Universität Hamburg

Iver Jackewitz

Evolutionary Application Service Providing (eASP)

Ein Ansatz der Softwarebereitstellung

Hamburg University Press

Evolutionary Application Service Providing (eASP)

Ein Ansatz der Softwarebereitstellung

Iver Jackewitz

Universität Hamburg

Evolutionary Application Service Providing (eASP)

Ein Ansatz der Softwarebereitstellung

Iver Jackewitz

Hamburg University Press
Hamburg

Dissertation zur Erlangung der Würde eines Doktors der Naturwissenschaften
(Dr. rer. nat.) am Fachbereich Informatik der Universität Hamburg.

Betreuer: Prof. Dr. Arno Rolf,
Prof. Dr. Christiane Floyd
Gutachter: Prof. Dr. Norbert Ritter

Tag der Einreichung: 6. März 2005
Tag der Disputation: 29. August 2005

Bibliografische Information der Deutschen Bibliothek

Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.ddb.de abrufbar.

Diese Publikation ist außerdem auf den Webseiten des Verlags Hamburg University Press *open access* verfügbar unter http://hup.rrz.uni-hamburg.de

Die Deutsche Bibliothek hat die Netzpublikation archiviert. Diese ist dauerhaft auf dem Archivserver Der Deutschen Biliothek verfügbar unter http://deposit.ddb.de

ISBN 3-937816-22-4

Produktion: Elbe-Werkstätten GmbH, Hamburg
http://www.ew-gmbh.de

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Zusammenfassung

Diese Arbeit beschäftigt sich mit der Bereitstellung von Software und erarbeitet einen Ansatz zur Analyse und Gestaltung von Softwarebereitstellungskontexten, in denen zentral eine Software für viele Nutzende bereitgestellt wird. Die Legitimation, sich mit der Bereitstellung von Software in der bzw. für die Informatik zu beschäftigen, wird aus folgenden Feststellungen gewonnen: (1) „Formal korrekte“ Software kann im Anwendungskontext nicht immer den erhofften Nutzen hervorbringen. (2) Die Bereitstellung einer Software korreliert mit der Zufriedenheit der Nutzerinnen und Nutzer einer Software, d. h. die Bewertung einer Software(-version) hängt u. a. von der Qualität ihrer Bereitstellung ab.

Die Bereitstellung der webbasierten Kooperationsplattform CommSy in zwei unterschiedlichen Konstellationen, die im Sinne der Aktionsforschung begleitet wurden, unterstreicht die Notwendigkeit eines Softwarebereitstellungsansatzes zum Verstehen und Gestalten von Softwarebereitstellungskontexten. Während die Bereitstellung in der Fallstudie CommSy@Uni.de nicht sehr erfolgreich verlief, kann sie in der Fallstudie CommSy@WissPro als erfolgreich bewertet werden. Es stellt sich hier die Frage nach Unterschieden in den Fallstudien, die nur mit einem umfassenden Blick detailliert beantwortet werden kann. Die Fallstudien geben den Anlass, vier Perspektiven auf die Softwarebereitstellung einzunehmen: Aufgaben, Organisation, Technik und Vorgehen, welche anschließend grundlegend für den Ansatz zur Softwarebereitstellung sind.

Als weitere Grundlage für den Ansatz wird das Application Service Providing herangezogen, da es sich in der Wirtschaft als ein Geschäftsmodell der Softwarebereitstellung etabliert hat. Allerdings offenbart es Schwächen hinsichtlich Aufgaben, Organisation und des Vorgehens, so dass Ansätze und Aspekte aus dem Informatikfeld „Informatiksysteme in Organisationen“ speziell der „organisationsbezogenen Softwareentwicklung“ herangezogen werden, um diese Schwächen zu überwinden. Dadurch wird das Application Service Providing mit diesen Ansätzen fundiert und das

Geschäftsmodell zu einem Softwarebereitstellungsansatz aufgewertet. Der erarbeitete Ansatz zur Softwarebereitstellung, welcher die Softwarebereitstellung als eine evolutionäre (evolutionary) Bereitstellung (Providing) einer Software (Application) als Dienstleistung (Service) versteht, umfasst alle vier genannten Perspektiven und wird eASP für evolutionary Application Service Providing genannt. Die Namensgebung spiegelt zum einen die Verbundenheit zum bzw. die Gründung auf dem Application Service Providing und zum anderen das grundlegende Verständnis von Softwarebereitstellung wider.

Die analytische Anwendung von eASP auf die Fallstudien zeigt dessen praktische Verwendbarkeit und darüber hinaus Unterschiede, Schwierigkeiten und Stolpersteine in den beiden Fallstudien auf, so dass die Frage nach dem Scheitern bzw. Nicht-Scheitern differenziert für beide Fallstudien beantwortet werden kann.

Die Herausforderung dieser Arbeit liegt in der Fundierung und Erweiterung des Geschäftsmodells Application Service Providing zum Softwarebereitstellungsansatz eASP, um es für die Informatik nutzbar zu machen. Durch eASP können z. B. in der evolutionären und partizipativen Softwareentwicklung und der IT-unterstützten Organisationsgestaltung die Auswirkungen und Leistungen der Softwarebereitstellung berücksichtigt werden. Dabei wird eASP nicht als das Modell der Softwarebereitstellung verstanden, sondern als ein Muster, welches mit den vier Perspektiven Aufgaben, Organisation, Technik und Vorgehen und den darin enthaltenen Konzepten Möglichkeiten offenbart, eine konkrete Softwarebereitstellung zu verstehen und zu gestalten. Darüber hinaus werden Beispiele aufzeigt, die zur Inspiration in anderen Bereitstellungskontexten dienen und dem jeweiligen Kontext entsprechend angepasst werden müssen.

Abstract

This paper is concerned with the provision of software and the conception of an approach to analysis and formation of software provision contexts in which centralised software is installed for several users. The legitimation for an engagement in software production in informatics is founded on the following premises: (1) „formally correct“ software is not always capable of providing the necessary benefit in the context of usage. (2) the provision of software correlates with the level of satisfaction of the users of a particular software, so the valuation of a software (version) depends partly on the quality of the provision.

The provision of the web-based co-operations platform CommSy in two different constellations – accompanied in the sense of action research – stresses the need for a software provision approach for the comprehension and design of software provision contexts. While the provision in the case study CommSy@Uni.de was not entirely successful, in the case study CommSy@WissPro it was successful. The question here is „where are the differences?“ which can only be fully answered by a comprehensive inspection. Thus there differentiate from the case studies four perspectives towards the software provision: tasks, organisation, technique and procedure, which must be included in the approach to the software provision.

As a basis for the approach Application Service Providing is drawn upon, since it has established itself in industry as a business model of software provision. However, it reveals weaknesses with regard to the tasks, the organisation and the procedure, so that aspects of the informatics field „information systems in organisations“ particularly of „organisationally related software development“ are specifically drawn upon in order to overcome these weaknesses. Thus the Application Service Providing is substantiated by these approaches and the business model is revaluated as a software provision approach. The developed approach to the software provision which conceives software provision as an evolutionary provision of an application as a service includes all four mentioned perspectives and is

called eASP for evolutionary Application Service Providing. The nomenclature reflects on the one hand the connection to as well as the foundation on the Application Service Providing and on the other hand the fundamental understanding of software providing.

The analytical usage of eASP for the case studies exemplifies their practical usability and in addition reveals differences, difficulties and stumbling blocks in both case studies so that the question as to the failure or non-failure of each case study can be solved separately and in a differentiated manner.

The challenge this paper presents is comprised in the foundation and extension of the working model Application Service Providing into the software provision approach eASP in order to make it useful for informatics. With the aid of eASP, for example, the effects and achievements of the software provision in evolutionary and participatory software development can be taken into consideration. Hereby, eASP is not conceived as the model of software provision but as a model which – together with the four perspectives tasks, organization, technique and procedure as well as the concept contained in these – reveals the possibilities of comprehending and designing software provision and presents examples which serve as inspiration in other provisionary contexts and must correspondingly be adapted depending on the context.

- 1 -
Einleitung

Informationstechnik (IT) bzw. Informations- und Kommunikationstechnik (IuK) haben sich in der Wirtschaft, Wissenschaft und Gesellschaft etabliert. Insbesondere die Entwicklungen im Bereich der Internettechnologien führten zu vielfältigen Vernetzungsmöglichkeiten und netzwerkartigen Organisationsstrukturen. Als Konsequenz wurde die Netzwerkorganisation zum Leitbild unserer Gesellschaft zu Anfang des 3. Jahrtausends ausgerufen (Castells 2001). Parallel wird IT bzw. IuK zu einer Infrastruktur unserer Gesellschaft, wie der öffentliche Nahverkehr oder das (stationäre) Telefon, also zu einer Selbstverständlichkeit.

Bei Kommunikations- und Kooperationssoftware, als Teil der informationstechnischen Infrastruktur, wird zunehmend die Erfahrung gesammelt, dass das bloße Vorhandensein der Software nicht immer zu einer sinnvollen Nutzung führt. Die Art und Weise, wie eine Software bereitgestellt wird, trägt maßgeblich dazu bei, dass die Nutzungspotentiale einer Software ausgeschöpft werden können. In diesem Zusammenhang wird unter Softwarebereitstellung mehr verstanden als nur der technische Betrieb. Neben der Technik gehören zur Softwarebereitstellung auch Fragestellungen zu Aufgaben und Aufgabenverteilungen, zu Akteuren und Akteursbeziehungen, zu Dienstleistungen und Verträgen, zu Veränderungen und Vorgehensweisen. Das Verständnis von Softwarebereitstellung fußt hierbei auf dem Application Service Providing (ASP) und umfasst technische und organisatorische Aufgaben, welche Nutzenden eine Software zentral als Dienstleistung so zur Verfügung stellt, dass eine gesicherte Nutzung möglich ist.

In dieser Arbeit wird ausgehend von ASP und vor dem Hintergrund des gerade skizzierten Verständnisses von Softwarebereitstellung ein Ansatz zur Analyse und Gestaltung der Softwarebereitstellung in Organisationen erarbeitet. Er soll Fragen zum Verständnis und zu gestalterischen Möglichkeiten in konkreten Bereitstellungskontexten beantworten und Schwierigkeiten in der Softwarenutzung erkennen sowie Optionen zur Problemlösung anbieten. Die Erarbeitung des Ansatzes, der mit eASP für evolutionary Application Service Providing benannt wird, steht im Zentrum dieser Arbeit.

1.1 Motivation

Meine Motivation für die Auseinandersetzung mit dem Thema Softwarebereitstellung liegt in vier Punkten begründet. Erstens wird in der Informatik den Wirkungen von Software im Einsatzkontext bisher zu wenig Aufmerksamkeit geschenkt. Zweitens beeinflusst die Softwarebereitstellung die Bewertung einer Software durch Nutzende, was insbesondere für partizipative und evolutionäre Softwareentwicklungsmethoden relevant ist. Drittens ist das Geschäftsmodell Application Service Providing (ASP) zur Bereitstellung von Software geeignet, der Informatik wichtige Anregungen zu geben. Allerdings ist es noch zu unfundiert und undifferenziert, als dass es mit Blick auf die ersten beiden Aspekte von großem Nutzen wäre. Viertens habe ich die Bereitstellung einer Kooperationsplattform im universitären Kontext über fünf Jahre begleitet.

Der erste Punkt, die mangelnde Einbeziehung des Kontextes in die Informatik, liegt im kritisierten Verständnis der Informatik als Ingenieurswissenschaft begründet. Im Zuge dessen wird die Informatik im Sinne eines Dreischritts von Formalisierung, Algorithmisierung und Maschinisierung verstanden (vgl. Floyd 1994; Rolf 1998; Siefkes 2003). Vorgänge werden durch Gesetzmäßigkeiten beschrieben, die Darstellungen in umsetzbare Anweisungen überführt, die letztendlich rechnergestützt ausgeführt werden. Soziale und organisatorische Randbedingungen werden ignoriert (Rolf 1998, S. 17), die Herstellung von z. B. Softwaresystemen wird aus dem Entstehungs- und Einsatzkontext getrennt (Floyd 1994, S. 30). Es wird kritisiert, dass die Informatik sich meist nur mit dem Funktionieren von IT-Systemen befasst. Wie sie wirken und ob sie ihren Zweck erfüllen, ist diesem Verständnis der Informatik nicht zugänglich. Eine funktionierende und „formal korrekte“ Software kann die Situation bzw. den Kontext dennoch ungewünscht verändern. Ob eine Softwareentwicklung sinnvoll war, zeigt sich erst in der Nutzung (vgl. Floyd 1994, S. 33). So wird die Berücksichtigung des Kontextes für die Informatik gefordert (vgl. Floyd 1994; Rolf 1998; Siefkes 2003); eine Aufgabe, der sich das Forschungsgebiet „Informatiksysteme in Organisationen“ widmet. Ein Ansatz zur Softwarebereitstellung als Ergänzung dieses Forschungsgebietes kann der Informatik helfen, die Gebrauchstauglichkeit ihrer hervorgebrachten Artefakte zu bewerten.

Der zweite Punkt, Beeinflussung der Bewertung einer Software durch die Bereitstellung, wurde durch Strauss u. a. (2003) für die Kooperationsplattform CommSy (2004) nachgewiesen. Aus der Sicht von partizipativen und evolutionären Softwareentwicklungsmethoden bedeutet dies, dass die in der Bereitstellung einer Softwareversion gewonnenen Erkenntnisse zur Entwicklung der neuen Version nicht nur auf die eingesetzte Softwareversion selbst zurückzuführen sind. Die Bereitstellung kann die Erkenntnisse über die Version „verwässern“. Eine schlechte Version kann durch qualitativ hochwertige Bereitstellung als positiver angesehen werden, als sie tatsächlich ist. Eine qualitativ hochwertige Version kann durch eine schlechte Bereitstellung von den Nutzenden „unter Ihrem Wert“ bewertet werden. Das bedeutet, die Softwareentwicklung muss die Erkenntnisse über eine Softwareversion immer im Verhältnis zur Qualität der Bereitstellung bewerten. Insofern müssen zyklische Softwareentwicklungsmethoden auch die Softwarebereitstellung in ihre Betrachtung einbeziehen.

Der dritte Punkt, ASP, hat in der Erkenntnis seinen Ursprung, dass trotz Bedarf keine Ansätze in der Informatik zur Softwarebereitstellung zu finden sind. In den Blick „über den Tellerrand“ gelangt schnell das Outsourcing, welches durch die rasante Entwicklung in der Informations- und Kommunikationstechnologie und durch den Aufbau von Netzwerkorganisationen in Wirtschaft und Gesellschaft ermöglicht wurde. Zunächst wurden gesamte Geschäftsprozesse aus Unternehmen ausgelagert (Business Outsourcing) oder gesamte bzw. Teile der IT-Abteilung (IT-Outsourcing). In jüngerer Zeit werden von Unternehmen und auch Privatpersonen einzelne Softwarepakete als Dienstleistungen von entsprechenden Anbietern über das Internet „gemietet“. Das Application Service Providing (ASP) als spezielle Form des IT-Outsourcings entstand. ASP ist ein Geschäftsmodell hinsichtlich der Bereitstellung von Software, welches die Bereitstellung umfassend betrachtet. Es benennt Aufgaben und Aufgabenbereiche, die in Wertschöpfungsketten angeordnet werden. Es benennt Akteure, die an der Softwarebereitstellung beteiligt sind, und deren benötigte (Kern-)Kompetenzen. Es benennt Vertragsarten und -formen, sog. Service Level Agreements (SLAs), die detailliert Rechte und Pflichten in einer Kunden-Dienstleister-Beziehung definieren. Dennoch ist ASP in verschiedenen Aspekten zu oberflächlich und nicht fundiert genug, um der Informatik hinsichtlich der ersten beiden Punkte meiner Motivation hilfreich zu sein. Es stellt aber ein gutes Grundgerüst für einen Ansatz der Softwarebe-

reitstellung dar, das zunächst fundiert werden muss, damit es für die Informatik nutzbar sein kann.

Der vierte Punkt, Erfahrung, resultiert aus den Erkenntnissen während der über fünf Jahre dauernden Bereitstellung und Entwicklung der Kooperationsplattform CommSy, die ich seit Beginn 1999 begleitet und mitgestaltet habe. Herauszuheben sind zwei konkrete Bereitstellungskontexte:

1. In Kooperation des Technologietransfervereins HITeC e.V. des Fachbereichs Informatik der Universität Hamburg mit dem Startup-Unternehmen Uni.de AG wurde CommSy in Zeitraum 2000 – 2002 nicht sehr erfolgreich bundesweit bereitgestellt.

2. Im Forschungs- und Entwicklungsprojekt WissPro ist die bundesweite Bereitstellung von CommSy im Zeitraum 2001 – 2003 als erfolgreich zu bewerten.

Hier stellen sich Fragen nach den Unterschieden in den beiden Kontexten. Ein Ansatz zur Softwarebereitstellung soll tiefe Einblicke in diese Bereitstellungskontexte, die in dieser Arbeit als Fallstudien genutzt werden, ermöglichen und Hinweise für die zukünftige Gestaltung der Bereitstellung von CommSy geben.

1.2 Einordnung in die Informatik

Diese Arbeit ordnet sich in das Forschungsgebiet „Informatiksysteme in Organisationen“ der Informatik, speziell in den Bereich „organisationsbezogene Softwareentwicklung“, ein. In diesem Gebiet werden technische und soziale Aspekte im Zusammenhang mit der gesellschaftlichen Einbindung der beteiligten Akteure bei der Softwareentwicklung und -nutzung verknüpft. Eine Grundannahme in diesem Gebiet ist, dass ein Verständnis der Zusammenhänge zwischen Organisationen und Softwareentwicklung notwendige Voraussetzung ist, um Ansätze und Methoden hervorzubringen, die die organisatorischen Bedingungen unmittelbar berücksichtigen und dadurch zu organisatorisch angemesseneren Prozessen und Produkten der Softwareentwicklung führen (vgl. Klischewski 2004).

Seit einigen Jahren findet der Begriff „organisationsbezogene Softwareentwicklung“ (Floyd 1994) im Fachbereich Informatik der Universität

Hamburg Verwendung, um interdisziplinäre Aspekte in der Softwaretechnik und Angewandten und Sozialorientierten Informatik zu thematisieren und zu verknüpfen. Hiermit ist das Erkenntnisinteresse verbunden, sich systematisch mit den Wechselwirkungen zwischen Anwenderorganisation und Softwareentwicklung auseinander zu setzen. Angrenzende Forschungsgebiete stellen auf dem Gebiet „Wirkung von Anwendungen in Organisationen“ die Wirtschaftsinformatik dar, die im anglo-amerikanischen und skandinavischen Raum geprägten Information Systems (IS) und der Bereich „Social Informatics“ (vgl. Kling 1999) sowie bei der Einbeziehung der Organisation in die Softwareentwicklung die Mensch-Computer-Interaktion und das Systems Development, speziell im Skandinavischen. Insbesondere der Methodenrahmen der Softwareentwicklung STEPS (Softwaretechnik für evolutionäre, partizipative Systementwicklung) (Floyd 1986; 1991b; Floyd u. a. 1989b) und das Konzept der IT-unterstützten Organisationsgestaltung (Rolf 1998) sind im an der Universität Hamburg entwickelten Verständnis von Informatiksystemen in Organisationen in diesem Forschungsgebiet von zentraler Bedeutung.

Die Softwarebereitstellung ordnet sich sehr gut in das Gebiet „Informatiksysteme im Kontext“ bzw. der organisationsbezogenen Softwareentwicklung ein, da es die entwickelte Software(-version) im Anwendungskontext zur Nutzung bringt. Sie ist damit in der Schnittmenge von Softwareentwicklung und Organisationsgestaltung verortet. Die Integration des in dieser Arbeit entwickelten Ansatzes zur Analyse und Gestaltung der Softwarebereitstellung in die im Gebiet „Informatiksysteme in Organisationen“ beheimateten Modelle und Methoden ist eines der intendierten Ergebnisse dieser Arbeit.

1.3 Intendierte Ergebnisse

Um die Softwarebereitstellung verstehen und gestalten zu können und sie als Teil von Informatiksystemen in Organisationen und der organisationsbezogenen Softwareentwicklung zu charakterisieren, verfolgt diese Arbeit zwei Ziele:

1. Erarbeitung eines Analyse und Gestaltungsrahmens der Softwarebereitstellung und

2. Integration der Softwarebereitstellung in den Methodenrahmen der Softwareentwicklung STEPS und das Konzept der IT-unterstützten Organisationsgestaltung.

Auf Grundlage des Geschäftsmodells Application Service Providing (ASP) wird durch eine Fundierung und Erweiterung mit Modellen und Methoden aus „Informatiksysteme in Organisationen“ ein Analyse- und Gestaltungsansatz der Softwarebereitstellung entwickelt, welcher eASP (evolutionary Application Service Providing)[1] genannt wird. eASP bezeichnet die evolutionäre Bereitstellung einer Anwendung als Dienstleistung und korrespondiert damit mit dem in dieser Arbeit zugrunde liegende Verständnis von Softwarebereitstellung als zentrale Dienstleistung für viele Nutzende.

Zur Integration der Softwarebereitstellung in STEPS und die IT-unterstützte Organisationsgestaltung wird nach der Entwicklung des Ansatzes eASP diskutiert, wie eASP sich in die beiden Ansätze einbetten kann. In STEPS wird die Softwarebereitstellung als Tätigkeit im Softwareentwicklungsprozess identifiziert und in das zyklische Projektmodell eingeordnet. Die Integration in die IT-unterstützte Organisationsgestaltung gelingt, indem die Softwarebereitstellung den dort enthaltenen Gestaltungsprozess zu einem Zyklus formt. Die Einordnung in beide Ansätze erweitert diese und verankert die Softwarebereitstellung im Forschungsgebiet „Informatiksysteme im Kontext“.

1.4 Methodisches Vorgehen

Ausgehend von dem Bedarf der Einbeziehung der Softwarebereitstellung in die Wissenschaftsdisziplin Informatik, speziell im Gebiet „Informatiksysteme in Organisationen“, wird aus der Praxis das Geschäftsmodell ASP herangezogen und auf Grundlage der Empirie mit Methoden und Modellen aus der organisationsbezogenen Softwareentwicklung fundiert und erweitert. Resultat der Fundierung und Erweiterung ist der Analyse- und Gestaltungsansatz eASP. Seine Anwendbarkeit und Praxistauglichkeit wird mit Hilfe der Empirie überprüft, die im Sinne der Aktionsforschung gestaltet

1 Das „e“ bei eASP ist nicht zu verwechseln mit dem „E“ bei z. B. E-Learning oder E-Government. Das große „E“ steht bei den entsprechenden Begriffen für Elektronisch bzw. Electronic. Das „e“ von eASP bedeutet evolutionär bzw. evolutionary und ist zur besseren Unterscheidbarkeit immer klein und ohne nachfolgenden Bindestrich geschrieben.

und dokumentiert worden ist. Die Anwendung von Ansätzen aus dem Gebiet der Informatik, in dem der Bedarf konstatiert wird, sichert die direkte Vermittlung der Ergebnisse in die Forschung, wie in den letzten Kapiteln dieser Arbeit gezeigt wird. Darüber hinaus erleichtert der starke Bezug auf ein Praxismodell die Rückführung der Ergebnisse in die Praxis.

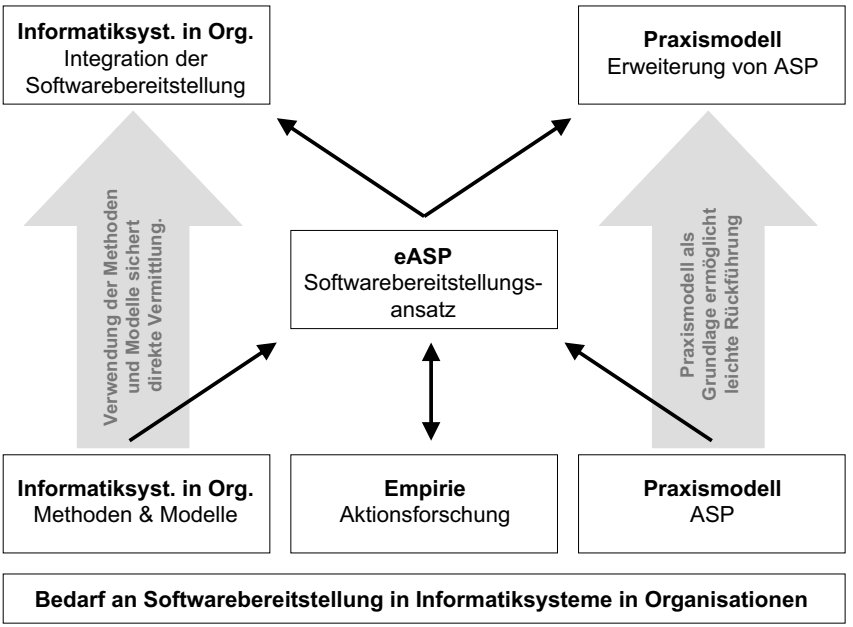

Abbildung 1: Methodisches Vorgehen

1.5 Weiterer Aufbau

Diese Arbeit gliedert sich in drei Teile. Im ersten Teil *Grundlagen* wird das Thema dieser Arbeit „Softwarebereitstellung“ beschrieben: Die empirischen Grundlagen, das Geschäftsmodell ASP und Methoden und Modelle aus dem Gebiet „Informatiksysteme in Organisationen“. Im zweiten Teil *Herleitung* werden diese Methoden und Modelle auf ASP angewendet und mit dem Ergebnis die Fallstudien korreliert. Im dritten Teil *Ergebnis* wird

der Softwarebereitstellungsansatz noch einmal kompakt dargestellt, diskutiert und in das Gebiet „Informatiksysteme in Organisationen“ eingeordnet.

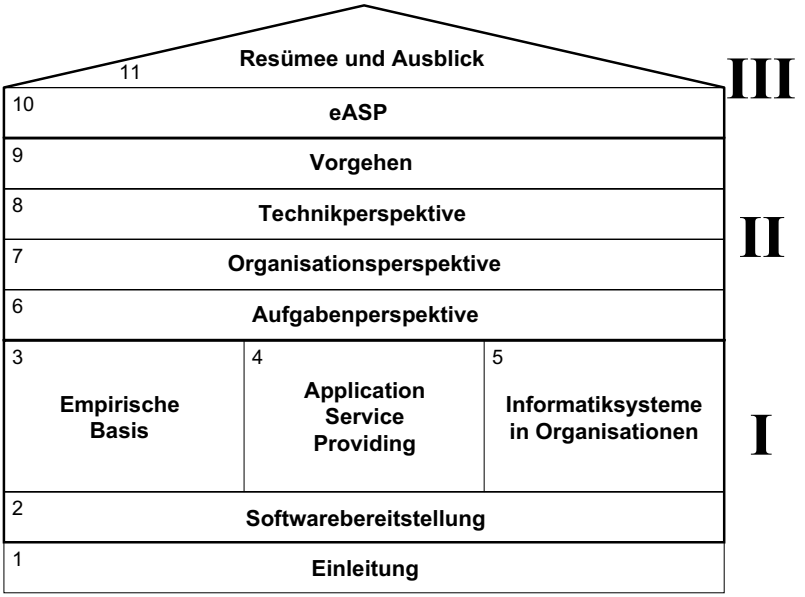

Abbildung 2: Gliederung dieser Arbeit

I Grundlagen zur Herleitung von eASP

Im ersten Teil werden die Grundlagen dieser Arbeit gelegt. So nimmt diese Arbeit ihren Ausgangspunkt in der Klärung des Verständnisses des Begriffs „Softwarebereitstellung“ (Kapitel 2).

Auf diesem Fundament aufbauend werden Erfahrungen in der Softwarebereitstellung präsentiert, die die empirische Basis in dieser Arbeit bilden (Kapitel 3). Im Mittelpunkt stehen dabei die webbasierte Kooperationsplattform CommSy und die beiden Bereitstellungskontexte CommSy@Uni.de und CommSy@WissPro (Fallstudien). Die Bereitstellung von CommSy in den Fallstudien verlief sehr unterschiedlich. Während sie in einem Fall scheiterte, war sie im anderen erfolgreich. Es stellt sich die Frage nach dem Warum, die unmittelbar zu der Frage nach Unterschieden zwischen den Fallstudien führt. Diese Unterschiede lassen sich hinsichtlich

der Perspektiven Aufgaben, Organisation, Technik und Vorgehen ordnen. So werden ausgehend von den zwei empirischen Fallstudien zur Bereitstellung von Software vier Perspektiven (Aufgaben, Organisation, Technik und Vorgehen) auf die Softwarebereitstellung eingenommen, gemäß denen der zweite Teil dieser Arbeit strukturiert wird.

Die anschließende Darstellung des Status quo der Diskussion um das Geschäftsmodell ASP macht deutlich, dass es den genannten vier Perspektiven nicht gerecht wird und daher nicht ohne weiteres als Ansatz zur Softwarebereitstellung gelten kann (Kapitel 4). Insbesondere in den Perspektiven Aufgaben, Organisation und Vorgehen erscheint ASP wenig detailliert und fundiert. Zur Überwindung der in Kapitel 4 erkannten Schwächen von ASP werden in Kapitel 5 Methoden und Modelle aus dem Gebiet „Informatiksysteme in Organisationen“ vorgestellt.

Die Darstellung der jeweiligen Grundlagen erfolgt weitgehend ohne direkt erkennbaren Bezug zueinander, so dass sie hinsichtlich des Aufbaus nebeneinander stehen und als Grundpfeiler dieser Arbeit interpretiert werden können. Die Verbindung der im ersten Teil vorgestellten Grundlagen erfolgt im zweiten Teil „Herleitung von eASP“.

II Herleitung von eASP

Ein Ansatz zur Softwarebereitstellung muss den Perspektiven Aufgaben, Organisation, Technik und Vorgehen gerecht werden, so dass sich der zweite Teil dieser Arbeit jeweils einer Perspektive in einem Kapitel annimmt und dadurch der Ansatz eASP hergeleitet wird.

Zur Ausgestaltung der jeweiligen Perspektive werden auf Aspekte von ASP Methoden und Modelle aus dem Gebiet „Informatiksysteme in Organisationen“ angewendet. Anschließend werden die Fallstudien CommSy@Uni.de und CommSy@WissPro in jeder Perspektive betrachtet, um zum einen die praktische Verwendbarkeit von eASP, zum anderen die Unterschiede in den Fallstudien pro Perspektive aufzuzeigen.

Am Anfang wird die Aufgabenperspektive eingenommen, um grundsätzlich das Feld der Aufgabenbereiche in der Softwarebereitstellung aufzufächern (Kapitel 6). Anschließend wird die Organisationsperspektive eingenommen, in der es um die Übernahme bzw. Durchführung der zuvor präsentierten Aufgaben geht (Kapitel 7). In der Technikperspektive werden dann technische Grundlagen der Bereitstellung behandelt (Kapitel 8). Den

drei Perspektiven fehlt eine zeitliche Komponente, so dass im letzten Kapitel von Teil II eine Vorgehensweise aufgezeigt wird, die den Umgang mit inneren und äußeren Einflüssen auf die Softwarebereitstellung ermöglicht (Kapitel 9).

III Ergebnis - der Ansatz eASP

Im dritten und letzten Teil wird der Softwarebereitstellungsansatz eASP noch einmal kompakt dargestellt. Nach der kompakten Darstellung wird eASP in das Gebiet „Informatiksysteme in Organisationen“ eingeführt, indem es in den Methodenrahmen der Softwareentwicklung STEPS und das Konzept der IT-unterstützten Organisationsgestaltung integriert wird. Es folgen Aussagen zur Verallgemeinerbarkeit von eASP über die Fallstudien hinaus und eine Diskussion, inwiefern eASP als Erweiterung von ASP gelten kann (Kapitel 10).

Zum Schluss wird diese Arbeit resümiert und ein Ausblick auf unbeantwortet gebliebene Fragen, zukünftig zu erledigende Aufgaben und in Zukunft zu erforschende Aspekte im Zusammenhang mit der Bereitstellung von Software gegeben (Kapitel 11).

1.6 Schreibweisen

In dieser Arbeit wird auf Kunstwörter wie „Nutzer/innen“ oder „NutzerInnen“ sowie auf die lange Form „Nutzerinnen und Nutzer“ aus Gründen der Lesbarkeit verzichtet. Selbstverständlich gilt das im Folgenden Geschriebene für Wissenschaftlerin und Wissenschaftler sowie für Leserin und Leser. Da das Deutsche meist keinen neutralen Ausdruck hat, der beide Geschlechter umfasst, beschränke ich mich darauf, die männliche Form zu verwenden. Ich folge damit grammatikalischem Brauch und bringe keinerlei Diskriminierung wegen des Geschlechts zum Ausdruck. Darüber hinaus habe ich versucht, wenn möglich, eine geschlechtsneutrale Form zu finden und diese zu verwenden, z. B. „Studierende“ statt „Studenten“.

Während im Plural geschlechtsneutrale Formulierungen größtenteils zu finden waren, war dies im Singular meist nicht möglich. So kann natürlich gefragt werden, warum ich bei der Verwendung des Singulars nicht auf den geschlechtsneutralen Plural ausgewichen bin. Da es sich im Folgenden bei der Verwendung des Singulars in der Regel um die Bezeichnung einer

Rolle (Nutzer, Entwickler, Bereitsteller) handelt, bin ich hier dem Konzept der Rolle gefolgt, das generell den Singular als Schreibweise vorsieht. Der Verzicht auf Wendungen wie „der/die Benutzer/in“ dient wiederum der einfachen Lesbarkeit und drückt nicht meine Einstellung zu Fragen der Emanzipation der Frau aus.

Ich habe Quellen nicht mit der in den in Informatik-Publikationen üblichen Abkürzungen in Verweisen auf Quellen angegeben, sondern mit vollständigen Personennamen, da es mir bei den Zitaten auf die Urheber als Personen ankommt, die für die Aussagen stehen. Quellenverweise in Zitaten wurden mit in das Literaturverzeichnis übernommen, um die Zitate vollständig zu nennen und den Lesern die Nutzung dieser Quellen zu erleichtern. Soweit möglich, wurden die in Zitaten verwendeten Quellenverweise ebenfalls recherchiert.

Abschließend sei noch erwähnt, dass diese Arbeit mit Microsoft Word 2000 und 2002 verfasst und dann in LaTeX gesetzt wurde.

I

Grundlagen zur Herleitung von eASP

- 2 -
Begriffsbestimmung „Softwarebereitstellung“

Das intendierte Ziel dieser Arbeit ist die Entwicklung eines Analyse- und Gestaltungsansatzes für die Softwarebereitstellung (eASP). So stellt die Softwarebereitstellung das Thema und den Rahmen für diese Arbeit dar und wird im folgenden Kapitel näher betrachtet. Zunächst folgt eine Definition der Softwarebereitstellung, um das dieser Arbeit zugrunde liegende Verständnis von Softwarebereitstellung zu erläutern und zu begrenzen. Danach wird dem Ursprung dieses Verständnisses nachgegangen. Anschließend folgt eine Abgrenzung zu anderen verwandten und ähnlichen Begriffen.

2.1 Definition

Das dieser Arbeit zugrunde liegende Verständnis von Softwarebereitstellung wird wie folgt definiert:

Die Softwarebereitstellung umfasst technische und organisatorische Aufgaben, welche Nutzenden eine Software zentral als Dienstleistung so zur Verfügung stellt, dass eine gesicherte Nutzung möglich ist.

Kernaspekte dieser Definition sind:

- *Eine Software* beschränkt das Verständnis von Softwarebereitstellung für diese Arbeit auf die Bereitstellung einer einzigen Software. Potentiell ist die ganzheitliche Betrachtung der Bereitstellung verschiedener Software denkbar, aber wesentlich komplexer, da in diesem Fall Interdependenzen zwischen den verschiedenen Softwarebereitstellungen entstehen. So wird aus Gründen der Komplexitätsreduktion in dieser Arbeit auf die Bereitstellung einer Software fokussiert.
- *Gesicherte Nutzung* bedeutet die Sicherung der kurzfristigen und der langfristigen Verfügbarkeit der Software und Supportmaßnahmen.

Server bzw. Software und Servicedienste wie Webseiten, elektronische Handbücher, Hotlines usw. müssen im Arbeitsalltag der Nutzenden ständig verfügbar sein. Auch die langfristige Pflege und Wartung der Hard- und Software sowie der Servicedienste müssen gesichert werden. Nur unter diesen Voraussetzungen können Nutzende die bereitgestellte Software aufgabenangemessen in Ihren Arbeitsprozess bzw. Arbeitsalltag integrieren.

- *Technische und organisatorische Aufgaben* erweitern den Blick über den meist unter Softwarebereitstellung verstandenen technischen Betrieb von Software auf organisatorische Maßnahmen. Es müssen die Benutzerbetreuung, Fragen der Etatisierung und Kosten der Bereitstellung, Aufgaben in der Softwareentwicklung, u. a. Fehlerbehebung und Anpassung, sowie Aufgaben im Marketing und in der Kundenbetreuung beachtet werden.

- Mit *zentral* wird hervorgehoben, dass die Software an einem Ort für potentiell viele gleichzeitig bereitgestellt wird. Die Nutzenden müssen nicht (können aber) in unmittelbarer räumlicher oder organisatorischer Nähe dieses zentralen Ortes beheimatet sein. Institutionell werden keine Grenzen gesetzt. Organisatorisch entspricht dies einer 1:N-Beziehung. Technisch erinnert dies bewusst an das in der Informatik bekannte Client/Server-Prinzip, ohne Thinclients oder Terminal/Host-Ansätze auszuklammern.

- Der Begriff *Dienstleistung* vermittelt eine starke Kundenorientierung, die sich u. a. in Servicequalität und Kundenzufriedenheit ausdrückt. Darüber hinaus wird die Kooperation zwischen Bereitsteller und Nutzer als Dienstleistungsbeziehung interpretiert. Dadurch rücken vertragsrechtliche und monetäre Aspekte in den Vordergrund.

- *„welche Nutzenden eine Software … zur Verfügung stellt“*: Mit der Nennung der Nutzenden wird die Softwarebereitstellung im Einsatzkontext verankert. Die Softwarebereitstellung verbleibt damit nicht auf der Ebene von Planung und Herstellung der Bereitstellung, sondern bezieht die Durchführung explizit mit ein.

Diese Sichtweise auf die Bereitstellung von Software grenzt sich von einer schlichten Downloadmöglichkeit mit anschließender lokaler Installation ab. Die autarke Einzelinstallation wird in dieser Arbeit nicht betrachtet.

2.2 Ursprung

Im Folgenden wird der Ursprung des in dieser Arbeit verwendeten Verständnisses von Softwarebereitstellung und dessen Entwicklung vom Outsourcing über das Application Service Providing bis zur oben skizzierten Definition dargestellt.

Das in dieser Arbeit verwendete Verständnis von Softwarebereitstellung hat im Outsourcing seinen Ursprung. „Outsourcing“ ist eine Komposition aus den Wörtern „outside resource using“ (Ströbel 2000, S. 3). Es bezeichnet, aus der Sicht eines Kunden, die Benutzung einer Ressource bei einem externen Anbieter. Darüber hinaus definieren einige Autoren (vgl. TripleTree 2003; Zahn u. a. 1999) Outsourcing als den Transfer von internen Leistungen zu externen Anbietern. Sie beziehen damit auch den Prozess der Auslagerung ins Outsoucing ein. Grover u. a. (1997, S. 80) sehen Outsourcing dagegen als „organizational decision to turn over part or all of an organisation's IS function to external service providers in order for an organization to be able to achieve its goals“. Hierbei schränken Grover u. a. (1997), wie auch Lacity und Hirschheim (1999), die auszulagernden Ressourcen bereits auf Informationssysteme (IS) bzw. Informations-Technologie (IT) ein. Andere Autoren (vgl. Stahlknecht 2000; TripleTree 2003) sehen IT-Outsourcing als einen durchaus bedeutenden Aspekt von Outsourcing, aber Outsourcing an sich wird umfassender als „Verlagerung von Wertschöpfungsaktivitäten eines Unternehmens auf andere Unternehmen“ (Stahlknecht 2000, S. 5) verstanden. Dabei unterscheiden Stahlknecht (2000, S. 5f.) und Lacity und Hirschheim (1999, S. 328f.) totales (komplettes) und partielles (selektives) Outsourcing. Während Stahlknecht (2000, S. 5) neben dieser funktionalen Dimension noch die zeitliche anführt und in befristetes und dauerhaftes Outsourcing unterscheidet, definieren Lacity und Hirschheim (1999, S. 328f.) noch das „value-added outsourcing“ und das „co-operative outsourcing“.

Ende des letzten Jahrtausends entstanden zur Zeit des New-Economy-Hypes viele Start-up-Unternehmen, die Software, auf ihren eigenen Servern installiert, mittels Internettechnologien anderen Unternehmen kosten-

pflichtig zur Verfügung stellten. Diese Start-up-Unternehmen (Application Service Provider) hatten das IT-Outsourcing auf ein Application Outsourcing (TripleTree 2003, S. 1 und S. 30) via Internettechnologien beschränkt und als Geschäftsmodell erfolgreich umgesetzt. Das Geschäftsmodell wurde nach den „Bereitstellern“ Application Service Providing (ASP) genannt (siehe Kapitel 4).

Obwohl ASP als eine spezielle Form des Outsourcings gilt (vgl. Stahlknecht 2000; CherryTree 1999; Farleit 2000), wird es nicht Outsourcing oder nach TripleTree (2003, S. 1 und S. 30) Application Outsourcing genannt. Dennoch handelt es sich bei dem von vielen Autoren beschriebenen ASP immer um die Benutzung von (aus Sicht des Kunden) extern erbrachten Leistungen, also einem „outside resource using“, auch wenn z. B. Unterschiede im „Traditional Outsourcing Model“ und dem „Application Service Provision Model“ auf der Ebene von Vertragslaufzeiten, Unternehmensgröße von Kunden, standardisierter oder Individualsoftware usw. gesehen werden (Endres 2004; Yao und Murphy 2002).

Die Betrachtung von ASP als ein spezielles Outsourcing wird von einigen Autoren kritisiert. Sie verstehen Application Service Providing als Bereitstellung einer Applikation als Dienstleistung, vollkommen unabhängig davon, ob die Bereitstellung intern oder extern erfolgt (vgl. Picot und Jahn 2000; Riemer und Ahlemann 2001; Rosenhagen 2002). Es wird argumentiert, dass bei der Betrachtung von ASP als Spezialfall des Outsourcings zwei verschiedene Gegenstandsebenen vermischt werden. Zum einen geht es um die Frage „inside“ oder „outside resource using“ und zum anderen um die Bereitstellung an sich, bei der ASP von z. B. Application Hosting unterschieden werden muss (vgl. Riemer und Ahlemann 2001, S. 745).

Diese Kritik, verbunden mit der rasch voranschreitenden Entwicklung in der Internettechnologie, die den physischen Standort der zentralen Softwarebereitstellung obsolet macht, führt zu einem neuen, dem in Abschnitt 2.1 angegebenen Verständnis von Softwarebereitstellung. In diesem Verständnis spielt die physische und institutionelle Verortung von Bereitsteller und Nutzer, von Server und Client keine Rolle mehr. Vielmehr geht es um die Bereitstellung einer Software als Dienstleistung, sei es nun intern oder extern. Wichtig ist, dass die Nutzung der zentral für viele Nutzende zur Verfügung gestellten Software in allen Belangen gesichert ist. Insofern umfasst die Softwarebereitstellung nicht mehr nur technische, sondern auch organisatorische Aufgaben.

Dieses weiterentwickelte Verständnis der Softwarebereitstellung überwindet ihre bisherige Begrenzung auf Outsourcing-Szenarien und nimmt Insourcing-Szenarien als Gegenpol in die Betrachtung auf. Beide Szenarien spannen in ihren Extremen ein Feld auf, in dem es verschiedene Mischszenarien gibt. Welche „Mischung“ in einem konkreten Bereitstellungskontext passt, ist von vielen Faktoren abhängig und nicht mehr eine generelle Frage des In- oder Outsourcings. Welches diese Faktoren sind, wird durch den in dieser Arbeit entwickelten Ansatz zur Analyse und Gestaltung der Softwarebereitstellung deutlich werden.

2.3 Abgrenzung

Im Kontext der Bereitstellung von Software werden weitere Begriffe neben der Bereitstellung verwendet, die ähnliche oder gleiche Dinge, Untergeordnetes oder Übergeordnetes ansprechen. Diese sind: Installation, Anpassung und Konfiguration, Hosting, Wartung, Benutzer- und Benutzungsbetreuung, Nutzung und Infrastrukturen. In diesem Abschnitt wird die Bereitstellung von Software von diesen Begriffen abgegrenzt.

2.3.1 Installation

In der Softwaretechnik bzw. dem Softwareengineering wird unter Installation die Einbettung der Software in die Systemumgebung des Auftraggebers (Duden 1993, S. 664) bzw. die „technische Übertragung aus der Entwicklungs- in die Einsatzumgebung“ (Floyd und Züllighoven 2002, S. 776) verstanden. Während Floyd und Züllighoven (2002, S. 776) die „notwendigen organisatorischen Umstellungen und die Schulung der Benutzer“ nicht mehr als Teil der Installation, sondern als Teil der „Systemeinführung“ neben der Installation sehen, ist laut Informatik-Duden (1993, S. 664) die Schulung der Nutzer integraler Bestandteil der Installation.

Beiden Ausprägungen von Installation bzw. der Systemeinführung fehlt im Sinne der Bereitstellung die Kontinuität. Die Installation wird nur ein einziges mal durchgeführt, während die Bereitstellung kontinuierlich erfolgen muss. Dennoch müssen bei der Bereitstellung die bereitzustellende Software, eventuelle Übertragungstechniken und ggf. Clientsoftware installiert werden. Ebenso müssen, wie beide Sichten andeuten, neben der Technik weitere Aufgaben zur Installation übernommen werden: Benutzerschu-

lung, organisatorische Anpassungen usw. So kann die Installation als eine Aufgabe der Bereitstellung angesehen werden.

2.3.2 Anpassung und Konfiguration

Während in der Informatik unter der Anpassung von Software eher die softwaretechnische Anpassung einer Software an zusätzliche oder sich verändernde Anforderungen verstanden wird (vgl. Sinz 2002; Duden 1993), meint Konfiguration eine Veränderung von Software nur durch das Setzen von Umgebungsvariablen in Konfigurationsdateien oder das Einstellen an der Benutzungsoberfläche (Konfigurationsoptionen). Bei der Konfiguration werden die Softwarearchitektur und der Softwarecode im Gegensatz zur Anpassung nicht verändert.

Wie bei der Installation fehlt der Anpassung und der Konfiguration die Kontinuität der Bereitstellung, auch wenn diese Aufgaben z. B. aufgrund von Veränderungen in Hard- und Software häufiger durchgeführt werden müssen. Ihre Durchführung bleibt über einen längeren Zeitraum betrachtet punktuell. Trotzdem sind dies Aufgaben, die in der Softwarebereitstellung wahrgenommen werden müssen, und so können sie als wiederkehrende Aufgaben in der Bereitstellung gelten.

2.3.3 Hosting

Application Hosting bezeichnet den Betrieb einer Hardware- und Netzinfrastruktur. Im Leistungsumfang ist die Software selbst nicht enthalten, sondern nur deren Speicherung und deren Betrieb. d. h. die Lizenzgebühren bzw. Investitionen in die Entwicklung von Software liegen nicht beim Hosting-Partner, sondern beim Hosting-Kunden (vgl. Riemer und Ahlemann 2001, S. 745). Darüber hinaus liegt auch die Benutzerbetreuung in Kundenhand.

Das Hosting bezeichnet den Betrieb der bereitzustellenden Software, d. h. die Installation und Wartung der benötigten Hard- und Software. Die Softwarebereitstellung wird umfassender verstanden, so dass das Hosting bzw. der Betrieb als eine Aufgabe der Softwarebereitstellung interpretiert werden kann.

2.3.4 Wartung

Die Wartung wird als Pflege der aktuellen bzw. im Betrieb befindlichen Software(-version) angesehen (Duden 1993; Floyd und Züllighoven 2002). Während Floyd und Züllighoven (2002, S. 776) unter Pflege primär die Fehlerbehebung verstehen und pragmatisch zwischen „der Pflege der aktuellen Version und der Entwicklung einer neuen Version“ unterscheiden, differenziert der Informatik-Duden (1993) eine Software nicht in verschiedene Versionen. Vielmehr bezieht er in die Wartung neben der Fehlerbehebung auch den „Austausch von bestimmten Algorithmen durch leistungsfähigere (z. B. schnellere) oder [den] Einbau zusätzlicher Benutzerfunktionen“ mit ein (Duden 1993, S. 664).

Beide Sichtweisen auf den Begriff Wartung nehmen nur die technische Seite in den Blick. Der Begriff Softwarebereitstellung ist umfassender, da auch organisatorische Belange „gewartet“ werden müssen, insbesondere, wenn Fehler auftreten und bereinigt werden oder eine neue Version installiert wird. In diesem Sinne kann die Wartung als eine Aufgabe der Softwarebereitstellung verstanden werden.

2.3.5 Benutzer- und Benutzungsbetreuung

Benutzer- und Benutzungsbetreuung unterscheiden sich ebenfalls von der Softwarebereitstellung. Dabei ist die Benutzungsbetreuung nicht eine genderneutrale Schreibweise von Benutzerbetreuung, sondern geht über das Verständnis der Benutzerbetreuung hinaus.

Der Begriff Benutzerbetreuung bzw. Anwenderbetreuung leitet sich von der speziell in der Wirtschaftsinformatik diskutierten Thematik des Benutzer-Service ab (vgl. Heinrich 1992; 1999; Heinrich und Hänschel 1996; Knolmayer 1996). Unter Benutzerbetreuung wird hierbei ein kontinuierlicher Betreuungsprozess verstanden, der nicht bei der initialen Schulung endet (Heinrich 1992; Knolmayer 1996). Zur Benutzerbetreuung zählen u. a. Schulung, Coaching, Support-Hotline (Hilfestellung per Telefon oder E-Mail), Weiterbildungsangebote und Workshops zum Erfahrungsaustausch (vgl. Jackewitz 1998; Pape und Jackewitz 2002; Pape u. a. 2002b). Da die Benutzerbetreuung sich stark auf die Betreuung der Nutzenden konzentriert, verliert sie technische Aspekte, wie Wartung, Installation und Konfiguration, aus dem Blick. Insofern kann sie als eine Aufgabe in die Softwarebereitstellung einsortiert werden.

Die Benutzungsbetreuung nimmt sich der Betreuung der Nutzung an und versteht dabei die Benutzerbetreuung als ein Aufgabengebiet neben anderen. Weitere Aufgabengebiete sind u. a. Auswahl, Einsatz und Test neuer Hard- und Software(-versionen), Installation und Wartung bestehender Softwarepakete, Bestellung von Hard- und Software, Standardisierung, Lagerhaltung von Ersatzteilen, Koordination der zentralen und dezentralen Benutzungsbetreuung (vgl. Jackewitz 1998; Langel-Nentwig 1991; Maisberger 1987; Mertens 1985; Weltz und Ortmann 1987; Pape und Jackewitz 2002; Pape u. a. 2002b). Die Benutzungsbetreuung deckt mit ihrem umfassenderen Verständnis verschiedene Aspekte der Softwarebereitstellung ab. Geringe Beachtung schenkt die Benutzungsbetreuung allerdings der Fehlerbehebung (Softwareentwicklung) oder dem Marketing. Darüber hinaus wird der Dienstleistungsbegriff in der Benutzungsbetreuung wenig betrachtet, während er bei der Softwarebereitstellung grundlegend ist.

2.3.6 Nutzung

Unter der Nutzung von Software wird umgangssprachlich die Benutzung von Software verstanden. Im Sinne der Softwarebereitstellung kommt diese Aufgabe den Nutzenden zu.

Pape (2004, S. 71) versteht die Softwarenutzung umfassender als situative Gesamtheit aller Aktivitäten, die dazu beitragen, dass ein Softwaresystem zur Unterstützung bestehender oder neuer Handlungsweisen routiniert verwendet werden kann. Dieses Verständnis umfasst die Bereitstellung von Software, deren organisatorische Einbettung in Handlungsweisen (z. B. Unternehmensprozesse) und deren soziale Einbettung in die Organisation. In diesem Verständnis ist die Softwarebereitstellung ein Teil der Softwarenutzung.

Zur Organisation der Softwarenutzung entwickelt Pape (2004) einen Analyse- und Gestaltungsrahmen, in den sich der in dieser Arbeit entwickelte Ansatz eASP zur Analyse und Gestaltung der Softwarebereitstellung einordnen und abgrenzen lassen muss. Zur Illustration der Einordnung und Abgrenzung wird im Folgenden auf den Analyse- und Gestaltungsrahmen für die Organisation der Softwarenutzung von Pape (2004) eingegangen.

Analyse- und Gestaltungsrahmen der Softwarenutzung
Dem Analyse- und Gestaltungsrahmen liegen drei Annahmen der Softwarenutzung zugrunde (vgl. Pape 2004, S. 5 und S. 11f.):

1. Für einen Nutzungskontext gibt es nicht das richtige Softwaresystem, sondern nur ein mehr oder weniger passendes.

2. Die Passung zwischen Softwaresystem und Nutzungskontext wird durch zahlreiche und vielfältige Aktionen bestimmt.

3. Die Softwarenutzung erfordert ein kontinuierliches und wechselseitig abgestimmtes Engagement der beteiligten Akteure.

Aufbauend auf dieser Grundlage entwickelt Pape (2004) seinen Analyse- und Gestaltungsrahmen der Organisation der Softwarenutzung, bestehend aus Aktionen, Episoden und Strukturen, die jeweils charakteristische Merkmale aufweisen und vor dem Hintergrund bestimmter Strategien zur Analyse und Gestaltung zu bewerten sind:

- Eine *Aktion* ist örtlich und zeitlich eindeutig definiert und kann bestimmten Akteuren zugeordnet werden. Durch Aktionen erfolgt die schrittweise Fortsetzung der organisatorischen Praxis in konkreten Situationen. Aktionen unterscheiden sich in ihrer Form (Ort und Zeit, beteiligte Akteure usw.) und den Motiven der handelnden Akteure. Die mit Aktionen verbundenen Strategien sind die Auseinandersetzung mit der Vorgeschichte, der Einsatz als Wendepunkt, das Anerkennen von Aktionen anderer und die Antizipation nachfolgender Geschehnisse (Pape 2004, S. 161f.).

- Die *Episode* stellt eine Reihe von mehreren Aktionen mit angebbarem Anfang und Ende dar. Hierbei geht es weniger um eine Vollständigkeit, als vielmehr um die Verbindungen zwischen den einzelnen Aktionen, die beteiligte Akteure zwischen ihnen herstellen. Episoden unterscheiden sich in ihrer Agenda und ihrem Rhythmus. So sind das Aufstellen der Agenda, das Rhythmisieren der Episode und die Schaffung eines Überblicks zentrale Aspekte der Episode (Pape 2004, S. 165f.).

- Als *Strukturen* werden Konstellationen aus verallgemeinerbaren Verfahrensweisen (Regeln), allokativen und autoritativen Ressourcen sowie örtlichen und zeitlichen Regionalisierungsweisen (Struktureigenschaften) verstanden. Auf diese Strukturen wird in einzelnen und zusammenhängenden Aktionen wiederholt Bezug genommen. Merkmale von Strukturen sind Struktureigenschaften, Ressourcen und Regeln. Im alltäglichen Handeln, d. h. in den Aktionen, dienen sie der expliziten Veränderung und (Re-)Produktion (Pape 2004, S. 168f.).

Mit der Konzeption einzelner und zusammenhängender Aktionen können Bedingungen und Konsequenzen für die Organisation der Softwarenutzung aufgezeigt und gestalterisch angewendet werden. „Die wiederholte Bezugnahme auf bestimmte Bedingungen und Konsequenzen ermöglicht es darüber hinaus, auf Strukturen einzugehen, die eine gewisse Relevanz über verschiedene Situationen hinweg aufweisen“ (Pape 2004, S. 168).

Gegenüber dem Analyse- und Gestaltungsrahmen konzentriert sich der Softwarebereitstellungsansatz eASP auf die Organisation der Softwarebereitstellung. Er wird durch die Angabe u. a. von Aufgaben, Akteuren, Beziehungen, Kosten und eines Vorgehens wesentlich konkreter und anschaulicher als der Analyse- und Gestaltungsrahmen zur Organisation der Softwarenutzung (siehe Kapitel 10). Pape (2004, S. 171) verzichtet, aus Gründen der Begrenztheit, auf eine konzeptionelle Angabe von Aufgaben und Akteuren und nutzt nicht deren Vorteil der Beispielhaftigkeit. Insofern unterscheiden sich die beiden Ansätze thematisch und in ihrer Konkretisierung und Anschaulichkeit. eASP kann allerdings aufgrund seiner Beispielhaftigkeit als Weiterführung bzw. als Ausgestaltung des Analyse- und Gestaltungsrahmens in Bezug auf die Softwarebereitstellung gewertet werden.

2.3.7 Infrastrukturen

Der Begriff „Infrastruktur“ (vgl. Bolle 1965; Duden 1980; Brockhaus 1989; Bleek 2004) bezeichnet:

1. die Gesamtheit von bodenständigen militärischen oder strategisch wichtigen Anlagen (Kasernen, Flugplätze, Brücken usw.),

2. den erforderlichen wirtschaftlich-organisatorischen Unterbau einer hoch entwickelten Wirtschaft zur Versorgung und Nutzung eines bestimmten Gebietes oder

3. „eine einer Gruppe von Personen zur Verfügung stehenden Zusammenfassung von Personen, Organisationen, Gerätschaften, Installationen, Regelungen, Standards und damit verbundenen Dienstleistungen, die zur Umsetzung von Aktivitäten langlebig zur Verfügung steht und als selbstverständlich betrachtet werden kann“ (Bleek 2004, S. 40).

Bei der Softwarebereitstellung werden einer Gruppe von Personen von einer Gruppe von Personen Software und entsprechende Dienstleistungen angeboten, so dass die Nutzenden in ihrem Arbeitsalltag unterstützt werden. Die Software wird dabei langfristig zur Verfügung gestellt und kann als selbstverständlich angesehen werden. In diesem Sinne ist die Softwarebereitstellung, insbesondere nach der Definition von Bleek (2004, S. 40), als eine Infrastruktur bzw. Infrastrukturaufgabe zuwerten.

Eine Eingrenzung des Begriffs „Infrastruktur“ stellt der Begriff „Software-Infrastruktur“ dar. Zur Einordnung und Abgrenzung der Softwarebereitstellung bzw. des Ansatzes eASP zur Software-Infrastruktur wird im Folgenden auf das Software-Infrastrukturkonzept von Bleek (2004) eingegangen.

Software-Infrastrukturen
Bleek (2004) leitet aus dem Infrastruktur-Begriff die Software-Infrastruktur ab. Er versteht darunter eine durch die Sichten von Personen konstituierte Infrastruktur, bestehend aus der Zusammenstellung von Personen, Organisationen, Technik und Vereinbarungen und den damit verbundenen Dienstleistungen für die Erledigung von Aufgaben. Im Vordergrund stehen dabei Informationstechnik, Informationsverarbeitung und Informationen. Anwendungsbezogene Softwarebestandteile werden betrachtet, existierende Hard- und Systemsoftware vorausgesetzt (Bleek 2004, S. 43f.).

Bei einer Software-Infrastruktur handelt es sich um die Nutzung und Bereitstellung von verschiedenen Anwendungen und Softwaresystemen, die sich technisch und/oder organisatorisch wechselseitig beeinflussen können. An der Nutzung und Bereitstellung dieser Software sind verschiedene Akteure beteiligt. Zwischen ihnen laufen nicht nur harmonische Prozesse ab, was zwangsläufig zu Irritationen und Störungen im Netzwerk der Software-Infrastruktur führt. Um diese Interferenzen frühzeitig zu erkennen und ihnen entgegenzuwirken, wird ein Interferenz-Management vorge-

schlagen. Es umfasst das Herstellen und Aufrechterhalten eines kontinuierlichen Überblicks, die Pflege von Kommunikation mit Beteiligten, das Identifizieren, Reduzieren, Behandeln, Ausgrenzen und Lindern von Interferenzen. Interferenzen zu vermeiden, wird als ein unerreichbares Ziel angesehen (Bleek 2004, S. 167f.).

Damit Softwareentwicklungsvorhaben nicht durch andere Vorhaben gelähmt werden, wird um ein konkretes Entwicklungsprojekt eine „lokale Hülle“ gelegt. In diesem abgeschlossenen Bereich können klassische Softwareentwicklungsmethoden angewendet werden. Zusätzlich muss im Rahmen einer Software-Infrastruktur der globale Bezug gesichert sein, um das Entwicklungsvorhaben (insbesondere nach Ablauf) in die Software-Infrastruktur einordnen zu können. Zur Durchführung wird ein zyklischer Prozess vorgeschlagen (Bleek 2004, S. 153f.).

Im Software-Infrastrukturkonzept würde sich die Bereitstellung einer bestimmten Software in einer lokalen Hülle konkretisieren. Das Konzept bietet darüber hinaus einen interessanten Rahmen, der es erlaubt, die Bereitstellung einer bestimmten Software in Verbindung zur Bereitstellung anderer Software sowie weiterer Infrastrukturvorhaben und -projekte zu sehen. Bleek (2004) richtet jedoch die Ausgestaltung der lokalen Hülle stark auf Entwicklungsprojekte aus, nicht auf Bereitstellungsprojekte. Da der Ansatz eASP zur Analyse und Gestaltung der Softwarebereitstellung (siehe Kapitel 10) sich grundsätzlich auf Bereitstellung und nicht auf Entwicklung konzentriert, stellt er eine andere Ausgestaltung der lokalen Hülle dar. eASP grenzt sich somit von der Ausgestaltung als Softwareentwicklungsprojekt ab, kann aber als ergänzende Ausgestaltung einer lokalen Hülle in das Software-Infrastrukturkonzept eingeordnet werden.

2.4 Zwischenergebnis

Die Softwarebereitstellung wurde in diesem Kapitel über das Outsourcing und Application Service Providing eingeführt, von anderen, ähnlichen Begriffen abgegrenzt und folgendermaßen definiert:

Die Softwarebereitstellung umfasst technische und organisatorische Aufgaben, welche Nutzenden eine Software zentral als Dienstleistung so zur Verfügung stellt, dass eine gesicherte Nutzung möglich ist.

Damit ist das Thema dieser Arbeit beschrieben und der erste Schritt für die Entwicklung eines Analyse- und Gestaltungsansatzes der Softwarebereitstellung getan. Im nächsten Schritt wird von zwei Fallstudien der Softwarebereitstellung berichtet, um eine empirische Basis aufzubauen, auf die der Ansatz eASP später angewendet werden kann.

- 3 -
Erfahrungen in Softwarebereitstellungsprojekten

Die empirische Basis für diese Arbeit stellen Erfahrungen in der Bereitstellung, Entwicklung und Nutzung der webbasierten Kooperationsplattform CommSy in zwei Kontexten über insgesamt vier Jahre dar. Die Betrachtung der ineinander verwobenen Erfahrungen erfolgt anhand folgender vier Untersuchungsbereiche:

1. die Software CommSy
2. die Bereitstellung von CommSy auf Anbieterseite
3. die Nutzung von CommSy auf Nutzerseite
4. die Entwicklung von CommSy auf Entwicklerseite

Zur Illustration der Untersuchungsbereiche wird in diesem Kapitel zunächst auf die angewendeten empirischen Methoden zur Datenerhebung eingegangen und dann CommSy vorgestellt. Daran schließt sich die Beschreibung der Bereitstellung von CommSy in den zwei Kontexten CommSy@Uni.de und CommSy@WissPro (Fallstudien) an. Angaben über die Nutzung und die Entwicklung fließen in die Beschreibungen der Fallstudien und die Produktbeschreibung sukzessiv ein.

Die in diesem Kapitel dargestellte empirische Basis dient zum einen der Darstellung des praktischen Bedarfs an einem Ansatz zur Analyse und Gestaltung der Softwarebereitstellung und der Ableitung von Anforderungen an diesen Ansatz aus der Praxis. Zum anderen wird auf die hier dargestellten Fallstudien der Ansatz später angewendet. So dient dieses Kapitel auch als Prüfstein, an dem sich der Ansatz messen lassen muss.

3.1 Evaluationsmethodik

Die nachfolgende Beschreibung der Entwicklung, Bereitstellung und Nutzung von CommSy basiert auf Daten, die mit unterschiedlichen Methoden

erhoben wurden. Im Folgenden werden sie separiert nach Entwicklung, Bereitstellung und Nutzung dargestellt, auch wenn sich die separat dargestellten Vorgehensweisen bei der Evaluation und Dokumentation nicht so eindeutig, wie hier angedeutet, trennen lassen. Es entspricht vielmehr der Realität, dass in allen Vorgehensweisen auch Ergebnisse und Aussagen über die jeweils anderen Aspekte erhoben bzw. dokumentiert wurden. So ist die vollzogene Separierung analytischer Natur, um die Bemühungen in der Evaluation verständlich darstellen zu können.

Evaluation der Softwareentwicklung von CommSy

Die Dokumentation der Software CommSy und deren Entwicklung sind im bisherigen Entwicklungsprozess fest verankert. Der in einer Versionsverwaltung vorliegende CommSy-Code stellt eine Dokumentation dar, in der sich Veränderungen detailgetreu nachvollziehen lassen. Zusätzlich zum Code sind während der Entwicklung von CommSy verschiedene Dokumente entstanden, z. B. zur Softwarearchitektur, Klassendiagramme, ER-Diagramme, Programmierguidelines, Papier- und elektronische Prototypen, Prozessablaufpläne usw. Zusätzlich wurden von wichtigen Bestandteilen von CommSy in jeder Version Screenshots erstellt. Diese wurden in Vorträgen und wissenschaftlichen Arbeiten über CommSy verwendet, die ebenfalls zur Dokumentation von CommSy beitragen (vgl. Jackewitz 2000).

Das zugrunde liegende didaktische Konzept, die daraus abgeleiteten Designkriterien, die Softwarearchitektur und die Benutzeroberfläche zeichnen CommSy aus. Sie sind in Protokollen von Workshops und Diskussions- bzw. Arbeitstreffen dokumentiert und später in Artikeln auf Tagungen und Konferenzen veröffentlicht worden (vgl. Jackewitz u. a. 2002c; Hankel u. a. 2003; Janneck u. a. 2003; Jackewitz u. a. 2004).

Der Entwicklungsprozess wurde ebenfalls in verschiedenen Publikationen dokumentiert (vgl. Bleek 2004; Bleek und Pape 2001; Floyd u. a. 2004; Pape 2004; Simon u. a. 2004).

Evaluation der Bereitstellung von CommSy

In dieser Arbeit wird die Bereitstellung von CommSy in zwei Kontexten betrachtet. Zum einen in einer Kooperation eines Wirtschaftsunternehmens mit einer Universität, zum anderen in einem in der Universität verankertem Forschungs- und Entwicklungsprojekt. In beide Bereitstellungskontex-

te war ich stark involviert, so dass meine Person nicht als unabhängiger Beobachter zu werten ist. Mit meinem Handeln hatte ich Einfluss auf den Forschungsgegenstand und habe ihn, im Rahmen meiner Möglichkeiten, gestaltet. Diese Nicht-Unabhängigkeit steht einer wissenschaftlich fundierten Evaluation keineswegs entgegen. Das angewandte Forschungsdesign kann als Aktionsforschung (vgl. Avison u. a. 1999; Fricke 1997) bezeichnet werden, ohne an dieser Stelle die sozialwissenschaftliche Diskussion einer notwendigen Unabhängigkeit im Gegensatz zu einer notwendigen Beteiligung im untersuchten Kontext aufzuarbeiten. Zu einer Vertiefung dieser Diskussion sei auf Argyris u. a. (1985, S. 237), Avison u. a. (1999, S. 94ff.), Frank u. a. (1998, S. 72) und Fricke (1997, S. 5ff.) verwiesen.

Die Datenbasis zur Bereitstellung von CommSy wird u. a. durch ein von mir geführtes Forschungstagebuch (Flick 1999, S. 191) gespeist. Daneben habe ich über 3000 E-Mails, die die Bereitstellung von CommSy zum Thema haben, archiviert. Diese E-Mails sind von mir nach Kontext, Thema bzw. Themenfeld und Beteiligte aufbereitet worden. Beispielsweise sind 754 E-Mails dem Bereitstellungskontext CommSy@Uni.de zu zuordnen. Darüber hinaus habe ich Workshops und Projekttreffen hinsichtlich der Bereitstellung von CommSy protokolliert. Zusätzlich sind zu verschiedenen Treffen Diskussionspapiere und weitere Vorlagen entstanden, im Fall CommSy@Uni.de zwei rechtsverbindliche Kooperationsverträge.

Neben meiner Sichtweise auf die Bereitstellung von CommSy wurden auch Ansichten von anderen Beteiligten in Artikeln zur CommSy-Bereitstellung festgehalten (vgl. Bleek und Pape 2001; Bleek u. a. 2003; Bleek und Jackewitz 2004; Simon u. a. 2004). Ein Kollege und ich haben, im Rahmen der vom Forschungs- und Entwicklungsprojekt WissPro (2003) organisierten wissenschaftlichen Tagung „Medienunterstütztes Lernen – Vernetzt, Verteilt, Verloren?“, eine Gruppendiskussion (Flick 1999, S. 132ff) mit zwölf Experten für die Entwicklung, Nutzung und Bereitstellung von Neuen Medien in der Bildung durchgeführt. Die Ergebnisse wurden in zwei Artikeln dokumentiert (vgl. Pape u. a. 2002b; Pape und Jackewitz 2002).

Evaluation der Nutzung von CommSy

Zur Evaluation der Nutzungs wurden verschiedener quantitativer und qualitativer Methoden im Sinne einer Triangulation (Flick 1999, S. 249ff.) ein-

gesetzt (Strauss u. a. 2003). Dabei setzten quantitative, hypothesenüberprüfende Verfahren auf qualitativen, hypothesengenerierenden Verfahren auf.

Fokusgruppen (Krueger und Casey 2000) bildeten eine breite empirische Basis an Nutzungskontexten und -erfahrungen, um davon ausgehend weitere Forschungsfragen abzuleiten. Mit dem Einverständnis der Beteiligten wurden alle Gruppen- und Einzelinterviews auf Tonträgern aufgezeichnet und anschließend im Wortlaut transkribiert und anonymisiert. Es entstanden mehrere hundert Seiten Text, die im Sinne der Grounded Theory (Strauss und Corbin 1996) ausgewertet wurden (Strauss u. a. 2003, S. 10f.).

Aufbauend auf den Interviewergebnissen wurden Online-Fragebögen entwickelt und Hyperlinks auf diese Fragebögen per E-Mail an Lehrveranstalter und Studierende mit CommSy-Erfahrung geschickt. Bei einer Rücklaufquote von ca. 20% konnten Aussagen über den Einsatzkontext, das Nutzungsverhalten, die didaktische Einbettung, die Zufriedenheit der Nutzenden, CommSy selbst und die Bereitstellung von CommSy gemacht werden (Strauss u. a. 2003, S. 11f.).

Ergänzend zu den gerade beschriebenen Evaluationsmethoden wurde eine anonymisierte Analyse von Logfiles (vgl. Döring 2003) von CommSy-Projekträumen durchgeführt, um „verschiedene Nutzungstypen (z. B. Viel- und Wenignutzer) zu vergleichen, Muster und Regelmäßigkeiten der Nutzung zu erfassen und Nutzungsschwerpunkte und -anlässe zu identifizieren“ (Strauss u. a. 2003, S. 12).

3.2 Kooperationsplattform CommSy

[1]CommSy (für Community System) ist eine webbasierte Kooperationsplattform zur Unterstützung des Studiums an deutschen Hochschulen. Zunächst für projektorientierte Lehrveranstaltungen konzipiert, wurde CommSy zu einer Plattform für universitäre Lerngemeinschaften (Fachbereiche, Studiengänge, Forschungsgemeinschaften usw.) weiterentwickelt, so dass nicht nur einzelne Lehrveranstaltungen, sondern der Kommunikations- und Kooperationsbedarf im gesamten Präsenzstudium mit CommSy unterstützt werden kann.

1 Grundlegende und frühere Gedanken zum Abschnitt 3.2 finden sich bei Jackewitz u. a. (2002c; 2004).

Die Entwicklung von CommSy orientiert sich an didaktisch motivierten Designprinzipien, die Aussagen über die Mensch-Computer-Interaktion, die Mensch-Mensch-Interaktion mittels CommSy und die Einordnung von CommSy in eine Medien-Infrastruktur machen. Sie spiegeln sich in der Gestaltung – von grundlegenden Technologieentscheidungen über die Auswahl von bestimmten angebotenen Funktionalitäten bis hin zum Screen-Design – wider und haben sich in der praktischen Umsetzung bewährt.

An der hier beschriebenen Version von CommSy liegt mein Anteil insbesondere in der Leitung des Entwicklungsprozesses, in der Umsetzung großer Teile der softwaretechnischen Architektur und in der Konzeption des Portals und des Gemeinschaftsraumes (s.u.). Die CommSy-Projekträume (s.u.) habe ich als erster Programmierer maßgeblich gestaltet, diese Aufgabe aber später abgegeben.

Zunächst wird auf die Bedeutung der zwei von CommSy unterstützten Perspektiven auf das Studium für die Gestaltung von CommSy eingegangen, dann der Aufbau und die Funktionalität beschrieben und schließlich werden die übergeordneten Designprinzipien vorgestellt. Die Eignung der Designprinzipien wird mit Ergebnissen aus empirischen Untersuchungen des CommSy-Einsatzes (vgl. Strauss u. a. 2003) untermauert. Zusätzlich fließt in den folgenden Abschnitten die Beschreibung von Implikationen des Dargestellten auf die Softwarebereitstellung ein.

3.2.1 Softwareunterstützung für universitäre Lerngemeinschaften

CommSy nimmt sich der Unterstützung universitärer Lerngemeinschaften unter zwei Perspektiven an (vgl. Jackewitz u. a. 2002a;b):

- In *einzelnen Veranstaltungen* steht die kooperative Verwirklichung von konkreten, praktischen Aufgaben im Vordergrund, die sich die Lernenden selbst gestellt haben. Den dafür notwendigen Arbeits- und Lernprozess planen und verantworten die Lernenden selbst (vgl. Janneck und Krause 2004).

- Für *das Studium insgesamt* sind Bezüge zwischen Lehrveranstaltungen aufzuzeigen und zu stärken. Einzelne Lehrveranstaltungen sollen nicht – wie oft üblich – zusammenhanglos nebeneinander stehen. Sie

sollen Bausteine darstellen, die Studierende in ihrem Studium zu einer kohärenten Einheit zusammensetzen können.

Das so skizzierte Verständnis geht von ganzheitlichem und projektorientiertem Lernen aus, das von den Lernenden eigenverantwortlich gestaltet und von Lehrenden beratend begleitet werden soll (vgl. Cohn und Farau 1993; Frey 2002; Gudjons 1998; Rogers 1974). Die Informationstechnik soll den dafür notwendigen Austausch aller Beteiligten unterstützen, aber nicht vorausgreifend lenken. Damit stellen sich Anforderungen an die Lernenden, an die Lehrenden und auch an die Softwareunterstützung.

- *Lernende* verfolgen aktiv ihre eigenen Lerninteressen und bestimmen selbst, welche Inhalte für sie persönlich bedeutsam sind. Sie werden nicht in erster Linie als „Rezipienten", sondern als „Produzenten" fachlicher Inhalte gesehen, die sie z. B. der (Fachbereichs-) Öffentlichkeit präsentieren und gegenseitig nutzen oder kommentieren können.

