6 Diskussion

Webbasierte Informationsquellen zum Zweck des Wissenserwerbs stehen dem medizinischen Anwender in einem vielfältigen Angebot zur Verfügung. Während anfänglich die Validierung der Internettechnologie von digitalisierten Lehrsammlungen im Focus wissenschaftlicher Publikationen [17, 45, 70] stand, erkannte man auch das effizienzsteigernde Potential von Zusammenarbeit [18, 56, 60] und integrativen Darstellungen verschiedenartiger Medien [63, 74, 91] in der *Browser*-Applikation. Weitere Vorteile der digitalen und über das Internet vernetzten Wissensbasen erschlossen sich über die Flexibilisierung der Applikationen, die durch Datenbank-*Retrievals* [79, 98] und den Einsatz systemübergreifender Datenstrukturstandards basierend auf XML [19, 28, 59, 93] ermöglicht wurden. Aktuellere Realisierungen adressieren die Zusammenstellung von Wissenselementen [24, 36] und die Integration [73, 78] in den individuellen Arbeitsfluss des medizinischen Anwenders, sowie die Adaptivität [92] des Lernsystems. Eine klassifizierende Untersuchung [72] von 262 webbasierten Online-Lernangeboten systematisierte die analysierten Systeme nach den Kriterien Sprache, Fachgebiet, Medieneinsatz, Zielgruppe und Interaktionstyp. Die Erhebung erfolgte aus spezialisierten Datenbanken [5, 22, 48, 49, 82-85] deutscher Universitäten und zentraler Einrichtungen für medizinische Fragestellungen. Das in dieser oben genannten Analyse ermittelte Vorkommen bestimmter Medien zur Wissensvermittlung in Lehrsystemen ergab, dass 1,5 % der untersuchten Systeme Geräusche, 3,4 % Sprache, 9,2 % Animationen und 12,6 % Videos beinhalteten. Das verdeutlicht das Bestreben, bei der Vermittlung grundlegender Inhalte die Vorteile aller verfügbarer Medien auszuschöpfen. In 89,7 % der in der Studie betrachteten Systeme waren Texte enthalten, gefolgt 64,5 % der Systemen mit Bildern und 42,4 % mit Grafiken. Diese Medienarten dominieren demnach bei der Vermittlung von Wissen und werden gezielt durch die übrigen Medien ergänzt. Die angezeigte Notwendigkeit der Verwendung multimedialer Daten zur effizienten Wissensvermittlung verleiht der Forderung nach Unterstützung verschiedenster Medien durch den zugrundeliegenden *Teaching-File-Server* Nachdruck. Die Unterscheidung des Interaktionstyps wurde hierbei einem methodischen Grundtypus [11] gleichgesetzt, die Benutzerinteraktion mit dem Lernsystem entsprach also den Möglichkeiten, die sich über eine HTML-Schnittstelle bei der Benutzung eines gängigen *Browsers* erzielen lassen und keine lokalen Installationen voraussetzen. Im Ergebnis wurden die Angebote zu über

