




CampusSourceEngine

—

Die Schnittstelle von *e-Learning* Systemen zum HIS-GX System der HIS GmbH

Arbeitsgruppe CampusSource

CampusSource	(Dr.-Ing. Michael Stepping, Dipl.-Ing. Manfred Postel)
FernUniversität Hagen	(Dipl.-Ing. Thorsten Kisner, Prof. Firoz Kaderali, Prof. Hans Werner Six)
Universität Dortmund	(Dipl.-Phy. Arne von Irmer, Josef Hüvelmeyer)
Universität zu Köln	(Matthias Kunkel)
Universität Münster	(Dipl.-Inf. L. Dewanto, Dr. F. Bensberg, Prof. L. Grob)
Universität Duisburg-Essen	(Dipl.-Ing. Dieter Nastoll, Dr. Bruno Lix)
HIS GmbH	(Dr. Ludwig Leute, Dr. Jauer)

Gefördert durch das 
Ministerium für Wissenschaft und Forschung des Landes Nordrhein-Westfalen,
freigegeben durch die HIS GmbH.

Inhalt

1	Einleitung	1
2	Die Funktion von CampusSource	2
3	Identifizierung gemeinsamer Dienste	3
4	Stand der Arbeiten	5
4.1	Rollenkonzept – „Prüfung“	5
4.2	Authentifizierung - Identity Management – <i>Single Sign-On</i>	6
4.3	Model Driven Architecture - MDA	9
4.4	<i>e-Learning</i>	13
4.5	Technische Implementierung	14
5	Gemeinsame Basisdienste	16
5.1	Datenquellen	18
5.2	Schnittstellen	21
5.3	Dokumentation / Meta-Information	22
6	Zusammenfassung	23
	Literatur	24

1 Einleitung

In den letzten Jahren sind an vielen Hochschulen und anderen Aus- und Weiterbildungseinrichtungen zahlreiche Anstrengungen unternommen worden, einen Online-Auftritt über das Internet bereitzustellen. Daraus entstanden im Laufe der Zeit immer mächtigere *e-Learning*-Plattformen, die insbesondere die Lehre an den Aus- und Weiterbildungseinrichtungen unterstützen.

Die Vorteile durch den Einsatz dieser Plattformen bestehen darin, dass u. a. zeit- und ortsunabhängiges Informieren, Lehren und Lernen ermöglicht wird. Dies sind nur einige der Funktionalitäten die durch eine *e-Learning*-Plattform erfüllt werden können.

Unter einer *webbasierten Lernplattform* beziehungsweise *e-Learning*-Plattform wird ganz allgemein eine serverseitig installierte Anwendung verstanden, die beliebige Lerninhalte über das Internet vermittelt und die Organisation der dabei notwendigen Lernprozesse unterstützt [BHM].

Dabei werden fünf grundlegende Funktionsbereiche von webbasierten Lernplattformen unterschieden. Dies sind im einzelnen die *Präsentation von Inhalten* (Learning Content), *Werkzeuge* zur Erstellung von Aufgaben und Übungen (Assignment), *Evaluations- und Bewertungshilfen* (Assessment), *Administration* (von Lernenden, Lehrenden, Inhalten, Kursen, Lernfortschritten, Terminen, usw.) und *Kommunikation* (synchron beziehungsweise asynchron) [BHM]. Der umgangssprachlich verwendete Begriff „Lernplattform“ wird als Synonym für die exaktere Bezeichnung *Learning Management System (LMS)* verwendet [BHM].

Im Gegensatz dazu hat sich im Bereich der Verwaltungen der Hochschulen in NRW die Software der Hochschul-Informationssysteme GmbH – *HIS*¹ – als Standard etabliert. Die *HIS*-Software verwaltet Datenbestände, die von zentraler Bedeutung für die Hochschulen sind.

Der Einsatz von *e-Learning*-Plattformen und deren Werkzeugen an den Universitäten erfordert eine bessere Integration der *HIS*-Daten in die Infrastrukturen der Hochschulen.

Integration mit den Datenbeständen bedeutet, dass aus der verteilten Infrastruktur einer Hochschule die für *e-Learning* relevanten Daten eruiert und bereitgestellt werden. Ziel ist es, die bestehenden Datenbestände unter anderem in den *HIS*-Systemen über standardisierte Schnittstellen dem Bereich *e-Learning* zur Verfügung zu stellen.

Als zentrales Bindeglied dient hier die *CampusSourceEngine (CSE)*. Ziel der *CSE* ist die Definition und Implementation einer einheitlichen Schnittstelle der bei *CampusSource* verfügbaren *e-Learning*-Plattformen zu den *HIS*-Datenbeständen.

Weiteres, hier nicht relevantes Ziel der zu implementierenden *CSE*-Architektur ist es, einerseits eine zielgerichtete Konvergenz der verfügbaren Systeme und Werkzeuge in *CampusSource* zu erlangen, andererseits den Anwendern die Möglichkeit zu geben, aus einer Vielzahl von Funktionalitäten, die die unterschiedlichen Systeme anbieten, eine passgenaue, individuelle Plattform zusammenzustellen.

¹ Die Software der Hochschul-Informationssysteme GmbH – *HIS* –, Hannover, ist in vielen Bundesländern für die Verwaltungen der Hochschulen verpflichtend eingeführt.

Die konkrete HIS-Schnittstelle, als eine zentrale Aufgabe der *CampusSourceEngine*, ist in Abbildung 1 als „Mittler“ zwischen den verschiedenen Plattformen und anderen in der Hochschule vorhandenen Ressourcen dargestellt.

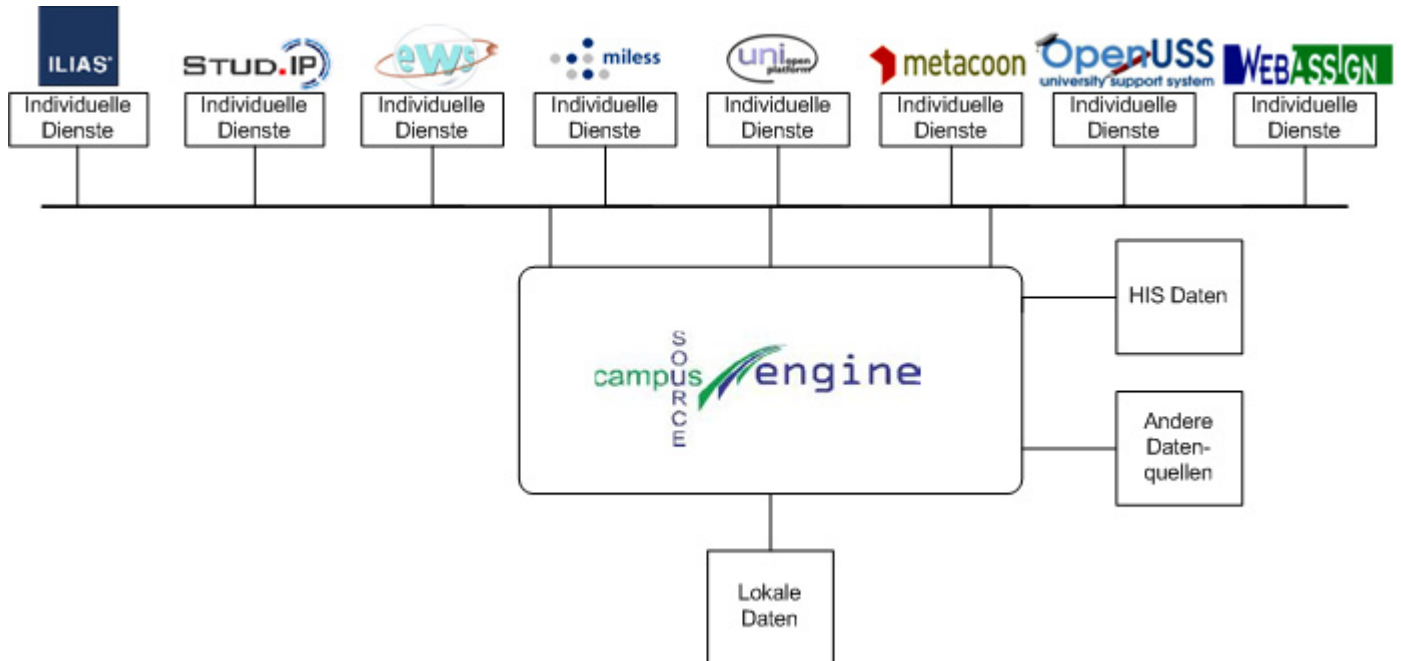


Abbildung 1: CampusSourceEngine als Mittler zwischen Plattformen und der HIS-Schnittstelle

Mit dieser Konstruktion wird es einfacher, flexibel auf die unterschiedlichen Anforderungen der Softwareinfrastruktur der Hochschulen reagieren zu können (z. B. die unterschiedlichen Lernplattformen). Diese unterschiedlichen Anforderungen werden in der CSE gebündelt.

2 Die Funktion von CampusSource

An den deutschen Hochschulen sind zahlreiche, verschiedene *e-Learning*-Plattformen und weitere darin integrierte oder gesondert verfügbare Werkzeuge (Übungssysteme, Bibliothekssysteme, Prüfungssoftware, Content Erstellungs-, Autoren- und Managementsysteme, usw.) im Einsatz. Im Rahmen der Open Source-Initiative *CampusSource* werden zur Zeit 18 Systeme¹ angeboten. Sie sind überwiegend im Bereich der Hochschulen und Universitäten entstanden und derzeit bei einem Drittel aller deutschen Hochschulen (ca. 110 von 330) im Einsatz. Die Systeme stehen unter der GNU *General Public License* als Open Source-Software zur Verfügung. *CampusSource* ist weltweit die größte Open Source-Softwarebörse für *e-Learning*-Plattformen und -Werkzeuge. Zurzeit besteht die Community aus ca. 3500 registrierten Anwendern und Institutionen.

Eine hohe Anzahl von auf dem Markt befindlichen konkurrierenden Open Source *e-Learning*-Plattformen bedeutet, dass die Entwicklergemeinschaft und auch die Zahl der Anwender pro System („Administration und Wartung – Administration and Maintenance“) kleiner wird. Derzeit sind nach [BHM] weltweit mehr als 400 proprietäre und Open Source *e-Learning*-Plattformen am „Markt“ verfügbar.

