

Eine Middleware Infrastruktur für das Teilen von Lerndaten und Diensten zwischen Bildungseinrichtungen

Francois Dubois¹, Truong-Sinh An¹, Agathe Merceron¹

Abstract: Im Projekt “Smart Learning im Handwerk” (SLHw) ist vorgesehen, dass mehrere Bildungseinrichtungen den gleichen Kurs anbieten und dabei dieselben Lernmaterialien nutzen können. Ferner sollen Benutzerinteraktionen mit dem Material lokal gespeichert werden. Um diesen Anforderungen gerecht zu werden, wurde eine Middleware Infrastruktur entwickelt, welche Schnittstellen bereitstellt, um die Kommunikation zwischen Bildungsanbietern und die Nutzung von Diensten zu erlauben, ohne dass diese Bildungsanbieter oder die Dienste direkt miteinander kommunizieren. Dieser Beitrag präsentiert diese Middleware Infrastruktur, ihren Einsatz mit drei Bildungseinrichtungen und vier Kursen sowie eine erste technische Evaluation.

Keywords: Middleware Infrastruktur, serviceorientierte Architektur (SOA), Application Programming Interface (API), Datenmodell, Virtuelle Organisation.

1 Einleitung

Digitale Lernplattformen sind essentiell für Bildungseinrichtungen; insbesondere beim lebenslangen Lernen sind diese ein wesentlicher Bestandteil und gewinnen zunehmend an Bedeutung. Eine Bildungseinrichtung kann beispielsweise eine Universität, eine Schule oder ein externer Dienstleister sein, der Bildungsangebote bereitstellt. In diesem Beitrag wird eine Handwerkskammer (HwK) als externer Dienstleister aufgeführt. Während die ersten digitalen Lernplattformen eine monolithische Architektur zugrunde legten, werden vermehrt auch modulare und serviceorientierte Plattformen entwickelt. Die große Nutzer-Community der weit verbreiteten Moodle Lernplattform stellt immer neue Module bereit, welche in Moodle integriert werden können. Ein Modul, das für Moodle entwickelt wird, kann nicht ohne Weiteres in eine andere Lernplattform integriert werden. Im Zuge der Weiterentwicklung von Moodle und dem vermehrten Einsatz mobiler Endgeräte wird neben dem modularen Aufbau die Integration von serviceorientierten Ansätzen hin zu einer “next generation platform” [Da07] verfolgt.

Es ist nicht selten, dass eine Bildungseinrichtung mehrere Lernplattformen nutzt, oder dass Lehrende an mehreren Bildungseinrichtungen lehren, die verschiedene Lernplattformen einsetzen, wie es insbesondere im Rahmen des lebenslangen Lernens und bei Kooperationen unter diesen auftritt. Um die Kommunikation zwischen den Lernplattformen zu fördern - z. B. der Austausch von Lernmaterialien - wurden Spezifikationen wie LTI entwickelt. Jede Lernplattform, die LTI-Consumer-fähig ist,

¹ Beuth Hochschule für Technik Berlin, Luxemburger Straße 10, 13353 Berlin,
{vorname}.{nachname}@beuth-hochschule.de

kann externes Lernmaterial präsentieren, welches der LTI-Spezifikation folgt. Im Projekt “Smart Learning im Handwerk - SLHw” [Kr16a] wird der Ansatz verfolgt, dass Lernmaterialien, die für eine Handwerkskammer (HwK) entwickelt wurden, auch von anderen HwK genutzt werden können, mit der Besonderheit, dass die Partner weiterhin ihre eigenen Lernplattformen einsetzen. Hierbei müssen es nicht die gleichen Lernplattformen sein. Ferner sieht das Projekt die Entwicklung einer *Recommendation Engine* [Kr16b] und einer *Learning Analytics Platform* [An16] vor. Diese Dienste sollen prinzipiell auch für alle HwK zur Verfügung stehen. Zur Realisierung müssen Benutzerinteraktionen zu dem Lernmaterial erfasst werden. Die Speicherung muss wiederum lokal unter Hoheit der jeweiligen HwK erfolgen, da diese über eine eigene Nutzerverwaltung verfügt. Die Hoheit aus Sicht eines Nutzers, wird dann wirksam, wenn Daten mehrerer Bildungseinrichtungen durch eine Anwendung zentralisiert werden. In diesem Fall muss der Nutzer für jede Bildungseinrichtung dem Anfordern von Daten zustimmen, indem Zugangsdaten bereitgestellt werden.