- *Lehrende* nehmen die Rolle von Lernbegleitern ein. Sie stellen Räume und Ressourcen bereit, geben Orientierungshilfen, bringen relevante Texte und Materialien ein, strukturieren, kommentieren, bieten ihr Expertenwissen an oder vermitteln Kontakte zu anderen Experten. Auf diese Weise können Lehrende auch ihren eigenen Lernprozess als Teil der gemeinschaftlichen Auseinandersetzung auffassen und ihre spezifischen fachlichen Interessen vertreten.

- Die *Softwareunterstützung* soll zum einen die gleichberechtigte Kommunikation und Kooperation von Lernenden und Lehrenden fördern. Zum anderen sollte sie ein für Lernende, für Lehrende und für die interessierte Öffentlichkeit zugängliches Archiv etablieren, dessen Inhalt von den jeweiligen Mitgliedern einer Lerngemeinschaft selbst bestimmt wird und nicht von einer zentralen Instanz.

Für die Bereitstellung von CommSy bedeutet dies, mit den spezifischen und individuellen Interessen dreier unterschiedlicher Gruppen (Lehrende, Lernende und Gäste) in zwei verschiedenen Kontexten (Veranstaltung, Studium) umzugehen. Dies impliziert ein weit gefächertes Potential an Hilfeanfragen zur Nutzung und an unterschiedlichen Anforderungen an die

Software CommSy. Darüber hinaus ergeben sich hier Restriktionen für die technische Wartung des Servers. Da CommSy als zusätzliches Medium neben der Präsenzlehrveranstaltung im Studium zum Einsatz kommt, gibt es keine klar definierbaren Nutzungszeiträume. Die Analyse von Webserver-Logdaten bestätigte dies. Zwischen drei und vier Uhr nachts waren durchschnittlich die wenigsten Zugriffe verzeichnet, es gab aber keinen Zeitraum ohne Zugriffe. Notwendige Wartungsarbeiten schränken damit potentiell immer Nutzende in ihrer Nutzung ein. Dies könnte Nutzende in kritische Situationen bringen, wenn „morgen eine Prüfung stattfindet“ oder „bis morgen zur Vorbereitung ein Text gelesen werden soll“. So müssen vorgesehene Ausfallzeiten des Servers langfristig geplant und angekündigt werden, damit sich die Nutzenden darauf einstellen können.

3.2.2 Aufbau von CommSy

CommSy[2] unterstützt die skizzierten Perspektiven mit zwei Bereichen:

- Ein einzelner *Gemeinschaftsraum* ist allen Mitgliedern einer Lerngemeinschaft zugänglich und unterstützt mit Strukturierungs- und Archivierungsfunktionen vor allem das Studium in seiner Gesamtheit. Der Begriff „Lerngemeinschaft“ wird hier sehr allgemein für alle Menschen in einer Organisation verwendet, die im weitesten Sinne miteinander lernen wollen, z. B. für Professoren, wissenschaftliche Mitarbeiter und Studierende einer Universität.

- Die *Projekträume* stehen Gruppen mit bis zu 30 Teilnehmenden innerhalb der Lerngemeinschaft zur Verfügung, die zeitlich befristet an einem bestimmten Thema zusammen arbeiten. Sie unterstützen damit vor allem einzelne Lehrveranstaltungen, etwa Seminar- oder Projektgruppen.

Das „Betreten“ eines CommSys, sei es nun der Projektraum oder der Gemeinschaftsraum (technisch gesehen die Authentisierung eines Benutzenden gegenüber dem System), erfolgt über das Portal (siehe Abbildung 3).

2 Ich beziehe mich hier auf die CommSy-Version 2.2, die zum Zeitpunkt der Fertigstellung dieses Kapitels aktuell war. In den CommSy-Versionen 2.0 und 2.1 werden teilweise andere Begrifflichkeiten verwendet. In älteren Versionen wurden nur einzelne Lehrveranstaltungen mit Projekträumen unterstützt.

Das Portal ermöglicht der Gemeinschaft, sich der interessierten Öffentlichkeit zu präsentieren und bietet einen prominenten Zugang zu Hilfethemen, wie z. B. die Beschreibung des Systems oder Hinweise zur Moderation eines Projektraums. Es dient unerfahrenen Nutzenden und Interessierten als Orientierung. Gäste haben auf dem Portal außerdem die Möglichkeit, die Mitgliedschaft zu beantragen.

Abbildung 3: Portal eines CommSys (Version 2.2)

Öffentlichkeit
Während Projekträume nur für die jeweiligen Teilnehmer einer (Lehr-)Veranstaltung zugänglich sind, steht der Gemeinschaftsraum auch Gästen offen. In CommSy werden daher drei Öffentlichkeitsebenen unterschieden: die Projektöffentlichkeit in einem Projektraum, die Gemeinschaftsöffentlichkeit im Gemeinschaftsraum sowie die Weltöffentlichkeit in Form des eingeschränkten Zugangs zum Gemeinschaftsraum für Gäste.

- *Projektöffentlichkeit*: Ein Projektraum bietet einer Gruppe innerhalb der Lerngemeinschaft die Möglichkeit, in einer geschützten Umgebung zu kommunizieren, sich zu koordinieren und Materialien auszutauschen. Nutzende, die nicht Teilnehmer eines Projektraumes sind

(Gemeinschaftsöffentlichkeit), oder Gäste (Weltöffentlichkeit) haben keinen Zugang.

- *Gemeinschaftsöffentlichkeit*: Alle angemeldeten Nutzer eines CommSys haben Zugang zum Gemeinschaftsraum und im Raum Zugriff auf sämtliche Inhalte. Insbesondere können Materialien, z. B. Arbeitsergebnisse, aus Projekträumen in den Gemeinschaftsraum übernommen werden.

- *Weltöffentlichkeit*: Ausgewählte Inhalte im Gemeinschaftsraum können weltöffentlich, d. h. auch für Gäste, lesbar gemacht werden. Die Lerngemeinschaft kann sich dadurch einer größeren Öffentlichkeit präsentieren. Damit ein Inhalt weltöffentlich wird, ist die Zustimmung eines Redakteurs (s.u.) erforderlich. In Projekträumen wird keine Weltöffentlichkeit zugelassen. Es können aber Inhalte aus einem Projektraum in den Gemeinschaftsraum übernommen und dort weltöffentlich präsentiert werden.

Rechtekonzept
CommSy zielt auf eine freie und verantwortungsvolle Benutzung ab und erreicht dies durch ein „offenes“ Rechtekonzept, das auf gegenseitigem Vertrauen und verantwortlichem Handeln aufbaut. Generell wird nur zwischen Gästen und angemeldeten Benutzern unterschieden:

- *Gäste* dürfen den Gemeinschaftsraum betreten und dort weltöffentliche Inhalte lesen. Sie dürfen aber keine Änderungen bzw. Einträge vornehmen und können keine Projekträume betreten.

- *Angemeldete Benutzer* können den Gemeinschaftsraum betreten und dort alle Inhalte lesen, uneingeschränkt neue Einträge erstellen und ihre eigenen Einträge ändern oder löschen. Insbesondere können sie Veranstaltungen ankündigen und Projekträume eröffnen.

Über diese Unterscheidung hinaus können angemeldeten Benutzern bestimmte Rollen in einem CommSy übertragen werden:

- Angemeldete Benutzer können *Teilnehmer* in Projekträumen werden. Die Teilnahme muss beantragt und nachfolgend von einem Moderator des Projektraumes (s.u.) bestätigt werden. Teilnehmer eines Pro-

jektraums können – wie im Gemeinschaftsraum – Einträge lesen, eigene verfassen, ändern und löschen.

- *Redakteure* haben die Aufgabe, den Gemeinschaftsraum zu betreuen. Dazu gehören insbesondere die Konfiguration hinsichtlich Farbe, Name usw., die Verwaltung von Benutzerkennungen und die Freigabe von Materialien für die Weltöffentlichkeit. Redakteure haben nicht automatisch Zutritt zu allen Projekträumen, sondern dürfen, wie alle anderen Benutzer, nur die Projekträume betreten, in denen sie Teilnehmer sind. Sie haben auch nicht das Recht, fremde Einträge im Gemeinschaftsraum zu ändern oder zu löschen. Die redaktionellen Tätigkeiten werden durch eine zusätzliche Rubrik „Konfiguration“ im Gemeinschaftsraum unterstützt.

- *Moderatoren* haben analog die Aufgabe, die Benutzung eines Projektraums zu moderieren, den Projektraum zu konfigurieren und Teilnehmer zuzulassen bzw. abzulehnen. Moderatoren sind nicht berechtigt, die Einträge anderer Benutzer im Projektraum zu ändern oder zu löschen. Die Moderation eines Projektraums wird durch die zusätzliche Rubrik „Konfiguration“ unterstützt. Anfänglich ist nur der Veranstalter des Projektraums auch Moderator. Er kann aber nach Belieben anderen Teilnehmern die Moderationsrolle übertragen und ggf. selber auf dieses Recht verzichten.

Zusammenfassend kann das offene Rechtekonzept durch zwei Prinzipien beschrieben werden:

- *„Alle dürfen alles“* bedeutet, dass jeder angemeldete Benutzer, der Zugriff auf den Gemeinschaftsraum oder einen Projektraum hat, jeweils alle Beiträge lesen und unbeschränkt neue Einträge erstellen darf, ohne dass es eine Differenzierung zwischen verschiedenen Rollen gibt. Eine Ausnahme bildet die spezielle Rubrik „Konfiguration“.

- *„Urheberrecht“* meint, dass nur der Verfasser eines Eintrags diesen auch ändern oder löschen darf. Andere Benutzer – einschließlich Redakteure oder Moderatoren – können dies nicht. Einzige Ausnahme sind Materialien in Projekträumen, die vom Verfasser für die gemeinschaftliche Bearbeitung freigegeben wurden. Zudem kann jeder Be-

nutzer seine eigenen Beiträge nach eigenem Ermessen ändern oder löschen, ohne darin von der Software beschränkt zu werden.

Das offene Rechtekonzept ist ein zentrales Merkmal von CommSy. Dabei wird nicht von einer naiven Sicht auf Kooperation ausgegangen. Konflikte werden als notwendiger und unvermeidlicher Teil eines jeden Arbeits- und Lernzusammenhangs gesehen. Die Entwickler von CommSy glauben nicht, dass sie durch Softwaremechanismen vermieden oder ungeschehen gemacht werden können, sondern sind überzeugt, dass sozial verhandelt werden muss. Es ist beispielsweise keine Lösung, „unangemessene“ Beiträge durch einen privilegierten Moderator kommentarlos löschen zu lassen. Die vermeintliche Unangemessenheit sollte in der Gruppe thematisiert werden. CommSy ermöglicht durch das offene Rechtekonzept die gleichberechtigte und uneingeschränkte Nutzung für alle. Durch Rollen werden die Benutzer in den Aufgaben unterstützt, die sie für eine bestimmte Gruppe übernehmen, ohne dass damit eine Machtposition verknüpft ist.

Für die Bereitstellung stellt das offene Rechtekonzept eine Schwierigkeit dar, da zwischen Professoren und Studierenden keine eindeutige Verknüpfung zu den Rollen Lehrende und Lernende gezogen werden kann. Lehrende und Lernende bzw. Professoren und Studierende haben die gleichen Rechte. So ist z. B. nicht auf den ersten Blick erkennbar, ob eine Veranstaltung von einem Studierenden oder einem Professor eingerichtet wurde oder ob die Supportanfrage zur Einrichtung eines Projektraums von einem Studierenden oder einem Professor stammt. Umgekehrt lässt das Wissen um den Status des Nutzenden (Professor oder Studierender) keinen Rückschluss auf die Benutzung als Lehrender oder Lernender zu. Zur Betreuung der Nutzenden ist dieses Wissen allerdings sehr hilfreich, um die Beweggründe des Nutzenden zu antizipieren und angemessen in Inhalt und Form reagieren zu können.

Allgemeiner Aufbau

Inhalte sind im Gemeinschaftsraum und in den Projekträumen in Rubriken (Neuigkeiten, Termine, Materialien, Diskussionen, Personen, Gruppen, Themen, Institutionen) untergliedert. Prinzipiell sind alle Rubriken gleich aufgebaut.

Jede Rubrik hat eine Übersicht, in der die wichtigsten Informationen zu allen Einträgen einer Rubrik archivartig dargestellt werden. Archivartig

bedeutet, alle Einträge sind generell nach Aktualität sortiert. Alte Einträge werden am Ende bzw. unten dargestellt, sie verlassen im Laufe der Benutzung den Sichtbarkeitsbereich. Ein Zugriff ist aber weiterhin möglich. Die Übersichtsseite bietet zudem Sortier- und Suchmöglichkeiten.

Zu jedem Eintrag gibt es eine Detailansicht, die alle Informationen zu diesem Eintrag anzeigt, insbesondere die Angaben, wer welchen Eintrag wann erstellt und ggf. geändert hat. Einzelne Einträge können auf vielfältige Weise miteinander verlinkt und in Beziehung gesetzt werden (s.u.).

Der Gemeinschaftsraum und die Projekträume besitzen darüber hinaus eine Einstiegsseite (Home), auf der aktuelle Änderungen präsentiert werden. Sie dient weiterhin als „Wegweiser“ zu den angebotenen Rubriken. Regelmäßiges Aufsuchen der Einstiegsseite(n) hält die Nutzenden über aktuelle Geschehnisse in der Gemeinschaft und in den Projekträumen auf dem Laufenden.

CommSy-Gemeinschaftsraum

Abbildung 4: Einstiegsseite des Gemeinschaftsraumes (Version 2.2)

Der Gemeinschaftsraum dient einer Lerngemeinschaft zur Information und Strukturierung nach innen und gleichzeitig als „Visitenkarte“ für die Darstellung nach außen. Er gliedert sich in die Rubriken Ankündigungen, Veranstaltungen, Materialien, Themen, Institutionen, Personen und der nur für Redakteure sichtbaren Rubrik „Konfiguration“. Darüber hinaus haben angemeldete Benutzer die Möglichkeit, zu allen Einträgen, ausgenommen Personen, Anmerkungen zu schreiben.

- *Ankündigungen* weisen Nutzende des Gemeinschaftsraums auf aktuelle Ereignisse und interessante Informationen hin. Sie haben den Charakter eines „Schwarzen Bretts“, an das jedes Mitglied Nachrichten anschlagen kann. Wichtige redaktionelle Ankündigungen können auf der Portalseite veröffentlicht werden.

- *Veranstaltung*: Eine Beschreibung der (Lehr-)Veranstaltungen sowie den Zugang zu den Projekträumen bietet die Rubrik „Veranstaltungen“. Hier können Lehrveranstaltungen im Curriculum, aber auch extracurriculare Gruppen eingetragen sein, deren Teilnehmer sich z. B. zusammen auf eine Prüfung vorbereiten. Zu jeder Veranstaltung kann ein Projektraum eröffnet werden. Zusätzlich und unabhängig vom Projektraum können der Veranstaltung Materialien zugeordnet werden, die z. B. als Vorbereitung auf ein Seminar zu lesen sind oder Ergebnisse einer Projektarbeit darstellen. Eine Veranstaltung kann auch mit Themen und Institutionen verknüpft und so in einen inhaltlichen und organisatorischen Kontext gestellt werden.

- *Materialien*: Ein Material kann ein einfacher Literaturhinweis sein, es können aber auch elektronische Dokumente gespeichert oder ganze Texte in CommSy geschrieben werden. Teilnehmer können Materialien untereinander verknüpfen, z. B. um vorhandene Arbeiten unter einer neuen Sichtweise zusammenzustellen und zu kommentieren. Materialien können darüber hinaus Themen, Institutionen und Veranstaltungen zugeordnet werden. Sie können vom Gemeinschaftsraum in Projekträume kopiert werden, um z. B. als Grundlage der Projektarbeit zu dienen. Ergebnisse, die in Projekträumen erarbeitet wurden, können wiederum im Gemeinschaftsraum präsentiert werden. Auch der Transport von Materialien zwischen Projekträumen ist möglich. Langfristig entsteht im Gemeinschaftsraum ein Archiv, das

sich die Mitglieder der Lerngemeinschaft zu verschiedenen Anlässen, z. B. Projektarbeit, Prüfungsvorbereitung, Orientierung im Studium, und unter verschiedenen Perspektiven, Themen, Personen und organisatorischen Einheiten, erschließen können.

- *Themen* und *Institutionen*: Veranstaltungen und Materialien können unter einer thematischen und organisatorischen Perspektive strukturiert werden. Wie überall kann jeder angemeldete Benutzer neue Themen oder Institutionen eintragen: eine Struktur wird nicht vorgegeben und nicht aufgezwängt, sondern muss von der Lerngemeinschaft aktiv erarbeitet und gepflegt werden.

- *Personen*: Wissen ist immer stark mit Personen verknüpft. So kann jedes Mitglied in der Rubrik „Personen“ eine persönliche Seite einrichten und sich den anderen Mitgliedern vorstellen und Kontaktmöglichkeiten (E-Mail, Telefon usw.) angeben. Die persönliche Seite eines Benutzers ist darüber hinaus eine weitere Möglichkeit, die Inhalte des Gemeinschaftsraums zu erschließen, denn hier werden alle Materialien, die ein Benutzer einträgt, alle Veranstaltungen, die er (mit-)organisiert und alle Themen und Institutionen, denen er sich zuordnet, aufgelistet.

- *Konfiguration*: Die Redaktion eines Gemeinschaftsraums wird durch die spezielle Rubrik „Konfiguration“ unterstützt. Hier werden Einstellungen (Name, Farbgebung, Anpassung von Beschreibungs- und E-Mailtexten) für das gesamte CommSy und den Gemeinschaftsraum vorgenommen, Benutzerkennungen verwaltet und Materialien im Gemeinschaftsraum nach entsprechender urheberrechtlicher Prüfung für den weltweiten Zugriff freigegeben. Zudem besteht die Möglichkeit, Projekträume im Falle eines Missbrauchs zu sperren.

Der Gemeinschaftsraum bedeutet für die Bereitstellung von CommSy die Übernahme von redaktionellen Tätigkeiten. Speziell die zur Strukturierung des Gemeinschaftsraums verwendeten Rubriken „Themen“ und „Institutionen“ erfordern eine erhöhte Aufmerksamkeit. Zum einen müssen Nutzende motiviert werden, verschiedene Perspektiven auf vorhandene Inhalte im System zu explizieren, zum anderen darf kein Wildwuchs dem Auffinden von Inhalten entgegenstehen. Darüber hinaus müssen die Redakteure Materialien vor dem weltweiten Zugriff urheberrechtlich überprüfen. Beides

erfordert von den Redakteuren eine fachliche Expertise im mit CommSy unterstützten Kontext.

CommSy-Projektraum

Abbildung 5: Einstiegsseite des Projektraums (Version 2.2)

Projekträume unterstützen die Durchführung projektorientierter Veranstaltungen (Gudjons 1998; Frey 2002). Für Kommunikation und Koordination in Projektgruppen und den Umgang mit unterschiedlichen Arbeitsmaterialien stehen folgende Rubriken zur Verfügung: Neuigkeiten, Termine, Diskussionen, Materialien, Personen und Gruppen sowie die Rubrik „Konfiguration“ für Moderatoren.

- *Neuigkeiten* und *Termine*: Zur Koordination können innerhalb eines Projektraumes Neuigkeiten und Termine angekündigt werden. Die Neuigkeiten entsprechen den Ankündigungen im Gemeinschaftsraum und machen andere Teilnehmer auf wichtige Ereignisse oder neue Einträge innerhalb des Projektraums aufmerksam. Termine werden

zusätzlich mit Angaben zu Ort und Zeit versehen, halten in der Projektgruppe verabredete Termine fest oder weisen auf externe Veranstaltungen hin.

- *Diskussionen* und *Anmerkungen*: Der Kommunikationsunterstützung dienen die Rubrik „Diskussionen“ sowie die Möglichkeit, mit einer Anmerkung Einträge im jeweiligen Kontext direkt zu kommentieren. Diskussionen haben ein bestimmtes Thema, werden von einem Teilnehmer eingeleitet und können von diesem mit einer Zusammenfassung explizit beendet werden. Weitere Beiträge werden chronologisch dargestellt.

- *Materialien*: Die Rubrik „Materialien“ entspricht im Wesentlichen ihrem Pendant im Gemeinschaftsraum. Darüber hinaus ist es möglich, in einem Projektraum mit einem kooperativen Editor einfache Dokumente gemeinsam zu erstellen. Materialien können mit beliebigen Einträgen in einem Projektraum verknüpft werden, so dass beispielsweise ein Protokoll bei dem betreffenden Termin oder ein Literaturhinweis bei einem Diskussionsbeitrag, in dem eben diese Literatur diskutiert wird, angefügt ist. Im günstigsten Fall sind in einem Projektraum sämtliche Grundlagen, Zwischen- und Endergebnisse, mit und an denen eine Gruppe arbeitet, für alle Teilnehmer jederzeit und überall verfügbar.

- *Personen*: Wie im Gemeinschaftsraum können die Mitglieder eines Projektraumes eine persönliche Seite gestalten und Kontaktinformationen angeben. So sind Telefonnummern und E-Mail-Adressen jederzeit abrufbar, falls es die Projektarbeit erfordert, andere Teilnehmer zu kontaktieren.

- *Gruppen*: In der Rubrik „Gruppen“ können sich Teilnehmer innerhalb eines Projektraums zu Kleingruppen zusammenschließen und so ihre Gruppenstruktur im Projektraum rekonstruieren. Inhalte des Projektraumes können als bedeutsam für eine oder mehrere Gruppen gekennzeichnet und so strukturiert werden. Die Gruppenstruktur ist rein informativ und nicht mit speziellen Zugriffsrechten verbunden. Teilnehmer können sich selbst beliebig vielen Gruppen zuordnen.

- *Konfiguration*: Die Rubrik „Konfiguration“ dient der Teilnehmerverwaltung und der Anpassung des Projektraumes an die Bedürfnisse des Projekts. Es gibt verschiedene Optionen, die von Moderatoren konfiguriert, jedoch in der Projektgruppe verhandelt werden sollten. Dazu gehören die Wahl eines Namens für den Projektraum und die an der Benutzungsschnittstelle verwendete Sprache (Deutsch oder Englisch), die Farbgebung und die Anordnung der Rubriken auf der Homepage, um so den Projektraum den verschiedenen Erfordernissen bei unterschiedlichen Kooperationsstilen anzupassen. Die Anpassungen wirken sich jeweils für alle Teilnehmer des Projektraums aus, Personalisierungen sind nicht möglich (vgl. Finck u. a. 2004).

Die Projekträume stellen bei der Bereitstellung von CommSy eine Besonderheit dar, denn diejenigen, die CommSy bereitstellen, haben aufgrund der offenen Rechtestruktur und der Abgeschlossenheit der Projekträume keinen Einblick in die Projekträume. Dies führt zu Schwierigkeiten, wenn sich Fragen von Nutzenden direkt auf spezielle Projekträume beziehen, z. B. „Warum kann ich das Material X aus dem Projektraum Y nicht herunterladen?“ oder „Warum kann ich nicht in den Projektraum Z hinein?“. Durch das offene Rechtekonzept wird eine Betreuungsindirektion aufgebaut:

$$Redaktion \Rightarrow Veranstalter/Moderatoren \Rightarrow Teilnehmer$$

Die Redaktion eines Gemeinschaftsraums kann Teilnehmenden von Projekträumen bei speziellen Fragen zu einem Projektraum nicht helfen. Daraus folgt, dass die Veranstalter bzw. Moderatoren eines Projektraums eine erhöhte Betreuungsverantwortung gegenüber den Teilnehmenden in ihrem Projektraum haben. Aufgabe der Redaktion wiederum ist es, die Veranstalter/Moderatoren mittels eines „Train-the-Trainer“-Konzeptes in die Lage zu versetzen, dieser Betreuungsverantwortung gerecht zu werden.

3.2.3 Übergeordnete Designprinzipien

CommSy liegen drei zentrale Designprinzipien zugrunde, die bei der Softwareentwicklung handlungsleitend waren und sind (vgl. Pape u. a. 2002a). Die Designprinzipien adressieren die Interaktion der Benutzer mit der Software, die Interaktion der Benutzer untereinander sowie die Einbettung von CommSy in eine größere Kooperations-Infrastruktur. Die Designprinzipien sind:

- Einfachheit in der individuellen Benutzung,
- verantwortungsvolle Benutzung in der Gemeinschaft und
- Einbettung in einen Medienmix.

Im Folgenden werden diese Designprinzipien beschrieben und deren konkrete Umsetzung aufgezeigt.

Einfachheit in der individuellen Benutzung
Durch das Prinzip „Einfachheit in der individuellen Benutzung“ tritt die Benutzung und Administration der Technik hinter die Auseinandersetzung mit den Inhalten zurück. Lernende und Lehrende haben ein berechtigtes Interesse daran, dass sie die begrenzte Zeit in einer Lehrveranstaltung nicht auf das Erlernen der Handhabung einer Software oder deren Installation und Konfiguration, sondern auf die Auseinandersetzung mit den fachlichen Inhalten verwenden können. In der Gestaltung von CommSy wird Einfachheit in der individuellen Benutzung erreicht durch:

- *Aufgabenangemessene Funktionalität*: CommSy hat einen auf die Unterstützung von individuellem und gemeinschaftlichem Wissensaufbau abgestimmten Funktionsumfang. Die einzelnen Funktionalitäten sind offen gestaltet (vgl. DeMichelis 2003), so dass sie flexibel verwendet werden können.
- *Einfacher Aufbau*: Die Rubriken sind konsistent strukturiert. Damit ergibt sich ein wiederkehrendes Benutzungsschema, das leicht erlernbar ist.
- *Einfaches Layout*: Die Präsentation von Informationen ist zweckmäßig, denn auf grafische Elemente wird weitgehend verzichtet. Dadurch ist die Größe der einzelnen HTML-Seiten kleiner und der Bildschirmaufbau auch bei langsamen Internetverbindungen zügig.
- *Einfacher Zugriff*: Der Zugriff auf CommSy erfolgt über einen zu W3C-Standards kompatiblen Webbrowser, so dass clientseitig kein zusätzlicher Installations- und Konfigurationsaufwand anfällt. Spezielle Plug-ins, Cookies, Java oder JavaScript sind nicht erforderlich. Somit können auch restriktiv konfigurierte Clients, z. B. in Rechnerpools, problemlos verwendet werden.

- *Einfache Technologie*: CommSy ist in PHP implementiert und verwendet eine MySQL-Datenbank zur Datenhaltung. Beide Technologien sind unter Open Source-Lizenzen verfügbar. PHP ist recht einfach zu erlernen, so dass sich interessierte Entwickler schnell in die CommSy-Entwicklung einbringen können. Darüber hinaus bieten viele Internet-Service-Provider (ISPs) PHP und MySQL kostengünstig an.

In der Einschätzung der CommSy-Nutzenden ist die Umsetzung des Prinzips „Einfachheit in der individuellen Benutzung“ gelungen: Die in der Evaluation (Strauss u. a. 2003) der Nutzung befragten Studierenden bewerteten CommSy mehrheitlich als „einfach“ oder „sehr einfach“ zu benutzen. Auch die Lehrenden teilten diese Einschätzung; zudem gaben viele von ihnen die unkomplizierte Benutzbarkeit und den angemessenen Funktionsumfang explizit als den Vorzug an, der den Ausschlag zugunsten von CommSy im Vergleich zu Konkurrenzprodukten gab.

Diese positive Einschätzung der Benutzbarkeit korrespondiert mit den Ergebnissen der Erhebung aufgetretener Benutzungsprobleme: Sowohl von Professoren und wissenschaftlichen Mitarbeitern als auch von Studierenden werden kaum Handhabungsprobleme, die auf eine mangelnde Benutzbarkeit zurückzuführen sind, beschrieben. Neben technischen Problemen (Erreichbarkeit der Server, Qualität der Internetverbindung etc.) wurden v.a. Probleme beschrieben, die in fehlender Moderation oder mangelnder Einbettung in den Veranstaltungskontext begründet sind.

Zudem spielt die Benutzbarkeit des Systems eine entscheidende Rolle für die Zufriedenheit mit dem Systemeinsatz. Je mehr die Teilnehmer der Aussage zustimmen, CommSy sei einfach zu benutzen, desto zufriedener sind sie mit dem Systemeinsatz. Umgekehrt sind häufige Probleme bei der CommSy-Nutzung negativ mit der Zufriedenheit korreliert[3]. Daraus lässt sich folgern, dass die einfache und unmittelbare Benutzbarkeit einer Lernplattform ein entscheidender Faktor für den Erfolg des Einsatzes ist.

Für die Bereitstellung bedeutet das Prinzip „Einfachheit in der individuellen Benutzung“ auf der einen Seite eine Entlastung bei der Beantwortung von Supportanfragen zu Handhabungsproblemen. Auf der anderen Seite folgt aus diesem Prinzip, dass die Bereitstellung, insbesondere der tech-

3 Die entsprechenden Rangkorrelationen sind allesamt auf dem Niveau von $\alpha = 0.01$ signifikant (Strauss u. a. 2003)

nische Betrieb und die Betreuung der Nutzenden bei Handhabungsfragen, nicht von den Nutzenden (Lehrenden und Lernenden) übernommen werden dürfen. Diese Aufgaben müssen andere Personen leisten.

Verantwortungsvolle Benutzung in der Gemeinschaft
CommSy soll Eigeninitiative und Verantwortlichkeit in der Gruppe unterstützen und dadurch die Auseinandersetzung mit Inhalten und Perspektiven anderer Mitglieder ermöglichen. So kann Wissen gemeinschaftlich erworben und im interdisziplinären Austausch erprobt und gefestigt werden. CommSy unterstützt die verantwortungsvolle Benutzung in der Gemeinschaft durch:

- *Geschlossene Benutzergruppe*: CommSy wendet sich mit seinen unterschiedlichen Räumen jeweils an bestimmte Benutzergruppen, z. B. an Angehörige eines Fachbereichs oder an Teilnehmer einer Lehrveranstaltung. Dadurch wird erreicht, dass der Adressatenkreis teilweise persönlich bekannt ist, was den Aufbau einer vertrauensvollen Arbeitsbeziehung unterstützt.

- *Keine anonymen Beiträge*: Mit jedem Eintrag wird der Name des Autors gespeichert und in der Detailansicht des Eintrags angezeigt. So wird die Übernahme von Verantwortung für eigene Beiträge eingefordert und die Zuordnung von Beiträgen zu Mitgliedern unterstützt. Verwirrung durch anonyme oder automatisch generierte Beiträge, deren Urheber nicht erkennbar sind, wird dadurch vermieden.

- *Offenes Rechtekonzept*: Auf die Implementierung einer komplexen Rechteverwaltung wurde bei CommSy absichtlich verzichtet, denn freie und uneingeschränkte Benutzung fördert die Eigeninitiative. So wird der Aufbau von bestimmten Teamstrukturen und Rollenverteilungen durch das System nicht präjudiziert.

- *Betonung der Gemeinschaft*: Im Gemeinschafts- bzw. Projektraum werden allen Mitgliedern jeweils dieselbe Ansicht und dieselben Inhalte präsentiert. Dies ermöglicht eine transparente gemeinschaftliche Nutzung und erleichtert die Kommunikation über Inhalte und die Orientierung im virtuellen Raum. „Individualisierbarkeit“ wird nicht im Sinne einer Anpassbarkeit an die Bedürfnisse einzelner Personen

verstanden (ISO 1996), sondern als Anpassbarkeit an die Bedürfnisse der ganzen Gemeinschaft.

In ihrer Bewertung des Systemdesigns loben die in der Evaluation der Nutzung (Strauss u. a. 2003) befragten Lehrenden ausdrücklich das „demokratische“ Rechtesystem von CommSy. Sie weisen allerdings explizit darauf hin, dass bestehende asymmetrische soziale Positionen allein durch den Einsatz einer Software nicht verändert werden können. Die angemessene didaktische Einbettung einer Software ist die entscheidende Voraussetzung für den zufrieden stellenden Einsatz.

Während ein Großteil der Professoren und wissenschaftlichen Mitarbeitern die gemeinschaftsstiftende Funktion von CommSy betont, wegen der größeren Nähe, die sie hierdurch zu ihren Studierenden empfinden, sowie der stärkeren Transparenz der Lernprozesse, fällt das Urteil der Studierenden differenzierter aus (Strauss u. a. 2003). Die Mehrheit von 65% bewertet den Einsatz von CommSy als „nicht“ oder „wenig“ gemeinschaftsstiftend. Generell haben die Teilnehmer im Rahmen ihrer Arbeit mit CommSy Anonymität, wie sie häufig im Zusammenhang mit virtueller Kommunikation beklagt wird, kaum empfunden. Dies ist sicherlich auch darin begründet, dass CommSy nicht für den Einsatz in rein virtuellen Veranstaltungen, sondern zur Unterstützung von Präsenzveranstaltungen konzipiert wurde und so hauptsächlich verwendet wird.

Für die Bereitstellung bedeutet das Prinzip „Verantwortungsvolle Benutzung in der Gemeinschaft“, dass nicht diejenigen, die die Bereitstellung von CommSy leisten, für die Inhalte verantwortlich sind, sondern die Nutzenden selbst. Diese Entlastung auf rechtlicher Ebene (Urheberrecht, Datenschutz) zieht eine weitere Entlastung mit sich. Bei Fragen zu Einträgen können sich Nutzende direkt an den Einsteller wenden, sie müssen sich nicht an die Bereitstellung wenden. Darüber hinaus liegt die Betreuungsverantwortung während der Präsenzveranstaltung, analog zu den Projekträumen im System, bei den Veranstaltern und damit außerhalb der Reichweite der Redakteure bzw. weiterer Personen, die die Bereitstellung von CommSy leisten.

Einbettung in einen Medienmix

Die Entwickler von CommSy halten ein allumfassendes Werkzeug für die universitäre Lehre, mit dem alle Kommunikationsbedürfnisse abgedeckt

werden können, für nicht machbar und nicht erstrebenswert. Die angebotene Funktionalität von CommSy orientiert sich daher an den zwei genannten Perspektiven auf das Studium: einzelne Lehrveranstaltungen und das Studium insgesamt (vgl. Abschnitt 3.2.1). Das bedeutet umgekehrt, dass ggf. zusätzliche Werkzeuge herangezogen werden müssen, um bestimmte Aufgaben zu erledigen.

Die empirischen Ergebnisse zeigen, dass die Verwendung unterschiedlichster Kommunikationsmedien eine Realität darstellt (Strauss u. a. 2003). Vor allem E-Mail wurde als zusätzliches Kommunikationsmedium benutzt. Andere Medien wie z. B. klassische (papierbasierte) Seminarordner, Mailinglisten oder Websites spielten demgegenüber eine untergeordnete Rolle. Zusätzliche Tools zur synchronen Diskussion wie Chat und Messaging fanden kaum Verwendung. Der persönliche Kontakt und die unmittelbare Kommunikation über das Telefon galten den Studierenden hingegen als unverzichtbar.

Der Umgang mit einem Medienmix verlangt sicherlich ein erhöhtes Maß an Medienkompetenz (vgl. Schiersmann u. a. 2002). Die Benutzer müssen einschätzen, welches Medium für ein Kommunikationsbedürfnis in einer konkreten Situation angemessen ist und wie sie es im gewählten Medium konstruktiv umsetzen können. Im Sinne eines Probehandelns ergibt sich in der universitären Lehre die Gelegenheit, die Brauchbarkeit unterschiedlicher Medien in der Gruppenarbeit zu erproben und dadurch Medienkompetenz aufzubauen.

In der Bereitstellung tauchen durch das Prinzip „Einordnung in einen Medienmix“ Fragen zu anderen Medien und Schnittstellen auf. Bei Bedarf müssen ggf. weitere softwaretechnische Medien bereitgestellt und auf deren Kompatibilität geachtet werden. Ein Medienmix vergrößert die Komplexität in der Bereitstellung.

3.2.4 Bemerkungen

Auch wenn in diesem Abschnitt vor allem Produktmerkmale beschrieben wurden und nur am Rande auf empirisches Material eingegangen worden ist, kann die Qualität und der Nutzen einer Software nicht an den vorhandenen „Features“ und dem „schicken Design“, also den direkt an der Software ablesbaren Merkmalen, gemessen werden. Diese geben nur begrenzt Hinweise auf die Nutzbarkeit einer Software und sollten daher nicht zur

Bewertung herangezogen werden. Stattdessen sind die reale Nutzung und die Zufriedenheit der Nutzenden Qualitätskriterien, an denen sich Software messen lassen muss (vgl. Floyd 1994; Siefkes 2003).

CommSy wurde und wird seit Oktober 1999 in über 500 Veranstaltungen von mehr als 6.000 Benutzern an Hochschulen und Fachbereichen sowie im Schulunterricht, in der Lehrerfortbildung und von extracurricularen Lern- und Studiengemeinschaften verwendet (Stand: 31.12.2003). Die durchgeführte quantitative und qualitative Evaluation des Einsatzes von CommSy zeigt auf, dass ein Großteil der Nutzenden CommSy als für sie nützlich und einfach zu handhaben bewertet hat und insgesamt mit CommSy zufrieden war. Diese Zufriedenheit ist allerdings nicht nur auf die Software CommSy, sondern auch auf die Bereitstellung von CommSy zurückzuführen (Strauss u. a. 2003).

3.3 Zwei Fallstudien

Vor der Präsentation der Fallstudien „CommSy@Uni.de“ und „CommSy@WissPro“ wird die Entwicklung von CommSy als Vorschichte geschildert, da sie Auswirkungen auf die Fallstudien hat. Die Beschreibung der Fallstudien und des Werdegangs von CommSy sind bewusst als Erzählung und anonymisiert gestaltet, um an dieser Stelle noch keine Art der wissenschaftlichen Auswertung vorwegzunehmen. Die Auswertung erfolgt in den Kapiteln 6, 7, 8 und 9. Hinsichtlich der Anonymisierung wurden nicht, wie üblich, Personennamen verfremdet, sondern gänzlich auf sie verzichtet. Stattdessen findet im Folgenden die Benutzung von Rollenbezeichnungen Verwendung.

3.3.1 Vorgeschichte

Die Entstehungsgeschichte von CommSy (vgl. Bleek 2004; Pape 2004; Simon et al. 2004), in die ich seit Beginn involviert war und die ich bis heute aktiv begleitet und gestaltet habe, reicht in das Jahr 1999 zurück.

In der Universität Hamburg im Fachbereich Informatik formierte sich, außerhalb des Lehrbetriebs, im Frühjahr 1999 eine kleine Gruppe, bestehend aus einem Hochschullehrer, drei wissenschaftlichen Mitarbeitern, einem externen Promoventen und vier Studierenden. Sie definierte sich über das gemeinsame Interesse an einer kritischen Auseinandersetzung mit dem

Thema „Knowledge Management“. Die Gruppe war durch die individuellen Bedürfnisse (Studienarbeit, Diplomarbeit, Doktorarbeit, wissenschaftliche Reputation) ihrer Mitglieder geprägt.

Auf einer der ersten Sitzungen formulierten die Mitglieder den Bedarf nach einem eignen Bereich im World Wide Web (WWW), welcher abseits der monatlichen Gruppensitzungen als öffentliche Präsentation und gleichzeitig als interne Kommunikationsunterstützung fungieren sollte. Die Mitglieder wollten untereinander ihre Forschungsergebnisse und Ausarbeitungen zum Thema „Knowledge Management“ über den Webbereich austauschen.

Der Arbeitsbereich Softwaretechnik (SWT) des Fachbereichs Informatik stellte der Gruppe einen Netscape Enterprise Webserver zur Verfügung und richtete Zugangsberechtigungen zum Webbereich ein. Eine Webseitenstruktur wurde in HTML implementiert, die gemeinsam editierbare Seiten für Termine und Neuigkeiten und einen eigenen persönlichen Bereich für jedes Gruppenmitglied vorsah. Diese statischen Webseiten sollten mit dem im Fachbereich vorhandenen Webseiteneditor Netscape Composer bearbeitet und auf dem Webserver publiziert werden.

Die Gruppensitzung am 14.07.1999 ergab, dass die Mitglieder den freien Webbereich nicht annahmen. Gründe bestanden in der Schwierigkeit des Publizierens von Dokumenten und in der Verlinkung verschiedener Webseiten. Außerdem wurde festgestellt, dass die zunächst vorgesehenen persönlichen Bereiche dem Wesen einer Gruppe entgegenstehen und deshalb der Webbereich auf solche dysfunktionalen Elemente verzichten sollte.

Bis zur nächsten Gruppensitzung am 23.08.1999 hatten sich drei Gruppenmitglieder des Webbereichs angenommen und beschlossen, ihn mit dynamischen Webseiten neu zu gestalten. Die Skriptsprache PHP wurde mit dem Netscape Enterprise Webserver kombiniert und auf der Sitzung der neue Webbereich (das Ur-CommSy) vorgestellt. Er bestand aus einer Einstiegsseite, welche die letzten zehn Neuigkeiten und die nächsten Termine anzeigte. Über sie gelangte das Gruppenmitglied in die Rubriken Neuigkeiten, Termine und Quellen. Einträge in den Rubriken wurden in einer MySQL-Datenbank gespeichert. In Verbindung mit PHP war es nun möglich, mit dem Webbrowser in die jeweiligen Rubriken Beiträge einzugeben. Die Benutzungshürde „Netscape Composer“ wurde dadurch überwunden. Der neue Webbereich fand positiven Anklang und erhielt den Namen

KnowNet als Abkürzung für Knowledge Network. Die für das System gefundene Bezeichnung übertrug sich auch auf die Gruppe.

Im September 1999 wurden Diskussionsforen, ein Bereich zur Darstellung der beteiligten Personen, ein Chat und die Möglichkeit, an alle Beiträge Anmerkungen und Dateien anzuhängen, gestaltet. Ebenfalls im September konnte der Gruppe KnowNet die Rubrik Wissenspools zur Verfügung gestellt werden. Die Rubrik funktionierte ähnlich wie die Diskussionsforen, war aber weniger zum Diskutieren, sondern mehr zum Archivieren von Beiträgen und Dateien gedacht. Ein Mitglied trug zu dieser Zeit die Verantwortung für Pflege, Fehlerbehebung und Weiterentwicklung des KnowNets.

Abbildung 6: Einstiegsseite des Ur-CommSys KnowNet (Version 0.1)

Während der Lehrplanung zum Wintersemester 1999/2000 hatten zwei Angehörige der Gruppe KnowNet die Idee, das entwickelte Community System in einem Projektseminar einzusetzen. Sie konzipierten aus dem Ur-CommSy die nächste Version, um die Lehrveranstaltung damit zu unterstützen. Der PHP-Code wurde kopiert, eine neue Datenbank aufgesetzt und Änderungen, auf der Grundlage des PHP-Codes des KnowNets, implemen-

tiert. Die Rubrik Wissenspools wurde entfernt, da sie sich nicht eindeutig von den Diskussionsforen abgrenzte. Hinzu kamen die Rubriken Gruppen, Suchen und Intern. Die Rubrik Suchen stellte eine Suchfunktion, beschränkt auf Dateien und Quellen, zur Verfügung. Die Rubrik Intern umfasste eine Sammlung von Hyperlinks zu Informationen rund um das System, u. a. zur Administrationsseite des Webservers, zur offiziellen Webseite von PHP und MySQL usw.

Abbildung 7: Einstiegsseite des 2. CommSys (Version 0.2)

Dem Projektseminar „Virtuelle Gemeinschaften, Intranets, Knowledge Networks“ des Arbeitsbereichs Angewandte und Sozialorientierte Informatik (ASI) im Fachbereich Informatik der Universität Hamburg wurde das Community System Ende Oktober 1999 unter dem Kürzel CS vorgestellt. Im Laufe des Projektseminars änderte sich der Name in CoSy. Ebenfalls im Oktober entschied die Gruppe KnowNet, die Weiterentwicklung des Community Systems aus der Arbeitsgemeinschaft zu lösen und separat zu handhaben.

Zum Betrieb und zur Pflege des Servers sowie zur Weiterentwicklung des Community Systems und zum Benutzersupport fand sich innerhalb des Projektseminars ein Team von Studierenden. Es setzte ein CVS-Repository für die Weiterentwicklung auf, re-implementierte die Diskussionsforen im Sinne von Newsgroups, änderte das Layout und integrierte eine E-Mail-funktionalität.

Im Januar 2000 ging das nächste Community System mit dem Namen PrüVerNet online. Im Rahmen des Projektes zur Einführung einer Prüfungsverwaltungssoftware in die Universität Hamburg (Jackewitz 2000) sollte das PrüVerNet allen Benutzern der neuen Prüfungsverwaltungssoftware ein Forum zum Austausch von Benutzungsproblemen und deren Lösungen sein. Aufgrund des heterogenen Projektumfeldes konnte sich das PrüVerNet nicht etablieren und wurde noch im selben Jahr wieder abgeschaltet.

Ende Februar, nach Abschluss des Projektseminars, wurde das CoSy in CommSy umgetauft, da eine Firma ein ähnliches Community System unter dem Namen CoSy veröffentlicht hatte.

Im Sommersemester 2000 setzten die Entwickler einen CommSy-Projektraum als Unterstützung einer Veranstaltung im Fachbereich Erziehungswissenschaft und einen im Fachbereich Informatik auf. Im Hinblick auf den technischen Betrieb bedeutete dies, für jede der Veranstaltungen erneut einen Webserver (mittlerweile Apache) mit PHP zu konfigurieren, eine MySQL-Datenbank aufzusetzen, die CommSy-Tabellenstruktur einzupflegen und den PHP-Code zu kopieren. Beide Veranstaltung erhielten eine komplett unabhängige Installation, so dass im Laufe der Zeit simultan mehrere CommSy-Projekträume in unterschiedlichen Versionen - aufgrund der stetigen Weiterentwicklung - entstanden. Da viele Veranstalter keine Projektgruppe mit der Bereitstellung des CommSys beauftragten, übernahmen zu dieser Zeit die Entwickler die Bereitstellung für jede einzelne Veranstaltung. Zum Kreis der Entwickler zählten mittlerweile bis zu 15 Personen. Fünf von Ihnen leisteten Implementierungsarbeiten, die anderen arbeiteten u. a. an der Konzeption von CommSy und waren Test-User.

Durch vermehrte Anfragen, einen CommSy-Projektraum benutzen zu dürfen, stießen die Entwickler bei der Bereitstellungsleistung an ihre Grenzen. Sie mussten für jede Veranstaltung den kompletten Bereitstellungsaufwand wiederholen. Die damit verbundenen Aufgaben erledigten sie teilweise in ihrer Freizeit, parallel zu Weiterentwicklung, Studium, Diplomarbeit

bzw. zu ihrer Arbeit als wissenschaftliche Mitarbeiter. Zum Ende des Sommersemesters 2000 unterstützten sie mehr als zehn Lehrveranstaltungen der Universität Hamburg in verschiedenen Fachbereichen.

Von Juli bis Oktober 2000 fand die „Internationale Frauenuniversität (IFU)“ (Neusel 2000) als ein Projekt der Weltausstellung EXPO2000 statt. Sie bestand aus mehreren Projektbereichen, die über die gesamte Welt verteilt waren. Der Projektbereich „Information“ wurde im Fachbereich Informatik der Universität Hamburg geleitet und mit zwölf CommSy-Projekträumen unterstützt (Bleek 2004). Eine statische HTML-Seite diente als Einstiegsseite in die verschiedenen Projekträume.

Abbildung 8: Überblicksseite der zwölf CommSys für die IFU

Zur Unterstützung der IFU erweiterte das Entwicklerteam den PHP-Code des CommSys derart, dass die Daten mehrerer CommSy-Projekträume in nur einer MySQL-Datenbank gespeichert werden konnten. Dies erleichterte den technischen Betrieb dahingehend, dass alle zwölf CommSy-Projekträume mit der gleichen Version der Datenbankstruktur arbeiteten. So konnte die Datenbank der Projekträume gemeinsam gepflegt werden. Die Konfiguration eines Webservers plus PHP-Interpreter und die Kopie des PHP-Codes waren allerdings weiterhin erforderlich, denn es existierten immer noch fast unabhängige Installationen, die „nur“ auf der Datenbank zusammenliefen.

Zum Wintersemester 2000/2001 traten weitere Lehrveranstalter und der Studiengang Wirtschaftsinformatik, in Form einer Gruppe Studierender,

mit der Bitte an die Entwickler heran, CommSy benutzen zu dürfen. Es gab nun zwar die Möglichkeit, die Daten mehrerer CommSy-Projekträume in einer Datenbank zu halten, doch aufgrund der teilweise unterschiedlichen Einsatzkontexte entstanden nach und nach diverse CommSy-Datenbanken. Außerdem war das Einrichten eines CommSy-Projektraums weiterhin mit dem Kopieren des PHP-Codes und dem Konfigurieren des Webservers, u. a. für die Zugangsberechtigungen, verbunden. Mit der steigenden Anzahl an Benutzern häuften sich auch die Fragen zu Benutzungsproblemen. So übernahmen die Entwickler ab Sommer 2000, neben der Entwicklung und dem technischen Betrieb, die Aufgaben der Benutzerbetreuung.

Die Benutzerbetreuung, der technische Betrieb und die Weiterentwicklung von CommSy-Projekträumen konnten Mitte 2000 von den Entwicklern nicht mehr mit der von ihnen angestrebten Qualität geleistet werden. Sie entschieden sich, die Aufgaben des Betriebs und die der Benutzerbetreuung an einen professionellen Application Service Provider abzugeben, um sich verstärkt der Weiterentwicklung widmen zu können. Die Firma Uni.de AG schien ein solcher Application Service Provider zu sein.

3.3.2 CommSy@Uni.de

1999 wurde die Firma Uni.de AG, im Folgenden Uni.de genannt, finanziert durch Venture Capital, gründet. Das Konzept des Startup-Unternehmens sah vor, unter der Internetadresse `http://www.uni.de/` Hochschulen in Deutschland eine gemeinsame Kommunikationsplattform anzubieten. Anfangs konzentrierte sich Uni.de speziell auf Studierende und bot kostenlos E-Mail, SMS-Versand und Informationen über das Studieren in Deutschland. Ende 1999 entschloss sich Uni.de, auch wissenschaftliches Personal von Hochschulen mit entsprechenden Dienstleistungen anzusprechen. So entstand im Frühjahr 2000 ein erster Kontakt zwischen Uni.de und der CommSy-Entwicklung. Uni.de sah in CommSy die Möglichkeit, Wissenschaftler an Hochschulen als Nutzende zu gewinnen. Die CommSy-Entwicklung erhoffte sich, von den Aufgaben des Betriebs und der Benutzerbetreuung entlastet zu werden.

Am 17.05.2000 fand ein erstes Treffen zwischen dem Gründer von Uni.de und vier CommSy-Entwicklern statt. Der deutschlandweite Einsatz der CommSy-Projekträume wurde diskutiert. Es ging um notwendige technische Weiterentwicklungen (u. a. die Implementation eines CommSy-

Servers, so dass nur eine Installation für beliebig vieler CommSy-Projekträume notwendig wurde) und um fachliche Begleitmaßnahmen (u. a. Moderationskonzept, Benutzerbetreuung), die nach damaligen Erfahrungen den Erfolg von CommSy erst ermöglichten. Ergebnis des Treffens war, Uni.de das Nutzungsrecht von CommSy gebührenfrei zu überlassen. Im Gegenzug sollte Uni.de die notwendige technische Weiterentwicklung von CommSy und die Vorbereitung und anfängliche Durchführung der erforderlichen fachlichen Begleitmaßnahmen finanzieren. Nach einer Übergangszeit von einem halben Jahr sollte ein Mitarbeiter von Uni.de die fachlichen Begleitmaßnahmen übernehmen.

Die CommSy-Entwicklung stellte in den folgenden Tagen einen Katalog mit 24 technischen und acht fachlichen Arbeitspaketen zusammen, aus dem sich Uni.de nach eigenem Interesse Pakete auswählen konnte (vgl. Bleek 2004, S. 264f.). Aufgrund von Verhandlungen zur firmeninternen Umstrukturierung von Uni.de verzögerte sich diese Auswahl. Uni.de sollte in der Unternehmensgruppe FirstCampus (Student Holding SAS) aufgehen und dadurch den Status einer eigenständigen Firma verlieren. Das Ziel der Unternehmensgruppe war das europaweite Angebot von nationalen Plattformen für die jeweilige wissenschaftliche Gemeinschaft des europäischen Landes.

Am 10.07.2000 forderte Uni.de, CommSy auch europaweit einsetzen zu dürfen. Die Forderung war auf den Übergang in die Unternehmensgruppe FirstCampus zurückzuführen. Die CommSy-Entwicklung lehnte dies aus Gründen der nicht evaluierten Übertragbarkeit auf andere Länder und Wissenschaftskulturen ab. Der Streit um den europaweiten Einsatz verzögerte die Vertragsverhandlungen um mehrere Wochen und endete schließlich in einem Kompromiss, der Uni.de erlaubte, für jedes weitere europäische Land der FirstCampus Unternehmensgruppe zehn Testräume einzurichten. Sollten die jeweiligen Länder mehr CommSy-Projekträume anfordern, bestünde Verhandlungsbedarf. Genutzt wurden diese Testräume jedoch nie.

Der CommSy-interne Zeitplan sah vor, innerhalb der Semesterferien (Juli bis September 2000) alle vorbereitenden Maßnahmen und Weiterentwicklungen abzuschliessen. Im Oktober 2000 (Beginn des Wintersemesters 2000/2001) sollte CommSy offiziell online gehen. Am 17.07.2000 fand ein Workshop statt, auf dem die Entwickler die Arbeitspakete unter sich aufteilten. Uni.de hatte zu diesem Zeitpunkt noch keine Auswahl getrof-

fen. Für den 20.07.2000, Datum des zweiten Treffens zwischen CommSy-Entwicklung und Uni.de, war der Abschluss eines Kooperationsvertrags mit Festlegung der Arbeitspakete geplant.

Uni.de suchte überwiegend technische Arbeitspakete aus, deren Anzahl unter der Minimalgrenze der von der CommSy-Entwicklung festgelegten unabdingbaren Arbeitspakete lag. Am Ende des Verhandlungstages einigte man sich auf Arbeitspakete im Wert von 50 Personentagen (PT): 40 PT technische Weiterentwicklung, zehn PT fachliche Vorbereitung und Begleitmaßnahmen. Weiterhin wurde vereinbart, den Vertrag nach Einarbeitung der getroffenen Auswahl an Arbeitspaketen schnellstmöglich zu unterschreiben. Leider hatten die anwesenden Entwickler Abhängigkeiten in den technischen Arbeitspaketen übersehen. Sie erhöhten eigenmächtig die mit Uni.de vereinbarten 50 PT um 16 weitere PT auf insgesamt 66 PT. Dies lehnte Uni.de strikt ab. Ein für den 01.08.2000 vereinbartes Telefongespräch zwischen den Technikexperten beider Seiten, in dem die Auswahl der technischen Arbeitpakete erneut diskutiert werden sollte, fand nicht statt. Erst am 24.08.2000 kam es zu diesem Gespräch. Sie einigten sich auf 36 PT für technische Arbeitspakete und 10 PT für fachliche Arbeitspakete (vgl. Bleek 2004, S. 264f.).

Außerdem wurde vereinbart, dass Uni.de zusätzlich zum Kooperationsvertrag einen Beratungsvertrag zur Moderation eines Projektraums für alle Veranstalter von CommSy-Projekträumen bei Uni.de mit der CommSy-Entwicklung abschließen sollte. Dieser Beratungsvertrag sah ebenfalls vor, dass die CommSy-Verantwortlichen von Uni.de Mitglieder in diesem Projektraum werden sollten, so dass die Entwickler sie, im Sinne der Benutzerbetreuung, als Moderatoren anlernen könnten.

Ziel der beiden Verträge (Kooperations- und Beratungsvertrag) war, aus Sicht der Entwickler, Uni.de zu ermöglichen, CommSy-Projekträume erfolgreich zu betreiben. So forderte die CommSy-Entwicklung Uni.de auf, Aufgaben in der Benutzerbetreuung zu übernehmen. Dabei ließen sich die Entwickler von der Einsicht leiten, dass die Technik allein nicht zu einer sinnvollen Nutzung führt (vgl. Pape 2004) und eine schlechte Bereitstellung auf CommSy zurückfallen würde (Strauss u. a. 2003). Der antizipierte Sinn beider Verträge aus Sicht von Uni.de war, CommSy von den Entwicklern zu kaufen. CommSy wurde aus Haftungsgründen nicht verkauft, sondern Uni.de kostenfrei überlassen. Vertragsgegenstände waren die Dienstleistungen Entwicklung und Bereitstellung.

Uni.de schickte den von den Geschäftsführern unterschriebenen Kooperationsvertrag am 25.08.2000 nach Hamburg. Er wurde, begründet durch Urlaubszeit und Krankheit, erst am 14.09.2000 im Postfach des Hamburger Informatik Technologie-Center e.V. (HITeC)[4] gefunden. Inzwischen hatten viele Studierende Ferienjobs angenommen oder mit Studien- und Diplomarbeiten begonnen, so dass der ursprüngliche Online-Termin (01.10.2000) nicht eingehalten werden konnte. Daher wurde der Vertrag von HITeC nicht unterzeichnet und ein neuer Zeitplan, der den Online-Termin auf Mitte März verschob, ausgearbeitet. Der neue Zeitplan wurde mit dem noch abzuschließenden Beratungsvertrag am 22.09.2000 an Uni.de geschickt. Trotz erheblicher Einwände bestätigte Uni.de am 27.09.2000 den neuen Zeitplan. Am 02.11.2000 traf der von Uni.de unterschriebene Nachtrag zum Kooperationsvertrag mit geändertem Zeitplan in Hamburg ein. Der Beratungsvertrag erreichte die Entwickler erst am 26.02.2001.

Abbildung 9: CommSy-Einstiegsseite bei Uni.de (Version 1.0)

4 HITeC übernimmt den Technologietransfer des Fachbereichs Informatik der Universität Hamburg mit Partnern aus der Wirtschaft und war offizieller Vertragspartner von Uni.de.

Durch einen Wechsel des technischen Ansprechpartners bei Uni.de verzögerte sich die Installation von CommSy auf den von Uni.de in einem Rechenzentrum in München gemieteten Servern. Die Übergabe einiger Passwörter wurde vergessen. Erst zum 15.03.2001 konnten die Entwickler den CommSy-Code aufspielen, den Apache Webserver installieren, den PHP-Interpreter konfigurieren und die MySQL-Datenbank aufsetzen.

Am 01.04.2001 ging CommSy unter der URL `http://commsy.uni.de/` offiziell online. Zum gleichen Zeitpunkt begann, im Rahmen des Beratungsvertrags, das Coaching für den CommSy-Verantwortlichen von Uni.de, der zu diesem Zeitpunkt gerade wechselte. Die Entwickler berieten fortan den Uni.de-Mitarbeiter u. a. hinsichtlich der Konfiguration des CommSy-Projektraums für die Veranstalter und formulierten E-Mails zur Einladung in diesen Projektraum vor. Diese Hilfe wurde von Uni.de nur in den ersten Tagen angenommen und zügig umgesetzt.

Am 02.04.2001 blendete Uni.de über den CommSy-Projekträumen Werbebanner ein, um die für Nutzende kostenlose Bereitstellung zu refinanzieren. Aufgrund technischer Schwierigkeiten in der Bannerrotation wurde immer nur ein Werbebanner angezeigt.

Abbildung 10: Das erste Werbebanner unter http://commsy.uni.de/

Am 03.04.2001 wiesen die Entwickler auf diese Tatsache hin und äußerten Bedenken hinsichtlich der sexistischen Aufmachung. Diese Befürchtungen erwiesen sich als berechtigt. Noch am selben Tag trafen bei Entwickler und Uni.de E-Mails ein, deren Verfasser mit Nichtnutzung der Projekträume drohten. Die Beschwerden setzten sich bis zur Entfernung des sexistischen Werbebanners und der Behebung der Rotationsschwierigkeiten am 05.04.2001 fort. Die anschließend angezeigten Werbebanner erzeugten keinen Protest, dennoch boten verschiedene Personen finanzielle Mittel für werbefreie CommSy-Projekträume an und formulierten diesbezüglich Anfragen an Uni.de. Diese Möglichkeit der Vermarktung kam nicht zustande, denn Uni.de war nicht bereit, in die technische Weiterentwicklung zur konfigurierbaren Anzeige von Werbebannern zu investieren.

Zur besseren Vermarktung erarbeitete Uni.de stattdessen ein neues Layout für die Einstiegsseite zu den Projekträumen. Der neue Designvorschlag erreichte die Entwickler am 25.04.2001. Er wurde nie umgesetzt, obwohl Uni.de den Designvorschlag selbst ins CommSy integrieren wollte und die Entwickler kostenlose Hilfe anboten.

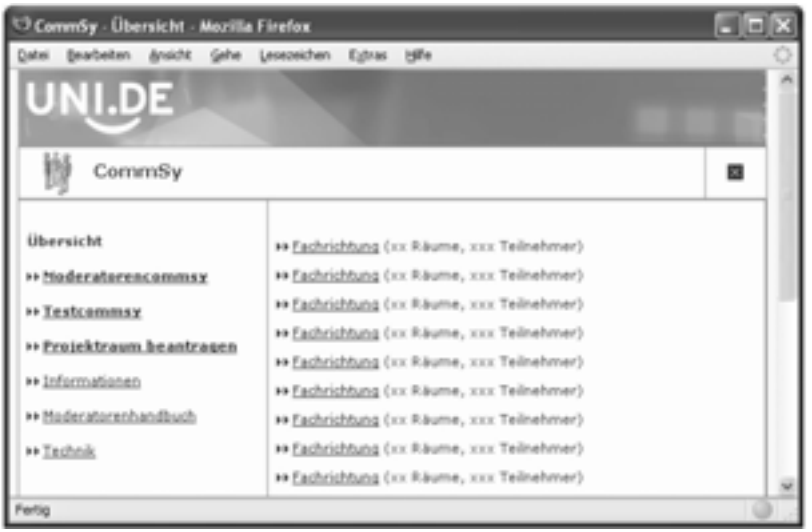

Abbildung 11: Neues Layout für die CommSy-Einstiegsseite bei Uni.de

Bei den Entwicklern häuften sich in den ersten Wochen der Bereitstellung Benutzeranfragen zu den bei Uni.de angebotenen CommSy-Projekträumen. Uni.de hatte keinen offiziellen Ansprechpartner für CommSy auf den eigenen Webseiten benannt. In einer Telefonkonferenz am 17.04.2001 sprachen die Entwickler diesen Missstand an. Schließlich waren sie die Kooperation eingegangen, um vom technischen Betrieb und der Benutzerbetreuung entbunden zu werden. Auch nach der Telefonkonferenz mussten die Entwickler die Aufgaben der Benutzerbetreuung und des Betriebs des CommSy-Servers übernehmen, denn die schlechte Betreuung von CommSy durch Uni.de drohte den Ruf von CommSy zu beschädigen. Die Kooperation zwischen den Entwicklern und Uni.de kühlte nach dieser Telefonkonferenz ab. Einerseits wurden die Entwickler nicht von den Aufgaben des Betriebs und der Benutzerbetreuung entlastet, sondern hatten durch die Kooperation mit Uni.de zusätzlichen Aufwand. Andererseits schaffte Uni.de es nicht, die

CommSy-Projekträume durch Werbebanner refinanzieren; daher betrachtete es die CommSy-Projekträumen als Verlustgeschäft.

Am 26.06.2001 erreichte die Entwickler die inoffizielle Information, dass die Personen, die ursprünglich den Kontakt zu der CommSy-Entwicklung hergestellt hatten, Uni.de verlassen hatten. Damit hatte das Uni.de-Team die beiden größten Befürworter von CommSy verloren. Im Juli 2001 erhielten die Entwickler zwei E-Mails, die diesen Sachverhalt offiziell bestätigten. Konsequenz war, dass bei Uni.de für die Betreuung von CommSy keine Ressourcen mehr zur Verfügung standen. Die CommSy-Projekträume sollten aus Sicht von Uni.de als kostenloser Dienst unbetreut weiterlaufen. Die Entwickler sahen die Kooperation mit Uni.de an diesem Punkt als gescheitert an. Aufgrund der bestehenden Verträge hielten sie ihr Engagement bis zum Ende des Beratungsvertrages aufrecht und stellten ihre Bemühungen erst am 08.11.2001 offziell ein. Bei späteren Anfragen verwiesen die Entwickler auf die beendete Kooperation und forderten die Nutzenden auf, sich direkt an Uni.de zu wenden.

Anfang Dezember 2001 und Ende März 2002 häuften sich bei den Entwicklern Beschwerden hinsichtlich Verbindungsprobleme zum CommSy-Server bei FirstCampus, mittlerweile Nachfolger von Uni.de, zu dem die Entwickler keinen Zugang mehr hatten. Am 05.04.2002 kündigten die Entwickler offiziell den Kooperationsvertrag zum Ende des Jahres, so dass der CommSy-Server zu diesem Zeitpunkt hätte abgeschaltet werden müssen. Trotz mehrmaliger Hinweise von Seiten der Entwickler wurde die CommSy-Bereitstellung bei FirstCampus nicht eingestellt.

3.3.3 CommSy@WissPro

Im März 2001 begann das Forschungs- und Entwicklungsprojekt WissPro (Abkürzung für „Wissensprojekt: Informatiksysteme im Kontext – Vernetzte Lerngemeinschaften in gestaltungs- und IT-orientierten Studiengängen“) (vgl. Pape u. a. 2004). Eine Projektaufgabe von WissPro war die Weiterentwicklung und Bereitstellung von CommSy. Durch die sich schon im Frühjahr 2001 anbahnenden Schwierigkeiten in der Kooperation von Uni.de mit den Entwicklern von CommSy wurde die auch im Projekt WissPro angestrebte Kooperation mit Uni.de nicht eingegangen. Der Betrieb der verschiedenen weiterentwickelten CommSy-Versionen wurde vielmehr auf dem projekteigenen Server sichergestellt. Eine angemessene Benutzerbe-

treuung für die Nutzenden der jeweiligen Versionen wurde von Mitarbeitern des Projekts durchgeführt. Hierbei übernahmen zunächst alle Mitarbeiter kooperativ alle Aufgaben.

Während des Projekts WissPro nahmen u. a. drei Aspekte auf die Entwicklung und Bereitstellung von CommSy wesentlichen Einfluss:

1. Die Projekträume wurden durch ein Archiv und ein Portal erweitert, um neben der einzelnen Lehrveranstaltung auch das Studium bzw. die Bildungsinstitution als Ganzes zu unterstützen. Archiv und Portal wurden im Wintersemester 2002/2003 zum ersten Mal präsentiert. Zum Wintersemester 2003/2004 wurden sie zum CommSy-Gemeinschaftsraum vereint.

2. Die Zahl der Nutzenden stieg während der Projektlaufzeit stetig an.

3. Die CommSy-Entwicklung wurde von einem Closed-Source-Prozess in einen Open-Source-Prozess überführt.

Mit dem Beginn der Entwicklung von Archiv und Portal verteilten sich die Projektmitarbeiter auf drei Teams für die Produkte Projektraum, Archiv und Portal. In den einzelnen Teams wurden weiterhin von allen Mitarbeitern alle Aufgaben der Entwicklung und Bereitstellung in Personalunion pro Produkt übernommen. Die parallele Entwicklung und Bereitstellung der drei Produkte führte zu einer Konkurrenzsituation innerhalb des Forschungs- und Entwicklungsprojektes; z. B. achteten die Projektmitarbeiter genau darauf, wie studentische Hilfskräfte eingesetzt wurden.
Um die Jahreswende 2001/2002 führten verschiedene projektinterne Begebenheiten dazu, die Entwicklung der drei Produkte zusammenzulegen; u. a. verliebten sich die Verantwortliche für das Archiv und der Verantwortliche für die Projekträume (vgl. Pape und Rolf 2004). Sie heirateten im April 2004. Im Zuge der Zusammenlegung der drei Produkte verschmolzen auch die drei Teams. Die Verschmelzung hatte zur Folge, dass die vormals in Personalunion übernommenen Aufgaben neu verteilt werden mussten. Von nun an übernahmen Teammitglieder schwerpunktmäßig bestimmte Aufgaben in der Entwicklung und Bereitstellung. Das Produkt wurde weiterhin unter dem Namen CommSy präsentiert, da dieser sich bereits etabliert hatte. Die einzelnen Bestandteile hießen von da an CommSy-Projekträume, CommSy-Archiv und CommSy-Portal. Später wurden das Archiv und das Portal zum CommSy-Gemeinschaftsraum vereint.

Abbildung 12: Einstiegsseite des Archivs ‹mind› (lauffähiger Prototyp)

Die stetig steigende Anzahl von Nutzenden trug ebenfalls zur expliziten Übernahme von bestimmten Aufgaben in der Benutzerbetreuung bei. Mit der steigenden Anzahl an Nutzenden gingen vermehrt Supportanfragen ein, die in dieser Fülle nicht mehr „nebenbei“ beantwortet werden konnten. Es bildete sich ein Benutzerbetreuungsteam, das sich diesen Anfragen bewusst annahm und darüber hinaus die existierenden CommSys moderierte und redaktionell betreute. Die Zahl der Nutzenden erhöhte sich von ca. 1500 im Jahre 2001 über ca. 2500 im Jahre 2002 auf ca. 5000 im Jahre 2003[5]. Das Projekt WissPro profitierte von den bereits im Fachbereich Informatik der Universität Hamburg vorhandenen Nutzenden sowie von Nutzenden bei Uni.de, die mit der dortigen Bereitstellung nicht zufrieden waren. Ende

5 Die Nutzungszahlen lassen sich in den jeweiligen Datenbanken der CommSy-Installationen ablesen und sind gerundet, da es an dieser Stelle um die Darstellung der Steigerungsrate geht. Genaue Zahlen befinden sich in der Auswertung der Fallstudien in Teil II dieser Arbeit.

Abbildung 13: Einstiegsseite des Campus (Papierprototyp)

2001 wurde eine E-Mail an alle Veranstalter von CommSy-Projekträumen bei Uni.de verschickt, um auf den neuen CommSy-Service im Rahmen des Projekts WissPro aufmerksam zu machen. Darüber hinaus warb das Projekt auf der CeBIT 2003 und auf der LearnTec 2002 und 2003 für CommSy und den Bereitstellungsservice des Projekts.

Neben dem Entwicklungsteam und dem Benutzerbetreuungsteam entstand schließlich ein drittes Team, das sich der Pflege des Servers annahm. Die Aufgaben dieses Teams bestanden vor allem darin, die Basissoftware des Servers (Linux, Apache, PHP, MySQL) auf dem neuesten Stand zu halten und entsprechend zu konfigurieren. Darüber hinaus wurden die kontinuierlich entstandenen CommSy-Versionen in den laufenden Betrieb eingespielt.

Die Bildung dreier Teams für verschiedene Aufgabenfelder und die Tatsache, dass bis auf eine Ausnahme jeder nur Mitglied in einem Team war, ermöglichte den Mitarbeitern, sich auf ihre jeweiligen Kompetenzen zu konzentrieren. Sie verloren aber den direkten Kontakt zu den anderen

Aufgabenfeldern. So erhielten die Benutzerbetreuer wenig Einblick in die neusten Entwicklungen und hatten Schwierigkeiten, bei Supportanfragen zu erkennen, ob es sich um Benutzungsprobleme oder Softwarefehler handelte. Umgekehrt verloren die Entwickler den engen Kontakt zu den Nutzenden. Sie hatten teilweise keine Kenntnisse über Softwarefehler, die bei der Benutzung aufgetraten, und wussten nicht, wie Neues von den Nutzenden aufgenommen wurde bzw. welche Anforderungen die Nutzenden an die Weiterentwicklung stellten.

Dieser durch die Teambildung veränderten Kommunikationssituation wurde im Projekt bewusst begegnet, indem auf wöchentlich stattfindenden Treffen über Themen gesprochen wurde, die sich übergreifend auf Entwicklung, Betrieb und Benutzerbetreuung auswirkten. Darüber hinaus richteten das Entwicklerteam, das Betriebsteam und das Benutzerbetreuungsteam weitere Kommunikationskanäle (spezifische E-Mailadressen, eigener CommSy-Projektraum) ein. Die Kommunikationskanäle ermöglichten einen flexiblen Austausch über auftretende Fehler, neueste Entwicklungen und das Vorhandensein neuer Versionen. Größere Design- und Weiterentwicklungsentscheidungen wurden ebenfalls mit allen Teams gemeinsam getroffen.

Zu den drei bereits genannten Teams formierte sich im Herbst 2002 ein viertes, welches sich der Evaluation der CommSy-Nutzung annahm. Dieses Evaluationsteam führte Interviews durch und konzipierte Offline- sowie Online-Fragebögen. So wurde die CommSy-Nutzung seit dem Wintersemester 2002/2003 kontinuierlich evaluiert, und die Ergebnisse über die etablierten Kommunikationskanäle wurden an die Entwicklung, den Betrieb und die Betreuung weitergeleitet.

Im April 2003 wurde CommSy, mittlerweile bestehend aus Projekträumen und einem Gemeinschaftsraum (ehemals Archiv und Portal), unter die GNU General Public License (GPL) gestellt. Damit war der erste Schritt zu einer Open Source Entwicklung getan. Dieser Schritt zog mit sich, dass die verschiedenen Teams, auch wenn sie noch bis Ende 2003 gemeinsam im Projekt WissPro verankert waren, sich verstärkt auf die explizit ausgehandelten Kommunikationskanäle einließen. Spontane „Flurgespräche“ fanden immer seltener statt. Die verschiedenen Teams verbanden damit zwei Motivationen: Zum einen wurde im Hinblick auf die Open-Source-Entwicklung bereits frühzeitig eine Open-Source-ähnliche Organisation des Entwick-

lungsprozesses initiiert, zum anderen war nur so die Öffnung des Entwicklungsprozesses und das Einbeziehen von externen Interessierten möglich.

3.3.4 Blick in die Zukunft

Während der Projektlaufzeit von WissPro entstanden Bemühungen, die Bereitstellung und Entwicklung nachhaltig in der Universität Hamburg zu verankern. Für die Bereitstellung von CommSy für Hamburger Hochschulen wurde bereits 2001 ein Projektantrag an das E-Learning-Consortium Hamburg (ELCH) gestellt, der in der ersten Förderrunde abgelehnt, in der zweiten angenommen wurde. So konnte durch das ELCH-Projekt „CommSy goes Hamburg“ die Bereitstellung zunächst für das Jahr 2004 finanziell gesichert werden. Für die nachhaltige Verankerung der Entwicklung von CommSy in der Universität Hamburg wurden Gespräche mit entsprechenden Entscheidungsträgern aufgenommen.

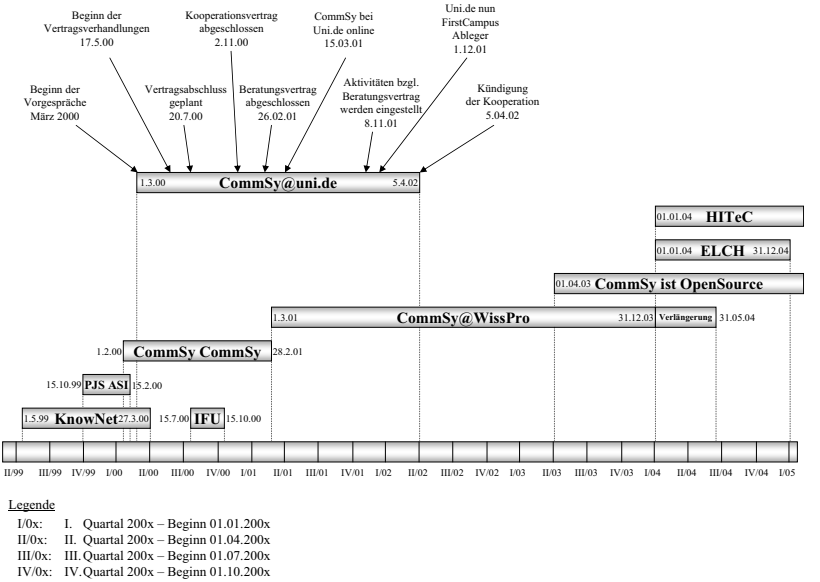

Abbildung 14: Werdegang von CommSy 1999 - 2004

Anfang 2004 formierten sich alle CommSy-Interessierten in einem neu initialisierten CommSy-Projekt beim Hamburger Informatik-Technologie Center e.V. (HITeC), um einen Rahmen zu schaffen, auch in Zukunft angestellt oder ehrenamtlich, vor Ort oder entfernt an der Weiterentwicklung und Bereitstellung von CommSy partizipieren zu können. Darüber hinaus wurde mit dem CommSy-Projekt bei HITeC der Öffnung im Sinne eines Open Source Prozesses Rechnung getragen und diese nicht nur auf die Weiterentwicklung, sondern auf das gesamte CommSy-Projekt übertragen.