86 % dem Interaktionstyp eines Präsentations- und Browsersystems zugeschrieben, auf die Zuordnung zu einem Simulationssystems entfielen circa 10 %, lediglich ungefähr 4 % der Angebote wurden als tutorielles System klassifiziert. Diese Verteilung unterstreicht die hauptsächliche Verbreitung von Wissenselementen, die vorrangig in singulären Wissensbasen vorgehalten werden. Damit zeugt sich deutlich das Potential der in dieser Arbeit vorgeschlagenen Öffnung dieser abgeschlossenen Wissensbasen mit dem Ziel diese Medienobjekte mittels Web-Service-basierten Interoperabilität zu komplexeren Wissensobjekten zu kombinieren. Die Simulations- und tutoriellen Lernsysteme sind in Deutschland häufig in Verbundprojekte deutscher Universitäten integriert, die über Mittel aus Länder- bzw. Bundesebene finanziert sind. Derartige Verbundprojekte, die einen ausdrücklichen Schwerpunkt in der medizinischen Lehre haben, unterstützen vorrangig das didaktische Konzept des Fall- bzw. Problemorientierten Lernens. Einige der zumindest bis Ende des Jahres 2003 aktiven Projekte charakterisiert ein Übersichtsartikel [46], der die Verbundprojekte *CAMPUS* [15], *LernIS-KHK* [43], *Docs´n-Drugs* [23], *k-MED* [44], *Prometheus* [69] beschreibt. Er thematisiert hierbei die Problematik der Nachhaltigkeit, der im Projektzeitraum erzielten Ergebnisse und Systeme, dass heißt die dem Projektziel zugeordnete Auswirkung im Projektumfeld oder Betrieb der Systeme über den Forschungszeitraum hinaus. Zumindest durch die lehrbetriebliche Einbindung des Systems *Docs´n-Drugs* an der Universität Ulm und *CAMPUS* an den Universitäten Heidelberg und Heilbronn ist diese wünschenswerte Nachhaltigkeit gelungen. Trotz der teilweise sich überschneidenden Ziele der einzelnen Projekte sind die in den jeweiligen Lernsystemen verfügbaren Lehrinhalte zueinander inkompatibel und stellen somit erneut, wenn auch mit komplexerem Inhalt, abgeschlossene Systeme dar. Zur Wahrung der Nachhaltigkeit der entwickelten Einzelsysteme und aus Gründen des Investitionsschutzes versucht ein weiteres Verbundprojekt, *Caseport* [16], mit einem Portal für fallorientierte Lehr- und Lernsysteme, die bisher entwickelten Systeme *CAMPUS, Casus, Docs´n-Drugs* und *D3-Trainer* über eine *XML*-basierte Datenschnittstelle zu integrieren, um zumindest die erzeugten Lehrmaterialien in einer neuen, einheitlichen Datenstruktur vorzuhalten. Eine grundlegende, durch die Wahl von Internet-Standards bereitgestellte Interoperabilität, wie in dieser Arbeit entwickelt wurde, ist jedoch nicht erkennbar.

Aufgrund der bis dato nicht vorhandenen *Web-Service*-basierten Interoperabilität bei aktuell existierenden radiologischen Wissensbasen, richtet sich die weitere Diskussion auf die technologischen Alternativen zur Herstellung der angestrebten Interoperabilität. Außerhalb des

medizinischen Kontextes, läuft seit April 2002 ein Pilotprojekt seitens des Betreiber der *WWW*-Suchmaschine „*Google*“ [32]. Die Ergebnisse des Pilotbetriebes [31] sind bisher noch unveröffentlicht, jedoch ist mit dem Angebot der Suchfunktion des Suchdienstes *Google* als *Web-Service* ist eine technologische und methodische Kompatibilität zu dem in dieser Arbeit vorgeschlagenen radiologischen *Web-Service* zu erwarten. Diese Entwicklung, eine breite kompatible Interoperabilität zwischen verschiedenen, auch fachlich unspezifischen *Web-Services* entstehen zu lassen, ist zum einen konzeptionell sinnvoll (vgl. Abbildung 6) und bereits zuvor als Methode für eine standardisierte Kopplung von Systemen [75], gefordert.

6.1 Interoperabilität

Die Relevanz eines automatisierbaren, internetbasierten-, plattform- und programmiersprachenunabhängigen Zugangs zu bestehenden Wissensbasen ist gekoppelt an die tatsächlich erreichbare Integrationsfähigkeit der Wissensobjekte in den individuellen Kontext [28, 29] der Applikationen. In diesem Zusammenhang ist es notwendig, dass die eingesetzte Technologie und der diensterbringende Server die Datenobjekte in einem vom Client wählbaren Detailierungsgrad anbietet, ohne jedoch den Kontext dieser Objekte zu verlieren. Hierzu stellt die programmtechnische Aufbereitung der Datenobjekte ein wesentliches Element dar, die unmittelbar über das *RPC*-Konzept dem *Client* nutzbar gemacht wird (vgl. Kapitel 4.1).
Die Ausgestaltung der Interoperabilität lässt sich über unterschiedliche Strategien realisieren, die im folgenden hinsichtlich ihrer Eignung für die in dieser Arbeit entwickelten, ohne zusätzliche Interaktion erfolgende Vernetzung zweier Kommunikationspunkte in einer *Client-Server*-Konstellation, diskutiert werden.

