Komponentenkataloge auf Basis eines einheitlichen Spezifikationsrahmens – ein Implementierungsbericht

Sven Overhage

Technische Universität Darmstadt, Fachgebiet Wirtschaftsinformatik I – Entwicklung von Anwendungssystemen, Institut für Betriebswirtschaftslehre, Hochschulstraße 1, D-64289 Darmstadt, Tel.: +49 (6151) 16-6688, Fax: -4301, E-Mail: overhage@bwl.tu-darmstadt.de, URL: http://www..bwl.tu-darmstadt.de/bwl8

Zusammenfassung. Im Rahmen dieses Beitrages werden mögliche Weiterentwicklungen des im Memorandum des Arbeitskreises 5.10.3 enthaltenen Spezifikationsrahmens diskutiert. Die vorgeschlagenen Erweiterungen basieren auf Anforderungen, die während der Entwicklungsphase an den zu implementierenden Komponentenkatalog gestellt wurden. Im Mittelpunkt stehen dabei Erweiterungen für eine integrierte Spezifikation verschiedener Komponentenarten sowie eine verbesserte Kompatibilität mit dem existierenden Spezifikationsstandard UDDI.

Schlüsselworte: Komponentenkatalog, Komponente, Framework, Spezifikation, UDDI

1 Einleitung

Das Vorhandensein standardisierter Werkzeuge zur Unterstützung der Spezifikation und des Austauschs von Komponenten ist für die komponentenorientierte Anwendungsentwicklung von zentraler Bedeutung [Szyp1998:317-318]. Ein wichtiger Bestandteil solcher Werkzeuge sind Komponentenkataloge, die die strukturierte Speicherung von Komponenten und deren Spezifikationen gemäß einem (jeweils implementierten) Spezifikationsrahmen ermöglichen.

Komponentenkataloge finden sich beispielsweise in Komponenten-Repositorien, also speziell auf die Komponentenorientierung abgestimmten Entwicklungsdatenbanken, mit denen Unternehmen die von ihnen entwickelten bzw. erworbenen Komponenten sowie deren Verwendung in Anwendungen (letzteres lässt sich auch als Konfigurationsmanagement bezeichnen) verwalten können. Von zentraler Bedeutung sind sie darüber hinaus auch für global zugängliche Komponentenmarktplätze (eine Auflistung solcher findet sich beispielsweise bei [Fett2002]) und Komponentenverzeichnisse, also den externen Komponentenmarkt. Hier bilden sie die Plattformen für den Austausch von Komponenten zwischen Produzenten und Konsumenten. Letzteren ermöglichen sie die Auswahl und Entnahme geeigneter Komponenten zur Konfigurierung von Anwendungen, Produzenten das Dokumentieren und Einstellen fertiger Komponenten für den Verkauf an potenzielle Nachfrager. Schließlich können Komponentenkataloge auch bei der Implementierung von Entwicklungswerkzeugen (beispielsweise im Rahmen von CASE Tools für die automatisierte Komponentenspezifikation) eingesetzt werden.

Komponentenkataloge sind spezialisierte Metainformationssysteme (und gehören somit bereits selbst zur Klasse der Repository-Systeme, vgl. [Ortn1999]), deren Metaschema das strukturierte Verwalten von Komponenten und zugehörigen Dokumentationsdaten ermöglicht. Ihr Metaschema basiert auf einem Spezifikationsrahmen, wodurch die Struktur der für eine Komponente jeweils zu speichernden Metadaten vorgegeben wird.

Im Rahmen des vorliegenden Beitrags werden einige Erfahrungen aus der Implementierung eines Komponentenkatalogs berichtet, dem unter anderem der im Memorandum des Arbeitskreises „Komponentenorientierte betriebliche Anwendungssysteme“ entwickelte Spezifikationsrahmen (vgl. [Turo2002]) zu Grunde gelegt wurde. Dabei werden nach einem einführenden Projektbericht zunächst wichtige Anforderungen an den zu Grunde gelegten Spezifikationsrahmen genannt und anschließend diskutiert, in wie weit der Spezifikationsrahmen des Arbeitskreises diese erfüllt. Darüber hinaus werden einige (strukturerhaltende) Weiterentwicklungen bzw. Änderungen vorgeschlagen, die sich vor allem auf die Eignung zur Spezifikation verschiedener Komponentenarten sowie eine verbesserte Kompatibilität zu existierenden Standards konzentrieren.