Ebenfalls im Jahre 2004 startete das Forschungsprojekt VIRKON (Arbeiten in VIRtuellen Konstrukten, Organisationen und Netzen) mit einer Laufzeit von knapp zweieinhalb Jahren. Für das CommSy-Projekt ist dieses Forschungsprojekt interessant, da es die Unterstützung von Freelancer-Netzwerken mit Neuen Medien untersucht und u. a. CommSy zur medialen Unterstützung einsetzt.

3.4 Zwischenergebnis

Die dargestellten Fallstudien bringen zum Ausdruck, dass im Fall CommSy@Uni.de die Bereitstellung weniger erfolgreich und im Fall CommSy@WissPro erfolgreicher war. Warum? Was unterscheidet diese Fälle voneinander? Um die Unterschiede fundiert herauszuarbeiten, müssen die Fallstudien tiefgründiger betrachtet werden. Zu diesem Zweck wird zunächst der Begriff der Perspektive aus der Informatik herangezogen, und anschließend werden vier verschiedene Perspektiven auf die Softwarebereitstellung motiviert.

Unter einer Perspektive wird in der Informatik die Menge aller Annahmen über relevante Aspekte eines interessierenden Gegenstandbereichs aus einem gemeinsamen Blickwinkel verstanden. Perspektiven sind nicht an Personen gebunden, sondern eine Person nimmt zeitlich verschoben unterschiedliche Perspektiven ein. Zwischen zwei oder mehreren Personen können sich gemeinsame Perspektiven bilden (vgl. Lehner u. a. 1995; Floyd 1989). Perspektivität bedeutet, dass bestimmte Phänomene ausgeblendet oder nicht einbezogen werden. „Perspektivität ist prinzipiell mit Blindheit verbunden. Ich sehe nicht, was ich aus meiner Perspektive nicht sehen kann. Die Blindheit kann niemals ausgeschaltet werden. Voraussetzung für das Entstehen von tieferen Einsichten ist die Selbstreferenz und die Interaktion (Kreuzung) von Perspektiven. [...] Perspektivische Blindheit kann

durch Selbstreferenz überwunden werden (‚Wenn ich sehe, daß ich blind bin, kann ich sehen')" (Floyd 1989, S. 8).

Aus meinen Erfahrungen bei der Bereitstellung von Software, insbesondere in den beiden Fallstudien, schlage ich, in Ermangelung an Alternativen aus der Literatur, die Aufgaben-, Organisations- und Technikperspektive sowie das Vorgehen zur Betrachtung der Softwarebereitstellung vor. Für mich stellen sie die Hauptaspekte bei der Bereitstellung von Software dar. So konzentriere ich mich mit einer Perspektive auf einen Gegenstandsbereich der Softwarebereitstellung, immer mit dem Wissen, dass nur das Einnehmen und Kreuzen aller Perspektiven zu einem Gesamtverständnis führt.

Aufgabenperspektive

Bei der Bereitstellung von Software fallen in den Fallstudien augenscheinlich verschiedene Aufgaben an, die über den technischen Betrieb der Software hinausgehen. Welche Aufgabenbereiche sind neben dem technischen Betrieb relevant? Welche Aufgaben gehören zu diesen Aufgabenbereichen und welche Rollen nehmen welche Aufgaben wahr?

Mit einer entsprechenden Differenzierung, Komplettierung und Systematisierung der relevanten Aufgaben bei der Softwarebereitstellung können die Fallstudien fundierter betrachtet werden. So sind detaillierte Antworten auf folgende Fragen möglich:

- Welche Aufgaben wurden in welcher Fallstudie wahrgenommen und welche nicht?
- Welche Aufgaben hätten wahrgenommen werden sollen und welche Konsequenzen ergaben sich aus dem Nichtwahrnehmen einzelner Aufgaben?

Die Aufgabenperspektive eröffnet eine über die technischen Aufgaben hinausgehende Sicht auf die Softwarebereitstellung, indem sie weitere, für die Bereitstellung notwendige Aufgabenbereiche in den Blick nimmt.

Organisationsperspektive

Übergreifend muss die Bereitstellung von Software in geeigneter Art und Weise organisiert werden. Dabei gewährleistet die gewählte Organisations-

form letztlich die Bereitstellung der Software. Um die Bereitstellung gewährleisten zu können, müssen die notwendigen Aufgaben übernommen werden. Organisation bedeutet die Zuordnung von Aufgaben zu entsprechenden Akteuren. Hier stellt sich die Frage, wer diese Akteure sind bzw. sein könnten und in welcher Rolle sie zur Bereitstellung beitragen. Welche Rollen lassen sich insgesamt unterscheiden und wie verteilen sich die Aufgaben auf diese Rollen?

Neben den beteiligten Rollen und Akteuren ist es wichtig, die Kooperation unter den Akteuren zu betrachten. Wie können die verschiedenen Akteure in ihrem Handeln synchronisiert werden? Wie und mittels welcher Wege kommunizieren und kooperieren sie? Was steht einer Kooperation der beteiligten Akteure entgegen? Wie werden die Kooperationen verbindlich geregelt und wie finden sich die Kooperationspartner? Wer investiert was und wie viel in die Kooperation und was kostet dies die Beteiligten?

Durch die Betrachtung der Handelnden und deren Beziehungen zueinander können die beiden Fallstudien eingehender hinterfragt und detailliert bewertet werden:

- Wer hat welche Aufgaben in welcher Form in den Fallstudien übernommen?
- Wie gestaltete sich die Kooperation unter den beteiligten Akteuren?
- Gab es verbindliche Regelungen, die die Kooperation mit Pflichten und Rechten definierten – und wurden sie befolgt?
- Was kostete die Bereitstellung in den Fallstudien die beteiligten Akteuren und welchen Gewinn zog jeder Einzelne aus der Kooperation?

Die Betrachtung der Softwarebereitstellung gewinnt in der Organisationsperspektive an Kontur, da in dieser Perspektive den Akteuren die Aufgaben zugewiesen werden und zudem auf die Beziehung der beteiligten Akteure und deren Rollen eingegangen wird.

Technikperspektive

Bei der Bereitstellung von Software wird aus technischer Sicht nicht nur die bereitzustellende Software benötigt, sondern auch Hardware und zusätzliche Software wie Betriebssysteme. Da auf Seiten des Bereitstellers und des

Nutzers Unterschiede hinsichtlich der Technik bestehen können, ist weiterhin hinsichtlich der Hard- und Software zwischen dem Bereitsteller und den Nutzenden zu unterscheiden. Hinzu kommt der Übertragungsweg zwischen Nutzer und Bereitsteller, der ebenfalls betrachtet werden muss.

Diese differenzierte Sichtweise lässt genauere Aussagen hinsichtlich der technischen Unterschiede in den Fallstudien zu:

- Welche Unterschiede in der vorhandenen technischen Infrastruktur in den beiden Fallstudien gab es?
- Waren die entsprechenden technischen Infrastrukturen für eine Softwarebereitstellung von CommSy geeignet?

Die Technikperspektive betrachtet die vorhandene technische Infrastruktur über die bereitzustellende Software hinaus und erweitert damit den Blick auf die Technik.

Vorgehen

Die Bereitstellung einer Software erstreckt sich in der Regel über einen längeren Zeitraum. In diesem Zeitraum ist die Bereitstellung verschiedenen sich wandelnden Einflüssen ausgesetzt. Welche Einflüsse sind dies? Wo liegen deren Ursprünge und wie wirken sie sich auf die Softwarebereitstellung aus? Wie können die beteiligten Akteure mit diesen Veränderungen umgehen? Wie muss eine Softwarebereitstellung gestaltet sein, um die vorherrschende Dynamik aufzufangen? Wie können die beteiligten Akteure die Softwarebereitstellung gestalten? Wann tun sie es bzw. wann sollten sie es tun?

Durch die Betrachtung des Vorgangs über einen gewissen Zeitraum werden Veränderungen sichtbar. Diese Veränderungen und ihre Ursachen müssen hinsichtlich der beiden Fallstudien hinterfragt werden:

- Welche Veränderungen basierend auf welchen Ursachen beeinflussten die Softwarebereitstellung von CommSy?
- Wann und wie haben die beteiligten Akteure gestaltend auf die Softwarebereitstellung eingewirkt und welche Konsequenzen hatte dies?
- An welchem Punkt war die Bereitstellung in dem einen Fall wirklich gescheitert und wie hätte dies verhindert werden können?

Durch die zeitliche Betrachtung der Softwarebereitstellung wird eine Prozesssicht eingenommen und die Softwarebereitstellung entlang einer fortschreitenden Zeitlinie gesehen. Diese Prozesssicht ermöglicht, die in den bereits vorgestellten Perspektiven erkannten Aspekte über die Zeit in Beziehung zu setzen und diese Beziehung zu gestalten. So stellt die Prozesssicht nicht nur eine Perspektive dar, sondern insbesondere ein gestaltendes Moment, in dem die anderen Perspektiven zusammengeführt werden. So kann durch die zeitliche Perspektive aus einem Vorgang ein Vorgehen und aus der Vorgangsperspektive eine Vorgehensweise zur Softwarebereitstellung werden.

Im Folgenden wird in dieser Arbeit sprachlich etwas ungenau von den vier Perspektiven Aufgaben, Organisation, Technik und Vorgehen gesprochen, obwohl, wie gerade erläutert, eigentlich der Vorgang die Perspektive wäre und nicht das Vorgehen. Da der hier zu erarbeitende Ansatz eASP zur Softwarebereitstellung nicht nur erklären, sondern auch bei der Gestaltung helfen soll, und die Gestaltung sich speziell im Vorgehen ausdrückt, wird aus Gründen der einfacheren Benennbarkeit der vier Aspekte das Vorgehen als Perspektive bezeichnet.

Als Zwischenergebnis dieses Kapitels bleibt festzuhalten, dass bei der Bereitstellung von Software Fragen aufgeworfen werden

- zur Technik (Welche Anwendung eignet sich für eine zentrale Bereitstellung?),
- zu den Aufgaben (Welche Aufgaben fallen bei der Bereitstellung an?),
- zur Organisation (Wer übernimmt welche Aufgaben?) und
- zum Vorgang (Wie verändert sich die Bereitstellung im Laufe der Zeit?).

Das bedeutet, ein Ansatz zur Softwarebereitstellung muss sich den hier gestellten Fragen in allen vier Perspektiven annehmen. Im nächsten Kapitel wird das Geschäftsmodell Application Service Providing (ASP) vorgestellt und anschließend untersucht, wie es den hier vorgestellten vier Leitperspektiven auf die Bereitstellung einer Software gerecht werden kann.

- 4 -
Geschäftsmodell Application Service Providing

In diesem Kapitel wird das Geschäftsmodell Application Service Providing (ASP) vorgestellt und anschließend hinsichtlich der im vorigen Kapitel vorgestellten Perspektiven bewertet. Die Bewertung gibt Auskunft über die Schwächen im ASP und gibt somit den Anstoß für die im folgenden Kapitel herangezogenen Grundlagen aus der Informatik. Im zweiten Teil dieser Arbeit kann dann, auf Grundlage dieses Kapitels, ASP fundiert und zum Ansatz eASP erweitert werden.

Die beiden Begriffe Application Service Provider und Application Service Providing tauchten signifikant zum ersten Mal 1999 in der Literatur auf (vgl. CherryTree 1999; Gillian u. a. 1999; Sound 1999). Spätere Autoren beziehen sich insbesondere auf Gillian u. a. (1999). Die „Wiege“ von ASP ist bei Unternehmensberatungen (vgl. Heere 2001; Citrix 2000; CherryTree 1999; Farleit 2000; Gillian u. a. 1999; Mercer 2000; Sound 1999) zu sehen. Thematisch ist ASP aus dem Outsourcing hervorgegangen und stellt in den Augen vieler Autoren eine spezielle Form des IT-Outsourcings dar (vgl. CherryTree 1999; Farleit 2000; Günther u. a. 2001; Stahlknecht 2000). Die Interpretation von ASP als Geschäftsmodell (vgl. Burris 2002; Grohmann 2002; Jaruzelski u. a. 2000; Mercer 2000; Picot u. a. 2000) führt zu einer Darstellung der Vorteile (vgl. Günther u. a. 2001; Sound 1999; Tao 2000) und Nachteile (vgl. Picot und Jahn 2000; Stahlknecht 2000) von ASP aus Kundensicht (vgl. Jayatilaka u. a. 2002) und aus Bereitstellersicht (vgl. Tao 2000). Dabei stehen meist technische Aspekte im Vordergrund (vgl. Heere 2001; Citrix 2000; Tao 2000) oder die Auseinandersetzung mit den beteiligten Akteuren (vgl. Eisenmann und Pothen 2001; Farleit 2000; Gillian u. a. 1999; Jaruzelski u. a. 2000; Seltsikas und Currie 2002; Tao 2000). Darüber hinaus beschäftigt sich die ASP-Literatur mit Service Level Agreements (SLAs) (vgl. Falkowski und Voß 2003; ITAA 2000; Trieneken u. a. 2004), ASP-Strategien in bestimmten Kontexten, z. B. der Hochschule (vgl. Gerhard und Mayr 2002), und präsentiert Evaluationsergebnisse über Angebot und Nachfrage im ASP-Markt (vgl. Günther u. a. 2001). Außer-

dem versuchen unternehmensübergreifende Konsortien (vgl. ASPIC 2002; ASPKD 2002) die Diskussionen im Themenfeld ASP zu bündeln und aufzubereiten.

Die angegebenen Quellen betrachten ASP unterschiedlich detailliert, mit heterogenen Schwerpunkten und oftmals nur den Application Service Provider im Gegensatz zum Application Service Providing. Dennoch kann ASP grundlegend – die verschiedenen Definitionen vereinheitlichend – wie folgt definiert werden:

Application Service Providing (ASP) bezeichnet die Bereitstellung (Providing) einer Anwendung (Application) als Dienstleistung (Service).

Da ASP, neben dem Verständnis als eine spezielle Form des Outsourcings, hauptsächlich als Geschäftsmodell angesehen wird, ist die Beschreibung von verschiedenen Aspekten des ASP in diesem Kapitel hinsichtlich der Hauptkomponenten eines Geschäftsmodells, Value Proposition, Architektur der Wertschöpfungskette und dem Ertragsmodell (Stähler 2001, S. 41f.), strukturiert:

- *Value Proposition*: Ein Geschäftsmodell enthält eine Beschreibung, welchen Nutzen Kunden oder andere Partner des Unternehmens aus der Verbindung mit diesem ziehen können. Die Beschreibung nennt sich Value Proposition.
- *Architektur der Wertschöpfungskette*: Zu einem Geschäftsmodell gehört neben der Value Propostion eine Beschreibung, wie der Nutzen generiert wird. Dies beinhaltet die Darstellung der verschiedenen Stufen der Wertschöpfung.
- *Ertragsmodell*: Das Ertragsmodell des Geschäftsmodells hält fest, welche Einnahmen das Unternehmen aus welchen Quellen generiert. Die zukünftigen Einnahmen entscheiden maßgeblich über den Wert des Geschäftsmodells.

ASP-spezifische rechtliche und technische Aspekte, die sich in den drei Hauptkomponenten nicht finden lassen, sind als zusätzliche Abschnitte diesem Kapitel beigefügt.

4.1 Value Proposition

Ehe auf den Nutzen für Kunden und Partner des Unternehmens, das die Bereitstellung von Software anbietet, eingegangen werden kann, ist zunächst eine Identifikation der beteiligten Akteure notwendig. Zu den Akteuren bietet die ASP-Literatur zwei Modelle an: das ASP-Player Modell und das ASP-Schichtenmodell. Hinsichtlich des Nutzens werden Vor- und Nachteile meist für Kunden in der ASP-Literatur diskutiert.

4.1.1 Beteiligte Akteure

Gillian u. a. (1999) und Günther u. a. (2001) setzten sich mit beteiligten Akteuren beim ASP in einem ASP-Player Modell und Farleit (2000) sowie Tao (2000) in einem ASP-Schichtenmodell auseinander.

ASP-Player Modell
Bei Gillian u. a. (1999, S. 8) und Günther u. a. (2001) findet sich ein ASP-Player Modell, welches Akteure (Pure ASPs, Network Provider, Application Vendors, Service Firms, Hardware Vendors, other Software Vendors und Distributors) als Application Service Provider mit Hilfe von benötigten Kernkompetenzen (Gillian u. a. 1999, S. 7) identifiziert.

Diese Kernkompetenzen unterteilen sich in drei Bereiche:

- Im *Service* sind Kompetenzen in der technischen Infrastruktur, im Vertrieb, im Projektmanagment und in der Kundenbetreuung erforderlich.
- Kompetenz im *Networking* bedeutet insbesondere mit Datenbeständen und großen Netzwerken umgehen zu können. Darüber hinaus spielen die Überwachung und Sicherheit der Netzwerke und Technik eine große Rolle.
- Für die *Application* müssen Kompetenzen hinsichtlich der Integration der Anwendung in organisatorische Prozesse sowie der Überwachung und Abrechnung von Lizenzen vorhanden sein. Darüber hinaus stellt der Benutzersupport eine wichtige Aufgabe dar.

Je nach konkretem Szenario werden diese Kompetenzen und Leistungen mehr oder weniger stark benötigt. Unternehmen, die Teile der aufgeführten Kompetenzen vorweisen, haben die Möglichkeit, nach dem ASP-Player

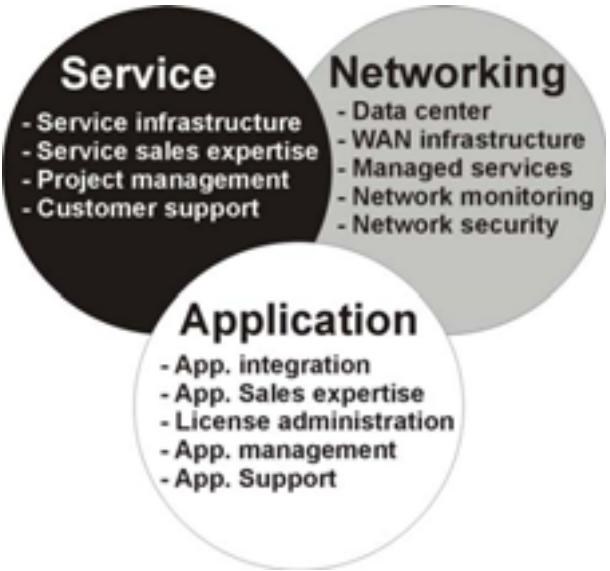

Abbildung 15: Kompetenzen eines Application Service Providers

Modell (Gillian u. a. 1999, S. 8) auf dem ASP-Markt eine von zwei verschiedenen Rollen einzunehmen:

- Der *Application Service Provider* (in der Abbildung 16 innerhalb des Dreiecks) stellt den Kunden eine Anwendung als Dienstleistung zur Verfügung.
- *Partner* (in der Abbildung 16 außerhalb des Dreiecks) helfen dem Application Service Provider bei der Bereitstellung von Software.

Folgende Unternehmen finden sich in der einen oder anderen Rolle wieder (vgl. Gillian u. a. 1999; Günther u. a. 2001):

- *Pure Plays* sind Application Service Provider, die extra für die Bereitstellung von Anwendungen in einem ASP-Modell entstanden bzw. gegründet worden sind.
- *Service Firms* bieten Dienstleistungen (Beratung, Support, usw.) zu bestimmten Anwendungen an. Sie können potentielle Partner (im ei-

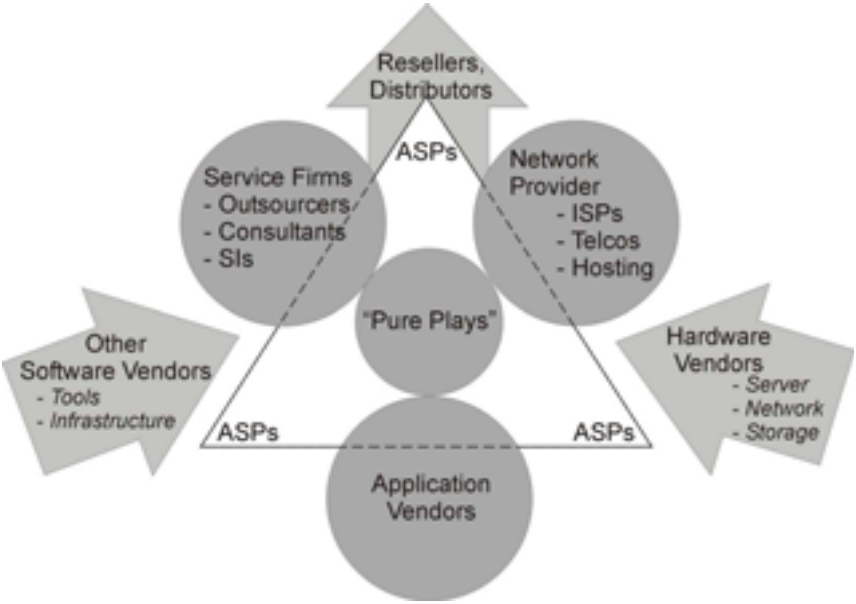

Abbildung 16: ASP-Player Modell

genen Kreis, aber außerhalb des Dreiecks) oder selber Application Service Provider (im eigenen Kreis und innerhalb des Dreiecks) sein.

- *Network Provider* oder auch Hoster stellen Rechenkapazität und den Zugang zum Internet bereit. Sie können potentielle Partner oder selber Application Service Provider sein.

- *Application Vendors* (Softwarehersteller) stellen Anwendungen bzw. speziell ASP-Anwendungen her. Sie können potentielle Partner oder selber Application Service Provider sein.

- *Other Software Vendors* (andere Softwarehersteller), die zusätzliche Software (Abrechnungssysteme, Virenscanner, usw.) zur Bereitstellung von Anwendungen in einem ASP-Modell entwickeln, stellen meist nicht selber Software bereit. Es würde sie von ihren Kernkompetenzen zu weit wegführen. Sie können als Partner von Application Service Provider auftreten.

- *Hardware Vendors* (Hardwarehersteller) liefern benötigte Hardware und sind wie die anderen Softwarehersteller als potentielle Partner einzustufen.

- *Distributors* und *Resellers* erwerben ASP-Dienstleistungen, die sie an Kunden weiterverkaufen. Sie treten als Vermittler oder als eigenständige Partner dem Kunden gegenüber auf und gelten ebenfalls als potentielle Partner.

ASP-Schichtenmodell
Bei Farleit (2000) und Tao (2000) findet sich ein ASP-Schichtenmodell, um die beteiligten Akteure zu identifizieren. Die verschiedenen Schichten werden von Provider-Rollen (Solution providers, Software providers, Infrastructure providers und Network service providers) gebildet (Farleit 2000, S. 4).

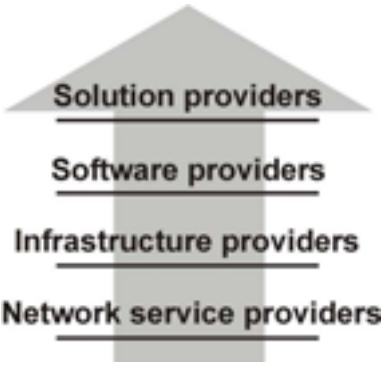

Abbildung 17: ASP-Schichtenmodell

Die angegebenen Provider sind nicht als real existierende Unternehmen zu betrachten, sondern als Rollen, die von Unternehmen eingenommen werden können. Nach dem Schichtenmodell können Unternehmen in einem ASP-Szenario eine, mehrere oder auch alle Rollen einnehmen (vgl. Farleit 2000; Tao 2000):

- *Network service providers* bieten Netzwerkdienste wie Kommunikation, Server Ressourcen und „value-added“ IP-Dienste an. Zur Kommunikation gehören physikalische Netzwerkressourcen (u. a. Kabel,

Switches, Router) und das damit verbundene Management von Kommunikation, Performance, Ausfallsicherheit und Security. Zu den Server Ressourcen gehören z. B. unterbrechungsfreie Stromversorgung, physikalische Sicherheit und technische Netzwerkdienste. „Value-added" IP-Dienste enthalten u. a. VPN, Netzwerk-Caching, Streaming Media, Firewalls und Directory Services.

- *Infrastructure providers* sind - auf dem Network service provider aufbauend - für die bereitzustellende Anwendung und die zusätzlich benötigte Software verantwortlich. Kann ein Infrastructure provider die Leistungen des Network service providers erbringen, wird er ASP Infrastructure Provider (AIP) genannt. Die AIPs bieten ihren Kunden (Service providers) zusätzlich Dienstleistungen an, um ihnen den Um- oder Einstieg bzw. die Möglichkeit, Anwendungen in einem ASP-Szenario anbieten zu können, zu erleichtern.

- *Software providers* stellen die Software her bzw. zur Verfügung, die im ASP-Szenario bereitstellt werden soll. Oder sie stellen Tools oder Frameworks zur Verfügung, mit denen ASP-Anwendungen gestaltet werden können.

- *Solution providers* führen die Leistungen der einzelnen Anbieter der verschiedenen Schichten zu Paketen zusammen und vermarkten diese mit entsprechenden Servicegarantien (Service Level Agreement – SLAs). Die Solution providers sind die Schnittstelle zwischen Anbieter und Kunden. Sie sind für den Kunden der „single point of contact" (Grohmann 2002, S. 71). Kunden nehmen die anderen Anbieter, die sich eventuell hinter dem Solution provider befinden, nicht wahr. So wird der Solution Provider auch Application Service Provider genannt.

4.1.2 Vor- und Nachteile

In der ASP-Literatur werden überwiegend die Vorteile für Kunden beim ASP herausgestellt. Seltener findet eine Auseinandersetzung mit den Nachteilen statt.

Die Vorteile von ASP aus Sicht des Kunden sind (vgl. Bleek und Pape 2001; Citrix 2000; CherryTree 1999; Endres 2004; Falkowski und Voß

2003; Grohmann 2002; Günther u. a. 2001; Mercer 2000; Sound 1999; Stahlknecht 2000; Tao 2000):

- *Konzentration auf Kernkompetenzen*: Die Konzentration auf Kernkompetenzen versetzt die beteiligten Parteien in die Lage, sich auf ihre eigentliche Arbeit zu konzentrieren, und setzt beim Kunden gebundene Ressourcen (Personal, Investitionen, usw.) frei, die anderweitig eingesetzt werden können.
- *Kosten reduzieren*: Mit ASP geht eine Kostenreduzierung, bezogen auf die ehemals intern erbrachten Leistungen, einher.
- *Hohe Kompetenz des Anbieters*: Der Anbieter vereint hohe Kompetenzen in seinem Bereich, die intern nicht hätten aufgebaut werden können.
- *Bessere Leistungen*: Der Anbieter bietet in der Regel das neueste Know-how, die neuesten Technologien und beides in einer besseren Qualität (24x7 Verfügbarkeit, höhere Performance, schnellere Reaktions- und Lösungszeiten bei Störungen usw.), als dies intern möglich wäre.
- *Kostentransparenz*: Im Gegensatz zu einer internen Lösung bringt ASP eine Kostentransparenz mit sich, die sich in den zu erbringenden, in Verträgen definierten Leistungen und deren Preisen begründet.
- *Flexibilität*: Bei Veränderungen des Kunden bzw. im Umfeld des Kunden kann sehr flexibel auf andere Anbieter zurückgegriffen werden, ohne Altlasten im eigenen Unternehmen „mitschleppen“ zu müssen.

Als Nachteile beim ASP werde folgende Punkte diskutiert (vgl. Grohmann 2002; Günther u. a. 2001; Picot und Jahn 2000; Rolf 2004b; Stahlknecht 2000):

- *Abhängigkeit*: Beim ASP begeben sich Kunden in ein starkes Abhängigkeitsverhältnis von dem Dienstleistungsanbieter bzw. sind stark von der Funktionsfähigkeit der Übertragungstechniken (in der Regel

Internettechnologien) abhängig. Treten beim Dienstleister oder auf dem Weg vom Dienstleister zum Kunden Probleme auf, so hat der Kunde keinen direkten Einfluss mehr auf die Softwarebereitstellung. Er muss warten, bis die anderen Beteiligten reagieren, meist ohne zu wissen, wer alles an „seiner“ Bereitstellung beteiligt ist.

- *Sicherheitsverlust*: Nicht nur der Kontrollverlust im Abhängigkeitsverhältnis beim ASP trägt zu einem Sicherheitsverlust des Kunden bei. Die persönlichen Daten der Kunden sind auf den Servern des Dienstleistungsanbieters oder weiterer Unterauftragnehmer gespeichert. Der direkte Zugriff geht den Kunden verloren, diesen Zugriff hat nur der Dienstleistungsanbieter. Damit sind hohe Sicherheitsanforderungen seitens der Kunden an den Dienstleistungsanbieter verknüpft. Keiner darf auf fremde, unternehmensinterne Daten zugreifen. Doch reichen diese Sicherheitsmechanismen? Gibt es ein Sicherheitsloch? Kann dem Dienstleistungsanbieter in letzter Instanz vertraut werden?

- *Kompetenzverlust*: Mit der Auslagerung der Softwarebereitstellung werden die einstmals dafür gebundenen Ressourcen frei. Monetäre Ressourcen werden eingespart, personelle Ressourcen rationalisiert oder umverteilt. Dies bedeutet u. a. die Abgabe der Kompetenzen, die sich im Bereich der Softwarebereitstellung aufgebaut haben, und erhöht das Abhängigkeitsverhältnis zwischen Kunde und Dienstleistungsanbieter weiter. Die Kunden können die Softwarebereitstellung nicht nur nicht mehr selbst leisten, sondern sind auch in Vertragsverhandlungen durch fehlendes Know-how dem Dienstleistungsanbieter unterlegen. So geht mit dem Kompetenzverlust auch ein Kontrollverlust einher (vgl. Endres 2004).

- *Risiko des Verlusts von Kernkompetenzen*: Neben dem Verlust von Kompetenzen mahnt Rolf (2004b) das Risiko des Verlusts von Kernkompetenzen an, „die Werte generieren, schwer imitierbar sind und die besondere Kompetenz der Firma und damit einer Volkswirtschaft ausmachen“ (Rolf 2004b, S. 11). Kernkompetenzen sind schwer zu bestimmen und werden meist erst im „Erfolgsfall“ sichtbar. Sie sind hochdynamisch und entwickeln sich kontinuierlich weiter. So kann

ein Unternehmen mit dem Outsourcing der Softwarebereitstellung u.U. Kernkompetenzen unwissentlich aufgeben (Rolf 2004b).

4.2 Architektur der Wertschöpfungskette

In der ASP-Literatur sind verschiedene Wertschöpfungsketten zu finden.

4.2.1 Einfache Wertschöpfungskette

Abbildung 18: Einfache Wertschöpfungskette

Die einfache ASP-Wertschöpfungskette besteht aus folgenden Komponenten (vgl. Balasubramanian u. a. 2002; Jaruzelski u. a. 2000):

- *Solution Provisioning* enthält die Herstellung entsprechender ASP-Anwendungen und der zusätzlichen Software, die ebenfalls zum Betrieb benötigt wird.
- *Solution Distribution* umfasst alle Fassetten der Bereitstellung einer Anwendung für Kunden, d. h. die komplette Betreuung der technischen Infrastruktur.
- *Service Integration* fokussiert auf die Integration der ASP-Dienstleistungen in die Organisation der Kunden. Diese Dienstleistungen sind u. a. Analyse der Unternehmensprozesse, Integration und Anpassung der Anwendung.
- *Costumer Interface*-Aktivitäten beinhalten die Akquirierung von Kunden und die Pflege der Kundenbeziehungen.

4.2.2 Technische Wertschöpfungskette

Die Komponenten der etwas ausdifferenzierteren technischen ASP-Wertschöpfungskette gruppiert Grohmann (2002, S. 64) wie folgt:

Abbildung 19: Technische Wertschöpfungskette

- *Netzwerkdienste*: Unter Netzwerkdienste fallen Transport & Access und die Network Operation.
- *Rechenzentrum*: Datacenter Operation und Application Operation gehören zum Rechenzentrum.
- *Application Support*: Die Application Management Maintenance bilden den Application Support.
- *Schnittstelle zum ASP-Anwender*: Die Gebiete System- und Business Integration und Customer Relation Management bilden zusammen die Schnittstelle zum Anwender.

4.2.3 Betriebswirtschaftliche Wertschöpfungskette

Abbildung 20: Betriebswirtschaftliche Wertschöpfungskette

Die organisatorischen und betriebswirtschaftlichen Aufgaben und Aspekte der betriebswirtschaftlichen ASP-Wertschöpfungskette aus Sicht des Application Service Providers sind im Einzelnen (vgl. Grohmann 2002, S. 66ff.):

- *ASP-Softwareentwicklung*: Bei der ASP-Softwareentwicklung ist aus Sicht des Application Service Providers speziell die ASP-Readyness der angebotenen Anwendungen wichtig. Dies gilt insbesondere für Anwendungen von Drittanbietern, die der Application Service Provider in sein Portfolio aufnimmt. Die ASP-Readyness von Anwendungen kann in speziellen Test- und Evaluierungszentren nachgewiesen werden.

- *Beratungs- und Vertragsgestaltung*: Ein Unternehmen wird entweder neue Anwendungsbereiche über einen Application Service Provider einführen oder bestehende Anwendungen zu einem Application Service Provider auslagern. In beiden Fällen wird das Unternehmen auf seine bestehende IT-Infrastruktur zurückgreifen. Der Application Service Provider muss für den Kunden Bedarfsspezifikationen erarbeiten und ASP-Anwendungen in die Infrastruktur des Kunden organisatorisch integrieren können. Eine Kosten-Nutzen-Analyse belegt dem Kunden die Vorteile oder Nachteile. Kommt es zur Vertragsgestaltung, müssen Service Level Agreements (SLAs) und Fallback-Lösungen definiert werden.

- *Hosting*: Das Hosting umfasst die Herstellung der Systemverfügbarkeit und die Kontrolle der technischen Bereitstellung sowie die Gewährleistung der Sicherheit der Daten.

- *Datenübertragung*: Der Weg vom Kunden bzw. Benutzer zum Server, auf dem die Anwendung installiert ist, kann nicht allein vom Application Service Provider bereitgestellt werden. Das bedeutet, dass er mit z. B. Telekommunikationsanbietern kooperieren muss. Dem Application Service Provider fällt dabei die Aufgabe zu, die Datenübertragung zu überwachen und zu koordinieren. Außerdem muss er für eine entsprechende Zugangsüberwachung und -verwaltung sorgen.

- *Anwendungsadministration*: In einem ASP-Modell wird eine Software vom Softwarehersteller dem Application Service Provider zur Vermietung an Dritte überlassen. Dieses lässt sich der Softwarehersteller vom Application Service Provider nutzungsabhängig entgelten. Zu diesem Zweck muss der Application Service Provider Nut-

zungsdaten sammeln und dem Softwarehersteller gegenüber legitimieren. Daneben müssen die Anwendungen auf Verfügbarkeit überwacht und gewartet werden. Zur Wartung zählt auch das Update-Management, das zyklische Updates möglichst ohne Ausfallzeiten organisieren muss.

- *User Support*: Als „single point of contact“ ist der Application Service Provider für alle Belange des Kunden erster Ansprechpartner. Dies bedeutet einerseits dem Kunden eine umfangreiche Benutzerbetreuung bieten zu können und andererseits eine Verrechnungsorganisation zu haben, die entsprechend benutzte Dienstleistungen in Rechnung stellen kann.

4.2.4 Kombinierte Wertschöpfungskette

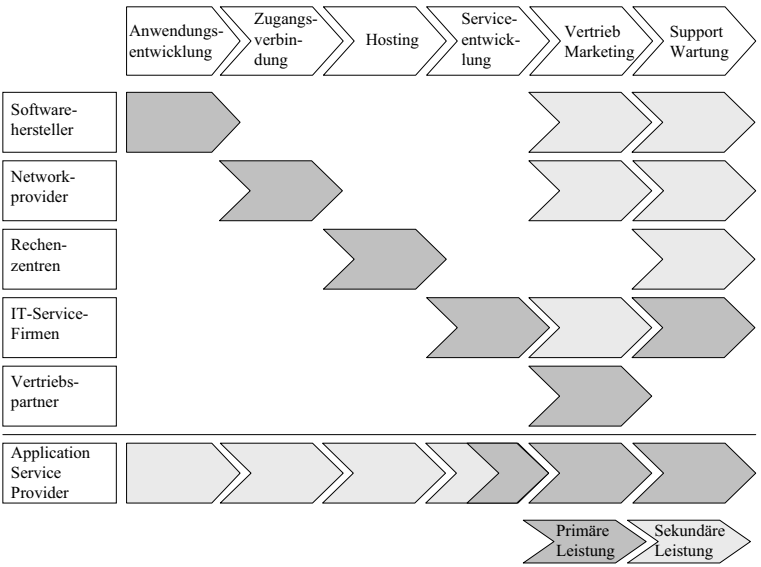

Abbildung 21: Kombinierte Wertschöpfungskette

Picot u. a. (2000) stellen eine Wertschöpfungskette für ASP vor, in der Aufgabenfelder in eine aufeinander aufbauende Reihenfolge gebracht und darüber hinaus mit möglichen Akteuren in Beziehung gesetzt werden. Die Aufgabenfelder im Einzelnen sind (vgl. Dechant u. a. 2004; Günther u. a. 2001; Picot u. a. 2000; Riemer und Ahlemann 2001):

- Die *Anwendungsentwicklung* umfasst die Herstellung und das Bug-Fixing einer Anwendung.
- Die *Zugangsverbindung* umfasst die Herstellung und Aufrechterhaltung einer Verbindung vom Kunden zur Anwendung.
- Das *Hosting* bezeichnet den kompletten Betrieb der technischen Infrastruktur inklusive angebotener Anwendungen.
- Die *Serviceentwicklung* bezeichnet die Installation und Bereitstellung von Serviceprozessen. Hier wird der technische und organisatorische Rahmen für eine umfassende Benutzerbetreuung geplant und realisiert.
- Unter *Vertrieb* und *Marketing* wird die Präsentation der angebotenen Anwendungen und Dienstleistungen auf dem Markt bzw. den potentiellen Kunden gegenüber verstanden, um Kunden zu akquirieren. Der Vertrieb konzentriert sich beim ASP im Gegensatz zum klassischen Vertrieb auf die Vertragsgestaltung und Abrechnung.
- *Support* und *Wartung* müssen bei einem gewonnenen Kunden durchgeführt werden. Hierunter fällt die Benutzerbetreuung und Softwarewartung.

Die Horizontale wird durch die Aufgabenfelder gebildet. In der Vertikale sind mögliche Akteure, die Kompetenzen in den verschiedenen Bereichen vorweisen können, aufgeführt. Alle Akteure haben neben Ihrer Kernkompetenz meist auch (als Marktteilnehmer) Kompetenzen in den Bereichen Marketing/Vertrieb und Support/Wartung, die sie in ein ASP-Modell einbringen können. IT-Service-Firmen sind auf Kunden- und Benutzerbetreuung spezialisiert.

Der Application Service Provider muss in erster Linie Kompetenzen im Bereich Marketing, Vertrieb und in Teilen auch Support haben, da er die

Anwendung als Dienstleistung den Kunden und Nutzern in einem ASP-Modell anbieten will. Kompetenzen in anderen Aufgabenfeldern sind möglich, können aber auch mit Hilfe der genannten Akteure hinzugekauft werden.

4.3 Ertragsmodell

Hinsichtlich des Ertragsmodells ist interessant, was in welcher Form Kunden als Services angeboten wird und wie diese Dienstleistungen in Rechnung gestellt werden können. In der ASP-Literatur findet sich dazu die Präsentation einer ASP-Marktlandschaft, die verschiedene Arten von Anwendungen mit verschieden quantitativen und qualitativen Serviceangeboten in Beziehung setzt. Weiterführend werden in der ASP-Literatur ertragsgenerierende Ereignisse diskutiert, die einmalig, wiederkehrend oder nutzungsabhängig abgerechnet werden können.

4.3.1 Services

Folgende Arten von Anwendungen können bei der Bereitstellung in einem ASP-Szenario unterschieden werden (sortiert vom Einfachsten zum Komplexesten) (vgl. Gillian u. a. 1999, S. 5f):

- *Personal Applications* sind z. B. Office-Anwendungen und andere Anwendungen für Einzelpersonen aus den Bereichen Spiele, Unterhaltung, Heimarbeit usw.
- Zu *Collaborative Applications* zählen Groupware, E-Mail, Konferenzsysteme usw.
- Mit Commerce Applications sind typische e-Commerce-Anwendungen gemeint: Onlineshops, Internetbanking usw.
- *CRM Applications* enthalten z. B. Kundenservice, Marketingapplikationen, Sales Force Automation.
- *ERM Applications* übernehmen Aufgaben in den Bereichen Rechnungswesen, Personalwesen, Lagerhaltung, Facility Management etc.

- *Vertical Applications* sind branchenspezifische Lösungen, z. B. Patientenabrechnungssysteme in der Krankenhausbranche oder Workflowsysteme in der Versicherungsbranche.

- Zu den *Analytic Applications* zählen z. B. Finanzanalysen und Risikoanalysen, d. h. Anwendungen, die Unternehmen bzw. Unternehmensprobleme analysieren können.

Ähnliche Klassifizierungen von ASP-fähigen Anwendungen finden sich bei Dechant u. a. (2004) und Liess (2000).

Die verschiedenen Arten von Anwendungen können in drei Dienstleistungsstufen angeboten werden (vgl. Gillian u. a. 1999; Riemer und Ahlemann 2001):

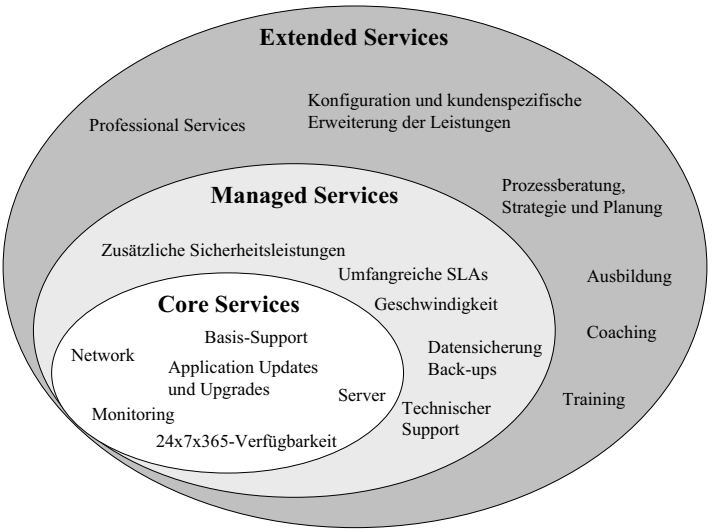

Abbildung 22: ASP-Leistungssystem

- *Core Services*: Die Basisdienstleistungen umfassen die ASP-Kernleistungen: Hosting, Monitoring und Wartung der Anwendung. Es

wird die ständige Verfügbarkeit garantiert und Updates bzw. Upgrades werden eingespielt. Meist gehört eine minimale Benutzerbetreuung zu den Basisleistungen, um ein Mindestmaß an Kundenzufriedenheit zu erreichen.

- *Managed Services*: Die Managed Services umfassen die Core Services und weitere Leistungen, die für eine stabile, schnelle und sichere Benutzung notwendig sind. Dazu gehören skalierbare Dienstleistungen hinsichtlich Geschwindigkeit, weitgehender technischer Support und die Sicherung der kundenindividuellen Daten. Diese skalierbaren Leistungen werden beim ASP in Service Level Agreements (SLAs) als Teil von Verträgen schriftlich fixiert.

- *Extended Services*: Die Extended Services erweitern die Managed Services um Konfiguration und Customizing sowie eine umfangreiche Benutzerbetreuung (Schulung, Training, Coaching, usw.). Weiter zählen hierzu Professional Services, wie Prozessberatung, sowie Strategie und Planung des Einsatzes der ASP-Anwendungen beim Kunden.

ASP-Angebote verknüpfen die Art der Anwendung mit entsprechenden Dienstleistungsstufen und positionieren sich größtenteils in den grau ausgezeichneten Quadranten der ASP-Marktlandschaft in Anlehnung an Gillian u. a. (1999, S. 5):

- *Personal ASP*: Im Personal ASP werden einfache, sofort benutzbare Anwendungen (Office-Software, Termin- und Reiseplanung, E-Mail, Adressenverwaltung, Desktop Publishing [DTP]) angeboten. Hohe Kundenzahlen und das damit verbundene hohe Datenaufkommen sind typisch für Personal ASP, genauso wie nur eine minimale Benutzerbetreuung und vergleichsweise keine weiterreichenden Dienstleistungen.

- *Collaborative ASP*: Im Collaborative ASP wird Groupware im weitesten Sinne angeboten: Conferencing, Unified Messaging, Projektmanagement, Sales Automation, Onlineshops usw. Hier steht insbesondere die Verfügbarkeit der Anwendungen im Fokus, die durch entsprechende SLAs schriftlich in Verträgen fixiert werden. Hinzukommen noch Leistungen der Sicherheit und des technischen Supports.

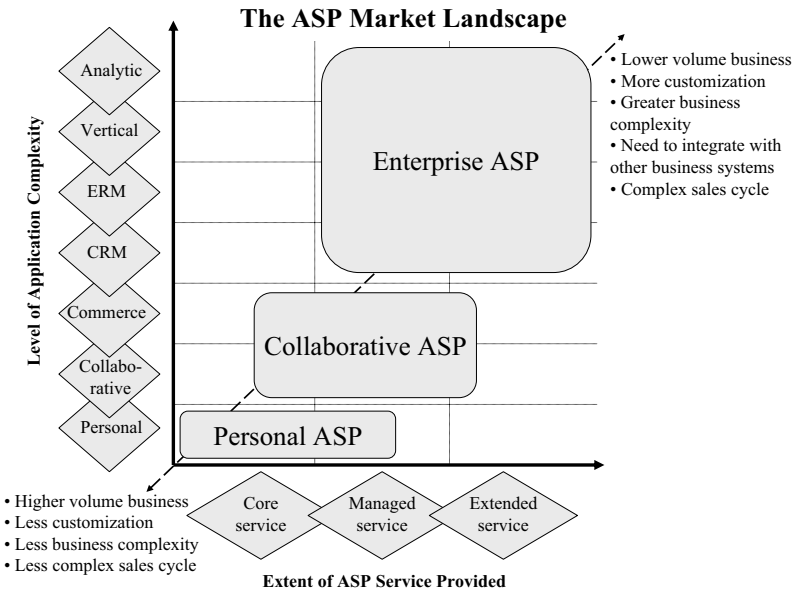

Abbildung 23: ASP-Marktlandschaft

- *Enterprise ASP*: Im Enterprise ASP sind insbesondere hohes Applikations- und Branchenwissen sowie Erfahrungen in der Softwareintegration wichtig. Angebotene Anwendungen sind hier meist analytische und branchenspezifische Lösungen sowie komplexe CRM- und ERP-Systeme. In diesem Fall sind auch die Professional Services und eine umfassende Benutzerbetreuung wichtig.

4.3.2 Abrechnungsmöglichkeiten

Die gerade erläuterten Services müssen über entsprechende Gebühren abgerechnet werden:

- *Einmalige Gebühren* werden einmal, meist am Anfang der Bereitstellung, für Einrichtungsarbeiten in Rechnung gestellt. Die Einrichtung eines Telefonanschlusses lassen sich z. B. Telekommunikationsunternehmen mit einer Einrichtungsgebühr vergüten.

- *Wiederkehrende Gebühren* werden regelmäßig erhoben und sind von der tatsächlichen Nutzung unabhängig. Die Internet-Flatrate ist ein typisches Bespiel für eine wiederkehrende, nutzungsunabhängige Gebühr.

- Die *nutzungsabhängige Gebühr* wird in Abhängigkeit zur Nutzung, d. h. pro wirklich genutzten Zeiteinheiten oder Volumeneinheiten, erhoben. Internet-By-Call Angebote fallen beispielsweise unter die nutzungsabhängigen Gebühren.

Gebührenpflichtig sind u. a. (vgl. Grohmann 2002, S. 72):

- *Daten-Verkehr (Traffic)*: Applikationsbezogene Ereignisse, z. B. das Eintragen eines Beitrags oder das Herunterladen von Informationen, erzeugen Datenverkehr, der nach Zeit oder Volumen nutzungsabhängig abgerechnet werden kann.

- *Administration*: Die Administrationskosten beim Application Service Provider sind größtenteils Personalkosten für „Mitarbeiter im Data Operation Center, Network Operation Center und Security Center, also in den Überwachungseinheiten für die unterschiedlichen Bereiche des Rechenzentrums“ (Grohmann 2002, S. 75). Diese Kosten treten monatlich auf und werden meist nutzungsunabhängig verrechnet.

- *Lizenzen*: Eventuell auftretende Kosten bzgl. Software-Lizenzen werden unterschiedlich auf den Kunden umgelegt.

  1. Lizenzen werden direkt an bestimmte Anwender (named user) gebunden und einmalig oder wiederkehrend abgerechnet.
  2. Lizenzen werden nutzungsabhängig entsprechend der gleichzeitigen Nutzung (concurrent user) in Rechnung gestellt.
  3. In einer ressourcenabhängigen Abrechnung wird die Lizenzgebühr nach Prozessor bzw. Central Process Unit (CPU) einmalig, wiederkehrend oder nutzungsabhängig erhoben.

- *Wartung*: Die Wartung (Kauf, Austausch, Reparatur, Installation, Updates usw.) der Rechenzentrumskomponenten erzeugt Kosten, die dem Kunden monatlich bzw. nutzungsabhängig in Rechnung gestellt werden.

- *Support*: Support umfasst Leistungen, die der Application Service Provider direkt für den Kunden bzw. Nutzer anbietet, z. B. Benutzersupport, Schulungen, Coaching und Seminare. Diese Leistungen werden unterschiedlich abgerechnet.

4.4 Rechtliche Aspekte

Verträge zwischen beteiligten Akteuren bei der Bereitstellung von Software haben u. a. zwei Bedeutungen. Zum einen dienen sie der „Begründung eines Schuldverhältnisses“ (BGB 2002, §311), denn nur mit Verträgen können Vertragsparteien, bei unterlassener Erfüllung der vertraglich geregelten Verpflichtungen, auf Schadensersatz klagen. Dabei haben die Vertragspartner auch für das Verschulden ihrer Mitarbeitenden zu haften (Schneider und Themann 2002, S. 149). Für die Begründung eines Schadensersatzanspruchs muss, neben dem Verstoß gegen die vertraglichen oder gesetzlichen Pflichten, ein Schaden nachgewiesen werden. Zum anderen dienen Verträge der Überlassung von Nutzungsrechten von Software. In so genannten Nutzungsvereinbarungen (= urheberrechtliche Lizenzen) wird die Übertragung der Rechte vom Programmierer über die Entwicklerorganisation und Application Service Provider zum Kunden geregelt:

- *Programmierer* $\Rightarrow$ *Entwicklerorganisation*: Erschafft ein Programmierer, also ein Angestellter der Entwicklerorganisation, eine Software, dann ist dieser der Urheber. Allerdings stehen dem Arbeitgeber laut Gesetz (vgl. Schneider 2002, S. 143) bei Bestehen eines Arbeits- oder Dienstverhältnisses alle vermögensrechtlichen Befugnisse an der Software zu. Insofern liegen die Verwertungsrechte an dem Werk bei der Entwicklerorganisation.

- *Entwicklerorganisation* $\Rightarrow$ *Application Service Provider*: Um die Anwendung Kunden zur Verfügung stellen zu können, muss der Application Service Provider sich umfassende Nutzungsrechte von der Entwicklerorganisation sichern. Hierzu gehören speziell das Recht zur Verbreitung in jeder Form und das Recht zur öffentlichen Wiedergabe (vgl. Schneider 2002, S. 142).

- *Application Service Provider* $\Rightarrow$ *Kunde*: Der Application Service Provider will die Anwendung möglichst vielen Kunden zur Verfü-

gung stellen. Deshalb wird er seinen Kunden nur ein einfaches und nicht ein ausschließliches Nutzungsrecht gewähren, da es im Widerspruch zur parallelen Nutzung durch mehrere Kunden steht.

In der ASP-Literatur wird hauptsächlich der Vertrag zwischen Application Service Provider und Kunden betrachtet. Dabei wird ASP oberflächlich als Vermietung von Software gesehen, im Gegensatz zu herkömmlichen Formen der Softwareüberlassung, die juristisch oft einem Kaufvertrag entsprechen. Vermietung heißt in dieser vereinfachten Betrachtung, dass mehrere Kunden (Mieter) eine Anwendung vom Application Service Provider (Vermieter) für einen bestimmten Zeitraum mieten. Zwipf und Schönfelder (2002, S. 123) kritisieren an dieser Betrachtung, dass sie einerseits der Komplexität von ASP nicht gerecht wird und andererseits juristisch umstritten ist. So wird im Folgenden auf verschiedene Vertragsarten und auf für ASP typische Service Level Agreements (SLAs) eingegangen.

4.4.1 Vertragsarten

Die im Vertragsrecht des BGB (2002) definierten Vertragstypen sind seit 1896 unverändert geblieben. Der deutsche Gesetzgeber sah keine Veranlassung, Typen bestimmter IT-Verträge gesetzlich zu regeln, da die im Gesetz geregelten Vertragstypen auch im ASP vorkommen und insofern anzuwenden sind (Zwipf und Schönfelder 2002, S. 123). Die für ASP möglichen Vertragsarten sind (vgl. König 1996):

- *Miete*: Durch den Mietvertrag wird der Vermieter zur Überlassung des Gebrauchs einer Sache, der Mieter zur Zahlung des entsprechenden Entgelts verpflichtet. Für ASP ist problematisch, dass der Mietvertrag den direkten Zugriff auf die Mietsache fordert, der bei ASP nicht gegeben ist. Die gemietete Anwendung ist beim Application Service Provider oder weiteren Partnern installiert und steht dem Kunden juristisch gesehen nie vollständig zur Verfügung. Die „Zugangsmöglichkeit“ ist hierbei nicht mit dem juristisch gemeinten „Zugriff“ gleichzusetzen. Außerdem hat der Vermieter „[...] die vermietete Sache dem Mieter in einem zu dem vertragsmäßigen Gebrauch geeigneten Zustand zu überlassen und sie während der Mietzeit in diesem Zustand zu erhalten“ (BGB 2002, §536). Diese dauerhafte

Gewährleistung schließt alle Pflegeleistungen, die der Application Service Provider sich normalerweise bezahlen lässt, mit ein.

- *Pacht*: Die Pacht ist der Miete sehr ähnlich, mit dem Unterschied, dass dem Pächter nicht nur der Gebrauch, sondern auch der „[...] Genuss der Früchte, soweit sie nach den Regeln einer ordnungsmäßigen Wirtschaft als Ertrag anzusehen sind, während der Pachtzeit zu gewähren“ (BGB 2002, §581 (1)) ist.

- *Leasing*: Leasing ist als besonderer Vertragstyp von der Rechtsprechung anerkannt und bezeichnet den Verkauf einer Sache vom Hersteller an einen Finanzier, der die Sache an den Leasingnehmer für die Dauer des Leasingvertrages überlässt. Das Leasing ist der Miete sehr ähnlich, wobei noch zwischen Finanzierungsleasing und Operatingleasing unterschieden wird. Beim Finanzierungsleasing ist die oben genannte Dreierkonstellation fest vorgeschrieben, während im Operationleasing auch mehreren Leasingnehmer eine Sache zur Verfügung gestellt werden kann.

- *Werkvertrag*: Beim Werkvertrag wird der Unternehmer zur Herstellung des versprochenen Werkes und der Besteller zur Entrichtung der vereinbarten Vergütung verpflichtet (vgl. BGB 2002, §631). Die zeitliche Befristung von Dienstleistungen steht hier dem Werkvertrag nicht entgegen, doch „Gegenstand des Werkvertrages kann sowohl die Herstellung oder Veränderung einer Sache als ein anderer durch Arbeit oder Dienstleistung herbeizuführender Erfolg sein“ (BGB 2002, §631 (2)). Problematisch für ASP ist hier, dass die Vergütung erst fällig wird, wenn der Besteller das Werk abnimmt, d. h. der Besteller einen (wirtschaftlichen) Erfolg erfahren hat. Mit dieser Verpflichtung zum Erfolg geht eine umfassende Schadensersatzverpflichtung einher.

- *Dienstvertrag*: Durch den Dienstvertrag wird der Dienstverpflichtete zur Leistung der versprochenen Dienste und der Dienstberechtigte zur Entrichtung der vereinbarten Vergütung verpflichtet (BGB 2002, §611). Im Gegensatz zum Werkvertrag schuldet der Dienstleister nur die Tätigkeit, nicht den Erfolg. Problematisch ist hier, insbesondere für den Kunden, die nur geringe Haftungsfähigkeit des Dienstleistungsanbieters.

Jeder genannte Vertragstyp ist mehr oder weniger gut auf ASP anzuwenden. Den einen Vertragstypus, der am besten zu ASP passt, gibt es nicht. Der ASP-Vertrag entspricht vielmehr einer Mischform, bestehend aus einem Hauptvertrag als Rahmen und Einzelverträgen zu den einzelnen Dienstleistungen (Zwipf und Schönfelder 2002, S. 126f.). Denn „die vertraglichen Pflichten der Parteien eines ASP-Vertrages ergeben sich nicht daraus, was man als ASP-Vertrag zu verstehen hat, sondern daraus, was der Inhalt der jeweiligen Leistungspflicht ist, über welche sich die Vertragspartner verständigt haben, und welche sie in gedanklichen oder ausdrücklich erwähnten Leistungspaketen zu einem ASP-Vertrag geschnürt haben“ (Zwipf und Schönfelder 2002, S. 126).

Dies bedeutet für den ASP-Vertrag, dass die Einzelverträge einem der oben genannten Vertragstypen entsprechen können und bei der Separierung von Einzelleistungen diese juristisch unabhängig vom Typus des Rahmenvertrages zu handhaben sind. Allgemeine Vertragselemente für den Rahmenvertrag eines ASP-Vertragwerks sind bei Zwipf und Schönfelder (2002, S. 129ff.) zu finden.

4.4.2 Service Level Agreements

Service Level Agreements (SLAs) werden beim ASP zur schriftlichen Fixierung von Qualität und Quantität von Dienstleistungen zwischen Application Service Provider und Kunden benutzt und sind daher oft Bestandteil in ASP-Verträgen.

Rosenhagen (2002, S. 169) definiert SLAs wie folgt: „SLA sind Vereinbarungen zwischen einem IT-Leistungserbringer (z. B. einem ASP oder einer IT-Abteilung) und einem Unternehmen oder anderen Fachabteilungen über die Qualität und Quantität von Dienstleistungen, in denen die gegenseitigen Rechte und Pflichten für eine gewisse zeitliche Dauer vereinbart werden und zwar mit dem Zweck, Serviceziele zu definieren, sie mit messbaren Kriterien zu beschreiben und der Möglichkeit der Leistungsauswertung unter Anwendung vereinbarter Messmethoden.“

Das jeweilige Regelbedürfnis in einem SLA richtet sich in erster Linie zunächst nach den jeweils individuell zwischen IT-Leistungserbringer und Kunde vereinbarten Leistungen und der zugrunde gelegten technischen Ausgestaltung (vgl. Rosenhagen 2002, S. 171). d. h. einheitliche SLAs gibt es nicht. Vielmehr sind in der Praxis Basis-SLAs zu finden, auf die indivi-

duelle, bedarfsgerechte SLAs im Falle einer Kooperation aufgesetzt werden (vgl. Günther u. a. 2001, S. 557).

Diese Basis-SLAs bestehen aus typischen Bestandteilen (vgl. Rosenhagen 2002; Falkowski und Voß 2003), die meist in unterschiedlichen Servicestufen, z. B. Standard, Enhanced und Business Critical (Burris 2002, S. 253ff.), angeboten werden. Einen Leitfaden für SLAs hat die Information Technology Association of America (ITAA 2000) herausgegeben. Eine kritische Auseinandersetzung und ein Vorgehensmodell zur Spezifikation von SLAs findet sich bei Trieneken u. a. (2004).

Netzwerk-SLA
Das Netzwerk-SLA bestimmt die vereinbarte Leistungsgüte aller Netzwerkkomponenten und -prozesse. Zum Netzwerk-SLA zählt u. a.:

- *Verfügbarkeit und Ausfallzeiten*: Die Verfügbarkeit wird üblicherweise in Prozent angegeben. Bezugsgröße ist hier ein konkret definierter Zeitraum. 99% garantierte Verfügbarkeit bedeuten einen tolerierten Ausfall von ca. 88 Stunden im Jahr, das sind rund 100 Minuten pro Woche. Geplante Ausfallzeiten (z. B. Einspielung von Updates) sind in dieser Zeit meist berücksichtigt, sollten aber explizit aufgeführt werden. Wichtig ist darüber hinaus die klare Differenzierung der Verfügbarkeit des Netzwerks von der Verfügbarkeit der konkreten Anwendung und des Servers. Die Ausfälle summieren sich beim Kunden intransparent, so dass bei Schuldzuweisungen eine klare Differenzierung hilft, den Schuldigen zu finden.

- *Netzwerkausstattung und Netzwerkarchitektur*: Ausstattung und Architektur des Netzwerks sind grundlegende Voraussetzungen. Erwartet werden in aller Regel skalierbare und redundante Netze.

- *Netzsicherheit*: Zur Netzwerksicherheit zählt die Sicherung der Verbindung und der „demilitarized zone (DMZ)“ (siehe Abschnitt 4.5.3). Hier müssen Sicherheitsmechanismen wie z. B. Firewalls, Verschlüsselung und Tunneling definiert werden.

- *Datendurchsatz und Antwortzeiten*: Der Datendurchsatz und die Antwortzeiten hängen von der Qualität und der Bandbreite des Übertragungsmediums ab. Eng damit zusammen hängen Antwortzeiten und

Verlust von Datenpaketen. Lange Antwortzeiten (Latenz) wirken sich schlecht bei Streaming Medien (Audio oder Video) aus. Verlustfreie Datenübertragung wird, wie die Verfügbarkeit, in Prozent gemessen. In der Regel werden 99% Übertragungssicherheit angeboten, d. h. dass 1% der Datenpakete verloren gehen können.

Hosting-SLA
Zur Hosting-SLA gehört die Festlegung der Leistungsgüte aller Hardware-Komponenten und der System-Software:

- *Verfügbarkeit und Antwortzeiten*: Ähnlich wie beim Netzwerk adressiert dieser Punkt die Verfügbarkeit der Server bzw. der benötigten Hard- und Software. Dabei hängen die Verfügbarkeit und insbesondere die Antwortzeiten von den gleichzeitig zugreifenden Nutzern maßgeblich ab. Antwortzeiten werden in (Milli-)Sekunden und die Verfügbarkeit in Prozent eines definierten Zeitbereichs angegeben.

- *Datensicherheit*: Zur Datensicherheit im Hosting-SLA gehören insbesondere alle Maßnahmen zur Vorbeugung von Datenverlusten: z. B. das Durchführen, Lagern und bei Bedarf das Wiedereinspielen von Back-ups. Es können Lagerungszeiträume, Reaktionszeiten und die Häufigkeit von Backups bestimmt werden.

- *Physische Sicherheit*: Neben der „virtuellen Sicherheit“ spielt die physische Sicherheit beim ASP eine Rolle. Hier kann insbesondere auf Sicherheitspersonal, Überwachungstechnik, Brandschutz, direkte Verbindung zu und Reaktionszeit von Feuerwehr und Polizei usw. geachtet werden.

Application-SLA
Das Application-SLA adressiert die Güte der angebotenen Anwendung. Hierzu zählen folgende Aspekte:

- *Verfügbarkeit*: Die Verfügbarkeit spielt insbesondere auf die Absturzsicherheit der bereitgestellten Software an. Hierbei ist zu beachten, dass die Anwendungsverfügbarkeit in engem Kontakt zur Netzwerk- und Serververfügbarkeit steht. Benutzende haben eine End-to-End-Sichtweise und keine Component-to-Component-Sichtweise. Das bedeutet, dass dem Nutzer nicht transparent ist, an welcher Komponente die Nutzung scheitert, wenn eine Anwendung nicht verfügbar ist.

- *Schnelligkeit*: Wichtig sind bei der Anwendung die Antwortzeiten. Die Performance muss unter dem Blickwickel von großen Lasten bewertet werden.

- *Sicherheit*: Sicherheit bezieht sich hier auf die Mehrbenutzer- und Mandantenfähigkeit der Anwendung, d. h. der Qualität der Abschirmung von Kundendaten vor unbefugtem Zugriff. Dieser Punkt streift insbesondere viele Aspekte des Datenschutzes.

- *Updates und Upgrades*: Hier wird definiert, welche Version der Anwendung bereitgestellt wird und wie viele bzw. wann Updates und Upgrades eingespielt werden. Das Einspielen von Patches zur Behebung von Fehlern und Sicherheitslücken ist meist kostenlos.

- *Eigentumsrecht*: Es sollte vertraglich geregelt werden, dass die Kundendaten Eigentum des Kunden sind, auch wenn sie nicht auf deren Servern, sondern, von den Kunden aus gesehen, auf externen Servern liegen.

Support-SLA
Zum Support-SLA gehören verschiedene Kategorien von Betreuungsleistungen, die u. a. nach Verfügbarkeit, Reaktionszeit, Zeitaufwand und/oder Mitarbeiterzahl bewertet werden können. Dabei kommen unterschiedliche Hilfsmittel, wie Help-Desk, Call-Center oder Monitoring- und Reportingtools, zum Einsatz. Hinter allen Maßnahmen steht dabei die Kundenzufriedenheit als höchstes Ziel:

- *Reaktiver Benutzersupport*: Der Benutzersupport reagiert auf Benutzeranfragen per Telefon oder E-Mail z. B. auch mittels eines Call-Centers.

- *Aktive Anwendungsunterstützung*: Die Anwenderunterstützung bietet von sich aus Schulungen, Coaching, Workshops, Diskussionsforen, Seminare usw. an.

4.5 Technik

Bei der Betrachtung der Technik von ASP werden in der ASP-Literatur drei Bereiche als relevant erachtet. Zum Ersten die Basistechnologie, auf der

Anwendungen bereitgestellt werden. Zum Zweiten die Anwendung selbst, und zum Dritten braucht ASP eine geeignete technische Infrastruktur, die die Bereitstellung einer Anwendung und zusätzliche benötigte Anwendungen (z. B. Abrechnungs- und Sicherungssysteme) realisiert.

4.5.1 Basistechnologie

Als technische Grundlage von ASP dient die Internettechnologie (Eilingsfeld und Schätzler 1997; Gralla 1997; Schiffer und Templ 2002; Steinmetz und Mühlhäuser 2002) und hier insbesondere die Protokoll TCP (Transport bzw. Transmission Control Protocol) und IP (Internet Protocol), bekannt als TCP/IP (RFC1180 1991).

Außerdem setzt ASP auf dem Client/Server-Prinzip auf. Anwendungen, d. h. die Funktionalität und die Daten, befinden sich zentral auf einem Server. Eine redundante Installation von mehreren Servern wird als eine zentrale Installation betrachtet. Ein Client kann mittels TCP/IP, darauf aufsetzenden Protokollen und entsprechender Client-Software auf die zentral installierten Anwendungen zugreifen. Beim Zugriff des Clients auf die serverseitigen Anwendungen kann zwischen zwei Vorgehensweisen unterschieden werden: Server-based Computing und Web-Applikationen (vgl. Citrix 2000; Grohmann 2002; Tao 2000).

Server-based Computing
Terminalprogramme (z. B. Microsoft Windows Terminal Server [WTS], Citrix Systems Independent Software Architecture [ICA], Tarantella von der Firma Tarantella) erlauben es, normale Anwendungen auf einem Server von einem Client aus fernzusteuern. Das bedeutet, dass die (z. B. Tastatur-) Eingabe des Nutzenden und die (z. B. Bildschirm-)Ausgabe der Anwendung vom Terminalprogramm mittels Internet-Technologien vom Client zum Server übertragen werden. Die Anwendung und die Eingabegeräte (Tastatur, Maus, usw.) erkennen nicht, dass sie nicht zum selben Computer gehören. Die Anwendung muss in diesem Fall nicht speziell für den entfernten Betrieb angepasst bzw. programmiert worden sein. Vorteile des Server-based Computing sind u. a.:

- Bereits existierende, internet-untaugliche Anwendungen können mittels Internettechnologie bereitgestellt werden.

- Ressourcenfressende Anwendungen sind auf alter, eigentlich nicht mehr unterstützter Hardware wieder verfügbar, da die eigentliche Anwendung auf einem entsprechend leistungsfähigen Server bereitsteht.

Nachteile des Server-based Computing sind u. a.:

- Für jeden Anwender muss eine eigene Session gestartet werden. Simultane Zugriffe auf Anwendungen finden nicht statt, d. h., dass je nach Anwendung und gleichzeitig erfolgenden Zugriffen, die Leistungsfähigkeit des Servers stark beansprucht und schnell ausgereizt wird.

Web-Applikationen
In diesem Fall sind die angebotenen Anwendungen speziell für die entfernte Benutzung in größtenteils Web-Programmiersprachen (Java, JavaScript, PHP, Perl, Python usw.) programmiert. Der Zugriff vom Client auf die Anwendung erfolgt in der Regel mit einem Webbrowser, eventuell mit Zuhilfenahme verschiedener Plug-ins. Vorteile von Web-Applikationen sind u. a.:

- Ein ressourcenschonenderer Betrieb, da die benötigten Infrastrukturkomponenten in der Regel gut skalierbar sind und ein Load-Balancing einfacher einzurichten ist.
- Die Web-Applikation ist nicht durch ein umgebenes Terminalprogramm eingeschränkt, sondern kann die Möglichkeiten der Internettechnologie voll ausschöpfen.

Nachteile von Web-Applikationen sind u. a.:

- Web-Applikationen müssen entwickelt werden.
- Große und komplexe Anwendungen in Web-Programmiersprachen zu implementieren ist wesentlich aufwändiger, wenn nicht sogar unmöglich.

4.5.2 Anwendungen

Bei der Bereitstellung von Anwendungen in einem ASP-Szenario ist ein wesentlicher Faktor der Kostenersparnis die Tatsache, dass eine Anwendung vielen Nutzenden zur Verfügung gestellt wird. Das bedeutet, dass verschiedene Personen dieselbe Anwendung auf derselben Infrastruktur nutzen, ohne von den Mitnutzenden etwas zu bemerken. Diese 1:N-Bereitstellung zieht verschiedene Anforderungen an die Anwendung nach sich.

Mehrbenutzer- und Mandantenfähigkeit
Die Anwendung muss mehrbenutzerfähig sein. Sie muss mit geeigneten Authentisierungs- und Sicherungsmechanismen den Nutzenden eindeutig identifizieren und ihm nur seine Daten präsentieren. Bei Kooperationsanwendungen muss zusätzlich zur Mehrbenutzerfähigkeit eine Mandantenfähigkeit gegeben sein, so dass Nutzende unterschiedlicher Gruppen bzw. Unternehmen nur Zugriff auf die Daten ihrer Gruppenmitglieder bzw. Kollegen bekommen und nicht Daten von fremden Mandanten. Insofern gehört zu jeder ASP-Anwendung eine Zugriffsverwaltung, die Nutzende eindeutig identifiziert (meist mit Kennung und Passwort) und sie verschiedenen Gruppen bzw. Mandanten zuordnen kann.

Ein weiterer Aspekt der Mehrbenutzerfähigkeit und insbesondere der Mandantenfähigkeit ist die Anpassung der Anwendung in Funktionalität und Benutzungsschnittstelle an die persönlichen Anforderungen bzw. die Anforderungen der entsprechenden Mandanten. Dieser Aspekt steht allerdings in einem gewissen Widerspruch zur Kostenersparnis. Eine große Flexibilität zieht eine entsprechende Komplexität mit sich, die die Grenze zwischen der Bereitstellung von einer Anwendung für viele Kunden, zu einer Bereitstellung von einer Anwendung für einen Kunden, überschreiten lässt. Beispielsweise können Updates oder Supportanfragen bei einer hochkomplexen Anwendung nicht mehr für alle Kunden gemeinsam geleistet werden. Der Einsparungsvorteil wäre dahin.

Gewisse Anpassungen sind aber erwünscht, so dass bei ASP-Anwendungen oft ein Mittelweg gewählt wird. Die angebotene Funktionalitätsvielfalt ist fest vorgegeben. Sie kann eventuell ausgeblendet und an unternehmenskritische Prozesse und Abläufe leicht angepasst werden. Die Benutzeroberfläche wiederum ist hinsichtlich des „Look and Feel“ sehr flexi-

bel an die Bedürfnisse der Nutzenden bzw. der Mandaten über Template-Techniken anpassbar.

Plattformunabhängigkeit
Plattformunabhängigkeit bedeutet in diesem Fall die Betriebs- und Nutzungsmöglichkeit auf unterschiedlicher Hard- und Software.

Auf Serverseite muss eine Plattformunabhängigkeit nicht unbedingt gegeben sein, da die zentral bereitgestellte Infrastruktur auf Seiten des Bereitstellers (in der ASP-Literatur Application Service Provider genannt) in der Regel nicht gewechselt wird. Auf der Nutzerseite ist eine Plattformunabhängigkeit unabdingbar, da so ein großer Kundenkreis erschlossen werden kann und die Kunden beliebige Wechsel ihrer eigenen IT-Infrastruktur vollziehen können. d. h. durch die clientseitige Plattformunabhängigkeit ist die Infrastruktur des Nutzers bzw. Mandanten komplett unabhängig von der Infrastruktur des Bereitstellers.

Eine Plattformunabhängigkeit wird bei ASP-Anwendungen via Terminalprogramme als auch bei Web-Applikationen in der Regel durch die plattformübergreifende Verbreitung der Client-Software (Terminal-Client als eigenen Software oder Plug-In bzw. Webbrowser und eventuell Plug-ins) erreicht. Bei Web-Applikationen muss zusätzlich auf internationale Standards geachtet werden, um eine breite Unterstützung vorhandener Webbrowser zu erreichen.

Das Aufsetzen einer ASP-Anwendung auf internationalen Standards ist ebenfalls Grundlage für die Integration der ASP-Anwendung mit anderen Anwendungen des Kunden bei möglicherweise anderen Application Service Providern oder mit beim Kunden lokal installierten Anwendungen.

4.5.3 Technische Infrastruktur

Zum Betrieb einer Anwendung in einem ASP-Modell gehört eine geeignete technische Infrastruktur, die zum einen das Zusammenspiel der internen Komponenten und den gesicherten Zugriff von außen realisiert (vgl. Grohmann 2002, S. 51ff.).