6.1.1 Shared Datapool

Die Strategie, Interoperabilität zwischen den beteiligten Applikationen über einen gemeinsamen Datenspeicher herzustellen, erfordert zur Implementierung eine zuvor festgelegte Einigung auf die Datenstruktur [16], die Adresse des gemeinsamen Datenspeichers und auf die Zugriffsmodalitäten auf den Datenspeicher. Aus diesen Anforderungen an die beteiligten Applikationen leitet sich ein starrer, im Vorfeld eindeutig definierter Konsens ab, der eine weitreichende Homogenität der Applikationen in der Struktur ihrer Datenhaltung enthält. Diese Strategie markiert die unterste Stufe in der Komplexität von Interoperabilität und bietet keine

technischen Möglichkeiten für eine auf funktionaler Ebene stattfindende Interoperabilität. Für den in dieser Arbeit gewählten Einsatzzweck ist diese Strategie daher ungenügend.

6.1.2 Bridging

Die Interoperabilität mittels Brückenelementen konzentriert die bereitgestellte Funktionalität auf die Homogenisierung von Datenstrukturen der beteiligten Applikationen mittels Konvertierung [96, 97]. Damit ist die beim *Shared Datapool* vollzogene Einigung auf eine gemeinsame Datenstruktur verlagert in eine zu implementierende Konverterfunktion. Eine Interoperabilität kann hierbei nur durch die Bereitstellung und Anpassung des Konverters erreicht werden.

6.1.3 Screen-Scraping

Eine weitere Strategie besteht im sogenannten programmtechnischen, automatisierten „Auskratzen“ von Daten aus den für den Menschen konzipierten Ausgaben einer Applikation, mit der auf technischer Ebene „kommuniziert“ werden soll. Diese Strategie der Datengewinnung wird nicht selten bei den sogenannten Meta-Suchmaschinen [68] eingesetzt. Der Erfolg dieser Datengewinnung setzt ein statisches Human-Computer-Interface (*HCI*) der Applikation voraus und muss stets individuell angepasst und programmiert werden. Diese Strategie ist für das Ziel dieser Arbeit, die automatisierte Integration von Daten in den Workflow der Anwender, lediglich in einem Aspekt von Interesse. Dieser besteht darin, dass alle potentiellen Datenquellen über eine web-basierte Benutzeroberfläche für Menschen verfügen. Nicht zuletzt aufgrund der stark unterschiedlichen Benutzerinteraktion und dem damit implizierten Aufwand in der Programmierung des jeweiligen *Screen-Scrapings*, ist dieses Konzept nur eingeschränkt für die angestrebte Interoperabilität geeignet.

6.1.4 Message-Passing

Die Strategie, alle an einer Interoperabilität beteiligten Partner mittels Nachrichten kommunizieren zu lassen ist durchaus umsetzbar, weil diese Form zunächst keinen Einschränkung der Komplexität von Informationen unterliegt. Die Spezialisierung erfolgt in einem zweiten Schritt, in dem die Grammatik und die verarbeitbaren Nachrichten definiert werden müssen. Über diese Festlegung der Definition müssen sich alle potentiellen Partner einer Interoperabilität informieren. Im Sinne einer nicht-diskriminierenden Dienstintegration sind jedoch

proprietäre Definitionen [71] abzulehnen. Ein exemplarisches Beispiel einer gelungenen Interoperabilität auf Basis von *Message-Passing* ist die Dienstintegration des *World-Wide-Web* (WWW) in Nutzung sogenannter *Browser*-Applikationen wie *Netscape-Navigator*, *Internet-Explorer* oder *Opera*, die mittels des *HTTP*-Protokolls (vgl. Kapitel 3.3.3) Nachrichten an den *Server* übertragen und damit die Elemente von HTML-Seiten anfordern.

