

# Architekturschema für VU-Systeme

H.-W. Six, J. Voss (Hagen), W. Schäfer (Paderborn)

## 1 Einleitung

Ein VU-System, das eine Virtuelle Universität implementiert, ist ein komplexes Softwaresystem, das aus verschiedenen Komponenten zusammengesetzt ist. Die Komponenten unterstützen verschiedene Funktionsbereiche der VU, wie z.B. die Bereitstellung von Lehr-/Lern-Material, Informationen aller Art, Arbeitsumgebungen für Lehrende und Lernende, administrative Tätigkeiten (z.B. Einschreiben, Belegen, Organisation von Klausuren und Prüfungen) und Kommunikations- und Kollaborationsdienste für alle beteiligten Gruppen.

In den Universitäten existiert bereits eine Reihe von Systemen, die verschiedene Funktionsbereiche einer VU implementieren. Sie sind entweder von Grund auf neu entwickelt oder Weiterentwicklungen von kommerziellen Produkten und seltener auch rein kommerzielle Produkte. Könnten diese Teillösungen geeignet kombiniert werden, ließen sich weite Bereiche einer VU abdecken. Dem stehen derzeit allerdings technische Inkompatibilität (z.B. fehlende technische Standardisierung der Schnittstellen) und überlappende Funktionalität (fehlende fachliche Abgrenzung der Komponenten) entgegen.

Um ein Zusammenspiel der Teilsysteme bzw. Komponenten zu erreichen, müssen deren Funktionalität und Schnittstellen aufeinander abgestimmt sein. Die konsequente Fortführung dieses Gedankens mündet in ein Architekturschema für VU-Systeme, welches die Entwickler von VU-Komponenten in die Lage versetzt, neue oder verbesserte Komponenten so zu entwickeln, dass sie leicht in bestehende VU-Infrastrukturen integriert werden können. Erste, noch wenig konkrete Überlegungen zu einer derartigen Rahmenarchitektur finden sich z.B. bei F. Milius (Information Management & Consulting 14; 1 (1999)). Ein geeignetes Architekturschema für VU-Systeme inklusive Schnittstellenstandards stellt daher eine zentrale technische Voraussetzung für den Erfolg der Open Source Initiative dar.

## 2 VU-spezifische Anforderungen an ein Architekturschema

Wir diskutieren nur Anforderungen, die sich aus der VU-Domäne ableiten und nicht ohnehin für jede Softwarearchitektur gelten („Software Engineering Folklore“). Letztere Anforderungen umfassen z.B. Schichtenarchitektur inkl. Trennung der Benutzungsschnittstelle von der Anwendung, Modularität, Kapselung und Geheimnisprinzip, starke Kohäsion, schwache Kopplung, Skalierbarkeit, Verwendung von Standards usw.

### 2.1 Kombinierbarkeit von VU-Komponenten

Fachbereiche und Fächer verwenden spezifische Lehrveranstaltungen, Lehr-/Lernszenarien und organisatorische Abläufe, die jeweils von einer oder mehreren VU-Komponenten unterstützt werden müssen. Hieraus leitet sich ein Architekturschema ab, das die einfache Kombinierbarkeit von VU-Komponenten erlaubt, so dass diese leicht zu einer individuellen Arbeitsumgebung zusammengesteckt werden können.

## 2.2 Adaptierbarkeit der Oberfläche

Fachbereiche und Fächer unterscheiden sich nicht nur in ihren Lehr-/Lernformen, sondern auch bezüglich spezifischer ästhetischer Vorstellungen und Gewohnheiten, die zum Teil in der Fachtradition begründet sind, zum Teil aber auch auf individuelle Präferenzen der Lehrenden zurückgehen. Für die Akzeptanz eines VU-Systems ist es daher wichtig, anpassbare Oberflächen anzubieten, um den unterschiedlichen Vorstellungen der Nutzer mit geringem Aufwand nachkommen zu können. Dabei müssen sich die Oberflächen der eingesetzten VU-Komponenten so einpassen lassen, dass jeweils das einheitliche Erscheinungsbild gewahrt bleibt (corporate identity).