Interner Bereich
Um eine Anwendung in einem ASP-Modell technisch zu betreiben, sind verschiedene Komponenten (Server) erforderlich, die bei Bedarf auch zu Clustern zusammengefasst werden können. Die Zusammensetzung kann je

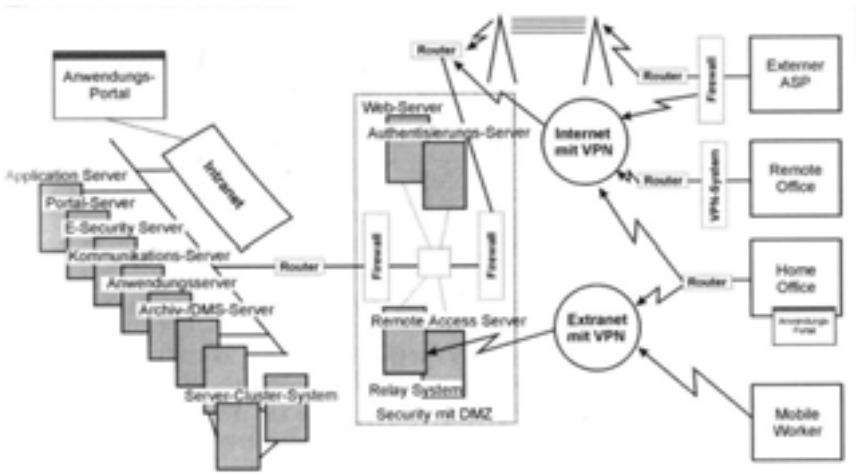

Abbildung 24: Technische Infrastruktur beim ASP

nach Kontext unterschiedlich sein. Eine detaillierte Betrachtung der Kommunikation zwischen den verschiedenen Komponenten über Middleware-Techniken wie z. B. CORBA, Enterprice JavaBeans oder Microsoft DNA findet sich bei Tao (2000).

In der Regel kommen folgende Komponenten im internen Bereich zum Einsatz (vgl. Grohmann 2002, S. 51f.):

- *Anwendungsserver*: Auf dem Anwendungsserver laufen die Anwendungen, die für Kunden zur Verfügung gestellt werden sollen.
- *Applikationsserver*: Der Applikationsserver trägt dem zentralen, serverbasierten Betrieb Rechnung. Er muss Anwendungen, die nicht von sich aus ASP-fähig sind, entweder in ein webfähiges Format umwandeln oder ein entsprechendes Terminalprogramm zur Verfügung stellen.
- *Portalserver*: Der Portalserver stellt das Anwendungsportal zur Verfügung. Das Anwendungsportal bindet die verschiedenen Anwendungen in einer einheitlichen Benutzeroberfläche zusammen und ist der Einstieg in die bereitgestellten Anwendungen. Außerdem ist der Portalserver für die kundenspezifischen Anpassungen (z. B. Personalisierung) zuständig.

- *Kommunikationsserver*: Der Kommunikationsserver übernimmt z. B. den E-Mail-Verkehr, die Termin- und Aufgabenorganisation der Kunden und die Kommunikation zwischen Kunden und Application Service Provider.

- *E-Security-Server*: Der E-Security-Server ist für die interne Sicherheit zuständig. Seine Aufgaben sind der Viren- und Spam-Schutz. Außerdem können hier ein Intrusion Detection System (IDS) und weitere interne Sicherheitsmassnahmen installiert werden.

- *DMS-Server*: Zur Sicherung der Daten werden Archiv- und Dokumentenmanagementserver eingesetzt. Sie archivieren Inhalte revisionssicher und stellen geeignete Suchmöglichkeiten bereit.

- *Billingserver*: Verrechnungssysteme (vgl. Grohmann 2002, S. 74f.) müssen relevante Nutzungsdaten gegliedert nach Volumen, Zugriff und/oder Zeit erfassen, sie in ein Kalkulationsschema überführen und entsprechende Rechnungen generieren. Im ASP-Fachjargon werden diese drei Schritte Accounting, Mediation und Billing genannt. (Der Billingserver ist in der Abbildung 24 nicht enthalten.)

demilitarised zone – DMZ
Die Schnittstelle zwischen internem und externem Bereich wird im Fachjargon „entmilitarisierte Zone (demilitarised zone – DMZ)“ (Grohmann 2002, S. 53) genannt, da hier die Sicherheitsmechanismen beim Zugriff von außen greifen. Folgende Komponenten sind in der DMZ in der Regel zu finden (vgl. Grohmann 2002, S. 52f):

- *Webserver*: Der Webserver stellt den Zugang zum internen Bereich her.

- *Authentisierungsserver*: Auf dem Authentisierungsserver sind alle Zugangsberechtigungen der Nutzenden für die internen ASP-Anwendungen abgelegt. Bei jedem Zugangsversuch gleicht der Server die vom Nutzer geforderten Daten zur Authentisierung mit den hinterlegten Daten ab und vergibt daraufhin die entsprechenden Zugangsberechtigungen. Die erforderlichen Daten sind in der Regel Kennung und Passwort. Es sind aber auch andere Verfahren zur Authentisierung (Chipkarte, Stimmenerkennung, Fingerabdruck-Scanner) denkbar und vereinzelt schon im Einsatz.

- *Remote-Access-Server*: Der Remote-Access-Server ermöglicht den direkten und dedizierten Zugang zum internen Bereich. Dieser ist aber dennoch durch eine Firewall geschützt.

- *Firewall*: Zwei Firewalls stellen die Schnittstelle vom internen Bereich zur DMZ bzw. von der DMZ zum externen Bereich dar, mit Ausnahme des Remote-Access-Servers. Eine Firewall blockiert und kontrolliert verschiedene Protokolle, die auf dem TCP/IP-Protokoll aufsetzen. Durch die doppelte Sicherung in der DMZ soll Angreifern der Zugriff auf den internen Bereich erschwert werden.

Externer Bereich
Der Anwender im externen Bereich hat verschiedene Möglichkeiten auf die ASP-Infrastruktur zuzugreifen:

- *Wireless*: Neben dem Zugriff über das Festnetz, kann auch drahtlos auf die ASP-Infrastruktur zugegriffen werden. „Dem so genannten Wireless ASP (WASP) werden große Zukunftschancen eingeräumt, immerhin soll WASP dazu beitragen, die horrenden Kosten für die UMTS-Lizenzen über mobile ASP-Lösungen zu amortisieren“ (Grohmann 2002, S. 53).

- *Internet*: Das Internet, als einfachster Weg über das Festnetz, ermöglicht den Zugriff auf die ASP-Anwendungen von jedem Ort und jeder Zeit über verschiedenste Zugangsgeräte, solange sie internetfähig sind.

- *Extranet*: Wird der Internet-Zugriff über das Festnetz auf bestimmte Kunden begrenzt, spricht man von einem Extranet. Der Zugriff auf die ASP-Anwendungen erfolgt dann über den Remote-Access-Server.

- *VPN*: Zur Sicherung der Übertragung bzw. Verbindung des Kunden mit den ASP-Anwendungen können Virtual Private Networks (VPN) eingesetzt werden. Zwischen Anwender und DMZ wird ein VPN-Tunnel eingerichtet, der die Übertragung verschlüsselt. An den Enden des VPN-Tunnels müssen entsprechende VPN-Gateways zur Ver- und Entschlüsselung eingerichtet sein.

4.6 Zwischenergebnis

In Hinblick auf die aus den Fallstudien entwickelten Perspektiven Technik, Aufgaben, Organisation und Vorgehen (Abschnitt 3.4) bietet ASP bis auf die Technikperspektive wenig.

Zur Aufgabenperspektive bietet ASP Wertschöpfungsketten, die unterschiedlich detailliert Aufgabenfelder zeitlich separiert anordnen. Welche Verbindungen zwischen den einzelnen Wertschöpfungsketten existieren, welche Aufgabengranularität vorherrscht und welche Aufgaben warum zu Aufgaben einer höheren Ebene zusammengefasst werden, bleibt unklar. Darüber hinaus ist keine explizite und umfassende Beschreibung von Aufgaben im ASP enthalten. Sie tauchen implizit in den schon angesprochenen Wertschöpfungsketten oder in Servicebeschreibungen auf, in denen sie, im Sinn von Dienstleistungsstufen, zu Aufgabenbündeln zusammengeschnürt werden.

Für die Organisationsperspektive bietet ASP erste Ansätze durch das ASP-Player Modell, das Schichtenmodell und durch die kombinierte Wertschöpfungskette, die Aufgabenfelder mit Akteuren über Kompetenzen verknüpft. Welche Rollen gibt es außer dem Application Service Provider und seinen Partner noch? Warum wird im ASP nicht auf die Rolle des Nutzers und nur wenig auf die Rolle des Kunden eingegangen? Warum fehlt die Rolle des Kunden im ASP-Player Modell? Darüber hinaus erscheint die Zuordnung von Aufgaben zu Akteuren über Kernkompetenzen nach Gillian u. a. (1999) (Abschnitt 4.1.1) zu vage. Interessant ist, welcher Akteur in welcher Rolle welche Aufgaben übernimmt. Im ASP fehlt eine Aufgabensystematik. Außerdem ist im ASP keine Auseinandersetzung mit dem Akteursbegriff und den (teilweise nichtharmonischen) Beziehung zwischen Akteuren zu finden. Hinsichtlich der Beziehungen werden im ASP nur Vertragsarten und SLAs diskutiert, die die Beziehungen zwischen den Akteuren als Dienstleistungsbeziehungen charakterisieren, ohne dass sich die Autoren in der ASP-Literatur bisher mit dem Dienstleistungsbegriff auseinandergesetzt haben. Hinsichtlich der Kosten werden nur anrechenbare Leistungen betrachtet, Transaktionskosten werden übersehen.

Zu einer Vorgangsperspektive bzw. zu einem Vorgehen bietet die ASP-Literatur die schon erwähnten Wertschöpfungsketten und sehr ausdifferenzierte SLAs an, die alle Eventualitäten abdecken sollen, so dass „kein Vorgehen nötig wird“. Wie sich Veränderungen auf die Bereitstellung auswir-

ken, wie mit diesen Einflüssen im Prozess konstruktiv umgegangen werden kann, welche Akteure in welchem Umfang Gestaltungsmöglichkeiten in der Bereitstellung haben und wie eine Softwarebereitstellung grundsätzlich gestaltet werden kann, wird nicht thematisiert.

Zusätzlich zur Kritik, hinsichtlich der Perspektiven, muss die Verankerung von ASP als eine spezielle Form des IT-Outsourcings aus Sicht der Fallstudien kritisiert werden. In der Fallstudie CommSy@WissPro wurde CommSy nicht nur extern, sondern auch intern, d. h. im Fachbereich Informatik für Fachbereichsmitglieder, bereitgestellt. Durch die Fixierung auf das Outsourcing ist ASP nicht mehr auf die Fallstudie anwendbar. Wird allerdings unter ASP die Bereitstellung einer Anwendung als Dienstleistung verstanden, so wird die Fixierung auf das Outsourcing aufgehoben (vgl. Picot und Jahn 2000; Riemer und Ahlemann 2001; Rosenhagen 2002). In diesem Sinne muss der im Folgenden aus ASP zu gestaltende Ansatz eASP zur Softwarebereitstellung in Hinblick auf die vier Perspektiven Aufgaben, Organisation, Technik und Vorgehen insbesondere darauf achten, dass die Perspektiven eine interne Bereitstellung nicht ausschließen.

- 5 -
Informatiksysteme in Organisationen

Im folgenden Kapitel wird das Teilgebiet der Informatik „Informatiksysteme in Organisationen“ vorgestellt, um das Geschäftsmodell ASP später methodisch-theoretisch zum Bereitstellungsansatz eASP ergänzen und fundieren zu können. Darüber hinaus wird eASP am Ende dieser Arbeit in dieses Gebiet eingegliedert, indem es in die folgenden und später näher beschriebenen zwei Methodenrahmen aus dem Gebiet „Informatiksysteme in Organisationen“ integriert wird:

- Die *IT-unterstützte Organisationsgestaltung* nach Rolf (1998) legt den Schwerpunkt auf die von Software beeinflusste Organisationsgestaltung. Dieser Ansatz spannt in einer perspektivischen Verknüpfung eine Matrix auf, in der sich Technik- und Organisationsoptionen in unterschiedlich granularen Ebenen anordnen lassen.
- Der *Methodenrahmen der Softwareentwicklung STEPS* nach Floyd (1986; 1991b) begreift den Softwareentwicklungsprozess als Design- und kooperativen Lernprozess aller Beteiligten. STEPS beruht auf einem zyklischen Projektmodell und wird durch Methoden konkretisiert, die auf der grundlegenden evolutionären Sichtweise aufbauen.

Wie sich der erarbeitete Ansatz eASP in die beiden Methodenrahmen einbinden lässt, wird im dritten Teil dieser Arbeit (Kapitel 10) behandelt.

Zur Überwindung der im vorigen Kapitel in den Perspektiven Aufgaben, Organisation, Technik und Vorgehen (Abschnitt 4.6) aufgedeckten Schwächen und zur methodisch-theoretischen Fundierung von ASP werden in diesem Kapitel folgende Methoden und Modelle herangezogen (in Klammern sind die Perspektiven angegeben, in denen diese Ansätze später Verwendung finden):

- Aufgabenbezogene Anforderungsanalyse (Aufgaben – Kapitel 6)
- Akteursmodell (Organisation – Kapitel 7)
- Netzwerkorganisation (Organisation – Kapitel 7)

- Serviceflow-Management/Serviceflows (Organisation – Kapitel 7)
- Transaktionskostentheorie (Organisation – Kapitel 7)
- Zyklisches Projektmodell von STEPS (Vorgehen – Kapitel 9)

Während in den Perspektiven Aufgaben und Vorgehen später jeweils ein Ansatz auf ASP übertragen werden kann, müssen in der Organisationsperspektive verschiedene Ansätze einbezogen werden. Dies ist notwendig, um u. a. den zur Beschreibung der Organisation einer Softwarebereitstellung geeigneten Ansatz des kollektiven Rollennetzes entwickeln zu können. In der Technikperspektive (Kapitel 8) bietet ASP bereits Umfassendes, und es werden später nur kleine Ergänzungen und Veränderungen daran vorgenommen, so dass in diesem Kapitel kein Ansatz hierfür herangezogen werden muss.

Die Auswahl der Ansätze und Methodenrahmen gründet sich auf die gezielte Ergänzung und Fundierung von ASP in den vier Perspektiven im zweiten Teil und der Erweiterung der beiden Methodenrahmen im dritten Teil dieser Arbeit. Darüber hinaus entspricht diese Auswahl dem am Fachbereich Informatik der Universität Hamburg entwickelten Verständnis von „Informatiksystemen in Organisationen“, welches sich u. a. in der Manifestation des Studienprofils ISO ausdrückt. In Anlehnung an den Schwerpunkt „Organisationsbezogene Softwareentwicklung (OSE)“ des Studienprofils strukturiert dieses Kapitel die skizzierten Ansätze und Methodenrahmen in zwei Abschnitte:

- Der Abschnitt *softwarebezogene Organisationsentwicklung* enthält die Ansätze, die auf die Organisationsgestaltung fokussieren.
- Der Abschnitt *organisationsbezogene Softwareentwicklung* hält die Ansätze bereit, die verstärkt die Softwareentwicklung in den Blick nehmen.

Diese Trennung ist im Studienprofil ISO und dem Schwerpunkt OSE nicht vorgesehen. Vielmehr sind alle im Folgenden dargestellten Ansätze dem Schwerpunkt OSE zuzuordnen. Dennoch hilft die sicherlich nicht trennscharfe Unterscheidung, die verschiedenen Ansätze besser voneinander abzugrenzen.

5.1 Softwarebezogene Organisationsentwicklung

Der folgende Abschnitt präsentiert die Ansätze, die primär auf die Organisationsgestaltung bei Informatiksystemen in Organisationen fokussieren. Hierzu gehört die IT-unterstützte Organisationsgestaltung, das Akteursmodell, die Netzwerkorganisation, die Transaktionskostentheorie und der Ansatz des Serviceflow-Management als ein Anwendungsmodell von Informatiksystemen in Organisationen.

5.1.1 IT-unterstützte Organisationsgestaltung

Rolf (1998) schlägt als Leitbild für die Organisations- und Wirtschaftsinformatik die IT-unterstützte Organisationsgestaltung vor. Dieses Leitbild kehrt das Verhältnis von Softwareentwicklung und Organisationsgestaltung um, welches in der Softwaretechnik primär auf der Softwareentwicklung liegt, auch wenn zyklische Softwareentwicklungsmethoden die Anpassung des Einsatzkontextes explizit in den Softwareentwicklungszyklus einbeziehen (vgl. Wulf und Rohde 1995; Floyd u. a. 1997).

Die IT-unterstützte Organisationsgestaltung geht davon aus, dass bei der Modellierung von Informationssystemen die Organisation einer Institution (oder Teile von Ihr) verändert wird. Die veränderten organisatorischen (und sozialen) Phänomene eröffnen neue Möglichkeiten zu informationstechnischen Innovationen. So stehen organisatorische Phänomene einerseits und informationstechnische Potentiale andererseits in einem Wechselverhältnis. Diese organisatorische bzw. technische Sichtweise kann in Perspektiven unterteilt werden: in die Perspektive auf den Arbeitsplatz, auf die Arbeitsgruppe und auf die unternehmensweite Organisation (Rolf 1998, S. 147f.).

Die Verknüpfung dieser Perspektiven kann Top-Down oder Bottom-Up erfolgen. Die Top-down-Sicht geht bei der Modellierung eines Informationssystems von der Perspektive der unternehmensweiten Organisation aus, die auf Arbeitsgruppen und Arbeitsplätze „herunter gebrochen“ werden kann. Die Bottom-up-Sicht modelliert ein Informationssystem aus der Perspektive des Arbeitsplatzes heraus. Sie impliziert, dass sich die Modellierung zu einer Unterstützung von Arbeitsgruppen und der unternehmensweiten Organisation zusammenfügt. Jede Perspektive braucht die größere Einheit als Gesamtzusammenhang und die kleinere Einheit als innere Struktur (Rolf 1998, S. 150). So besteht auf der einen Seite eine Verknüp-

fung zwischen den Perspektiven und auf der anderen Seite kommt es zu Wechselwirkungen zwischen organisatorischer und technischer Sichtweise (Rolf 1998, S. 161).

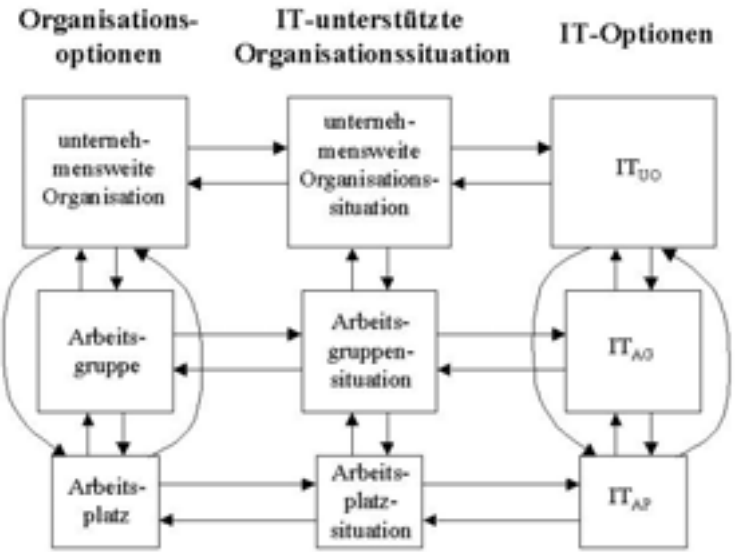

Abbildung 25: Perspektivische Verknüpfung

Horizontal ist die Gliederung in die Perspektiven Arbeitsplatz, Arbeitsgruppe und unternehmensweite Organisation dargestellt. Vertikal stehen links die organisatorische und rechts die technische Sichtweise. Vertikal in der Mitte befindet sich die Verknüpfung der organisatorischen und technischen Sichtweise zu einem Informationssystem. Die Verknüpfung führt zu einer IT-unterstützten Organisationssituation.

Die IT-unterstützte Organisationsgestaltung ist weder allein aus der Top-down-Sicht noch aus der Bottom-up-Sicht erfolgreich zu leisten (Rolf 1998, S. 155). Es geht vielmehr um die Verknüpfung der organisationsweiten Perspektive mit den Perspektiven auf die Arbeitsgruppe und den Arbeitsplatz, wobei das Zentrum des Gestaltungsprozesses die Arbeitsgruppe darstellt. „Von hier aus kann die perspektivische Verknüpfung sowohl mit der unter-

nehmensweiten Organisation als auch mit dem Einzelarbeitsplatz gelingen. Auf diese Weise kann die Verbindung zur Organisations- und Arbeitsgestaltung sichergestellt werden. Die Mitarbeiter der Arbeitsgruppe und IT-Organisationsexperten erarbeiten gemeinsam in Organisationsworkshops und mit Modellierungswerkzeugen die Aufgaben und Kooperationsbeziehungen der Arbeitsgruppen sowie ihre Einbindung in die Kernprozesse der Organisation“ (Rolf 1998, S. 159).

Inwiefern der im zweiten Teil erarbeitete Softwarebereitstellungsansatz eASP die IT-unterstützte Organisationsgestaltung ergänzen und erweitern kann, wird im dritten Teil dieser Arbeit dargestellt (Abschnitt 10.2.2).

5.1.2 Akteursmodell

Durch Touraine (1984) etablierten sich Akteure als Modellansatz in soziotechnischen Wissenschaftsbereichen. Akteure sind „weder voluntaristisch handelnde Individuen, noch ein seine historische Mission erfüllendes Subjekt“ sondern „kollektive Handlungseinheiten, die gleichsam unterhalb der Ebene gesellschaftlicher Strukturen und oberhalb einzelner Handlungen konzeptionell anzusiedeln sind“ (Rammert 1993, S. 100). „Sie entstehen, indem Mitglieder mit gleichen Werten oder Interessen ihr Handeln koordinieren und Allianzen oder Konkurrenzen zu anderen Akteuren aufbauen“ (Rolf 1998, S. 19). Dabei zeichnen sie sich durch eine Handlungsfähigkeit aus, die durch ihre Mitglieder bestimmt wird. „Kollektive Handlungseinheiten können zwar nur durch ihre einzelnen Mitglieder handeln; diese Handlungen werden jedoch aufgrund ihrer Organisiertheit einem kollektiven Akteur zugerechnet“ (Rammert 1993, S. 100f.). Akteure haben u. a. folgende Merkmale (vgl. Rammert 1993; Rolf 1998):

- Akteure nehmen Bezug auf einen gemeinsamen kulturellen Hintergrund und formulieren daraus strategische Orientierungen.
- Akteure haben erkennbare Abgrenzungen und Beziehungen zu anderen Akteuren.
- Zwischen Akteuren laufen eine Vielzahl von harmonischen und nicht harmonischen Interaktionen ab, die zu Korrekturen, Ermutigungen oder neuen Allianzen führen. Es finden permanent Rückkopplungen und Wechselwirkungen statt.

Insbesondere der dritte Punkt beschreibt das potentiell zwischen Akteuren vorhandene Konfliktpotential. „Die Beziehungen zwischen den Akteuren sowohl hinsichtlich ihres Konfliktpotentials als auch hinsichtlich ihrer kulturellen Orientierung bestimmen ihr Handeln“ (Rammert 1993, S. 100).

Zu Akteuren zählen „formale Organisationen, wie Unternehmen, Verbände, Parteien, Behörden und Forschungsinstitutionen, als auch durch kulturelle Orientierungsmodelle verbundene soziale Ensembles, wie Gruppen, informelle Netzwerke und soziale Bewegungen“ (Rammert 1993, S. 101).

Mit dem Akteursmodell (vgl. Rammert 1993; Rolf 1998) sind die Voraussetzungen vorhanden, die zahlreichen passiven und aktiven Akteure in einer Organisation mit ihren Orientierungen, Leitbildern und Machtspielen in den Gestaltungsprozess der IT-unterstützten Organisationsgestaltung einzubeziehen (Rolf 1998, S. 154). Dies ist kein kleiner Anspruch, da zwischen den Akteuren eine Vielzahl von nicht harmonischen Interaktionen ablaufen können (Rolf 1998, S. 19). Der Konflikt zwischen den Akteuren erhält durch den Einsatz von Informationssystemen noch zusätzliche Brisanz, da „die Einführung oder Veränderung von Technik der ideale Nährboden ist, um in Organisationen diese Machtspiele spielen zu können“ (Rolf 1998, S. 22). So ist es notwendig, alle Akteure in den Gestaltungsprozess einzubeziehen, denn nur unter der Berücksichtigung aller Akteure kann eine IT-unterstützte Organisationsgestaltung gelingen (Rolf 1998).

Durch das Akteursmodell können in der Organisationsperspektive alle an der Softwarebereitstellung Betroffenen erkannt und beteiligt werden (Kapitel 7).

5.1.3 Netzwerkorganisation

Netzwerke sind eine sehr alte Form sozialer Organisation und gelten als die flexibelsten und anpassungsfähigsten Formen der Organisation. Sie sind fähig, sich mit ihrer Umwelt und mit den Knoten, aus denen sich das Netzwerk zusammensetzt, zu entwickeln. Andererseits haben sie beträchtliche Schwierigkeiten, Funktionen zu koordinieren, Ressourcen für bestimmte Ziele zu bündeln und die Komplexität einer Aufgabe ab einer bestimmten Größe zu bewältigen (Castells 2001). Durch das heutige technische Vernetzungsniveau können Netzwerke ihre Schwächen erstmals überwinden und ihre Stärken insgesamt ausspielen. Traditionelle Hierarchien werden obso-

let (Rolf 2004a). So sind für Castells (2003, S. 432f.) informatikgestützte Netzwerke heutzutage die überlegene Morphologie für Organisationen.

Ein Netzwerk (vgl. Castells 2001; 2003; Rolf 2004a) bezeichnet eine Reihe miteinander flexibel verknüpfter Knoten, wobei Individuen, Gruppen oder Unternehmen Knoten sein können. Besteht in einem Netzwerk ein spezifischer Bedarf, so wird ein neuer Knoten initiiert, von außen ins Netzwerk eingebunden oder Verknüpfungen zu anderen Netzwerken etabliert. Verliert ein Knoten seine Relevanz, wird er aus dem Netzwerk herausgelöst, und das Netzwerk reorganisiert sich. So dezentralisiert ein Netzwerk Leistungskompetenzen, eröffnet Partizipation und bewahrt Flexibilität. Netzwerke sind hochgradig dynamische, offene Systeme, die jederzeit erneuert werden können, ohne dass das Gleichgewicht in Gefahr gerät (Rolf 2004a).

An dieser „enthusiastischen“ Sichtweise auf Netzwerke wird kritisiert, dass die Bezeichnungen von Netzwerkorganisationen und -gesellschaften irreführend ist, da dieses Bild eine gleichberechtigte und weitgehend autonome Zusammenarbeit einer Gruppe von Akteuren suggeriere (vgl. Rolf 2004a; Sennett 1999; Sydow 1999). Tatsächlich gehe die Macht von einem Zentrum aus, das den scheinbar autonomen Mitgliedern des Netzwerkes die Ziele diktiere. Rolf (2004a) sieht in den Herrschern der Netzwerke multinationale Unternehmen, die in der Lage sind, den Netzwerkakteuren ihre Logik aufzuzwingen. „Sie dirigieren das Netzwerk über für sie transparente Softwaresysteme“ (Rolf 2004a, S. 24).

Für die Softwarebereitstellung werden u. a. durch die Netzwerkorganisation, in Kombination mit dem Akteursmodell, Vernetzungsmöglichkeiten und flexible Kooperationsbeziehungen der beteiligten Akteure in der Organisationsperspektive sichtbar (Kapitel 7).

5.1.4 Transaktionskostentheorie

Die Transaktionskostentheorie befasst sich mit der Organisation von Tauschbeziehungen. Die grundlegende Untersuchungseinheit stellt die einzelne Transaktion dar, die als Übertragung von Verfügbarkeitsrechten definiert wird (Picot u. a. 1998, S. 41). Betrachtet werden dabei nicht der physische Austausch oder die Erbringung von Dienstleistungen. Untersucht wird der mit einer arbeitsteiligen Aufgabenerfüllung verbundene Koordinationsaufwand – die Transaktionskosten. Transaktionskosten sind die anfallenden

Kosten der Information und Kommunikation für Anbahnung, Vereinbarung, Abwicklung, Kontrolle und Anpassung eines als fair empfundenen Leistungsaustausches. Die Höhe dieser Transaktionskosten hängt von bestimmten Eigenschaften der zu erbringenden Leistungen, von Verhaltensmerkmalen der ökonomischen Akteure und von der gewählten Einbindungs- bzw. Organisationsform ab (vgl. Picot u. a. 1998; Weiß und Längsfeld 2000).

„Unternehmungen als integrierte, in sich arbeitsteilige Gebilde haben nur dann ein Existenzrecht, wenn sie in ihrem Binnenbereich die mit jeder arbeitsteiligen Leistungserstellung verbundenen Koordinationsprobleme besser lösen können, als dies bei einer Abwicklung mit externen Partnern über den Markt der Fall wäre. Transaktionskosten sind damit der Effizienzmaßstab zur Beurteilung und Auswahl unterschiedlicher institutionelle Arrangements“ (Picot u. a. 1998, S. 41).

Zur Auswahl geeigneter Koordinationsformen wurde von Williamson (1991) ein Transaktionskostenmodell entwickelt, das als „organizational failure framework“ weite Verbreitung gefunden hat. In diesem Modell werden folgende Einflussgrößen auf die Transaktionskosten benannt (vgl. Picot u. a. 1998, S. 43):

- *Spezifität*: Der Spezifitätsgrad einer Transaktion ist umso höher, je größer der Wertverlust ist, der entsteht, wenn die zur Aufgabenerfüllung erforderlichen Ressourcen nicht eingesetzt werden. Dabei kann sich die Spezifität auf das erforderliche Know-how, auf die zu tätigenden Investitionen, auf Sicherungsbedürfnisse und/oder auf anderweitige Verfahrensbesonderheiten beziehen. Die Spezifität wird übereinstimmend als Haupteinflussgröße bezeichnet.

- *Opportunismus*: Opportunistisches Verhalten liegt vor, wenn sich Beteiligte nicht ausschließlich in verständigungsorientierter Weise verhalten, sondern vielmehr strategisch handeln, indem sie versuchen, ihre eigenen Interessen auch zum Nachteil anderer und unter Missachtung sozialer Normen zu verwirklichen. Spezifität wird erst in Verbindung mit Opportunismus zu einem Problem.

- *Unsicherheit*: Die Unsicherheit drückt sich in der Anzahl und dem Ausmaß nicht vorhersehbarer Aufgabenänderungen aus. In einer unsicheren Umwelt wird die Vertragserfüllung durch häufige Änderungen von Terminen, Preisen, Konditionen und Mengen erschwert und

verkompliziert. Dies hat häufige Vertragsmodifikationen und damit die Inkaufnahme erhöhter Transaktionskosten zur Folge.

- *Beschränkte Rationalität*: Der Mensch kann nicht nur rational handeln, da seine Informationsverarbeitungskapazität eingeschränkt ist und die Komplexität aufgrund des Auftretens kommunikativer Probleme bei schwer übermittelbarem Wissen weiter steigt. Unsicherheit wird in Verbindung mit der beschränkten Rationalität zum Problem.
- *Informationsverkeilung*: Als Informationsverkeilung werden in erster Linie Situationen asymmetrischer Informationsverteilung bezeichnet, bei denen die Gefahr besteht, dass ein Transaktionspartner seinen Informationsvorsprung opportunistisch ausnutzt.

Die folgende Abbildung verdeutlicht die Beziehung der Einflussgrößen untereinander (Picot u. a. 1998):

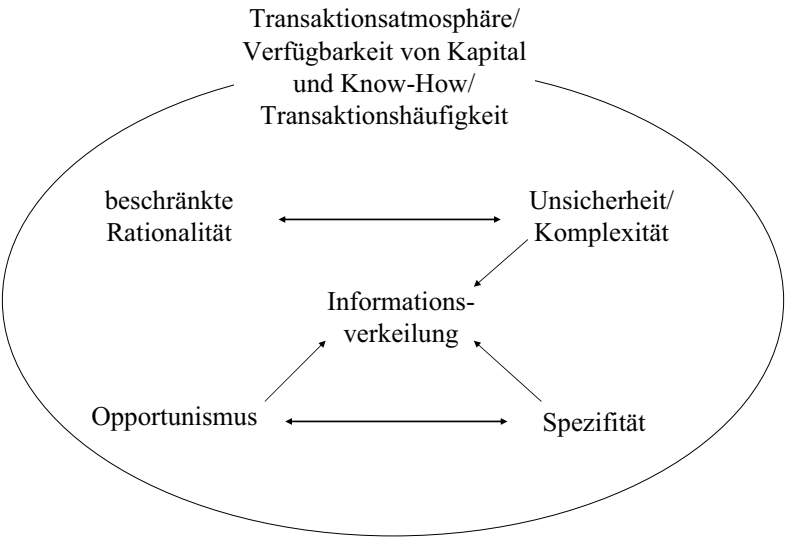

Abbildung 26: Einflussgrößen auf die Transaktionskosten

Neben diesen Einflussfaktoren müssen drei weitere Einflussgrößen berücksichtigt werden, die nachrangig sind, aber dennoch einen nicht unwesentlichen Einfluss haben (Picot u. a. 1998, S. 44):

- *Transaktionshäufigkeit*: Die Transaktionshäufigkeit bestimmt die Amortisationszeit und damit die ökonomische Vorteilhaftigkeit hierarchischer Unternehmensstrukturen oder langfristiger Kooperationsbeziehungen im Gegensatz zu Marktmodellen.

- *Transaktionsatmosphäre*: Die Transaktionsatmosphäre umfasst alle für die Koordination einer Leistungsbeziehung relevanten sozialen, rechtlichen und technologischen Rahmenbedingungen. Gegenseitiges Vertrauen und ähnliche Werthaltungen der Transaktionspartner verringern die Wahrscheinlichkeit opportunistischen Verhaltens und machen damit transaktionskostenintensive Schutzklauseln überflüssig. Moderne Informations- und Kommunikationssysteme können die Möglichkeiten rationalen Verhaltens erweitern, den Spezifitätsgrad einer Transaktion verändern und generell Kosten der Informationsübertragung senken. Vertrauen und effiziente Informations- und Kommunikationssysteme begünstigen damit marktwirtschaftliche oder kooperative Formen der Aufgabenerfüllung.

- *Verfügbarkeit von Know-how und Kapital*: Wenn ein Unternehmen das für die Eigenerstellung spezifischer und unsicherer Leistungen erforderliche Kapital und Know-how nicht besitzt, kann es die entsprechenden Leistungen nicht selbst erstellen. In diesem Fall muss es, unter Inkaufnahme höherer Transaktionskosten, langfristig angelegte Kooperationsbeziehungen mit externen Partnern eingehen.

Die Auswahl der jeweiligen effizienten Koordinationsform variiert je nach dem Spezifitätsgrad der jeweils zu erstellenden Leistung. Für Leistungen geringer Spezifität sind Märkte, für Leistungen hoher Spezifität jedoch eher hierarchische Koordinationsformen transaktionskostenminimierend. Zwischen den Extremformen Markt und Hierarchie gibt es ein vielfältiges Spektrum an Zwischenformen. Picot u. a. (1998, S. 45) zählen zu den Mischformen langfristig angelegte Unternehmenskooperationen, strategische Allianzen, Joint Ventures, Franchisingsysteme, Lizenzvergabe an Dritte, dynamische Netzwerke sowie langfristige Abnahme- und Belieferungsverträge. „Die scheinbar einfache Wahl zwischen unternehmensinterner und

unternehmensexterner Erstellung entpuppt sich damit als komplexe Optimierungsaufgabe innerhalb eines breiten Kontinuums von Möglichkeiten“ (Picot u. a. 1998, S. 45).

Zur Begründung von Entstehungen von Unternehmen ist die Transaktionskostentheorie einsetzbar, ebenso beim Verwischen von Unternehmensgrenzen. Darüber hinaus kann sie wertvolle Gestaltungsempfehlungen zu Fragen der internen Organisationsgestaltung oder der räumlich dezentralen, betriebsübergreifenden Aufgabenabwicklung geben (vgl. Picot u. a. 1998, S. 46f.). Dabei gestaltet sich die Vorgehensweise bei einer Transaktionskostenanalyse wie folgt: „Nachdem Spezifitäts- und Unsicherheitsgrad einer Austauschbeziehung ermittelt wurden, ist vor dem Hintergrund der jeweiligen Transaktionsatmosphäre unter Berücksichtigung der zu erwartenden Transaktionshäufigkeit und angesichts der Verfügbarkeit von Know-how und Kapital diejenige Koordinationsform auszuwählen, die die vergleichsweise geringsten Transaktionskosten verursacht“ (Picot u. a. 1998, S. 44).

Für die Softwarebereitstellung ist u. a. interessant, dass durch die Transaktionskostentheorie in Zusammenhang mit der Netzwerkorganisation in der Organisationsperspektive deutlich wird, dass das Finden, Eingehen und Einhalten der Kooperationsbeziehungen Aufwand bedeutet und Kosten verursacht (Kapitel 7). Darüber hinaus müssen die Kosten für die Bereitstellung einer Software bestimmbar sein, um Entscheidungen hinsichtlich der Frage *Insourcing oder Outsourcing* fällen zu können. „Wenn man die eigenen Kosten nicht kennt, ist auch schwer festzustellen, ob eine Auslagerung rentabel ist“ (Endres 2004, S. 548).

5.1.5 Serviceflow-Management/Serviceflows

Als Anwendungsmodell von Informatiksystemen in Organisationen stellt das Serviceflow-Management die Organisation von Serviceprozessen anhand von Serviceflows in und zwischen Organisationen dar.

Die Grundlage eines Serviceflows ist ein zeitlich beschränkter, aus Teilleistungen bestehender Serviceprozess mit definiertem Anfang und vorab erkennbarem Ende. Ein Servicenehmer wird von einem oder verschiedenen Servicegebern durch den Prozess zur Bedürfnisbefriedigung hingeführt. Die Bedürfnisbefriedigung wird am Ende des Prozesses erreicht, d. h. der Serviceflow ist beendet, sobald die Bedürfnisbefriedigung einsetzt (vgl. Klischewski und Wetzel 2000).

Das Serviceflow-Management (vgl. Kaschek u. a. 2001; Klischewski und Wetzel 2001a; 2002b; Wetzel und Klischewski 2002) unterstützt die Ausführung von Serviceprozessen in heterogenen Netzwerken von Servicegebern (Dienstleistungsnetzwerke). Es zielt primär auf die Unterstützung von Mitarbeitern der Servicegeber und ihren Organisationen. Der Serviceprozess wird als eine Folge von Servicepunkten modelliert, an denen Servicegeber und Servicenehmer aufeinander treffen. In den Servicepunkten werden die aktuelle Situation und das Anliegen des Servicegebers neu bewertet, wobei die Servicepunkte jeweils durch die dort auszuführenden Aktivitäten und ihre jeweiligen Vor- und Nachbedingungen beschrieben werden. Bei der Durchführung von Serviceleistungen werden die beteiligten Servicegeber und ggf. der Servicenehmer über den bisherigen Verlauf und die aktuelle Planung des sich entfaltenden Serviceprozesses informiert. Serviceprozesse basieren auf Prozessmustern, die Standardabläufe beschreiben, aber im Prozess situationsabhängig angepasst werden können.

Zur technischen Unterstützung bzw. Umsetzung von Serviceflow und Serviceflow-Management siehe Klischewski und Lenk (2002), Klischewski und Wetzel (2001a;c;b; 2002a;b) und Wetzel und Klischewski (2002). Beispiele von Serviceleistungen und deren Umsetzungen als Serviceflows sind u. a. Aufenthalt eines Patienten im Krankenhaus (Klischewski und Wetzel 2001b; 2003; Klischewski u. a. 2001) und umfassende Serviceleistungen einer Stadtverwaltung (Klischewski und Wetzel 2001c;b; 2002a).

Bei der Bereitstellung von Software sollte die Bedürfnisbefriedigung für einen Nutzer jedes Mal gegeben sein, wenn er die angebotene Dienstleistung in Anspruch nimmt. d. h. die Bereitstellung von Software ist eine kontinuierliche Dienstleistung, die permanent angeboten werden muss und nicht erst anlaufen sollte, wenn ein Nutzer auf sie zugreift. Außerdem wird der Nutzer nicht durch einen Prozess begleitet, sondern er wendet lediglich die Software an, ohne mit anderen an der Softwarebereitstellung beteiligten Akteuren in Kontakt zu treten. Aus diesen Gründen kann die Softwarebereitstellung nicht als Serviceflow interpretiert werden.

Interessant für den Softwarebereitstellungsansatz am Konzept der Serviceflows und des Serviceflow-Management ist die Auseinandersetzung mit dem Dienstleistungsbegriff (vgl. Berekoven 1986; Ertel 1986), die im ASP fehlt.

„Services sind soziale Beziehungen“ (Klischewski 2000; Kaschek u. a. 2001). Mit „soziale[n] Beziehungen“ sind Beziehungen zwischen sozialen

Akteuren gemeint (vgl. Kaschek u. a. 2001, S. 16). Der Begriff Service beinhaltet das Erkennen und Befriedigen von menschlichen Bedürfnissen oder auch von kollektivem Bedarf. Bedürfnisse sind hierbei nicht abstrakt, sondern subjektiv und situativ an Orte und Zeiten gebunden. Es gibt ein Individuum oder Kollektiv in der Rolle des Bedürftigen und ein Individuum oder Kollektiv in der Rolle des Dienstleisters, welcher das Bedürfnis befriedigen kann. Eine Dienstleistung gilt als erfolgreich, wenn der Bedürftige zufrieden gestellt und bereit ist, dem Dienstleister einen z. B. monetären Gegenwert zu leisten. Letztlich kann jedoch nur der Servicenehmer einschätzen, ob die angestrebte Befriedigung stattgefunden hat (vgl. Klischewski 2000, S. 19). „Objektiv feststellen [. . .] lassen sich höchstens die mit dem Service verbundenen Verrichtungen und deren Vergleich mit entsprechenden Soll-Größen. Grundlage der Serviceleistung ist daher in der Regel eine mündliche oder schriftliche Vereinbarung zwischen Dienstleister und Kunde, die Art und Umfang der Servicebeziehung (inklusive Gegenleistung Kunde) vorher beschreibt“ (Klischewski 2000, S. 19f.).

5.2 Organisationsbezogene Softwareentwicklung

In diesem Anschnitt werden Ansätze präsentiert, die im Gebiet „Informatiksysteme in Organisationen“ primär auf die Softwareentwicklung fokussieren. Dazu gehört der Methodenrahmen STEPS, das zyklische Projektmodell von STEPS und die aufgabenbezogenen Anforderungsermittlung in der Softwareentwicklung. Weiterhin wird das kollektive Rollennetz vorgestellt, das sich hauptsächlich auf die aufgabenbezogene Anforderungsermittlung gründet.

5.2.1 Methodenrahmen der Softwareentwicklung STEPS

STEPS[1] (Floyd 1986; 1991b; Floyd u. a. 1989b; 1997) ist ein methodischer Ansatz zur Softwareentwicklung und betrachtet die Softwareentwicklung als Design (Entwurf und Gestaltung), konstituiert durch ineinander verschränkte, wechselseitige Lernprozesse aller Beteiligten bei der Herstellung und dem Einsatz von Software. Die wichtigsten Elemente von STEPS sind (Floyd 1986; 1991b):

1 Softwaretechnik für evolutionäre, partizipative Systementwicklung

- Eine prozessorientierte Sichtweise auf die Softwareentwicklung,
- ein zyklisches Projektmodell anhand von Systemversionen,
- ein menschenzentrierter Qualitätsbegriff, orientiert am Einsatz von Software,
- ein abgestimmtes Repertoire an Methoden für die Aufgaben der Softwareentwicklung,
- gestaltbildende Projekttechniken für die kooperative Systementwicklung.

„Kennzeichnend für STEPS ist die Betrachtung von Software im Einsatzkontext, wobei die Einbettung in die unterstützten Arbeits- und Kommunikationsprozesse im Vordergrund stehen. Da von einem Zusammenhang von Softwareeinführung und organisatorischer Veränderung ausgegangen wird, wird Softwareentwicklung als integrativer Teil einer übergreifenden Organisationsentwicklung gesehen. [...] Als übergreifendes Leitbild dient die Unterstützungssicht der Softwareentwicklung, bei der die Gebrauchstauglichkeit von Software im Kontext qualifizierter Arbeit maßgeblich ist.

STEPS beruht auf einem zyklischen Projektmodell, das die Aufgaben der Entwickler und Anwender sowie ihre Kooperation verdeutlicht und Partizipation begünstigt. Jeder Entwicklungszyklus dient der Bereitstellung einer Systemversion (z. B. Ausbaustufe), die in der Organisation eingesetzt wird und für die eine entsprechende ‚Umfeldvorbereitung' in Form von Organisationsmaßnahmen und organisatorischen Anpassungen zu leisten ist. Innerhalb eines jeden Zyklus kommen die verschiedenen Formen des Prototyping zum Tragen" (Floyd u. a. 1997, S. 15f.).

„Mit STEPS ist auch ein eigenes Methodenverständnis verbunden, das nicht von festen Methoden ausgeht, sondern von Prozessen der Methodenentwicklung, -anpassung und -weiterentwicklung vor dem Hintergrund einer methodischen Tradition. Daher wird STEPS auch als Methodenrahmen gekennzeichnet, der durch verschiedene Methoden ausgefüllt werden kann, sofern diese der grundlegenden evolutionären Sichtweise angepaßt werden (das bedeutet u. a.: Unterstützung von Perspektivität, keine vorgegebene Reihenfolge der Arbeitsschritte, Orientierung auf inkrementelles Arbeiten)" (Floyd u. a. 1997, S. 16).

Inwiefern der im zweiten Teil erarbeitete Softwarebereitstellungsansatz eASP den Methodenrahmen STEPS ergänzen und erweitern kann, wird im dritten Teil dieser Arbeit dargestellt (Abschnitt 10.2.1).

5.2.2 Zyklisches Projektmodell von STEPS

In zyklischen Modellen „wird Software nicht mehr als ein Produkt, sondern als eine Folge von Versionen verstanden. Die Softwareentwicklung besteht dann aus einer Folge von Zyklen, in denen aufbauend auf die letzte Version eine neue Version hergestellt und eingesetzt wird“ (vgl. Floyd und Züllighoven 2002, S. 777). Folgende Abbildung stellt das zyklische Projektmodell im Methodenrahmen STEPS dar (Floyd u. a. 1997):

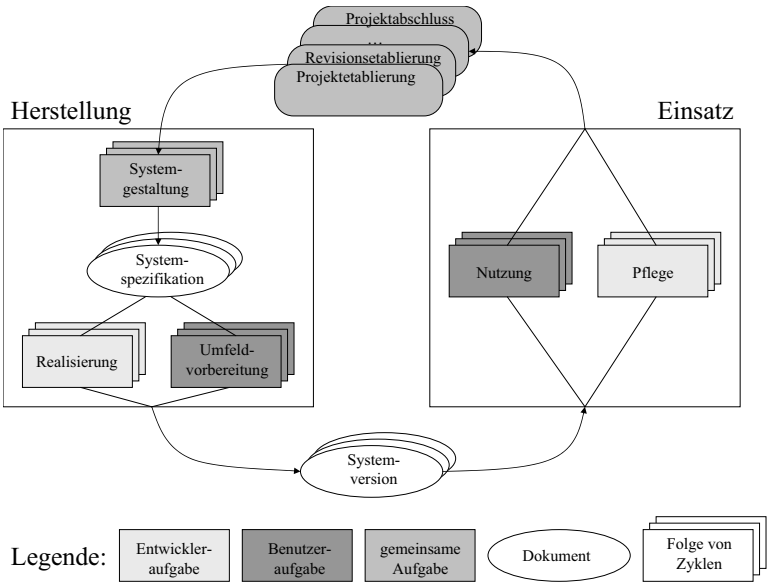

Abbildung 27: Zyklisches STEPS-Projektmodell

Ein Zyklus beginnt mit einer expliziten Etablierung des Zyklus durch die beteiligten Akteure, insbesondere der Entwickler und Benutzer. Gemeinsam wird nach der Etablierung das System gestaltet, wobei zunächst die Ermittlung der Anforderungen und anschließend die Systemgestaltung im

Vordergrund stehen. Ergebnis der Systemgestaltung ist eine Systemspezifikation in Form eines Dokuments. Dieses dient als Ausgangspunkt einerseits für die Entwickler, um daraus in prototypischen Zyklen eine Softwareversion zu erstellen, und andererseits für die Benutzer, um in ihrem Arbeitsumfeld entsprechende Vorbereitungen für die Softwaresystemnutzung zu treffen. Ergebnis beider Aktivitäten ist die Systemversion, die sich dann in der Nutzung durch die Benutzer und gepflegt durch die Entwickler im Einsatzkontext beweisen muss. Voraussetzung für dieses Vorgehen ist, dass die erstellten Versionen, Erstversion sowie alle weiteren Versionen, einsatzfähig sind und die Arbeitsprozesse der Benutzer sinnvoll unterstützen (Floyd 1991b, S. 23ff.). Dieses Vorgehen versteht sich nicht als ideale Vorgehensweise, sondern als „Rahmen, der zu einer Klasse von situationsspezifischen Strategien für individuelle Projekte führt. [...] das zyklische Projektmodell ist wie eine Landkarte, die ausgewählte Aspekte des Territoriums der Softwareentwicklung aufzeigt und miteinander verknüpft“ (Floyd 1991b, S. 25).

[2]Die Motivation für zyklische Vorgehensweisen entspringt der Einsicht, dass Arbeitsprozesse zur Softwareentwicklung in ihrem Verlauf nicht vollständig vorherbestimmbar sind, und dass das System, in Anpassung an veränderte Anforderungen aus dem Einsatzbereich, revidierbar sein muss (vgl. Floyd u. a. 1990; Kilberth u. a. 1994). Diese Anforderung an die Vorgehensweise führt zu einer Abkehr von Phasenmodellen (vgl. Lehner u. a. 1995), die Aktivitäten in einer linearen Weise mit vorweg bestimmten Zwischenergebnissen gliedern (vgl. Floyd und Züllighoven 2002; Lehner u. a. 1995).

Ein zyklisches Vorgehen fördert die Verschränkung zwischen der Herstellung und der Nutzung von Software sowie wechselseitige Lernprozesse zwischen allen Beteiligten (vgl. Floyd 1994). In kurzen Entwicklungszyklen können prototypische Systemversionen erstellt werden, die möglichst umgehend in ihre vorgesehenen Verwendungszusammenhänge eingebracht werden, um zeitnahe Erfahrungen für die Entwicklungsarbeit zu sammeln. Es geht darum, miteinander Probleme zu erschließen, tragfähige Lösungen zu erarbeiten, diese zu bewerten und zu revidieren, um so schrittweise ein gemeinsames Verständnis der Software sowie die mit ihr verbundenen Veränderungen der Handlungsmöglichkeiten im Einsatzkontext zu erlangen. Dazu sind geeignete Darstellungsmittel einzusetzen, die helfen, ein

2 Frühere Gedanken zu den folgenden drei Absätzen finden sich bei Jackewitz und Pape (2004).

gemeinsames Verständnis der betreffenden Organisationssituation herauszubilden (vgl. Züllighoven 1998; Krabbel und Wetzel 1998).

Jeder Zyklus der Softwareentwicklung dient sowohl der Herstellung einer Softwareversion als auch den Vorbereitungen im Einsatzumfeld der Software, wie zum Beispiel Qualifikationsmaßnahmen oder organisatorischen Anpassungen (vgl. Wulf und Rohde 1995; Floyd u. a. 1997; Rolf 1998). Gegenüber den Phasenmodellen wird damit die Verquickung von Diskursebenen mit der zeitlichen Reihenfolge von Entwicklungsschritten aufgehoben. Die Aktivitäten in unterschiedlichen Handlungsfeldern sind nicht grundsätzlich getrennten Arbeitsphasen oder Projektabschnitten zugeordnet, sondern werden zeitlich flexibel miteinander verzahnt (vgl. Floyd 1994; Floyd und Züllighoven 2002; Züllighoven 1998). Das bedeutet insbesondere auch, dass die Interaktion zwischen Entwicklern und Benutzern nicht auf bestimmte Phasen beschränkt, sondern kontinuierlich betrachtet wird. Methodisch lässt diese Interaktion u. a. durch die verschiedenen Formen des Prototypings unterstützen (Floyd u. a. 1997).

Neben anderen Prototyping-Konzepten (Lehner u. a. 1995, S. 93ff) bezieht sich STEPS insbesondere auf das evolutionäre Prototyping. Ziel des evolutionären Ansatzes ist die permanente Adaption und Weiterentwicklung der Software an sich verändernde Anforderungen. Der Begriff Evolution bzw. evolutionär „... erinnert an Darwin und provoziert Fragen nach dem Reproduktions- und Selektionsmechanismus von Software, für die es keine Antwort gibt. Ich betone an dieser Stelle immer, dass ‚Evolution' nicht auf der Ebene der Softwareveränderung selbst, sondern auf der Ebene unseres Verständnisses über die wünschenswerte Funktionalität und die Nutzungsmöglichkeiten von Software anzusiedeln ist. [...] Zum anderen erweckt der Sprachgebrauch den Eindruck, dass der Wandel, wie bei der natürlichen Evolution, sehr langsam erfolgt, und dass ‚niemals etwas fertig' wird. Schnell folgt der Einwand, das sei nicht praxisrelevant. In den Augen der Verfechter der evolutionären Systementwicklung verhält es sich genau umgekehrt: Prototyping und Ausbaustufen ermöglichen eine feingranulare Terminplanung, gravierende Missverständnisse werden frühzeitig beseitigt, die Erprobung liefert Hinweise auf die tatsächlich gebrauchte Funktionalität, das Aufwand-Nutzen-Verhältnis kann flexibler gestaltet werden" (Floyd u. a. 1997, S. 15).

Das zyklische Projektmodell wird für ein Vorgehen in der Softwarebereitstellung benötigt, da das Geschäftsmodell ASP nicht auf ein Vorgehen

eingeht. So wird in Kapitel 9 das zyklische Projektmodell in der Softwareentwicklung auf die Softwarebereitstellung adaptiert.

5.2.3 Aufgabenbezogene Anforderungsermittlung

Der Aufgabenbegriff wird in der Arbeitspsychologie, in Arbeitswissenschaften und der betriebswirtschaftlichen Organisationslehre zur Analyse und Beschreibung von Organisationen und menschlichem Handeln verwendet. Von dort hat der Aufgabenbegriff Einzug in die Informatik gefunden (vgl. Gryczan 1996; Krabbel 2000) und wird in der anwendungsorientierten Softwaretechnik folgendermaßen definiert (vgl. Krabbel 2000, S. 53):

- „Eine Aufgabe ist das Ziel oder die Pflicht, die ein Mensch vor sich sieht“ (Floyd 1991a, S. 7).
- „Eine Aufgabe wird von einer sachkundigen Person in einem Anwendungsbereich verantwortlich übernommen. Eine Aufgabe bildet eine sinnvolle und zielgerichtete Einheit [...]“ (Gryczan u. a. 1998, S. 604).
- „Unter einer Aufgabe verstehen wir eine situationsbedingte Aufforderung (gegebenenfalls auch an sich selbst) zur Ausführung von Tätigkeiten oder zur Erreichung von Zielen“ (Hesse u. a. 1994, S. 42).

In der aufgabenbezogenen Anforderungsermittlung erfolgt eine Ausdifferenzierung des Aufgabenbegriffs unter Einbeziehung des Begriffs der funktionellen Rolle. Eine funktionelle Rolle (vgl. Nygaard und Handlykken 1981; Floyd 1986; 1991a; Floyd und Züllighoven 2002) wird durch eine spezielle Aufgabe definiert oder besteht aus einer Bündelung von zusammengehörigen Aufgaben. Sie kann von einer oder mehreren Personen wahrgenommen werden. Umgekehrt kann eine Person auch mehrere funktionelle Rollen einnehmen. Die funktionelle Rolle ist unterschiedlich „... from that which is commonly used in social psychology where a role is described by the expectations which a person will adapt to in a given position in an organisation. ... [functional] roles specify functions and not persons.“ (Nygaard und Handlykken 1981, S. 161). Wesentlich ist auch der Unterschied zwischen einer funktionellen Rolle und einer Stelle in einer Organisation: „Eine funktionelle Rolle kann zwar einer Stelle entsprechen. Aber die Person einer Stelle kann über mehr als eine funktionelle Rolle verfügen“

(Krabbel 2000, S. 54). Mit der funktionellen Rolle ist es möglich, unabhängig von den Stellen und Personen einer Organisation die Verantwortung und Zuständigkeiten für Aufgaben zu beschreiben und die Zuordnung zu konkreten Personen in einem späteren Schritt nach zu holen. Die Wurzeln des Konzeptes der funktionellen Rolle sind in den frühen 70er Jahren zu finden. Nygaard und seine Kollegen prägten den Begriff der funktionelle Rolle im Zusammenhang mit der Entwicklung einer objektorientierten Methode zur Systembeschreibung und der Entwicklung der Programmiersprache DELTA (vgl. Fjellheim u. a. 1974; Nygaard 1976; Floyd u. a. 1989a).

Eine Aufgabensystematisierung bei Verwendung der funktionellen Rolle gestaltet sich wie folgt (vgl. Floyd 1991a; Floyd u. a. 1998a;b; Krabbel u. a. 1996; Krabbel 2000):

- Eine *Aufgabe* ist die kleinste unterscheidbare Einheit innerhalb eines Aufgabengebietes. Eine Aufgabe selbst kann aus Tätigkeiten bestehen.

- Ein *Aufgabengebiet* einer funktionellen Rolle bezeichnet eine Menge von miteinander materiell, informationell oder zeitlich verknüpften Aufgaben, die in ihrer Gesamtheit von anderen Aufgabengebieten der funktionellen Rolle trennbar sind. Ein Aufgabengebiet kann aus mehreren oder nur einer Aufgabe bestehen.

- Eine *übergeordnete Aufgabe* wird durch einen Organisationsbereich wahrgenommen. Sie besteht aus der Gesamtheit der Aufgabengebiete der dort zusammenwirkenden funktionellen Rollen.

- Eine *übergreifende Aufgabe* umfasst mehrere Aufgabengebiete, die, im Zusammenhang einer Vielzahl von funktionellen Rollen verschiedener Organisationsbereiche, an unterschiedlichen Orten bearbeitet werden. Sie erfordert ein hohes Maß an Flexibilität, da ihre Erledigung von äußeren Faktoren abhängig ist. Zu ihrer Durchführung ist eine Vielzahl von Handlungen zur Koordination notwendig.

Die aufgabenbezogene Anforderungsermittlung hilft in der Aufgabenperspektive, die verschiedenen Aufgaben strukturiert erfassen zu können (Kapitel 6). Darüber hinaus stellt sie die Grundlage für das im Folgenden erarbeitete kollektive Rollennetz dar.

5.2.4 Kollektives Rollennetz

Im ASP wurde eine Schwäche bei der Zuordnung von Aufgaben zu Akteuren konstatiert, und weder das Akteursmodell noch die eben dargestellte aufgabenbezogene Anforderungsermittlung leisten eine Verknüpfung von Akteuren und Aufgaben. So wird im Folgenden eine Verknüpfung der beiden Ansätze erarbeitet, in dessen Mittelpunkt das kollektive Rollennetz steht.

Konzeption des kollektiven Rollennetzes
Die kollektive Rolle wird in Anlehnung an die funktionelle Rolle und unter Einbeziehung der Aufgabenstruktur aus der aufgabenbezogenen Aufgabenanalyse (Abschnitt 5.2.3) und des Akteursmodells (Abschnitt 5.1.2) sowie inspiriert durch das ASP-Schichtenmodell (Abschnitt 4.1.1) wie folgt definiert:

- Eine kollektive Rolle entspricht einer übergeordneten Aufgabe.
- Eine kollektive Rolle kann von einem oder mehreren Akteuren wahrgenommen werden.
- Ein Akteur kann eine oder mehrere kollektive Rollen einnehmen.

Dabei stellt die kollektive Rolle, unter Heranziehung der Netzwerkorganisation (Abschnitt 5.1.3), ein Netz aus funktionellen Rollen dar, die in Kooperation (d. h. im Kollektiv) die übergeordnete Aufgabe leisten (vgl. Oberquelle 1987, S. 16 und 21ff.). Die kollektive Rolle entspricht damit dem abstrakten Abbild des Organisationsbereichs aus der Aufgabenanalyse und schafft so die Verknüpfung von der übergeordneten Aufgabe zu den Akteuren. Akteure werden hier im Sinne des Akteursmodells als real existierende Organisationen (z. B. Firmen) oder Organisationseinheiten/-bereiche (z. B. Abteilungen) verstanden, die eine oder mehrere kollektive Rollen wahrnehmen können, in Analogie zur funktionellen Rolle, die von einer oder mehreren Personen eingenommen werden kann.

Das kollektive Rollenetz verbindet die kollektiven Rollen miteinander, die durch Kommunikation oder Kooperation in Beziehung stehen. Hierbei hat die Benennung als „kollektives Rollennetz“ eine doppelte Bedeutung. Das kollektive Rollennetz ist nicht nur ein Netz aus kollektiven Rollen, sondern die kollektiven Rollen übernehmen die übergreifende Aufgabe in

Kooperation, so dass das Netz aus kollektiven Rollen selbst ein Kollektiv darstellt.

Aufgaben	**Rollen (Abstraktion)**	**Akteure**
Übergreifende Aufgabe	Kollektives Rollennetz (Netz kollektiver Rollen)	Akteursnetzwerk (Unternehmensnetzwerk, Arena)
Übergeordnete Aufgabe	Kollektive Rolle (Netz funktioneller Rollen)	Akteur(e) (Unternehmen / Org.bereich[e])
Aufgabengebiet Aufgabe	Funktionelle Rolle	Person(en)

Tabelle 1: Rollen als Abstraktion zwischen Aufgaben und Akteuren

Durch die Abstraktion der wirklichen Akteure ist es möglich, die Organisation einer beliebigen übergreifenden Aufgabe zu beschreiben und die Zuordnung zu Akteuren in einem späteren Schritt nachzuholen. Insofern ist das kollektive Rollennetz nicht flexibel im Sinne des Netzwerks, sondern muss als Schablone für ein Akteursnetzwerk verstanden werden, welches die übergreifende Aufgabe leistet bzw. leisten soll. Das kollektive Rollennetz stellt eine Vorgabe dar. Es benennt, welche kollektiven Rollen zur Ausführung einer übergreifenden Aufgabe wahrgenommen werden müssen. Die entsprechenden Akteure siedeln sich nach Vorgabe des kollektiven Rollennetzes in einem entsprechenden Akteursnetzwerk an, das wiederum die in der Netzwerkorganisation beschriebene Flexibilität aufweist und Akteure dynamisch abstößt oder neu aufnehmen kann.

Darüber hinaus muss im Sinne des Akteursmodells das Konfliktpotential bei der Kooperation von Akteuren betrachtet werden. Es ist unumgänglich, die Ziele der verschiedenen beteiligten Akteure in den Blick zu nehmen, aus denen sich u. a. vorhandene bzw. mögliche Konflikte erklären lassen. Dabei ist wichtig, zwischen den konkreten Zielen hinsichtlich der Kooperation und allgemeinen Akteurszielen zu unterscheiden.

Im Sinne der Netzwerkorganisation geht die Macht von einem Akteur im Netzwerk aus, daher wird auch im Akteursnetzwerk, welches das kollektive Rollennetz ausfüllt, ein Akteur die Machtposition innehaben. Eine Zuordnung der Machtposition an eine kollektive Rolle im kollektiven Rollennetz ist schwierig, da potentiell der mächtigste Akteur jede kollektive Rolle einnehmen könnte. Dennoch sollte, im Hinblick auf die übergreifende Aufgabe, eine kollektive Rolle definiert werden, bei der es aufgrund z. B.

einer zentralen Position Sinn macht, den mächtigsten Akteur im kollektiven Rollennetz zu verorten. Dies ermöglicht Schieflagen im Akteursnetzwerk wahrzunehmen.

Für die Bereitstellung ist mit dem kollektiven Rollennetz ein Ansatz erarbeitet worden, mit dem die Organisation der Softwarebereitstellung in der Organisationsperspektive auf einer abstrakten Ebene gestaltet werden kann (Kapitel 7). Darüber hinaus schließt er die bei ASP entdeckte Lücke hinsichtlich der Verknüpfung von Aufgaben und Akteure.

Unterschied zwischen Akteur und Rolle
Da eine Person sowohl einem oder mehreren Akteuren angehören als auch eine oder mehrere funktionelle Rollen einnehmen kann, stellt sich die Frage, wo der Unterschied zwischen einer Rolle (funktionell wie kollektiv) und einem Akteur liegt bzw. wie sich diese beiden Begriffe voneinander abgrenzen. Ein konkretes Beispiel ist die Bezeichnung „Lehrender“ bzw. „Lehrende“. Ist dies eine Rolle oder ein Akteur?

Akteure zeichnen sich durch Handlungsfähigkeit aus (Rammert 1993, S. 100f), während eine funktionelle bzw. kollektive Rolle eine Abstraktion von Aufgaben und Aufgabengebieten bzw. übergeordneten Aufgaben darstellt und nicht selbst handeln kann (vgl. Nygaard und Handlykken 1981; Floyd 1986; 1991a; Floyd und Züllighoven 2002). Ein Akteur kann aktiv handeln, eine Rolle nicht (vgl. Oberquelle 1987, S. 19). Hier offenbart sich ein Unterschied: da Akteure handlungsfähig sind, können sie Rollen einnehmen.

Ein weiterer Unterschied besteht in der Zugehörigkeit. Eine Rolle wird flexibel eingenommen, d. h. das Einnehmen von Rollen durch eine Person ist flüchtig. Rollen können generell schnell eingenommen und schnell wieder abgelegt werden. Rollen sind z. B. Verkäufer, Käufer, Anbieter, Nachfrager usw. Die Zugehörigkeit von Personen zu Akteuren ist z. B. durch Arbeitsverträge oder den Status gegeben. Sie kann nicht so einfach und schnell aufgegeben und gewechselt werden wie eine Rolle. Akteure sind z. B. Gewerkschaften, Rechenzentren, Professoren usw. Es besteht also ein Unterschied zwischen „Verkäufer“ und „Verkaufspersonal“. Eine Person, welche in einem Kaufhaus Waren verkauft, nimmt während ihrer Arbeitszeit mehrmals die Rolle des Verkäufers ein, wenn sie einem Käufer (Rolle) etwas verkauft. Sie gehört während, aber auch außerhalb der Tätigkeit des Verkaufens zum Akteur Verkaufspersonal. Diese Zugehörigkeit besteht

auch dann, wenn z. B. ein Streik das Warenhaus blockiert und die Person keinen Kundenverkehr hat.

Der Begriff „Lehrender“ muss daher als Rolle identifiziert werden, der z. B. an einer Hochschule durch Mitglieder der Akteure Professoren, Wissenschaftliche Mitarbeiter oder Studierende eingenommen werden kann. Ebenfalls wird deutlich, dass das Begriffspaar „Lehrende und Studierende“ irreführend ist, da hier Rolle und Akteur vermischt werden. Die Verwendung dieses Begriffpaars suggeriert, dass Studierende nicht die Rolle des Lehrenden einnehmen können. Dies mag für die traditionelle Lehrform der Vorlesung gelten, ist aber bei Referats- oder Projektseminaren nicht gegeben. In diesen Lehrformen arbeiten Studierende eigenständig und bringen in der Rolle des Lehrenden ihren Kommilitonen ihre Ausarbeitungen näher. Das Begriffspaar „Lehrende und Studierende“ steht projektorientierten Lehrformen diametral entgegen. Besser ist, von „Lehrenden und Lernenden“ zu sprechen.

5.3 Zwischenergebnis

Mit der Darstellung der Aufgabensystematik aus der aufgabenbezogenen Anforderungsanalyse, des zyklischen Projektmodells von STEPS, des Akteursmodells, der Netzwerkorganisation, der Transaktionskostentheorie und des kollektiven Rollennetzes als Verknüpfung von Aufgaben und Akteuren sind für alle in Abschnitt 4.6 entdeckten Defizite von ASP methodisch-theoretisch Grundlagen aus der Informatik vorbereitet worden. Sie müssen nun im Folgenden auf ASP angewandt werden, um ASP entsprechend zu fundieren und die in Abschnitt 4.6 konstatierten Schwächen zu überwinden. Diese Anwendung erfolgt pro Perspektive im folgenden Teil II dieser Arbeit.

II

Herleitung von eASP

- 6 -
Aufgabenperspektive – Was ist zu tun?

Dieses Kapitel nimmt sich der Schwächen von ASP in der Aufgabenperspektive an (Abschnitt 4.6):

- Unterschiedlich benutzte Aufgabengranularität,
- unklare Verknüpfung von Aufgaben zu Aufgabenbündeln und
- fehlende Gesamtsicht von Aufgaben.

Um sie zu überwunden, wird die Aufgabensystematik aus der aufgabenbezogenen Anforderungsermittlung (Abschnitt 5.2.3) als Grundlage für eine umfassende Beschreibung aller Aufgaben herangezogen. Die Aufgaben werden aus ASP-Wertschöpfungsketten (Abschnitt 4.2) und -Services (Abschnitt 4.3.1) extrahiert.

6.1 Konzeption

Im Folgenden wird eine Beschreibung von Aufgaben präsentiert, die sich anhand der Systematik in der aufgabenbezogenen Anforderungsermittlung „vom Groben ins Feine“ bewegt:

- übergreifende Aufgabe
- übergeordnete Aufgabe
- Aufgabengebiet
- einzelne Aufgabe

Zu beachten ist, dass die Ausführungen auf den Ebenen „einzelne Aufgabe“ und „Aufgabengebiet“ als Orientierungen dienen, die situativ an den konkreten Softwarebereitstellungskontext angepasst werden müssen.

6.1.1 Übergreifende Aufgabe „Software x bereitstellen“

Die Aufgabe *Software x bereitstellen* wird als übergreifende Aufgabe (Abschnitt 5.2.3) definiert und als Ausgangspunkt der Aufgabensystematisierung für die Softwarebereitstellung verwendet. Das x entspricht der konkreten Software, die bereitgestellt werden soll. Aus der Definition als übergreifende Aufgabe ergibt sich,

1. dass sie von einer Vielzahl von funktionellen Rollen an unterschiedlichen Orten bearbeitet werden muss,
2. dass sie ein hohes Maß an Flexibilität erfordert, da ihre Erledigung von äußeren Faktoren abhängig ist,
3. und dass zu ihrer Durchführung eine Vielzahl von Handlungen zur Koordination notwendig ist.

Punkt 1 führt zu den nachfolgend aus den ASP-Wertschöpfungsketten dargestellten übergeordneten Aufgaben, die auf der nächsten Ebene zu funktionellen Rollen führen. Punkt 2 ist für das Vorgehen in Kapitel 9 relevant. Punkt 3 ergänzt die übergeordneten Aufgaben durch die übergeordnete Aufgabe Kooperation, die u. a. der Koordination (Aufgabengebiet) aller Beteiligten dient.

Die übergreifende Aufgabe *Software x bereitstellen* besteht aus den übergeordneten Aufgaben *Anwendungsentwicklung*, *Betrieb*, *Benutzerbetreuung*, *Marketing*, *Kundenbetreuung* und *Zugriffsermöglichung* (vgl. Abschnitt 4.2 über Wertschöpfungsketten) sowie aus der übergeordneten Aufgabe *Kooperation* (vgl. Abschnitt 5.2.3: Definition der übergreifenden Aufgabe).

6.1.2 Übergeordnete Aufgaben

Die übergeordneten Aufgaben der übergreifenden Aufgabe *Software x bereitstellen* werden im Folgenden näher erläutert. Aus den Abschnitten 4.2 (ASP-Wertschöpfungsketten) und 4.3.1 (ASP-Services) ergeben sich die Aufgabengebiete der ersten sechs übergeordneten Aufgaben. Die der Kooperation ergeben sich der Definition der übergreifenden Aufgabe und aus den Ausführungen im Kapitel Organisationsperpektive. Dort wird ergänzend dargestellt, dass die übergeordnete Aufgabe Kooperation eine Sonderrolle einnimmt, da sie von allen Dienstleistern wahrgenommen werden

muss[1]. Die übergeordnete Aufgaben im Detail sind (vgl. Bleek und Jackewitz 2004; Jackewitz 2004):

- Die *Anwendungsentwicklung* umfasst die Herstellung, (Weiter-)Entwicklung und das Bug-Fixing der Anwendung, die zur Verfügung gestellt werden soll. Die Anwendungsentwicklung besteht aus den Aufgabengebieten: Leitung, Planung, Programmierung, Dokumentation, Entwicklungsumgebung, Releasemanagement und Fehlermanagement.

- Unter *Betrieb* wird die technische Bereitstellung der Infrastruktur inklusive der bereitzustellenden Anwendung und der zusätzlich benötigten Software verstanden. Aufgabengebiete: Hardware, Basissoftware, Anwendung, Datensicherheit und Ausfallsicherheit.

- Die *Benutzerbetreuung* umfasst alle Maßnahmen hinsichtlich eines „direkten“ Kontakts mit den Nutzenden. Aufgabengebiete: Handhabungssupport, Evaluation der Nutzung, Kommunikation.

- Mit *Marketing* wird die Präsentation der angebotenen Dienstleistungen - innerhalb eines Unternehmens sowie auf dem freien Markt - inkl. entsprechender Werbung bezeichnet. Marketing bedeutet Bekanntmachung der angebotenen Dienstleistungen. Aufgabengebiete: Bedarfsanalyse und Werbung.

- Unter *Kundenbetreuung* werden die Vertragsgestaltung und die Abrechnung von Dienstleistungen verstanden. Aufgabengebiete: Kommunikation, Verträge und Abrechnung.

- Die *Zugriffsermöglichung* stellt und wartet die technische Verbindung vom Nutzer zum Bereitsteller. Aufgabengebiete: Netzverbindung und Netzsicherheit.

- Die *Kooperation* umfasst alle Aufgaben, die die beteiligten Dienstleister in die Lage versetzen, die Software x und dazugehörige Aufgaben und Dienstleistungen wahrzunehmen bzw. zur Verfügung zu stel-

1 Diese Vorbezüglichkeit ist unschön, aber der Tatsache geschuldet, dass zwischen allen Perspektiven Querbezüge und gewisse Abhängigkeit bestehen, die in einem linearen Text nicht aufgelöst werden können.

len. Aufgabengebiete: Abrechnung, Kommunikation, Training, Vertragswerk und Koordination.

6.1.3 Aufgabengebiete, Aufgaben und funktionelle Rollen

Zur Erarbeitung der Aufgabengebiete tragen insbesondere die Ausführungen zu den ASP-Services (vgl. Abschnitt 4.3.1) bei. Sie gliedern die im vorigen Abschnitt genannten Aufgabengebiete in einzelne Aufgaben. Eine Beschreibung von konkreten Aufgaben ist in der ASP-Literatur nicht zu finden. Auf dieses Defizit haben Pape und Jackewitz (2002), Bleek u. a. (2003) und Jackewitz (2004) bereits hingewiesen und Aufgabenbeschreibungen erarbeitet. Über funktionelle Rollen wird die Verknüpfung der dargestellten Aufgaben zu konkreten Personen geleistet.

Um den Lesefluss dieser Arbeit nicht allzu sehr zu erschweren, wird auf eine detaillierte Beschreibung aller Aufgabengebiete, Aufgaben und funktionellen Rollen an dieser Stelle verzichtet und auf den Anhang verwiesen. Im Anhang befinden sich entsprechende Beschreibungen von Aufgabengebieten (Anhang A.1), Aufgaben (Anhang A.2) und funktionellen Rollen (Anhang A.3).

Es sei noch einmal darauf hingewiesen, dass die im Anhang und in der folgenden Zusammenfassung dargestellten Aufgabengebiete, Aufgaben und funktionellen Rollen nicht als fest angesehen, sondern im konkreten Bereitstellungsszenario angepasst und ggf. ergänzt werden müssen. Die Ausdifferenzierung der übergreifenden Aufgabe *Software x bereitstellen* auf den hier nicht ausführlich beschriebenen Ebenen dient der Anschaulichkeit, nicht der Vollständigkeit.