2 Das CompoNex-Projekt

Unter dem zusammenfassenden Titel „CompoNex“ (ein Kurzwort, das gleichzeitig für „Component Exchange“ – den Austausch von Komponenten – und „Component Nexus“ – die Verbindung von Komponenten – steht) wird zur Zeit im Rahmen eines Dissertationsprojekts die Schaffung einer Familie einheitlicher Werkzeuge zur Verwaltung bzw. zum Austausch von Komponenten sowohl für die unternehmensinterne als auch die unternehmensübergreifende Verwendung (beispielsweise beim Aufbau globaler Komponentenmarktplätze) angestrebt. Dabei steht vor allem der Aufbau modularer, jeweils für den Verwendungszweck angepasster Metainformationssysteme auf einer gemeinsamen Basis von spezialisierten (Repository-) Komponenten im Mittelpunkt.

Eine der dabei zu entwickelnden Komponenten ist der in diesem Beitrag angesprochene Komponentenkatalog, der die Speicherung von Komponenten und deren Spezifikationen realisiert. Darüber hinaus bietet er die Möglichkeit, Komponentenspezifikationen nach verschiedenen Kriterien zu durchsuchen (z.B. nach dem Enthaltensein eines vorgegebenen Texts, nach der Zugehörigkeit zu einer bestimmten Anwendungsdomäne etc.) und Komponenten zu entnehmen. Die angebotenen Dienste werden jedem Anwender durch eine eingebaute Benutzer- und Rechteverwaltung selektiv zur Verfügung gestellt. Der Komponentenkatalog wurde als eigenständige Komponente realisiert, d.h. er bietet seine Dienste über eine Schnittstelle an und lässt sich mit anderen Komponenten kombinieren.

Zur Implementierung wurde die Microsoft .NET Komponententechnologie eingesetzt. Hierdurch ergab sich die Möglichkeit, neben einer technologieabhängigen Schnittstelle (in Form eines .NET Interfaces) auch ohne großen Mehraufwand eine unabhängige XML Web-Service-Schnittstelle zu realisieren, wodurch die Kombination mit Komponenten verschiedener Plattformen ermöglicht wird. Die somit erreichte Unabhängigkeit von konkreten Komponententechnologien ist ein wichtiges Architekturkriterium, das die plattformübergreifende Verwendung des Katalogs (beispielsweise in verschiedenen plattformspezifischen CASE Tools) ermöglicht.

Neben dieser speziellen Anforderung an die Implementierung existieren noch eine Reihe konzeptioneller Anforderungen aus der Theorie der Komponentenorientierung, von denen einige im Folgenden

besonders hervorgehoben werden sollen:

Unterstützung verschiedener konzeptioneller Komponentenarten. Eine komponentenorientierte Anwendung besteht in der Regel aus verschiedenen Komponentenarten, die jeweils einen unterschiedlichen Beitrag zu deren Architektur leisten. Über eine allgemeine Komponentendefinition hinaus können daher die folgenden Komponentenarten (in Anlehnung an [Turo2002:1-3], [Szyp1998:273-279], [Griss2001], [LORS2001], [RaTu2001]) unterschieden werden:

- Systemkomponenten (die synonym auch als Middleware-Komponenten bezeichnet werden) stellen spezialisierte anwendungsunabhängige (generische) Dienste zur Verfügung. Beispiele für solche Komponenten sind Datenbank-Management-Systeme, Workflow-Management-Systeme oder Verschlüsselungsprogramme.
- System-Frameworks geben eine anwendungsunabhängige Architektur vor, gemäß der die zuvor genannten Systemkomponenten während des Prozesses der Anwendungsentwicklung an speziell dafür vorgesehenen Schnittstellen (hot spots) miteinander verbunden werden können. Auf diese Weise ergibt sich das aus dem klassischen Software Engineering bekannte Basissystem einer Anwendung. Der Definition folgend können beispielsweise Applikationsserver oder Betriebssysteme als System-Frameworks bezeichnet werden.
- Fachkomponenten realisieren spezialisierte anwendungsspezifische Dienste, beispielsweise eine Bilanzierung nach einer bestimmten Vorschrift oder eine Reisebuchung. Obwohl an dieser Stelle häufig Dienste aus dem Bereich betrieblicher Anwendungen genannt werden, können Fachkomponenten jedoch auch Dienste aus anderen Anwendungsbereichen realisieren (im Gegensatz zur Definition in [Turo2002:1], die eher sog. „Business Objects“ charakterisiert).
- Anwendungs-Frameworks stellen in Analogie zu den bereits genannten System-Frameworks eine Architektur für die Verbindung von Fachkomponenten zur Verfügung. Als Beispiel für ein solches Framework lässt sich das San Francisco Framework der IBM [IBM2000] anführen.