## 2.3 Austauschbarkeit und Erweiterbarkeit

Der rasante technologische Fortschritt und die damit einhergehende Entwicklung neuer Lehr-/Lernszenarien werden zur Weiterentwicklung bzw. Ersetzung existierender VU-Komponenten und zu zusätzlichen VU-Komponenten für neuartige Funktionsbereiche führen. Derartige Komponentenersetzungen und Systemerweiterungen müssen mit möglichst geringen Auswirkungen auf die existierenden VU-Komponenten durchführbar sein.

## 2.4 Integration in die IT-Infrastruktur der Hochschule

Zur Arbeitserleichterung für Lehrende und Studierende sowie zur Redundanzvermeidung (und damit zur Konsistenzerhaltung zwecks Reduktion von Fehlerquellen) ist Voraussetzung, dass Daten, die an verschiedenen Stellen benötigt werden, nur einmal erfasst und nur an einer Stelle geändert werden müssen. Da alle wesentlichen Daten über Studierende, Lehrveranstaltungen, Prüfungen usw. bereits in der zentralen IT-Infrastruktur der Hochschulen gehalten werden, stellt die Benutzung dieser zentralen Datenbanken an Stelle einer redundanten lokalen Datenhaltung eine wichtige Anforderung an VU-Komponenten dar.

# 3 VU-Komponenten

VU-Komponenten verkapseln logisch zusammenhängende Funktionen eines VU-Systems und bieten sie über eine Web-Oberfläche an. VU-Komponenten unterstützen beispielsweise den Übungsbetrieb, die Erstellung und Bearbeitung von Lehrmaterialien, verschiedene Kommunikationsformen (synchron, asynchron, privat, öffentlich,...), Studentensekretariat, Prüfungsamt, Bibliothek und Shop. Eine VU-Komponente implementiert typischerweise in Kooperation mit anderen VU-Komponenten einen *Lehr-/Lernbereich*, beispielsweise eine komplette Kursumgebung mit Übungen, Kommunikationsangeboten, Klausur usw.

## 3.1 Architekturschema

Ein typische VU-Komponente weist softwaretechnisch eine klassische Dreischichtenarchitektur auf. Die unterste Schicht bildet die lokale Datenhaltung der Komponente. Darauf aufbauend folgt die eigentliche Anwendungslogik. Die oberste Schicht wird von der Web-Oberfläche der Komponente gebildet, die den Nutzern, insbesondere Studierenden und Dozenten, die Navigation und das Aufrufen der bereitgestellten Funktionen ermöglicht. Abb. 1 zeigt diesen Sachverhalt.

Die lokale persistente Datenhaltung einer VU-Komponente speichert z.B. persönliche Notizen der Studierenden, bei der Arbeit anfallende Zwischenergebnisse usw., um die persönliche Arbeitsumgebung eines Studierenden jederzeit wieder herstellen zu können. Eine Speicherung in