6.1.4 Zusammenfassender Überblick

In folgender Tabelle ist die Aufgabenstruktur für den Ansatz eASP zur Softwarebereitstellung zusammenfassend dargestellt. Sie wird im Weiteren dazu dienen, die Bewertung der Betrachtung der beiden Fallstudien zu visualisieren.

Übergeordnete Aufgabe	**Aufgabengebiet / funktionelle Rolle**	**Aufgabe**
Anwendungsentwicklung	Leitung / Entwicklungsleiter	Entwicklung koordinieren und überwachen
		Entwicklungsentscheidungen treffen

Übergeordnete Aufgabe	**Aufgabengebiet / funktionelle Rolle**	**Aufgabe**
	Planung / Entwicklungsplaner	Anforderungen analysieren
		Implementierungen planen und Zeitplan erstellen
	Programmierung / Programmierer	Features programmieren
		Fehler beheben
		Softwarearchitektur pflegen
	Dokumentation / Programmierer	Benutzerdokumentation pflegen
		Entwicklerdokumentation pflegen
	Entwicklungsumgebung / Entwicklungsadministrator	Entwicklungsclients pflegen
		Entwicklungsserver pflegen
		Zusätzlich benötigte Software pflegen
	Releasemanagement / Entwicklungsplaner	Release identifizieren, zusammenstellen und veröffentlichen
	Fehlermanagement / Fehlerbeauftragter	Hotline anbieten
		Fehlerberichte in Entwicklungsprozess integrieren
		Fehlerbehebung veröffentlichen
Betrieb	Hardware / Hardwareadministrator	Serverhardware pflegen
	Basissoftware / Softwareadministrator	Zusätzlich benötigte Software pflegen
	Anwendung / Anwendungsadministrator	Anwendung installieren, konfigurieren und pflegen
	Datensicherheit / Softwareadministrator	Back-up der Daten durchführen
		Virenscan der Daten durchführen
		Weitere Sicherheitsmechanismen einrichten
	Ausfallsicherheit / Hardwareadministrator	Klima- und Brandschutz installieren
		Stromversorgung sichern
Benutzerbetreuung	Handhabungssupport / Benutzerbetreuer	FAQ pflegen
		Hotline (Telefon, E-Mail) anbieten
	Kommunikation / Benutzerbetreuer	Benutzerwünsche und Fehlerberichte weiterleiten
		Informationsmaterial erstellen und verteilen
		Informationsveranstaltungen und Workshops organisieren
	Evaluation der Nutzung / Evaluator	Evaluation konzipieren und durchführen
		Ergebnisse in die Bereitstellung integrieren
Marketing	Bedarfsanalyse / Analyst	Bedarfsanalyse konzipieren und durchführen
		Ergebnisse in die Bereitstellung integrieren
	Werbung / Werbefachmann	Informationsmaterialien entwickeln und verbreiten
		Software und Dienstleistungen präsentieren
		Werbestrategie entwickeln
Kundenbetreuung	Kommunikation / Kundenbetreuer	Hotline (Telefon, E-Mail) anbieten
		Informationsmaterialien entwickeln und verschicken
		Kundenwünsche weiterleiten
	Verträge / Kundenbetreuer	Standardverträge entwickeln
		Vertragsverhandlungen führen
	Abrechnung /	Rechnungen stellen

Übergeordnete Aufgabe	**Aufgabengebiet / funktionelle Rolle**	**Aufgabe**
	Buchhalter	Zahlungen überwachen
Zugriffsermöglichung	Netzverbindung / Netzwerkadministrator	Verbindung etablieren und pflegen
	Netzsicherheit / Netzwerkadministrator	Netzwerksicherheit installieren und pflegen
Kooperation (alle Dienstleister)	Abrechnung / Verwalter	Rechnung stellen
		Zahlungen überwachen
	Kommunikation / Ansprechpartner	Internen Newsletter herausbringen
		Feedback geben
	Koordination / Koordinator	Beteiligung an Kooperationstreffen
		Überblick über Bereitstellung behalten
	Training / Trainer	Interne Workshops durchführen
	Vertragswerk / Verwalter	Kooperationsverträge entwickeln
		Vertragsverhandlungen führen
		Vertragsbrüche verhandeln

Tabelle 2: Aufgaben bei der übergreifenden Aufgabe *Software x bereitstellen*

6.2 Anwendung auf die Fallstudien

Die Anwendung der Aufgabenstruktur auf die Fallstudie CommSy@Uni.de und CommSy@WissPro verdeutlicht, welche Aufgaben wahrgenommen oder auch nicht wahrgenommen wurden. Dabei werden hinsichtlich der Aufgabenstruktur die Elemente der tabellarischen Übersichten in den Fallstudien nach den folgenden Kriterien gewertet:

- Das Element bestand bzw. wurde wahrgenommen:
  Weißer Hintergrund
- Das Element wurde teilweise wahrgenommen:
  Hellgrauer Hintergrund
- Das Element wurde nicht wahrgenommen:
  Dunkelgrauer Hintergrund
- Über das Element kann keine Aussage getroffen werden:
  Schwarzer Hintergrund

6.2.1 CommSy@Uni.de

Bei der Betrachtung der Fallstudie CommSy@Uni.de ist festzustellen, dass die Aufgabenstruktur aufgrund der speziellen Situation der Bereitstellung von CommSy durch Uni.de mit Aufgaben, Aufgabengebieten und funktionellen Rollen erweitert werden muss. Diese werden vorgestellt, ehe die Fallstudie hinsichtlich der Aufgaben analysiert wird.

Zusätzliche Aufgaben
In der Fallstudie CommSy@Uni.de mussten zusätzliche Aufgaben ausführen werden:

- *Apache Webserver pflegen*: Der Webserver (Apache) lieferte die Daten von CommSy mittels Webseiten aus. Er wurde installiert, konfiguriert und gewartet.
- *Benutzerkennungen verwalten*: Benutzerkennungen wurden freigeschaltet oder gesperrt, vergessene Passwörter neu gesetzt.
- *Didaktische Modelle entwickeln, erproben, auswerten und veröffentlichen*: CommSy folgt bestimmten didaktischen Konzepten, die (weiter-)entwickelt werden mussten. Diese Konzepte wurden dokumentiert und veröffentlicht, um den Nutzenden den Sinn des Designs von CommSy näher zu bringen.
- *Finanzierungsmodelle erstellen*: Die Finanzierung von CommSy erwies sich als schwierig, da sie, aufgrund von konkurrierenden unentgeltlichen Angeboten, für die Nutzenden ebenfalls kostenfrei sein sollte. Aus diesem Grund wurden alternative Finanzierungskonzepte erarbeitet.
- *MySQL-Datenbankserver pflegen*: Zur Speicherung der Daten wurde eine MySQL-Datenbank benötigt. Ein entsprechender Server musste installiert, konfiguriert und gewartet werden.
- *PHP-Interpreter pflegen*: Es wurde ein PHP-Interpreter benötigt, der installiert, konfiguriert und gewartet werden musste, da der CommSy-Code in PHP geschrieben war.

- *Projekträume verwalten*: CommSy-Projekträume mussten einmalig freigeschaltet oder ggf. gesperrt werden. Bei einer Sperrung wurde ein Kommunikationsprozess mit dem Veranstalter initiiert.
- *Resultate in den Entwicklungsprozess integrieren*: Die didaktischen Konzepte geben Rahmenbestimmungen vor, nach denen sich das Design von CommSy zu richten hat.
- *Sprechstunden anbieten*: Die didaktische Beratung erforderte in Einzelgesprächen mit den Veranstaltern, die CommSy-Projekträume einsetzten, Erwartungen an den Einsatz zu klären und Hinweise zur didaktischen Einbettung von CommSy in die Lehrveranstaltung zu geben. Geeignetes Mittel war eine regelmäßige Sprechstunde.
- *Veranstalterforum moderieren*: Dieses Forum diente allen Veranstaltern von CommSy-Projekträumen als Austauschmöglichkeit und der Benutzerbetreuung als Ort für die didaktische Beratung. Das Forum (ein CommSy-Projektraum) wurde eingerichtet und moderiert.

Zusätzliche Aufgabengebiete
Folgende zusätzliche Aufgabengebiete bestanden in der Fallstudie CommSy@Uni.de:

- *Didaktik*: CommSy unterstützt Lehr-Lernformen nach bestimmten didaktischen Prinzipien. Sie mussten entwickelt, gepflegt und veröffentlicht werden, um Nutzern das Design von CommSy verständlich zu machen und somit die „richtige“ (gedachte) Benutzung von CommSy zu fördern. Aufgaben: Resultate in den Entwicklungsprozess integrieren, didaktische Modelle entwickeln, erproben, auswerten und veröffentlichen.
- *Didaktische Beratung*: CommSy unterstützt projektorientierte Lehre bzw. projektorientiertes Arbeiten. Der Erfolg von CommSy beruht auf seiner Einbettung in eine Lehrveranstaltung und hängt somit von Lehrveranstalter ab. Die didaktische Beratung sollte den Veranstalter unterstützen, CommSy geeignet einzubeziehen. Über ein Veranstalterforum (CommSy-Projektraum) konnten die Veranstalter sich untereinander austauschen und die Benutzung eines CommSy-Projektraums aus Sicht eines normalen Benutzers erfahren. Aufgabe: Sprechstunden anbieten und Veranstalterforum moderieren.

- *Redaktion*: Die Redaktion eines CommSy-Servers umfasste die Verwaltung von Benutzerkennungen und CommSy-Projekträumen, so dass Nutzer zu CommSy einen Zugang erhielten. Aufgaben: Benutzerkennungen und Projekträume verwalten.

Zusätzliche funktionelle Rollen
In der Fallstudie CommSy@Uni.de existierten zusätzliche Rollen:

- *Didaktiker*: Der Didaktiker befasste sich mit den zugrunde liegenden didaktischen Konzepten. Aufgabengebiet: Didaktik.
- *Redakteur*: Der Redakteur eines CommSy-Servers verwaltet Benutzerkennungen und Projekträume. Aufgabengebiet: Redaktion.
- *Veranstaltungsbetreuer*: Der Veranstaltungsbetreuer beriet die Veranstalter bei der Integration von CommSy in deren Veranstaltungen. Aufgabengebiet: Didaktische Beratung.

Analyse der Fallstudie
Die Fallstudie CommSy@Uni.de kann wie folgt analysiert und bewertet werden:

Übergeordnete Aufgabe	**Aufgabengebiet / funktionelle Rolle**	**Aufgabe**
Anwendungsentwicklung	Leitung / Entwicklungsleiter	Entwicklung koordinieren und überwachen
		Entwicklungsentscheidungen treffen
	Planung / Entwicklungsplaner	Anforderungen analysieren
		Implementierungen planen und Zeitplan erstellen
	Programmierung / Programmierer	Features programmieren
		Fehler beheben
		Softwarearchitektur pflegen
	Dokumentation / Programmierer	Benutzerdokumentation pflegen
		Entwicklerdokumentation pflegen
	Entwicklungsumgebung / Entwicklungsadministrator	Entwicklungsclients pflegen
		Entwicklungsserver pflegen
		Zusätzlich benötigte Software pflegen
	Releasemanagement / Entwicklungsplaner	Release identifizieren, zusammenstellen und veröffentlichen
	Fehlermanagement / Fehlerbeauftragter	Hotline anbieten
		Fehlerberichte in Entwicklungsprozess integrieren
		Fehlerbehebung veröffentlichen
	Didaktik / Didaktiker	Resultate in den Entwicklungsprozess integrieren
		Didaktische Modelle entwickeln, erproben, auswerten und veröffentlichen

Übergeordnete Aufgabe	**Aufgabengebiet / funktionelle Rolle**	**Aufgabe**
Betrieb	Hardware / Hardwareadministrator	Serverhardware pflegen
	Basissoftware / Softwareadministrator	Betriebssystem pflegen
		MySQL-Datenbankserver pflegen
		PHP-Interpreter pflegen
		Apache Webserver pflegen
	Anwendung / Anwendungsadministrator	CommSy installieren, konfigurieren und pflegen
	Datensicherheit / Softwareadministrator	Back-up der Daten durchführen
		Virenscan der Daten durchführen
		Weitere Sicherheitsmechanismen einrichten
	Ausfallsicherheit / Hardwareadministrator	Klima- und Brandschutz installieren
		Stromversorgung sichern
Benutzerbetreuung	Handhabungssupport / Benutzerbetreuer	FAQ pflegen
		Hotline (Telefon, E-Mail) anbieten
	Kommunikation / Benutzerbetreuer	Benutzerwünsche und Fehlerberichte weiterleiten
		Informationsmaterial erstellen und verteilen
		Informationsveranstaltungen und Workshops organisieren
	Evaluation der Nutzung / Evaluator	Evaluation konzipieren und durchführen
		Ergebnisse in die Bereitstellung integrieren
	Didaktische Beratung / Veranstaltungsbetreuer	Sprechstunden anbieten
		Veranstalterforum moderieren
	Redaktion / Redakteur	Benutzerkennungen verwalten
		Projekträume verwalten
Marketing	Bedarfsanalyse / Analyst	Bedarfsanalyse konzipieren und durchführen
		Ergebnisse in die Bereitstellung integrieren
	Werbung / Werbefachmann	Informationsmaterialien entwickeln und verbreiten
		Software und Dienstleistungen präsentieren
		Werbestrategie entwickeln
Kundenbetreuung	Kommunikation / Kundenbetreuer	Hotline (Telefon, E-Mail) anbieten
		Informationsmaterialien entwickeln und verschicken
		Kundenwünsche weiterleiten
	Verträge / Kundenbetreuer	Finanzierungsmodelle erstellen
		Standardverträge entwickeln
		Vertragsverhandlungen führen
	Abrechnung / Buchhalter	Rechnungen stellen
		Zahlungen überwachen
Zugriffsermöglichung	Netzverbindung / Netzwerkadministrator	Verbindung etablieren und pflegen
	Netzsicherheit / Netzwerkadministrator	Netzwerksicherheit installieren und pflegen
Kooperation (alle Dienstleister)	Abrechnung / Verwalter	Rechnung stellen
		Zahlungen überwachen

Übergeordnete Aufgabe	Aufgabengebiet / funktionelle Rolle	Aufgabe
	Kommunikation / Ansprechpartner	Internen Newsletter herausbringen
		Feedback geben
	Koordination / Koordinator	Beteiligung an Kooperationstreffen
		Überblick über Bereitstellung behalten
	Training / Trainer	Interne Workshops durchführen
	Vertragswerk / Verwalter	Kooperationsverträge entwickeln
		Vertragsverhandlungen führen
		Vertragsbrüche verhandeln

Tabelle 3: Analyse der übergreifenden Aufgabe *CommSy bereitstellen* in der Fallstudie CommSy@Uni.de

Die dunkle Färbung der Tabelle soll verdeutlichen, dass bei der übergreifenden Aufgabe *CommSy bereitstellen* viele Problemfelder auftraten, obwohl alle übergeordneten Aufgaben wahrgenommen wurden. Die Problemfelder im Detail:

- *Anforderungen analysieren*: Nutzeranforderungen sind nicht durch eine entsprechende empirische Analyse erhoben worden. Anforderungen wurden vielmehr antizipativ bestimmt.
- *Softwarearchitektur pflegen*: Die Pflege der Softwarearchitektur wurde unter Termindruck stark vernachlässigt.
- *Dokumentation*: Eine Benutzerdokumentation wurde nicht kontinuierlich, dafür aber ein Moderationshandbuch gepflegt. Entwicklerdokumentation existierte nur minimal, nicht zentralisiert.
- *Fehlermanagement*: Eine systematische Verwaltung des Fehlerbehebungsprozesses, von der Meldung eines Fehlers über die Behebung und die Dokumentation bis hin zur Bekanntmachung der Fehlerbehebung, bestand nicht. Zu Beginn des Entwicklungsprozesses für Uni.de wurden ein Karteikasten und ein Bug-Tracker nur willkürlich genutzt. Karteikasten und Bug-Tracker setzten sich in der CommSy-Entwicklung nicht durch.
- *Basissoftware*: Die benötigte Software für den Betrieb von CommSy wurde installiert und einmal konfiguriert. Später fand weder eine Wartung noch eine weitere Konfiguration der Software statt.

- *Anwendung*: Uni.de spielte nur dreimal Updates von CommSy auf seinen Server ein. Das Aufspielen des vierten Updates scheiterte an der fehlenden Konfiguration des Webservers. Ein weiterer Versuch wurde nicht unternommen.

- *Datensicherheit*: Es sind keine Einbruchsversuche auf den CommSy-Server bekannt. Da der CommSy-Server in einem privatwirtschaftlichen Rechenzentrum stand, waren sehr wahrscheinlich eine Firewall und auch weitere Sicherheitsmechanismen aktiv. Bestätigt werden kann das an dieser Stelle nicht. Über Viren in den Dateien bzw. Datenverlust durch Serverabstürze ist ebenfalls nichts bekannt geworden. Daher gibt es keinen Anhaltspunkt für Virenscans und Datenbackups.

- *Ausfallsicherheit*: Es gab mehrere Ausfälle des CommSy-Servers. Warum der Server ausfiel, wurde nie herausgefunden. Über das Vorhanden- oder Nichtvorhandensein einer Ausfallsicherheit kann keine klare Aussage gemacht werden. Da der CommSy-Server in einem privatwirtschaftlichen Rechenzentrum (anfangs in München, später in Frankreich) installiert war, hat diese Sicherheit mit hoher Wahrscheinlichkeit bestanden.

- *Handhabungssupport*: Eine FAQ wurde nicht gepflegt und eine Hotline für Benutzer zur Lösung von Benutzungsproblemen nicht explizit angeboten. Dennoch wurde im ersten halben Jahr ein Handhabungssupport über die E-Mail-Adresse info@commsy.de geleistet.

- *Kommunikation (Benutzerbetreuung)*: Ein Betreuungsmarketing fand nicht statt. Benutzer wurden weder über die angebotenen Dienstleistungen der Benutzerbetreuung informiert noch auf die richtigen Ansprechpartner aufmerksam gemacht.

- *Evaluation der Nutzung*: Eine Evaluation der Nutzung mittels quantitativer oder qualitativer Erhebungsmethoden wurde nicht durchgeführt.

- *Didaktische Beratung*: Das Veranstalterforum wurde über ein halbes Jahr moderiert, danach wurde die Moderation eingestellt. Da Uni.de

nur einmal, relativ am Anfang, in das Veranstalterforum einlud, fanden nur wenige Veranstalter in das Veranstalterforum. Eine regelmäßige Sprechstunde z. B. mittels Chat wurde nicht angeboten.

- *Redaktion*: Das Freischalten der Projekträume wurde während des gesamten Zeitraums von Uni.de übernommen. Sie erfüllte die Aufgabe gegen Ende der Kooperation nicht mehr zur Zufriedenheit der Nutzenden. Beispielsweise erfolgte auf eine Anfrage nach einer Benutzerkennung keine Reaktion.

- *Marketing*: Marketing fand nicht statt. Mit einer Ausnahme: Auf den Webseiten von Uni.de `http://www.uni.de/` waren spärlich Informationen zu finden, die von den Entwicklern vorbereitet und von Uni.de überarbeit auf deren Webseite gestellt wurden.

- *Kundenbetreuung*: Eine Kundenbetreuung fand nur in einem sehr geringen Maße in Bezug zur Bannerwerbung statt. Hier sind Geschäftsbeziehungen gepflegt worden.

- *Kommunikation (Kundenbetreuung)*: Es gab keine Kunden, demnach auch keine Pflege von Kundenbeziehungen.

- *Verträge*: Für die Werbebanner bestanden Verträge. Ansonsten wurden keine weiteren Finanzierungsmodelle für Kunden umgesetzt, obwohl Anfragen nach alternativen Finanzierungsmodellen bestanden. Alternative Sponsoringideen wurden zwar aufgeworfen, aber nicht verfolgt. Die Finanzierung erfolgte nur über das Sponsoring der Bannerwerbung.

- *Abrechnung (Kundenbetreuung)*: In diesem Fall können nur hinsichtlich der Werbebanner Aufgaben aus dem Gebiet der Abrechnung wahrgenommen worden sein, da nur diese Kundenbeziehung existierte. Die Entwickler sollten an den Einnahmen durch die Bannerwerbung prozentual beteiligt werden, haben aber nie entsprechende Tantiemen erhalten. Über die Abrechnung der Werbebanner kann also keine Aussage getroffen werden.

- *Netzsicherheit*: Eine Sicherung des Übertragungsweges vom Client zum Server hat es nicht gegeben. So wurden weder Virtual Privat Networks (VPNs) noch SSL-Verschlüsselung eingesetzt.

- *Kooperation*: Bis auf eine Ausnahme bestanden keine Aktivitäten, den jeweils anderen Dienstleister in die Lage zu versetzen, die übergreifende Aufgabe *CommSy bereitstellen* wahrnehmen zu können. Es wurden keine Workshops durchgeführt. Kommunikation fand bis auf anfängliches Feedback auf Grund von Unstimmigkeiten, welches darüber hinaus schnell eingestellt wurde, nicht statt. Allein der CommSy-Projektraum für Veranstalter, in dem die Entwickler Uni.de beraten sollten, kann als Aktivität in diese Richtung gewertet werden. Wobei Uni.de dieses Angebot nicht annahm und die Entwickler es nicht wirklich verfolgten.

Die vertraglichen Grundlagen für die Kooperation zwischen den Entwicklern von CommSy und Uni.de wurden entwickelt, verhandelt und unterschrieben. Entsprechende Rechnungen sind gestellt und der Zahlungsverkehr überwacht worden. Vertragsbrüche, wie die fortgeführte Bereitstellung von CommSy durch Uni.de trotz beendetem Kooperationsvertrag, wurden nicht verhandelt.

Im Marketing und in der Kundenbetreuung sind große Defizite zu verzeichnen. Daher konnte keine finanzielle Basis für die Bereitstellung von CommSy bei Uni.de aufgebaut werden. Weitere Defizite sind in der Benutzerbetreuung zu erkennen, da viele Aufgaben nur zeitweilig erfüllt wurden. Die Defizite in der Weiterentwicklung sind kleiner zu bewerten, sie beziehen sich weitgehend auf die Dokumentation und die Fehlerverwaltung. Über den Betrieb kann nur die Aussage getroffen werden, dass CommSy technisch, mit teilweise längeren Ausfallzeiten, zur Verfügung stand. Hinsichtlich der Kooperation zwischen den Entwicklern und Uni.de bestanden, bis auf eine anfängliche Kommunikation zur Initialisierung der Kooperation, keine weiteren Aktivitäten. Einzig der Zahlungsverkehr wurde kontinuierlich überwacht und entsprechende Rechnungen wurden gestellt.

6.2.2 CommSy@WissPro

Im Folgenden wird die Fallstudie CommSy@WissPro aus der Aufgabenperspektive betrachtet. Hier zeigen sich Unterschiede zur vorherigen Fallstudie.

Zunächst werden zusätzliche Aufgaben, Aufgabengebiete und funktionelle Rollen vorgestellt, bevor die Fallstudie hinsichtlich der Aufgaben analysiert wird.

Zusätzliche Aufgaben
Wie in der ersten Fallstudie ergeben sich Änderungen in der Aufgabenstruktur. Dabei sind die Erweiterungen der Aufgabenstruktur durch zusätzliche funktionellen Rollen und Aufgabengebiete mit der vorigen Fallstudie identisch, so dass sie an dieser Stelle nicht näher betrachtet werden müssen. Auch auf der Ebene von Aufgaben sind die Unterschiede sehr gering, so dass für die Bereitstellung von CommSy im Forschungs- und Entwicklungsprojekt WissPro nur auf die Unterschiede in den Aufgaben zu der Bereitstellung von CommSy bei Uni.de eingegangen wird.

Im Gegensatz zur ersten Fallstudie gibt es hier die Aufgabe Projekträume verwalten nicht, denn in der CommSy Version ab 2.0 müssen nicht mehr nur einzelne Projekträume verwaltet werden, sondern ganze CommSys mit beliebig vielen Projekträumen. Aus diesem Grund ist die Verwaltung der Projekträume in die Hand der Projektraumveranstalter sprich Nutzer gelegt worden. Dafür gab es ab Version 2.2 den Gemeinschaftsraum, für den die Redaktion verantwortlich ist und in dem sie Nutzungsanreize geben soll.

- *CommSys verwalten*: Ein CommSy konnte hinsichtlich Farbe, Name, Rubriken und weiterer Einstellmöglichkeiten, wie die Gestaltung des Portals, konfiguriert werden. Darüber hinaus mussten eventuell Projekträume bei Missbrauch gesperrt oder Materialien bei Bedarf weböffentlich freigeschaltet werden.

- *Nutzungsanreize geben*: Im Gemeinschaftsraum wurden von der Redaktion Nutzungsanreize in Form von kontinuierlich eingestellten Beispielen (Einträgen) gegeben.

Analyse der Fallstudie
Die Fallstudie CommSy@WissPro kann wie folgt bewertet werde:

Übergeordnete Aufgabe	**Aufgabengebiet / funktionelle Rolle**	**Aufgabe**
Anwendungsentwicklung	Leitung / Entwicklungsleiter	Entwicklung koordinieren und überwachen
		Entwicklungsentscheidungen treffen
	Planung / Entwicklungsplaner	Anforderungen analysieren
		Implementierungen planen und Zeitplan erstellen
	Programmierung / Programmierer	Features programmieren
		Fehler beheben
		Softwarearchitektur pflegen

Übergeordnete Aufgabe	Aufgabengebiet / funktionelle Rolle	Aufgabe
	Dokumentation / Programmierer	Benutzerdokumentation pflegen
		Entwicklerdokumentation pflegen
	Entwicklungsumgebung / Entwicklungsadministrator	Entwicklungsclients pflegen
		Entwicklungsserver pflegen
		Zusätzlich benötigte Software pflegen
	Releasemanagement / Entwicklungsplaner	Release identifizieren, zusammenstellen und veröffentlichen
	Fehlermanagement / Fehlerbeauftragter	Hotline anbieten
		Fehlerberichte in Entwicklungsprozess integrieren
		Fehlerbehebung veröffentlichen
	Didaktik / Didaktiker	Resultate in den Entwicklungsprozess integrieren
		Didaktische Modelle entwickeln, erproben, auswerten und veröffentlichen
Betrieb	Hardware / Hardwareadministrator	Serverhardware pflegen
	Basissoftware / Softwareadministrator	Betriebssystem pflegen
		MySQL-Datenbankserver pflegen
		PHP-Interpreter pflegen
		Apache Webserver pflegen
	Anwendung / Anwendungsadministrator	CommSy installieren, konfigurieren und pflegen
	Datensicherheit / Softwareadministrator	Back-up der Daten durchführen
		Virenscan der Daten durchführen
		Weitere Sicherheitsmechanismen einrichten
	Ausfallsicherheit / Hardwareadministrator	Klima- und Brandschutz installieren
		Stromversorgung sichern
Benutzerbetreuung	Handhabungssupport / Benutzerbetreuer	FAQ pflegen
		Hotline (Telefon, E-Mail) anbieten
	Kommunikation / Benutzerbetreuer	Benutzerwünsche und Fehlerberichte weiterleiten
		Informationsmaterial erstellen und verteilen
		Info.veranstaltungen und Workshops organisieren
	Evaluation der Nutzung / Evaluator	Evaluation konzipieren und durchführen
		Ergebnisse in die Bereitstellung integrieren
	Didaktische Beratung / Veranstaltungsbetreuer	Sprechstunden anbieten
		Veranstalterforum moderieren
	Redaktion / Redakteur	Benutzerkennungen verwalten
		CommSys verwalten
		Nutzungsanreize geben
Marketing	Bedarfsanalyse / Analyst	Bedarfsanalyse konzipieren und durchführen
		Ergebnisse in die Bereitstellung integrieren

Übergeordnete Aufgabe	**Aufgabengebiet / funktionelle Rolle**	**Aufgabe**
	Werbung / Werbefachmann	Informationsmaterialien entwickeln und verbreiten
		Software und Dienstleistungen präsentieren
		Werbestrategie entwickeln
Kundenbetreuung	Kommunikation / Kundenbetreuer	Hotline (Telefon, E-Mail) anbieten
		Informationsmaterialien entwickeln und verschicken
		Kundenwünsche weiterleiten
	Verträge / Kundenbetreuer	Finanzierungsmodelle erstellen
		Standardverträge entwickeln
		Vertragsverhandlungen führen
	Abrechnung / Buchhalter	Rechnungen stellen
		Zahlungen überwachen
Zugriffsermöglichung	Netzverbindung / Netzwerkadministrator	Verbindung etablieren und pflegen
	Netzsicherheit / Netzwerkadministrator	Netzwerksicherheit installieren und pflegen
Kooperation (alle Dienstleister)	Abrechnung / Verwalter	Rechnung stellen
		Zahlungen überwachen
	Kommunikation / Ansprechpartner	Internen Newsletter herausbringen
		Feedback geben
	Koordination / Koordinator	Beteiligung an Kooperationstreffen
		Überblick über Bereitstellung behalten
	Training / Trainer	Interne Workshops durchführen
	Vertragswerk / Verwalter	Kooperationsverträge entwickeln
		Vertragsverhandlungen führen
		Vertragsbrüche verhandeln

Tabelle 4: Analyse der übergreifenden Aufgabe *CommSy bereitstellen* in der Fallstudie CommSy@WissPro

Im Vergleich mit der Fallstudie CommSy@Uni.de fällt auf, dass bei CommSy@WissPro die graue Färbung der Tabelle insgesamt heller geworden ist. Dies deutet auf eine intensivere Wahrnehmung der Aufgaben hin. Dennoch gab es verschiedene Problemfelder:

- *Anforderungen analysieren*: Die Analyse von Anforderungen setzte im Projekt WissPro erst später ein.
- *Entwicklerdokumentation*: Während der Benutzerdokumentation im Projekt WissPro sofort eine größere Beachtung zugesprochen wurde

(vgl. Großmann u. a. 2004), rückte die Notwendigkeit einer Entwicklerdokumentation erst spät in den Blick des Entwicklungsteams.

- *Datensicherheit*: In diesem Bereitstellungskontext bestand eine tägliche Datensicherung des CommSy-Servers durch das Rechenzentrum der Universität Hamburg (Regionales Rechenzentrum - RRZ). Weitere Sicherheitsmaßnahmen, wie z. B. Virenscans, wurden nicht vorgenommen. Es ist aber kein Fall bekannt, dass in den CommSy-Server eingebrochen bzw. Viren über ihn verteilt wurden.

- *Ausfallsicherheit*: Zur Ausfallsicherheit gehörte die Bereitstellung einer unterbrechungsfreien Stromversorgung (USV) für den CommSy-Server. Klima- und Brandschutz bestand bereits im Rahmen der Sicherung des Fachbereichs Informatik der Universität Hamburg.

- *FAQ pflegen*: Eine FAQ wurde zunächst aufgebaut, dann aber nur sporadisch gepflegt.

- *Didaktische Beratung*: Zur didaktischen Beratung wurden zunächst Sprechstunden mit Lehrveranstaltern durchgeführt und ein Veranstalterforum angeboten. Die Moderation des Veranstalterforums wurde, aus Mangel an Beteiligung von Veranstaltern, zur Hälfte der Laufzeit des Projektes eingestellt. Ungefähr zur selben Zeit wurde das Angebot an Sprechstunden zurückgenommen, denn sie stellten eine erhebliche Arbeitsbelastung dar und ihre Wirkung war umstritten (vgl. Jackewitz 2004; Pape und Rolf 2004).

- *Marketing*: Aspekte des Marketings standen nicht im Fokus des Projekts. Dennoch wurden Maßnahmen hinsichtlich eines Marketings ergriffen, z. B. wissenschaftliche Publikationen rund um CommSy (vgl. CommSy 2004) und die Teilnahmen an Konferenzen: LearnTec 2002 und 2003, CeBIT 2003. Die Nutzung von CommSy hat sich im Wesentlichen mittels Mund-zu-Mund-Propaganda unter den Nutzern verbreitet. Da ein finanzielles Interesse an der Bereitstellung von CommSy erst spät in WissPro geweckt wurde, beschäftigten sich die Mitarbeiter erst gegen Ende des Projekts mit Werbestrategien.

- *Kundenbetreuung*: Während der Fallstudie CommSy@WissPro bestand eine Dienstleister-Kunden-Beziehung zwischen dem Projekt

WissPro und dem Bundesministerium für Bildung und Forschung (BMBF), welches das Projekt finanzierte. Es sind Vertragsverhandlungen geführt, Rechnungen gestellt und der Zahlungsverkehr überwacht worden. Außerdem erhielt das BMBF jährlich Informationsmaterialien in Form von formalen Projektberichten und einen inhaltlichen Bericht zum Projektende. So bestand in dieser Fallstudie eine Kundenbetreuung der „besonderen Art". Da es während der Laufzeit von WissPro nur Nutzer der Bereitstellung von CommSy, aber keine zahlenden Kunden gab, bestand kein Anlass zu einer „normalen" Kundenbetreuung.

- *Abrechnung*: Aufgrund des anfangs fehlenden finanziellen Interesses wurden zwischen den Dienstleistern (Benutzerbetreuungs-, Betriebs- und Entwicklungsteam) keine Zahlungen vereinbart.

- *Vertragswerk*: Ebenfalls aus dem Mangel an einem finanziellen Interesse an der Bereitstellung von CommSy existierten auch keine Verträge zwischen den Dienstleistern. Unabhängig davon hatte natürlich jeder Mitarbeiter des Projekts Arbeitsverträge mit der Universität Hamburg. Darin waren zwar Aufgabengebiete enthalten, sie wirkten sich aber nicht direkt auf die Kooperation zwischen den Dienstleistern aus. Dennoch wurden im Prozess der Bereitstellung von CommSy im Projekt WissPro Kommunikationsregeln aufgestellt. Diese Regeln wurden situativ angepasst und führten letztendlich zu einer funktionierenden Kooperation.

Zusammenfassend ist in dieser Fallstudie festzustellen, dass die Erledigung der übergeordneten Aufgaben insgesamt zu einer erfolgreichen Bereitstellung führte, wie durch Evaluationsergebnisse (Strauss u. a. 2003) bestätigt wurde.

6.3 Zwischenergebnis

Durch die Aufgabenperspektive wird deutlich, welche Aufgaben zur Bereitstellung von Software gehören und welchen funktionellen Rollen diese zuzuordnen sind.

Als übergreifende Aufgabe und als Ausgangspunkt für die Aufgabenstruktur wurde die Aufgabe *Software x bereitstellen* definiert. Sie besteht

aus den übergeordneten Aufgaben *Anwendungsentwicklung*, *Betrieb*, *Benutzerbetreuung*, *Marketing*, *Kundenbetreuung* und *Zugriffsermöglichung*, erarbeitet aus ASP-Services (vgl. Abschnitt 4.3.1) und aus ASP-Wertschöpfungsketten (vgl. Abschnitt 4.2). Weiterhin besteht die übergreifende Aufgabe *Software x bereitstellen* aus der übergeordneten Aufgabe *Kooperation*. Sie ergibt sich aus der Definition der übergreifenden Aufgabe aus der aufgabenbezogenen Anforderungsermittlung (vgl. Abschnitt 5.2.3) und muss von allen beteiligten Dienstleistern übernommen werden.

Mit der vorgestellten Aufgabensystematisierung konnten Problemfelder in den beiden Fallstudien benannt werden. Welcher Dienstleister für welche Aufgabe bzw. funktionelle Rolle verantwortlich war und warum Aufgaben nicht übernommen wurden bzw. warum die Kooperation zwischen den Beteiligten in der einen Fallstudie nicht, in der anderen funktionierte, kann die vorgestellte Aufgabenperspektive nicht benennen. Sie bleibt an dieser Stelle abstrakt, ohne eine Zuordnung der Aufgaben zu den beteiligten Akteuren zu leisten oder die Verbindungen unter den Beteiligten genauer zu betrachten. Um dies zu erreichen, wird im nächsten Kapitel die Organisationsperspektive eingenommen.

- 7 -
Organisationsperspektive – Wer tut was?

Die Schwächen von ASP in der Organisationsperspektive sind (Abschnitt 4.6):

- Unklare Verknüpfung von Aufgaben zu Akteuren,
- unzureichende und unfundierte Benennung von kollektiven Rollen,
- keine Auseinandersetzung mit dem Akteursbegriff und den Beziehungen zwischen Akteuren,
- fehlende Auseinandersetzung mit dem Dienstleistungsbegriff und
- fehlende Betrachtung von Transaktionskosten.

Zur Überwindung dieser Schwächen werden das Akteursmodell (Abschnitt 5.1.2), die Netzwerkorganisation (Abschnitt 5.1.3) und das Konzept des kollektiven Rollennetzes (Abschnitt 5.2.4) auf das ASP-Player-Modell und das ASP-Schichtenmodell (Abschnitt 4.1.1) angewandt. Das Konzept des kollektiven Rollennetzes leistet dabei die Verknüpfung von den im vorigen Kapitel vorgestellten Aufgaben über die kollektiven Rollen zu den beteiligten Akteuren. Dargestellt werden darüber hinaus die Folgen für den Ansatz eASP aus der Auseinandersetzung mit dem Dienstleistungsbegriff (Abschnitt 5.1.5) und mit der Transaktionskostentheorie (Abschnitt 5.1.4), die die Ausführungen im ASP über vertragliche Aspekte (Abschnitt 4.4.1) ergänzen.

7.1 Konzeption

In diesem Abschnitt wird ein kollektives Rollennetz für den Softwarebereitstellungsansatz eASP präsentiert und der Schritt vom kollektiven Rollennetz als Muster bzw. Schablone zum wirklichen Akteursnetzwerk betrachtet. Danach werden die Auswirkungen des Dienstleistungsbegriffs auf

den Kontext der Softwarebereitstellung dargestellt, die zu vertraglichen Regelungen, Gebühren und Kosten sowie zu verschiedenen Qualitäten und Quantitäten in der Bereitstellung führen.

7.1.1 Kollektives Rollennetz

Für den Softwarebereitstellungsansatz eASP können aus den Arbeiten von Farleit (2000), Günther u. a. (2001), Gillian u. a. (1999) und Tao (2000) zum ASP-Player- und ASP-Schichtenmodell (Abschnitt 4.1.1) folgende kollektive Rollen mit entsprechenden übergeordneten Aufgaben und Verbindungen zu anderen kollektiven Rollen herausgearbeitet werden (vgl. Bleek und Jackewitz 2004; Jackewitz 2004):

- Der *Kunde* kauft[1] bzw. bezahlt die vom Dienstleistungsanbieter angebotenen, kostenpflichtigen Anwendungen und Dienstleistung für die Nutzer (seine Angestellten oder Kunden). Der Ansprech- bzw. Kooperationspartner in Fragen der Abrechnung, Gewährleistungspflichten und weiterer rechtlicher Aspekte ist für den Kunden der Dienstleistungsanbieter.

- Der *Nutzer* benutzt die vom Dienstleistungsanbieter angebotenen und vom Kunden bezahlten Anwendungen und Dienstleistungen.

- Der *Dienstleistungsanbieter* stellt dem Kunden Dienstleistungen aus einer Hand zur Verfügung. Er übernimmt die Kundenbetreuung, d. h. er pflegt die Kundenbeziehungen und rechnet die genutzten Dienstleistungen gegenüber dem Kunden ab. Um entsprechenden Dienstleistungen zur Verfügung zu stellen bzw. diese vermarkten zu können, kooperiert der Dienstleistungsanbieter mit dem Anwendungspartner, Hostingpartner, Zugangspartner, Supportpartner und Marketingpartner. Außerdem muss insbesondere er, als zentraler Punkt im kollektiven Rollennetz, in der Koordination aller Beteiligten Verantwortung übernehmen, zumal die Bereitstellung einer Software in seinem Namen für Kunde und Nutzer erfolgt.

1 Eine kollektive oder funktionelle Rolle kann per Definition nicht aktiv handeln (siehe Abschnitt 5.2.3 und 5.2.4); dies können nur die Akteure, die die Rolle einnehmen. Dennoch wird im Folgenden diese etwas unsaubere Schreibweise gewählt, um Formulierungen wie „Die Akteure, die die Rolle des Kunden einnehmen, kaufen . . . “ zu vermeiden.

- Der *Anwendungspartner* stellt die Anwendung (Software) für den Dienstleistungsanbieter zur Verfügung. Eine entsprechende Anwendungsentwicklung muss vor und während der Überlassung an den Dienstleistungsanbieter stattfinden.
- Der *Hostingpartner*[2] übernimmt den Betrieb, d. h. das Hosting der Anwendung und zusätzlicher Software für den Dienstleistungsanbieter. Er stellt große Teile der technischen Infrastruktur bereit.
- Der *Zugangspartner* bietet dem Dienstleistungsanbieter den Zugang vom Kunden bzw. Nutzer zum Hostingpartner an. Er ist für die Zugangsermöglichung verantwortlich.
- Der *Supportpartner*[3] übernimmt die Benutzerbetreuung der Nutzer beim Kunden im Namen des Dienstleistungsanbieters.
- Der *Marketingpartner* übernimmt das Marketing, u. a. die Platzierung der angebotenen Dienstleistungen im Markt und die Kundenakquirierung im Namen des Dienstleistungsanbieters.
- Der *Hardwarezulieferer* versorgt die Partner mit entsprechend benötigter Hardware.
- Der *Softwarezulieferer* versorgt die Partner mit entsprechend benötigter Software.

Die kollektiven Rollen können hinsichtlich der Erbringung von Dienstleistungen in Schichten angeordnet werden. Hierbei definiert sich eine Schicht über angebotene Dienstleistungen, die sie der darüber liegenden Schicht anbietet und die letztendlich nötig ist, um der obersten Schicht (der Kundenschicht) eine Anwendung in einem Softwarebereitstellungsmodell anbieten zu können.

2 Hier wird auf den englischen Begriff „Hosting“ zurückgegriffen, da der Begriff „Betriebspartner“ zu sehr an den Betrieb als Organisation bzw. Unternehmen erinnert und nicht an die übergeordnete Aufgabe „Betrieb“ der technischen Infrastruktur.

3 Hier wird ebenfalls auf den englischen Begriff „Support“ zurückgegriffen, da „Benutzerbetreuungspartner“ zu lang und „Betreuungspartner“ zu sehr an den Begriff „Benutzungsbetreuung“ erinnert, der umfassender als die „Benutzerbetreuung“ verstanden wird (vgl. Jackewitz (1998); Pape (2004); Pape u. a. (2002b); Pape und Jackewitz (2002) bzw. Abschnitt 2.3.5).

Durch die Erbringung von Dienstleistungen ergibt sich eine Verflechtung der kollektiven Rollen zwischen den Schichten, die vertraglich geregelt werden sollte. Damit eine kollektive Rolle einer Schicht bestimmte Dienstleistungen „nach oben“ anbieten kann, sind darüber hinaus auch Kooperationen von kollektiven Rollen innerhalb einer Schicht nötig, die ebenfalls vertraglich festgehalten werden sollten. So ergibt sich auch eine Verknüpfung von kollektiven Rollen innerhalb einer Schicht, was zum folgenden kollektiven Rollennetz führt.

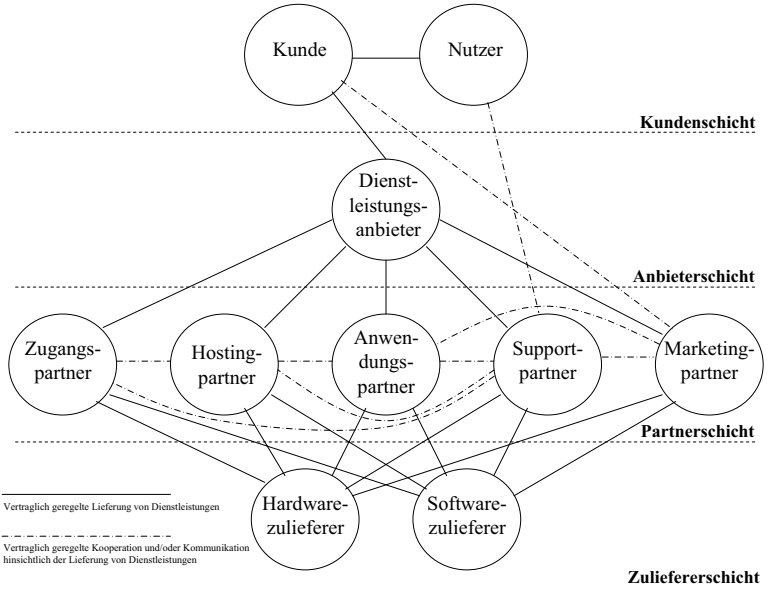

Abbildung 28: Kollektives Rollennetz bei der Bereitstellung von Software

Die Verbindungen zwischen den kollektiven Rollen veranschaulichen die vertraglich (Abschnitt 7.1.4) zu regelnde Kooperation und/oder Kommunikation bzw. die ebenfalls vertraglich zu regelnde Dienstleistungserbringung. Zusammen ergeben die Verbindungen das nötige Kommunikationsnetz der kollektiven Rollen zur Bereitstellung und Nutzung einer Software. Dabei zeigen die durchgezogenen Linien die Kommunikation hinsichtlich der Vorbereitung der Bereitstellung, d. h. der Aushandlung von Dienstlei-

stungsbeziehungen zwischen den kollektiven Rollen, sowie die Kommunikation hinsichtlich der Abrechnung der angebotenen und genutzten Dienstleistungen. Die gestrichelten Linien zeigen die Kommunikation, die während der konkreten Softwarenutzung ebenfalls auftritt. So gibt es z. B. bei der Bereitstellung einer Software Kommunikation zwischen dem Nutzer und dem Supportpartner. Die vertragliche Regelung für diese Kommunikation wird jedoch über den Kunden, Dienstleistungsanbieter und Supportpartner vorher sichergestellt. In diesem Sinne ist im Dienstleistungsanbieter auch die Position zu erkennen, die der mächtigste Akteur im entsprechenden Akteursnetzwerk einnehmen sollte. Der Dienstleistungsanbieter ist der zentrale Punkt bei der Bereitstellung von Software und trägt damit die Verantwortung für die Softwarebereitstellung. Die Schichten von oben nach unten im Detail sind:

- *Kundenschicht*: Die Kundenschicht besteht aus den kollektiven Rollen Kunde und Nutzer. Der Nutzer verwendet die vom Kunden erworbenen Dienstleistungen des Dienstleistungsanbieters. Zur Spezifikation dieser Nutzung kann es zwischen dem Nutzer und dem Kunden vertragliche Regelungen geben. Im Beispiel eines großen Unternehmens, welches Dienstleistungen bei einem Dienstleistungsanbieter bezieht, sind das Unternehmen der Kunde und die Angestellten die Nutzer, die über einen Arbeitsvertrag mit dem Unternehmen verbunden sind. Wenn eine Einzelperson eine Software bei einem Dienstleistungsanbieter benutzt, dann ist sie gleichzeitig Nutzer und Kunde. Eine vertragliche Regelung in diesem Fall ist sehr unwahrscheinlich. Sollte ein Sponsor die Bereitstellung einer Software finanzieren, ist dieser der Kunde und die Personen, die der Sponsor unterstützen will, sind die Nutzer. Vertragliche Regelungen in Form von Nutzungsbedingungen sind hier üblich.

  Die Verbindung zwischen dem Kunden und dem Marketingpartner zwecks Kundenwerbung (z. B. Bekanntmachung von neuen Dienstleistungsangeboten) sowie zwischen dem Nutzer und dem Supportpartner zwecks Benutzerbetreuung (z. B. Handhabungssupport) wird insbesondere über den Dienstleistungsanbieter vertraglich geregelt.

- *Anbieterschicht*: Auf der Anbieterschicht befindet sich nur der Dienstleistungsanbieter, der als „single point of contact“ (Grohmann 2002,

S. 71) Dienstleistungen von den kollektiven Rollen der Partnerschicht kauft bzw. bündelt und der Kundenschicht anbietet. Da es neben den vertraglichen Verbindungen auch Kommunikationsbeziehungen gibt, die an dem Dienstleistungsanbieter „vorbeilaufen", sollte er besser mit „single point of contract" und nicht „contact" bezeichnet werden.

- *Partnerschicht*: Auf der Partnerschicht befinden sich Supportpartner, Zugangspartner, Hostingpartner, Anwendungspartner und Marketingpartner, die ihre Dienstleistungen dem Dienstleistungsanbieter verkaufen und garantieren müssen. Zur Lieferung der entsprechenden Dienstleistungen müssen die kollektiven Rollen in dieser Schicht teilweise in Kooperation und Kommunikation treten. Diese Verknüpfungen sind generell in Verträgen mit dem Dienstleistungsanbieter geregelt. Sie sollten darüber hinaus in schriftlichen Vereinbarungen zwischen den jeweiligen kollektiven Rollen spezifiziert und im Sinne einer Dienstleistungserbringung verstanden werden. Die Verknüpfungen im Einzelnen:
  - Der Supportpartner sollte mit dem Anwendungspartner, Hostingpartner und Zugangspartner bei komplexen Support-Anfragen Rücksprache halten.
  - Der Supportpartner muss vom Marketingpartner über Marketingmaßnahmen informiert werden, um entsprechend auf diesbezügliche Supportanfragen reagieren zu können.
  - Der Marketingpartner benötigt Informationen über die zu bewerbende Anwendung vom Anwendungspartner.
  - Hostingpartner und Anwendungspartner sollten hinsichtlich des Betriebs der Anwendung kooperieren.
  - Zugangspartner und Hostingpartner müssen gemeinsam den Zugang zum Dienstleistungsangebot sichern.
- *Zuliefererschicht*: Auf der Zuliefererschicht befinden sich die Rollen der Hard- und Softwarezulieferer, die die kollektiven Rollen der Partnerschicht mit entsprechender benötigter Hard- und Software beliefern. An der Bereitstellung der eigentlichen Software sind sie nur sekundär beteiligt.

Beim Übergang vom abstrakten kollektiven Rollennetz zu konkreten Akteuren muss das Konfliktpotential bei der Kooperation von Akteuren miteinbezogen werden (vgl. Abschnitt 5.1.2). Konflikte sind auf der Ebene von Akteuren in der Regel Zielkonflikte. Es ist daher unumgänglich, die Ziele der verschiedenen beteiligten Akteure in den Blick zu nehmen. Wichtig ist, zwischen den konkreten Zielen hinsichtlich der Kooperation und allgemeinen Akteurszielen zu unterscheiden.

7.1.2 Akteursnetzwerk

Mit dem kollektiven Rollennetz ist die Verknüpfung von Aufgaben zu kollektiven Rollen gelungen und der Schritt zu den beteiligten Akteuren nicht mehr weit. Je nach Bereitstellungsszenario beteiligen sich andere Akteure an der Bereitstellung einer Software bzw. nehmen auf sie Einfluss. Für den Ansatz eASP wird daher kein generelles Akteursnetzwerk einer Softwarebereitstellung definiert. Dennoch werden in diesem Abschnitt in Rekurrierung auf die Netzwerkorganisation (Abschnitt 5.1.3) allgemeine Aussagen über das Akteursnetzwerk erarbeitet, das die Softwarebereitstellung durchführen soll.

Bereitstellungsnetzwerke von verschiedenen Anwendungen sind grundsätzlich unterschiedlich. Dies bedeutet allerdings nicht, dass ein Akteursnetzwerk nicht verschiedene Anwendungen bereitstellen kann. Dieser Fall ist im Sinne der Wirtschaftlichkeit sogar eher wahrscheinlich. Die Bereitstellung einer Anwendung kann aber auch von komplett unterschiedlichen Akteursnetzwerken getragen werden, die sich z. B. im Markt als Konkurrenten gegenüberstehen. Ein Kunde hat dadurch die Möglichkeit, sich ein Netzwerk der Bereitstellung einer Anwendung auszusuchen und an sein eigenes anzuknüpfen. Da die Bereitstellung selbst ein Netzwerk ist, kann ein Unternehmen auch bestimmte kollektiven Rollen übernehmen und quasi für sich selbst die Bereitstellung in Teilen oder gesamt übernehmen.

Grundlage für den flexiblen, orts- und/oder zeitunabhängigen Austausch von

- einzelnen technischen Komponenten,
- Akteuren,
- gesamten Bereitstellungsnetzwerken und

- der Anwendung (inkl. Bereitstellungsnetzwerk) durch eine andere Anwendung (inkl. Bereitstellungsnetzwerk)

ist die Setzung auf internationale Standards und die Wahl von (fast) überall vorhandenen Technologien (z. B. Internettechnologien) (Abschnitt 8.1). Die Wahl der Bereitstellung einer Anwendung kann demnach (aus Sicht eines Kunden) unabhängig von örtlichen oder zeitlichen Restriktionen getroffen werden. Die Entscheidung wird zum großen Teil durch die anfallenden Kosten (Gebühren, Aufwand und Transaktionskosten - siehe Anschnitt 7.1.5) bestimmt. Mögliche weitere Faktoren sind:

- Die grundsätzliche Unterscheidung in einerseits unternehmenskritische und andererseits nicht kritische Daten und die damit verbundene Frage: Welche Daten (zuzüglich der entsprechenden Anwendungen) eines Unternehmens lassen sich outsourcen?

- Die Reputation des Dienstleistungsanbieters (als Repräsentant des Netzwerks) bzw. des gesamten Bereitstellungsnetzwerkes. Hier stellt sich im Einzelfall die Vertrauensfrage: Kann ein Kunde dem Dienstleistungsanbieter bzw. dem gesamten Netzwerk vertrauen?

- Die Zukunftsfähigkeit eines Bereitstellungsnetzwerkes. Wird dieses Bereitstellungsnetzwerk (in welcher Form auch immer) langfristig existieren?

7.1.3 Dienstleistung

In der ASP-Literatur ist eine Diskussion des Begriffs Dienstleistung bzw. synonym des Begriffs Service nicht zu finden. Die Begriffe werden als selbstverständlich vorausgesetzt. Für den Ansatz eASP wird die Auseinandersetzung mit dem Dienstleistungsbegriff aus dem Serviceflowansatz (Abschnitt 5.1.5) übernommen und die Auswirkungen auf die Softwarebereitstellung im Folgenden diskutiert. Insbesondere die Beziehungen der beteiligten Akteure erscheinen unter dem Begriff Dienstleistung in einem neuen Licht.

Der Begriff Dienstleistung bzw. die Anwendung des Begriffs auf den Kontext der Softwarebereitstellung stellt die Bereitstellung einer Software unter das Primat der Kundenorientierung:

- *Dienstleistungsgeflecht*: Die Betrachtung von Servicegeber und Servicenehmer kann auf die Beziehung der kollektiven Rollen Kunde und Dienstleistungsanbieter angewendet werden. Dabei können beiden kollektiven Rollen die jeweils verbleibenden Rollen auf den jeweiligen Seiten zugeordnet und beide können als Kollektiv im Sinne eines Netzwerks verstanden werden. Darüber hinaus können auch die Beziehungen zwischen allen kollektiven Rollen als Dienstleistungsbeziehung angesehen werden, da zwischen ihnen Leistungen angeboten bzw. in Anspruch genommen werden. Dies bedeutet, dass das kollektive Rollennetz ist als ein Dienstleistungsgeflecht anzusehen.

- *Vertragsgeflecht*: Dem Verständnis des Begriffs Service folgend bedeutet ein Dienstleistungsgeflecht auch ein Vertragsgeflecht. Es sieht Verträge zwischen jeder kollektive Rolle mit jeder anderen kollektive Rolle vor, mit der sie in Kontakt bzw. Kooperation steht. Auf diese Weise werden Art und Umfang der Servicebeziehung definiert und transparent.

- *Gebühren und Kosten*: In vertraglichen Regelungen werden insbesondere die monetären Gegenleistungen der Servicenehmer definiert. Die Betrachtung dieser monetären Vereinbarungen, in den Verträgen zwischen allen kollektiven Rollen, ist eine gute Approximation der Kosten einer Softwarebereitstellung. Die Kosten treten in den vertraglichen Regelungen meist als Gebühren auf, die unterschiedlich (einmalig, wiederkehrend, nutzungsabhängig) abgerechnet werden können.

- *Qualität und Quantität*: Da sich die Existenz eines Servicegebers auf der Zufriedenheit seiner Servicenehmer gründet, wird er alles daransetzen, diese Zufriedenheit zu erzeugen. Da er wirtschaftlichen Zwängen unterliegt, wird er aber nur soviel Engagement in die Bedürfnisbefriedigung investieren, bis sie sich einstellt (vgl. Trieneken u. a. 2004). Dies tritt bei vielen unterschiedlichen Servicenehmern zu unterschiedlichen (Zeit-)Punkten ein, was bedeutet, dass Dienstleistungen in verschiedenen Qualitäten und Quantitäten vom Servicegeber angeboten und durchgeführt werden. Qualität und Quantität beziehen sich dabei auf die mit der Dienstleistung verbundenen Aufgaben.

Im Folgenden wird sich den drei Aspekten Vertragsgeflecht, Gebühren und Kosten sowie Qualität und Quantität angenommen, da eine Auseinandersetzung mit diesen Aspekten, im Gegensatz zum Dienstleistungsgeflecht (Abschnitt 7.1.1), bisher nicht stattgefunden hat.

7.1.4 Vertragliche Regelungen

Bereits in der Aufgabenstruktur (Kapitel 6) und dem kollektiven Rollennetz wurden vertragliche Vereinbarungen zwischen den kollektiven Rollen angemahnt. Diese Forderungen werden nun begründet und konkretisiert.

§311 des Bürgerlichen Gesetzbuchs (BGB 2002) nimmt wie folgt zum Sinn von Verträgen Stellung: „Zur Begründung eines Schuldverhältnisses durch Rechtsgeschäft [...] ist ein Vertrag zwischen den Beteiligten erforderlich, soweit nicht das Gesetz ein anderes vorschreibt." Demnach sollten vertragliche Regellungen zwischen allen an der Softwarebereitstellung beteiligten Akteuren vereinbart werden, um Rechte und Pflichten in der Kooperation, insbesondere in Hinblick auf Schwierigkeiten, eindeutig zu klären (vgl. Rosenhagen 2002; Zwipf und Schönfelder 2002).

Neben der „Begründung eines Schuldverhältnisses" (BGB 2002, §311) haben die vertraglichen Regelungen noch eine andere Funktion. Sie machen die Aufgaben, Aufwände, Ressourcen und Werte der angebotenen bzw. angenommenen Dienste und Dienstleistungen transparent und damit kalkulierbar.

Insofern macht es Sinn, bereits zwischen den kollektiven Rollen Verträge zu vereinbaren, auch wenn verschiedene kollektive Rollen vom selben Akteur übernommen werden. So kann auch innerbetrieblich eine Transparenz und Kalkulierbarkeit bei der Bereitstellung von Software erreicht werden. Es müssen in diesem Fall nicht Verträge im harten juristischen Sinne, sollten aber dennoch schriftliche Fixierungen von Aufgaben und Pflichten mit entsprechenden Qualitäts- und Quantitätsangaben sein.

Die Art der Verträge ist vom Inhalt, der von den Vertragspartnern vereinbarten Leistungspflicht, abhängig und kann verschiedene Mischformen von juristischen Vertragsarten enthalten (Abschnitt 4.4.1). Die Benutzung von SLAs in den vertraglichen Regelungen macht die Leistungen in der Bereitstellung von Software in ausreichender Detailtiefe transparent und klärt darüber hinaus Verantwortlichkeiten.

Die in der ASP-Literatur diskutierten SLAs (Abschnitt 4.4.2) sind eine gute Grundlage hinsichtlich vertraglicher Regelungen. Allerdings decken sie nicht die übergeordnete Aufgabe der Kooperation ab, die durch die Definition der Softwarebereitstellung als übergreifende Aufgabe hinzu gekommen ist (Abschnitt 6.1.1). Insofern müssen die in Abschnitt 4.4.2 beschriebenen SLAs (Netzwerk-, Hosting-, Application-, Support-SLA) um eine Kooperations-SLA erweitert werden.

Das Kooperations-SLA stellt die schriftliche Fixierung von Aufgaben und Pflichten hinsichtlich der übergeordneten Aufgabe Kooperation dar. Es gibt dementsprechend einen Rahmen für die Kooperation unter den beteiligten Akteuren vor. Dazu gehören im Einzelnen (vgl. Falkowski und Voß 2003, S. 5f):

- *Kommunikation*: Hinsichtlich der Kooperation müssen bei den kollektiven Rollen Ansprechpartner für die anderen kollektiven Rollen benannt und deren „Reaktionszeit“ vereinbart werden. Darüber hinaus sind Kommunikationswege für unterschiedliche Kommunikationsanlässe zu definieren.

- *Eskalationsmanagement*: Bei plötzlich auftretenden Störungen muss auf ein zuvor vereinbartes Reaktionsschema zurückgegriffen werden können. Dieses Schema legt fest, in welchen Schritten und welchen Zeiträumen, wer welche Störungsbeseitigungsmaßnahmen umsetzt. Dieses Schema definiert das Zusammenspiel der verschiedenen Support-Instanzen der kollektiven Rollen und deren Einschaltungskriterien.

- *Vertragsdauer und Kooperationszyklen*: Die Vertragsdauer wird in der Regel auf ein bis drei Jahre festgelegt. Die Vertragspartner sollten zusätzlich jederzeit die SLAs an neu entstehende bzw. veränderte Bedürfnisse anpassen können. Dazu können periodische Treffen (z. B. jährlich) vereinbart werden (siehe Kapitel 9 zum Vorgehen in der Softwarebereitstellung).

- *Beendigungsvereinbarungen*: Neben einem Eskalationsmanagement sollten auch Vereinbarungen für die Beendigung der Kooperation geschlossen werden, insbesondere Regelungen über die Freigabe und den Transfer der für die Bereitstellung wichtigen Daten (z. B. Kundendaten).

Mehrere Kooperationszyklen (siehe Abschnitt 9.1.1) innerhalb einer Vertragsdauer bedeuten für Kooperationsverträge zwischen kollektiven Rollen, dass in Rahmenverträgen die langfristige Kooperation erklärt und in konkreten Verträgen die Kooperation ausdifferenziert werden sollte. Dieser Vorschlag für Vertragsformen in eASP folgt den Ausführungen von Zwipf und Schönfelder (2002) zu Vertragsarten und -formen im ASP (Abschnitt 4.4.1).

7.1.5 Gebühren und Kosten

Bei der Betrachtung der Kosten einer Softwarebereitstellung ist zwischen den gesamten Kosten und den in Rechnung gestellten Kosten zu unterscheiden. Zur besseren Unterscheidung werden die in Rechnung gestellten Kosten fortan Gebühren genannt. Gebühren können folgendermaßen klassifiziert werden (Abschnitt 4.3.2):

- einmalig (z. B. Einrichtungsgebühr oder Softwarelizenzkosten beim Kauf)
- wiederkehrend (z. B. monatliche Grundgebühr, jährliche Lizenzkosten)
- nutzungsabhängig (z. B. nach Übertragungsvolumen oder Zeiteinheiten berechnete Kosten)

Klassisches Beispiel dieser verschiedenen Gebührenarten ist das Telefon. Für die Einrichtung eines Telefonanschlusses wird eine einmalige Einrichtungsgebühr erhoben. Monatlich werden dann eine wiederkehrende Grundgebühr und eine nutzungsabhängige Gebühr für die geführten Telefongespräche in Rechnung gestellt. Die Gebühren pro Kunde sagen allerdings nur wenig über die gesamten Kosten der Bereitstellung aus, selbst wenn sie über alle Kunden addiert werden. Sämtliche im Dienstleistungsnetzwerk in Rechnung gestellten Gebühren, also auch die erhobenen Gebühren zwischen anderen Akteurrollen außer Kunde und Dienstleister, sind nur eine gute Approximation der gesamten Kosten der Bereitstellung, treffen sie aber nicht genau.

Die Kosten der Bereitstellung ergeben sich als Addition aus den Aufwänden aller Beteiligten sowie den zur Kooperation nötigen Transaktions-

kosten (Abschnitt 5.1.4). Diese Kosten können in Menschjahren, -monaten bzw. -tagen berechnet werden.

Transaktionskosten bezeichnen den mit einer arbeitsteiligen Aufgabenerfüllung verbundenen Koordinationsaufwand (Picot u. a. 1998; Weiß und Längsfeld 2000). Für an der Softwarebereitstellung beteiligte Akteure sind Transaktionskosten die anfallenden Kosten

- in der Anbahnung ihrer Beteiligung, verursacht durch die Suche nach Informationen und das Führen von Gesprächen mit potentiellen Partnern, Kunden oder Dienstleistern,
- in Vertragsverhandlungen zur Vereinbarung der Beteiligung im Akteursnetzwerk der konkreten Softwarebereitstellung,
- in der zur Leistungserbringung notwendigen Abstimmung zwischen allen Beteiligten,
- in der Kontrolle der vereinbarten Leistungserbringung und des Zahlungsverkehrs und
- in der Anpassung der Beziehungen im Bereitstellungsnetzwerk bei auftretenden Veränderungen.

Für die Berechnung der Bereitstellungskosten einer Software bedeutet dies, dass die Dienstleistungserstellungs- und -nutzungskosten zum größten Teil in den übergeordneten Aufgaben *Anwendungsentwicklung*, *Benutzerbetreuung*, *Betrieb*, *Kundenbetreuung*, *Marketing* und *Zugriffsermöglichung* und der größte Teil der Transaktionskosten in der übergeordneten Aufgabe *Kooperation* zu verorten sind.

Letztendlich interessieren einen beteiligten Akteur bei der Bereitstellung von Software seine tatsächlichen Kosten, die sich aus seinem eigenen Arbeitsaufwand (in z. B. Menschmonaten oder -tagen) und den von anderen Akteuren abgerechneten Leistungen berechnen. Bei der Berechnung der Kosten für einzelne beteiligte Akteure müssen dann noch die Transaktionskosten des einzelnen Akteurs addiert werden (vgl. Jayatilaka u. a. 2002). Auf der Grundlage der Aufschlüsselung seiner Kosten kann ein Akteur dann, im Sinne der Transaktionskostentheorie (Abschnitt 5.1.4), seine Beteiligung an der Softwarebereitstellung bewerten und für sich entscheiden, ob z. B. ein Outsourcing der Softwarebereitstellung oder von Teilen

der Softwarebereitstellung wirtschaftlich vorteilhafter gegenüber einer internen Lösung ist (Endres 2004).

7.1.6 Qualitäten und Quantitäten

Den Ausführungen über die ASP-Marktlandschaft „The Application Service Provider Market Landscape“ von Gillian u. a. (1999, S. 5) (Abschnitt 4.3.1) sowie den Folgerungen aus der Auseinandersetzung mit dem Dienstleistungsbegriff (Abschnitt 7.1.3) können zwei interessante Aspekte für den Bereitstellungsansatz eASP entnommen werden:

1. Die in Kapitel 6 vorgestellten Aufgaben können von den entsprechenden Akteuren in verschiedenen Quantitäts- und Qualitätsstufen angeboten werden (folgt aus dem Dienstleistungsbegriff).

2. Je nach Software sind andere Quantitäts- und Qualitätsstufen in der Bereitstellung erforderlich. Folgende These wird aufgestellt: Je komplexer die bereitgestellte Software, desto höher ist die benötigte Quantität und Qualität in der Bereitstellung (folgt aus der ASP-Marktlandschaft).

Hier ist mit Quantität die Anzahl der wahrgenommenen Aufgaben und mit Qualität die Qualität ihrer Durchführung gemeint, was je nach Aufgabe größere Schnelligkeit, Sorgfalt, Ausführlichkeit oder ähnliches bedeutet.

In Anlehnung an Gillian u. a. (1999) und den Perspektiven Arbeitsplatz, Arbeitsgruppe und Unternehmen aus der IT-unterstützten Organisationsgestaltung (Abschnitt 5.1.1) werden aus Kundensicht folgende drei Nutzungsarten von Softwarebereitstellungsangeboten für den Ansatz eASP definiert:

- *Persönliche Nutzung*: Für die persönliche Nutzung werden einfache, sofort nutzbare Anwendungen (Office-Software, Termin- und Reiseplanung, E-Mail, Adressenverwaltung, Desktop Publishing) angeboten. Hohe Kundenzahlen und ein hohes Datenaufkommen sind typisch für diese Art der Nutzung. Die Bereitstellungsleistungen umfassen überwiegend technische Aspekte (Betrieb, Zugriffsermöglichung) und einen minimalen Benutzersupport. d. h. die Quantität und in Teilen auch die Qualität sind hier eher gering.

- *Kooperative Nutzung*: Unter der kooperativen Nutzung wird Groupware im weitesten Sinne verstanden: Conferencing, Unified Messaging, Projektmanagement, Sales Automation, Onlineshops, usw. In der kooperativen Nutzung steht insbesondere die Verfügbarkeit der Anwendungen im Fokus, die durch entsprechende SLAs schriftlich in Verträgen fixiert werden. Zum Leistungsumfang persönliche Nutzung kommen hier Leistungen der Sicherheit und des technischen Supports hinzu. Das heißt, dass die Qualität in den einzelnen Aufgaben ist im Vergleich zur persönlichen Nutzung höher und auch die Quantität der übernommenen Aufgaben ist gewachsen.

- *Organisatorische Nutzung*: Mit organisatorischer Nutzung wird die Nutzung eines Softwarebereitstellungsangebotes durch ein gesamtes Unternehmen in den Bereichen Customer Relationship Management (CRM) und Enterprise Resource Planning (ERP) verstanden. Wichtig sind hohes Applikations- und Branchenwissen sowie Erfahrungen in der Softwareintegration. Im Vergleich zur kooperativen Nutzung sind weitreichende Dienstleistungen vorrangig in der Benutzerbetreuung und der organisatorischen Integration wichtig. Quantität und Qualität in der Bereitstellung sind auf hohem Niveau anzusiedeln.

7.2 Anwendung auf die Fallstudien

Die Anwendung der Organisationsperspektive verdeutlicht bei beiden Fallstudien, welche Akteure beteiligt waren und welche Aufgaben sie übernommen oder nicht übernommen haben. Darüber hinaus werden die Beziehungen zwischen den Akteuren sichtbar, und es können Aussagen über die teilweise bestandenen vertraglichen Regelungen und Kosten getroffen werden.

7.2.1 CommSy@Uni.de

Bei der Bereitstellung von CommSy durch Uni.de waren folgende Akteure, gruppiert nach den kollektiven Rollen, beteiligt:

- *Kunde*: Als Kunden wurden zunächst deutschsprachige (Fach-)Hochschulen und später die wissenschaftlichen Gemeinschaften in ver-

schiedenen europäischen Ländern anvisiert. Übergangsweise sollten Werbeagenturen die Bereitstellung mittels Werbebanner finanzieren.

- *Nutzer*: Nutzer waren Studierende und das wissenschaftliche Personal bzw. Schüler und Lehrer von Hochschulen, Fachhochschulen und Schulen in Deutschland.
- *Dienstleistungsanbieter*: Der Dienstleistungsanbieter war Uni.de.
- *Anwendungspartner*: Der Anwendungspartner war HITeC.
- *Hostingpartner*: Der Hostingpartner wechselte von einem privatwirtschaftlichen Rechenzentrum in München zu einem privatwirtschaftlichen Rechenzentrum in Frankreich.
- *Zugangspartner*: Der Zugangspartner war jeweils auch das entsprechende privatwirtschaftliche Rechenzentrum.
- *Supportpartner*: Die Rolle des Supportpartners sollte Uni.de mit Unterstützung von HITeC übernehmen. Faktisch wurde sie zunächst nur von HITeC eingenommen; nach ca. einem halben Jahr wurde sie nicht mehr ausgefüllt.
- *Marketingpartner*: Uni.de oblag die Rolle des Marketingpartners in der Bereitstellung von CommSy, nahm diese Rolle aber kaum wahr.
- *Hardwarezulieferer*: Wer die Hardwarezulieferung für den Betrieb übernommen hat, ist nicht bekannt. Die Hardware für die Entwicklung stellte der Fachbereich Informatik der Universität Hamburg.
- *Softwarezulieferer*: Die Open-Source-Community war der Softwarezulieferer, da die zusätzlich benötigte Software ausschließlich Open Source war.

Die vertraglich geregelte Kommunikation und Kooperation stellte sich in dieser Fallstudie wie folgt dar:

- *Kunde (Nutzer) – Dienstleistungsanbieter*: Uni.de stellte CommSy den deutschsprachigen (Fach-)Hochschulen gebührenfrei zur Verfügung, somit war eine vertragliche Regelung dieser Kooperation nicht notwendig.

- *Dienstleistungsanbieter – Hostingpartner und Zugangspartner*: Hinsichtlich dieser Kooperation wurden sicherlich Verträge von Uni.de mit dem entsprechenden Rechenzentrum vereinbart. Genaue Angaben über diesen Punkt können, auf Grund der insgesamt schlechten Kooperation, in dieser Fallstudie nicht angegeben werden.

- *Dienstleistungsanbieter – Anwendungspartner*: Uni.de und HITeC vereinbarten einen Kooperationsvertrag, der die ausschließliche Nutzung (Übertragung der Nutzungsrechte) von CommSy durch Uni.de für einen Zeitraum von mindestens anderthalb Jahren, mit der Ausnahme des „Eigenbedarfs“, vorsah.

- *Dienstleistungsanbieter – Supportpartner und Marketingpartner sowie Supportpartner – Hostingpartner, Zugangspartner und Marketingpartner*: Aufgrund der schlechten Kooperation kann über vertragliche Regelungen innerhalb Uni.de und zwischen Uni.de und den privatwirtschaftlichen Rechenzentren keine Aussage getroffen werden. Als HITeC die Rolle des Supportpartners einnahm, bestanden keine vertraglichen Regelungen.

- *Zugangspartner – Hostingpartner*: Wegen der schlechten Kooperation kann über vertragliche Regelungen innerhalb des professionellen Rechenzentrums keine Aussage getroffen werden.

- *Hostingpartner – Anwendungspartner*: Hier bestand keine vertragliche Regelung. Zeitweise war sogar unklar, wer Ansprechpartner beim Hostingpartner war.

- *Anwendungspartner – Supportpartner*: Zwischen HITeC und Uni.de bestand ein sechsmonatiger Beratungsvertrag. HITeC sollte Uni.de in die Lage versetzen, die Benutzerbetreuung leisten zu können. Wie oben dargestellt, lief diese vertragliche Regelung darauf hinaus, dass HITeC die Rolle des Supportpartners bis zum Ende des Beratungsvertrages ausfüllte.

- *Anwendungspartner – Marketingpartner*: Es bestand keine vertragliche Regelung.

- Die *Zulieferer – Partner – Verbindungen* sind für diesen Fall nicht von besonderem Interesse, so dass von einer detaillierten Betrachtung Abstand genommen wird.

Ziele der Hauptakteure Uni.de und HITeC waren in dieser Fallstudie:

- *Uni.de*: Uni.de beabsichtigte, mit ihren Webdienstleistungen auf der Domäne uni.de der deutschsprachigen wissenschaftlichen Gemeinschaft ein virtuelles Zuhause zu geben. Diese Dienstleistungen sollten für Nutzer gebührenfrei sein und mussten sich über Werbebanner refinanzieren. Mit der zusätzlichen Bereitstellung von CommSy sollten primär Wissenschaftler angesprochen werden, da Studierende bereits in ausreichendem Maße die Angebote von Uni.de nutzten. Insgesamt musste Uni.de als Wirtschaftsunternehmen natürlich rentabel arbeiten, so dass das Unternehmensziel die Gewinnerwirtschaftung war.