Diese Anforderungen entsprechen nicht der Funktionalität gängiger Lernplattformen. Um diesen Anforderungen gerecht zu werden, wurde eine Middleware Infrastruktur (MI) entwickelt, welche Schnittstellen bereitstellt, um die Kommunikation zwischen Bildungsanbietern und die Nutzung von Diensten zu erlauben, ohne dass die Bildungsanbieter oder die Dienste direkt miteinander kommunizieren. Somit ist das Integrieren einer weiteren Bildungseinrichtung oder eines neuen Dienstes mit minimalem Aufwand möglich. In diesem Beitrag wird die MI vorgestellt. Dabei wurde Wert auf die Verwendung üblicher Standards und Spezifikationen im E-Learning gelegt. Dies führte zur Entwicklung mehrerer Editoren, um insbesondere Übungen nach der QTI-Spezifikation und Lernmaterial nach der IMS-Spezifikation zu erstellen, da es keine quelloffenen Editoren am Anfang des Vorhabens gab, die Standards und Spezifikationen umsetzen, wie sie im Projekt erforderlich sind.

Der Beitrag ist wie folgt organisiert. Der nächste Abschnitt stellt vorangegangene Arbeiten dar. Im Abschnitt 3 werden die Middleware Infrastruktur (MI) sowie das Repository mit den Editoren eingeführt. Abschnitt 4 präsentiert den aktuellen Einsatz der MI als auch eine technische Evaluierung. Der Beitrag endet mit einer Zusammenfassung und Diskussion.

2 Vorangegangene Arbeiten

Bei der Entwicklung einiger Lernplattformen werden Anwendungsdienste ähnlich des IMS Abstract Framework [IM03], in eigenständige Komponenten aufgeteilt. Mit diesem Ansatz ist es beispielsweise möglich, die Kurs- und Nutzerverwaltung von einem Repository für Lerninhalte zu kapseln und in eine Lernplattform zu integrieren. Dabei kann es sich um eine serviceorientierte Architektur (SOA) handeln. Hierbei wird die infrastrukturelle Kommunikation zwischen verschiedenen Anbietern (Provider) und Konsumenten (Consumer) von Dienstleistungen (Services) geregelt.

Klassisch wird in diesem Zusammenhang oftmals das find-bind-execute- Paradigma angewendet [Zh05]. Hierbei stellt ein Service Consumer eine Anfrage an einen Service Broker. Dieser hat als Vermittler die Aufgabe, einen passenden Service Provider aufzufinden (find). Servicebeschreibungen registrierter Service Provider sind zu diesem Zweck zentral in einer Service Registry hinterlegt. Sobald ein Service Consumer eine passende Servicebeschreibung von dem Service Broker erhalten hat, kann dieser eine Verbindung zu einem Service Provider herstellen (bind). Abschließend erfolgt das Ausführen einer Dienstleistung (execute).

Das folgende Beispiel illustriert, wie Service Consumer und Service Provider in der E-Learning-Branche aussehen können. Im Projekt SLHW wurde eine *Learning Companion App* (LCA) [Kr16a] entwickelt, die Lerninhalte und Empfehlungen für Lerninhalte präsentiert. Diese App benötigt den Zugang zu einem Repository, von dem die Lerninhalte angefordert werden. Die LCA kann als Service Consumer aufgefasst werden. Diese nutzt die Dienstleistungen des Repositories, welche von einer Bildungseinrichtung als Service Provider bereitgestellt werden.

Die Problematik, unter welchen Bedingungen und Restriktionen ein Service Provider Dienstleistungen teilen kann, wird unter anderem im Bereich des Grid Computing diskutiert. Ein Zusammenschluss von Service Providern (Bildungseinrichtungen) für eine zentrale Bündelung von Ressourcen (Lerninhalten) wird in diesem Zusammenhang als Virtuelle Organisation bezeichnet [FK03]. In der E-Learning-Branche wurde das Konzept des Grid-Computing bereits in der Architektur E-Learning Grid erprobt [IJ14]. Durch die Zentralisierung von Service Providern könnten zudem interuniversitäre Anwendungsszenarien realisiert werden, wie beispielsweise gemeinsame Rahmenpläne und Lerninhalte, auch bekannt als Joint-/Double-Degree [AQS08]. Die hier vorgestellte Middleware Infrastruktur basiert auf dem SOA-Paradigma [Zh05] und Microservices [FL15]. Diese Ansätze erlauben ähnliche Anwendungsszenarien wie im E-Learning Grid [IJ14, S. 131].

In [Le08] werden verschiedene Formen des rechnergestützten Lernens vereint. Exemplarisch verknüpft eine serviceorientierte Architektur Multimediageräte für die Video- und Audioaufnahme einer Präsenzveranstaltung sowie die virtuelle Welt “Second Life” als auch das Lernmanagementsystem Stud.IP. Die Grundidee hierbei ist eine Präsenzveranstaltung, um eine virtuelle Veranstaltung zu erweitern. Die dazu benötigten Webservices werden klassisch, wie auch in diesem Beitrag über einen Service Broker, verwaltet. Besitzt eine Anwendung keine Webservice-Schnittstellen, wie beispielsweise Second Life, werden diese über sogenannte Surrogates adaptiert und repräsentiert. Das Adaptieren von Protokollen und Transferdaten wird im Kapitel Infrastruktur näher erläutert.