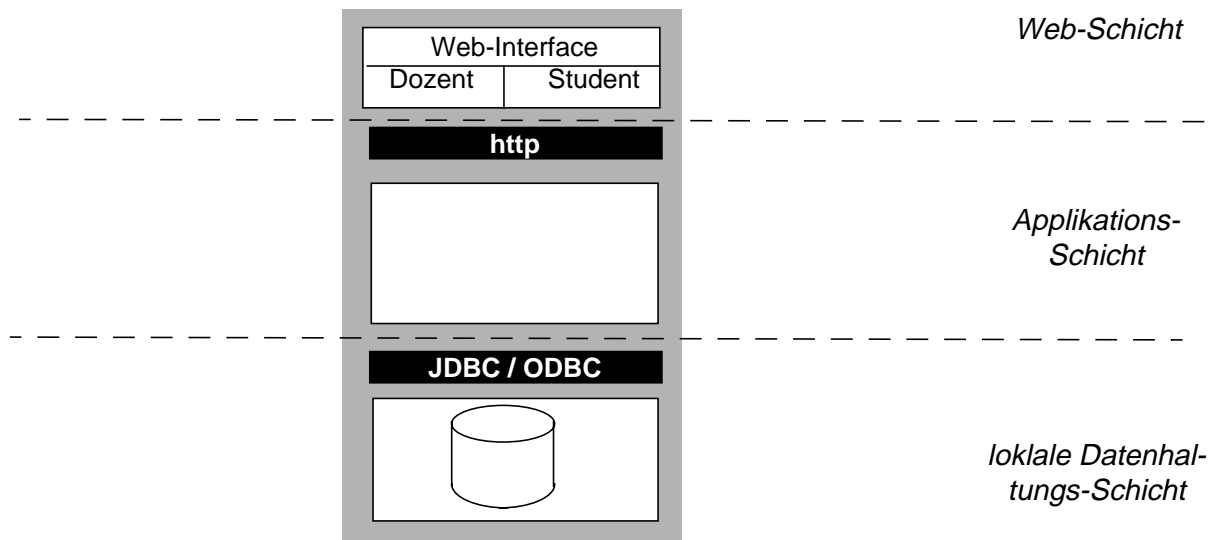


Abb. 1 Architekturschema einer VU-Komponente

zentralen Datenbanken ist für diese Art von Daten nicht angemessen. Sie würde insbesondere die Änderbarkeit und Austauschbarkeit der Komponenten erschweren. Trotzdem ist die Realisierung der lokalen Datenhaltung aller VU-Komponenten in einem gemeinsamen Datenbanksystem möglich und zur Erleichterung der Administration auch sinnvoll. Die Anbindung erfolgt über den ODBC-Standard bzw. bei Verwendung von Java über JDBC.

Allerdings muss nicht jede VU-Komponente über eine lokale Datenhaltung verfügen. So kann beispielsweise eine Lehrangebotsinformationskomponente alle erforderlichen Daten aus zentralen Datenbanken beschaffen.

### 3.2 Web-Oberfläche

Die Oberfläche einer VU-Komponente besteht aus einer Menge von untereinander verlinkten Web-Seiten. Üblicherweise sehen Web-Seiten *Einstiegspunkte* vor, die von anderen VU-Komponenten angesteuert werden können. Typische Beispiele hierfür sind:

- Ein oder mehrere personalisierte Einstiege in eine bestimmte Veranstaltung.
- Ein personalisierter Index über mehrere Veranstaltungen, z.B. ein Verzeichnis mit allen aktuell belegten Kursen eines Studierenden.

Die *Adaptierbarkeit der Oberfläche* stellt eine technische Herausforderung dar. Für eine genauere Betrachtung ist eine Unterscheidung der Web-Seiten in *Individualseiten*, die von den Benutzern erstellt werden, und *Systemseiten* sinnvoll. Systemseiten werden von der VU-Komponente generiert und stellen z.B. Navigationshilfen bereit oder bieten Funktionen an. In erster Linie müssen die von der VU-Komponente generierten Systemseiten anpassbar sein, die Individualseiten sind ohnehin frei gestaltbar. Insgesamt können für eine Web-Seite einer VU-Komponente abhängig vom Anwendungskontext verschiedene Adaptionvarianten sinnvoll sein:

- Wahl zwischen einer vom System generierten und einer vom Autor individuell gestalteten Seite (vgl. Informatik-VU).
- Individuell gestaltete Seite mit Variablen, die vom System gefüllt werden (vgl. WebAssign).
- Nach bestimmten Regeln frei „programmierte“ Seite (z.B. Java Server Pages).

## 4 VU-Systeme

Ein VU-System setzt sich aus kooperierenden VU-Komponenten zusammen, die sich auf zentralen Basisdiensten wie z.B. Directory Services, zentralen Datenbanken usw. abstützen. Abb. 2 zeigt schematisch die sich ergebende 4-schichtige Architektur.