- *HITeC*: Als Technologietransferverein des Fachbereichs Informatik der Universität Hamburg ist HITeC generell an Kooperationen mit Wirtschaftsunternehmen interessiert. Als gemeinnütziger Verein hat HITeC nicht die Gewinnerwirtschaftung zum Ziel, sondern die Umsatzgenerierung. Hinsichtlich der Kooperation mit Uni.de bestanden die konkreten Ziele, CommSy weiter zu verbreiten und die Aufgaben des Betriebes und der Benutzerbetreuung abzugeben, um sich wieder ganz der Entwicklung von CommSy widmen zu können.

Es ist festzustellen, dass Uni.de, als zentraler Akteur im Akteursnetzwerk, im Gegensatz zu den anderen Akteuren mehrere Rollen eingenommen hat. Weiterhin ist festzustellen, dass hinsichtlich der Kooperationen der kollektiven Rollen kaum vertragliche Regelungen bestanden. Einzig zwischen HITeC und Uni.de bestanden zwei Verträge. Der erste Vertrag (Kooperationsvertrag) regelte die Überlassung der Nutzungsrechte vom Anwendungspartner (HITeC) zum Dienstleistungsanbieter (Uni.de) und die Weiterentwicklung von CommSy. Es wurden allerdings keine detaillierten Vereinbarungen in Form von SLAs ausgehandelt. Insbesondere wurden keine Punkte, im Sinne einer Kooperations-SLA, in den Vertrag aufgenommen. Der zweite Vertrag (Beratungsvertrag) regelte ähnlich unspezifisch die Beratung des Supportpartners (Uni.de) durch den Anwendungspartner (HITeC).

So ist es nicht verwunderlich, dass bei den aufgetretenen Problemen weder ein Problemmanagement aktiv wurde noch eine entsprechende Kommunikation einsetzte.

Das Finanzierungsmodell, über Bannerwerbung die Bereitstellung von CommSy zu finanzieren, war nicht tragfähig, da die Nutzenden von CommSy nicht genug Traffic erzeugten. So stand die Bereitstellung von CommSy bei Uni.de im Widerspruch zum Unternehmensziel der Gewinnerwirtschaftung. Als Ergebnis hat Uni.de sämtliche Aufgaben, die mit der Bereitstellung von CommSy verbunden waren und die bei der kooperativen Nutzung von CommSy in entsprechender Qualität und Quantität hätten übernommen werden müssen, vernachlässigt bzw. die entsprechenden Rollen nicht oder nur teilweise weiter ausfüllt. Das Ziel der Erhöhung des wissenschaftlichen Niveaus von `http://www.uni.de/` gab Uni.de damit aber nicht gänzlich auf, sondern die CommSy-Projekträume existierten unbetreut weiter. Dies führte dazu, dass Nutzende sich bei Problemen an HITeC wandten, was im Widerspruch zu dessen Ziel stand, die Aufgaben der Benutzerbetreuung und des Betriebes abzugeben. Da die schlechte Bereitstellung auf CommSy selbst zurückzufallen drohte, übernahm HITeC zunächst die Aufgaben der Benutzerbetreuung. Als HITeC darüber hinaus die Aufgaben des Betriebs hätte übernehmen müssen, zog sich HITeC resigniert aus der Kooperation zurück. So scheiterte die Kooperation trotz zweier vertraglichen Regelungen zwischen HITeC und Uni.de. Die Bereitstellung von CommSy bei Uni.de kann als quantitativ und qualitativ gering und insgesamt als unzureichend charakterisiert werden.

Trotz des gescheiteren Finanzierungsmodells gab es in diesem Bereitstellungskontext einen Zahlungsverkehr. Es wurde eine einmalige Gebühr für die Weiterentwicklung erhoben sowie wiederkehrende Gebühren für die Beratung in der Benutzerbetreuung über den Zeitraum von einem halben Jahr. Ob Uni.de Einnahmen aus den Werbebannern bezogen hat, ist unklar, da im Kooperationsvertrag die Beteiligung von HITeC an diesen Werbeeinnahmen mit 30% veranschlagt wurde, aber trotz mehrmaliger Nachfragen nie entsprechende Tantiemen geleistet wurden.

Die Erstellung einer detaillierten Aufstellung der Kosten der Bereitstellung von CommSy bei Uni.de ist mangels Einblick in dieses Bereitstellungsszenario schwierig. Insgesamt sind die Transaktionskosten als sehr hoch zu bewerten. Insbesondere die Zeit der Vertragsverhandlungen erstreckte sich über ein halbes Jahr. Auf Seiten der Anwendungsentwick-

ler wurden ca. vier Menschmonate von drei Personen eingesetzt. Nach der Vertragsunterzeichnung sanken die Transaktionskosten, bis sie dann in der Zeit kurz nach der Eröffnung des Angebots im April 2001 wegen aufkommender Irritationen wieder anstiegen (ca. ein Menschmonat). Aufgrund der darauf folgenden Resignation in der Kooperation und als Folge der nicht mehr stattfindenen Kommunikation tendierten die Transaktionskosten gegen Null. Der Aufwand für die Weiterentwicklung ist mit 46 Menschtagen und für den Beratungsvertrag mit 30 Menschtagen für HITeC zu berechnen. Darin sind allerdings noch nicht die Transaktionskosten, hinsichtlich der Erbringung der Dienstleistung innerhalb des Anwendungsentwicklers berücksichtigt. Diese lassen sich im Nachhinein leider nur noch schätzen: ca. zehn Menschtage für die Weiterentwicklung und fünf Menschtage für die Beratung.

Zusammenfassend scheiterte die Bereitstellung von CommSy in der Fallstudie CommSy@Uni.de an der mangelnden Übernahme der kollektiven Rollen Dienstleistungsanbieter, Support- und Marketingpartner und den damit verbundenen übergeordneten Aufgaben durch Uni.de. Die mangelnde Übernahme der Aufgaben lässt sich u. a. mit der nicht vorhandenen Refinanzierbarkeit, die im Widerspruch zur Gewinnerwirtschaftung steht, erklären. Undifferenzierte vertragliche Vereinbarungen gaben den beteiligten Akteuren keine Sicherheit für ihr Handeln. Fehlende vertragliche Regelungen sind ein Grund für die mangelnde Transparenz in der Kooperation aller Beteiligten und der damit verbundenen schlechten Kommunikation. Zudem haben HITeC und Uni.de zu wenig Engagement in der übergeordneten Aufgabe Kooperation gezeigt.

7.2.2 CommSy@WissPro

Bei der Bereitstellung von CommSy im Forschungs- und Entwicklungsprojekt WissPro waren folgende Akteure in den entsprechenden kollektiven Rollen beteiligt (vgl. Jackewitz 2004):

- *Kunde*: Der zahlende Kunde war in dieser Fallstudie das Bundesministerium für Bildung und Forschung (BMBF), welches das Projekt WissPro finanzierte. Als Kunden wurden auch Lehrveranstalter anvisiert, aber zu einer Bezahlung der Bereitstellungsleistungen durch Lehrveranstalter kam es nicht.

- *Nutzer*: CommSy wurde im Projekt WissPro überwiegend projektextern genutzt. Nutzer waren Wissenschaftler, Studierende, Lehrer, Schüler usw. an deutschsprachigen Bildungsinstitutionen.
- *Dienstleistungsanbieter*: Das Projekt WissPro hat CommSy im deutschen Sprachraum allen Interessenten gebührenfrei zur Verfügung gestellt.
- *Anwendungspartner*: Die Weiterentwicklung von CommSy war Teil des Projekts WissPro. Diese Rolle wurde im Projekt vom CommSy-Entwicklungsteam übernommen.
- *Hostingpartner*: Das Projekt WissPro betrieb den CommSy-Server, d. h. administrierte, konfigurierte und übernahm eine kontinuierliche Wartung. Ein Betriebsteam betreute den Betrieb.
- *Zugangspartner*: Der Zugangspartner zum Internet waren das Regionale Rechenzentrum der Universität Hamburg und das Rechenzentrum des Fachbereichs Informatik. Die Benutzer von CommSy waren über ihre Universitäten ans Internet angebunden bzw. haben private Zugänge benutzt.
- *Supportpartner*: Aus dem Projekt WissPro heraus wurde von einem Benutzerbetreuungsteam eine umfassende Betreuung angeboten.
- *Marketingpartner*: Das Projekt WissPro nahm diese Rolle ein. Es betrieb durch Publikationen ein überwiegend „wissenschaftliches“ Marketing. CommSy wurde u. a. auf verschiedenen Konferenzen und Messen präsentiert.
- *Hardwarezulieferer*: Aus Gründen der Schleichwerbung wird der einzige Hardwarezulieferer hier nicht namentlich benannt.
- *Softwarezulieferer*: Der CommSy-Server basiert auf freier Software, so dass die Open Source-Gemeinschaft diese Rolle einnahm.

Ziele des Hauptakteurs WissPro in dieser Fallstudie waren

- die Erprobung von didaktischen Lehr-/Lernkonzepten in Kombination mit der Software CommSy in vom Projekt begleiteten Lehrveranstaltungen,

- die Verbreitung von CommSy und
- die Verstetigung der CommSy-Entwicklung in einem Open Source Prozess.

Die Bereitstellung von CommSy im Projekt WissPro kann, im Verhältnis zur Software CommSy, als angemessen in Quantität und Qualität charakterisiert werden. Anfangs lagen Quantität und Qualität über den Erfordernissen. Sie wurden im Laufe der Zeit auf ein angemessenes Niveau zurückgestuft.

Vertragliche Regelungen hinsichtlich der Bereitstellung von CommSy bestanden in dieser Fallstudie nur indirekt. Das BMBF hat hinsichtlich des Projekts WissPro Vereinbarungen mit den am Projekt beteiligten Hochschulen geschlossen. Die Mitarbeiter des Projekts hatten Arbeitsverträge mit ihren Hochschulen. Weitere vertraglichen Regelungen zwischen den einzelnen kollektiven Rollen bestanden nicht. Später wurden Kommunikationsregeln und -kanäle im Projektverlauf etabliert, die vertraglichen Regelungen ohne schriftliche Fixierung im Sinne eines Vertrages entsprachen. Das waren:

- Wöchentliche Sitzungen zur Diskussion und Information über die Bereitstellung von CommSy.
- Workshops zu unregelmäßigen Zeitpunkten, die sich spezieller Themen im Detail annahmen. Bei den Themen handelte es sich überwiegend um Aspekte in der Weiterentwicklung von CommSy.
- Etablierung von Prozessen des Updates des CommSy-Servers, des Debuggings und des Releasemanagements.
- Einrichtung und Nutzung eines Bereichs zur Koordination der Bereitstellung und Entwicklung bei SourceForge.net. Zusätzlich wurde ein CommSy-Projektraum zur Kommunkation verwendet.

Bei einer Laufzeit von 34 Monaten und einer durchschnittlichen Beteiligung von sieben Personen sind die Transaktionskosten zur Koordination der CommSy-Bereitstellung (und Weiterentwicklung) mit zwölf Menschmonaten zu berechnen. Diese Zahl errechnet sich durch die Zeit der

regelmäßigen Koordinationstreffen und der durchgeführten Workshops, jeweils multipliziert mit den anwesenden Personen, wobei ein Menschmonat aus 20 Arbeitstagen à acht Arbeitsstunden besteht und im Jahr mit 260 Arbeitstagen gerechnet wurde.

Gebühren tauchten in dieser Fallstudie je nach Sichtweise einmalig oder wiederkehrend auf. Einmalig, da das BMBF dem Projekt entsprechende Gelder bewilligt hatte. Wiederkehrend, da die Mitarbeiter ihre Bezüge monatlich erhielten. Zwischen den kollektiven Rollen wurden keine Gebühren erhoben.

CommSy	**Benutzerkennungen**	**Projekträume**
CommSy-Campus	2574	237
WissPro CommSy	617	50
Permanent	782	16
Wintersemester 01/02	774	47
Sommersemester 02	262	19
Summe	**5009**	**369**

Tabelle 5: Nutzungszahlen in der Fallstudie CommSy@WissPro (16.10.2003)

Im Fall der Bereitstellung von CommSy im Projekt WissPro waren am 16.10.2003 insgesamt 5009 Benutzerkennungen und 369 CommSy-Projekträume auf dem CommSy-Server des Projektes verzeichnet. Der Aufwand zur Bereitstellung von CommSy hinsichtlich dieser Nutzungszahlen verteilte sich auf die übergeordneten Aufgaben in Personentagen pro Jahr wie in der Tabelle 6 angegeben. Die dort angegebenen Zahlen sind das Resultat aus einer Befragung der Mitarbeiter des Projekts, die Aufgaben in den entsprechenden übergeordneten Aufgaben wahrgenommen haben.

Zur Aufwandsabschätzung ist zu sagen, dass die Anwendungsentwicklung und der Betrieb unabhängig von den Nutzungszahlen sind, während die Benutzerbetreuung direkt mit ihnen zusammenhängt. Der Aufwand für die Zugriffsermöglichung ist in dieser Fallstudie nicht zu bemessen, da der CommSy-Server neben vielen anderen Servern über das Regionale Rechenzentrum der Universität Hamburg und das Rechenzentrum des Fachbereichs Informatik der Universität Hamburg mit dem Internet verbunden

war. Diese Zugriffsermöglichung wird pauschal abgegolten und lässt sich, hinsichtlich des Aufwands eines einzigen Servers, nicht differenzieren. Der Aufwand für das Marketing wurde während der Befragung nicht erhoben und kann nachträglich, aufgrund diverser Publikationen und einiger Beteiligungen an Messen und Konferenzen, nicht mehr geschätzt werden.

Übergeordnete Aufgabe	**Personentage pro Woche**	**Personentage pro Jahr**	**Stellen ca.**
Anwendungsentwicklung	13	676	3
Betrieb	0,25	13	0
Benutzerbetreuung	1,25	65	1/3
Marketing	?	?	?
Kundenbetreuung	0	0	0
Zugriffsermöglichung	?	?	?
Kooperation	5	260	1
Summe	**19,5**	**1014**	**4 1/3**

Tabelle 6: Aufwand der Bereitstellung von CommSy in der Fallstudie CommSy@WissPro

Die Bereitstellung von CommSy im Rahmen des Projekts WissPro war insgesamt erfolgreich. Anfängliche, kleine Kommunikationsprobleme sind u. a. auf die nicht vorhandenen Regelungen zwischen den kollektiven Rollen zurückzuführen. Nachdem feste, wenn auch nicht schriftlich fixierte Kommunikationsregeln etabliert waren, besserten sich Kommunikation und Kooperation. Die Finanzierung durch das BMBF gestaltete die Bereitstellung von CommSy im Rahmen des Projekts WissPro trotz „wissenschaftlichen“ Drucks und der zeitlichen Befristung zunächst entspannt, da das existenzielle Ziel der Gewinnerwirtschaftung nicht bestand. Zum Ende des Projekts baute sich allerdings durch die zeitliche Befristung ein Druck der Rentabilität auf.

7.3 Zwischenergebnis

Durch die Organisationsperspektive wird deutlich, welche Aufgaben welchen Rollen auf den verschiedenen Ebenen zuzuordnen sind. So lassen sich die Handlungen der beteiligten Akteure, unter Berücksichtigung ihrer unterschiedlichen Ziele und möglicher Konflikte, besser interpretieren.

Als kollektive Rollen wurden auf Dienstleisterseite der Dientsleistungsanbieter, Hosting-, Anwendungs-, Support-, Marketing- und Zugangspartner sowie Hardware- und Softwarezulieferer identifiziert und mit entsprechenden übergeordneten Aufgaben verknüpft, wobei die Aufgabe Kooperation von allen Rollen zu leisten ist. Auf Kundenseite stehen zahlender Kunde und Nutzer.

Beim Übergang vom abstrakten kollektiven Rollennetz zu konkreten Akteuren muss das Konfliktpotential bei der Kooperation von Akteuren einbezogen werden. Konflikte sind auf der Ebene von Akteuren meist Zielkonflikte. Es ist daher unumgänglich, die Ziele der verschiedenen beteiligten Akteure in den Blick zu nehmen. Dabei ist wichtig, zwischen den konkreten Zielen hinsichtlich der Kooperation und allgemeinen Akteurszielen zu unterscheiden.

Durch die Betrachtung der Softwarebereitstellung als Dienstleistung werden Dienstleistungen bzw. Services als (soziale) Beziehungen zwischen Servicegeber und Servicenehmer interpretiert. Dies bedeutet, dass ein Service dann als erfolgreich zu bewerten ist, wenn der Bedarf des Servicenehmers durch den Servicegeber befriedigt wurde. Da dies aus der Sicht eines Servicegebers bei mehreren Servicenehmern zu unterschiedlichen (Zeit-)Punkten eintritt, wird er aus wirtschaftlichen Gesichtspunkten seine Dienstleistungen in verschiedenen Qualitäten und Quantitäten anbieten. Dabei stehen die notwendigen Qualitäten und Quantitäten in Beziehung zur bereitgestellten Software bzw. der Nutzungssituation. Bei einer persönlichen Nutzung (Einzelnutzung) von z. B. Office-Anwendungen ist das Niveau in Qualität und Quantität niedrig, bei einer kooperativen Nutzung (Gruppennutzung) von z. B. Groupware höher und bei einer organisatorischen Nutzung (Unternehmensnutzung) von CRM- oder ERP-Systemen ist das Niveau auf hoher Ebene anzusiedeln.

Dienstleistungen müssen vertraglich geregelt werden, falls einer der an der Dienstleistung beteiligten Partner Rechtsansprüche gegen den anderen gelten machen will. Die Benutzung von Service Level Agreements

(SLAs) in den Bereichen Netzwerk, Betrieb, Anwendung, Betreuung und Kooperation macht die Leistungen in der Bereitstellung von Software in einer ausreichenden Detailtiefe transparent und klärt darüber hinaus Verantwortlichkeiten. So dienen vertragliche Regelungen nicht nur als Rechtssicherheit, sondern auch als Dokumentation der Dienstleistungsbeziehungen. Dabei werden Rahmenverträge zur grundsätzlichen Kooperation und konkrete, kurze Einzelverträge zur Ausdifferenzierung konkreter Aspekte vorgeschlagen (siehe auch Abschnitt 9.1.4).

Bei einer Dienstleistung steht der Servicegeber in der Pflicht, den Dienst zu leisten, und der Servicenehmer, den Servicegeber zu entlohnen. Dies führt zu Gebühren und Kosten einer Softwarebereitstellung. Gebühren können einmalig, regelmäßig oder nutzungsabhängig erhoben werden. Da die im gesamten Dienstleistungsnetzwerk einer Software in Rechnung gestellten Kosten letztendlich den Aufwand der einzelnen Akteure refinanzieren sollen, ist die Summe aller Gebühren eine gute Approximation der tatsächlichen Kosten der Softwarebereitstellung. Um die tatsächliche Höhe genau zu ermitteln, müssen die Transaktionskosten bestimmt und eingerechnet werden. Die genaue Aufschlüsselung der Kosten für einen Akteur liefert Hinweise zur Entscheidung, bestimmte Komponenten oder die gesamte Bereitstellung einer Software aus- oder einzugliedern.

In der Organisationsperspektive wurden Anforderungen an die Organisation und die zu leistenden Dienste sichtbar. Um die Anforderungen an die Technik in den Blick zu nehmen, wird im Folgenden die Technikperspektive eingenommen.

- 8 -
Technikperspektive – Was wird benötigt?

Für die Technikperspektive bietet ASP viel, so dass die aus den Fallstudien hervorgegangenen Fragen an die Technik (Abschnitt 3.4) beantwortet werden können. Daher lehnt sich der Softwarebereitstellungsansatz eASP, hinsichtlich technischer Aspekte, sehr nah an die Vorgaben aus der ASP-Literatur an (Abschnitt 4.5). Insofern ist dieses Kapitel kürzer gehalten als die Kapitel der anderen Perspektiven. Auf Textdoppelungen wurde verzichtet. Dennoch müssen technische Aspekte bei der Plattformunabhängigkeit (Abschnitt 4.5.2) und der technischen Infrastruktur (Abschnitt 4.5.3) abgewandelt werden, da der Softwarebereitstellungsansatz eASP im Gegensatz zu ASP auch innerhalb einer Organisation anwendbar sein soll (Kapitel 7).

8.1 Konzeption

Unter Heranziehung der ASP-Literatur über technische Aspekte (Abschnitt 4.5) werden im Folgenden für die Technikperspektive des Ansatzes eASP die technische Infrastruktur und Anforderungen an die bereitzustellende Anwendung skizziert.

8.1.1 Technische Infrastruktur

Für die Technikperspektive ist ausschlaggebend, dass die Software zentral für viele Nutzende bereitgestellt wird. Dies legt eine Client/Server-Architektur nahe, ohne an dieser Stelle die Diskussion um Thinclients zu ignorieren oder ältere Architekturen (Terminal-Host) auszuschließen.

Die in der ASP-Literatur diskutierte Gliederung der technischen Infrastruktur (Abschnitt 4.5.3) ist dem Grund geschuldet, Software und Daten vor unbefugtem Zugriff zu schützen und sie nur jenen anzubieten bzw. diejenigen zugreifen zu lassen, die zur Benutzung berechtigt sind. Dies gilt für ASP ebenso wie für den Fall eines größeren Unternehmens, das organisationsintern für viele Mitarbeiter eine Software zur Verfügung stellt. Das

bedeutet, dass die Gliederung der technischen Infrastruktur auch auf organisationsinterne Softwarebereitstellungen anzuwenden ist und die Bereiche Intern, DMZ und Extern für den Ansatz eASP im Gegensatz zum ASP wie folgt zu definieren sind (vgl. Abschnitt 4.5.3):

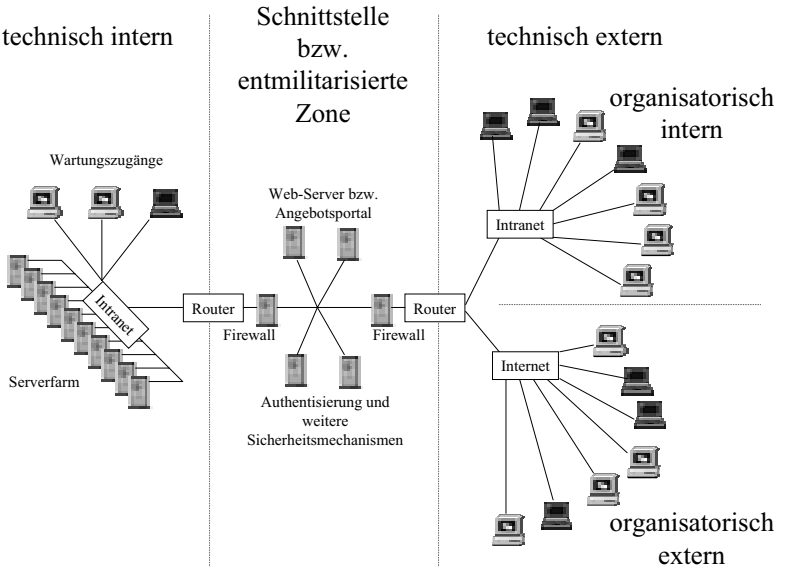

Abbildung 29: Technische Infrastruktur für eASP

- *Interner Bereich*: Der interne Bereich bezeichnet technisch die für die Softwarebereitstellung einer Software benötigte Hard- und Software sowie deren Vernetzung untereinander. Ein internes Netzwerk verbindet die verschiedenen Serverkomponenten (z. B. Anwendungsserver, Applikationsserver, Portalserver, Kommunikationsserver, DMS-Server, E-Security-Server, Billingserver). Es erlaubt dem Wartungspersonal, sich administrativ um die Softwarebereitstellung zu kümmern. Über die interne Vernetzung können keine Nutzenden auf die zentral bereitgestellten Anwendungen zugreifen. Organisatorisch entspricht der interne Bereich damit der kollektiven Rolle des Hosting-

partners, sei es nun eine IT-Abteilung in einem Unternehmen oder ein eigenständiger Dienstleister.

- *Entmilitarisierte Zone*: Die Schnittstelle zwischen internem und externem Bereich wird „entmilitarisierte Zone (demilitarised zone – DMZ)“ genannt. Bei einem Zugriff von außen greifen hier die verschiedenen Sicherheitsmechanismen (Authentisierungsserver, Firewall, VPN usw.).
- *Externer Bereich*: Der technisch externe Bereich bezeichnet die Verbindung vom entmilitarisierten Bereich zum Nutzer bzw. zum Endgerät (PC, Notebook, PDA, Handy usw.) des Nutzers.
    - *Organisatorisch intern*: Wenn Mitarbeiter des selben Unternehmens als Nutzer auf die bereitgestellte Anwendung zugreifen, sind sie technisch gesehen im externen Bereich, organisatorisch aber im internen Bereich. Sie greifen also über das Intranet des Unternehmens auf die (wahrscheinlich) vom Rechenzentrum des Unternehmens bereitgestellte Software zu.
    - *Organisatorisch extern*: Ist die Bereitstellung einer Software aus einem Unternehmen an einen professionellen Dienstleister ausgelagert, sind die Nutzer, aus Sicht des Dienstleisters, nicht nur technisch extern, sondern auch organisatorische extern. Sie greifen über das Internet auf die Anwendung beim Dienstleister zu.

Im Gegensatz zum ASP (vgl. Abschnitt 4.5.3) wird im Ansatz eASP zwischen einer technischen und einer organisatorischen Sicht auf die technische Infrastruktur unterschieden. Die Betrachtung und Konzipierung von organisatorisch intern und extern als technisch gleich, ermöglicht, vom Kunden aus gesehen, ein flexibles Outsourcing oder Insourcing der Softwarebereitstellung bzw. aus Sicht eines Dienstleistungsanbieters die flexible Ankopplung von Kunden an die eigene Infrastruktur.

Die hier vorgestellte technischen Infrastruktur für Softwarebereitstellungen stellt einen Möglichkeitsraum dar. Welche Server bzw. Serverkomponenten und welche Sicherheitsmechanismen zum Einsatz kommen und ob die Nutzenden organisatorisch intern oder extern angesiedelt sind, ist von Bereitstellungsszenario zu Bereitstellungsszenario verschieden.

8.1.2 Anwendungen

Die bereitzustellende Anwendung muss auf die zuvor dargestellte technische Infrastruktur aufsetzen. Hierbei ist nicht relevant, ob die Software als Web-Applikation oder im Sinne eines Server-based Computing angeboten wird (vgl. Anschnitt 4.5.1), sofern Standards eingehalten werden und nicht proprietäre Technik eingesetzt wird. Die Einhaltung von weltweiten Standards gewährleistet eine hohe Flexibilität und Freiheit in Bezug auf den Austausch von verschiedenen Komponenten bzw. der Verlagerung des zentralen Betriebs von extern nach intern oder umgekehrt.

Die Forderung nach einer Plattformunabhängigkeit hängt eng mit dem Ziel zusammen, auf technischer Ebene eine möglichst große Freiheit und Flexibilität zu erhalten. In der ASP-Literatur wird sie für die Clientseite vehement, für die Serverseite dagegen nicht gefordert (Abschnitt 4.5.2).

Eine Plattformunabhängigkeit auf Clientseite bedeutet, dass Hard- und Software auf der Clientseite von der Serverseite vollkommen unabhängig sind. Das bedeutet:

- Veränderungen auf Serverseite (z. B. Umstellung vom Webserver X auf den Webserver Y) haben keine Auswirkungen auf die Clientseite.
- Es können beliebige Hard- und (Basis-)Softwarekombinationen (Windows-PCs, Linux-PCs, Macs, Sun-Solaris usw.) auf der Clientseite existieren.
- Auf der Clientseite besteht jederzeit die Möglichkeit, Veränderungen an der Hard- und (Basis-)Software vorzunehmen (z. B. die Umstellung von Betriebssystem X zu Betriebssystem Y).

Im Sinne der oben beschriebenen Flexibilität und Freiheit und im Gegensatz zur ASP-Literatur muss die Plattformunabhängigkeit auch auf Serverseite gewährleistet werden. Nur dann ist der Austausch von verschiedenen Serverplattformen möglich, was wiederum die flexible Aus- oder Eingliederung des Betriebs der Software in einer Organisation technisch ermöglicht. Dies ist insbesondere dann für Kunden wichtig, wenn der Dienstleistungsanbieter den Betrieb der Software nicht länger garantieren kann. Durch die Plattformunabhängigkeit auf Serverseite wird die Suche nach einem neuen Dienstleister erleichtert bzw. der Kunde kann, bis er einen neuen

Dienstleistungsanbieter gefunden hat, selbst die Software betreiben und die entsprechenden Dienstleisterrollen einnehmen.

Hinsichtlich der Mehrbenutzer- und Mehrmandantenfähigkeit gelten die in der ASP-Literatur beschriebenen Aspekte auch für eASP (siehe Abschnitt 4.5.2).

8.2 Anwendung auf die Fallstudien

In der Technikperspektive sind viele Gemeinsamkeiten in den Fallstudien zu erkennen, da in beiden Fallstudien CommSy bereitgestellt wurde. Zunächst wird die in beiden Studien gemeinsame Anwendung CommSy betrachtet und anschließend auf die Unterschiede in der technischen Infrastruktur eingegangen.

Wie in Abschnitt 3.2 beschrieben, ist CommSy eine webbasierte Client/Server-Anwendung und setzt damit auf den zurzeit vorherrschenden Standards (Client/Server-Architektur und Internettechnologien) auf. Da für die Benutzung von CommSy nur ein zu W3C-Standards kompatibler Webbrowser benötigt wird, ist clientseitig die Plattformunabhängigkeit gegeben.

Serverseitig werden für CommSy der Webserver Apache, die Skriptsprache PHP und die Datenbank MySQL benötigt. Unter dem Betriebssystem Linux läuft diese Kombination problemlos. Unter Windows treten dagegen Probleme auf, denn PHP konnte unter Windows (bisher) nicht als Modul in den Webserver Apache geladen, sondern musste als CGI-Variante benutzt werden. Bei anderen Webservern, wie dem Internet Information Server von Microsoft, sind ähnliche Probleme festgestellt worden. Zudem ist die Wahl der Datenbank fest vorgegeben. CommSy kann trotz ausschließlicher Nutzung von Open Source-Software nicht als serverseitig plattformunabhängig angesehen werden, da es nur unter verschiedenen Linux-Varianten (z. B. Red Hat, GenToo, Debian) auf Intel-PCs sowie auf Sparcs-Solaris-Kombinationen (Unix) stabil betrieben werden kann.

CommSy ist mehrbenutzerfähig, da der Zugriff auf die in CommSy enthaltenen Daten über einen Authentisierungsmechanismus gesichert ist. Benutzer können nur die für sie relevanten Daten einsehen und nur selbst eingestellte Daten ändern oder löschen.

Eine Mehrmandantenfähigkeit ist in CommSy ansatzweise gegeben. Mandanten können, in den verschiedenen bei Uni.de und in WissPro einge-

setzten Versionen, CommSy (d. h. CommSy-Projekträume bei Uni.de und CommSy-Gemeinschaftsraum und -Projekträume bei WissPro) hinsichtlich Namensgebung, Farbgebung und Reihenfolge der Rubriken anpassen. Es ist nicht möglich, das Aussehen (Oberfläche) von einem Projektraum bzw. einem CommSy (Gemeinschaftsraum und Projekträume) insgesamt zu verändern. Die Mehrmandantenfähigkeit ist bei CommSy also eingeschränkt.

Bei der technischen Infrastruktur lassen sich Unterschiede in den Fallstudien erkennen, auf die im Folgenden getrennt eingegangen wird.

8.2.1 CommSy@Uni.de

Der CommSy-Server für die Bereitstellung von CommSy in Kooperation mit Uni.de befand sich zunächst in einem Rechenzentrum in München und später in einem in Frankreich. Alle Nutzenden griffen ausschließlich über das Internet auf CommSy zu. Eine mit VPN gesicherte Verbindung bestand nicht. Ob Firewalls und eine entsprechend gesicherte DMZ vorhanden waren, kann nicht belegt werden.

Im internen Bereich der Rechenzentren müssen ein Webserver und ein Datenbankserver vorhanden gewesen sein. Auf Seiten von Uni.de war zusätzlich ein Portalserver installiert, der CommSy in ihr Angebot eingebunden hat. Speziell für CommSy war kein Billingserver bereitgestellt, da die Finanzierung von CommSy über Werbebanner gesichert werden sollte und nicht über Abrechnungen der einzelnen Nutzenden.

8.2.2 CommSy@WissPro

Im Forschungs- und Entwicklungsprojekt WissPro wurde ein CommSy-Server unter Red Hat-Linux 7.x und später Red Hat Fedora betrieben. Zum Einsatz kamen der Apache Webserver 1.3.x/2.0.x, der Skriptinterpreter PHP 4.2.x/4.3.x und die Datenbank MySQL 3.23.x.

Der Server war über das Intranet des Fachbereichs Informatik der Universität Hamburg mit dem deutschen Forschungsnetz verbunden, welches wiederum mit dem Internet in Verbindung stand. Das Rechenzentrum des Fachbereichs Informatik handhabte den Zugang von außen sehr rigide, u. a. war eine Firewall mit entsprechend strengen Regeln bzw. Filtern installiert. Insofern gelangten die fachbereichsexternen Nutzer, die ausschließlich über das Internet auf CommSy zugriffen, nur durch eine beim Rechenzentrum

des Fachbereichs Informatik angesiedelte DMZ zum CommSy-Server (interner Bereich).

Fachbereichsinterne Personen traten direkt über das Intranet des Fachbereichs mit dem CommSy-Server in Verbindung, mussten also nicht durch die DMZ. Insofern war der CommSy-Server im fachbereichsinternen Netz nicht durch eine Firewall geschützt. Die Zugriffskontrolle auf die gespeicherten Daten leistete CommSy, so dass aus einem internen nicht automatisch ein unerlaubter Zugriff wurde. Dennoch verstößt die beschriebene technische Infrastruktur an dieser Stelle gegen die Vorstellungen des Ansatzes eASP.

Die in CommSy abgelegten Daten wurden alle 24 Stunden durch ein Backup gesichert. Darüber hinaus war die Stromzufuhr des Servers durch eine unterbrechungsfreie Stromversorgung (USV) gesichert.

8.3 Zwischenergebnis

Die Betrachtung der Technik offenbart Anforderungen an die bereitzustellende Anwendung und an eine geeignete technische Infrastruktur.

Die zentral für viele Nutzende bereitgestellte Software muss mehrbenutzer- und mandantenfähig sein. Nutzende sollten, trotz der gemeinsam genutzten Installation, nur eigene Daten abrufen können und Mandanten müssen die Anwendung hinsichtlich ihrer Bedürfnisse und des Look and Feels anpassen dürfen.

Aufgrund des einheitlichen, zentralen Betriebs für viele Nutzende bieten sich Client/Server-Strukturen an. Dabei muss auf Standards aufgesetzt werden, da diese eine client- und serverseitige Plattformunabhängigkeit gewährleisten. Clientseitig, damit für Nutzende durch ihre technische Infrastruktur keine technischen Barrieren bestehen. Serverseitig, damit in einer Organisation eine flexible Aus- oder Eingliederung des zentralen Betriebs der Software technisch möglich wird. In der technischen Infrastruktur muss sich die geforderte Plattformunabhängigkeit auf der Client- und auf der Serverseite widerspiegeln. Es ist zwischen einem technisch internen und einem technisch externen Bereich sowie einer entmilitarisierten Zone (DMZ) zu unterscheiden. Der technisch externe Bereich muss weiter getrennt werden in einen organisatorisch internen und einen organisatorisch externen Bereich, um auch organisationsinterne Bereitstellungsszenarien mit eASP betrachten zu können.

Die technische Perspektive kann Aussagen über die Art der bereitzustellenden Software, deren Beschaffenheit und die geeignete technische Infrastruktur machen. Wie in den vorherigen Perspektiven fehlt die Zeitlichkeit, durch die eine Prozesssicht auf die Perspektiven eingenommen werden kann. Die Einnahme einer Prozesssicht erfolgt im nächsten Kapitel.

- 9 -
Vorgehen – Wie passiert es?

In der technischen Perspektive konnten Aussagen über die Art der bereitzustellenden Software, deren Beschaffenheit und die geeignete technische Infrastruktur gemacht werden. In der Aufgabenperspektive wurden alle notwendigen Aufgaben sichtbar und in der Organisationsperspektive stand die Organisation der Bereitstellung im Vordergrund. Es fehlt eine zeitliche Perspektive, in der eine Prozesssicht auf die drei bisher vorgestellten Perspektiven eingenommen wird. Die Prozesssicht ermöglicht zum einen Entwicklungen über die Zeit deutlich werden zu lassen, zum anderen gestaltend auf die Bereitstellung einzuwirken. Im folgenden Kapitel wird die Vorgangsperspektive eingenommen, die sich durch Gestaltungsmöglichkeiten von einer passiven Perspektive zu einer aktiven Handlung (vgl. Abschnitt 3.4) wandelt und daher mit Vorgehen betitelt wird.

In diesem Kapitel werden die Schwächen von ASP, mit Zeitlichkeit umzugehen, diskutiert. Zeitlichkeit spiegelt sich bei ASP in den, bis ins Kleinste auszudifferenzierenden SLAs wider, so dass alle Eventualitäten, die über die Zeit auftauchen könnten, abgedeckt sind. Diese Strategie verlagert den Umgang mit antizipierten Veränderungen auf den Anfang einer Dienstleistungsbeziehung zwischen Kunde und Dienstleister und führt dort zu erhöhten Transaktionskosten. Die Vorgehensweise, zu Beginn möglichst viel vorwegzunehmen und festzuschreiben, entspricht der bereits in Abschnitt 1.1 kritisierten Sichtweise der Informatik als Ingenieurswissenschaft. Im Folgenden wird das zyklische Projektmodell des Methodenrahmens STEPS (Abschnitt 5.2.1) auf die Bereitstellung einer Software adaptiert. Dem Ansatz eASP wird damit eine Vorgehensweise zur Verfügung gestellt, die nicht versucht alle möglichen Veränderungen im Vorwege zu definieren, sondern auf Veränderungen im Prozess zu dem Zeitpunkt ihrer Entstehung bzw. Aktualität eingeht.

9.1 Konzeption

Zur Motivation einer zyklischen Vorgehensweise werden zunächst Einflüsse auf die Bereitstellung einer Software skizziert:

- Der *gesellschaftliche Wandel* gibt Rahmenbedingungen vor, die Einfluss auf die Softwarebereitstellung haben. In einem Moment ist es en vogue, die Bereitstellung von Software auf externe Dienstleister zu übertragen, im nächsten Moment entpuppt sich diese Verhaltensweise als vorübergehender Trend. Terroranschläge erschüttern das Vertrauen in Kooperationspartner und in die Sicherheit von Softwaresystemen und führen zu strategischen Neuausrichtungen auch im Bereich der Softwarebereitstellung. Steuerreformen, Arbeitsmarktpolitik, das Ende einer Legislaturperiode und (Neu-)Wahlen verändern die gesellschaftlichen Rahmenbedingungen.

- Die *Technikentwicklung* gewinnt im Bereich der Informations- und Kommunikationstechnologie immer mehr an Geschwindigkeit. Kleinere, aber leistungsfähigere Hardware erschließt im Bereich der Softwarebereitstellung neue Möglichkeiten. Neuere Versionen von Softwareprodukten erweitern den Funktionsumfang, neuartige Software erweitert das Portfolio. Mit steigenden Möglichkeiten erhöhen sich auch die Bedürfnisse.

- *Wirtschaftliche Rahmenbedingungen* beeinflussen z. B. Investitionen in die Bereitstellung von Software. Schlechte (welt-)wirtschaftliche Konjunkturdaten oder steigende Aktienkurse lassen Akteure ihr Engagement hinsichtlich der Bereitstellung oder Nutzung von Software überdenken.

Diese Einflüsse wirken sich auf ein konkretes Bereitstellungsszenarios folgendermaßen aus:

- *Technik*: Die Performance der Anbindung oder der serverseitigen Rechenleistung steigt. Die bereitgestellte Software ändert sich im Laufe der Bereitstellung durch das Einpflegen von Softwareupdates. Zusätzliche Funktionen stehen zur Verfügung, andere verändern sich oder verschwinden.

- *Aufgaben*: Neue Aufgaben entstehen oder verändern sich. Sie müssen oder können in anderen Qualitäten nachgefragt und durchgeführt werden. Aufgaben verschieben sich eventuell zwischen funktionellen Rollen.

- *Dienstleistungen*: Durch neue oder veränderte Aufgaben entstehen neue oder veränderte Dienstleistungen, die dementsprechend angeboten und abgerechnet werden müssen. Kooperationsverträge zwischen den beteiligten Akteuren bzw. kollektiven Rollen müssen ergänzt oder neu verhandelt werden.

- *Akteurskonstellation*: Die Zuordnung der kollektiven Rollen zu Akteuren bzw. das Akteursnetzwerk verändert sich durch das Ausscheiden, Zurückziehen, Auseinanderfallen oder Verdrängen von Akteuren.

Der mögliche Einfluss der oben dargestellten Rahmenbedingungen führt ausschließlich über die beteiligten Akteure zu den genannten Veränderungen. Die beteiligten Akteure sind es, die selbst Änderungsprozessen unterliegen und sich aus Softwarebereitstellungskonstellationen zurückziehen oder neu einbringen. Sie sind diejenigen, die neue Hardware erfinden, neue Software programmieren, existierende Software weiterentwickeln und entscheiden, diese neuen Techniken für das Bereitstellungsszenario einzusetzen. Die beteiligten Akteure übernehmen neue Aufgaben, bieten diese als Dienstleistungen an und wollen sie entsprechend neu oder verändert vergütet sehen. d. h. Einflüsse auf die Softwarebereitstellung wirken sich immer (vermittelt) über die beteiligten Akteure aus.

Um mit diesen Einflüssen und Veränderungen in der Bereitstellung einer Software umgehen zu können, wird für eASP im Folgenden das zyklische Projektmodell des Methodenrahmens STEPS (Abschnitt 5.2.1) auf die Bereitstellung einer Software übertragen. Anschließend wird gesondert auf Gestaltungsmöglichkeiten beteiligter Akteure und kontinuierliche Projekttreffen eingegangen. Anschließend wird dargestellt, welche Konsequenzen eine zyklische Vorgehensweise auf vertraglichen Regelungen zwischen den Akteuren bzw. zwischen kollektiven Rollen hat.

9.1.1 Zyklische Vorgehensweise

In diesem Abschnitt wird eine zyklische Vorgehenswiese präsentiert. Die Softwarebereitstellung wird als eine Folge von Zyklen begriffen, die beliebig oft, aber endlich durchlaufen werden (vgl. Bleek und Jackewitz 2004; Trieneken u. a. 2004). Als Grundlage dient das zyklische Projektmodell von STEPS. Die vier Hauptelemente des zyklischen Vorgehens: Etablierung, Herstellung, Systemversion und Einsatz (Abschnitt 5.2.2) werden nun auf die Softwarebereitstellung übertragen.

- *Etablierung*: Die Etablierung eines neuen Zyklus in der Bereitstellung einer Software stellt den Start des neuen Zyklus dar. Initiiert wird er durch die beteiligten Akteure, welche die vorherrschende Bereitstellung (an bestimmten Stellen) als unbefriedigend empfinden. Sie nehmen den für sie (an bestimmten Stellen) instabilen Bereitstellungsrahmen (s.u.) zum Anlass, einen neuen Zyklus in der Softwarebereitstellung zu etablieren. Er wird mit der Herstellung eines neuen, wieder stabilen Bereitstellungsrahmens fortgeführt.

- *Herstellung*: Alle an der Softwarebereitstellung beteiligten oder von ihr betroffenen Akteure tragen in ihren kollektiven Rollen zur Gestaltung des Bereitstellungsrahmens bei. Das bedeutet nicht nur die zentrale oder dezentrale Verhandlung von Dienstleistungsverträgen, mit anschließender Niederschrift, sondern auch die Schaffung der Voraussetzungen zur Erfüllung der vereinbarten Dienstleistungen innerhalb der beteiligten Akteure. Die Herstellung des Bereitstellungsrahmens beinhaltet die Gestaltung eines Dienstleistungsnetzwerks respektive Vertragsnetzwerks und die Vorbereitung des Bereitstellungskontextes.

  Der gesamte Bereitstellungsrahmen muss verhandelt werden, wenn sich eine Softwarebereitstellung neu gründet oder gravierende Änderungen stattfinden, wenn z. B. der Akteur in der zentralen Rolle des Dienstleistungsanbieters sich aus dem Netzwerk löst. Generell muss nicht der gesamte Bereitstellungsrahmen zur Diskussion gestellt, sondern nur bestimmte Ausschnitte müssen neu gestaltet werden, wenn beispielsweise eine neue Softwareversion die alte ablösen soll oder ein Akteursaustausch beim Hostingparter stattgefunden

hat. Bereiche, die von diesen Änderungen nicht direkt betroffen sind, müssen sie nur zur Kenntnis nehmen.

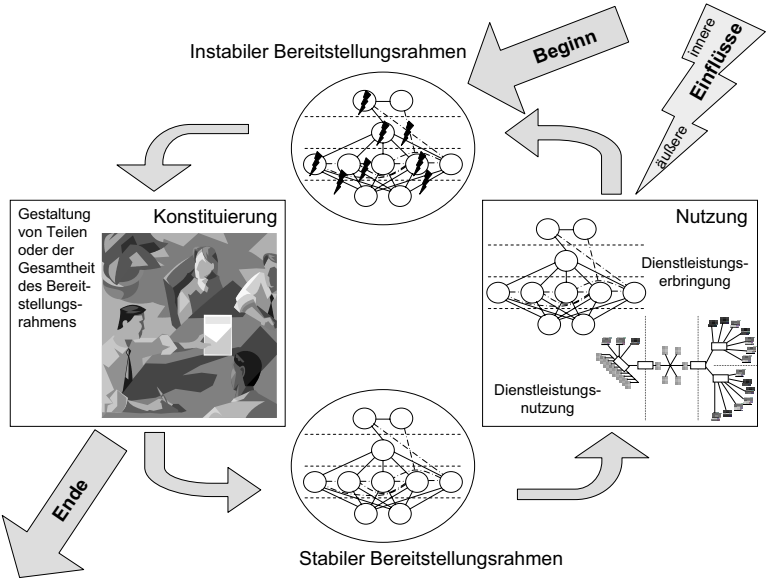

Abbildung 30: Zyklisches Verständnis der Softwarebereitstellung

- *Bereitstellungsrahmen*[1]: Ergebnis der Herstellung ist ein an die aktuellen Bedürfnisse und Bedingungen angepasster Bereitstellungsrahmen, der die Vorbereitung des Kontextes, insbesondere innerhalb der Akteure, und die rechtliche Absicherung durch Dienstleistungsverträge mit einschließt. In diesem Sinne ist er in dieser Situation als stabil zu bezeichnen. Der gestaltete Bereitstellungsrahmen ist Grundlage und Voraussetzung für den Einsatz, in dem er sich bewähren muss. Außerdem stellt er eine Momentaufnahme der in dieser Situation angemessenen Organisation der Softwarebereitstellung dar.

  Der Bereitstellungsrahmen setzt sich aus folgenden Komponenten zusammen:

1 Der Bereitstellungsrahmen entspricht im zyklischen Projektmodell von STEPS der Systemversion.

- *Kontext*: Zum Kontext gehört das Akteursnetzwerk mit seinen Verbindungen zwischen den beteiligten Akteuren, welche sich nach Vorgabe des kollektiven Rollennetzes im Akteursnetzwerk zusammenfinden, sowie die interne, auf die Dienstleistungserbringung und -nutzung ausgerichtete Organisation der Akteure selbst. Der Kontext bezeichnet die reale Situation.
- *Dokumentation*: Die Dokumentation ist ein Abbild des Kontextes zu einem bestimmten Zeitpunkt. Zur Dokumentation des Kontextes gehören die zwischen den Akteuren getroffenen Vereinbarungen, die oft in schriftlicher Form als Verträge niedergelegt werden.

In diesem Sinne ist ein Bereitstellungsrahmen als stabil zu bezeichnen, wenn Kontext und Dokumentation einander entsprechen, und als instabil, wenn Kontext und Dokumentation nicht mehr zueinander passen.

- *Einsatz*: Das Anbieten von Dienstleistungen aller Akteure in ihren kollektiven Rollen sowie das Nutzen der angebotenen Dienstleistungen durch entsprechende Akteure bringen den Bereitstellungsrahmen „zum Einsatz“. Hierbei ist im Nutzen der angebotenen Dienstleistungen die Nutzung der bereitgestellten Software inbegriffen.

  Im Laufe der Softwarebereitstellung, d. h. beim Einsatz des Bereitstellungsrahmens, wirken innere und äußere Einflüsse auf die Bereitstellung ein. Sie wirken nicht direkt, sondern vermittelt über die beteiligten Akteure. Diese Einflüsse bewirken Veränderungen in den Akteuren, unabhängig ob sie von außen (z. B. durch Steuererhöhungen oder -vergünstigungen) oder von innen (z. B. durch Einstellung neuer Mitarbeiter) motiviert werden. Diese Veränderungen beeinflussen die Softwarebereitstellung und führen zu einer, aus Sicht bestimmter Akteure, unbefriedigenden Bereitstellung. Der vormals vereinbarte, stabile Bereitstellungsrahmen wird instabil. Die mit der Bereitstellung unzufriedenen Akteure nehmen dies zum Anlass, einen neuen Zyklus zu initiieren und die Herstellung eines neuen, wieder stabilen Bereitstellungsrahmens zu etablieren.

  Die Herstellung und der Einsatz sind nicht als zeitlich separiert zu betrachten, auch wenn die Darstellung in Abbildung 30 und die lineare

Textdarstellung auf diesen Seiten dies suggeriert. Es ist vielmehr so, dass die eigentliche Bereitstellung im Abschnitt der Herstellung fortgeführt wird, allerdings instabil, d. h. weniger befriedigend.

Anfang und Ende einer Softwarebereitstellung sind unterschiedlich aus der Sicht eines beteiligten Akteurs und aus einer akteursübergreifenden Sicht auf die Softwarebereitstellung:

- *Akteur*: Für einen Akteur stellt sein Eintritt in das Bereitstellungsnetzwerk den *Beginn* für die Softwarebereitstellung dar. Er ist mit seinem Wunsch, sich an der Bereitstellung zu beteiligen, sei es als Dienstleister oder als Kunde, ein äußerer Einfluss, der zur Instabilität des Bereitstellungsrahmens beiträgt. Der Akteur ist somit ein Teil der Instabilität und Anlass, einen neuen Zyklus zu etablieren. In der nachfolgenden Herstellung muss mit allen beteiligten Akteuren, also auch mit ihm, der Bereitstellungsrahmen an die neue Situation angepasst werden.

  Ein beteiligter Akteur betrachtet die Bereitstellung einer Software als *beendet*, wenn er sich als Kunde aus der Nutzung oder als Dienstleister aus der Bereitstellung zurückzieht bzw. von anderen Akteuren ausgeschlossen wird. Der Rückzug bzw. Ausschluss führt zu einer instabilen Bereitstellung. In der folgenden Phase der Herstellung eines neuen Bereitstellungsrahmens ist der Akteur dann nicht mehr im Bereitstellungsrahmen existent (*formales Ende*) und wurde ggf. durch einen anderen ersetzt.

- *Akteursübergreifende Sicht*: Aus einer „Metasicht“ stellt die erste Etablierung des ersten Zyklus, welcher zur ersten Herstellung eines ersten Bereitstellungsrahmens führt, den *Beginn* der Bereitstellung dar.

  Das *Ende* der Bereitstellung tritt ein, wenn es, aufgrund eines instabilen Zustands, kein weiteres etablierendes Moment mehr gibt, d. h. wenn kein Akteur einen neuen Zyklus mittels einer Etablierung initiiert. Das Akteursnetzwerk zerfällt, alle Akteure geben ihre Rollen im kollektiven Rollennetz auf. Ein Grund für das Ende der Bereitstellung und den endgültigen Zerfall des Akteursnetzwerks ist gegeben, wenn z. B. alle Kunden die Bereitstellung gekündigt haben oder der

Akteur, der die Rolle des Dienstleistungsanbieters einnahm, das Bereitstellungsnetzwerk verlässt und niemand seine Rolle übernimmt.

Zur Illustration der Bereitstellungszyklen werden nun drei Szenarien präsentiert, die die Softwarebereitstellung von CommSy aus Sicht eines Kunden, eines Marketingpartners und eines Dienstleistungsanbieters darstellen.

- *Kunde*: Eine Universität entschließt sich, ihre Präsenzlehre mit Neuen Medien zu unterstützen. Dazu muss sie (neben anderen Dingen) eine entsprechende Infrastruktur für Lehrende und Lernende bereitstellen. Im Zuge dessen entschließt sich die Universität, CommSy einzusetzen. Da sie nicht über das Wissen zur Bereitstellung eines CommSy-Servers verfügt, setzt sie sich mit dem Dienstleistungsanbieter CommSy e.V. in Verbindung [Beginn: Der Akteur tritt in den Bereitstellungskontext ein und macht den Bereitstellungsrahmen instabil.]. Dies wird zum Anlass genommen [Etablierung], einen Kooperationsvertrag zur Bereitstellung von CommSy zu vereinbaren [Herstellung: Die Universität wird in das Akteursnetzwerk als Kunde aufgenommen.]. Nach Vertragsunterzeichnung wird für die Universität ein CommSy eingerichtet und Lehrende sowie Lernende können CommSy nutzen [Einsatz: Die Universität nimmt die Dienstleistung des Dienstleistungsanbieters und die Nutzenden z. B. die des Supportpartners in Anspruch.].

  Zwei Jahre später und nach verschiedenen Zyklen, welche die Universität als Kunden nicht direkt betroffen haben und deren Änderungen in der Bereitstellung sie nur zur Kenntnis nahm, ist die Nutzerzahl von CommSy an der Universität derart gestiegen, dass die vereinbarten Kapazitäten hinsichtlich der CommSy-Bereitstellung nicht mehr ausreichen [Einsatz: Die Nutzung wird für die Universität unbefriedigend, der Bereitstellungsrahmen wird instabil]. So verhandelt CommSy e.V. als Dienstleistungsanbieter mit der Universität über neue Vereinbarungen und parallel dazu mit dem Hostingpartner, um die gestiegenen Bedürfnisse zu befriedigen [Herstellung]. Ergebnis sind zwei geänderte Dienstleistungsverträge und die Anpassung beim Hostingpartner hinsichtlich der zusätzlich benötigten Kapazitäten [Herstellung: stabilisierter Bereitstellungsrahmen].

Nach weiteren zwei Jahren entschließt sich die derzeitige Landesregierung, die Universität aus Kosteneinsparungsgründen zu schließen [Einsatz: äußerer Einfluss]. Die Universität muss die Verträge mit dem Dienstleistungsanbieter kündigen und zieht sich aus der Nutzung von CommSy zurück [Ende]. Da die Universität der größte Kunde mit den höchsten Nutzerzahlen war, liegen jetzt große Kapazitäten brach [Einsatz: instabiler Bereitstellungsrahmen]. Im Zuge dessen [Etablierung] verhandelt der Dienstleistungsanbieter mit seinen Partnern neue Verträge aus [Herstellung]. Ergebnis, es werden verschiedene Dienstleistungsverträge mit kleineren Kapazitäten unterschieben, bei den Partnern setzt entsprechender Stellenabbau ein [Herstellung: stabiler Bereitstellungsrahmen] und die Universität ist in dem nun wieder stabilen Bereitstellungsrahmen nicht mehr vorhanden [formales Ende].

- *Markteingpartner*: CommSy e.V. als Dienstleister tritt an eine Marketingagentur heran, um die Bereitstellung von CommSy zu fördern [Einsatz: instabiler Bereitstellungsrahmen, da CommSy e.V. mit der Bereitstellung nicht zu frieden ist; Beginn/Etablierung: Ein neuer Akteur soll ins Akteursnetzwerk aufgenommen werden.]. Ergebnis der anschließenden Verhandlungen [Herstellung] ist, dass die Marketingagentur im kommenden halben Jahr bei sämtlichen Hochschulen im deutschsprachigen Raum auf verschiedene Art und Weise die CommSy-Bereitstellung bekannt machen soll. Entsprechend stellt die Marketingagentur einen Mitarbeiter für diese Aufgabe ein [Herstellung: stabiler Bereitstellungsrahmen].

  Die Marketingagentur startet verschiedene Werbefeldzüge [Einsatz], die aber nicht zu der erwarteten Menge an Neukunden führt [Einsatz: instabiler Bereitstellungsrahmen]. Die Vergabe der Marketingaufgaben an die Marketingagentur wird von CommSy e.V. als Fehler angesehen. Die Verträge mit der Agentur werden nicht verlängert [Ende für die Marketingagentur] und die Aufgaben wieder von CommSy e.V. übernommen [Herstellung: stabiler Bereitstellungsrahmen; formales Ende: Die Marketingagentur wurde aus dem Bereitstellungsnetzwerk entfernt.].

- *Dienstleistungsanbieter*: In dem Forschungsprojekt ProWiss wurde anderen Hochschulen CommSy zur Verfügung gestellt. Nach Ende des Projekts [Einsatz: instabiler Bereitstellungsrahmen] übernahm der Verein CommSy e.V. die kollektive Rolle des Dienstleistungsanbieters [Herstellung: stabiler Bereitstellungsrahmen].

  Fortan leitete CommSy e.V. als zentraler Akteur die Bereitstellung: siehe obige Beispiele [Durchlaufen von Zyklen].

  Durch kontinuierlich steigende Kundenzahlen kann CommSy e.V. in seiner Rechtsform als „gemeinnütziger Verein“ der zunehmenden Bedeutung als Dienstleistungsunternehmen nicht mehr gerecht werden [Einsatz: instabiler Bereitstellungsrahmen]. Aus dem Verein heraus wird eine GmbH als Spin-Off gegründet [Beginn: für die GmbH], die fortan die Bereitstellung von CommSy leisten bzw. als Dienstleistungsanbieter koordinieren soll [Herstellung]. CommSy e.V. zieht sich auf die Rolle des Anwendungspartners zurück, um die Open Source Weiterentwicklung von CommSy zu sichern [Herstellung: stabiler Bereitstellungsrahmen].

9.1.2 Gestaltungsmöglichkeiten

Das Verständnis der Softwarebereitstellung als eine Folge von Zyklen bzw. die Darstellung dessen im vorigen Abschnitt deuten die Gestaltungsmöglichkeiten auf die Softwarebereitstellung an. Zunächst ist festzustellen, dass nur die beteiligten Akteure direkten Einfluss auf die Softwarebereitstellung und damit Gestaltungsmacht haben. Des Weiteren haben sie diese Macht zu jedem Zeitpunkt und können die Bereitstellung jederzeit durch ihr Handeln bestätigen oder ihm zuwider handeln, welches ein Nicht-Handeln einschließt (vgl. Giddens 1997; Ortmann u. a. 1997; Orlikowski 1992; Pape 2004). In diesem Sinne sind immer alle Akteure zu jedem Zeitpunkt an der Gestaltung der Softwarebereitstellung beteiligt, und da sie selbst äußeren und inneren Einflüssen (siehe Einleitung zu Abschnitt 9.1) unterliegen, werden sie den vereinbarten Bereitstellungsrahmen in ihrem Handeln nicht immer bestätigen können. Zwangsläufig führt dies zu Veränderungen in der Softwarebereitstellung.

Eine hohe Bedeutung, hinsichtlich der Gestaltung der Softwarebereitstellung, kommt dabei dem Akteur zu, der die Rolle des Dienstleistungsanbieters innehat. Als zentrale kollektive Rolle sollte es in seinem primären

Interesse liegen, eventuell eintretende Veränderungen, d. h. die Instabilität der Bereitstellung, zu entdecken, zu beheben oder sogar Änderungen zu provozieren. Seine wirtschaftliche Existenz, sein originäres Interesse und seine Existenzberechtigung in dem Bereitstellungsnetzwerk hängen von der Stabilität und Weiterentwicklung der Bereitstellung ab. Er kann, wie beschrieben, nicht direkt auf die Dienstleistungserbringung der anderen kollektiven Rollen einwirken, es sei denn, er übernimmt ebenfalls diese Rollen. Daher muss der Akteur (als Dienstleistungsanbieter) durch geeignete Maßnahmen ein Umfeld schaffen, in dem alle beteiligten Akteure den vereinbarten Bereitstellungsrahmen durch ihr Handeln bestätigen und somit stabil halten. Instabilitäten müssen früh erkannt werden und ihnen sollte durch aktives Handeln entgegengewirkt werden.

Da der Akteur in der Rolle des Dienstleistungsanbieters nicht direkt auf die Handlungen der anderen kollektiven Rollen einwirken kann, muss er die Bereitstellung differenziert beobachten. Außerdem kann der Bereitstellungsrahmen nicht nur in der Nutzung von allen bestätigt werden. Der Dienstleistungsanbieter kann eine formelle Bestätigung von allen beteiligten Akteuren einfordern, indem er regelmäßig und in kurzen Abständen (z. B. ein Jahr) zur Herstellung eines neuen Bereitstellungsrahmens aufruft (siehe nächsten Abschnitt). Durch diese aktive Einforderung müssen alle beteiligten Akteure den Bereitstellungsrahmen regelmäßig bewusst bestätigen oder anpassen. Der aktuelle Bereitstellungsrahmen bleibt entweder bestätigt und damit stabil oder er wird verändert und damit durch einen neuen Bereitstellungsrahmen ersetzt, der wieder besser zur vorherrschenden Situation passt.

Durch das regelmäßige Einfordern der Herstellung der Bereitstellung werden alle beteiligten Akteure aktiv und bewusst in die Gestaltung der Softwarebereitstellung einbezogen.

9.1.3 Kontinuierliche Treffen

Die Partizipation aller Akteure an Treffen zur Organisation der weiteren Zusammenarbeit sichert den bewussten Umgang mit Veränderungen, da die sich verändernden Akteure sich entsprechend in die Gestaltung der Softwarebereitstellung einbringen. Darüber hinaus sichern kontinuierliche Treffen in kurzen Zyklen von ca. einem Jahr den Umgang mit Veränderungen über den gesamten Zeitraum.

Da diese Zusammenkünfte der Aushandlung von Kooperationsbedingungen dienen, ist es nicht ratsam, nur ein Treffen mit allen beteiligten Akteuren zu veranstalten. Effektiver sind mehrere Treffen, entsprechend der Anzahl der Kooperationsbeziehungen, an denen dann jeweils nur die beteiligten Akteure teilnehmen.

Diesem Ansatz folgend hat vor allem der Akteur, der die kollektive Rolle des Dienstleistungsanbieters übernimmt, als zentrale Instanz im Akteursnetzwerk entsprechend viele Treffen zur Verhandlung der Kooperationsbeziehungen. Dies lässt auf einen großen Arbeitsaufwand schließen, der die Kosten im Sinne der Transaktionskostentheorie (Abschnitt 5.1.4) erheblich erhöht. Diese Annahme muss teilweise relativiert werden. Die Transaktionskosten einer einmaligen Aushandlung einer Kooperationsbeziehung für einen langen Zeitraum sind beträchtlich, weil Änderungen im Vorwege antizipiert werden und in entsprechend flexiblen Vereinbarungen münden müssen. Dies entfällt bei der Aushandlung von Kooperationsbeziehungen für einen kurzen Zeitraum, da über eventuell auftretende Veränderungen im Vorwege noch nicht verhandelt werden muss. Sie werden erst zum Ende des kurzen Zyklus zum Verhandlungsgegenstand.

Bei langfristigen Kooperationen akkumulieren sich die Transaktionskosten für kurzfristige Verträge und übersteigen mit der Zeit die Kosten für die einmalige Aushandlung der beim ASP üblichen Verträge. Insofern ist der Ansatz, nur kurzfristige Verträge zu vereinbaren, im Hinblick auf Transaktionskosten langfristig gesehen, kostspieliger. Der Vorteil kurzfristiger Verträge ist der Umgang mit Veränderungen im Prozess. Sie werden verhandelt, wenn sie auftreten bzw. spätestens zu Beginn eines neuen Zyklus. Insgesamt können die Transaktionskosten in diesem Ansatz höher sein, doch es wird eine Flexibilität und Stabilität gewonnen, die anders nicht erreichbar ist.

Allerdings werden sich diese Treffen als Grundlage des Veränderungsprozesses der Bereitstellung einer Software nicht von selbst koordinieren. Also muss insbesondere der Akteur, der die Rolle des Dienstleistungsanbieters einnimmt, die Verantwortung für das Aufgabengebiet Koordination der übergeordneten Aufgabe Kooperation übernehmen. Zum einen hat er die meisten Beziehungen zu anderen Beteiligten, zum anderen zählt für ihn, als zentrale und führende Instanz im Akteursnetzwerk, die Koordination der beteiligten Akteure zu seinen originären Interessen.

9.1.4 Langfristige Rahmen- und kurzfristige Detailverträge

Verträge bzw. vertragliche Regelungen gelten, wie in Abschnitt 7.1.4 dargestellt, als Rechtssicherheit. Sie dokumentieren, dass und wie die Vertragspartner miteinander kooperieren wollen. In Hinblick auf das zyklische Vorgehen (Abschnitt 9.1.1) wird für den Ansatz eASP die Kombination von langfristigen Rahmen- und kurzfristigen Detailverträgen vorgeschlagen, die u. a. durch die Auseinandersetzung mit Vertragsarten in der ASP-Literatur motiviert ist (Abschnitt 4.4.1) und in Abschnitt 7.1.4 begründet wurde.

Mit langfristigen Rahmenverträgen wird die Kooperation der beteiligten Akteure auf eine für alle sichere Basis gestellt und dokumentiert, dass hinsichtlich der Bereitstellung einer Software kooperiert werden soll. So wird in Rahmenverträgen der Rahmen für die Bereitstellung geklärt, ohne auf Details der Umsetzung der Kooperation näher einzugehen. Wie die Kooperation im Einzelnen organisiert werden soll, wird in kurzen Detailverträgen geregelt. Sie müssen nicht so ausführlich sein, wie es in der ASP-Literatur (Abschnitt 4.4.2) diskutiert wird. Der Umgang mit Veränderungen wird im zyklischen Vorgehen in den Prozess gelegt, so dass neue kurze Detailverträge als Konsequenz der kontinuierlichen Treffen entstehen. Die Detailverträge sind in zweierlei Hinsicht als kurz zu bezeichnen. Zum einen gelten sie für einen kurzen Zeitraum, z. B. einem Jahr. Sie sollten im Sinne des zyklischen Vorgehens auch bei keinem Veränderungsbedarf explizit verlängert bzw. bestätigt werden. Zum anderen sind die vertraglichen Regelungen, aus den bereits im vorigen Abschnitt erläuterten Gründen, kürzer als die der ASP-Verträge für längere Zeiträume. Ob die Detailverträge rechtlich einem Miet-, Pacht-, Leasing-, Werk- oder Dienstleistungsvertrag entsprechen (Abschnitt 4.4.1), ist von Kooperation zu Kooperation verschiedenen. Es ist durchaus möglich und wahrscheinlich, dass verschiedene Vertragsarten in einem langfristigen Rahmenvertrag zur Softwarebereitstellung zu finden sind.

Die Detailverträge haben neben der rechtlichen Sicherheit noch die Funktion der Dokumentation. Sie stellen eine Momentaufnahme der zum Verhandlungszeitpunkt vorherrschenden Situation in der Bereitstellung der Software dar, inklusive der Einflüsse und Veränderungen der beteiligten Akteure. Die Folge der vertraglichen Regelungen dokumentiert die Histo-

rie der Softwarebereitstellung und würde im Sinne von Rolf (1998, S. 24f. und S. 41f.) der Dokumentation des „Techniknutzungspfads“ entsprechen.

9.2 Anwendung auf die Fallstudien

Der Blick auf die Fallstudien aus zeitlicher Perspektive wird im Folgenden verdeutlichen, wie sich die Fallstudien über die Zeit entwickelt haben. Nachträglich kann nur der *Vorgang* sichtbar gemacht und nicht auf das aktuelle oder zukünftige *Vorgehen* eingegangen werden, da die Bereitstellung von CommSy in den Fallstudien abgeschlossen ist. Dennoch fördert ihre Betrachtung interessante Erkenntnisse zu Tage.

9.2.1 CommSy@Uni.de

In der Fallstudie „CommSy@Uni.de“ fanden zyklischen Treffen nur selten statt. Die zwei ausgearbeiteten und auf eine langfristige Kooperation ausgelegten Verträge sind nur deshalb als kurz zu bezeichnen, weil sie wichtige Quantitäts- und Qualitätsdefinitionen von Leistungen im Sinne der SLAs vermissen ließen. Außerdem wurden zukünftige Veränderungen in den Verträgen nicht berücksichtigt. So gestaltete sich die Kooperation als äußerst schwierig und letzten Endes als erfolglos. Das Auftreten von Veränderungen traf die Kooperation völlig unvorbereitet. Sie war weder durch Verträge abgesichert noch durch einen Kooperations- und Kommunikationsprozess, moderiert vom Dienstleistungsanbieter Uni.de, aufgefangen. Mit Blick auf die zyklische Vorgehensweise ist festzustellen, dass die Bereitstellung von CommSy durch Uni.de schnell instabil wurde und anfängliche Bemühungen beider Seiten, den vereinbarten Bereitstellungsrahmen zu ändern, zu keinem Abschluss führten. Veränderungen dagegen gab es verschiedene.

Die geplante Integration von Uni.de in das europaweit agierende Unternehmen FirstCampus, das in vielen Ländern webbasierte Universitätsportale anbieten wollte, führte zur Forderung seitens Uni.de, CommSy auch in anderen europäischen Ländern einsetzen zu dürfen. HITeC lehnt ab. Diese Forderung, die vier Monate nach der ersten Kontaktaufnahme gestellt wurde, verzögerte die Vertragsverhandlungen, so dass CommSy bei Uni.de, nicht wie geplant zum Wintersemester 2000/2001, sondern erst zum Sommersemester 2001 an den Start gehen konnte. Hätten sich die Vertragsparteien zunächst auf die Bereitstellung für den deutschsprachigen Raum

beschränkt und die Frage der europäischen Nutzung in einem von Uni.de moderierten Prozess diskutiert, dann wäre CommSy bereits ein halbes Jahr früher bei Uni.de online gewesen und die europäische Frage hätte den Start nicht verzögert. Letztendlich wurde trotz des ausgehandelten Nutzungsrechts nicht ein einziger CommSy-Projektraum in anderen europäischen Ländern genutzt.

Die Initiatoren der Kooperation bei Uni.de sahen in CommSy die Möglichkeit, neben Studierenden auch Wissenschaftlern mit ihrem Universitätsportal `http://www.uni.de/` einen Mehrwert bieten zu können. Im Gegensatz zur Zahl der Studierenden war die Anzahl der Benutzerkennungen für CommSy gering. Dennoch hielten die Initiatoren bei Uni.de an CommSy als Erweiterung ihres Portfolios fest. Mit CommSy wollte sich Uni.de in den Universitäten etablieren, wofür Studierende nicht die geeignete Zielgruppe waren. Der ständige Wechsel der Ansprechpartner auf der Seite von Uni.de führte letztendlich zum Erliegen der Kommunikation zwischen Uni.de und dem CommSy-Team.

In der zweiten Hälfte des Jahres 2001 wurde Uni.de Teil der Unternehmensgruppe FirstCampus und büßte damit seine Eigenständigkeit als souveräne Firma ein. Durch FirstCampus erfolgte eine Prüfung aller Geschäftsprozesse von Uni.de mit der Absicht, wirtschaftlich rentable Zweige zu fördern und unrentable abzustoßen. CommSy war aufgrund der niedrigen Nutzungszahlen, im Gegensatz zu anderen Bereichen bei Uni.de, mit Werbebannern nicht finanzierbar und daher unrentabel. FirstCampus entschied, die Personalkosten (Betreuungsaufwand) zu reduzieren. CommSy lief seitdem als unbetreuter Dienst. Die Anfragen der Nutzenden an das CommSy-Team mehrten sich, und die Mitarbeiter sahen sich kaum in der Lage, den zusätzlichen Arbeitsaufwand zu bewältigen. Die kollektiven Rollen wechselten von einem Akteur zum anderen. Die oben skizzierten Vorkommnisse führten letztendlich zur Beendigung der Kooperation. Seitdem läuft CommSy, trotz mehrerer Mahnungen, unbetreut bei Uni.de weiter, mittlerweile in einer völlig veralteten Version.

Resümierend ist festzustellen, dass sich unterschiedliche Veränderungen in der Bereitstellung von CommSy bei Uni.de ergaben, die zu verschiedenen Zyklen in der Bereitstellung führten. Doch diese Zyklen wurden nicht explizit mit allen Beteiligten etabliert und kein neuer Bereitstellungsrahmen wurde ausgehandelt. Die beteiligten Akteure reagierten auf die für

sie unbefriedigende Bereitstellung individuell und (überspitzt ausgedrückt) ohne die anderen beteiligten Akteure zu unterrichten.

9.2.2 CommSy@WissPro

In der Fallstudie „CommSy@WissPro“ sind keine schriftlich fixierten Verträge zwischen den kollektiven Rollen (bzw. den Akteuren, die die Rollen einnahmen) vereinbart worden. Es bestanden jedoch ein Vertrag zwischen dem BMBF und der Universität Hamburg und Arbeitsverträge zwischen den Projektmitarbeitern und der Universität Hamburg. In diesen Verträgen sind zur Softwarebereitstellung von CommSy keine Aussagen enthalten, dennoch gaben sie den Rahmen für die Arbeit und die Aufgaben im Forschungs- und Entwicklungsprojekt WissPro vor. Daher können die vorhandenen Verträge als langfristige Rahmenverträge interpretiert werden.

In diesem Bereitstellungsszenario übernahm das Projekt WissPro alle für die Bereitstellung relevanten kollektiven Rollen. Am Anfang wurden alle Aufgaben in der CommSy-Bereitstellung, im Zuge der technischen Weiterentwicklung von CommSy (Projekträumen, Portal, Archiv), von allen Mitarbeitern in Personalunion für die jeweiligen Produkte übernommen. Dies führte zu drei Bereitstellungskonstellationen und einer gewissen Konkurrenz der Teams untereinander. Die Beteiligten reagierten auf diese Situation, indem sie die drei Produkte zu einem vereinten und anschließend die Aufgaben neu verteilten. Es entstanden wiederum drei Teams (Benutzerbetreuung, Betrieb und Entwicklung), wobei die teamübergreifende Kommunikation zunächst vernachlässigt wurde. Die Teams erkannten das Problem und diskutierten fortan übergreifende Themen der Bereitstellung auf den wöchentlichen Projektsitzungen. Bei größerem Diskussionsbedarf wurden Workshops initiiert, die nicht regelmäßig, sondern bei Notwendigkeit stattfanden. Auch wenn bei diesen Workshops die Koordination der Beteiligten nicht im Vordergrund stand, koordinierten sich alle, neben den wöchentlichen Sitzungen und den verschiedenen Medien, auch über diese Treffen.

In dieser Fallstudie bestanden keine kurzfristigen Detailverträge und auch keine explizite Bestätigung oder Veränderung des Bereitstellungsrahmens. Dennoch wurden Kommunikationsregeln aufgestellt und auf unbefriedigende Bereitstellungssituationen zügig reagiert, indem die Bereitstellung an die veränderten Bedingungen angepasst wurde. Dies entspricht dem in Abschnitt 9.1.1 vorgestellten zyklischen Vorgehen, ohne dass bei Comm-

Sy@WissPro das Vorgehen explizit etabliert bzw. durch vertragliche Regelungen dokumentiert worden wäre.

9.3 Zwischenergebnis

Die Betrachtung von Einflüssen und Veränderungen über einen längeren Zeitraum ist eine Voraussetzung für die Gestaltung der Softwarebereitstellung. Dabei wird deutlich, dass alle beteiligten Akteure durch ihr Handeln jederzeit die Softwarebereitstellung gestalteten. Sie werden die Softwarebereitstellung durch ihr Verhalten ändern, da sie selbst fortwährend Änderungen unterliegen, die ihr Verhalten beeinflussen. Der vormals vereinbarte Bereitstellungsrahmen wird im Laufe der Zeit zwangsläufig instabil.