Eine ähnliche SOA wird in [ZT09] vorgestellt. Besonders hervorzuheben ist hierbei die Herangehensweise Kontext-differenzierte Schnittstellen einer Universität, wie die Administration (Personal, Räume, Termine), die Wissensverwaltung (Archive) und die Lernsysteme zu betrachten. Ein Vorteil von SOA besteht darin, heterogene Systeme zu

orchestrieren. Auf diese Weise können die bereitgestellten Dienstleistungen für Arbeits-, Informations- und Lernprozesse zentralisiert werden. Der Hauptunterschied zwischen dem Ansatz in [ZT09] und dem Ansatz in diesem Beitrag liegt in der Beschreibung der Dienste, welche die Infrastruktur bereitstellt. In [ZT09] wird speziell ein XML-basiertes Format entwickelt: Service Technology-independent Language (STiL). Jeder Dienst wird in STiL repräsentiert. In diesem Beitrag werden Dienste als Microservices beschrieben. Jeder Microservice verfügt über eine REST-API. Sofern vorhanden, verwenden die Schnittstellen und Datenmodelle Standards oder verbreitete Spezifikationen, wie LTI für Lehrmaterialien oder xAPI für Benutzerinteraktionen.

Ein weiterer Ansatz nach [MH12] ist die Konzeption von Lernmanagementsystemen durch die Verwendung von Cloud-Lösungen. Besonders bei Bildungseinrichtungen, die keine Infrastruktur oder Lerninhalte bereitstellen können ist dies attraktiv. Das Teilen oder die Nutzung von Lerninhalten erfolgt über die Cloud. Im Vergleich zu diesem Beitrag ist die Datenhoheit, trotz möglicher Zugriffseinschränkungen einer Bildungseinrichtung, nicht im Fokus, da die Daten extern verwaltet werden. Die Middleware Infrastruktur verwaltet keine Daten und regelt nur den Zugriff. Zudem wird bei einer Cloud-Lösung weniger die Heterogenität der Systeme betrachtet. Vielmehr verfolgt die Middleware Infrastruktur den Ansatz, etablierte Systeme innerhalb einer Bildungseinrichtung zu integrieren, was eine Kooperation oder das Teilen von Lerninhalten durch beliebiges Orchestrieren der Dienstleistungen nicht ausschließt.

3 Infrastruktur

Die Grundidee der Middleware Infrastruktur (MI) und deren wichtigsten Komponenten werden im Folgenden exemplarisch erläutert. Zwei Bildungseinrichtungen B1 und B2 nutzen die eigenen Lernplattformen B1_L1 und B2_L1 für die Verwaltung ihrer Nutzer. Zudem sollen Kurse aus deren lokalen Repositories (B1_R1, B2_R1) den Lernenden zur Verfügung gestellt werden. Die Lernenden verwenden die *Learning Companion App*, welche die Kurse einer Lernplattform bereitstellt. In diesem Szenario sind beide Lernplattformen und Repositories als Service Provider zu betrachten und die *Learning Companion App* als Service Consumer. Der Service Broker mit Hilfe der Service Registry ermöglicht es zu entscheiden, von welcher Bildungseinrichtung, B1 oder B2, die Kursmaterialien angefordert werden. Da die Kursmaterialien aus verschiedenen Repositories angefordert werden, können potenziell unterschiedliche Datenmodelle vorliegen. Zu diesem Zweck hat die MI die Aufgabe, ein einheitliches Datenmodell bereitzustellen, in dem die Datenmodelle der Service Provider transformiert werden. Auf diese Weise kann gewährleistet werden, dass eine Interoperabilität für Service Consumer entsteht. Nachfolgend werden die Begrifflichkeiten von SOA auf die Komponenten der MI überführt.

3.1 Middleware und Management

Aus technischer Sicht lässt sich die Middleware Infrastruktur in die fünf Systemkomponenten Third Party Application (TPA), Third Party Module (TPM), Middleware API (MWA), Hub und Client Application (CA) unterteilen.

Eine TPA ist vergleichbar mit einem Service Provider. Hierbei könnte es sich um Repositories mit Lerninhalten, Learning Record Stores zum Speichern von Benutzerinteraktionen oder Learning Management Systemen von Bildungseinrichtungen handeln.

Ein Third Party Module ist ein Adapter, der ein Datenmodell einer TPA in einen einheitlichen Standard transformiert, um eine interoperable Distribution zu ermöglichen. Als Beispiel transformiert das TPM eine Kursstruktur aus der TPA B1_R1 in ein Datenmodell, das an die Spezifikation IMS CC angelehnt ist (siehe Abbildung 1). Als weiteres Beispiel werden die Benutzerinteraktionen mit dem Lernmaterial als Statements nach der xAPI-Spezifikation transformiert. In Abbildung 1 sind exemplarisch lokale Learning Record Stores (RS) B1_RS1 und B2_RS1 der jeweiligen Bildungseinrichtungen B1 und B2 zum Speichern dieser xAPI-Statements aufgeführt.