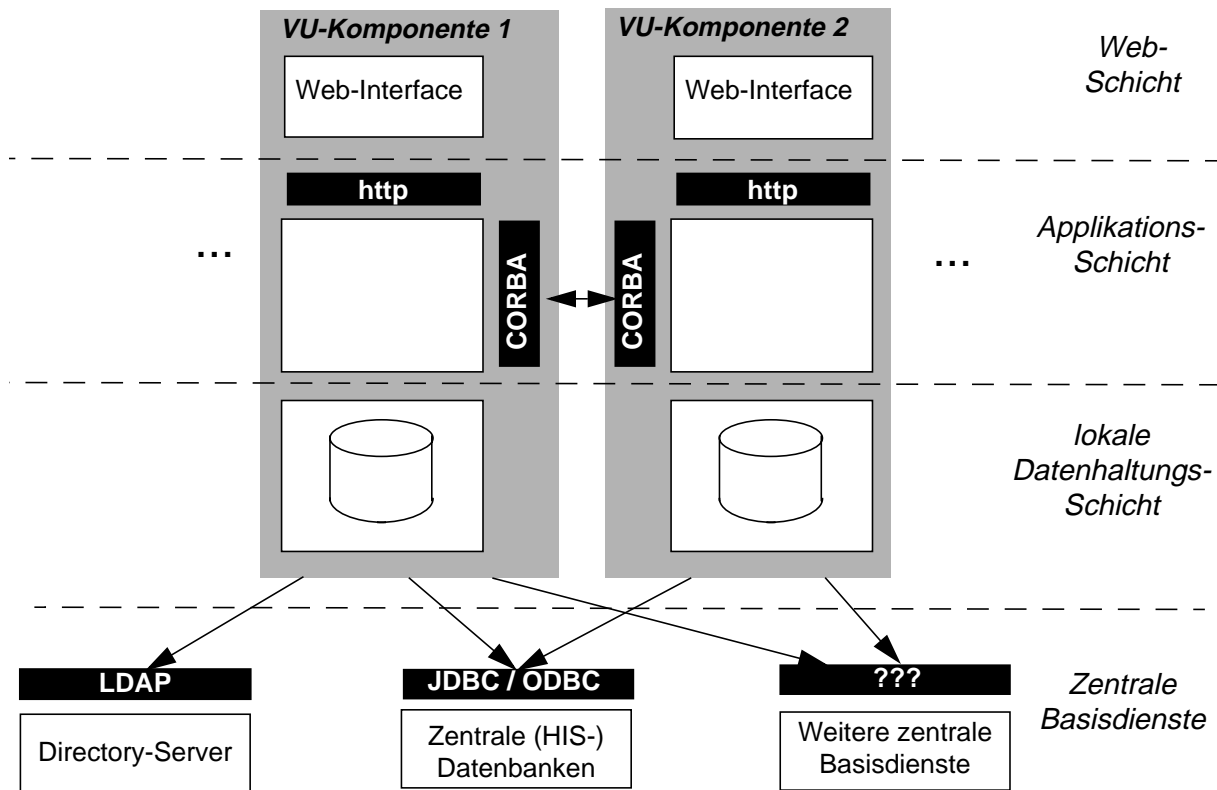


Abb. 2 Architekturschema für VU-Systeme

**Zentrale Basisdienste** sind nicht einer einzelnen VU-Komponente zuzurechnen, sondern stehen allen VU-Komponenten zur Verfügung. Hierunter fallen in erster Linie zentrale Datenbanken (HIS-Datenbanken) und Directory-Server, die z.B. Daten über Studierende (Stammdaten, Informationen über Belegungen, Studien- und Prüfungsleistungen, Passwörter) und Lehrveranstaltungen enthalten. Darüber hinaus kann man z.B. auch die Infrastruktur für elektronische Bezahlung zu diesen Basisdiensten zählen.

