 61-77.

[Sa11] Saint-Andre, P.: RFC 6120: Extensible Messaging and Presence Protocol (XMPP): Core. 2011. http://xmpp.org/

[SC02] Sun, C.; Chen, D.: Consistency maintenance in real-time collaborative graphics editing systems. ACM Transactions on Computer-Human Interaction, 9(1), 2002; S. 1–41.

[SE98] Sun, C.; Ellis, C.: Operational transformation in real-time group editors: issues, algorithms, and achievements. In: Proc. 1998 ACM Conf. on Computer Supported Cooperative Work. ACM, New York, 1998; S. 59–68.

[Su98] Sun, C. et al.: Achieving convergence, causality preservation, and intention preservation in realtime cooperative editing systems. ACM Transactions on Computer-Human Interaction, 5(1), 1998; S. 63–108.

[TPK07] Tolvanen, J.-P.; Pohjonen, R.; Kelly, S.: Advanced tooling for domain-specific modeling: Metaedit+. In: Sprinkle, J.; Gray, J.; Rossi, M.; Tolvanen, J.-P. (Hrsg.): 7th OOPSLA Workshop on Domain-Specific Modeling, Finland, 2007.

[W3C11] W3C: HTML5 Web Messaging. 2011. http: //w3.org/TR/webmessaging/

[XZS00] Xue, L.; Zhang, K.; Sun, C.: Conflict control locking in distributed cooperative graphics editors. In: Proc. 1st Int. Conf. on Web Information Systems Engineering. IEEE, Los Alamitos, 2000; S. 401–408.