Eine Middleware API (MWA) stellt einheitliche Vermittler-Schnittstellen für Dienstleistungen verschiedener TPA bereit und kann mit einem Service Broker verglichen werden, mit der Besonderheit, dass die Schnittstellen durch die Datenmodelle der Third Party Module definiert sind. An dieser Stelle wird das Konzept der Microservices [FL15] aufgegriffen, da flexible Anforderungen in der E-Learning-Branche, wie beispielsweise persönliche Lernumgebungen, zu berücksichtigen sind. Das Konzept der Microservices wird dadurch realisiert, dass es nicht eine einzige API (Application Programming Interface) gibt, sondern mehrere. Aktuell unterteilen sich die MWA nach Kurs- und Nutzerverwaltung (LMS), Aufzeichnen von Lernaktivitäten (LRS), Verwalten von Lerninhalten (LCMS) und Kommunikation zwischen Lernenden und Lehrenden durch Foren (COM, in Abbildung 1 nicht aufgeführt). Die COM-MWA versucht, hierarchische Kommunikationsstrukturen abstrakt abbilden zu können, was aufgrund einer fehlenden Standardisierung die grundlegende Adaptierung für verschiedene Foren ermöglichen könnte. Die verwendeten Entitäten sind Forum, Diskussion und Beitrag. Ein Beitrag kann durch Antworten beliebig verschachtelt werden. Ob eine Diskussion initial einen Beitrag voraussetzt, obliegt der Drittanbieteranwendung.

Jede MWA agiert unabhängig, wodurch auch neue Dienstleistungen nachträglich durch die Einführung weiterer MWA integriert werden können.

Ein pervasiver Ansatz, wie in [ZT09] erfordert, dass jede MWA unabhängig von der Drittanbieteranwendung (TPA) keine erneute Authentifizierung durch den Lernenden benötigt. Bei diesem Middleware Ansatz wird vorausgesetzt, dass die TPA ein Authentifizierungsverfahren wählen, mit dem es ermöglicht wird den Authentifizierungsprozess, wie z. B. bei Single-Sign-On (SSO), zu zentralisieren.

Eine Client-Anwendung (CA) beansprucht die Dienstleistungen einer MWA, um mit einer TPA verbunden zu werden. Im Rahmen des SLHw-Projekts werden insbesondere die *Learning Companion App* und die Editoren als CA integriert (siehe Abbildung 1). Welche TPA von der CA angefordert wird, ist durch sogenannte Routen gewährleistet, siehe Systemkomponente Hub im nächsten Absatz. Um Anforderungen des lebenslangen Lernens und der persönlichen Fortbildung durch Personal Learning Environments zu berücksichtigen, sind CA prinzipiell in der Lage, Dienstleistungen verschiedener MWA beliebig zu verknüpfen (Orchestration), um neue Lernerfahrungen bieten zu können. Es ist hervorzuheben, dass die MI kein Datenaustausch zwischen den TPA initiiert, was die Datenhoheit und Datentransparenz für alle Beteiligten begünstigt. Daraus folgt, dass datenschutzrechtliche Aspekte der E-Learning-Branche durch das verteilte System nicht ausgeweitet werden.

Der Hub ist für das Verwalten von CA-Eigümern (CA Owner in Abbildung 1) sowie Routen verantwortlich und erfüllt die Aufgaben einer Service Registry. In jeder Route wird hinterlegt, welche Zugangsdaten für eine Third Party Application benötigt werden und wie die TPA zu erreichen ist. Für jeden CA-Eigümer wird überprüft, welche Routen verwendet werden dürfen.