¹ Stand Learntec 2005, 10.02.-14.02.2005, Karlsruhe

Da es keine erzwungene Begrenzung der Anzahl der *e-Learning*-Plattformen gibt, wird der Weg der natürlichen Konvergenz der Systeme durch die Selektion mittels der Anwender erfolgen. Dabei wird von ausschlaggebender Bedeutung sein, inwieweit die Systeme einfach in die vorgefundene Software-Infrastruktur integriert werden können und inwieweit offene Schnittstellen vorhanden sind, im Hochschulbereich insbesondere die Schnittstellen zu den benutzten HIS-Systemen. Prognostiziert wird daher eine Marktbereinigung in den nächsten Jahren.

Alle in der CampusSource-Börse verfügbaren und im Einsatz befindlichen Open Source-Plattformen haben daher beschlossen, Ihre Kapazitäten zum Einsatz und Betrieb Ihrer Systeme derart zu bündeln, dass alle Grundfunktionalitäten, die alle Systeme benötigen, gemeinsam entwickelt bzw. weiterentwickelt und gepflegt werden.

Interessant ist in diesem Zusammenhang, dass sich für die HIS GmbH das Problem stellt, dass für viele im Einsatz der Hochschulen befindliche *e-Learning*-Plattformen, insbesondere auch kommerziellen *e-Learning*-Plattformen, proprietäre Anpassungen und Schnittstellen entwickelt und implementiert werden mussten. In Zeiten knapper finanzieller Mittel kann sich in der heutigen Situation weder eine Hochschule noch die HIS GmbH den „Luxus“ erlauben, weitere Schnittstellen zu spezifizieren, einzeln zu implementieren und zu pflegen. Daraus resultiert die Anforderung in erster Linie der HIS GmbH, möglichst einheitliche (offene) Schnittstellen zu **allen** *e-Learning*-Plattformen zu definieren.

In der Praxis ist dies allerdings sehr aufwändig, da auch unter Einhaltung vorhandener (auch externer) Standards eine Kompatibilität auf funktionaler Ebene zwischen verschiedenen Systemen noch nicht gegeben ist. Zum einen, weil die jeweiligen Architekturkonzepte und implementierten Schnittstellen sehr unterschiedlich sind und zum anderen, noch viel wichtiger, weil unterschiedliche Programmiersprachen und Bibliotheken zum Einsatz kommen.

Eine solche Modularität auf Funktionsebene und die Konvergenz zwischen den Systemen ist dringend gefordert und ist in den letzten Monaten im Rahmen dieses Projektes konzeptionell und beispielhaft, dargestellt im Folgenden, erarbeitet worden.

Ausgangspunkt sind die in *CampusSource* verfügbaren Systeme sowie die HIS Systeme.

Das Ziel war die Identifizierung der Schnittstellenmethoden und Attribute sowie das Protokoll für eine einheitliche und verbindliche HIS-CSE-Schnittstelle zwischen den *e-Learning*-Plattformen durch die *CampusSourceEngine* und dem Hochschulinformationssystem HIS.

3 Identifizierung gemeinsamer Dienste

Ausgehend von Abbildung 1 können gemeinsame, zu zentralisierende Dienste der einzelnen *e-Learning*-Plattformen herausgearbeitet werden. Diese sind in Abbildung 2 dargestellt als *Basismodule*. Diese Basismodule bilden in der Funktion der zentralen Dienstleister den Kern der *CampusSourceEngine*.

Nach der vorgeschlagenen Architektur beherbergen die dargestellten *standortspezifischen Module* dann die bisherigen plattformspezifischen Funktionalitäten der jeweiligen verwendeten *e-Learning*-Plattform.

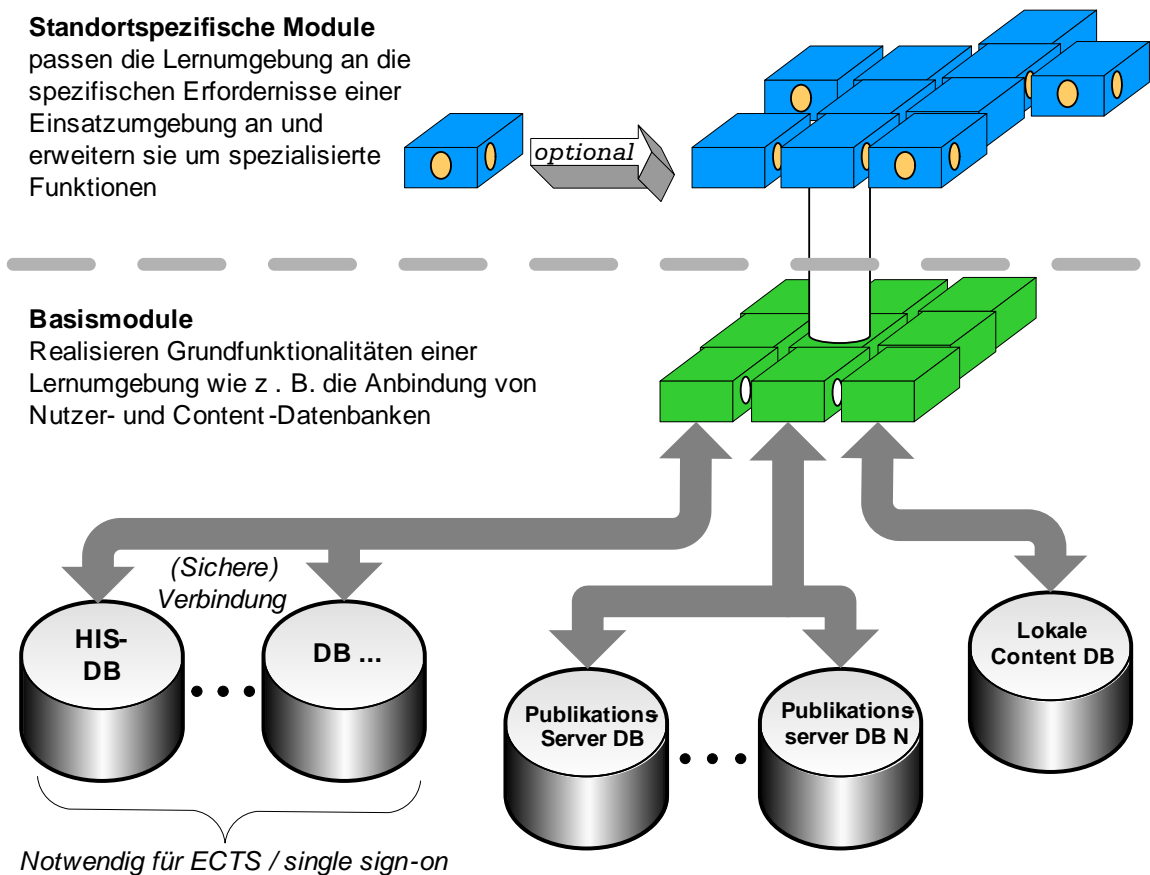


Abbildung 2: optionale und zentrale Dienste des e-Learnings

Die für die Realisierung der Funktionalitäten in den Lehr- und Lernplattformen benötigten Daten beziehungsweise Datensätze liegen gewöhnlich (historisch bedingt) in der Hochschuldateninfrastruktur verteilt vor (Abbildung 2). Dies sind die neben den Datenbanken des HIS, die Verzeichnisdienste des Rechenzentrums (DB, ...), die Datenbanken für die Webpräsenzen (Publikationsserver DB) und auch Datenbanken der *e-Learning*-Plattform (Lokale Content DB). In Deutschland bildet in den Hochschulen die HIS-Software ein Kernelement u.a. für die verwaltungsseitige und administrative Datenhaltung. Diese Datenhaltung bietet bisher keine für *e-Learning*-Plattformen geeignete Schnittstelle an.

In der Vergangenheit wurden zahlreiche, sehr unterschiedliche Vereinbarungen mit verschiedenen Entwicklern von Plattformen getroffen, umgesetzt und implementiert, um damit einen individuellen Zugriff zu den erforderlichen Daten aus den HIS Datenbanken zu ermöglichen. Bei dem steigenden Einsatz verschiedener Lernplattformen und Werkzeugen stellt sich dieser Ansatz als aufwändig, kostenintensiv und oft unzureichend heraus.

In der zu entwickelnden *CampusSourceEngine* (CSE) sollen deshalb in einem ersten Schritt grundlegende Basisfunktionalitäten zur HIS-Software spezifiziert und realisiert werden. Damit sollen möglichst alle Basis-Anfragen zu den erforderlichen Daten aus den HIS Systemen der verschiedenen Plattformen und Werkzeuge abgedeckt werden können. Ähnliche Schnittstellen zwischen der CSE und, wie oben angedeutet, anderen Datenquellen (z. B. Bibliotheksdatenbank, RADIUS-Services, Directory Services) sollen zukünftig ebenso spezifiziert und realisiert werden.

Die *CampusSourceEngine* wird damit eine zentrale Aufgabe als Datenvermittler zwischen den Plattformen und Werkzeugen im Bereich des *e-Learning* und den (HIS) Datenbanken der verschiedenen Hochschulsoftwareinfrastrukturen übernehmen. Da die CSE die dezentral

vorliegenden Daten in einem abstrahierten Datenmodell abbildet, werden die Daten weiter in den ursprünglich vorhandenen Datenbanken mit deren Administrations- und Wartungs-Werkzeugen verbleiben. Dies bedeutet zum Beispiel, dass für die bisherigen Mitarbeiter kein Umstieg auf neue Software-Werkzeuge notwendig wird.

4 Stand der Arbeiten

Im Folgenden werden verschiedene, ausgewählte Aspekte einer zu erstellenden *CampusSourceEngine* beschrieben.

Im Abschnitt „**Prüfung**“ werden Basisabläufe in den beteiligten Softwaresystemen dargestellt. Diese werden aufgeteilt in resultierende Funktionen der Basisdienste und Erweiterungsdienste der CSE.

Die Implementierung der Architektur unterstützenden Dienste für die CSE (z. B. zentrales Dienstverzeichnis) wird an dem Beispiel des „**Identity Managements**“ erläutert.

Der Abschnitt „**Model Driven Architecture**“ beschreibt mögliche formale Methoden der Modellierung von Basisabläufen mit Hilfe moderner Software-Engineering Techniken. In den Plattformen werden mit Hilfe der Abstraktion und geeigneter (Open Source) Werkzeuge Dienste, Funktionalitäten und Protokolle auf hohem Abstraktionsniveau modelliert und mit geeigneten Vorgehensweisen in Quellcode überführt.

Zum Schluss wird der momentane **Diskussionsstand** bezüglich der Schnittstellenmodelle zur HIS-Software aufgezeigt.