Zur **Kommunikation zwischen VU-Komponenten und Basisdiensten** werden die Standards LDAP (Directory-Server) und ODBC bzw. JDBC (zentrale Datenbanken) eingesetzt. Ggf. kann auch CORBA verwendet werden, ein weit verbreiteter Standard, der die Kommunikation zwischen (verteilten) Komponenten regelt und der in allen wichtigen Programmiersprachen implementiert ist. CORBA führt zu einer stärkeren Entkopplung der VU-Komponenten von den Basisdiensten, so dass die Auswirkung einer Änderung von Basisdiensten auf die VU-Komponenten verringert wird (z.B. wenn sich eine HIS-Datenbank ändert).

Die direkte **Kommunikation zwischen VU-Komponenten** stellt eine stärkere Integration als die Benutzung der unterliegenden zentralen Datenbanken dar. Situationen, in denen eine Kommunikation zwischen VU-Komponenten erforderlich wird, sind beispielsweise:

*Oberflächenintegration:* Eine VU-Komponente liefert die URLs seiner Einstiegspunkte, damit von anderen Komponenten aus dorthin navigiert werden kann, z.B. bei der Integration von VU-Komponenten zu Lehr-/ Lernbereichen.

*Administrationsunterstützung:* Eine VU-Komponente führt Administrationsfunktionen auf Veranlassung einer anderen VU-Komponente aus. Beispielsweise kann man beim Einrichten eines Kurses in der VU-Informatik veranlassen, dass automatisch eine entsprechende Übungsveranstaltung in WebAssign eingerichtet wird.

*Komponentenübergreifende Funktionen:* Manche Funktionen können nicht von einer Komponente allein ausgeführt werden. Beispiele sind:

- Die Klausuranmeldungskomponente fragt die in den Übungen erzielten Punktzahlen bei der Übungskomponente ab, um eine entsprechende Teilnahmevoraussetzung zu überprüfen.
- Die VU-Komponente „Persönliche Arbeitsumgebung“ stellt einen Index mit den zuletzt besuchten Seiten aus verschiedenen Komponenten zusammen.
- Ein offline-Reader lädt aktuelle Lehrmaterialien auf den PC des Studierenden herunter.

Technisch bietet sich CORBA für die Schnittstellen zwischen VU-Komponenten an. Für externe Komponenten, etwa einen Server beim Korrektor zuhause oder einen offline-Reader beim Studierenden, stellt HTTP eine zwar technisch weniger elegante, aber robuste und ebenfalls programmiersprachenunabhängige Alternative dar.

## 5 Lehr-/Lernbereiche

Typischerweise tragen mehrere VU-Komponenten zur Implementierung eines Lehr-/Lernbereichs bei. So wird ein virtuelles Seminar von VU-Komponenten unterstützt, welche Materialien bereitstellen, ein Diskussionsforum offerieren und Tools anbieten, in denen die Studierenden ihre Beiträge (ggfs. in Gruppenarbeit) erstellen. Die Integration dieser VU-Komponenten zu einer vollständigen und für alle Beteiligten (Studierende und Dozenten) leicht zu bedienenden Seminarumgebung ist eine wichtige Aufgabe, die auf verschiedene Weise realisiert werden kann. Zum einen kann die integrierte Umgebung (vom Dozenten) individuell von Hand gestaltet werden, eine Möglichkeit, die zwar größtmögliche Flexibilität bietet, aber nur von den wenigsten Dozenten geleistet werden kann. Zum anderen kann die integrierte Umgebung durch Konfiguration einer *Umgebungskomponente* erzeugt werden.

Umgebungskomponenten sind selbst wieder VU-Komponenten und besitzen wie diese eine Dreischichtenarchitektur. Über die Seiten ihrer Web-Oberfläche steuern sie die Navigation zu den Web-Seiten der integrierten VU-Komponenten. Sie bieten aber auch eigenständige, d.h. von den VU-Komponenten unabhängige Assistenz- und Integrationsfunktionen an:

- *Assistenzfunktionen* erlauben dem Dozenten, die Umgebung flexibel zu konfigurieren und veranlassen die erforderlichen Konfigurierungsmaßnahmen sowie die Generierung der für den Lehr-/ Lernbereich benötigten Web-Seiten.
- *Integrationsfunktionen* arbeiten analog, generieren die Web-Seiten aber nicht statisch, sondern dynamisch, womit flexibler auf aktuelle Situationen oder z.B. auf persönliche Vorlieben eines Studierenden reagiert werden kann.
- Assistenz- und Integrationsfunktionen können auch in andere VU-Komponenten integriert sein. Eine Lehrmaterialkomponente kann z.B. mit einer Übungs- und einer Kommunikationskomponente zusammengefasst sein und eine weitreichende Kursumgebung anbieten. Derartige große VU-Komponenten, die völlig unterschiedliche Funktionsbereiche bündeln, erschweren allerdings durch zu schwache Kohäsion die Kombinierbarkeit und Austauschbarkeit.

In der lokalen Datenhaltung kann eine Umgebungskomponente beispielsweise die persönliche Arbeitsumgebung eines Studierenden mit den zuletzt besuchten Seiten aus verschiedenen Komponenten speichern.

Abb. 3 zeigt das verfeinerte Architekturschema für VU-Systeme, das zwischen Umgebungskomponenten und „gewöhnlichen“ VU-Komponenten unterscheidet. Man erkennt, wie das Web-Interface einer Umgebungskomponenten zusammen mit den Web-Interfaces der integrierten VU-Komponenten die Web-Schnittstelle des zugehörigen Lehr-/Lernbereichs bildet.