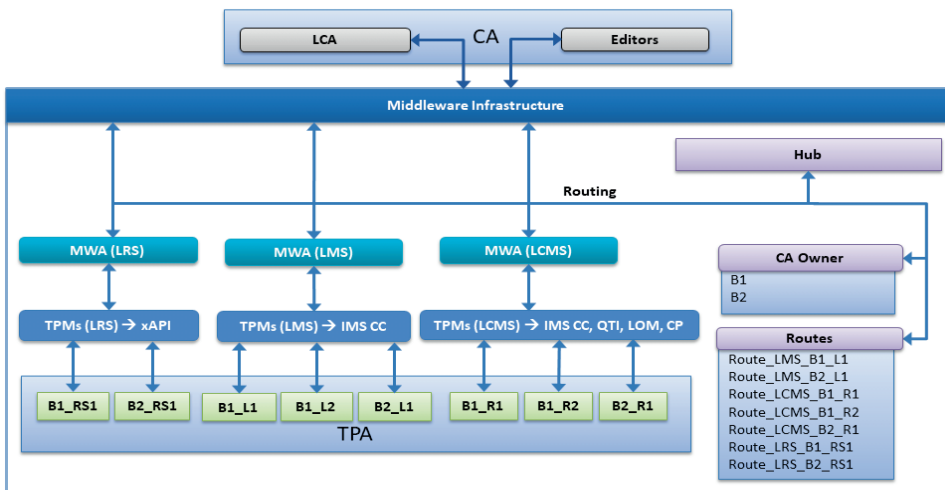


Abb. 1: Middleware Infrastruktur

3.2 Technische Umsetzung

Die Middleware Infrastruktur wurde mit der Softwarearchitektur Java EE entwickelt. Integriert wurde vor allem das Java Naming and Directory Interface (JNDI). Mithilfe dieser Programmierschnittstelle können Objektreferenzen unter Berücksichtigung der Objekt-ID in einer Registry für die spätere Nutzung abgelegt werden. Bei der Middleware Infrastruktur handelt es sich bei den Objektreferenzen um Data Access

The diagram illustrates the relationship between Middleware Infrastructure (Dependencies) and Domain specific solutions. It is divided into two main sections by a dashed line labeled "Core".

Middleware Infrastructure (Dependencies): This section at the top contains a row of eight teal boxes representing dependencies: Scan, DAO, DTO, Exception, Shared, Connector, Config, and Routing.

Domain specific solutions: This section at the bottom contains three boxes: Middleware API (MWA), Third Party Module (TPM), and Model.

Relationships:

- Middleware API (MWA):** Has solid arrows pointing to Scan, DAO, DTO, Exception, Shared, and Connector. It also has a solid arrow pointing to the Model box.
- Third Party Module (TPM):** Has solid arrows pointing to Exception, Shared, Connector, Config, and Routing. It also has a dashed arrow pointing to the Model box.
- Model:** A dark blue box at the bottom center, receiving solid arrows from MWA and TPM, and a dashed arrow from TPM.

Legend:

- Scope Compile:** Represented by a solid line.
- Scope Provided:** Represented by a dashed line.

Abb. 2: Middleware Infrastruktur (Abhängigkeiten)

Tab. 1: Funktionale Module

Beschreibung	Module
Auflösen von verfügbaren TPM in einer MWA	Scan
Aufsuchen der Data Access Objects 1	DAO
Transferobjekte für einen einheitlichen Datenaustausch	DTO
Fehlerbehandlung der Middleware Infrastruktur	Exception
Util-Klassen und geteilte Datenmodelle	Shared
Verbindungstypen der Third Party Modules	Connector
Logik zum Einlesen einer Konfigurationsdatei	Config
Anfragen von Routen	Routing

Innerhalb eines TPM werden alle funktionalen Module aus der Tabelle 1 über das Build-Management-Tool Maven mit dem Scope provided versehen. Das bedeutet, dass nur die MWA, die im späteren Verlauf beliebig viele TPM einbindet, Abhängigkeiten zur Verfügung stellt. Ein TPM nutzt die funktionalen Module der MWA.

3.3 Repository und Editoren.

Im Projekt SLHw wurde für die Ablage von Lernobjekten (LO) und Kursstrukturen ein Repository als TPA geschaffen. Zudem wurden zwei CA zum Verwalten und Präsentieren von LO entwickelt (vgl. Editors in Abbildung 1). Das Autorentool openStudio [An16] ist eine Webapplikation und ermöglicht durch diverse Editoren das Erstellen und Pflegen von LO – inklusive QTI-Aufgaben – und deren Metadaten sowie das Komponieren von LO in Lerneinheiten. Das Präsentationstool openPresentation [An16] visualisiert unter anderem die LO, welche durch die Dienstleistungen der LCMS-Middleware API von einem Repository (siehe B1_R1 oder B2_R1 in Abbildung 1) angefordert wurden. Die LO in dem Projekt SLHw sind die kleinste wiederverwendbare Einheit mit eigenen Metadaten nach der Spezifikation IMS LOM. Projektbezogen benötigt jedes LO Metadaten mit mindestens einem Lernziel und kann jede Form von Rich Media Typen annehmen. Ein LO kann demnach ein Video, eine Animation, ein Dokument oder eine Übung nach der Spezifikation IMS QTI sein. Technisch gesehen ist ein LO jedoch eine Kapselung verschiedener Typen nach der Spezifikation IMS CP.

4 Einsatz und Evaluierung

Das Szenario mit mehreren Handwerkskammern, welche auf dieselben Lernobjekte (LO) aus einem einzigen Repository zugreifen, wurde getestet, aber mit realen Lernenden aus organisatorischen Gründen nicht umgesetzt. Es wurde das Szenario von drei Bildungseinrichtungen realisiert, welche gemeinsam die *Learning Companion App* als Client-Anwendung benutzen, um Zugang zu vier Kursen zu ermöglichen.