4.1 Rollenkonzept – „Prüfung“

Im diesem Abschnitt „Rollenkonzept-Prüfung“ werden die benötigten Daten zur Anbindung eines Prüfungsdienstes, zu implementieren in die CSE, an HIS spezifiziert.

Für die Anbindung eines Prüfungsdienstes an HIS müssen Prüfungsdaten und Teile der personenbezogenen Daten des Prüfers (z. B. das Recht, eine Prüfung abzunehmen) und des Prüflings gelesen und Ergebnisse der Prüfungen geschrieben werden. Hierzu wird eine bidirektionale Schnittstelle zu HIS-GX mit den Modulen POS¹, SOS und ZUL spezifiziert.

Ein zusätzlicher Aspekt dieser Teilspezifikation ist die Absicherung des Datentransfers durch eine geeignete Sicherheitsinfrastruktur und Rechtemanagement. In diesem muss der Zugriff auf die Daten geregelt werden:

Ein Prüfer benötigt z. B. keinen Zugriff auf alle vorhandenen Daten des Prüflings, sondern nur auf einen definierten und für ihn relevanten Teilbereich.

Die zu vergebenden Rollen repräsentieren den Funktionsumfang, der einer Person oder Systemen zur Verfügung steht oder die jeweilige Sicht auf die Daten. In den Datenbanken HIS-SOS, HIS-POS und HIS-ZUL ist in den Studentenstammdaten beispielsweise die Bankverbindung gespeichert. Diese Daten sind für die meisten Benutzer der an die CSE angeschlossenen Module nicht von Interesse.

Für die Verwendung dieser Daten ist demnach eine spezielle Rolle erforderlich. Diese kann an die Anwendungsfälle (Use Case) innerhalb der CSE vergeben werden (Abbildung 3). Zusätzlich können Rollen an angeschlossene Module oder letztendlich an angemeldete Benutzer vergeben werden.

¹ Beschreibung der Module, siehe Kapitel „5.1 Datenquellen“.

Eine Speicherung dieser (neuen) Rollendaten in den HIS-Systemen ist nicht erforderlich, vielmehr bietet sich eine Speicherung innerhalb der CSE an. Das Rollenkonzept regelt so lediglich die Zugriffsrechte auf Daten und Funktionen innerhalb der CSE.

Ein Anwendungsfall ist beispielsweise die Änderung der E-Mail-Adresse in den Stammdaten von HIS. Bedingung für die Änderung ist eine Erlaubnis für die CSE, diese Daten grundsätzlich ändern zu dürfen.

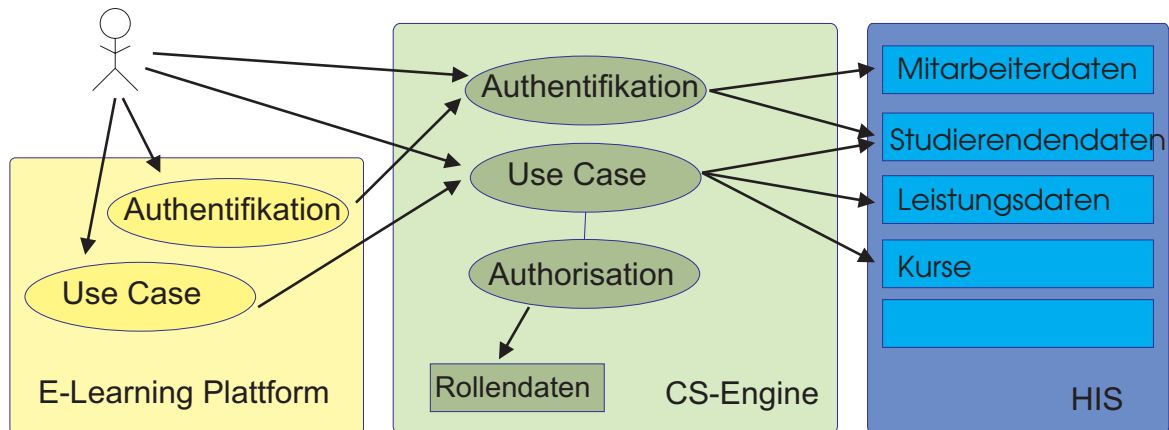


Abbildung 3: Rollenmodell

Aus diesem Rollenmodell (Abbildung 3) können erste, rudimentäre Klassen bezogen auf das Szenario „Prüfung“ entwickelt werden. Im Rahmen der Weiterentwicklung dieses Projekts sollen diese Klassen, Entitäten und Kardinalitäten noch verfeinert werden. Klassenspezifikationen sind im Klassenmodell dargestellt (Abbildung 4).

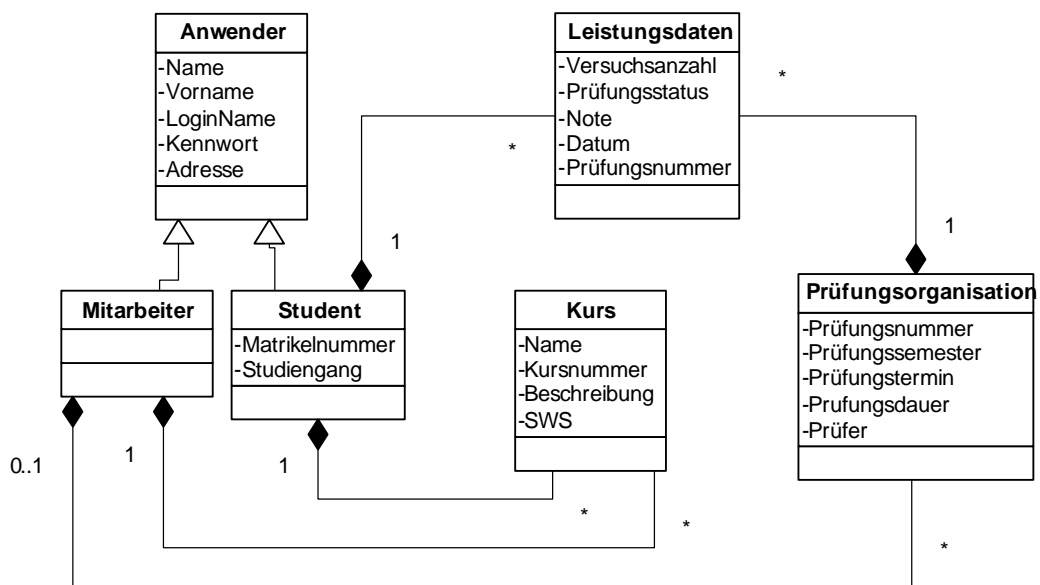


Abbildung 4: Klassenmodell Prüfung

4.2 Authentifizierung - Identity Management – Single Sign-On

Die Bedeutung des *Identity Managements* an den Hochschulen wächst mit der Zunahme der digitalen Dienste an den Hochschulen. Hier ist das *Single Sign-On* (SSO) ein wichtiges Ziel für die nächste Zukunft. Durch SSO werden die Anwendungen aus Benutzersicht einfacher,

aber es stellt sich die Frage: „Wem kann man digital trauen?“. Bisher wurde durch die Immatrikulation oder die Unterschrift unter einen Arbeitsvertrag an der Hochschule die Identität der handelnden Personen verifiziert. Durch das *European Credit Transfer System (ECTS)* kommen in diesem als auch in weiteren hier behandelten Punkten noch neue Anforderungen auf die Hochschulen zu.

Da das *Identity Management* mit dem *Single Sign-On* eine so zentrale Rolle innerhalb der *e-Learning*-Plattformen einnimmt, ist eine Verlagerung aus den einzelnen Plattformen in die *CampusSourceEngine* als zentraler Basisdienst sinnvoll.

Diese verifizierten Daten liegen in den HIS-GX Modulen POS, SVA und SOS vor. Durch eine stärkere Integration in die Hochschule wird eine Infrastruktur geschaffen, mit der Dienste möglich werden, die sich darauf verlassen müssen, dass die Identität eines Useraccounts abgesichert ist. Damit muss die Einbindung/Anbindung an HIS-LSF, welches das Modul für elektronische Lehrveranstaltungsverwaltung und ein elektronisches Vorlesungsverzeichnis ist, erfolgen. Die Identitäten und die Rolleninformationen werden zur Verfügung gestellt. Es ist ein hierarchisches Rollenmodell vorgesehen (Abbildung 5). Grob dargestellt hat eine Person eine „Tätigkeitsorientierte Rolle“, bspw. die Basisrolle „Student“, davon abgeleitet wird der „Account x“ und die konkrete Rolle „Berechtigung“.

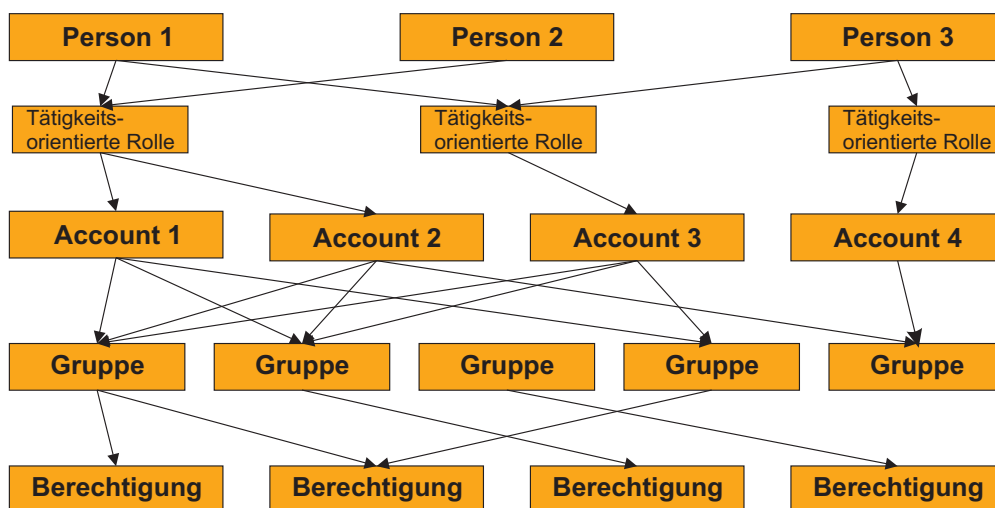


Abbildung 5: Rollenmodell *Identity Management*

Innerhalb von der CSE werden die Informationen über Identitäten und Rollen zwecks Zugriff und Zugriffsart ausgewertet. Die Rechteverwaltung erfordert:

- **Prinzipale** (Benutzer, Gruppen, etc.)
werden abgeleitet aus Identitäts- und Rolleninformationen – *Identity Management*
- **Objekte** (Dateien, Dokumente, etc.)
Knoten mit Vater-Kind-Beziehung modellieren ein hierarchisches Dokument (z. B. Dokument besteht aus vielen Dateien, auch mit unterschiedlichen Formaten)
- **Rechte** (permission, right, privilege, etc.)
Access Control Listen (ACL) entlang der Knoten, die die Dokumenthierarchie darstellen.