Da ein Akteur nicht auf alle Handlungen im Bereitstellungsnetzwerk direkt einwirken kann, ist es wichtig, dass alle an der Gestaltung fortwährend aktiv beteiligt werden. Diese Aufgabe fällt der Rolle Dienstleistungsanbieter als zentrale kollektive Rolle zu. Der Akteur, der die Rolle einnimmt, muss, zur Stabilisierung der Softwarebereitstellung, die Bestätigung des aktuellen Bereitstellungsrahmens oder dessen Veränderung von den anderen Akteuren periodisch einfordern.

Die Softwarebereitstellung verläuft zyklisch:

- Angefangen bei der Herstellung eines Bereitstellungsrahmens,
- über die Nutzung der angebotenen Dienstleistungen, d. h. der Ausfüllung des Rahmens mit Leben, und dem in der Nutzung instabiler werdenden Bereitstellungsrahmen,
- bis hin zur folgenden oder bereits durch den Dienstleistungsanbieter vorher initiierten Herstellung des nächsten Bereitstellungsrahmens.

In diesen Zyklen können neue Akteure jederzeit ein- und beteiligte jederzeit aussteigen. Rechtlich gesichert werden die verschiedenen Kooperationen in dem sich verändernden Dienstleistungsnetzwerk über langfristige Rahmen- und kurzfristige Detailverträge. Langfristige Rahmenverträge dienen der Bereitstellung als Grundlage, kurzfristige Detailverträge als Ausgestaltung einzelner Kooperationsaspekte. Hierbei müssen die Detailverträge die enthaltenen Aspekte detailliert klären, zukünftige aber nicht antizipieren, da Veränderungen durch das zyklische Vorgehen im Prozess verhandelt

werden. Ergebnisse dieser Verhandlungen sind angepasste, veränderte oder neue Detailverträge.

Die hier vorgestellte Vorgehensweise verbindet die in den vorigen drei Kapiteln dargestellten Perspektiven auf die Softwarebereitstellung und verzahnt sie über die gesamte Zeit einer Bereitstellung hinweg. Die zyklische Vorgehensweise stellt die Integration der Perspektiven und gleichzeitig den Kern des zu erarbeitenden Ansatzes eASP dar. Die Erarbeitung des Ansatzes ist durch die Darstellung der Vorgehensweise, als integrierender Kern, und der Perspektiven, als Ausgestaltungen und Ergänzungen des Kerns, abgeschlossen.

III

Ergebnis – der Ansatz eASP

- 10 -
Softwarebereitstellungsansatz eASP

Im Folgenden wird der Softwarebereitstellungsansatz eASP zusammenfassend und kompakt beschrieben, indem die in Teil II dargestellten Perspektiven zu einem homogenen Gesamtbild zusammengefügt werden. Für eine ausführliche Darstellung des Ansatzes eASP und der Perspektiven sei auf die entsprechenden Kapitel in Teil II verwiesen. Nach der kompakten Darstellung wird die Einbettung von eASP in das Gebiet „Informatiksysteme in Organisationen“ vollzogen, in dem eASP in die IT-unterstützte Organisationsgestaltung und den Softwareentwicklungsrahmen STEPS integriert wird. Daran anschließend wird die Wertigkeit von eASP für die Praxis betrachtet, indem es als Erweiterung von ASP dargestellt wird und zum Abschluss eine Diskussion über die Verallgemeinerbarkeit von eASP erfolgt.

10.1 Kompakte Zusammenfassung von eASP

Mit eASP als Abkürzung für evolutionary Application Service Providing wird die evolutionäre Bereitstellung einer Anwendung als Dienstleistung verstanden. Dabei haben die in eASP enthaltenen Begriffe folgende Bedeutung:

- *evolutionary (evolutionär)*: Unter evolutionär wird die fortlaufende Entwicklung und Veränderung der Bereitstellung aufgrund von äußeren und inneren Einflüssen verstanden. Dies bedeutet insbesondere die Berücksichtigung und Einbeziehung aller beteiligten Akteure (Partizipation), die Reflexion von Vergangenem sowie das bewusste Durchleben von verschiedenen wiederkehrenden Situationen (zyklisches Vorgehen).

- *Providing (Bereitstellung)*: Die Bereitstellung umfasst technische und organisatorische Aufgaben, um Nutzenden eine gesicherte Nutzung gewährleisten zu können. Zu den Aufgaben gehören neben dem technischen Betrieb u. a. auch die Benutzerbetreuung, Softwareentwicklung, Marketing und Kundenbetreuung.

Unter Bereitstellung wird hier verstanden, dass eine Anwendung von einem Anbieter den Kunden bzw. den Nutzenden angeboten wird. d. h. die Nutzenden greifen von ihrem Endgerät (PC, Organizer, Handy ...) auf eine Anwendung zu, die auf einem Server des Anbieters läuft (Client/Server-Prinzip).

- *Application (Anwendung)*: Mit Anwendung ist Software gemeint, die von vielen benutzt wird, im Gegensatz zu Software, die nur von einem genutzt wird.

- *Service (Dienstleistung)*: Durch den Begriff Dienstleistung entsteht eine starke Kundenorientierung. Der Fokus wird zum einen auf die Servicequantität und -qualität und zum anderen auf die Erbringung und Bezahlung von (Dienst-)Leistungen gelegt. Darüber hinaus werden durch den Dienstleistungsbegriff auch vertragsrechtliche Aspekte in den Blick genommen.

Um die Bereitstellung einer Software in Gänze zu begreifen, muss sie in einer Technik-, Aufgaben- und Organisationsperspektive betrachtet werden. Ergänzend muss die Vorgangsperspektive eingenommen werden, die durch Gestaltungsoptionen zum Vorgehen wird und die anderen Perspektiven in einer zyklischen Vorgehensweise verquickt:

- *Aufgabenperspektive*: Bei der Bereitstellung von Software sind verschiedene Aufgaben in verschiedenen Aufgabengebieten zu leisten.

- *Organisationsperspektive*: Die identifizierten Aufgaben müssen von Akteuren in unterschiedlichen Rollen in einem Akteursnetzwerk übernommen werden. Die Kooperation der beteiligten Akteure gleicht einem Geflecht aus Dienstleistungserbringung und -nutzung, das Kostentransparenz und rechtlicher Sicherheit bedarf.

- *Technikperspektive*: Eine geeignete Infrastruktur und die bereitzustellende Anwendung ist Grundlage jeder Softwarebereitstellung.

- *Vorgehen*: Einflüsse wirken kontinuierlich auf die Bereitstellung ein und verändern sie auf verschiedenen Ebenen. Mit diesen Veränderungen kann dauerhaft, aktiv und gemeinsam in einer zyklischen Vorgehensweise umgegangen werden.

Durch die genannten Perspektiven und die zyklische Vorgehensweise treten die angesprochenen Aspekte, über die die Bereitstellung von Software analysiert und gestaltet werden kann, deutlich hervor. eASP ist hauptsächlich ein Ansatz zur Analyse von Bereitstellungskontexten. Da es darüber hinaus beispielhafte Erzählungen und Erläuterungen anbietet, die auf konkrete Bereitstellungskontexte adaptiert werden können, zeigt eASP Gestaltungsoptionen auf, so dass der Ansatz eASP auch zur Gestaltung von Bereitstellungskonstellationen herangezogen werden kann.

10.1.1 Aufgabenperspektive

Zur Bereitstellung von Software gehört eine Fülle von Aufgaben. Zur besseren Handhabung wird sie in Aufgaben, Aufgabengebiete, übergeordnete Aufgaben und übergreifende Aufgaben unterteilt. Aufgaben stellen die kleinste Einheit dar, die zu Aufgabengebieten einer funktionellen Rolle gebündelt werden können. Übergeordnete Aufgaben bestehen aus Aufgabengebieten und werden von verschiedenen funktionellen Rollen eines Organisationsbereiches geleistet. Übergreifende Aufgaben bestehen wiederum aus übergeordneten Aufgaben und werden von kollektiven Rollen (s.u.) wahrgenommen. Bei der Definition der Bereitstellung einer Software als übergreifende Aufgabe ergibt sich folgende Aufgabenstruktur:

Übergeordnete Aufgabe	**Aufgabengebiet / funktionelle Rolle**	**Aufgabe**
Anwendungsentwicklung	Leitung / Entwicklungsleiter	Entwicklung koordinieren und überwachen
		Entwicklungsentscheidungen treffen
	Planung / Entwicklungsplaner	Anforderungen analysieren
		Implementierungen planen und Zeitplan erstellen
	Programmierung / Programmierer	Features programmieren
		Fehler beheben
		Softwarearchitektur pflegen
	Dokumentation / Programmierer	Benutzerdokumentation pflegen
		Entwicklerdokumentation pflegen
	Entwicklungsumgebung / Entwicklungsadministrator	Entwicklungsclients pflegen
		Entwicklungsserver pflegen
		Zusätzlich benötigte Software pflegen
	Releasemanagement / Entwicklungsplaner	Release identifizieren, zusammenstellen und veröffentlichen
	Fehlermanagement / Fehlerbeauftragter	Hotline anbieten
		Fehlerberichte in Entwicklungsprozess integrieren
		Fehlerbehebung veröffentlichen

Übergeordnete Aufgabe	**Aufgabengebiet / funktionelle Rolle**	**Aufgabe**
Betrieb	Hardware / Hardwareadministrator	Serverhardware pflegen
	Basissoftware / Softwareadministrator	Zusätzlich benötigte Software pflegen
	Anwendung / Anwendungsadministrator	Anwendung installieren, konfigurieren und pflegen
	Datensicherheit / Softwareadministrator	Back-up der Daten durchführen
		Virenscan der Daten durchführen
		Weitere Sicherheitsmechanismen einrichten
	Ausfallsicherheit / Hardwareadministrator	Klima- und Brandschutz installieren
		Stromversorgung sichern
Benutzerbetreuung	Handhabungssupport / Benutzerbetreuer	FAQ pflegen
		Hotline (Telefon, E-Mail) anbieten
	Kommunikation / Benutzerbetreuer	Benutzerwünsche und Fehlerberichte weiterleiten
		Informationsmaterial erstellen und verteilen
		Informationsveranstaltungen und Workshops organisieren
	Evaluation der Nutzung / Evaluator	Evaluation konzipieren und durchführen
		Ergebnisse in die Bereitstellung integrieren
Marketing	Bedarfsanalyse / Analyst	Bedarfsanalyse konzipieren und durchführen
		Ergebnisse in die Bereitstellung integrieren
	Werbung / Werbefachmann	Informationsmaterialien entwickeln und verbreiten
		Software und Dienstleistungen präsentieren
		Werbestrategie entwickeln
Kundenbetreuung	Kommunikation / Kundenbetreuer	Hotline (Telefon, E-Mail) anbieten
		Informationsmaterialien entwickeln und verschicken
		Kundenwünsche weiterleiten
	Verträge / Kundenbetreuer	Standardverträge entwickeln
		Vertragsverhandlungen führen
	Abrechnung / Buchhalter	Rechnungen stellen
		Zahlungen überwachen
Zugriffsermöglichung	Netzverbindung / Netzwerkadministrator	Verbindung etablieren und pflegen
	Netzsicherheit / Netzwerkadministrator	Netzwerksicherheit installieren und pflegen
Kooperation (alle Dienstleister)	Abrechnung / Verwalter	Rechnung stellen
		Zahlungen überwachen
	Kommunikation / Ansprechpartner	Internen Newsletter herausbringen
		Feedback geben
	Koordination / Koordinator	Beteiligung an Kooperationstreffen
		Überblick über Bereitstellung behalten
	Training / Trainer	Interne Workshops durchführen
	Vertragswerk /	Kooperationsverträge entwickeln

Übergeordnete Aufgabe	Aufgabengebiet / funktionelle Rolle	Aufgabe
	Verwalter	Vertragsverhandlungen führen
		Vertragsbrüche verhandeln

Tabelle 7: Aufgaben bei der übergreifenden Aufgabe *Software x bereitstellen*

Die Darstellung von Aufgaben ist als Rahmen und Möglichkeitsraum zu verstehen. In konkreten Bereitstellungsszenarien wird die Aufgabenfülle durch entsprechende Szenarienspezifika unterschiedlich ausfallen.

10.1.2 Organisationsperspektive

Die Organisation der Bereitstellung von Software bedeutet, die verschiedenen (übergeordneten) Aufgaben real existierenden Akteuren zu zuordnen. Um die verschiedenen Akteure, die von Bereitstellungsszenario zu Bereitstellungsszenario unterschiedlich sein können, in den Blick nehmen zu können, wird folgendes kollektives Rollennetz benutzt:

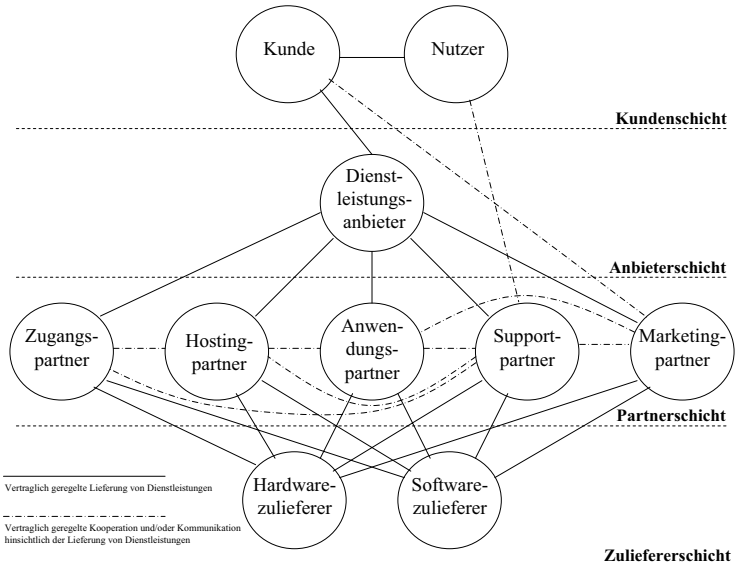

Abbildung 31: Kollektives Rollennetz bei der Bereitstellung von Software

Mit dem kollektiven Rollennetz kann die Organisation der Softwarebereitstellung auf einer abstrakteren Ebene dargestellt werden. Folgende kollektive Rollen sind an der Softwarebereitstellung beteiligt:

- Der *Kunde* kauft bzw. bezahlt die vom Dienstleistungsanbieter angebotenen, kostenpflichtigen Anwendungen und Dienstleistungen für die Nutzer (seine Angestellten oder Kunden).
- Der *Nutzer* verwendet die vom Dienstleistungsanbieter angebotenen und vom Kunden bezahlten Anwendungen und Dienstleistungen.
- Der *Dienstleistungsanbieter* stellt dem Kunden Dienstleistungen hinsichtlich der Softwarebereitstellung aus einer Hand zur Verfügung. Er übernimmt die Kundenbetreuung und ist für die Koordination verantwortlich.
- Der *Anwendungspartner* stellt die Anwendung (Software) für den Dienstleistungsanbieter zur Verfügung.
- Der *Hostingpartner* übernimmt den Betrieb, d. h. das Hosting der Anwendung und zusätzlicher Software für den Dienstleistungsanbieter.
- Der *Zugangspartner* bietet dem Dienstleistungsanbieter den Zugang vom Kunden bzw. Nutzer zum Hostingpartner an.
- Der *Supportpartner* übernimmt die Benutzerbetreuung der Nutzer beim Kunden im Namen des Dienstleistungsanbieters.
- Der *Marketingpartner* übernimmt das Marketing, also u. a. die Platzierung der angebotenen Dienstleistungen im Markt und die Kundenakquirierung im Namen des Dienstleistungsanbieters.
- Der *Hardwarezulieferer* versorgt die Partner mit entsprechend benötigter Hardware.
- Der *Softwarezulieferer* versorgt die Partner mit entsprechend benötigter Software.

Die angegebenen Verbindungen zwischen den kollektiven Rollen geben die vertraglich zu regelnde Kooperation und/oder Kommunikation wieder sowie die ebenfalls vertraglich zu regelnde Dienstleistungserbringung. Gemeinsam ergeben die Verbindungen das nötige Kommunikationsnetz der kollektiven Rollen zur Bereitstellung und Nutzung einer Software. Dabei zeigen die durchgezogenen Linien die Kommunikation hinsichtlich der Vorbereitung der Bereitstellung, d. h. der Aushandlung von Dienstleistungsbeziehungen zwischen den kollektiven Rollen, sowie die Kommunikation hinsichtlich der Abrechnung der angebotenen und genutzten Dienstleistungen. Die gestrichelten Linien zeigen die Kommunikation, die während der konkreten Softwarenutzung zusätzlich auftritt.

Der Begriff Dienstleistung bzw. die Anwendung des Begriffs auf den Kontext der Softwarebereitstellung stellt die Bereitstellung von Software unter das Primat der Kundenorientierung. Da sich die Existenz eines Servicegebers auf der Zufriedenheit seiner Servicenehmer gründet, wird er alles daran setzen, diese Zufriedenheit zu erzeugen. Da der Servicegeber aber wirtschaftlichen Zwängen unterliegt, wird er nur so viel Engagement in die Bedürfnisbefriedigung investieren, bis die Befriedigung eintritt. Da dies bei mehreren Servicenehmern zu unterschiedlichen Zeitpunkten eintritt, wird der Servicegeber die Dienstleistungen in verschiedenen Qualitäten und Quantitäten anbieten. Quantität bedeutet in diesem Zusammenhang Abstufungen in der Menge der angebotenen Aufgaben (Kernleistungen, weitergehende Leistungen, umfassende Leistungen), Qualität die Durchführung der einzelnen Aufgaben. Je nach bereitgestellter Anwendung bzw. Nutzung (Persönliche, Kooperative, Organisatorische Nutzung) werden verschiedene Quantitäts- und Qualitätsstufen benötigt.

Indem die an der Softwarebereitstellung beteiligten Akteure die Kooperation als Dienstleistung verstehen, streben sie vertragliche Regelungen an, die der rechtlichen Absicherung der verschiedenen Beziehungen dienen. Sie erfüllen drei Funktionen bei der Bereitstellung von Software:

1. Begründung eines Schuldverhältnisses zur rechtlichen Absicherung im Konfliktfall.

2. Spezifizierung der zu leistenden Aufgaben in Quantität und Qualität als Beitrag zur Transparenz und Kalkulierbarkeit der angebotenen Leistungen.

3. Dokumentation der Zusammenarbeit, insbesondere zur Regelung der Kooperation und Kommunikation.

Die Art der Verträge ist vom Inhalt der von den Vertragspartnern vereinbarten Leistungspflicht abhängig und kann verschiedene Mischformen von juristischen Vertragsarten (Miete, Pacht, Leasing, Werkvertrag, Dienstvertrag) enthalten. Die Benutzung von Service Level Agreements (SLAs) in den vertraglichen Regelungen macht die Leistungen in der Bereitstellung von Software in einer ausreichenden Detailtiefe transparent und klärt darüber hinaus Verantwortlichkeiten.

SLA-Bereich	SLA-Komponente	Erläuterung
Zugriff	Verfügbarkeit und Ausfallzeiten	Die Verfügbarkeit wird üblicherweise in Prozent angegeben. Bezugsgröße ist hier ein konkret definierter Zeitraum. Geplante Ausfallzeiten (z. B. Einspielung von Updates) sollten explizit aufgeführt werden.
	Netzwerkausstattung und Netzwerkarchitektur	Ausstattung und Architektur des Netzwerks sind grundlegende Voraussetzung. Erwartet werden skalierbare und redundante Netze.
	Netzsicherheit	Zur Netzwerksicherheit zählt die Sicherung der Verbindung zwischen Anwender und Anwendung und die Sicherung der entmilitarisierten Zone. Hier müssen Sicherheitsmechanismen wie Firewalls, Verschlüsselung, Tunneling usw. definiert werden.
	Datendurchsatz und Antwortzeiten	Der Datendurchsatz und die Antwortzeiten hängen von der Qualität und der Bandbreite des Übertragungsmediums ab. Verlustfreie Datenübertragung wird wie die Verfügbarkeit in Prozent gemessen.
Betrieb	Verfügbarkeit und Antwortzeiten	Die Verfügbarkeit und Antwortzeiten der Server bzw. der Hard- und Software des internen Bereichs hängen von den gleichzeitig zugreifenden Nutzern maßgeblich ab. Antwortzeiten werden in (Milli-)Sekunden und die Verfügbarkeit in Prozent eines definierten Zeitbereichs angegeben.
	Datensicherheit	Zur Vorbeugung von Datenverlusten gehört das Durchführen, Lagern und bei Bedarf das Wiedereinspielen von Back-ups. Es können Lagerungszeiträume, Reaktionszeiten und die Häufigkeit von Backups bestimmt werden.
	Physische Sicherheit	Zur physischen Sicherheit gehören u. a. das Sicherheitspersonal, Überwachungstechnik, Brandschutz, direkte Verbindung zu und Reaktionszeit von Feuerwehr und Polizei.
Anwendung	Verfügbarkeit	Die Verfügbarkeit spielt hier insbesondere auf die Absturzsicherheit der bereitgestellten Software an.
	Schnelligkeit	Die Performance muss insbesondere unter dem Blickwickel von großen Lasten bewertet werden.
	Sicherheit	Sicherheit bezieht sich hier auf die Mehrbenutzer- und Mandantenfähigkeit der Anwendung, d. h. der Qualität der Abschirmung von Kundendaten vor unbefugtem Zugriff. Dieser Punkt streift insbesondere viele Aspekte des Datenschutzes.
	Updates und Upgrades	Es muss definiert werden, welche Version der Anwendung bereitgestellt und wie viele bzw. wann Updates und Upgrades eingespielt werden.
	Eigentumsrecht	Es sollte vertraglich geregelt werden, dass die Kundendaten Eigentum des Kunden sind, auch wenn sie nicht auf deren Servern, sondern, von den Kunden aus gesehen, auf externen Servern liegen.

SLA-Bereich	SLA-Komponente	Erläuterung
Benutzerbetreuung	Reaktiver Benutzersupport	Der Benutzersupport reagiert auf Benutzeranfragen per Telefon oder E-Mail z. B. auch mittels eines Call-Centers.
	Aktive Anwendungsunterstützung	Die aktive Anwenderunterstützung bietet von sich aus Schulungen, Coaching, Workshops, Diskussionsforen, Seminare usw. an.
Kooperation	Kommunikation	Hinsichtlich der Kommunikation müssen bei den kollektiven Rollen Ansprechpartner für die anderen kollektiven Rollen sowie deren „Reaktionszeit“ benannt werden. Darüber hinaus sind Kommunikationswege für unterschiedliche Kommunikationsanlässe zu definieren.
	Eskalationsmanagement	Treten ad hoc Störungen auf, muss auf ein zuvor vereinbartes Reaktionsschema zurückgegriffen werden können. Dieses Schema legt fest, in welchen Schritten und welchen Zeiträumen wer welche Störungsbeseitigungsmaßnahmen umsetzt.
	Vertragsdauer und Kooperationszyklen	Die Vertragdauer wird i.d.R. auf ein Jahr festgelegt. Die Vertragspartner sollten im Anschluss Treffen vereinbaren, an denen die SLAs neu verhandelt werden.
	Beendigungsvereinbarungen	Die Beendigungsvereinbarungen beinhalten insbesondere Regelungen über die Freigabe und den Transfer der für die Bereitstellung wichtigen Daten (z. B. Kundendaten).

Tabelle 8: Service Level Agreements für eASP

Bei der Betrachtung der Kosten einer Softwarebereitstellung ist zwischen den gesamten Kosten und den in Rechnung gestellten Gebühren zu unterscheiden. Gebühren können einmalig (z. B. Einrichtungsgebühr), wiederkehrend (z. B. monatliche Grundgebühr) oder nutzungsabhängig (z. B. Übertragungsvolumen) berechnet werden. Alle im Dienstleistungsnetzwerk in Rechnung gestellten Gebühren sind eine gute Approximation der Kosten der Bereitstellung, treffen sie aber nicht genau. Die Kosten der Bereitstellung ergeben sich als Addition der Aufwände (Anwendungsentwicklung, Benutzerbetreuung, Betrieb, Kundenbetreuung, Marketing und Zugriffsermöglichung) aller Beteiligten sowie den zur Kooperation nötigen Transaktionskosten (Kooperation). Transaktionskosten sind z. B. anfallende Kosten in Vertragsverhandlungen, in der Kontrolle der vereinbarten Leistungserbringung und in der Anpassung der Beziehungen im Bereitstellungsnetzwerk bei auftretenden Veränderungen.

Letztendlich interessieren einen beteiligten Akteur bei der Bereitstellung von Software seine tatsächlichen Kosten, die sich aus seinem eigenen Arbeitsaufwand (z. B. in Menschmonaten oder -tagen), den von anderen Akteuren abgerechneten Leistungen und den Transaktionskosten des einzelnen Akteurs berechnen. Auf der Grundlage der Aufschlüsselung seiner Kosten kann ein Akteur seine Beteiligung an der Softwarebereitstellung bewerten und für sich entscheiden, ob z. B. ein Outsourcing der Softwa-

rebereitstellung oder von Teilen der Softwarebereitstellung wirtschaftlich vorteilhafter gegenüber einer internen Lösung ist.

10.1.3 Technikperspektive

Die zentrale Bereitstellung einer Software für viele Nutzer erfordert gewisse Voraussetzungen hinsichtlich der bereitzustellenden Anwendung und der technischen Infrastruktur.

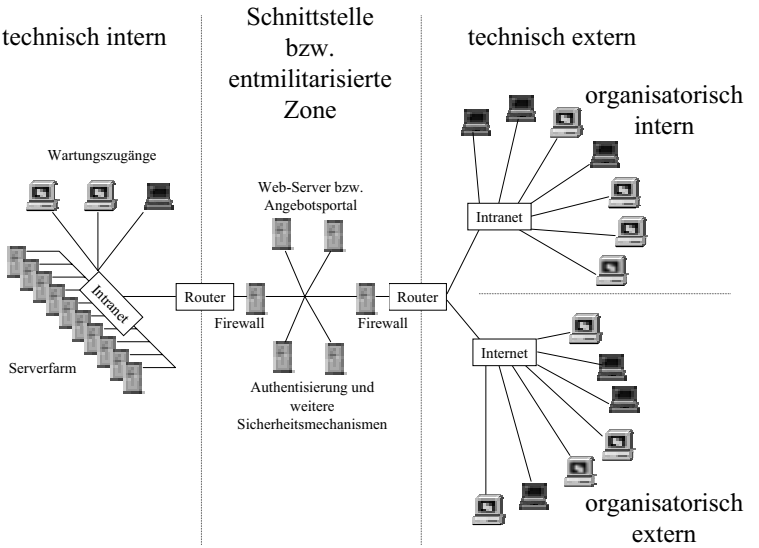

Abbildung 32: Technische Infrastruktur für eASP

Für die Technikperspektive ist ausschlaggebend, dass die Software zentral für viele Nutzende bereitgestellt wird. Dies legt eine Client/Server-Architektur nahe. Dabei kann die Infrastruktur technisch in einen internen (Serverfarm für den Betrieb der Software), einen entmilitarisierten (gesicherter Zugang zum internen Bereich) und einen externen Bereich (Verbindung zu den Nutzern) getrennt werden. Der technisch externe Bereich kann wiederum in organisatorisch intern (Intranet) und extern (Internet) aufgeteilt werden. Die Betrachtung und Konzipierung von organisatorisch in-

tern und extern als technisch gleich ermöglicht aus Sicht eines Kunden ein flexibles Outsourcing oder Insourcing der Softwarebereitstellung bzw. aus Sicht eines Dienstleistungsanbieters die flexible Ankopplung von Kunden an die eigene Infrastruktur.

Die bereitzustellende Anwendung muss daher server- und clientseitig plattformunabhängig sein. Änderungen in der Technik auf der einen oder anderen Seite dürfen keinen Einfluss auf die Bereitstellung und Nutzung der Software haben. Nur mit einer plattformunabhängigen Software wird die flexible Aus- oder Eingliederung des Betriebs bzw. das Anbinden von verschiedensten Kunden an die technische Infrastruktur eines Dienstleistungsanbieters ermöglicht.

Darüber hinaus muss eine zentral für viele bereitgestellt Anwendung mehrbenutzer- und mandantenfähig sein. Mehrbenutzerfähigkeit bedeutet, mit geeigneten Authentisierungs- und Sicherungsmechanismen den Nutzenden eindeutig zu identifizieren und ihm nur seine Daten zu präsentieren. Die Mandantenfähigkeit sichert bei kooperationsunterstützenden Anwendungen, dass Nutzende unterschiedlicher Gruppen bzw. Unternehmen nur die Daten ihrer Gruppenmitglieder bzw. Kollegen zu sehen bekommen und nicht Daten von fremden Mandanten. Eine weitere Anforderung ist die Anpassung der Anwendung hinsichtlich der Benutzungsschnittstelle an die persönlichen Bedürfnisse der Nutzenden bzw. die Anforderungen der entsprechenden Mandanten.

10.1.4 Vorgehen

Die Softwarebereitstellung wird durch gesellschaftliche und wirtschaftliche Prozesse beeinflusst. Diese äußeren Einflüsse wirken sich über die beteiligten Akteure auf die Softwarebereitstellung aus. Letztendlich sind sie es, die sich unter dem Druck der Veränderungen in Gesellschaft und Wirtschaft selbst wandeln. Durch folgende zyklische Vorgehensweise wird der Umgang mit diesen Veränderungen in den Prozess der Bereitstellung gelegt.

- *Etablierung*: Die Etablierung eines neuen Zyklus in der Bereitstellung einer Software stellt den Start für den neuen Zyklus dar. Initiiert wird ein neuer Zyklus durch die beteiligten Akteure, die die vorherrschende Bereitstellung (an bestimmten Stellen) als unbefriedigend empfinden.

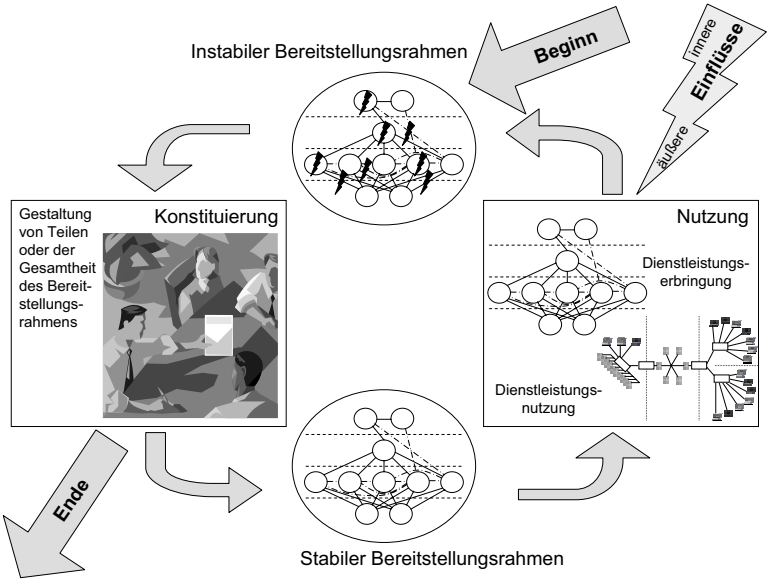

Abbildung 33: Zyklisches Verständnis der Softwarebereitstellung

- *Herstellung*: Alle an der Softwarebereitstellung beteiligten Akteure tragen in ihren kollektiven Rollen zur Gestaltung des Bereitstellungsrahmens bei. Dabei ist nicht nur das Verhandeln der Kooperationen mit anschließender Niederschrift von Dienstleistungsverträgen gemeint, sondern auch die Schaffung der Voraussetzungen zur Erfüllung der vereinbarten bzw. versprochenen Dienstleistungen. So bedeutet die Herstellung des Bereitstellungsrahmens die Gestaltung eines Dienstleistungsnetzwerks respektive Vertragsnetzwerks und die Vorbereitung des Bereitstellungskontextes. Dabei muss der gesamte Bereitstellungsrahmen nur verhandelt werden, wenn sich eine Softwarebereitstellung neu gründet oder gravierende Änderungen stattfinden. Generell werden nur bestimmte Ausschnitte neu gestaltet, wenn sich partielle Änderungen ergeben.

- *Bereitstellungsrahmen*: Das Ergebnis der Herstellung ist ein an die aktuellen Bedürfnisse und Bedingungen angepasster Bereitstellungs-

rahmen. Dies umfasst die Vorbereitung des Kontextes, insbesondere innerhalb der Akteure, und die rechtliche Absicherung durch Dienstleistungsverträge.

- *Einsatz*: Das Anbieten von Dienstleistungen aller Akteure in ihren kollektiven Rollen sowie das Nutzen der angebotenen Dienstleistungen durch entsprechende Akteure bringen den Bereitstellungsrahmen „zum Einsatz“. Hierbei ist die Nutzung der bereitgestellten Software durch die Nutzenden inbegriffen. Im Laufe der Softwarebereitstellung entstehen Veränderungen, die zu einer (aus Sicht bestimmter Akteure) unbefriedigenden Bereitstellung führen. Die mit der Bereitstellung unzufriedenen Akteure nehmen dies als Anlass, einen neuen Zyklus zu initiieren und die Herstellung eines neuen Bereitstellungsrahmens zu etablieren. Die „unbefriedigende“ Bereitstellung wird fortgesetzt, bis der neue Bereitstellungsrahmen „zum Einsatz“ kommt.

Der Anfang und das Ende einer Softwarebereitstellung sind aus der Sicht eines beteiligten Akteurs und einer akteursübergreifenden Sicht auf die Softwarebereitstellung unterschiedlich:

- *Akteur*: Für einen Akteur stellt sein Eintritt in das Bereitstellungsnetzwerk den *Beginn* für die Softwarebereitstellung dar. In der nachfolgenden Herstellung muss mit allen beteiligten Akteuren, also auch ihm, der Bereitstellungsrahmen an die neue Situation angepasst werden. Das *Ende* eines beteiligten Akteurs an der Softwarebereitstellung ist für ihn erreicht, wenn er sich aus der Bereitstellung zurückzieht oder von anderen Akteuren ausgeschlossen wird. In der folgenden Herstellung eines neuen Bereitstellungsrahmens ist er nicht mehr im Bereitstellungsrahmen vorhanden (*formales Ende*).

- *Akteursübergreifende Sicht*: Aus einer „Metasicht“ stellt die erste Etablierung des ersten Zyklus, welcher zur ersten Herstellung eines ersten Bereitstellungsrahmens führt, den *Beginn* der Bereitstellung dar. Das *Ende* der Bereitstellung tritt ein, wenn es aufgrund eines instabilen Zustands keine weitere Etablierung eines neuen Zyklus gibt. Das Akteursnetzwerk zerfällt, alle Akteure geben ihre kollektiven Rollen auf.

In der zyklischen Gestaltung der Softwarebereitstellung bestehen folgende Voraussetzungen:

1. Nur die beteiligten Akteure haben Einfluss auf die Softwarebereitstellung und damit Gestaltungsmacht. Äußere Einflüssen wirken über die Akteure auf die Bereitstellung.

2. Diese Macht haben sie zu jedem Zeitpunkt. Entweder wird der Bereitstellungsrahmen durch ihr Handeln instabil oder sie berufen eine konstituierende Phase ein und erarbeiten (verhandeln) einen neuen, stabileren Bereitstellungsrahmen.

3. Die beteiligten Akteure gestalten die Bereitstellung zu jedem Zeitpunkt. Durch ihr Handeln bestätigen sie entweder die vereinbarte Struktur in Form des Bereitstellungsrahmens oder handeln ihm zuwider, was Nicht-Handeln einschließt und den Bereitstellungsrahmen instabil werden lässt.

In diesem Sinne sind immer alle beteiligten Akteure zu jedem Zeitpunkt an der Gestaltung der Softwarebereitstellung beteiligt, und da sie Veränderungen unterliegen, werden sie den vereinbarten Bereitstellungsrahmen in ihrem Handeln nicht immer bestätigen können. Durch die regelmäßige Etablierung eines neuen Zyklus und das regelmäßige Einfordern der Herstellung eines neuen Bereitstellungsrahmens werden alle beteiligten Akteure aktiv und bewusst in die Gestaltung der Softwarebereitstellung einbezogen. Außerdem werden Veränderungen frühzeitig erkannt, so dass einer Instabilität der Bereitstellung aktiv entgegengewirkt werden kann. Die Verantwortung dieser Treffen liegt bei allen beteiligten Akteuren, doch besonders beim Akteur, der die Rolle des Dienstleistungsanbieters einnimmt. Der Dienstleistungsanbieter ist die zentrale Figur im Bereitstellungsnetzwerk und seine Existenzberechtigung hängt von der Stabilität und Weiterentwicklung der Bereitstellung ab.

Verträge bzw. vertragliche Regelungen gelten als Rechtssicherheit. Sie dokumentieren, dass und wie die Vertragspartner miteinander kooperieren wollen. Im Hinblick auf das zyklische Vorgehen wird eine Kombination von langfristigen Rahmen- und kurzfristigen Detailverträgen vorgeschlagen. Mit langfristigen Rahmenverträgen wird die Kooperation der beteiligten Akteure auf eine für alle sichere Basis gestellt und die Bereitstellung

grundlegend begründet. Wie die Kooperation im Einzelnen organisiert werden soll, wird in kurzen Detailverträgen geregelt. Dabei müssen die Detailverträge nicht Zukünftiges regeln. Der Umgang mit Veränderungen wird im zyklischen Vorgehen in den Prozess gelegt, so dass neue kurze Detailverträge als Konsequenz der zyklischen Treffen entstehen. Ob die Detailverträge rechtlich einem Miet-, Pacht-, Leasing-, Werk- oder Dienstleistungsvertrag entsprechen, ist von Kooperation zu Kooperation verschiedenen. So ist es möglich und wahrscheinlich, dass verschiedene Vertragsarten in einem langfristigen Rahmenvertrag zur Softwarebereitstellung zu finden sind.

10.2 eASP in „Informatiksysteme in Organisationen“

Zur Einbettung von eASP in das Gebiet „Informatiksysteme in Organisationen“ wird eASP im Folgenden in den Methodenrahmen der Softwareentwicklung STEPS und in das Konzept der IT-unterstützten Organisationsgestaltung integriert.

10.2.1 eASP im Methodenrahmen der Softwareentwicklung STEPS

Da die Softwarebereitstellung Einfluss auf die Zufriedenheit der Nutzenden einer Software hat (Strauss u. a. 2003), muss die Softwareentwicklung die Erkenntnisse über eine Softwareversion immer im Verhältnis zur Bereitstellung der Version bewerten. Wie kann eASP der Softwareentwicklung helfen, diesem Umstand zu begegnen, und wie lässt sich die Softwarebereitstellung in STEPS (Abschnitt 5.2.1) integrieren?

Floyd und Züllighoven (2002) unterscheiden bei der Softwareentwicklung zwischen produktbezogene Tätigkeiten (Anforderungsermittlung, Systemdefinition, Entwurf, Implementierung, Validation, Systemeinführung und Wartung) und prozessbezogene Tätigkeiten, „die die produktbezogenen ermöglichen und die Koordination und Kooperation im Projekt, die Produktverwaltung und die Qualitätssicherung umfassen“ (Floyd und Züllighoven 2002, S. 755). In den produktbezogenen Tätigkeiten Systemeinführung und Wartung finden sich Aufgaben der Bereitstellung. Zur Systemeinführung gehören die Installation der Software(-version), organisatorische Umstellungen, Schulung der Nutzenden und die Erarbeitung der technischen und der Benutzerdokumentation. Die Wartung umfasst die Fehler-

bereinigung der aktuellen Version, die sich von der Weiterentwicklung einer neuen Version unterscheidet, sich aber in der Praxis nicht scharf trennen lässt (Floyd und Züllighoven 2002, S. 776).

Aus Sicht der Softwarebereitstellung fehlen Aufgaben in der Benutzerbetreuung, die ein kontinuierliches Engagement erfordern und nicht mit einer einmaligen Schulung abgeschlossen sind. Dauerhaft angebotene E-Mail- oder Telefon-Hotlines, als Ergänzung oder Weiterführung der Schulung, sind darauf ausgerichtet, Nutzenden in Problemsituationen direkt und schnell helfen zu können. Darüber hinaus muss in der Nutzung eine zweite Validation der Softwareversion erfolgen, indem die Nutzung evaluiert wird. Ebenfalls fehlen Aufgaben im technischen Betrieb, denn nicht nur Fehlerbereinigungen müssen neu installiert werden, auch die technische Infrastruktur, auf der die Software(-version) aufsetzt, muss gewartet werden. Es wird deutlich, dass die in eASP dargestellten übergeordneten Aufgabengebiete Betrieb und Benutzungsbetreuung mit ihren Aufgabengebieten und Aufgaben die produktbezogenen Tätigkeiten Systemeinführung und Wartung erweitern und ergänzen. Daher kann die Softwarebereitstellung, als übergreifende Aufgabe, den Platz dieser produktbezogenen Tätigkeiten einnehmen. Damit ergänzt die Softwarebereitstellung (bzw. große Teile ihrer übergeordneten Aufgaben) die Softwareentwicklung auf der Ebene der produktbezogenen Tätigkeit.

Im zyklischen Projektmodell von STEPS ordnet sich die Softwarebereitstellung im Bereich Einsatz neben dem Bereich Nutzung ein und ersetzt die Pflege (Aufgabe der Entwickler). Da die Bereitstellung ein wesentlich umfangreicheres Aufgabenverständnis hat als die Pflege, ist die Benutzerbetreuung nicht mehr nur eine Aufgabe der Entwickler. Sie ist eine gemeinsame Aufgabe von Nutzern (z. B. in der Nutzungsevaluation), Entwicklern (z. B. in der Fehlerbereinigung) und der weiteren, in eASP genannten Personen bzw. funktionellen Rollen: Benutzerbetreuer, Hardware- und Softwareadministrator, Nutzungsevaluator usw. (Abschnitt 10.1.1). Die Bereitstellung im Sinne von eASP verleiht der zyklischen Softwareentwicklung einen umfassenderen Blick auf die Nutzung der eingesetzten Systemversionen, so dass zum einen der Einsatzkontext als Qualitätskriterium erschlossen, zum anderen die Bewertung einer Softwareversion in Relation zu ihrer Bereitstellung erfolgen kann. Wenn z. B. der Aufwand für den Benutzersupport einer Softwareversion sehr hoch ist, dann liegt es nahe, dass die

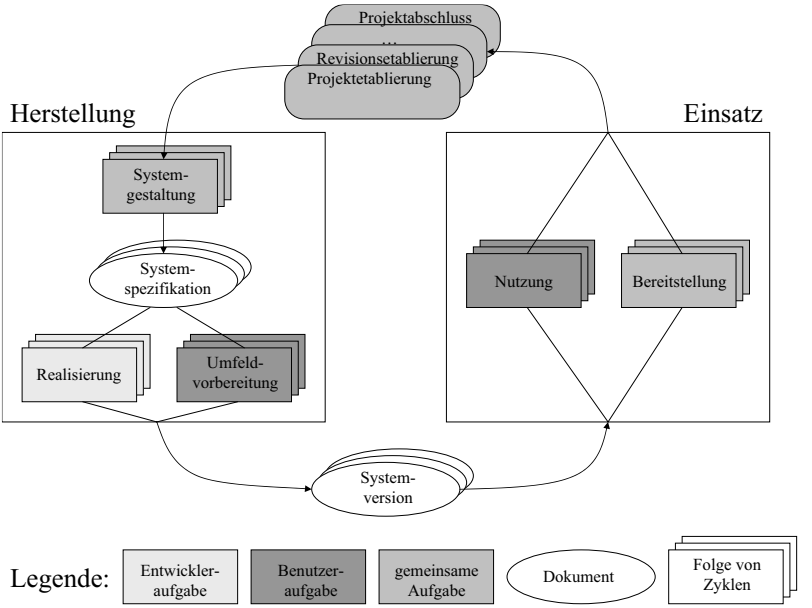

Abbildung 34: Zyklisches Projektmodell von STEPS mit Softwarebereitstellung

Softwareversion (noch) zu komplex und damit unbrauchbar ist, auch wenn sie allen formalen Anforderungen genügt.

Durch die Erweiterung des zyklischen Projektmodells durch die Bereitstellung wird die Verschränkung von Herstellung und Einsatz (Floyd und Züllighoven 2002, S. 779) in der zyklischen Softwareentwicklung nochmals hervorgehoben.

Die zyklische Darstellung im Projektmodell suggeriert eine lineare Anordnung: Versionen werden entwickelt, eingeführt und bereitgestellt. Aus der Nutzung ergeben sich Weiterentwicklungsbedürfnisse, die in die Entwicklung zurückfließen, und dort zu einer neuen Version führen. Die neue Version löst die alte ab und der Zyklus beginnt von neuem. Die Betonung der Bereitstellung macht deutlich, dass sie nicht mit dem Beginn der Entwicklung einer neuen Version aufhört, sondern weiterläuft, bis die alte Version entsorgt und durch eine neue ersetzt wird. d. h. die Bereitstellung der aktuellen Version und die Entwicklung der neuen Version laufen parallel.

Daraus folgt, dass zu einem beliebigen Zeitpunkt innerhalb des Zyklus einer Version immer der Zyklus einer anderen Version noch bzw. schon läuft:

1. *Entwicklung*: Während der Entwicklung von Version n wird die Version n-1 noch bereitgestellt, sofern die Version n nicht die erste Version war.

2. *Einführung*: Bei der Einführung von Version n muss Version n-1 entsorgt werden, sofern die Version n nicht die erste Version war.

3. *Bereitstellung*: Während der Bereitstellung von Version n wird bereits an der Entwicklung von Version n+1 gearbeitet, sofern Version n nicht die letzte Version war.

4. *Entsorgung*: Zum Zeitpunkt der Entsorgung von Version n wird Version n+1 eingeführt, sofern Version n nicht die letzte Version war.

Diese Betrachtungsweise führt zu einer Vielzahl von mit- und ineinander verschränkten Zyklen der verschiedenen Softwareversionen.
Resümierend ist festzustellen, dass die Softwareentwicklung, auf der Ebene von produktbezogenen Tätigkeiten, gut durch die Softwarebereitstellung bzw. durch bestimmte übergeordnete Aufgaben der Bereitstellung erweitert werden kann. Darüber hinaus hilft sie in zyklischen Softwareentwicklungsmodellen die Software im Kontext und in Relation zur Qualität der Bereitstellung bewerten zu können.

10.2.2 eASP in der IT-unterstützten Organisationsgestaltung

Ein idealtypischer Gestaltungsprozess in der IT-unterstützten Organisationsgestaltung (Abschnitt 5.1.1) besteht aus dem Verstehen der IT-unterstützten Organisationssituation und der IT-unterstützten kooperativen Organisationsgestaltung (Rolf 1998, S. 226).

Beim Verstehen der IT-unterstützten Organisationssituation werden die Ablauf- und Aufbauorganisation sowie der informationstechnische Stand der Organisation evaluiert. Mit Organisationsworkshops sollen die zumeist aus der Top-Down-Sicht evaluierten Ergebnisse in den Arbeitsgruppen mit den anderen Perspektiven und Sichtweisen verknüpft werden. Der Übergang vom Verstehen der Ist-Organisationssituation zum Soll ist fließend:

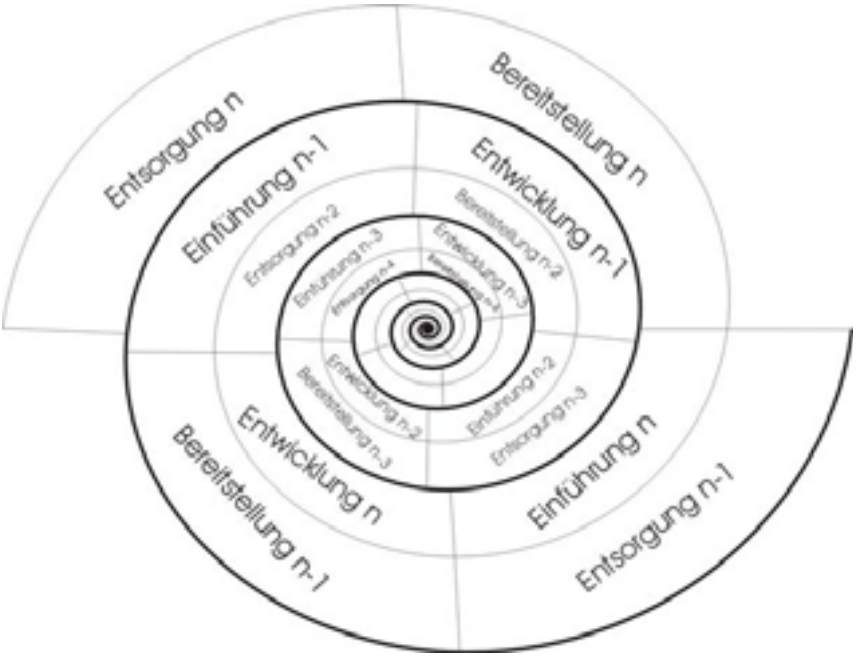

Abbildung 35: Spirale von Versionen bei der zyklischen Softwareentwicklung

Organisationsworkshops stoßen bei der Arbeitsgruppe die Suche nach Verbesserungen und Optionen an. Zugleich werden immer wieder Bezüge zum Ist hergestellt, um Annahmen abzusichern oder um prototypische Lösungen zu erproben (Rolf 1998, S. 226f.).

Die IT-unterstützte kooperative Organisationsgestaltung beginnt beim Erarbeiten von Organisations- und IT-Optionen. Der Diskurs über Organisations- und IT-Optionen muss schließlich in die Entwicklung der Systemvision und eines Ausbaustufenplanes münden. Der Ausbaustufenplan legt den Grad der Systemunterstützung fest und strukturiert die Einführungsreihenfolge. Das Modell sieht in der IT-unterstützten kooperativen Organisationsgestaltung weiterhin die Punkte der Softwareentwicklung bzw. des Customizing und die Punkte der Implementierung und Benutzerschulung vor, auf die das Modell allerdings nicht weiter eingeht (vgl. Rolf 1998, S. 227f.). Im Sinne eines zyklischen Vorgehens fehlt im Gestaltungsprozess

eine Verbindung, um wieder mit dem Verstehen der IT-unterstützten Organisationssituation beginnen zu können.

Da der Wert bzw. der Nutzen der gestalteten IT-unterstützten Organisation sich erst in der Nutzung bzw. dem Einsatz erweist und insbesondere dort die Wechselwirkungen zwischen Organisations- und Technikoptionen sichtbar werden, muss dieser Abschnitt in den Gestaltungsprozess integriert werden. Der erweiterte Gestaltungsprozess sähe dann wie folgt aus (vgl. Rolf 1998, S. 160):

1. Verstehen der IT-unterstützten Organisationssituation
2. IT-unterstützte kooperative Organisationsgestaltung
    - Erarbeitung von Organisations- und Technikoptionen
    - Systemvisionen und Ausbaustufen
    - Systementwicklung bzw. Customizing
    - Implementierung und Benutzerschulung
3. Nutzung der IT-unterstützten Organisation

So schließt sich an das Verstehen und Gestalten die Nutzung an, in der die IT-unterstützte Organisation zum Einsatz kommt. Aufgaben sind u. a. die IT-Unterstützung im Sinne von eASP bereitzustellen. Durch die Nutzung wird die Lücke zwischen der Implementierung und Benutzerschulung und dem Verstehen der IT-unterstützten Organisationssituation im Gestaltungsprozess geschlossen und der Prozess zu einem Zyklus verbunden.
Wie in zyklischen Softwareentwicklungsmodellen gilt auch hier, dass Verstehen und Gestalten einer IT-unterstützten Organisation mit der Nutzung verschränkt sind. Während die aktuelle IT-unterstützte Organisation sich in der Nutzung befindet und ihr im Laufe der Zeit Veränderungen widerfahren, wird versucht diese Veränderungen zu verstehen und auf dem Verständnis aufbauend eine neue IT-unterstützte Organisation zu gestalten. In einem fließenden Übergang wird sie in die Nutzung gebracht und die alte ersetzt. Es entstehen, ähnlich wie in der zyklischen Softwareentwicklung, mit- und ineinander verschränkte Zyklen (Abschnitt 10.2.1).

Mit eASP und den Ausgestaltungen seiner vier Perspektiven Aufgaben, Organisation, Technik und Vorgehen kann in der Nutzung der IT-

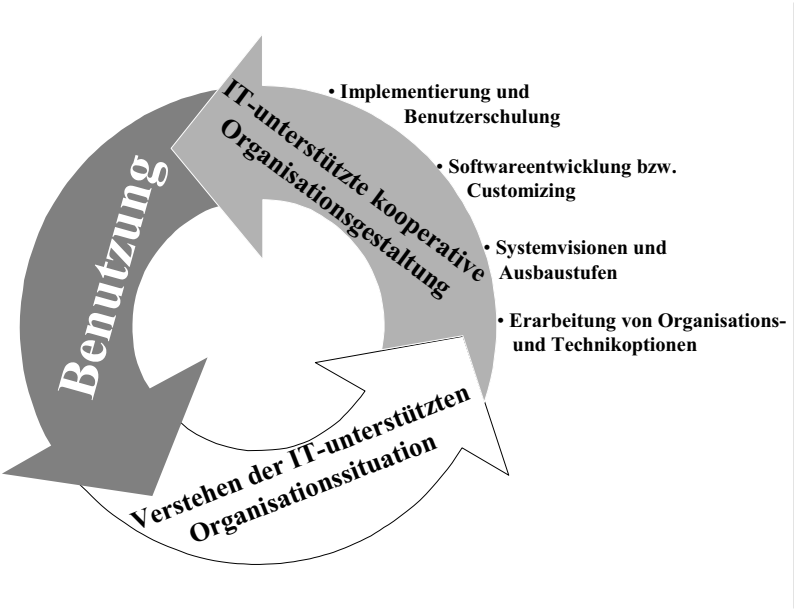

Abbildung 36: Erweiterte IT-unterstützte Organisationsgestaltung als Kreislauf

unterstützten Organisation die Bereitstellung der Informationstechnik gestaltet werden. Darüber hinaus werden durch eASP Aspekte in der Nutzung der IT-unterstützten Organisation sichtbar, die Grundlage für ein erneutes Verstehen und Gestalten, also einen neuen Zyklus, sein können. So ist zusammenfassend festzustellen, dass sich eASP hinsichtlich seiner gerade skizzierten Leistungen hervorragend in den Gestaltungsprozess der IT-unterstützten Organisationsgestaltung integrieren lässt.

10.3 eASP in der Praxis

Die Bewertung der Nutzbarkeit von eASP für die Praxis wird anhand der Erweiterung von ASP und der Verallgemeinerbarkeit von eASP diskutiert.

10.3.1 eASP als Erweiterung von ASP

Die Entstehung und Verbreitung von ASP (Kapitel 4) als (aus der Sicht eines Kunden) Outsourcing der Bereitstellung von Anwendungen bzw. als (aus der Sicht eines Dienstleistungsanbieters) Geschäftsmodell für die Bereitstellung von Anwendungen für unternehmensexterne Kunden wird als logischer Schritt nach der raschen und fortschreitenden Entwicklung in der Internettechnologie gesehen (vgl. Rolf 2004a). Die Einschränkung von ASP per definitionem auf Softwarebereitstellungsszenarien, bei denen der Kunde bzw. der Nutzer und der Dienstleistungsanbieter immer aus unterschiedlichen Unternehmen stammen müssen, ist eine falsche Schlussfolgerung aus der rasanten Entwicklung des Internets. Vielmehr ist durch die Technikentwicklung der physische Standort eines Knotens im Netzwerk (siehe Abschnitt 5.1.3) obsolet geworden. ASP kann daher unabhängig von Insourcing-Outsourcing-Diskussionen betrachtet werden. Letztendlich sind Produktions- und Transaktionskosten ausschlaggebend bei der Insourcing-Outsourcing-Entscheidung (vgl. Lammers 2004).

In diesem Sinne kann der in dieser Arbeit vorgestellte Ansatz zur Softwarebereitstellung als Erweiterung von ASP angesehen werden. eASP fundiert ASP durch das Heranziehen von Methoden und Modellen aus dem Bereich „Informatiksysteme in Organisationen“. Es erweitert ASP nicht nur um die zyklische Vorgehensweise, sondern detailliert ASP in vielen Bereichen.

eASP befreit ASP aus der Umklammerung des Outsourcing und führt ASP seiner wirklichen Bedeutung zu, die sich aus den Begriffen seines Namens definieren: Bereitstellung (Providing) einer Anwendung (Application) als Dienstleistung (Service) (vgl. Picot und Jahn 2000; Riemer und Ahlemann 2001; Rosenhagen 2002). Insofern ist eASP nicht nur als Erweiterung von ASP zu begreifen, sondern als ein umfassenderer Ansatz zur Analyse und Gestaltung von Bereitstellungskontexten. Das bisher in der Literatur mit ASP bezeichnete Geschäftsmodell wird durch eASP zum Application Outsourcing (TripleTree 2003) degradiert. Es entspricht damit einer Möglichkeit der Organisation von Bereitstellungsnetzwerken, die wie oben beschrieben dadurch gekennzeichnet ist, dass Kunde und Nutzer auf der einen Seite und der Dienstleistungsanbieter und seine Partner auf der anderen Seite immer aus unterschiedlichen Unternehmen stammen. Daneben betrachtet eASP auch die Organisation von Bereitstellungsnetzwerken,

in denen dies nicht der Fall ist. Bei eASP werden somit nicht per Definition verschiedene Organisationsformen ausgeschlossen.

10.3.2 Verallgemeinerbarkeit von eASP

Es stellt sich nun die Frage ob eASP, über die beiden Fallstudien hinaus, unabhängig von der bereitgestellten Software CommSy, den beteiligten Akteuren und dem universitären Bildungskontext seine Gültigkeit behält.

Diese Arbeit fokussierte von vornherein auf die zentrale Bereitstellung einer Software für viele Kunden bzw. Nutzende. Diese Sichtweise auf die Bereitstellung von Software grenzt sich von einer schlichten Downloadmöglichkeit mit anschließender lokaler und autarker Installation ab und stellt damit eine Einschränkung der Verallgemeinerbarkeit von eASP auf sämtliche Softwarebereitstellungskontexte dar.

Die Verallgemeinerbarkeit von eASP bei Szenarien, in denen zentral eine Software für viele bereitgestellt wird, kann als hoch bewertet werden, da sich, wie im vorigen Abschnitt dargestellt, eASP auf ASP gründet, eine Weiterentwicklung dessen darstellt und ASP in der Wirtschaft in diesen Kontexten weit verbreitet ist. Insofern kann eASP als Muster für alle Bereitstellungskontexte herangezogen werden, in denen auch ASP zur Anwendung kommen kann. Dies sind insbesondere Kontexte, in denen Software in einem Outsourcing-Szenario genutzt werden. Für Insourcing-Szenarien, in denen die Softwarebereitstellung in einer Organisation für die Organisation stattfindet, kann die Übertragbarkeit ebenfalls als hoch angesehen werden, da sich die Bereitstellung von Software beim Outsourcing und Insourcing nur in der organisatorischen Verankerung der beteiligten Akteure unterscheidet. Diese Bewertung muss mit der Anwendung von eASP auf entsprechende Bereitstellungsszenarien noch eruiert werden.

Die Anwendung von eASP auf Softwarebereitstellungskontexte bedeutet konkret, dass beteiligte Personen und Akteure die in eASP dargestellten Perspektiven einnehmen und danach deren Softwarebereitstellung gestalten können. Zur Gestaltung gehört z. B. die Aufgabenstruktur anzupassen und diese mit entsprechend beteiligten Akteuren über die Rollen und im Sinne des kollektiven Rollennetzes zu verknüpfen bzw. das zyklische Vorgehen zu initiieren.

- 11 -
Resümee und Ausblick

Diese Arbeit stellt eine Synthese dar, resultierend aus Erfahrungen bei der fünfjährigen Bereitstellung von CommSy. Dazu gehörten das Lesen diverser Artikel zur Bereitstellung und Entwicklung von Software und Organisationen, der Diskurs mit Kollegen und der Wunsch, die Bereitstellung von Software besser begreifen zu können. Das Aufgreifen verschiedener Gedanken, das Ausprobieren und das Fallenlassen zahlreicher Ideen führten zum Ansatz eASP.

eASP versucht mit Technikperspektive, Organisationsperspektive und Dienstleistungsperspektive sowie mit einer zyklischen Vorgehensweise die zentrale Bereitstellung einer Software für viele Nutzende zu analysieren, den Bereitstellungsprozess transparent zu gestalten, die Komplexität zu reduzieren und Ansatzpunkte zur Gestaltung der Bereitstellung zu liefern.

Zum Abschluss wird das mit meiner Arbeit Erreichte zusammengefasst, kritisch bewertet und Zukünftiges in den Blick genommen.

11.1 Zusammenfassung des Erreichten

Hauptanliegen dieser Arbeit ist die Entwicklung eines methodisch fundierten Ansatzes für das Praxisphänomen ASP, um es für die Informatik nutzbar zu machen. Die Umsetzung dieses Anliegens geschieht unter Berücksichtigung existierender Methoden und Modelle aus dem Gebiet „Informatiksysteme in Organisationen“, um den Ansatz in diese Bereiche der Informatik zu integrieren.

Ein Beitrag dieser Arbeit zur Informatik besteht darin, die Softwarebereitstellung für die Informatik zu thematisieren und sie in zwei Bereichen in den informatischen Diskurs einzubeziehen: in die Gestaltung von Software und Organisationen und in die Nutzung dieser Strukturen. Für Softwareentwicklungsprozesse und Prozesse zur IT-unterstützten Organisationsgestaltung stellt eASP eine Erweiterung dar, mit der in Gestaltungs- und Entwicklungsprozessen die Bereitstellung von Software betrachtet und konstruktiv für die Prozesse nutzbargemacht werden kann. Für die Nut-

zung von Software und Software-Infrastrukturen bietet eASP Ansätze zur Erklärung und Gestaltung, so dass eine Nachhaltigkeit in der Softwarebereitstellung für die Nutzenden gesichert werden kann. Mit eASP wird für die Informatik die Zeit nach der Entwicklung von Software und nach Ende eines Gestaltungsprojekts und -prozesses sichtbar. Das Verständnis von bzw. konkrete Erfahrungen in Softwarebereitstellungskontexten können so auf Softwareentwicklungsmethoden zurückwirken.

Zum Aufbau des Ansatzes eASP wurde das in der Wirtschaft bekannte Application Service Providing (ASP) herangezogen und auf die Informatik übertragen. So ist als ein Beitrag dieser Arbeit die methodische Auseinandersetzung von ASP innerhalb des Gebiets „Informatiksysteme in Organisationen“ zu sehen. Die Übertragung erforderte eine Fundierung und Erweiterung von ASP mit verschiedenen Ansätzen aus diesem Gebiet. Neben der Übertragung liefert diese Arbeit daher auch eine Fundierung von ASP und erweitert ASP zu dem Ansatz eASP.

Der Softwarebereitstellungsansatz eASP stellt einen weiteren Beitrag dar, der sich mit seinen Perspektiven und der zyklischen Vorgehensweise an Personen richtet, die Softwareentwicklungsprozesse oder Softwarebereitstellungsprojekte organisieren wollen oder müssen. Durch Technik-, Organisations- und Dienstleistungsperspektive sowie der vorgeschlagenen zyklischen Vorgehensweise werden den beteiligten Akteuren Ansatzpunkte zum Verstehen und Gestalten aufgezeigt, so dass sie miteinander die Bereitstellung von Software kooperativ leisten bzw. nutzen können.

Ein methodischer Beitrag dieser Arbeit ist die Verknüpfung von Aufgabenanalyse und Akteursmodell mit Hilfe des in dieser Arbeit vorgestellten kollektiven Rollennetzes. Die kollektive Rolle macht sich die Flexibilität der funktionellen Rolle auf der Ebene von Akteuren zunutze. So lässt sich die Bereitstellung von realen Akteuren abstrahieren und ein kollektives Rollennetz gestalten. Das kollektive Rollennetz liefert Ansatzpunkte zur Gestaltung und spiegelt die Organisation der Bereitstellung wider.

11.2 Kritische Bewertung des Erreichten

Der vorgestellte Ansatz zur Softwarebereitstellung eASP ist ein Beitrag zur Systematisierung von Softwarebereitstellungen und lässt einen umfassenden Blick auf Softwarebereitstellungskontexte zu. Wie lässt sich eASP aber in der Praxis verwenden? Ist die mit eASP sichtbar gewordene Komplexität

der Softwarebereitstellung in der Praxis handhabbarer oder ist die aufgebaute Systematisierung eher realitätsfern und nur theoretisch gebrauchstauglich?

Ansätze und Modelle versuchen komplexe Situationen zu erklären, indem sie durch bewusstes Ausblenden und Vereinfachen die Komplexität reduzieren. Die dadurch gewonnene Handhabbarkeit und das gewonnene Verständnis werden mit dem Nachteil erkauft, dass Ansätze und Modelle nicht die Realität darstellen, sondern nur ein vereinfachtes Abbild.

eASP stellt somit nicht die Realität sondern nur ein vereinfachtes Abbild von Softwarebereitstellungen dar. Die Verflechtungen auf technischer Ebene sind in der Praxis wesentlich komplexer. Bei der Benennung von Aufgaben kann nicht von einer Vollständigkeit gesprochen werden. Die Beziehungen zwischen Akteuren sind ungleich politischer als in eASP dargestellt. Die Möglichkeiten der Abrechnung und der juristischen Fallstricke sind vielschichtiger und das Vorgehen wird von zahllosen Veränderungen beeinflusst. Auch wenn eASP die gesamte Komplexität von Softwarebereitstellungen nicht erfassen kann, so macht der Ansatz dennoch auf verschiedene Aspekte aufmerksam. Er gibt einen Rahmen und durch seine Perspektiven eine Systematisierung vor, mit der verschiedene Szenarien der Softwarebereitstellung, mit Einschränkungen hinsichtlich der Vollständigkeit, analysiert und bewertet werden können.

eASP richtet sich vorrangig an Unternehmen oder Abteilungen, die als Dienstleister in einem Bereitstellungsnetzwerk agieren bzw. an die entsprechenden Personen, die die Softwarebereitstellung koordinieren und organisieren. Mit der Systematisierung von eASP erlangen diese Personen die Möglichkeit gestaltend, auf die Softwarebereitstellung einzuwirken.

Dabei bestehen Unterschiede in Organisation und Durchführung der Bereitstellung. Auf der theoretischen Ebene ist die Zuordnung von einer oder mehreren funktionellen Rollen zu einer konkreten Person etwas anderes als die Ausführung der entsprechenden Aufgaben durch die Person. Im Realen werden verschiedene Personen funktionelle Rollen wahrnehmen, indem sie die zugehörigen Aufgaben übernehmen und nicht indem eine theoretische Zuordnung erfolgte. Erst durch das Handeln der konkreten Personen wird die auf einer theoretischen Ebene organisierte Bereitstellung auch praktisch zu einer realen Bereitstellung (vgl. Giddens 1997; Ortmann u. a. 1997; Orlikowski 1992; Pape 2004).

Insofern stellt die mit eASP vorgestellte Struktur, die, wie erläutert, nur ein Modell und kein getreues Abbild der Realität ist, nur eine Seite der Bereitstellung dar, in Ergänzung zur Durchführung bzw. dem Handeln. In diesem Sinne ist eASP als Heuristik zu begreifen, die interessierten Personen einen Einblick in Softwarebereitstellungskontexte bietet und Gestaltungsideen offeriert, ohne den Anspruch zu erheben, ein realistisches Abbild von real existierenden Bereitstellungsszenarien zu sein. Aufgaben, funktionelle und/oder kollektive Rollen werden sich von Bereitstellungsszenario zu Bereitstellungsszenario unterscheiden. Dem Ansatz eASP ist nicht wichtig, die Bereitstellung von Software eins zu eins wiederzugeben, sondern auf verschiedene Möglichkeiten, wie Aufgaben, Rollen usw., hinzuweisen und konkrete Beispiele zu geben, mit denen sich die Bereitstellung einer Software erschließen lässt.

11.3 Offene Fragen – ein Blick nach vorne

Der hier erarbeitete Ansatz zur Softwarebereitstellung eASP muss hinsichtlich seiner Validität überprüft werden. Dies bedeutet, eASP auf weitere Bereitstellungsszenarien über CommSy hinaus anzuwenden. Folgende Dimensionen sind zu beachten:

- *Software*: In der Softwaredimension kann die Art der Software verändert werden. Wie gestaltet sich die Bereitstellung von anderer kooperativer Software, Software für einzelne Personen und hochkomplexe Wirtschafts- und Informationssysteme?

- *Einsatzkontext*: In der Dimension des Einsatzkontextes können verschiedene Kontexte betrachtet werden: andere universitäre oder Bildungskontexte, wirtschaftliche oder private Kontexte. Greift eASP auch bei diesen Kontexten?

Eine systematische Betrachtung von weiteren Softwarebereitstellungskontexten ermöglicht die Vertiefung und Verfeinerung der einzelnen Perspektiven und Aspekte von eASP. Dies führt zu größerer Transparenz und zu besserem Verständnis von Bereitstellungskontexten und bildet schließlich die Grundlage zur Validation des hier vorgestellten Softwarebereitstellungsansatzes eASP.

Darüber hinaus beschäftigt sich eASP nur mit der Betrachtung und Gestaltung einer Software bzw. eines Bereitstellungskontextes. Wie verhält sich die Softwarebereitstellungssituation bei mehreren gleichzeitig bereitgestellten Anwendungen? Soll und kann die Betrachtung von multiplen Bereitstellungskontexten in eASP integriert werden? Kann und sollte eASP als Ansatz für einen Bereitstellungskontext sinnvoll in Softwareinfrastrukturansätze eingebettet werden? Insbesondere die Einbettung in Ansätze zu Softwareinfrastrukturen ist zukünftig detailliert zu prüfen.

Verquickung von Softwareentwicklung und -bereitstellung

Der vorgestellte Bereitstellungsansatz eASP enthält ein zyklisches Vorgehen, bei dem sich die Herstellung und Nutzung eines Bereitstellungsrahmens gegenseitig bedingen. eASP beabsichtigt, alle beteiligten Akteure einzubeziehen, und organisiert die Bereitstellung in Aufgaben, funktionellen Rollen, Akteuren, Akteursnetzwerken usw. Ähnliches ist in zyklischen Projektmodellen zur Softwareentwicklung enthalten. Im Grunde unterscheiden sich zyklische Softwareentwicklungsmodelle und eASP methodisch kaum. eASP betrachtet den Prozess aus Sicht der Bereitstellung, Softwareentwicklungsmodelle betrachten ihn dagegen aus Sicht der Entwicklung. Da das eine im anderen, in Form eines Handlungsfeldes oder Aufgabengebiets, vorhanden ist und eine augenscheinlich starke Ähnlichkeit besteht, wäre für die Informatik interessant, einen Ansatz zu finden, der die Softwareentwicklung und -bereitstellung nicht in zeitlich aufeinander folgende Phasen trennt. Der im Folgenden kurz vorgestellte „Prioritätenansatz“ könnte ein solcher Ansatz sein.

Die Idee beim Prioritätenansatz ist nicht, wie in zyklischen Modellen bei der Linearität anzusetzen, sondern bei der Abgeschlossenheit der verschiedenen Handlungsfelder bzw. Phasen; in diesem Fall der Entwicklung und Bereitstellung (vgl. Pape 2004).

Dem Prioritätenansatz liegen drei wesentliche Überlegungen zugrunde:

1. Sämtliche Handlungsfelder bzw. (ehemalige) Phasen werden aufgetrennt und die enthaltenen Aufgaben in einer gemeinsamen Menge gesammelt.

2. Der Lebenszyklus einer Software besteht aus verschiedenen Lebensabschnitten, in denen alle in der gemeinsamen Aufgabenmenge vorhandenen Aufgaben durchgeführt werden können.

3. Alle durchgeführten Aufgaben sind, je nach Lebensabschnitt, unterschiedlich wichtig. Das bedeutet, dass eine nach Prioritäten geordnete Aufgabenliste zu jedem Zeitpunkt erstellt werden kann.

Ein Lebensabschnitt lässt sich nach der Priorisierung der in ihm durchgeführten Aufgaben benennen. Ein Lebensabschnitt würde „Entwicklung“ benannt werden, wenn darin überwiegend Entwicklungsaufgaben durchgeführt werden, bzw. „Bereitstellung“, wenn darin überwiegend Bereitstellungsaufgaben gelöst werden.

Bei dieser Betrachtung ist festzustellen, dass sich in den Lebensabschnitten die primären Aufgaben der einzelnen Abschnitte in jeweils anderen Lebensabschnitten in untergordneter Priorität wieder finden lassen. Für die Bereitstellung von Software bedeutet dies, dass Aufgaben der Bereitstellung nicht nur im Lebensabschnitt „Bereitstellung“ wahrgenommen werden müssen, sondern auch in anderen Lebensabschnitten. Umgekehrt spielt im Abschnitt „Bereitstellung“ nicht nur die in dieser Arbeit untersuchte Softwarebereitstellung eine Rolle, sondern auch die Aufgaben aus anderen Handlungsfeldern, z. B. der Softwareentwicklung.

Der Prioritätenansatz kann sowohl auf die Software als Ganzes als auch auf die Betrachtung von Softwareversionen oder auf bestimmte Kontexte eingesetzt werden, sofern deutlich wird, auf welcher Ebene die Betrachtung stattfindet. Wiederholungen von Lebensabschnitten sind nicht ausgeschlossen, sondern eher wahrscheinlich. Wichtig ist, dass in einem bestimmten Lebensabschnitt potenziell alle Aufgaben durchgeführt werden können. Priorität haben dabei bestimmte Aufgaben aus dem Handlungsfeld welches der Namensgeber für den Lebensabschnitt ist.

Dieser Ansatz erscheint mir für die Verquickung von Softwareentwicklung und Softwarebereitstellung geeignet.

IV

Anhang

- A -
Aufgabenperspektive

Präsentiert werden im Folgenden die in der Aufgabenperspektive (Kapitel 6) nicht ausführlich beschriebenen Aufgabengebiete, Aufgaben und funktionellen Rollen.

A.1 Aufgabengebiete in der Softwarebereitstellung

Die Aufgabengebiete im Detail in alphabetischer Reihenfolge (in Klammern sind die zugehörigen übergeordneten Aufgaben aufgeführt):

- *Abrechnung (Kooperation)*: Die Abrechnung umfasst die Verwaltung des Zahlungsverkehrs zwischen allen Dienstleistern. Aufgaben: Rechnung stellen; Zahlungsverkehr überwachen.
- *Abrechnung (Kundenbetreuung)*: Entsprechend den abgeschlossenen Verträgen müssen erbrachte Leistungen Kunden in Rechnung gestellt und diese Rechnung verschickt werden. Außerdem müssen die Zahlungseingänge überwacht und bei Zahlungssäumnissen ggf. mit Mahnungen reagiert werden. Aufgaben: Rechnungen stellen; Zahlungen überwachen.
- *Anwendung (Betrieb)*: Zum Betrieb der Anwendung gehört neben der Basissoftware (z. B. Betriebssystem) die bereitzustellende Anwendung selbst. Sie muss installiert, konfiguriert und regelmäßig aktualisiert werden. Aufgaben: Anwendung installieren, konfigurieren und pflegen.
- *Ausfallsicherheit (Betrieb)*: Betrifft die Sicherheit des Anwendungsservers an seinem Einsatzort. Kontinuierlich sicherzustellen sind u. a. ausreichende Kühlung, feuerfeste Unterbringung und ununterbrochene Stromversorgung. Aufgaben: Für Klima- und Brandschutz sorgen; Stromversorgung sichern.

- *Basissoftware (Betrieb)*: Zum Betrieb einer Software wird neben der Hardware auch Basissoftware benötigt. Sie muss installiert, konfiguriert und gewartet werden. Zur Wartung gehören u. a. das Einspielen von Updates, Sicherheitspatchen und Servicepacks bzw. die Migration auf neuere Versionen. Aufgaben: Zusätzlich benötigte Software pflegen.