4.1 Anwendung: 4 Kurse von 3 Bildungseinrichtungen

Im Rahmen des Projektvorhabens wurde der Fortbildungslehrgang “Gebäudeenergieberater” an der Handwerkskammer Berlin im Herbst 2016 mit einer Gesamtdauer von 24 Wochen durchgeführt. Weiterhin wurde die gesamte Infrastruktur auch im Wintersemester 2016/2017 an der Technischen Universität Berlin (TU-Berlin) und an der Beuth Hochschule für Technik Berlin (Beuth-HS) erprobt. An dem Kurs “Advanced Web Technologies” im Studiengang Master Informatik der TU-Berlin haben sich 142 Studenten über 12 Wochen kursbegleitend auf die Prüfung vorbereitet. Im Kontrast dazu wurde der Kurs “JavaFX” der Beuth-HS allen Online-Studenten im Studiengang Bachelor Medieninformatik über einen Zeitraum von 10 Wochen zur Verfügung gestellt. Die Teilnahme an dem Kurs war freiwillig und enthielt keine prüfungsrelevanten Leistungen. Es beantragten 53 Interessierte einen Zugang zu dem Kurs. Weiterhin wurde ein “Fitnesskurs” von der Handwerkskammer Berlin zu Demozwecken angeboten.

Drei der bisherigen Kurse starteten zu unterschiedlichen Terminen und liefen über eine Phase von 3 Monaten parallel. Hierbei betrieb jede Bildungseinrichtung ein eigenes Repository. Die insgesamt über 2400 LO wurden von den Bildungseinrichtungen mit openStudio eigenverantwortlich in die jeweiligen Repositories eingepflegt. Des Weiteren verwaltet jede Bildungseinrichtung mit deren Lernplattformen die eigenen Benutzer. Alle Kurse wurden den Benutzern über die *Learning Companion App* (LCA) bereitgestellt. Da die LCA einen Benutzer nicht einer Lernplattform zuordnen kann, spielt die MI mit ihren *Route*-fähigen Middleware APIs eine wesentliche Rolle bei der Kommunikation mit den entsprechenden TPA. Abstrakt beschrieben kommuniziert die LCA für die Benutzer- und Kursverwaltung mit der Lernplattform B1_L1 über die *Route* *Route_LMS_B1_L1*; für die Lerninhalte integriert die LCA das LTI-Tool openPresentation. Dieses kommuniziert mit dem Repository B1_R1 über die *Route* *Route_LCMS_B1_R1* (vgl. Abbildung 3, oben). Die Nutzer können auch die Lernplattform direkt als Zugangspunkt nutzen. In diesem speziellen Szenario integriert die Lernplattform B1_R1 das LTI-Tool openPresentation (vgl. Abbildung 3, unten).

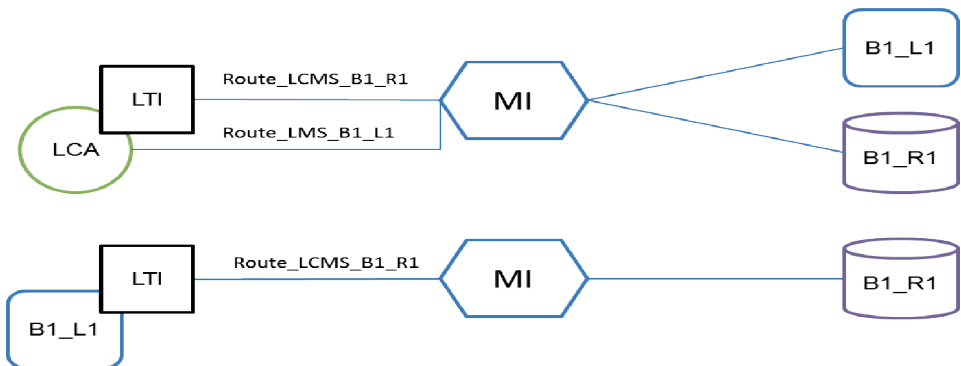


Abb. 3: Bildungseinrichtungen mit getrennten Repositories

Im Rahmen der Öffentlichkeitsarbeit wurde zusätzlich ein Demo-Setup (Lernplattform B1_L2, siehe Abbildung 1) aufgesetzt, bei dem Interessierten die Möglichkeit geboten wird, einen Ausschnitt des Fortbildungslehrgangs “Gebäudeenergieberater” oder einen “Fitnesskurs” zu testen. Hierbei liegt der Kurs “Gebäudeenergieberater” im Repository B1_R1 und der “Fitnesskurs” im Repository B1_R2. Die Interessierten können über die Demo Lernplattform B1_L2 den Kurs “Gebäudeenergieberater” und den “Fitnesskurs” aus den beiden Repositories B1_R1 und B1_R2 über die *Routen* Route_LCMS_B1_R1 und Route_LCMS_B1_R2 besuchen. Somit ist es möglich, dass über die Lernplattformen (B1_L1, B1_L2) auf denselben Kurs des Repository (B1_R1) zugegriffen werden kann. Ebenso ist es möglich, dass eine Lernplattform B1_L2 verschiedene Kurse aus unterschiedlichen Repositories (B1_R1, B1_R2) anfordert (siehe Abbildung 4).