Das *Single Sign-On* bietet und fordert eine einheitliche Datenhaltung, damit ein einheitlicher Zugang realisiert werden kann. Für diese „Elektronische Identität“ werden die folgenden Lösungen vorgeschlagen:

- Technische Lösungen:

- gemeinsames, universitätsweites Verzeichnis
- Identitäts-, Rechte-, Zugangsverwaltung
- Organisatorische Lösungen:
 - die Verwaltung als primäre Datenquelle
 - die Erzeugung von Kennung und Rolle beim ersten Kontakt (Immatrikulation, Einstellung).

Die heutigen Quellen für die Daten sind mannigfaltig und redundant. Beispielsweise sind auch in der Telefonanlage Daten zu einem Mitarbeiter genauso gespeichert, wie die bekannte Speicherung für einen Useraccount als den Mitarbeiter. Beide Systeme sind in der Regel nicht oder nur minimal in der Hochschulwelt miteinander gekoppelt. Als Datenquellen wurden identifiziert:

- HIS-SOS
- HIS-SVA
- Quelle für Strukturdaten (derzeit HISCOB /MBS)
- BuiSy
- Telefonanlagen
- Benutzerverwaltung Bibliotheken
- Ressourcendaten (Computer, Netzwerkkomponenten, ...)

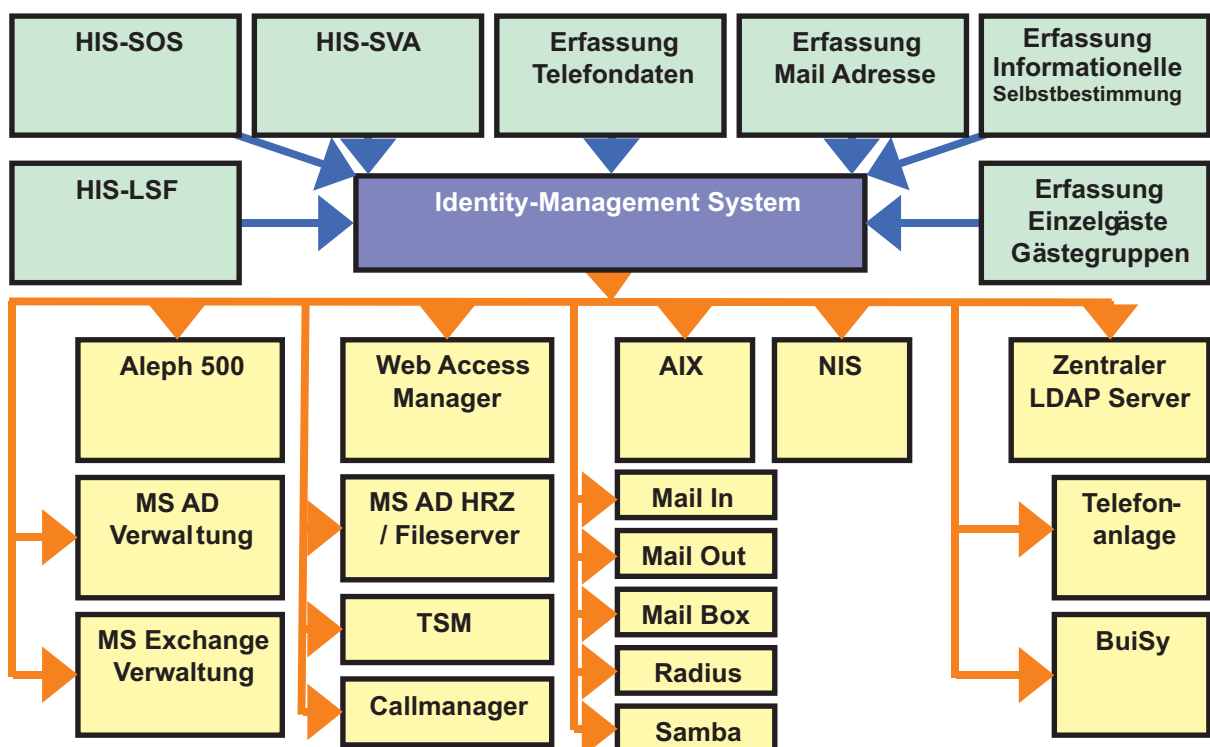


Abbildung 6: Identity Management System in Verbindung mit HIS

Abbildung 6 illustriert das Zusammenspiel der verschiedenen Datenquellen mit dem *Identity Management System*. Dieses bietet den darunter dargestellten Diensten (gelb) die eindeutige, verifizierbare und authentische Nutzeridentität an. Damit kann die Datenspeicherung zu der jeweiligen Identität in den einzelnen Teilsystemen entfallen.

4.3 Model Driven Architecture - MDA

Zur Realisierung der *CampusSourceEngine* werden Software-Entwicklungsprozesse benötigt. Die *Model Driven Architecture* ist eine in den letzten Jahren eingesetzte Entwurfstechnologie im Rahmen des Software Engineering. In den letzten zehn Jahren hat sich zur Konzeptionierung, zum Entwurf, zur Implementierung und zur Dokumentation die von Grady, Booch und Rumbaugh ursprünglich entworfene *Unified Modelling Language* etabliert. Die Model Driven Architecture baut hierauf auf und liefert viele automatisierte Entwurfsmuster, um von einer Abstraktionsebene auf die nächste Konkretisierungsebene zu gelangen.

Das eröffnet für den Entwurfsprozess der *CampusSourceEngine* eine Möglichkeit, auf einem hohen Abstraktionsniveau die gewünschten Prozesse beziehungsweise Funktionalitäten zu modellieren und möglichst vollautomatisch in ausführbaren Code umzusetzen.

Die bestehende CampusSource-Plattform OpenUSS wurde einem Reverse Engineering Prozess unterzogen und daraus wurden abstrakte Modelle der OpenUSS-*e-Learning*-Plattform erstellt. Aus diesen Modellen sind im letzten Schritt ausführbarer Code erzeugt und gegen den Originalcode geprüft worden.

Für diesen Umsetzungsprozess beziehungsweise für die MDA gibt es heute Open Source-Werkzeuge, die diese Design Patterns, bspw. nach *Gamma*, *Vlissides*, usw., auf das Domänenmodell und das daraus entstehende Applikationsmodell zur vollautomatischen Generierung des Klassenmodells anwenden. Somit ist z. B. mit der MDA ein größtmögliches, automatisiertes Top-Down-Design möglich. Zurzeit sind die Werkzeuge am meisten für die Programmiersprache Java fortgeschritten. Durch die Offenheit dieses Ansatzes ist eine Generierung für andere Programmiersprachen aber ebenso möglich. Es müssen dann „nur noch“ die Design Patterns der jeweiligen Programmiersprache implementiert werden.

Damit eröffnet sich für die Umsetzung der *CampusSourceEngine* ein hoher Zeitvorteil gegenüber bisherigen traditionellen Software-Entwicklungsprozessen.

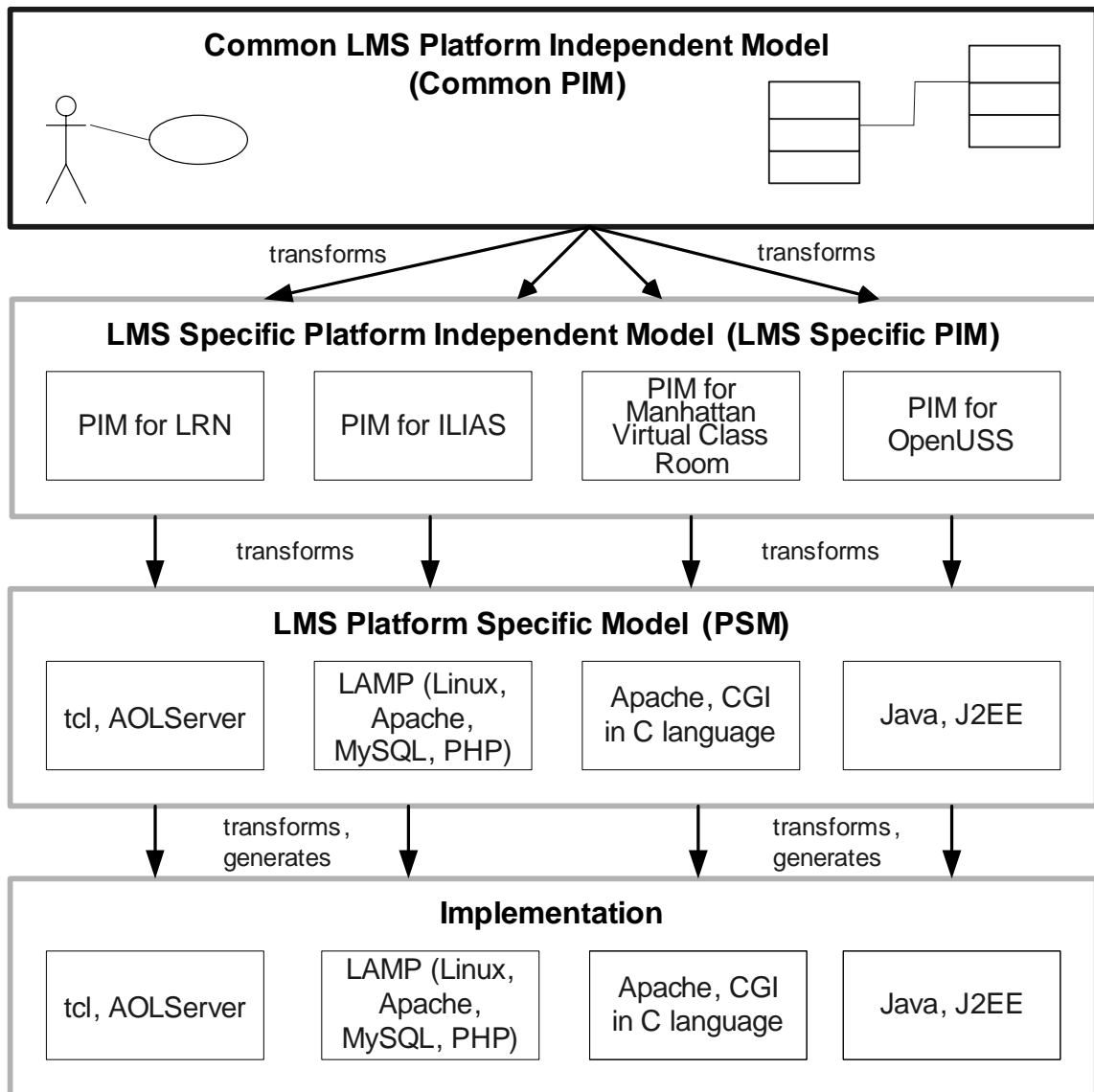


Abbildung 7: CampusSource Plattformen in MDA

Abbildung 7 stellt einen Überblick über die heutigen Plattformen und Techniken dar. Hier sind schon die gemeinsamen Basisdienste als *Common Platform Independent Model* gekapselt. Diese werden ergänzt durch die plattformspezifischen Dienste *LMS Specific Platform Independent Model*. Die Transformation ergibt das *Platform Specific Model*. Auf dieses PSM werden die programmiersprachenspezifischen Entwurfsmuster angewendet und es entsteht die Implementation in der jeweiligen Programmiersprache.