6.1.5 Remote-Procedure-Call (RPC)

Eine auf *RPC* basierende Interoperabilität erfährt durch die aktive, programmtechnisch implementierte Logik der *Remote-Procedure (RP)* eine weitreichende Flexibilität. Einerseits ist die Schnittstelle durch den Definitionsrumpf der *Remote-Procedure* hinsichtlich der Parametrisierung spezifiziert und ermöglicht der *Client*-Seite, die Funktionalität datentypisch passend zu integrieren, in dem sie einen entsprechenden Aufruf (*Call*) der *Remote Procedure* initiiert. Andererseits besteht auf der *Server*-Seite die Möglichkeit, Dateninhalte und Datentypen entsprechend der beabsichtigten Funktionalität programmtechnisch aufzubereiten. Nicht zuletzt profitiert die Daten- und Übertragungssicherheit von der gegebenen Möglichkeit seitens der *Remote-Procedures*, Fehlermeldungen an den aufrufenden *Client* zu übermitteln. Die Eignung des *RPC*-Konzepts ist abhängig von der Spezifikation bezüglich des Transportprotokoll und der Datentypisierung. Grundsätzlich ließe sich *RPC* mit proprietären Protokollen und Rechnerarchitekturabhängigen Datentypen koppeln. Der Nachteil besteht dann allerdings auf einer Festlegung eines abgeschlossenen Kreises von partizipierenden Systemen.
Zielführend im Sinne der internetbasierten, von Herstellern unabhängigen Interoperabilität kann deswegen nur eine Einbettung in offen gelegte, lizenzfreie Protokolle und Datentypdefinitionen sein. Aus diesem Grund wurden in dieser Arbeit zur methodischen Entwicklung von Interoperabilität keine neuen Protokolle entworfen sondern konsequent die verfügbaren und offen standardisierten Protokolle genutzt. Wie die erzielten Ergebnisse unterstreichen, konnte durch Verwendung von *SOAP* das *RPC*-Konzept plattformübergreifend umgesetzt werden. In Verbindung mit der *WSDL*-Beschreibung konnte die konkrete *RPC*-Implementierung vollständig in *XML* definiert werden. Die in dieser Arbeit eingesetzte Kombination aus *WSDL* und dem *SOAP*-basierten *RPC*-Konzept, besitzt den Vorteil der vollständigen und eindeutigen Definition der verfügbaren Funktionen, deren Parameter und Rückgabewerte, der Datentypen von Parametern und Rückgabewerten und der Bindung an ein Transportprotokoll. Diese Voll-

ständigkeit der Definition ist eine Voraussetzung für die vollautomatische Analyse durch Applikationen, die eine Funktionalität mittels *RPC* integrieren sollen.

6.2 Technologien

Es existieren eine Reihe von Software-Technologien, um eine Interoperabilität in heterogenen Plattformumgebungen zu ermöglichen. Die in dieser Arbeit als Lösungsansatz herangezogene Methode der sogenannten *Web-Services* wird im folgenden vor dem Hintergrund der übrigen Technologien diskutiert werden.

6.2.1 Common-Object-Request-Broker-Architecture (CORBA)

CORBA, entwickelt von der *Object Management Group* (OMG) findet bis dato Verwendung [55] für die Interoperabilität von Systemen in heterogenen, vernetzten Plattformumgebungen und ist spezifiziert in [21]. Signifikantes Element dieser Architektur ist der *Object-Request-Broker (ORB)*, der die Funktion erfüllt, die von *Clients* initiierten *RPC*'s an die entsprechenden Server-Komponenten zu vermitteln. Voraussetzung ist eine zuvor erfolgte Registrierung der Server-Komponenten im *ORB* mittels einer in der programmiersprachenunabhängigen *Interface-Definition-Language (IDL)* verfassten Schnittstellendefinition, aus der auf Seite der *Clients* die Dienst-Verbindungsstücke, die sogenannten *Stubs* generiert werden. Die *CORBA*-Spezifikation wurde von verschiedenen Software-Herstellern kommerziell implementiert, exemplarisch sei die *IONA*-Implementierung genannt [39]. Die Kommerzialisierung der verfügbaren Implementierung der *CORBA*-Technologie führte in der Praxis zur Situation von Inkompatibilitäten zwischen *IDL*-Schnittstellendefinitionen und online verfügbaren *ORB´*s. Folglich konnte nur eine herstellerabhängige Kompatibilität zwischen *Client*- und *Server*-Komponenten sichergestellt werden. Diese Einschränkung wurde nur unzureichend durch die Spezifikation des *Internet-Inter-Object-Request-Broker-Protocol (IIOP)* auf Basis des *TCP*-Protokolls (vgl. Kapitel 3.3.2) aufgebrochen, welches mit dem Ziel definiert wurde, die Interoperabilität zwischen den *ORB´*s zu vereinheitlichen.