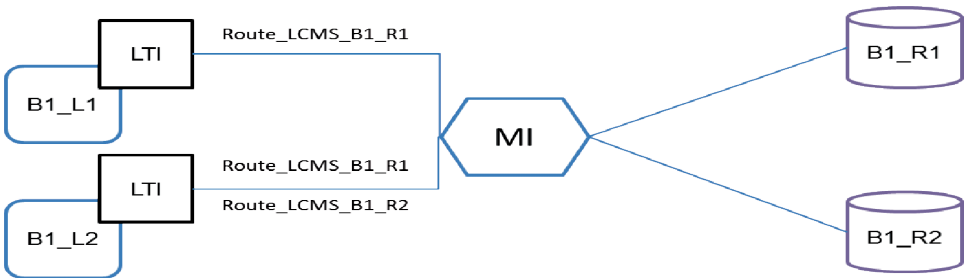


Abb. 4: Bildungseinrichtungen nutzen verschiedene Repositories

4.2 Technische Evaluation

Neben den realen Kursen erfolgte in einer kontrollierten Testumgebung eine vorläufige quantitative Evaluation der MI. Die Auswertung ist als Richtwert zu verstehen, da für Stresstests keine Produktionsumgebung gewährleistet werden konnte. Die Daten werden zudem durch geringe Latenzzeiten des Intranets verzerrt. Für alle Systemkomponenten wurden die gleichen technischen Spezifikationen gewählt. Jedes Testszenario erfolgte zeitlich versetzt 8-mal, um mögliche Abweichungen durch Störfaktoren zu minimieren. Das erste Ergebnis zeigt die Verarbeitungszeit der MWA, ohne Dienstleistungen einer TPA anzufragen. Bei 250 simultanen Clients mit jeweils 100 sequenziellen Anfragen beträgt die Laufzeit eines Requests im Durchschnitt 0,48 Sekunden. Das zweite Ergebnis ermittelte den Datendurchsatz der MI im Vergleich zu einem direkten Aufruf einer TPA. Der Overhead der MI hinsichtlich des Datendurchsatzes ergibt bei 50 simultanen Clients mit jeweils 250 sequenziellen Anfragen in der kontrollierten Testumgebung bei Moodle (TPA) eine Differenz von etwa 3%. Als Hauptfaktoren, die maßgeblich die Performance beeinflussen, konnte die Netzwerkinfrastruktur sowie die Performance der einzelnen TPA ermittelt werden. Zweitrangig ist die Datentransformation und das Auffinden eines TPM. Generiert und erfasst wurden die Testdaten mit Hilfe von Apache JMeter als CA Simulation und VisualVM für das Remote-Monitoring des Anwendungsservers der MI.

Für technisch versierte Leser ist anzumerken, dass ein Bottleneck durch die Verwendung eines Hubs ausgeschlossen wird, indem die Routen lokal in den MWA zwischengespeichert werden.

5 Zusammenfassung und Diskussion

In diesem Beitrag wurde die Middleware Infrastruktur vorgestellt, welche im Projekt SLHw entwickelt wurde. Diese serviceorientierte Infrastruktur erlaubt eine flexible Kommunikation und Mutualisierung von Diensten zwischen Bildungsanbietern. Sie wurde mit drei Bildungseinrichtungen und vier Kursen erfolgreich eingesetzt. Obwohl nur Szenarios im Bereich E-Learning vorgestellt wurden, ist die Middleware Infrastruktur unabhängig von der Anwendungsdomäne. Diese könnte beispielsweise mit Diensten und Third Party Applications aus dem Bereich E-Health erweitert werden: Angebot eines Fitness-Kurses mit Integration von Fitness-Armbändern. Voraussetzung ist, dass passende Third Party Modules und Middleware APIs entwickelt werden.

Die Middleware Infrastruktur hat Potenzial für virtuelle Organisationen, wie z.B. die Virtuelle Fachhochschule (VFH) [Vi17]. In der VFH bieten mehrere Hochschulen gemeinsam Online-Studiengänge an. Jede Hochschule ist für die Entwicklung von digitalen Medien in einigen Kursen verantwortlich. Mit der Autorensoftware *LOOP* (Learning Object Online Plattform) können die digitalen Medien entweder direkt in der zentralen Ablage oder an jedem Standort lokal entwickelt und gepflegt werden. Anschließend werden diese in das zentrale Repository übertragen. Die MI würde ermöglichen, dass die digitalen Medien aus den lokalen Repositories der jeweiligen verantwortlichen Hochschule zu einem einzigen Kurs virtuell zusammengeführt werden.