Dieses Vorgehen wurde anhand der Plattform *OpenUSS* exemplarisch verifiziert. Hier wurde im ersten Schritt durch Reverse Engineering der bestehenden Plattform das *Common PIM* und das *LMS Specific PIM* erstellt. Fehlten bei den Transformationen noch Entwurfsmuster, so wurden diese nachimplementiert. Das Ergebnis, bezogen auf OpenUSS, ist im Codevergleich zur Originalimplementierung eine nahezu identische Reproduktion des Codes.

Durch das Vorhandensein der *Platform Independent Modelle* können nun auf einer hohen Abstraktionsebene neue Dienste relativ einfach hinzugefügt werden und durch die automatisierte Implementierung steht dieser neue Dienst dann zeitnah zur Verfügung.

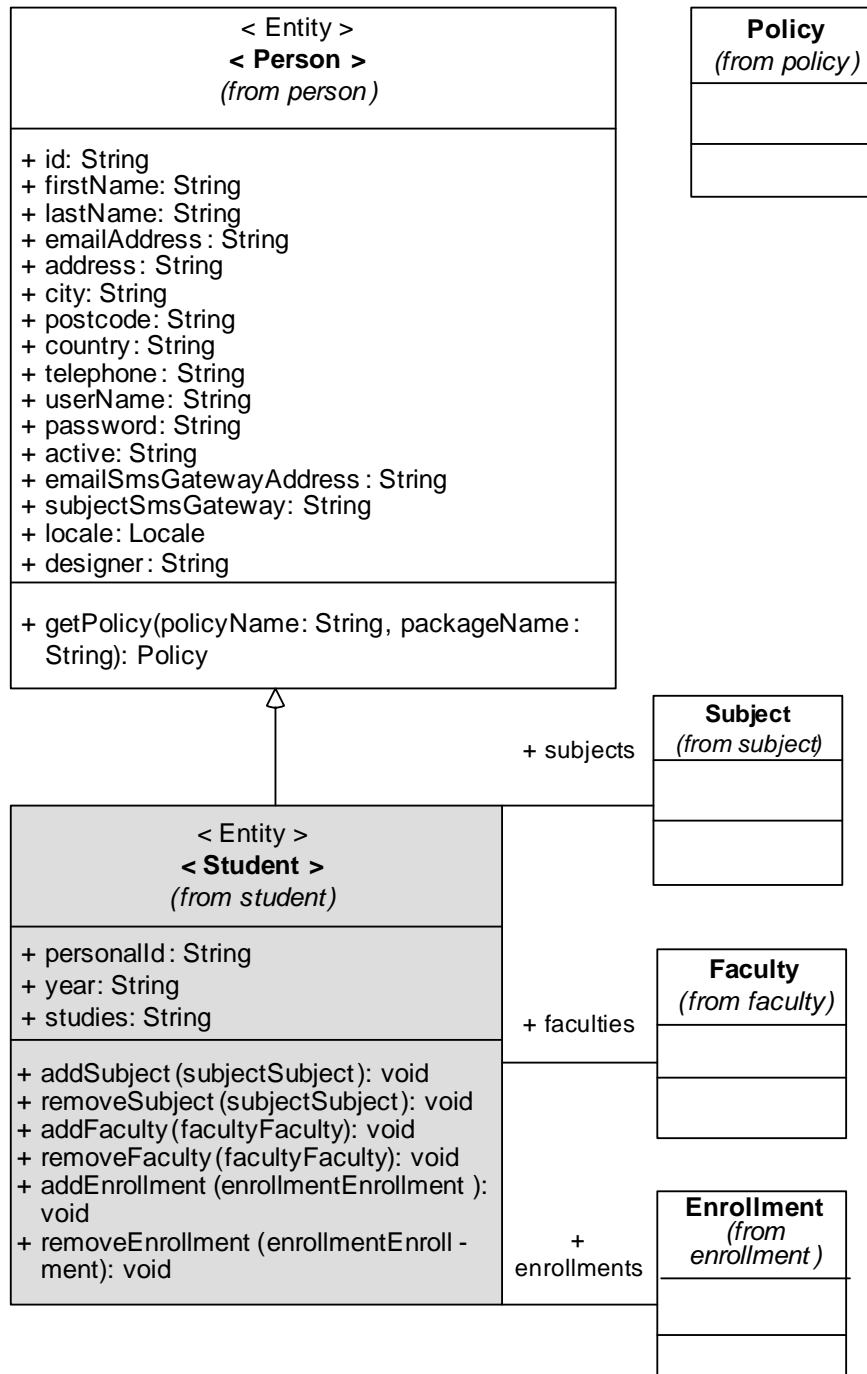


Abbildung 8: Person Entität – Modell

Abbildung 8 zeigt die Entitäten Person und Student als allgemeine Form in dem *Common Platform Independent Model*.

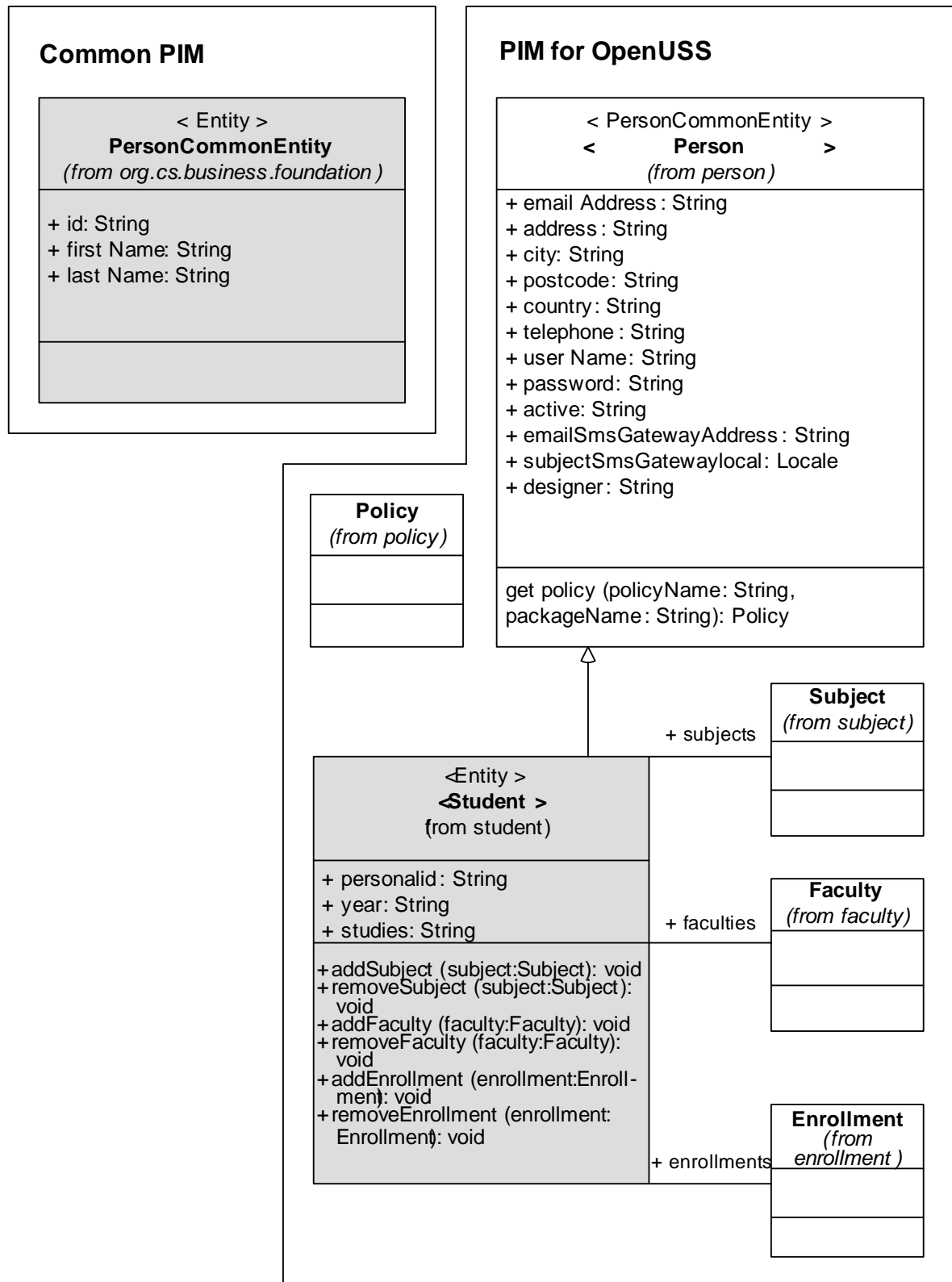


Abbildung 9: LMS Platform Independent Model (PIM)

Abbildung 9 zeigt die Konkretisierung der Entität `PersonCommonEntity` aus dem `Common PIM` in das `LMS PIM`.

In einem letzten, hier nicht mehr dargestellten Schritt kann unter Zuhilfenahme der Design Patterns die konkrete Implementierung in einer konkreten Programmiersprache erstellt werden.

Es ist gerade am letzten Schritt erkennbar, dass die Entwurfsmethoden für die Domänenmodell- und Applikationsmodell-Ebenen keine Programmiersprachen benötigen.