Vergleichend mit der *Web-Service*-Technologie besteht Übereinstimmung im Konzept der Nutzung von *RPC´*s und deren Schnittstellendefinition in Form einer von Programmiersprachen abstrahierenden Beschreibungssprache. Allerdings beschränkt sich die Beschreibung per *IDL* in *CORBA* tatsächlich auf die Schnittstelle einer *RPC*, während dies in der *WSDL*-

Beschreibung (vgl. Kapitel 3.6) lediglich ein Element der Spezifikation darstellt. Hinsichtlich der in dieser Arbeit entwickelter, spontaner Vernetzung zwischen *Clients* und *Servern* offenbart die *CORBA*-Technologie einen weiteren, konzeptionellen Nachteil: Zur Nutzung einer entfernten Funktionalität mittels *RPC* ist der *ORB* ein permanenter Vermittler und erfüllt bei jeder Nutzung prinzipiell die Verzeichnis-Funktionalität, die das *UDDI* (vgl. Kapitel 3.7.1) der *Web-Service*-Technologie darstellt. Der *ORB stellt* in *CORBA* eine Kommunikations-/ Vermittlungszentrale dar, die bei jeder Anfrage in ihrem Verzeichnis den passenden *Server* herausfindet und zu diesem durchstellt, während in der *Web-Service*-Technologie das *UDDI* die Funktion eines Branchen-Telefonbuchs darstellt, aus dem der *Client* die Kommunikationsadresse des *Servers* ausliest und zukünftig selbständig die Kommunikation zum Server bewerkstelligen kann. Die Metapher eines über eine Zentrale vermittelten Telefonats und einem auf Basis eines Telefonbuchs selbstgewählten Telefonats beschreibt die konzeptionellen Unterschiede sehr treffend.

6.2.2 Distributed-Component-Object-Model (DCOM)

DCOM [51] ist eine von der Firma Microsoft entwickelte Technologie und definiert die Kommunikation zwischen Software-Komponenten in verteilten Umgebungen. Obwohl programmiersprachenunabhängig angelegt, existiert keine einsetzbare Implementierung abseits der Windows-Plattform. Zudem schreibt DCOM ein festes Typsystem in der Kommunikation vor. Daraus resultiert die Notwendigkeit von Datentypkonvertierungen in das *DCOM*-Typsystem, teils auch bei simplen Datentypen wie beispielsweise Zeichenketten. Im Vergleich mit der *Web-Service*-Technologie macht die *XML-Schema* basierte Datentypisierung derartige Zwischenkonvertierungen überflüssig, die Daten werden direkt in das Typsystem der *Server*-Komponenten notiert, die von den *Client*-Komponenten aus dem entsprechenden Abschnitt der *WSDL*-Definition ausgelesen werden kann. Eingebettete Internetfunktionalitäten in Mobiltelefonen, *Personal-Digital-Assistents (PDA)* oder Navigationssystemen belegen die Notwendigkeit einer weitreichenden Plattformunabhängigkeit. Den hier entscheidenden Nachteil von *DCOM* erkennend, ist die Firma Microsoft inzwischen selbst eine der führenden Entwickler der *Web-Service*-Technologie und integriert diese in die sogenannte *Dot-Net-Strategie (.NET)[52]*.

6.2.3 Dot-Net (.NET)

Dot-Net (*.NET)* [52] ist eine Plattform, die von der Firma Microsoft entwickelt wurde. Zur Ausprägung einer Plattform zählen eine *.NET*-eigene Laufzeitschicht, eine sogenannte *Runtime*, die eine eigene Speicherverwaltung und Funktionsbibliothek besitzt. Ebenfalls in *.NET* ist eine spezielle Programmsprache definiert, die sogenannte *Intermediate-Language (IL)*. Die in *.NET* als *Common-Language-Runtime (CLR)* bezeichnete *Runtime*-Komponente dient zur Übersetzung des *IL*-Codes in Maschinencode der jeweiligen Hardwareplattform, der Verwaltung von Prozessen und nicht zuletzt zur Überprüfung von definierten Sicherheitsanforderungen. Mit *.NET* verbindet die Firma *Microsoft* eine Neuausrichtung ihrer strategischen Ziele im Bereich der Internettechnologie, um ihren Produkten eine möglichst flexible, plattformunabhängige Anbindung an das Internet zu verleihen. Die *Web-Service*-Technologie dient in *.NET* der Interoperabilität. Bezogen auf die in dieser Arbeit motivierte Interoperabilität mittels *Web-Services* zeigt die im Rahmen der Arbeit erfolgte Realisierung von Interoperabilität, mit der von *.NET* nicht unterstützten Programmiersprache *PHP*, die Unabhängigkeit der *Web-Service*-Technologie von den übrigen Bestandteilen der *.NET*-Technologie. Die herstellerunabhängige Realisierbarkeit von *Web-Services* ist damit belegt.