Es ist geplant, die Middleware Infrastruktur, das Repository, die Editoren und LTI-Tools zum Ende des Projektes im September 2017 quelloffen zu veröffentlichen.

Danksagung: Wir danken B. Bayat, M. Bärenfänger, R. Chandru, M. Dinziol, M. Edriss, I. Fritsch, H. John, D. Jürgensen, J. Kania, C. Krauß, E. Manthey, U. Meurer, S. Müller, M. Scharp und M. Zwicklbauer für Diskussionen. Das diesem Beitrag zugrundeliegende Vorhaben wird mit Mitteln des BMBF Förderkennzeichen 01PD14002B gefördert. Die Verantwortung für den Inhalt liegt bei den Autoren.

Literaturverzeichnis

- [An16] An, T.-S.; Dubois, F.; Manthey, E.; Merceron, A.: Digitale Infrastruktur und Learning Analytics in Co-Design. In: Proceedings of the Workshop Learning Analytics, co-located with the 13th e-Learning Conference of the German Society for Computer Science, Potsdam, Germany, September 11, 2016, <http://ceur-ws.org/Vol-1669/>, 2016.

- [AQS08] Aguirre, S.; Quemada, J.; Salvachúa, J.: Work in progress - Developing Joint Degrees through e-Learning systems. Proceedings - Frontiers in Education Conference, FIE, ISSN 15394565. doi: 10.1109/FIE.2008.4720474, S. 19–20, 2008.
- [Da07] Dagger, D.; O'Connor, A.; Lawless, S.; Walsh, E.; Wade, V. P.: Service-Oriented e-learning platforms: From monolithic systems to flexible services. IEEE Internet Computing, ISSN 10897801. doi: 10.1109/MIC.2007.70, S. 28–35, 2007.
- [FK03] Foster, I.; Kesselman, C.: The Grid 2: Blueprint for a New Computing Infrastructure. Elsevier Ltd., 2 edition, ISBN 978-1558609334, 2003.
- [FL15] Fowler, M.; Lewis, J.: Microservices: Nur ein weiteres Konzept in der Softwarearchitektur oder mehr? Objektspektrum, S. 1–7, 2015.
- [Fo04] Fowler, M. 2004. Inversion of Control Containers and the Dependency Injection pattern, <https://www.martinfowler.com/articles/injection.html#>, Stand: 13.02.2017.
- [IJ14] Ivanovic, M.; Jain, L. C.: E-Learning Paradigms and Applications: Agent-based Approach. Springer, ISBN 978-3-642-41964-5. doi: 10.1007/978-3-642-41965-2, 2014.
- [IM03] IMS Global Learning Consortium. IMS Abstract Framework: White Paper, 2003, <https://www.imsglobal.org/af/afv1p0/imsafwhitepaperv1p0.html>. Stand: 12.02.2017.
- [Kr16a] Krauss, C.; Merceron, A.; An, T.-S.; Zwicklbauer, M.; Arbanowski, S. The Smart Learning Approach - A mobile Learning Companion Application. In: Proceedings of The Eighth International Conference on Mobile, Hybrid, and On-line Learning (eLmL 2016). IARIA, April 24 - 28, Venice, Italy, 2016.
- [Kr16b] Krauss, C.: Smart Learning: Time-Dependent Context-Aware Learning Object Recommendations. Proceedings of the 29th International Florida AI Research Society Conference (FLAIRS-29), AAAI, Key Largo, S. 501-504, 2016.
- [Le08] Lehsten, P.; Thiele, A.; Zilz, R.; Dressler, E.; Zender, R.; Lucke, U.; Tavangarian, D.: Dienste-basierte Kopplung von virtueller und Präsenzlehre. In: DeLFI 2008, S. 77-88, 2008.
- [MH12] Masud, M. A. H.; Huang, X.: An e-learning system architecture based on cloud computing. system, 10/11, S. 736-740, 2012.
- [Vi17] Virtuelle Fachhochschule, <https://www.vfh.de>. Stand: 20.03.2017.
- [ZT09] Zender, R.; Tavangarian, D.: Service-oriented university: Infrastructure for the university of tomorrow. In: Intelligent Interactive Assistance and Mobile Multimedia Computing, S. 73-84. Springer Berlin Heidelberg, 2009.
- [Zh05] Zhu, H.: Challenges to reusable services. In: Proceedings - 2005 IEEE International Conference on Services Computing, SCC 2005, ISBN 0769524087. doi: 10.1109/SCC.2005.36, S. 243–244, 2005.