4.4 e-Learning

Der Ansatz des Blended Learning erfordert z. B. eine starke Kopplung der real gehaltenen Veranstaltungen mit den zugehörigen virtuellen Angeboten. HIS-GX mit den Modulen LSF hält die Daten der Veranstaltungen vor. Dieses Modul kann die Daten im Web darstellen oder generiert beispielsweise das Vorlesungsverzeichnis. Durch die Anbindung einer e-Learning-Plattform an HIS-GX-LFS können die Veranstaltungsinformationen z. B. um virtuelle Angebote erweitert werden. Der in HIS-GX-LFS generierte Stundenplan kann so für jede dargestellte Veranstaltung einen Link auf die dazugehörigen Lehr-/Lerninhalte in der e-Learning-Plattform darstellen.

e-Learning profitiert gleichzeitig von der Integration des *Identity Managements* und der Prüfungsdaten.

Ein Beispiel:

Die Identität des Studierenden ist gesichert, seine Beiträge in der synchronen und asynchronen Kommunikation einer Veranstaltung können zur Beurteilung des Studenten und damit ggf. zur Erlangung eines studienrelevanten Abschlusses der gesamten Veranstaltung (Präsenz- und virtuelle Veranstaltung) herangezogen werden.

Für die Realisierung des e-Learning werden von der CSE-HIS-Schnittstelle die im Folgenden dargestellten Objekte benötigt.

*Aus den **Anmeldedaten** (Benutzerangabe, Kennwort) wird die Identität erstellt und damit der Status (Lehrender, Studierender, Gast, unbekannt) ermittelt. Zu den personenbezogenen Daten (E-Mail-Adresse, usw.) stehen damit auch die Rolle und die damit eingeräumten Rechte fest.*

*Zu den Veranstaltungsdaten werden zum einen Informationen zur **Veranstaltung**, wie die Veranstaltungsnummer, benötigt. Zum anderen können die hinter einer Veranstaltung abzulegenden Daten, wie Titel, Lehrender, max. Teilnehmerzahl, abgerufen werden.*

*Die Veranstaltungsdaten werden in einer **Veranstaltungsliste** geführt. Diese enthält dann eine Liste aller Veranstaltungsnummern.*

*Die Veranstaltung enthält einen Verweis auf eine **Teilnehmerliste**. Diese wiederum enthält einen Rückwärtsverweis auf die Veranstaltung über eine vorrätig zu haltende Veranstaltungsnummer. Wichtigster Zweck der Teilnehmerliste ist aber die Liste der angemeldeten Studierenden.*

***Ergänzende Veranstaltungsinformationen** können beispielsweise die URL des Lernraums (Learning Space) sein.*

***Leistungsnachweise** enthalten eine Teilnahmebescheinigung, einen Leistungsnachweis und die zugeteilte Note oder die Credit Points.*

Die Geschäftsabläufe im Fachkonzept der Plattformen müssen die folgenden noch offenen Fragen beantworten:

- wo und wie werden Änderungen in den Datenbeständen (z. B. Änderung des Titels einer Veranstaltung) gespeichert und
- welche Daten müssen auch in einer Lernplattform gespeichert werden (redundante Datenhaltung)?

4.5 Technische Implementierung

Ein wichtiges Hemmnis bei den bisherigen Bemühungen war, dass erwägt wurde, die Persistenz aller Objekte von HIS übernehmen zu lassen. Dies hat HIS aus aufwandsbedingten Gründen abgelehnt. Daher sieht das generelle Konzept nun den Informationsaustausch zwischen dem Modul HIS-LSF und der CSE **nachrichtenbasiert** vor.

Jedes Projekt hat seine eigene Persistenz außerhalb von HIS und innerhalb, entweder der CSE oder der originären Plattform. Durch Nachrichten (sogenannte *Events*) werden diese Datenhaltungen synchronisiert. Jede Applikation muss *Events*, die über diese sehr einfache Schnittstelle in das System kommen, verarbeiten. Beispielsweise: Student ex- oder immatrikuliert; Veranstaltung gelöscht oder angelegt, usw..

Umgekehrt muss HIS entsprechende *Events* der Partnerprojekte verarbeiten, die über dieselbe einfache Schnittstelle eintreffen. Beispielsweise: Virtueller Lernraum zur Veranstaltung x auf Plattform y ist unter der URL: <http://www.yyy.de/xxx> erreichbar.

Vorteile dieser Technik sind:

- **Listen** (Broker) für die Interessenten einer Nachricht können geführt werden. Dadurch ist dieses System skalierbar und kann erweitert werden.
- Der Nachrichtensender kann ein „**Shoot and forget**“ machen. Er kann eine Ereignisnachricht „abfeuern“ und braucht nicht auf das Ergebnis zu warten (asynchroner nachrichtenbasierter Mechanismus).
- Das **Nachrichtenformat** ist ein definierter Vertrag zwischen HIS und CSE. Damit ist dies auch gleichzeitig in der Implementierung ein „Code-Point“ für Konformitätsmessungen. Es kann eine eindeutige Zuordnung von Fehlern zu deren Verursachern getroffen werden.
In einem späteren Schritt wird nur noch über neue Meldungen und deren konkretes Format verhandelt. Der Mechanismus bleibt auf beiden Seiten bestehen.
- Die eigentliche Schnittstelle ist **einfach realisierbar**. Es besteht die Empfehlung, die Verfahren SOAP/XML-RPC zu verwenden. Eine Implementierung dieser Verfahren kann in den gängigen Programmiersprachen JAVA/PHP/.Net/Pearl usw. einfach realisiert werden.
- Es kann ein **Sicherheitskonzept** implementiert werden. Bei jeder Nachricht sollte geprüft werden, ob der Sender der Nachricht authentifiziert und autorisiert ist, diese Nachricht zu senden.

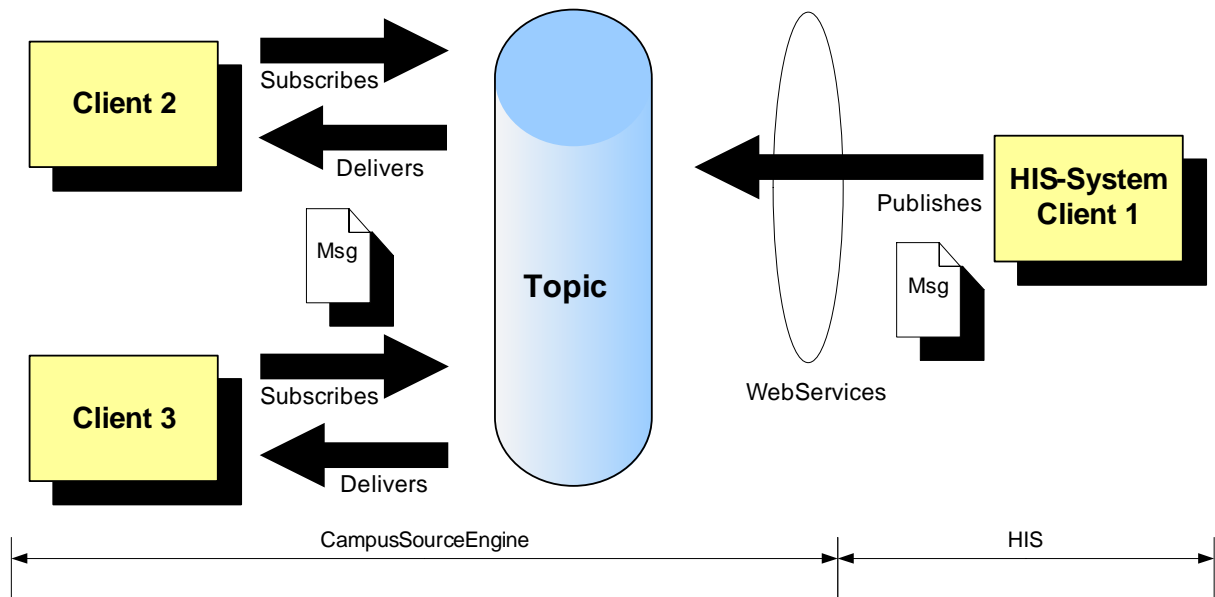


Abbildung 10: Ereignisbasierte Schnittstelle CSE-HIS

Abbildung 10 (ursprüngliche Quelle [SUN]) stellt den in der *Java 2 Enterprise Edition (J2EE)* definierten *Standard Java Message Service (JMS)* als Prinzip für die Kopplung von HIS an die *CampusSourceEngine* dar. In der Abbildung stellt der Client 1 ein HIS-Modul dar. Client 2 und 3 sind Applikationen, entweder in der *CampusSourceEngine* oder in einer originären Plattform.

Es sind zwei Anwendungsfälle zu unterscheiden:

- Nachricht vom HIS-Modul:
Client 1 (HIS-Modul) generiert eine Nachricht. Die Clients 2 und 3 sind an der Nachricht (*Message*) interessiert und haben sich an der Warteschlange (*topic*) angemeldet.
- Nachricht für das HIS-Modul:
Es kann mehrere Instanzen des Client 1 geben. Einziger Teilnehmer (*Subscriber*) ist Client 1.

Festzuhalten bleibt, dass die Nachrichtenwarteschlange (*topic*) innerhalb der *CampusSourceEngine* angesiedelt wird und die Schnittstelle mindestens zu den HIS-Modulen nachrichtenbasiert stattfinden wird.

5 Gemeinsame Basisdienste

Die von den einzelnen Hochschulen zu bewältigenden Problemstellungen sind in wesentlichen Bereichen vergleichbar und gruppieren sich häufig um die Grundfunktionalitäten einer Lernplattform herum. Exemplarisch seien die folgenden Punkte genannt:

- Personalisierter Zugang zum System und Authentifizierung der Nutzer gegenüber etablierten Nutzerdatenbanken (z. B. HIS an bundesdeutschen Hochschulen), auch mit dem Ziel eines universitätsweiten *Single Sign-On*.
- Entwicklung und Austausch nach Möglichkeit standardisierter Inhalte (Content) für Lehre und Forschung, Anbindung externer Publikationsserver unter Verwendung standardisierter Protokolle.
- Bereitstellung von Schnittstellen zur Integration möglicherweise externer Dienste wie z. B. einer webbasierten Literaturrecherche in den Beständen der Universitätsbibliothek.
- Unterstützung von Studienkontenmodellen wie z. B. ECTS (*European Credit Transfer and Accumulation System*).