- *Bedarfsanalyse (Marketing)*: Sie ermittelt den Bedarf der anzubietenden Dienstleistungen nebst Software bei der entsprechenden Zielgruppe, z. B. Markt oder Unternehmen. Darüber hinaus werden Konkurrenten und Konkurrenzprodukte sichtbar. Die Ergebnisse der Bedarfsanalyse müssen in Planung (Anwendungsentwicklung) und Werbung (Marketing) integriert werden. Aufgaben: Bedarfsanalyse konzipieren und durchführen; Ergebnisse in die Bereitstellung integrieren.

- *Datensicherheit (Betrieb)*: Datensicherheit bedeutet Sicherheit des Anwendungsservers auf Softwareebene. Auf dem Server gespeicherte Daten müssen mittels Back-up vor Datenverlust geschützt und die Verbreitung von Viren und anderem Schädlichen verhindert werden. Der Anwendungsserver muss mit entsprechenden Sicherheitsmechanismen (z. B. Intrusion Detection System (IDS)) ein unbefugtes Zugreifen auf die Daten unterbinden. Aufgaben: Back-up und Virenscan der Daten durchführen; notwendige Sicherheitsmechanismen einrichten.

- *Dokumentation (Anwendungsentwicklung)*: Eine nachhaltige Softwareentwicklung erfordert die Pflege einer Dokumentation in den Bereichen: Benutzer- und Entwicklerdokumentation. Die Benutzer benötigen die Dokumentation, um die Software sinnvoll einsetzen zu können. Den Entwicklern liefert sie Informationen darüber, wie und unter welchen Rahmenbedingungen die Software weiterentwickelt werden soll. Dies dient der Koordination der aktuellen Entwickler und dem Einbinden neuer Entwickler in den fortschreitenden Entwicklungsprozess. Aufgaben: Benutzerdokumentation und Entwicklerdokumentation pflegen.

- *Entwicklungsumgebung (Anwendungsentwicklung)*: Zur Entwicklung gehört der Betrieb einer Entwicklungsumgebung (Clients und Ser-

ver). In der Regel arbeiten mehrere Entwickler an einer Software, daher ist eine Versionsverwaltung erforderlich. Ebenso wird ein Bug-Tracker benötigt, um die gemeldeten Fehler besser verwalten zu können. Aufgaben: Pflege der Entwicklungsclients; des Entwicklungsservers; der zusätzlich benötigten Software (z. B. Versionsverwaltung, Bug-Tracker).

- *Evaluation der Nutzung (Benutzerbetreuung)*: Eine Evaluation der Nutzung macht die Gebrauchstauglichkeit der Software sichtbar. Ihre Ergebnisse geben Hinweise insbesondere für die Planung (Anwendungsentwicklung). Es sollten aber auch bei entsprechenden Ergebnissen andere Aufgabenbereiche kontaktiert werden: z. B. Handhabungssupport (Benutzerbetreuung). Aufgaben: Evaluation konzipieren und durchführen; Ergebnisse in die Bereitstellung integrieren.

- *Fehlermanagement (Anwendungsentwicklung)*: Fehler treten in der Benutzung auf und werden im der Regel vom Benutzer gemeldet. Dem Fehlermanagement obliegt die Aufgabe, dem Benutzer einen möglichst komfortablen Weg anzubieten, den Fehler anzuzeigen. Fehlermeldungen werden in die Fehlerverwaltung (Bug-Tracker) übernommen und zwecks Behebung an den Programmierer weitergegeben. Ist dies erfolgt, sollte derjenige, der den Fehler gemeldet hat, eine Information erhalten. Aufgaben: Hotline anbieten; Fehlerberichte in Entwicklungsprozess integrieren; Fehlerbehebung veröffentlichen.

- *Handhabungssupport (Benutzerbetreuung)*: Seitens der Benutzerbetreuung kann eine aktive Hilfe beim Umgang mit der Software in Form einer Hotline angeboten werden und eine passive in Form einer öffentlich zugänglichen und gepflegten FAQ. Aufgaben: Hotline anbieten; FAQ pflegen.

- *Hardware (Betrieb)*: Serverhardware, die zum Betrieb der Software benötigt wird, muss zur Verfügung gestellt und gewartet werden. Aufgaben: Serverhardware pflegen.

- *Kommunikation (Benutzerbetreuung)*: Zur Transparenz des Betreuungsangebotes und der zuständigen Ansprechpartner für den Benutzer ist ein „Betreuungsmarketing“ notwendig. Es sollten Informationsmaterialien zur Benutzerbetreuung erstellt und geeignet verteilt

werden (z. B. Aushänge, E-Mail, Webseite). Die Herausgabe regelmäßiger Newsletters, in denen auf kurzfristige Änderungen, besondere Angebote und Veranstaltungen hingewiesen wird, kann für das Betreuungsmarketing von Vorteil sein. Darüber hinaus fungiert die Benutzerbetreuung als Schnittstelle zwischen Nutzung und Entwicklung. Sie muss Fehlerberichte sowie Nutzerwünsche an die Anwendungsentwicklung weiterleiten. Aufgaben: Nutzerwünsche und Fehlerberichte weiterleiten; Informationsmaterial erstellen und verteilen; Informationsveranstaltungen und Workshops organisieren.

- *Kommunikation (Kundenbetreuung)*: Eine Hotline zur Kundenbetreuung sollte dem Kunden zur Klärung von Fragen und Problemen hinsichtlich der Nutzung des Angebotes zur Verfügung gestellt werden. Durch das Erstellen und Verschicken von Informationsmaterialien kann der Kunde informiert werden. Die Kundenbetreuung soll ein Vertrauensverhältnis zwischen Kunden und denjenigen, die die Software bereitstellen, aufbauen und den Kunden mit seinen Wünschen und Zielen in das Angebot integrieren. So müssen Kundenwünsche an entsprechende Stellen weitergeleitet werden. Aufgaben: Hotline (Telefon, E-Mail) anbieten; Informationsmaterialien entwickeln und verschicken; Kundenwünsche weiterleiten.

- *Kommunikation (Kooperation)*: Für die Kommunikation unter den Dienstleistern ist jeder gleichberechtigt verantwortlich. Das Aufgabengebiet umfasst die aktive Kommunikation, d. h. das Informieren aus eigenem Antrieb, und die reaktive Kommunikation, also das reagieren auf Handlungen anderer. Aufgaben: Interne Newsletter herausbringen; Feedback geben.

- *Koordination (Kooperation)*: Alle Dienstleister müssen sich hinsichtlich der Bereitstellung abstimmen und koordinieren. Aufgaben: Beteiligung an Koordinationstreffen; Überblick über Bereitstellung behalten.

- *Leitung (Anwendungsentwicklung)*: Die Entwicklungsleitung ist für die Weiterentwicklung der Software verantwortlich und übernimmt in dieser leitenden Funktion koordinierende und überwachende Aufgaben. Sie hat die endgültige Entscheidungsgewalt. Die an der Weiterentwicklung beteiligten Personen sind zu koordinieren, außerdem

ist die Einhaltung von Vereinbarungen (Programmierkonventionen, Softwarearchitektur, funktionelle Planung, strategische Planung usw.) zu überwachen. Aufgaben: Entwicklung koordinieren und überwachen; Entwicklungsentscheidungen treffen.

- *Netzsicherheit (Zugriffsermöglichung)*: Die Verbindung zwischen Anwendungsserver und dem Internet bzw. den Clients muss mittels Sicherheitsmechanismen (Firewall, IDS) gesichert werden, so dass nur authentisierte Benutzer an die Daten gelangen und kein unerlaubter Zugriff möglich ist. Aufgaben: Netzwerksicherheit installieren und pflegen.

- *Netzverbindung (Zugriffsermöglichung)*: Es muss eine stabile und schnelle Verbindung vom Anwendungsserver zu den Clients vorhanden sein. Da in der Regel über das Internet auf den Anwendungsserver zugegriffen wird, ist die Anbindung des Servers an das Internet performant und ausfallsicher zu gestalten. Aufgaben: Verbindung etablieren und pflegen.

- *Planung (Anwendungsentwicklung)*: Zur Entwicklungsplanung gehört das Analysieren von Anforderungen, um Bedürfnisse wahrzunehmen und darauf reagieren zu können. In Abstimmung mit Benutzer- und Kundenwünschen sowie durch Marktanalyse muss eine langfristige Planung erarbeitet werden, um auch Benutzern eine Perspektive und Sicherheit zu geben. Die langfristige, strategische Planung sollte in eine funktionelle münden, so dass sie von der Programmierung umgesetzt werden kann. Aufgaben: Anforderungen analysieren; Implementierungen planen; Zeitplan erstellen.

- *Programmierung (Anwendungsentwicklung)*: Die Entwicklung und Pflege einer wartbaren, wieder verwendbaren und erweiterbaren Softwarearchitektur sowie das eigentliche Programmieren neuer Features und die Fehlerbehebung gehören zur Programmierung. Aufgaben: Features programmieren; Fehler beheben; Softwarearchitektur pflegen.

- *Releasemanagement (Anwendungsentwicklung)*: Die Entwicklung einer Software beinhaltet das regelmäßige Herausbringen neuer Versionen. Ein Release besteht aus Codefragmenten in unterschiedlichen

Versionen. Aufgaben: Release identifizieren, zusammenstellen und veröffentlichen.

- *Training (Koordination)*: Das Aufgabengebiet Training soll alle Beteiligten in die Lage versetzen, die übergreifende Aufgabe *Software x bereitstellen* wahrnehmen zu können. Das bedeutet, dass in Workshops die Software im Detail vorgestellt wird und alle Beteiligten über ihre Leistungen und Möglichkeiten die anderen unterrichten. Aufgabe: Interne Workshops durchführen.

- *Verträge (Kundenbetreuung)*: Ebenfalls zur Kundenbetreuung gehören die Vertragsverhandlungen, in denen bei Bedarf auf vorgefertigte Standardverträge zurückgegriffen werden kann. Aufgaben: Standardverträge entwickeln; Vertragsverhandlungen führen.

- *Vertragswerk (Kooperation)*: Vertragliche Regelungen sind Grundlage für die Kooperation aller Dienstleister. Sie müssen ausgearbeitet, verhandelt und überwacht werden. Bei Vertragsbrüchen muss ein Lösungsprozess einsetzen. Aufgaben: Kooperationsverträge entwickeln; Vertragsverhandlungen führen; Vertragsbrüche verhandeln.

- *Werbung (Marketing)*: Um potentielle Kunden von der Benutzung von CommSy zu überzeugen, müssen Informationsmaterialien über das Angebot von CommSy erstellt und verteilt werden. Die Erstellung und Verteilung geschieht im Rahmen einer zuvor entwickelten Werbestrategie. Dazu gehören das Ansprechen von potentiellen Kunden und die Präsentation der Software und weiterer Dienstleistungen „vor Ort“. Aufgaben: Informationsmaterialien entwickeln und verbreiten; Software und Dienstleistungen präsentieren; Werbestrategie entwickeln.

A.2 Aufgaben in der Softwarebereitstellung

Folgende Aufgaben, dargestellt in alphabetischer Reihenfolge, sind in den im vorigen Abschnitt vorgestellten Aufgabengebieten enthalten (in Klammern sind die zugehörigen übergeordneten Aufgaben und Aufgabengebiete aufgeführt):

- *Anforderungen analysieren (Anwendungsentwicklung – Planung)* : Zur Weiterentwicklung müssen Anforderungen kontinuierlich analysiert werden, u. a. durch die Beobachtung des Marktes, aber auch durch die Bewertung von Anforderungen seitens der Kunden und Nutzer. Weiteren Input bietet eine geeignete Evaluation der Nutzung.

- *Anwendung installieren, konfigurieren und pflegen (Betrieb – Anwendung)*: Zum Betrieb einer Anwendung gehören die Installation, Konfiguration und das Updaten der entsprechenden Software.

- *Backup der Daten durchführen (Betrieb – Datensicherheit)*: Daten werden meist serverseitig gespeichert. Es sollte täglich ein Backup aller Daten erfolgen, um einem möglichen Datenverlust vorzubeugen. Dazu müssen sie auf einen unabhängigen Server gespielt und bei Bedarf auch wieder zurückgespielt werden.

- *Bedarfsanalyse konzipieren und durchführen (Marketing – Bedarfsanalyse)*: Eine Bedarfsanalyse muss entsprechend vorbereitet und dann durchgeführt werden.

- *Benutzerdokumentation pflegen (Anwendungsentwicklung - Dokumentation)*: Eine Benutzerdokumentation gehört entwickelt und gepflegt. Sie führt den Benutzer mit einem Benutzerhandbuch durch das System und beschreibt den Zweck der Software.

- *Benutzerwünsche und Fehlerberichte weiterleiten (Benutzerbetreuung – Kommunikation)*: Benutzerwünsche und Fehlerberichte müssen über betreffende Kanäle in den Entwicklungsprozess integriert werden. Die Benutzerbetreuung ist hier vermittelnde Station zwischen Anwendungsentwicklung und Nutzung.

- *Beteiligung an Koordinationstreffen (Kooperation – Koordination)*: Alle Dienstleister müssen sich an regelmäßigen Koordinationstreffen beteiligen, um die Bereitstellung einer Software abzustimmen.

- *Entwicklerdokumentation pflegen (Anwendungsentwicklung – Dokumentation)*: Entwicklerdokumentation muss entwickelt und gepflegt werden. Dazu gehören z. B.: Klassendiagramme, Konzeptionspapiere und Styleguides zum Programmieren, kommentierter Code.

- *Entwicklung koordinieren und überwachen (Anwendungsentwicklung – Leitung)*: Zur Koordination der Softwareentwicklung zählt die Abstimmung aller Beteiligten bzw. deren Arbeit in Hinblick auf das nächste Release. Darüber hinaus muss die Entwicklung hinsichtlich der Einhaltung von Standards, Softwarearchitektur und Planung überwacht werden.

- *Entwicklungsclients pflegen (Anwendungsentwicklung – Entwicklungsumgebung)*: Zur Entwicklung einer Software werden Entwicklungsclients gebraucht, die neben dem Betriebssystem meist ein CVS-Client und eine entsprechende Entwicklungsumgebung benötigen. Diese Software muss installiert, konfiguriert und gewartet werden.

- *Entwicklungsentscheidungen treffen (Anwendungsentwicklung – Leitung)*: Insbesondere in strittigen Situationen müssen richtungweisende Entscheidungen gefällt und verantwortet werden.

- *Entwicklungsserver pflegen (Anwendungsentwicklung – Entwicklungsumgebung)*: Zum Betrieb eines Entwicklungsservers gehört das Installieren, Konfigurieren und Warten der zu entwickelnden Software. Dieser Server ist nur für an der Entwicklung beteiligte Personen zugänglich.

- *Ergebnisse in die Bereitstellung integrieren (Marketing – Bedarfsanalyse und Benutzerbetreuung – Evaluation der Nutzung)*: Ergebnisse einer Analyse (Evaluation, Bedarfsanalyse usw.) geben Hinweise für andere Aufgabenbereiche. Diese Ergebnisse müssen kommuniziert werden, so dass sie in die betroffenen Aufgabengebiete integriert werden können.

- *Evaluation konzipieren und durchführen (Benutzerbetreuung – Evaluation der Nutzung)*: Eine Evaluation der Nutzung muss vorbereitet werden, indem z. B. Fragebögen oder Interviewleitfäden erstellt werden. Aus der durchgeführten Evaluation lassen sich Ergebnisse über die Nutzung der Software und die angebotenen Dienstleistungen ableiten.

- *FAQ pflegen (Benutzerbetreuung – Handhabungssupport)*: Der Aufbau und die Pflege von „Frequently Ask Questions“ ist ein Bau-

stein zur Begegnung von Fragen verschiedener Beteiligter. Die Fragen müssen gesammelt, beantwortet und dann allgemein zugänglich, z. B. im WWW, verfügbar gemacht werden.

- *Features programmieren (Anwendungsentwicklung – Programmierung)*: Nach den Vorgaben der Planung werden im Rahmen der Softwarearchitektur neue Features programmiert.

- *Feedback geben (Kooperation – Kommunikation)*: Feedback gehört zur reaktiven Kommunikation und sollte zwischen allen Dienstleistern uneingeschränkt ausgetauscht werden können.

- *Fehler beheben (Anwendungsentwicklung – Programmierung)*: Auftretende Fehler müssen behoben werden.

- *Fehlerbehebung veröffentlichen (Anwendungsentwicklung – Releasemanagement)*: Im Anschluss an die Fehlerbehebung sollte der Person, die den Fehler gemeldet hat, darüber informiert werden. Zudem müssen Fehlerdokumentationen mit neuen Releases veröffentlich werden.

- *Fehlerberichte in Entwicklungsprozess integrieren (Anwendungsentwicklung – Fehlermanagement)*: Gemeldete Fehler müssen in den Entwicklungsprozess über entsprechende Kanäle (z. B. Bug-Tracker) integriert werden, damit die Fehler geordnet behoben werden können.

- *Hotline anbieten (Benutzerbetreuung – Handhabungssupport & Kundenbetreuung – Kommunikation)*: Die Bereitstellung einer Hotline dient der Klärung von Fragen, der Annahme von Fehlern oder der Meldung von Störungen. Die Hotline kann per E-Mail oder per Telefon erreichbar sein und sollte 24 Stunden/sieben Tage in der Woche zur Verfügung stehen.

- *Implementierungen planen und Zeitplan erstellen (Anwendungsentwicklung – Planung)*: Aus Anforderungsanalysen und richtungweisenden Entwicklungsentscheidungen können Implementierungsaufgaben extrahiert und in einem Zeitplan angeordnet werden.

- *Informationsmaterialien entwickeln und verbreiten (Marketing – Werbung)*: Ziel des Marketings ist, potentielle Benutzer auf die Software durch entsprechende Informationsmaterialien aufmerksam zu machen. Diese Informationsmaterialen müssen über geeignete Kanäle verteilt werden.

- *Informationsmaterialien entwickeln und verschicken (Kundenbetreuung – Kommunikation)*: Bei der Kundenbetreuung werden u. a. über entsprechende Informationsmaterialien Kundenkontakte gepflegt.

- *Informationsmaterialien erstellen und verteilen (Benutzerbetreuung – Kommunikation)*: Bei der Benutzerbetreuung setzen Informationsmaterialien die Benutzer der Software über die Betreuungsangebote in Kenntnis. Diese Informationsmaterialen müssen erstellt und über geeignete Kanäle verteilt werden.

- *Informationsveranstaltungen und Workshops organisieren (Benutzerbetreuung – Kommunikation)*: Workshops und Informationsveranstaltungen dienen der Benutzerbetreuung als Instrument der Werbung für ihre Angeboten und Leistungen, aber auch als Betreuungsleistung, da auf Informationsveranstaltungen und Workshops Nutzer miteinander in Kontakt gebracht werden.

- *Interne Workshops durchführen (Kooperation – Training)*: Interne Workshops dienen der Vorbereitung aller Dienstleister, um die übergreifenden Aufgabe *Software x bereitstellen* wahrnehmen zu können. Jeder Beteiligte übernimmt abwechselnd die Leitung, um die anderen über die eigenen Möglichkeiten zu unterrichten. Diese Workshops dienen der Abstimmung unter den Dienstleistern. Unter anderem können auf diesen Workshops neue Versionen der Software vorgestellt werden.

- *Internen Newsletter herausbringen (Koordination – Kommunikation)*: Die Herausgabe eines internen Newsletters, d. h. eines Newsletters gezielt für alle Dienstleister, informiert über die neuesten Aktivitäten im Dienstleistungsnetzwerk.

- *Klima- und Brandschutz installieren (Betrieb – Ausfallsicherheit)*: Zur Wahrung der Ausfallsicherheit gehört die Kühlung des Serverumfeldes, um entstehende Abwärme schnell abführen zu können.

Hierzu ist ggf. die Installation und Wartung von entsprechender Klimatechnik von Nöten. Ein Brandschutz (feuerfestes Material, Sprinkleranlagen und eine möglichst kurze, garantierte Reaktionszeit der Feuerwehr) muss ebenfalls gegeben sein.

- *Kooperationsverträge entwickeln (Kooperation – Vertragswerk)*: Die Grundlage der Kooperation zwischen den Dienstleistern stellen vertraglich festgelegte Rechte und Pflichten dar, die verhandelt und in Vertragsformen festgehalten werden. Äußere Einflüsse verändern die Bereitstellungssituation im Laufe der Zeit, so dass Verträge ggf. angepasst werden müssen.

- *Kundenwünsche weiterleiten (Kundenbetreuung – Kommunikation)*: Entsprechende Kundenwünsche zur Software oder den zusätzlichen Dienstleistungen müssen an die verantwortlichen Stellen weitergeleitet werden.

- *Netzsicherheit installieren und pflegen (Zugriffsermöglichung – Netzsicherheit)*: Der Zugriff auf den Anwendungsserver sollte mit entsprechenden Sicherheitsmechanismen (z. B. SSL oder VPN) gesichert werden.

- *Rechnungen stellen (Kooperation – Abrechnung)*: Alle Dienstleister müssen ihre Leistungen anderen beteiligten Dienstleistern in Rechnung stellen.

- *Rechnungen stellen (Kundenbetreuung – Abrechnung)*: Für laufende Verträge müssen Rechnungen gestellt und an Kunden verschickt werden.

- *Releases identifizieren, zusammenstellen und veröffentlichen (Anwendungsentwicklung – Releasemanagement)*: Zur Veröffentlichung einer neuen Version der Software muss diese zunächst identifiziert und dann in einem Release zusammengestellt werden. Anschließend erfolgt die Veröffentlichung dieses Release über entsprechende Kanäle (z. B. Webseite).

- *Serverhardware pflegen (Betrieb – Hardware)*: Zum Betrieb eines Servers zur Bereitstellung einer Software wird entsprechende Hard-

ware benötigt. Diese muss zur Verfügung gestellt und gewartet werden.

- *Software und Dienstleistungen präsentieren (Marketing – Werbung)*: Zur Etablierung der Software gehört ihre Präsentation und der dazugehörigen Dienstleistungen z. B. bei öffentlichen Messen und Konferenzen sowie in Unternehmen.

- *Softwarearchitektur pflegen (Anwendungsentwicklung – Programmierung)*: Die Softwarearchitektur muss in Hinblick auf Wartbarkeit, Erweiterbarkeit und Fehlerbehebbarkeit geplant und kontinuierlich den sich verändernden Unständen angepasst werden.

- *Standardverträge entwickeln (Kundenbetreuung – Verträge)*: Für Kunden müssen aufgrund der unterschiedlichen Finanzierungsmodelle entsprechende Verträge entwickelt werden, die Rechte und Pflichten, Leistungen und Vergütung festhalten.

- *Stromversorgung sichern (Betrieb – Ausfallsicherheit)*: Die Stromversorgung für den Anwendungsserver muss sichergestellt und gegen Stromschwankungen und -ausfälle gesichert sein. Dies ist u. a. mit der Installation und Wartung von USVs (unterbrechungsfreie Stromversorgungsgeräte) zu erreichen.

- *Überblick über die Bereitstellung behalten (Kooperation – Koordination)*: Alle Dienstleister haben die Pflicht, die Bereitstellung der Software in ihrer Gesamtheit zu verstehen und den Überblick über die Bereitstellung zu behalten.

- *Verbindung etablieren und pflegen (Zugriffsermöglichung – Netzverbindung)*: Die Verbindung vom Server zu den Clients muss etabliert und gepflegt werden. Hierzu gehört in der Regel die Anbindung des Anwendungsservers an das Internet.

- *Vertragsbrüche verhandeln (Kooperation – Vertragswerk)*: Werden Vertragsbrüche erkannt, müssen alle Dienstleister über diese verhandeln.

- *Vertragsverhandlungen führen (Kooperation – Vertragswerk)*: Eine *Software x bereitstellen* wird als übergreifende Aufgabe meist in ei-

ner Kooperation von mehreren Beteiligten wahrgenommen. Als Kooperationsgrundlage dienen vertragliche Regelungen, die alle Dienstleister verhandeln müssen.

- *Vertragsverhandlung führen (Kundenbetreuung – Verträge)*: Mit Kunden müssen Verträge über die Nutzung bzw. Finanzierung geschlossen werden.

- *Virenscan der Daten durchführen (Betrieb – Datensicherheit)*: Serverseitig gespeicherten Daten müssen regelmäßig (täglich) nach Viren durchsucht werden. Bei positivem Befund sollten die Viren entfernt und betroffene Benutzer benachrichtig werden.

- *Weitere Sicherheitsmechanismen einrichten (Betrieb – Datensicherheit)*: Weitere Sicherheitsmechanismen schützen die Daten. Beispielsweise kann ein Intrusion Detection System (IDS) Eindringlinge im internen Bereich der technischen Infrastruktur aufspüren. Die weiteren Sicherheitsmechanismen müssen installiert, konfiguriert und gewartet werden.

- *Werbestrategie entwickeln (Marketing – Werbung)*: Zur Vermarktung der Software gehört eine geeignete Werbestrategie, um sie im Markt zu positionieren.

- *Zahlungen überwachen (Kooperation – Abrechnung)*: Die Verteilung der Einnahmen auf alle Dienstleister wird von allen Beteiligten überwacht.

- *Zahlungen überwachen (Kundenbetreuung – Abrechnung)*: Zahlungseingänge müssen überwacht werden und ggf. werden Mahnungen verschickt.

- *Zusätzlich benötigte Software pflegen (Anwendungsentwicklung – Entwicklungsumgebung)*: Zur Unterstützung der Entwicklung ist es u.U. notwendig weitere Software zu installieren und zu pflegen. Hierzu gehören u. a. ein Bug-Tracker für die Fehlerverwaltung und ein CVS-Server für die Versionsverwaltung.

- *Zusätzlich benötigte Software pflegen (Betrieb – Basissoftware)*: Als Softwaregrundlage eines (Anwendungs-)Servers wird ein Betriebs-

system benötigt. Es muss, wie zusätzlich benötigte Softwarekomponenten, installiert, konfiguriert und gewartet werden.

Die hier dargestellten Aufgaben werden nicht als fest angesehen, sondern müssen im konkreten Bereitstellungsszenario angepasst und ggf. ergänzt werden.

A.3 Funktionelle Rollen in der Softwarebereitstellung

Die funktionellen Rollen in alphabetischer Reihenfolge im Detail (in Klammern sind die zugehörigen übergeordneten Aufgaben aufgeführt):

- *Analyst (Marketing)*: Der Analyst analysiert den Bedarf nach einer Bereitstellung der Software. Aufgabengebiet: Bedarfsanalyse.
- *Ansprechpartner (Kooperation)*: Der Ansprechpartner eines Dienstleister steht den Ansprechpartnern der anderen Dienstleister zum Austausch von Informationen zur Verfügung. Aufgabengebiet: Kommunikation.
- *Anwendungsadministrator (Betrieb)*: Der Anwendungsadministrator ist für die Verfügbarkeit und Lauffähigkeit der bereitgestellten Anwendung zuständig. Aufgabengebiet: Anwendung.
- *Benutzerbetreuer (Benutzerbetreuung)*: Der Benutzerbetreuer betreut und hilft den Nutzenden bei Handhabungsproblemen. Außerdem ist diese funktionale Rolle für das Betreuungsmarketing verantwortlich. Aufgabengebiete: Handhabungssupport; Kommunikation.
- *Buchhalter (Kundenbetreuung)*: Der Buchhalter führt die Abrechnung bei bestehenden Verträgen durch. Aufgabengebiete: Abrechnung.
- *Entwicklungsadministrator (Anwendungsentwicklung)*: Der Entwicklungsadministrator ist für den Betrieb der Entwicklungsumgebung verantwortlich. Aufgabengebiet: Entwicklungsumgebung.
- *Entwicklungsleiter (Anwendungsentwicklung)*: Der Entwicklungsleiter leitet die Entwicklung der Software. Er koordiniert und trifft weitreichende Entscheidungen bei der Entwicklung. Aufgabengebiet: Leitung.

- *Entwicklungsplaner (Anwendungsentwicklung)*: Der Entwicklungsplaner plant die langfristige und kurzfristige Weiterentwicklung der Software, definiert neue Releases und stellt sie zusammen. Aufgabengebiete: Planung; Releasemanagement.

- *Evaluator (Benutzerbetreuung)*: Der Evaluator evaluiert die Nutzung der Software. Er bereitet die Evaluation vor, führt sie durch und wertet sie aus. Aufgabengebiet: Evaluation der Nutzung.

- *Fehlerbeauftragte (Anwendungsentwicklung)*: Der Fehlerbeauftragte ist für die Annahme, Verteilung und Rückmeldung der Fehlermeldungen und der Fehlerbehebungen im Fehlerprozess verantwortlich. Aufgabengebiet: Fehlermanagement.

- *Hardwareadministrator (Betrieb)*: Der Hardwareadministrator ist für die Hardware des Anwendungsservers verantwortlich. Aufgabengebiete: Hardware; Ausfallsicherheit.

- *Kundenbetreuer (Kundenbetreuung)*: Der Kundenbetreuer kümmert sich um die Kundenbetreuung und die Vertragsentwürfe. Aufgabengebiete: Verträge; Kommunikation.

- *Netzwerkadministrator (Zugriffsermöglichung)*: Der Netzwerkadministrator ist für die Verbindung vom Anwendungsserver zum Internet bzw. zum Client verantwortlich. Aufgabengebiete: Netzsicherheit; Netzverbindung.

- *Programmierer (Anwendungsentwicklung)*: Der Programmierer programmiert einerseits die Software, behebt Fehler und kümmert sich um die Softwarearchitektur und ist andererseits für die verschiedenen Dokumentationen (Benutzer- und Entwicklerdokumentation) verantwortlich. Aufgabengebiete: Dokumentation; Programmierung.

- *Softwareadministrator (Betrieb)*: Der Softwareadministrator ist für die Verfügbarkeit der zusätzlich benötigten Software (Betriebssystem, Webserver, PHP-Interpreter, MySQL-Datenbank usw.) und für die Sicherheit der Daten verantwortlich. Aufgabengebiete: Basissoftware; Datensicherheit.

- *Trainer (Kooperation)*: Der Trainer eines Dienstleisters leitet und moderiert Workshops. Er unterrichtet die anderen Dienstleister über die Möglichkeiten und Leistungen seines Dienstleisters, so dass sich die anderen in ihrer Arbeit auf diese Dinge einstellen können. Aufgabengebiet: Training.

- *Verwalter (Kooperation)*: Der Verwalter verwaltet die Abrechnung und ist für die vertraglichen Regelungen bzgl. der Kooperation aller Beteiligten jeweils aus Sicht eines Beteiligten zuständig. Aufgabengebiete: Abrechnung und Vertragswerk.

- *Werbefachmann (Marketing)*: Der Werbefachmann positioniert die Software im Markt mit einer geeigneten Werbestrategie und plant entsprechende Werbeaktionen. Aufgabengebiet: Werbung.

Literaturverzeichnis

(Argyris u. a. 1985) Argyris, C.; Putnam, R.; Smith, D. (Hrsg.): *Action Science: Concepts, Methods and Skills for Research and Intervention*. San Francisco: Jossey-Bass, 1985.

(ASPIC 2002) ASP Industry Consortium: *Homepage*. `http://www.aspconsortium.org` (03.05.2002). 2002. – Internationale ASP-Organisation und Interessenvertretung.

(ASPKD 2002) ASP-Konsortium Deutschland: *Homepage*. `http://www.asp-konsortium.de` (03.05.2002). 2002. – Deutsche ASP-Organisation und Interessenvertretung.

(Avison u. a. 1999) Avison, D.; Lau, F.; Myers, M.; Nielsen, P. A.: Action Research. In: *Communications of the ACM* 42 (1999), Nr. 1, S. 94–97.

(Balasubramanian u. a. 2002) Balasubramanian, P. R.; Wyner, G. M.; Joglekar, N.: The Role of Coordination and Architecture in Supporting ASP Business Models. In: Sprague, R. H. (Hrsg.): *Proceedings of the 35th Hawaii International Conference on System Science 2002*. Los Alamitos, CA, USA: IEEE, 2002.

(Berekoven 1986) Berekoven, L.: Der Dienstleistungsmarkt – Sachliche Besonderheiten und empirische Befunde. S. 24–37. In: Pestel, E. (Hrsg.): *Perspektiven der Dienstleistungsgesellschaft*. Göttingen: Vandenhoeck und Ruprecht, 1986.

(BGB 2002) Bundesrepublik Deutschland: *Bürgerliches Gesetzbuch*. `http://dejure.org/gesetze/BGB` (17.10.2003). 2002. – Rechtsvorschriften Bundesrepublik Deutschland.

(Bleek 2004) Bleek, W.-G.: *Software-Infrastruktur – Von analytischer Perspektive zu konstruktiver Orientierung*. Hamburg: Hamburg University Press, 2004.

(Bleek und Jackewitz 2004) Bleek, W.-G.; Jackewitz, I.: Providing an E-Learning Platform in a University Context – Balancing the Organisational Frame for Application Service Providing. In: Sprague, R. H. (Hrsg.): *Proceedings of the 37th Hawaii International Conference on System Science 2004*. Los Alamitos, CA, USA: IEEE, 2004.

(Bleek u. a. 2003) Bleek, W.-G.; Jackewitz, I.; Pape, B.: The Role of Coordination and Architecture in Supporting ASP Business Models. In: Sprague, R. H. (Hrsg.): *Proceedings of the 36th Hawaii International Conference on System Science 2003*. Los Alamitos, CA, USA: IEEE, 2003.

(Bleek und Pape 2001) Bleek, W.-G.; Pape, B.: Application Service Providing für vernetzte Projektarbeit – am Beispiel von CommSy@uni.de. S. 349–371. In: Engelien, M.; Neumann, D. (Hrsg.): *Virtuelle Organisation und Neue Medien – Workshop GeNeMe2001 – Gemeinschaften in Neuen Medien*. Lohmar, Köln: Josef Eul, 2001.

(Bolle 1965) Bolle, F.: *Knaurs Lexikon a-z*. München: Droemersche Verlagsgesellschaft Th. Knaur, 1965.

(Brockhaus 1989) Brockhaus: *Brockhaus-Enzyklopaedie in 24 Bänden*. Bd. 10: HERR – IS. 19. Auflage. Mannheim: F.A. Brockhaus GmbH, 1989.

(Burris 2002) Burris, A. M.: *Service Provider Strategy – Proven Secrets of xSPs*. Upper Saddle River: Prentice Hall PTR, 2002.

(Castells 2001) Castells, M.: Bausteine einer Theorie der Netzwerkgesellschaft. In: *Berliner Journal für Soziologie* 11 (2001), Nr. 4, S. 423–439.

(Castells 2003) Castells, M.: *Aufstieg der Netzwerkgesellschaft*. Bd. 1. Stuttgart: Uni Taschenbücher UTB, 2003.

(CherryTree 1999) CherryTree: *Application Service Providers (ASP)*. `http://www.cherrytreeco.com/reports/asp.pdf` (23.04.2003). 1999. – Spotlight Report.

(Citrix 2000) Citrix Systems Inc.: *Application Serving*. `http://www.amsys.net/pdf/citrixwhitepapers.pdf` (24.04.2003). 2000. – White Paper.

(Cohn und Farau 1993) Cohn, R. C.; Farau, A.: *Gelebte Geschichte der Psychotherapie: Zwei Perspektiven*. 4. Auflage. Stuttgart: Klett-Cotta, 1993.

(CommSy 2004) HITeC: *Homepage*. `http://www.commsy.de` (28.07.2004). 2004. – Hamburger Informatik Technologie-Center e.V..

(Dechant u. a. 2004) Dechant, H.; Stelzer, D.; Trost, R.: Heuristische Erlösprognosen für die Bewertung von Geschäftsmodellen im Application Service Providing. In: *Wirtschaftsinformatik* 46 (2004), Nr. 6, S. 446–458.

(DeMichelis 2003) DeMichelis, G.: The "Swiss Pattada". In: *Interactions* 10 (2003), Nr. 3, S. 44–53.

(Döring 2003) Döring, N.: *Sozialpsychologie des Internet – Die Bedeutung des Internet für Kommunikationsprozesse, Identitäten, soziale Beziehungen und Gruppen*. Bd. 2. Göttingen u.a.: Hogrefe, 2003.

(Duden 1980) Wissenschaftlicher Rat der Dudenredaktion (Hrsg.): *Die Rechtschreibung*. Bd. 1. 18. Auflage. Mannheim u.a.: Bibliographisches Institut, 1980.

(Duden 1993) Lektorat des B.I.-Wissenschaftsverlag (Hrsg.): *Informatik – Ein Sachlexikon für Studium und Praxis*. 2. Auflage. Mannheim u.a.: Dudenverlag, 1993.

(Eilingsfeld und Schätzler 1997) Eilingsfeld, F.; Schätzler, D.: *Intranets – firmeninterne Informationssysteme mit Internet-Technologie*. Heidelberg: dpunkt, 1997.

(Eisenmann und Pothen 2001) Eisenmann, T.; Pothen, S.: *Application Service Providers*. Cambridge: Havard Business School Publishing, 2001 (801310).

(Endres 2004) Endres, A.: Sind Outsourcing und Offshoring die neuen Heilmittel bei Informatik-Problemen? In: *Informatik-Spektrum* 27 (2004), Dezember, Nr. 6, S. 546–550.

(Ertel 1986) Ertel, R.: Was sind Dienstleistungen? Definitorische Anmerkungen. S. 15–23. In: Pestel, E. (Hrsg.): *Perspektiven der Dienstleistungsgesellschaft*. Göttingen: Vandenhoeck und Ruprecht, 1986.

(Falkowski und Voß 2003) Falkowski, T.; Voß, S.: Application Service Providing as Part of Intelligent Decision Support for Supply Chain Management. In: Sprague, R. H. (Hrsg.): *Proceedings of the 36th Hawaii International Conference on System Science 2003*. Los Alamitos, CA, USA: IEEE, 2003.

(Farleit 2000) Farleit Limited: *Anatomy of an ASP – Defining the new genus*. `http://epf-oi.uni-mb.si/clani/privka/rac-podp-revizija/asp-osuove-trend%i.pdf` (23.04.2003). 2000. – White Paper.

(Finck u. a. 2004) Finck, M.; Janneck, M.; Oberquelle, H.: Benutzergerechte Gestaltung von CSCL-Systemen. S. 203–219. In: Pape, B.; Krause, D.; Oberquelle, H. (Hrsg.): *Wissensprojekte – Gemeinschaftliches Lernen aus didaktischer, softwaretechnischer und organisatorischer Sicht*. Münster: Waxmann, 2004.

(Fjellheim u. a. 1974) Fjellheim, R.; Handlykken, P.; Nygaard, K.: *Report from a Seminar on System Description at 'Skogen', Roros*. Oslo: Norwegian Computer Center, 1974 (S-65).

(Flick 1999) Flick, U.: *Qualitative Forschung*. 4. Auflage. Reinbek: Rowohlt, 1999.

(Floyd 1986) Floyd, C.: STEPS – eine Orientierung der Softwaretechnik auf sozialverträgliche Technikgestaltung. S. 106–131. In: Riedemann, E.; Hagen, U. von; Heß, K.-D.; Wicke, W. (Hrsg.): *10 Jahre Informatik und Gesellschaft – eine Herausforderung bleibt bestehen*. Dortmund: Universität Dortmund, 1986. – Forschungsbericht Nr. 227.

(Floyd 1989) Floyd, C.: Softwareentwicklung als Realitätskonstruktion. S. 1–20. In: Lippe, W.-M. (Hrsg.): *Software-Entwicklung – Konzepte, Erfahrungen, Perspektiven*. Berlin u.a.: Springer, 1989.

(Floyd 1991a) Floyd, C.: Aufgabenbezogene Anforderungsermittlung. Kap. 3. In: Floyd, C. (Hrsg.): *Arbeitsunterlagen zur Lehrveranstal-*

tung Einführung in die Softwaretechnik. Hamburg: Universität Hamburg, Fachbereich Informatik, 1991.

(Floyd 1991b) Floyd, C.: *Evolutionäre Systementwicklung – STEPS*. Berlin: Technische Universität Berlin, 1991. – Unterlagen zum Seminar beim VDI-Verlag, 26.–27.02.1991.

(Floyd 1994) Floyd, C.: Software-Engineering – und dann? In: *Informatik-Spektrum* 17 (1994), Nr. 1, S. 29–37.

(Floyd u. a. 1998a) Floyd, C.; Gryczan, G.; Mack, J.: Programmklassifikation und Anforderungsermittlung. Kap. 6. In: Floyd, C. (Hrsg.): *Prüfungsunterlagen zur Lehrveranstaltung Einführung in die Softwaretechnik*. Hamburg: Universität Hamburg, Fachbereich Informatik, 1998.

(Floyd u. a. 1998b) Floyd, C.; Gryczan, G.; Mack, J.: Projektorganisation. Kap. 13. In: Floyd, C. (Hrsg.): *Prüfungsunterlagen zur Lehrveranstaltung Einführung in die Softwaretechnik*. Hamburg: Universität Hamburg, Fachbereich Informatik, 1998.

(Floyd u. a. 1997) Floyd, C.; Krabbel, A.; Ratuski, S; Wetzel, I.: Zur Evolution der evolutionären Systementwicklung: Erfahrungen aus einem Krankenhausprojekt. In: *Informatik-Spektrum* 20 (1997), Nr. 1, S. 13–20.

(Floyd u. a. 1989a) Floyd, C.; Mehl, W.-M.; Reisin, F.-M.; Schmitt, G.; Wolf, G.: Out of Scandinavia: Alternative Approaches to Software Design and System Development. In: *Human-Computer Interaction* 4 (1989), S. 253–350.

(Floyd u. a. 1990) Floyd, C.; Mehl, W.-M.; Reisin, F.-M.; Wolf, G.: *Projekt PetS: Partizipative Entwicklung transparenzschaffender Software für EDV-gestützte Arbeitsplätze*. Berlin: Technische Universität Berlin, 1990. – Endbericht an das Ministerium für Arbeit, Gesundheit und Soziales des Landes Nordrhein-Westfalen.

(Floyd u. a. 2004) Floyd, C.; Pape, B.; Bleek, W.-G.; Jackewitz, I.; Jeenicke, M.: Softwareentwicklung als Wissensprojekt – am Beispiel der

CommSy-Entwicklung. S. 389–411. In: Pape, B.; Krause, D.; Oberquelle, H. (Hrsg.): *Wissensprojekte – Gemeinschaftliches Lernen aus didaktischer, softwaretechnischer und organisatorischer Sicht*. Münster: Waxmann, 2004.

(Floyd u. a. 1989b) Floyd, C.; Reisin, F.-M.; Schmitt, G.: STEPS to Software Development with Users. In: Chezzi, C.; McDermid, J. A. (Hrsg.): *Proccedings of the 2nd European Software Engineering Conference ESEC '89 – Lecture Notes in Computer Science 387*. Berlin: Springer, 1989.

(Floyd und Züllighoven 2002) Floyd, C.; Züllighoven, H.: Softwaretechnik. In: Rechenberger, P.; Pomberger, G. (Hrsg.): *Informatik Handbuch*. München: Hanser Fachbuch, 2002.

(Frank u. a. 1998) Frank, U.; Klein, S.; Krcmar, H.; Teubner, A.: Aktionsforschung in der WI – Einsatzpotentiale und Einsatzprobleme. S. 71–90. In: Schütte, R.; Siedentopf, J.; Zelewski, S. (Hrsg.): *Wirtschaftsinformatik und Wissenschaftstheorie – Grundpositionen und Theoriekerne*. Essen: Universität Essen, 1998. – Arbeitsberichte des Institut für Produktion und Industrielles Informationsmanagement, Nr. 4.

(Frey 2002) Frey, K.: *Die Projektmethode: Der Weg zum bildenden Tun*. 9. Auflage. Weinheim: Beltz, 2002.

(Fricke 1997) Fricke, W. (Hrsg.): *Aktionsforschung und industrielle Demokratie*. Düsseldorf: Satz + Druck GmbH, 1997 (6). – Friedrich-Ebert-Stiftung, Forum: Zukunft der Arbeit.

(Gerhard und Mayr 2002) Gerhard, J.; Mayr, P.: Competing in the E-Learning Environment – Strategies for Universities. In: Sprague, R. H. (Hrsg.): *Proceedings of the 35th Hawaii International Conference on System Science 2002*. Los Alamitos, CA, USA: IEEE, 2002.

(Giddens 1997) Giddens, A.: *Die Konstitution der Gesellschaft: Grundzüge einer Theorie der Strukturierung*. 3. Auflage. Frankfurt(Main) u.a.: Campus Fachbuch, 1997.

(Gillian u. a. 1999) Gillian, C.; Graham, S.; Levitt, M.; McArthur, J.; Murray, S.; Turner, V.; Villars, R.; McCarthy, W. M.: *The ASPs Impact on*

the IT Industry: An ICS-Wide Opinion. `http://www.amsys.net/pdf/idpwhitepaper.pdf` (24.04.2003). 1999. – Bulletin, International Data Corporation (IDC), White Paper.

(Günther u. a. 2001) Günther, O.; Tamm, G.; Hansen, L.; Meseg, T.: Application Service Providers: Angebot, Nachfrage und langfristige Perspektiven. In: *Wirtschaftsinformatik* 43 (2001), Nr. 6, S. 555–567.

(Gralla 1997) Gralla, P.: *So funktionieren Intranets – Ein visueller Streifzug durch das Intranet*. München: Markt & Technik Buch und Software Verlag, 1997.

(Grohmann 2002) Grohmann, W. (Hrsg.): *ASP – Application Service Providing: Software auf Mietbasis: Kosten sparen – Wettbewerbsvorteile nutzen*. Köln: Deutscher Wirtschaftsdienst, 2002.

(Großmann u. a. 2004) Großmann, A.; Pape, B.; Simon, E.; Strauss, M.: Gestaltung der Benutzungsdokumentation für die Softwareunterstützung von Wissensprojekten. S. 343–357. In: Pape, B.; Krause, D.; Oberquelle, H. (Hrsg.): *Wissensprojekte – Gemeinschaftliches Lernen aus didaktischer, softwaretechnischer und organisatorischer Sicht*. Münster: Waxmann, 2004.

(Grover u. a. 1997) Grover, V.; Teng, J.; Cheon, M.: Towards a theoretically-based contingency model of information systems outsourcing. S. 79–101. In: Willcocks, L.; Lacity, M. C. (Hrsg.): *Strategic Sourcing of Information Systems: Perspectives and Practices*. New York: John Wiley & Sons, 1997.

(Gryczan 1996) Gryczan, G.: *Prozessmuster zur Unterstützung kooperativer Tätigkeit*. Wiesbaden: DUV, 1996.

(Gryczan u. a. 1998) Gryczan, G.; Krabbel, A.; Bäumer, D.; Wetzel, I.; Züllighoven, H.: Die WAM-Dokumenttypen. S. 601–661. In: Züllighoven, H. (Hrsg.): *Das objektorientierte Konstruktionshandbuch nach dem Werkzeug & Material-Ansatz*. Heidelberg: dpunkt, 1998.

(Gudjons 1998) Gudjons, H.: Was ist Projektunterricht? S. 14–27. In: Bastian, J.; Gudjons, H. (Hrsg.): *Das Projektbuch*. Hamburg: Bergmann und Helbig, 1998.

(Hankel u. a. 2003) Hankel, O.; Jackewitz, I.; Pape, B.; Strauss, M.: Technical and Didactical Scenarios of Student-centered Teaching and Learning. S. 411–419. In: Kerres, M.; Voß, B. (Hrsg.): *Digitaler Campus – Vom Medienprojekt zum nachhaltigen Medieneinsatz in der Hochschule*. Münster, New York: Waxmann, 2003.

(Heere 2001) Heere, E. C.: *Application Service Provider – A Definition and Rationale*. `http://www.amsys.net/pdf/asp-definition&rational.pdf` (24.04.2003). Februar 2001. – Amsys Inc.

(Heinrich 1992) Heinrich, L. J.: Organisation des Benutzer-Service. S. 308–318. In: Frese, E. (Hrsg.): *Handwörterbuch der Organisation*. Stuttgart: Schaeffer-Poeschel, 1992.

(Heinrich 1999) Heinrich, L. J.: *Informationsmanagement*. 6. Auflage. München, Wien: Oldenbourg, 1999.

(Heinrich und Hänschel 1996) Heinrich, L. J.; Hänschel, I.: Messen des Erfolgs des Benutzer-Service. In: *HMD – Praxis der Wirtschaftsinformatik* (1996), Nr. 189, S. 75–97.

(Hesse u. a. 1994) Hesse, W.; Barkow, G.; von Braun, H.; Kittlaus, H.-B.; Scheschonk, G.: Terminologie der Softwaretechnik: Ein Begriffssystem für die Analyse und Modellierung von Anwendungssystemen. Teil 1: Begriffssystematik und Grundbegriffe. In: *Informatik-Spektrum* 17 (1994), Februar, Nr. 1, S. 39–47.

(ISO 1996) DIN EN ISO 9241: *Ergonomische Anforderungen für Bürotätigkeiten mit Bildschirmgeräten, Teil 10: Grundsätze der Dialoggestaltung*. 1996.

(ITAA 2000) Information Technology Association of America: *ITAA's Application Service Provider (ASP) Service Level Agreement (SLA) Guidelines*. `http://www.itaa.org/asp/itaasla.pdf` (28.05.2003). 2000.

(Jackewitz 1998) Jackewitz, I.: *Benutzungsbetreuung*. Hamburg, Universität Hamburg, Fachbereich Informatik, Studienarbeit, 1998.

(Jackewitz 2000) Jackewitz, I.: *Computergestützte Benutzerforen zur Unterstützung von Softwareeinsatz in Organisationen*. Hamburg, Universität Hamburg, Fachbereich Informatik, Diplomarbeit, 2000.

(Jackewitz 2004) Jackewitz, I.: Bereitstellung einer kooperativen Lernplattform. S. 327–342. In: Pape, B.; Krause, D.; Oberquelle, H. (Hrsg.): *Wissensprojekte – Gemeinschaftliches Lernen aus didaktischer, softwaretechnischer und organisatorischer Sicht*. Münster: Waxmann, 2004.

(Jackewitz u. a. 2002a) Jackewitz, I.; Janneck, M.; Krause, D.; Pape, B.; Strauss, M.: Teaching Social Informatics as a Knowledge Project. S. 261–268. In: van Weert, T. J.; Munro, R. K. (Hrsg.): *Informatics and the Digital Society: Social, Ethical and Cognitive Issues*. Boston: Kluwer, 2002.

(Jackewitz u. a. 2002b) Jackewitz, I.; Janneck, M.; Krause, D.; Pape, B.; Strauss, M.: Wissensprojekt – eine Perspektive für die Softwareunterstützung im Informatikstudium. S. 443–451. In: Bachmann, G.; Haefeli, O.; Kindt, M. (Hrsg.): *Campus 2002*. Münster: Waxmann, 2002.

(Jackewitz u. a. 2002c) Jackewitz, I.; Janneck, M.; Pape, B.: Vernetzte Projektarbeit mit CommSy. S. 35–44. In: Herczeg, M.; Prinz, W.; Oberquelle, O. (Hrsg.): *Mensch und Computer 2002*. Stuttgart u.a.: Teubner, 2002.

(Jackewitz u. a. 2004) Jackewitz, I.; Janneck, M.; Strauss, M.: CommSy: Softwareunterstützung für Wissensprojekte. S. 186–202. In: Pape, B.; Krause, D.; Oberquelle, H. (Hrsg.): *Wissensprojekte – Gemeinschaftliches Lernen aus didaktischer, softwaretechnischer und organisatorischer Sicht*. Münster: Waxmann, 2004.

(Jackewitz und Pape 2004) Jackewitz, I.; Pape, B.: Einführung und Bereitstellung. S. 314–325. In: Haake, J.; Schwabe, G.; Wessner, M. (Hrsg.): *CSCL-Kompendium – Lehr- und Handbuch zum computerunterstützen kooperativen Lernen*. München, Wien: Oldenbourg, 2004.

(Janneck und Krause 2004) Janneck, M.; Krause, D.: Einladung zur Nachahmung: Offene Lernveranstaltungen mit Medienunterstützung. S. 74–89. In: Pape, B.; Krause, D.; Oberquelle, H. (Hrsg.): *Wissensprojekte*

– Gemeinschaftliches Lernen aus didaktischer, softwaretechnischer und organisatorischer Sicht. Münster: Waxmann, 2004.

(Janneck u. a. 2003) Janneck, M.; Krause, D.; Pape, B.; Strauss, M.: Softwareunterstützung für offene Seminare. S. 47–56. In: Bode, A.; Desel, J.; Rahmayer, S.; Wessner, M. (Hrsg.): *DeLFI 2003: Die erste e-Learning Fachtagung Informatik*. Bonn: Gesellschaft für Informatik, 2003.

(Jaruzelski u. a. 2000) Jaruzelski, B.; Ribeiro, F.; Lake, R.: *Understanding the Application Service Provider Model*. `http://www.boozallen.de/content/downloads/6_1000_ASP_Whitepaper.pdf` (03.06.2003). 2000. – Booz, Allen & Hamilton, White Paper.

(Jayatilaka u. a. 2002) Jayatilaka, B.; Schwarz, A.; Hirschheim, R.: Determinants of ASP Choice: an Integrated Perspective. In: Sprague, R. H. (Hrsg.): *Proceedings of the 35th Hawaii International Conference on System Science 2002*. Los Alamitos, CA, USA: IEEE, 2002.

(Kaschek u. a. 2001) Kaschek, R.; Klischewski, R.; Wetzel, I.: A Virtual Debate in Serviceflows. In: *EMISA Forum* (2001), Nr. 1, S. 16–23. – GI-FG 2.5.2.

(Kilberth u. a. 1994) Kilberth, K.; Gryczan, G.; Züllighoven, H.: *Objektorientierte Anwendungsentwicklung*. 2. Auflage. Braunschweig, Wiesbaden: Vieweg, 1994.

(Kling 1999) Kling, R.: What is Social Informatics and Why Does it Matter? In: *D-Lib Magazine* 5 (1999), Nr. 1.

(Klischewski 2000) Klischewski, R.: Abstrakte Bedürfnisse und konkrete Beziehungen oder: Wie man Services (nicht) modelliert. S. 19–26. In: Ebert, J.; Frank, U. (Hrsg.): *Modelle und Modellierungssprachen in Informatik und Wirtschaftsinformatik*. Koblenz: Fölbach, 2000. – Proceedings Modellierung 2000 (St. Goar, 5.–7.4.2000).

(Klischewski 2004) Klischewski, R.: *Systementwicklung als Vernetzung – Interoperabilität in der internetbasierten Verwaltung als Herausforderung*. Hamburg, Universität Hamburg, Fachbereich Informatik, Habilitation, 2004. – eingereicht: 2003, Kolloquium: 31.03.2004.

(Klischewski und Lenk 2002) Klischewski, R.; Lenk, K.: Understanding and Modelling Flexibility an Administrative Processes. S. 129–136. In: Traunmüller, R.; Lenk, K. (Hrsg.): *Electronic Government*. Berlin: Springer Lecture Notes, 2002. – Proceedings EGOV 2002.

(Klischewski und Wetzel 2000) Klischewski, R.; Wetzel, I.: Serviceflow Management. In: *Informatik-Spektrum* 23 (2000), Februar, Nr. 1, S. 38–46.

(Klischewski und Wetzel 2001a) Klischewski, R.; Wetzel, I.: Serviceflow Management für das organisationsübergreifende e-Government. S. 313–319. In: Bauknecht, K.; Brauer, W.; Mück, Th. (Hrsg.): *Informatik 2001. Wirtschaft und Wissenschaft in der Network Economy – Vision und Wirklichkeit*. Wien: Österreichische Computer Gesellschaft, 2001.

(Klischewski und Wetzel 2001b) Klischewski, R.; Wetzel, I.: System Development for Service Provider Networks. In: *Proceedings of SoftwareTrends.ch, Business Strategy Based Software Engineering*. Luzern, September 2001.

(Klischewski und Wetzel 2001c) Klischewski, R.; Wetzel, I.: XML-based Process Representation for e-Government Serviceflows. S. 789–802. In: Schmidt, B.; Stanoevska-Slabeva, K.; Tschammer, V. (Hrsg.): *Towards the E-Society: E-commerce, E-business, and E-government*. Dordrecht: Kluwer, 2001. – 13E 2001, IFIP.

(Klischewski und Wetzel 2002a) Klischewski, R.; Wetzel, I.: Serviceflow Management: Caring for Citizen's Concern in Designing E-Government Transaction Processes. In: Sprague, R. H. (Hrsg.): *Proceedings of the 35th Hawaii International Conference on System Science 2002*. Los Alamitos, CA, USA: IEEE, 2002.

(Klischewski und Wetzel 2002b) Klischewski, R.; Wetzel, I.: Vertragsbasiertes Prozessmanagement als Leitbild für die organisationsübergreifende Workflowunterstützung. S. 81–93. In: Desel, J.; Weske, M. (Hrsg.): *Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen*. Bonn: Gesellschaft für Informatik, Lecture Notes in Informatics, 2002. – Proceedings PROMISE 2002.

(Klischewski und Wetzel 2003) Klischewski, R.; Wetzel, I.: Serviceflow management for health provider networks. In: *Logistics Information Management* 16 (2003), Nr. 3/4, S. 259–269.

(Klischewski u. a. 2001) Klischewski, R.; Wetzel, I.; Baharami, A.: Modeling Serviceflow. S. 261–272. In: Godlevsky, M.; Mayr, H. (Hrsg.): *Information Systems Technology and its Applications*. Bonn: German Informatics Society, Lecture Notes in Informatics, 2001. – Proceedings ISTA 2001 (13.–15.6 Kharkiv, Ukraine).

(König 1996) König, M. M.: Besitzlos glücklich? Verträge zwischen Anwender und ASP. In: *c't – Magazin für Computer Technik* (1996), Nr. 19, S. 220–222.

(Knolmayer 1996) Knolmayer, G.: Benutzersupport: eine Kernkompetenz des IV-Bereichs? In: *HMD – Praxis der Wirtschaftsinformatik* (1996), Nr. 189, S. 7–24.

(Krabbel 2000) Krabbel, A.: *Entwurf, Auswahl und Anpassung aufgabenbezogener Domänensoftware*. Hamburg, Universität Hamburg, Fachbereich Informatik, Dissertation, 2000.

(Krabbel und Wetzel 1998) Krabbel, R.; Wetzel, I.: The Customization Process for Organizational Package Information Systems: A Challenge for Participatory Design. In: Chatfield, R. H.; Kuhn, S.; Muller, M. (Hrsg.): *Proceedings of the Participatory Design Conference: Broadening Participation 1998*. Palo Alto, 1998, S. 45–55.

(Krabbel u. a. 1996) Krabbel, R.; Wetzel, I.; Ratuski, S.: Objektorientierte Analysetechniken für übergreifende Aufgaben. S. 65–72. In: Ebert, J. (Hrsg.): *Softwaretechnik '96*. Koblenz, 1996. – Beiträge der GI-Fachtagung, 12.–13. September 1996.

(Krueger und Casey 2000) Krueger, R. A.; Casey, M. A.: *Focus groups: A practical guide for applied research*. Newbury Park, CA, USA: Sage Publications, 2000.

(Lacity und Hirschheim 1999) Lacity, M. C.; Hirschheim, R.: Information Technology Outsourcing – What Problems are we Trying to Solve?

S. 326–360. In: Currie, W. L.; Galliers, B. (Hrsg.): *Rethinking Management Information Systems*. New York: Oxford University Press, 1999.

(Lammers 2004) Lammers, M.: Make, Buy or Share – Combining Resource Based View, Transaction Cost Economics and Production Economies to a Sourcing Framework. In: *Wirtschaftsinformatik* 46 (2004), Nr. 3, S. 204–212.

(Langel-Nentwig 1991) Langel-Nentwig, H.: Der Benutzerservice als kritischer Erfolgsfaktor der Bürokommunikation. In: *Zeitschrift für Organisation (zfo)* (1991), Nr. 4, S. 277–279.

(Lehner u. a. 1995) Lehner, F.; Hildebrand, K.; Maier, R.: *Wirtschaftsinformatik – Theoretische Grundlagen*. München, Wien: Carl Hanser, 1995.

(Liess 2000) Liess, A.: ASP: Service Realität in Deutschland 2000. In: *IM – Informatik-Management & Consulting* (2000), Nr. 4, S. 21–27.

(Maisberger 1987) Maisberger, P.: Benutzer-Serice-Zentren: Der Weg zum mündigen Anwender – Ruf nach effizientem Computereinsatz wird immer lauter * Individuelle Datenverarbeitung (IDV) als Lösung. In: *Office Management* (1987), Nr. 4, S. 22–32.

(Mercer 2000) Mercer Management Consulting: *Application Service Providers. Where are the real profit zones?* `http://www.amsys.net/pdf/mercerasp.pdf` (24.04.2003). 2000.

(Mertens 1985) Mertens, P.: *Aufbauorganisation der Datenverarbeitung – Zentralisierung – Dezentralisierung – Informationszentrum*. Wiesbaden: Gabler, 1985.

(Neusel 2000) Neusel, A.: *Die eigene Hochschule – Internationale Frauenuniversität Technik und Kultur*. Bd. 1. Opladen: Leske + Budrich, 2000. – Schriftenreihe der Internationalen Frauenuniversität.

(Nygaard 1976) Nygaard, K.: ELTA-prosjektet og dets tilknytning til problemene i systemutvikling (The DELTA project and it's Relation to Problems in System Development) / Norwegian Computer Center. Oslo, 1976 (5). – Forschungsbericht. DELTA Projekt.

(Nygaard und Handlykken 1981) Nygaard, K.; Handlykken, P.: The System Development Process – It's Settings, some Problems and Needs for Methods. S. 157–172. In: Hünke, H. (Hrsg.): *Software Engineering Environments*. Amsterdam: North-Holland, 1981.

(Oberquelle 1987) Oberquelle, H.: *Sprachkonzepte für benutzergerechte Systeme*. Berlin u.a.: Springer, 1987. – Informatik-Fachberichte 144.

(Orlikowski 1992) Orlikowski, W. J.: The Duality of Technology: Rethinking the Concept of Technology in Organizations. In: *Organization Science* 3 (1992), Nr. 3, S. 398–427.

(Ortmann u. a. 1997) Ortmann, G.; Sydow, J.; Windeler, A.: Organisation als reflexive Strukturation. S. 315–354. In: Ortmann, G.; Sydow, J.; Türk, K. (Hrsg.): *Theorien der Organisation*. Opladen: Westdeutscher Verlag, 1997.

(Pape 2004) Pape, B.: *Organisation der Softwarenutzung*. Hamburg, Universität Hamburg, Fachbereich Informatik, Dissertation, 2004. – eingereicht: 2004, Disputation: 13.12.2004.

(Pape u. a. 2002a) Pape, B.; Bleek, W.-G.; Jackewitz, I.; Janneck, M.: Software Requirements for Project-Based Learning – CommSy as an Exemplary Approach. In: Sprague, R. H. (Hrsg.): *Proceedings of the 35th Hawaii International Conference on System Science 2002*. Los Alamitos, CA, USA: IEEE, 2002.

(Pape und Jackewitz 2002) Pape, B.; Jackewitz, I.: Nachfrage und Angebot zur Benutzungsbetreuung von Software im universitären Lehrbetrieb. S. 305–322. In: Engelien, M.; Neumann, D. (Hrsg.): *Virtuelle Organisation und Neue Medien – Workshop GeNeMe2002 – Gemeinschaften in Neuen Medien*. Lohmar, Köln: Josef Eul, 2002.

(Pape u. a. 2002b) Pape, B.; Jackewitz, I.; Bleek, W.-G.: Benutzungsbetreuung für Softwareunterstützung in Lehr-Lern-Situationen. S. 71–90. In: Bleek, W.-G.; Krause, D.; Oberquelle, H.; Pape, B. (Hrsg.): *Medienunterstütztes Lernen – Beiträge von der WissPro Wintertagung 2002*. Hamburg: Universität Hamburg, Fachbereichs Informatik, 2002. – FBI-HH-B-239/02.

(Pape u. a. 2004) Pape, B.; Krause, D.; Oberquelle, H. (Hrsg.): *Wissensprojekte – Gemeinschaftliches Lernen aus didaktischer, softwaretechnischer und organisatorischer Sicht*. Münster: Waxmann, 2004.

(Pape und Rolf 2004) Pape, B.; Rolf, A.: Integrierte Organisations- und Softwareentwicklung für kooperative Lernplattformen in der Hochschullehre. S. 287–310. In: Pape, B.; Krause, D.; Oberquelle, H. (Hrsg.): *Wissensprojekte – Gemeinschaftliches Lernen aus didaktischer, softwaretechnischer und organisatorischer Sicht*. Münster: Waxmann, 2004.

(Picot u. a. 2000) Picot, A.; Buttermann, A.; Walters, R.: Erfolgsfaktoren für Application Service Providing. In: *IM – Informatik-Management & Consulting* (2000), Nr. 15, S. 45–51. – Sonderausgabe: Application Service Providing.

(Picot und Jahn 2000) Picot, A.; Jahn, A.: ASP als innerbetriebliches Allheilmittel? In: *IM – Informatik-Management & Consulting* (2000), Nr. 15, S. 72–77. – Sonderausgabe: Application Service Providing.

(Picot u. a. 1998) Picot, A.; Reichwald, R.; Wigand, R. T. (Hrsg.): *Die grenzenlose Unternehmung – Information, Organisation und Management*. 3. Auflage. Wiesbaden: Gabler, 1998.

(Rammert 1993) Rammert, W. (Hrsg.): *Technik aus soziologischer Perspektive: Forschungsstand, Theorieansätze, Fallbeispiele – Ein Überblick*. Opladen: Westdeutscher Verlag, 1993.

(RFC1180 1991) Socolofsky, T.; Kale, C.: *A TCP/IP Tutorial*. `http://www.faqs.org/rfcs/rfc1180.html` (24.04.2003). 1991. – Spider Systems Limited, Request for Comments 1180.

(Riemer und Ahlemann 2001) Riemer, K.; Ahlemann, F.: Application Service Providing – Erfahrungsbericht aus Sicht eines Providers. S. 743–756. In: Buhl, H. U.; Huther, A.; Reitwiesner, B. (Hrsg.): *Information Age Economy*. Heidelberg: Physica, 2001. – 5. Internationale Tagung Wirtschaftsinformatik 2001.

(Rogers 1974) Rogers, C. (Hrsg.): *Lernen in Freiheit – Zur Bildungsreform in Schule und Universität*. München: Kösel, 1974.

(Rolf 1998) Rolf, A. (Hrsg.): *Grundlagen der Organisations- und Wirtschaftsinformatik*. Berlin: Springer, 1998.

(Rolf 2004a) Rolf, A.: *Informatiksysteme in Organisationen und globaler Gesellschaft. Teil B: Arbeit und Leben mit der Informationstechnik in der globalen Netzwerk-Ökonomie*. Hamburg: Universität Hamburg, Fachbereich Informatik, 2004 (FBI-HH-M-332/04).

(Rolf 2004b) Rolf, A.: Schlank ist nicht fit. In: *Die Tageszeitung (TAZ)* (2004), S. 11. – Ausgabe vom 26.03.2004.

(Rosenhagen 2002) Rosenhagen, K.: Das Service Level Agreement. S. 166–179. In: Grohmann, W. (Hrsg.): *ASP – Application Service Providing: Software auf Mietbasis: Kosten sparen – Wettbewerbsvorteile nutzen*. Köln: Deutscher Wirtschaftsdienst, 2002.

(Schiersmann u. a. 2002) Schiersmann, C.; Busse, J.; Krause, D.: *Medienkompetenz – Kompetenz für Neue Medien*. Berlin, 2002. – Studie im Auftrag des Forum Bildung, Workshop am 14. September 2001 in Berlin.

(Schiffer und Templ 2002) Schiffer, S.; Templ, J.: Internetdienste. In: Rechenberger, P.; Pomberger, G. (Hrsg.): *Informatik Handbuch*. München: Hanser Fachbuch, 2002.

(Schneider 2002) Schneider, J.: ASP und urheberrechtliche Aspekte. S. 138–143. In: Grohmann, W. (Hrsg.): *ASP – Application Service Providing: Software auf Mietbasis: Kosten sparen – Wettbewerbsvorteile nutzen*. Köln: Deutscher Wirtschaftsdienst, 2002.

(Schneider und Themann 2002) Schneider, J.; Themann, J.: ASP und datenschutzrechtliche Aspekte. S. 144–151. In: Grohmann, W. (Hrsg.): *ASP – Application Service Providing: Software auf Mietbasis: Kosten sparen – Wettbewerbsvorteile nutzen*. Köln: Deutscher Wirtschaftsdienst, 2002.

(Seltsikas und Currie 2002) Seltsikas, P.; Currie, W. L.: Evaluation The Application Service Provider (ASP) Business Model: The Challenge of Integration. In: Sprague, R. H. (Hrsg.): *Proceedings of the 35th Hawaii*

International Conference on System Science 2002. Los Alamitos, CA, USA: IEEE, 2002.

(Sennett 1999) Sennett, R.: *Der flexible Mensch. Die Kultur des Kapitalismus*. 9. Auflage. Berlin: Berliner Verlag, 1999.

(Siefkes 2003) Siefkes, D.: *Rahmen für eine Theorie der Informatik*. http://tal.cs.tu-berlin.de/lv/ss2003/basis_ig/Siefkes_RahmenTheorie_200%3-07-04.pdf (25.08.2004). 2003. – Unveröffentlichtes Manuskript vom 4.7.2003.

(Simon u. a. 2004) Simon, E.; Marinescu, I. L.; Finck, M.; Jackewitz, I.: CommSy Goes Open-Source. S. 311–326. In: Pape, B.; Krause, D.; Oberquelle, H. (Hrsg.): *Wissensprojekte – Gemeinschaftliches Lernen aus didaktischer, softwaretechnischer und organisatorischer Sicht*. Münster: Waxmann, 2004.

(Sinz 2002) Sinz, E. J.: Architektur von Informationssystemen. In: Rechenberger, P.; Pomberger, G. (Hrsg.): *Informatik Handbuch*. München: Hanser Fachbuch, 2002.

(Sound 1999) Sound Consultant: *Understanding the ASP Market. An ISV's Guide to Transitioning from Packaged Product to Online Service*. http://www.amsys.net/pdf/understandingtheaspmarket.pdf (24.04.2003). 1999.

(Stahlknecht 2000) Stahlknecht, P.: *Aktueller Stand und Entwicklungstendenzen im IT-Outsourcing*. http://www.gor-online.org/download/vor/Stahlknecht.pdf (23.04.2003). 2000. – Fachtagung der GOR-Arbeitsgruppe Wirtschaftsinformatik. Duisburg, 10. November 2000.

(Steinmetz und Mühlhäuser 2002) Steinmetz, R.; Mühlhäuser, M.: Rechnernetze. In: Rechenberger, P.; Pomberger, G. (Hrsg.): *Informatik Handbuch*. München: Hanser Fachbuch, 2002.

(Stähler 2001) Stähler, P.: *Geschäftsmodelle in der digitalen Ökonomie: Merkmale, Strategien und Auswirkungen*. Lohmar, Köln: Josef Eurl, 2001.

(Strauss und Corbin 1996) Strauss, A.; Corbin, J.: *Grounded Theory: Grundlagen Qualitativer Sozialforschung*. Weinheim: Beltz, 1996.

(Strauss u. a. 2003) Strauss, M.; Pape, B.; Adam, F.; Klein, M.; Reinecke, L.: *CommSy-Evaluationsbericht 2003: Softwareunterstützung für selbstständiges und kooperatives Lernen*. Hamburg: Universität Hamburg, Fachbereich Informatik, 2003 (FBI-HH-B-251/03).

(Ströbel 2000) Ströbel, M.: Dynamic Outsourcing of Services / Zurich Research Laboratory. Zürich, 2000. – Forschungsbericht. IBM.

(Sydow 1999) Sydow, J.: Management von Netzwerkorganisationen – Stand der Forschung. In: Sydow, J. (Hrsg.): *Management von Netzwerkorganisation*. Wiesbaden: Gabler, 1999.

(Tao 2000) Tao, L.: *Application Service Provider Model: Perspectives and Challenges*. `http://www.ssgrr.it/en/ssgrr2000/papers/034.pdf` (23.04.2003). 2000.

(Touraine 1984) Touraine, A.: *Le retour de l'acteur*. Paris: Fayard, 1984.

(Trieneken u. a. 2004) Trieneken, J. J. M.; Bouman, J. J.; van der Zwan, M.: Specification of Service Level Agreements: Problems, Principles and Practices. In: *Software Quality Journal* 12 (2004), März, Nr. 1, S. 43–57.

(TripleTree 2003) TripleTree: *2003 Outsourcing Update*. `http://www.cherrytreeco.com/reports/2003outupdate.pdf` (23.04.2003). 2003. – Spotlight Report.

(Weiß und Längsfeld 2000) Weiß, P.; Längsfeld, M.: Studie Virtuelle Unternehmen / FZI Forschungszentrum Informatik. Karlsruhe, 2000. – Studie.

(Weltz und Ortmann 1987) Weltz, F.; Ortmann, R. G.: Betreuung der Anwender beim Einsatz neuer Bürotechnik – ... ein Aufwand, der sich rechnet. In: *Office Management* (1987), Nr. 4, S. 6–15.

(Wetzel und Klischewski 2002) Wetzel, I.; Klischewski, R.: Serviceflow Beyond Workflow? Concepts and Architectures for Supporting Inter-Organizational Service Processes. In: Pidduck, A. B.; Mylopoulos, J.; Woo, C.C.; Ozsu, M. T. (Hrsg.): *Advanced Information Systems Engineering*. Berlin: Springer Lecture Notes in Computer Science, 2002. – Proceedings 14th CAiSE.

(Williamson 1991) Williamson, O. E.: Comparative Economic Organization: The Analysis of Discrete Structural Alternatives. In: *Administrative Science Quarterly* 36 (1991), Nr. 2, S. 269–296.

(WissPro 2003) WissPro: *Homepage*. `http://www.wisspro.de` (28.07.2004). 2003. – Forschungs- und Entwicklungsprojekt WissPro.

(Wulf und Rohde 1995) Wulf, V.; Rohde, M.: Towards an Integrated Organization and Technology Development. S. 55–64. In: Olson, G. M.; Shuon, S. (Hrsg.): *Proceedings of the Symposium on Designing Interactive Systems*. New York: ACM, 1995.

(Yao und Murphy 2002) Yao, Y.; Murphy, L.: Client Relationship Development for Application Service Providers: A Research Modell. In: Sprague, R. H. (Hrsg.): *Proceedings of the 35th Hawaii International Conference on System Science 2002*. Los Alamitos, CA, USA: IEEE, 2002.

(Zahn u. a. 1999) Zahn, E.; Barth, T.; Hertweck, A.: Outsourcing unternehmensnaher Dienstleistungen – Entwicklungsstand und strategische Entscheidungstatbestände. S. 3–38. In: Wisskirchen, F. (Hrsg.): *Outsourcing Projekte erfolgreich realisieren*. Stuttgart: Schäffer Poeschel, 1999.

(Züllighoven 1998) Züllighoven, H. (Hrsg.): *Das objektorientierte Konstruktionshandbuch nach dem Werkzeug & Material-Ansatz*. Heidelberg: dpunkt, 1998.

(Zwipf und Schönfelder 2002) Zwipf, J.; Schönfelder, J.: Rechtliche Einordnung das ASP-Modells, Vertragsarten ASP und Regelungsbedürftige Punkte. S. 121–137. In: Grohmann, W. (Hrsg.): *ASP – Application Service Providing: Software auf Mietbasis: Kosten sparen – Wettbewerbsvorteile nutzen*. Köln: Deutscher Wirtschaftsdienst, 2002.