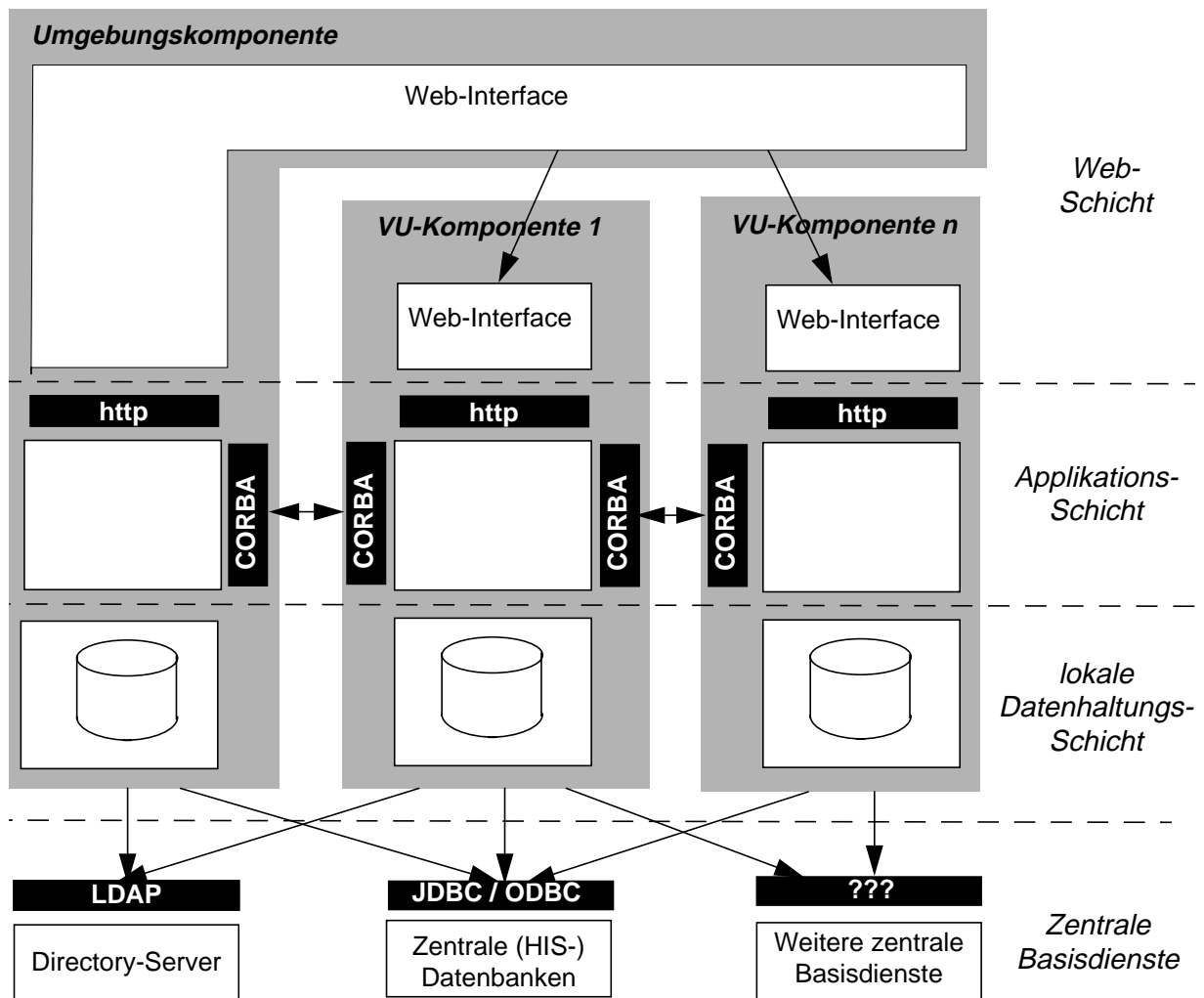


Abb. 3 Architekturschema für Lehr-/ Lernbereiche

## 6 Funktionaler Zuschnitt von VU-Komponenten

Für das Zusammenspiel von VU-Komponenten spielt neben Architekturschema und Schnittstellenstandards auch der funktionale Zuschnitt von VU-Komponenten eine wichtige Rolle. Außerdem benötigen Anwender Entscheidungshilfen, um geeignete VU-Komponenten für ihren Lehr-/ Lernbereich bzw. VU-System auszuwählen.

Um den funktionalen Zuschnitt von VU-Komponenten diskutieren und angebotene VU-Komponenten miteinander vergleichen zu können, ist es sinnvoll, sich zunächst einen Überblick über alle Funktionsbereiche einer virtuellen Universität zu verschaffen. Darauf aufbauend sol-

len Empfehlungen für den funktionalen Zuschnitt von VU-Komponenten und deren Schnittstellen entwickelt werden.

Nachfolgende Tabelle stellt einen ersten (unvollständigen) Versuch dar, VU-Funktionsbereiche zu identifizieren und zu charakterisieren.

		Beschreibung (aus student. Sicht)	Merkmale	Beispielkomponenten
<b>Lehrveranstaltungen</b>				
	Lehrmaterial	Durcharbeiten des Lehrmaterials	Strukturvorgaben, Formate, Freigabeverfahren, Autorenunterstützung	VU-Informatik, ET-Online, Ilias
	Übungen	Aufgaben lösen	Aufgabentypen, Korrekturmöglichkeiten, unterstützte Eingabeformate, Korrektorunterstützung	WebAssign, Ilias
	Labor	Experimente		„Reale Systeme im virtuellen Labor“
	Synchrone, veranstaltungsöffentliche Kommunikation	Vorlesung, Vortrag	mit Video, mit Audio, mit Folien, mit Aufzeichnung	VU-Informatik
	Asynchrone, veranstaltungsöffentliche Kommunikation	Diskussion		VU-Informatik, Ilias
	private Kommunikation			
	Forum für Gruppenarbeit		Zugriffsrechte	
<b>Administration</b>				
	Studentensekretariat	immatrikulieren, rückmelden, belegen		
	Prüfungsamt	für Prüfungen an-/abmelden		
	Organisation von (Präsenz-) Veranstaltungen	für Seminare, Praktika, Klausuren an-/abmelden		
	Raumverwaltung			
<b>Sonstiges</b>				

	Beschreibung (aus student. Sicht)	Merkmale	Beispielkomponenten
Bibliothek	Literaturecherche, Bücher reservieren, verlängern ...		
Shop	Zusatzmaterialien (CDs, Skripte usw.) kaufen		
öffentliche Information	Informationen über Lehrangebote, Prüfungsordnungen usw.		