Der modulare Aufbau eines Systems wird im Kontext objektorientierter Softwareentwicklung durch Architekturschemata ermöglicht, deren Bausteine von einander unabhängige und wiederverwendbare Komponenten sind, die Organisation der Plattform in einzelne Komponenten stellt eine Grundlage für deren Erweiterbarkeit und Anpassbarkeit dar. Komponenten zeichnen sich durch die folgenden charakteristischen Eigenschaften aus:

- In einem objektorientierten Softwareentwurf beschreiben fachliche Komponenten die einzelnen funktionalen Module einer Lernumgebung (z. B. Authentifizierung / Autorisierung, Anbindung von Content Repositories, usw.). Solche Geschäftskomponenten können sich ihrerseits auf technischer Ebene aus mehreren Softwarekomponenten zusammensetzen, die auf Ebene der Module einer Lernplattform somit nicht unbedingt über eine unmittelbare Entsprechung verfügen.
- Komponenten gliedern sich in eine Rahmenarchitektur ein und verfügen über wohldefinierte öffentliche Schnittstellen, sie sind zueinander interoperabel. Die öffentliche Schnittstelle einer Komponente stellt den alleinigen Zugang zu den von ihr angebotenen Funktionalitäten dar.
- Eine Komponente kann verschiedene Sichten bzw. Zugriffsmöglichkeiten auf die von ihr zur Verfügung gestellten Funktionalitäten anbieten, hierunter fallen insbesondere – aber nicht ausschließlich – verschiedene Formen von Benutzerschnittstellen (z. B. Web-Oberflächen, Zugang über mobilfunkbasierte Dienste wie SMS, WAP).

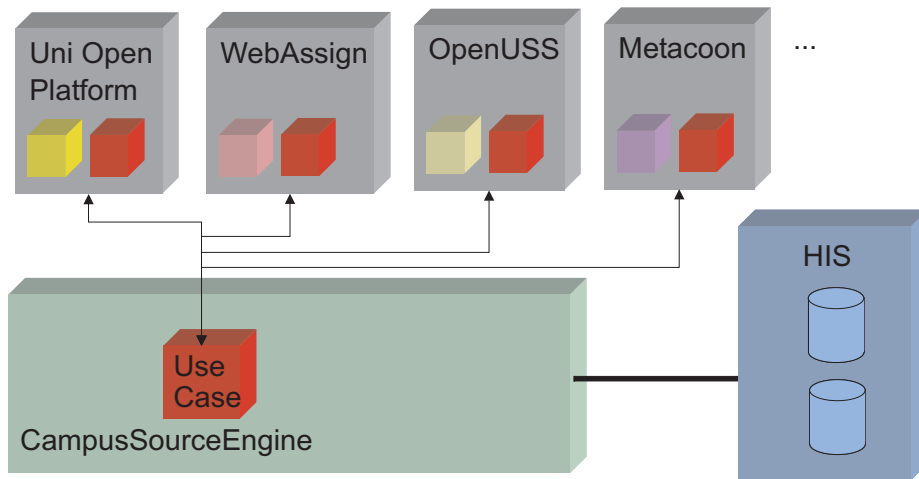


Abbildung 11: Ziel *CampusSourceEngine* mit Plattformen und HIS

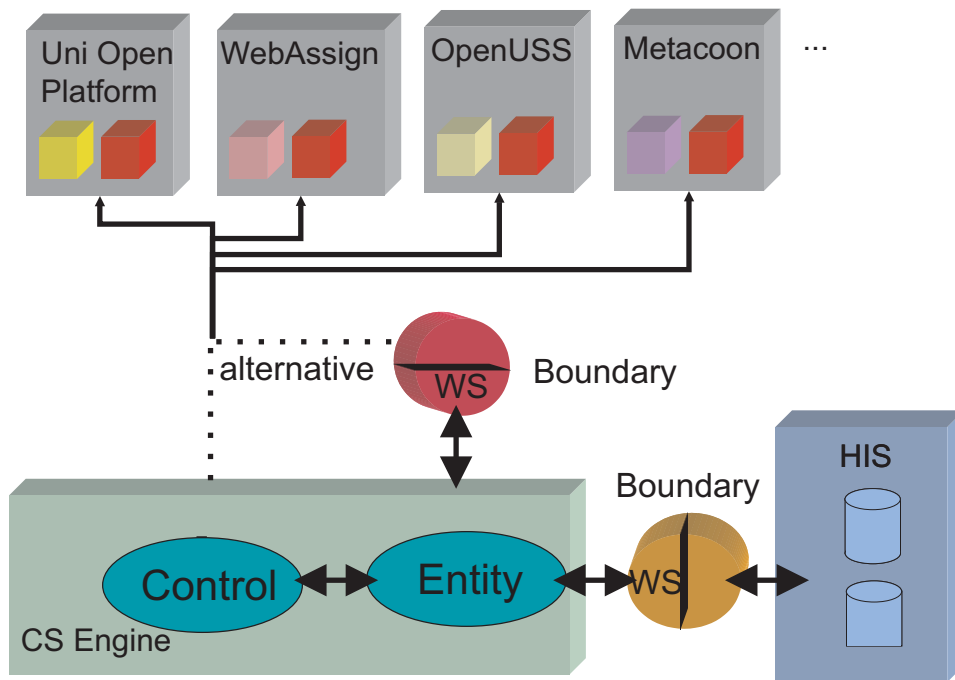


Abbildung 12: *CampusSourceEngine* mit den Modulgrenzen
(WS: WebServices - Boundary: Modulgrenze)

Die Abbildung 12 zeigt die Schnittstellen zu anderen Softwaresystemen (Boundary). Die Implementierung kann als WebServices gestaltet werden. Es werden Nachrichten im XML-Format zwischen den beteiligten Instanzen ausgetauscht.

Die speziellen Funktionalitäten der *e-Learning*-Plattformen können alternativ direkt auf Funktionalitäten der CSE oder aber über WebServices angekoppelt werden.

Welcher Zugriff verwendet wird, ist in dem jeweiligen Implementierungskonzept unter Laufzeitaspekten sorgfältig abzuwägen.

5.1 Datenquellen

Die Beschreibung der Datenquellen der HIS-Module entstammt [HIS].

5.1.1 HIS-LSF: Lehre, Studium, Forschung

LSF ist eine Web-Anwendung für Lehre, Studium und Forschung.

Das Modul LSF bietet Funktionen für die Erfassung und die Präsentationen von Lehrveranstaltungen und Forschungsprojekten und den damit verbundenen Ressourcen (Einrichtungen, Personen, Räume). Das Belegen von Veranstaltungen (insbesondere auch teilnahmebeschränkte Veranstaltungen) ist möglich.

LSF dient als Studieninformations-, Studienberatungs- und Planungssystem, so dass verschiedene Nutzerkreise (Studierende, Lehrpersonal, Administratoren, Raumverwalter) bei ihren spezifischen Planungen effektiv unterstützt werden. Studierende können durch die Integration der HIS-GX Module SOS und POS ihr Studium einfacher planen, weil auf Studien- und Prüfungsinformationen zurückgegriffen wird.

LSF ist eine rein webbasierte Anwendung, sämtliche Nutzer- und Administrationsfunktionen sind über einen Web-Browser zugänglich. Für die Realisierung wird in hohem Maße auf Produkte aus dem Open Source-Bereich zurückgegriffen.

LSF wird als Portal für Selbstbedienungsfunktionen genutzt, um den unterschiedlichen Nutzern die jeweils relevanten Funktionen anzubieten. Studierenden wird beispielsweise die Prüfungsanmeldung sowie eine Adressänderungsfunktion zur Verfügung gestellt. Lehrende können ihre Veranstaltungen bearbeiten oder Prüfungsergebnisse verbuchen, eventuell auch auf Haushaltsdaten zurückgreifen.

Das Modul LSF bietet Schnittstellen auf XML-Basis, um Daten aus den HIS-GX Modulen SOS, POS, SVA und BAU zu importieren. Ebenso können Fremdsysteme mit Hilfe dieser Schnittstellen angebunden werden.

5.1.2 HIS-QIS: Selbstbedienungsfunktion

QIS ist ein Bündel aufeinander abgestimmter Softwaremodule, die es berechtigten Dritten mit Internet- oder Intranetzugang erlauben, ausgewählte Funktionen der HIS-Verwaltungssoftware zu nutzen ("Selbstbedienung").

Die über QIS zugänglich gemachten Funktionen wie Semesterrückmeldung, Adressänderung, Ausdruck von Bescheinigungen, Prüfungsanmeldung, Notenverbuchung und Prüfungsstatistik, Materialanforderungen und Anzeige von Kontoständen können sowohl von dedizierten Service-Terminals, die auf dem Unicampus aufgestellt und in das Verwaltungs-LAN eingebunden sind, als auch über das Internet mit einem der marktgängigen WWW-Browser aufgerufen werden.

Je nach Anwendungsgebiet ist QIS in verschiedene Module aufgeteilt:

- QIS Modul FSV: Finanz- und Sachmittelverwaltung
- QIS Modul POS: Prüfungsverwaltung (Anwendungen für Prüfer und Studenten)
- QIS Modul SOS: Studentenverwaltung
- QIS Modul ZUL: Internet-Einschreibung

QIS Modul FSV zielt auf dezentrale Einrichtungen, die auf diesem Wege an das in der Zentralen Verwaltung betriebene Finanz- und Sachmittelverwaltungssystem FSV angeschlossen werden können.

Die in QIS bereitgestellten Funktionen können nach Wahl der nutzenden Hochschulverwaltung frei zusammengestellt und mit verschiedenen Sicherheitstechniken (Authentifizierung, Verschlüsselung) kombiniert werden.

5.1.3 HIS-SOS: Studentenverwaltung

SOS ist das Modul für die Studierendenverwaltung in der HIS-GX-Familie. Es ist integriert mit dem Prüfungsverwaltungsmodul POS, dem Zulassungsmodul ZUL und dem Lehrveranstaltungsmodul LSF. Es unterstützt alle Verwaltungsvorgänge der Studierendenverwaltung an deutschen Hochschulen und wird laufend an aktuelle Entwicklungen angepasst.

SOS ermöglicht u.a.:

- eine zeitgemäße, rationelle und effektive Abwicklung von Massen- und Einzelvorgängen wie Einschreibung, Rückmeldung, Fachwechsel, Wiedervorlage, Beitragskontrolle, Exmatrikulation, Beurlaubung, Ausweis- und Bescheinigungserstellung und Belegung
- Registrierung von Vor-, Zwischen- und Abschlussprüfungen
- Verwaltung von Chipkarten
- Automatische Vergabe von E-Mail-Adressen, Login-Kennungen und Passwörtern für die Studierenden
- Zentrale Erfassung und Fortschreibung der von verschiedenen Stellen innerhalb und außerhalb der Hochschule benötigten Studenten- und Absolventendaten
- Unterstützung von Studienkonten- und Studienguthaben-Modellen
- Schnittstellen zu zentralen Nutzerverzeichnissen
- Lieferung der Daten für die Amtliche Statistik (inkl. PL des Stat. Bundesamtes)
- Webbasierte Selbstbedienungsfunktionen (QIS-SOS)

Ein Datenträgeraustausch mit Kreditinstituten, Krankenkassen, statistischen Landesämtern und der ZVS ist realisiert. SOS enthält Schnittstellen zu MS-Office.

5.1.4 HIS-ZUL: Zulassung und Bewerbung

ZUL ermöglicht:

- Erfassung und Änderung von Bewerberdaten (Name, Anschrift) und der Antragsdaten (Studiengang); Haupt- und Hilfsantragsbearbeitung möglich
- Automatisierte Wartezeitberechnung, Erfassung von Sonderanträgen (z. B. Härtefall, Nachteilsausgleich)
- Unterstützung hochschuleigener Auswahlverfahren (Kriterien je Studiengang konfigurierbar)
- Erfassung der NC-Studiengänge (Kapazität, Überbuchung)
- Einrichtung des Hauptverfahrens (Quoten) entsprechend den Landesgesetzen
- Definition der Auswahl- und Sortierkriterien
- Erstellung von Ranglisten und Durchführung des Hauptverfahrens
- Verbuchung des Annahmeverhaltens
- Erzeugung der Bescheide über ISY
- Verarbeitung der ZVS-Daten über BSOS
- Verarbeitung von ASSIST-Bewerbern

- Online-Bewerbung über QISZUL

Schnittstellen zu SOS, ASSIST, ZVS und MS-Office

5.1.5 HIS-POS: Prüfungsverwaltung

Das Prüfungsverwaltungsmodul ermöglicht:

- Prüfungsanmeldung
- Kontrolle der Vorleistungen
- Erfassung der Prüfungsergebnisse
- Automatische Berechnung von Zwischen- und Gesamtnoten
- Erstellung von Zeugnissen, Diploma Supplement
- Erstellung von Bescheinigungen und Listen
- Erstellung von Prüfungsstundenplänen
- Verwaltung von Praktika und Praktikumsstellen
- Aufbereitung und Auswertung der Prüfungsdaten
- Lieferung der Prüfungsdaten an die amtliche Statistik
- Erfassung und Fortschreibung der Prüfungsordnungsdaten, insbesondere auch mit Credit-Point-Systemen
- Webbasierte Selbstbedienungsfunktionen (z. B. Anmeldung zu Prüfungen, Ergebnisübersichten, Notenverbuchung, Studienberatung) für Studierende und Prüfer mit dem Zusatzmodul QIS-POS

POS ist integriert mit dem Studentenverwaltungsmodul SOS, ISY und mit der Web-Anwendung für Lehre, Studium und Forschung LSF. Schnittstellen zu MS-Office sind realisiert.

5.1.6 HIS-SVA: Personal- und Stellenverwaltung

Das Modul SVA ermöglicht eine transparente Personal- und Stellenverwaltung. Dazu gehören:

- Einrichten von Stellen
- Stellenbewirtschaftung (Überwachung von freien, teilbesetzten und besetzten Stellen)
- Haushaltsplanmäßige Überwachung der Stellen (Haushaltsvermerke)
- Einstellen, Betreuen und Ausscheiden von Beschäftigten, Terminüberwachung
- Personalvorgänge zur Abwicklung von Verträgen mit Lehrbeauftragten oder mit aus Drittmitteln finanziertem Personal sowie Hilfskräften, Personalkostenkalkulation
- Hochschulstatistik, landesspezifische Statistiken, Schwerbehindertenstatistikdaten, Ausgleichsabgabe, Berechnung des Jahresurlaubsanspruchs
- Variable Listenauswertungen mit ISY

Schnittstellen zu MS-Office, Crystal Reports, statistischen Landesämtern, Finanzbehörde Hamburg, Systeme der Länder zur Berechnung von Bezügen, REHADAT-Elan, Zeiterfassungshardware.

Die Module COB, FSV, MBS, RKA sind in SVA integriert. LSF nutzt die Personaldaten aus SVA.

5.1.7 Andere

Quelle für Strukturdaten (derzeit HISCOB /MBS)

Quelle für Ressourcendaten

BuiSy

Telefonanlagen

Benutzerverwaltung Bibliotheken

Für Ressourcendaten (Computer, Netzwerkkomponenten, ...)

5.2 Schnittstellen

5.2.1 WebServices

Das zum gegenwärtigen Zeitpunkt populärste Konzept zur Realisierung dienstbasierter Systeme stellen die *Web Services* dar. Bei einem *Web Service* handelt es sich um eine über standardisierte Protokolle (z. B. HTTP, SMTP) in einem Netz erreichbare Applikation. Deren Schnittstellen können unabhängig von der zu Grunde liegenden Implementierung unter Verwendung einer XML-basierten, standardisierten Sprache beschrieben werden und zur Interaktion mit anderen Applikationen ebenfalls auf ein sprach- sowie plattform-unabhängiges, standardisiertes XML-Protokoll zurückgreifen.

Web Services werden charakterisiert durch (vgl. [CHA]):

- Aufbau auf XML
Sämtliche für Web Services relevanten Daten werden in Form wohldefinierter XML-Fragmente repräsentiert. Insbesondere kommt kein proprietäres (Binär-)Format zum Einsatz, was die den WebServices zu eigene Plattformunabhängigkeit ggf. einschränken würde.
- Geringe Kopplung
Zwischen Client und Webservice besteht eine geringe Kopplung, was eine weitestgehend unabhängige Fortentwicklung beider Komponenten ermöglicht sowie die Möglichkeiten zu deren Wiederverwendung und Wartbarkeit verbessert.
- Geringe Granularität auf der Ebene öffentlicher Schnittstellen
WebServices stellen Clienten eine Schnittstelle zur Verfügung, die von spezifischen Details ihrer Implementierung abstrahiert. In gewissem Sinne realisieren sie eine Fassade für die zu Grunde liegende, komplexe Geschäftslogik.
- Unterstützung synchroner oder asynchroner Operationsmodi
Clienten können synchron oder asynchron mit einem Webservice kommunizieren.
- Unterstützung von Remote Procedure Calls (RPCs)
WebServices stellen einen allgemeinen Mechanismus für RPCs zur Verfügung und vereinheitlichen so beispielsweise den Zugriff auf EJB- oder .NET-Komponenten.
- Möglichkeit zum Austausch von Dokumenten
In XML codierte Dokumente können zwischen WebServices ausgetauscht werden.

5.2.2 Funktionale/prozedurale Schnittstellen

Funktionale beziehungsweise prozedurale Schnittstellen bieten den Vorteil, dass ein Modul direkt ausführbaren Code des anderen Moduls ausführen kann. Damit ist ein erheblicher Geschwindigkeitsvorteil gegenüber den höher abstrahierten Schnittstellenmethoden gegeben.

Nachteile ergeben sich aus der Betriebssystemabhängigkeit, Programmiersprachenabhängigkeit und des erhöhten Aufwands für sicherheitstechnische Abfragen.

5.2.3 Bewertung

WebServices sind geeignet für verteilte Anwendungen. Für die Verwendung müssen XML-Objekte erstellt werden (Erstellung, Versenden, Parsen).

Die funktionale Schnittstelle liefert gleichen Funktionsumfang, aber ohne den aufwändigen Umweg der XML-Objekt-Erstellung, versenden und Dekodierung. Aber, sie ist nicht ohne weitere Mechanismen (CORBA, DCOM) in verteilten Umgebungen einsetzbar.

5.3 Dokumentation / Meta-Information

Die zu erstellenden Teilspezifikationen der Schnittstelle beinhalten:

- Eine Beschreibung der Funktion der auszutauschenden Daten
Es wird schematisch beschrieben, welche Aufgaben der jeweilige Dienst erledigt. Und welche Daten er prinzipiell dafür benötigt oder erstellt.
- Die Daten werden zu Entitäten zusammengefasst.
Die Entitäten werden in ihre Attribute aufgeschlüsselt. Die Funktion jedes Attribut wird kurz beschrieben, der Datentypen und eventuell die Größe spezifiziert.
- Beziehungen: Kardinalität und Assoziationstypen
- Datenfluss (lesen/schreiben)
- Die Art der Schnittstelle (Datenbanktabelle, Webservice, EJB etc.)
- Sicherheitsstufe (z. B. personenbezogene Daten)
Teile der Daten, die im HIS-GX liegen unterliegen dem Datenschutz. Jede Entität, für die das zutrifft muss den Umgang mit den Daten vorschreiben. Dies beinhaltet die Sicherheitsmechanismen, die für die jeweilige Entität adäquat ist.
Gleiches gilt für Daten, die den Verlauf einer Prüfung, das Erlangen eines studienrelevanten Abschlusses etc. beinhalten.
- Dienste
Sind zum Datenaustausch Methodenaufrufe notwendig?
Müssen Events implementiert werden?
Welchen Ablauf haben die Aufrufe? (Sequenzschema)
- Welches HIS-GX Modul ist für die Erfüllung der Schnittstelle nötig?

6 Zusammenfassung

In diesem Bericht „Schnittstellenspezifikation *CampusSourceEngine* und HIS“ wurden erste Anforderungen und Konzepte entwickelt. Realisierungsmöglichkeiten mittels *Model Driven Architecture* für den Implementierungsprozess und nachrichtenbasierten Schnittstellen für den Datenaustausch wurden vorgestellt.

Die nächste Phase wird nun zweigeteilt parallel angegangen. Zum einen wird die *CampusSourceEngine* an sich entwickelt und zum anderen wird die konkrete Ausgestaltung der Nachrichten vorbereitet. Der Vorteil der hier erarbeiteten Konzepte ist, dass keine programmiersprachenspezifischen Abhängigkeiten eingebracht werden.

Literatur

- [BHM] Baumgartner, Häfele, Meier-Häfele: Evaluation von Lernplattformen: Verfahren, Ergebnisse und Empfehlungen, 2004,
http://www.campussource.de/opensource/docs/evaluation_lernplattf_at.pdf
- [CHA] Chappell & Jewell (2002), pp. 6-7
- [HIS] <http://www.his.de>
- [SUN] <http://java.sun.com/j2ee/1.4/docs/tutorial/doc/images/jms-publishSubscribe.gif>