Während der Entwicklung einer komponentenorientierten Anwendung werden in der Regel mehrere Komponenten der einzelnen hier genannten Arten miteinander zu einem Gesamtsystem miteinander verknüpft [Szyp1998:278]. Die somit entstehende Anwendung lässt sich ebenfalls wiederum als (komplexe) Komponente betrachten, die beispielsweise im Enterprise Application Integration mit anderen Anwendungen kombiniert wird. Abb. 1 fasst die unterschiedenen Komponentenarten noch einmal grafisch zusammen. Ein Komponentenkatalog sollte in der Lage sein, alle diese zum Einsatz kommenden Arten gemeinsam verwalten (und dokumentieren) zu können, gleichzeitig jedoch auch zwischen diesen zu differenzieren.

Unterstützung verschiedener Arten der Wiederverwendung. Komponenten lassen sich auf verschiedene Arten wieder verwenden. Üblicherweise bezieht ein Konsument im Rahmen seiner Anwendungsentwicklung eine Kopie der Komponente und baut diese in seine Anwendung ein. In diesem Fall wird das Konzept der Komponente wieder verwendet. Allerdings erfolgt die Wiederverwendung in einem logischen Sinn, da eine gleiche (und nicht dieselbe) Komponente wieder verwendet wird.

Es ist jedoch auch möglich, während einer Anwendungsentwicklung auf dieselbe Komponente (also

die Originalfassung) zurückzugreifen. Dies geschieht über einen entfernten Aufruf der Komponente (unter Verwendung eines verteilten Anwendungskonzepts). Eine solchermaßen physische Wiederverwendung von Komponenten geschieht beispielsweise im Rahmen des Application Service Providing und hat mit der Entwicklung der XML Web-Services (die sich als von Dritten zur Verfügung gestellte – „hosted" – Komponenten charakterisieren lassen [Szyp2001], [Bett2001]) eine neue Bedeutung erlangt. Da mit beiden Arten der Wiederverwendung sowohl unterschiedliche Konfigurationsmaßnahmen als auch (zumindest in der Regel) unterschiedliche Lizenz- und Bezahlmodelle (z.B. Festpreise auf der einen und nutzungsabhängige Entgelte auf der anderen Seite) verbunden sind, sollten diese unterschieden werden.

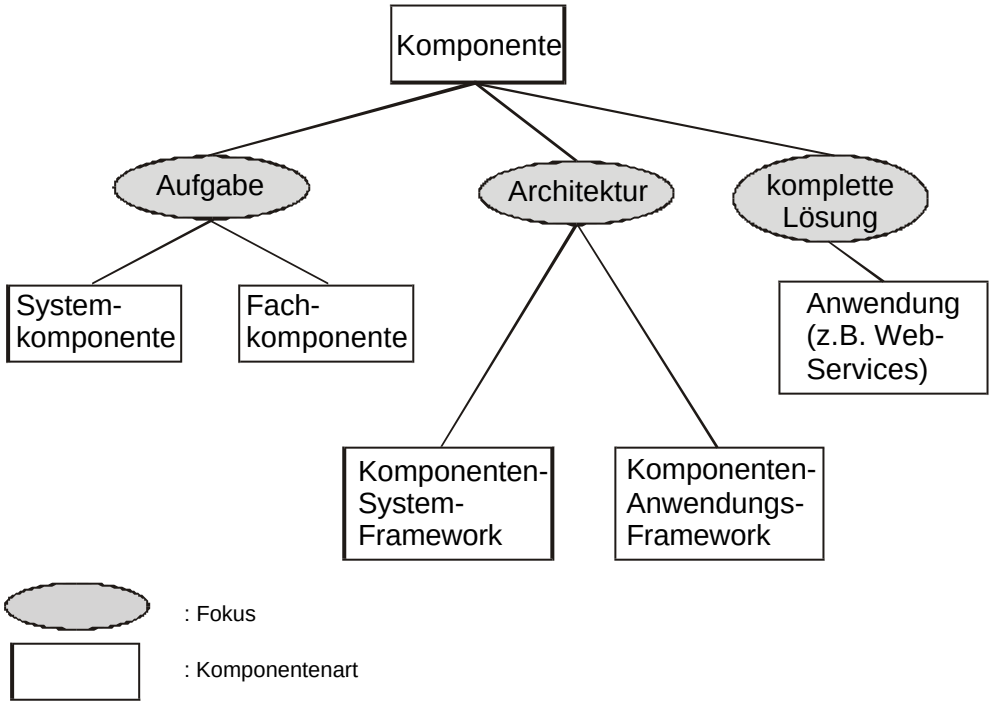

Abbildung 1 Komponentenarten

Unabhängig von der eben diskutierten logischen bzw. physischen Wiederverwendung können Komponenten unter Anpassung oder zumindest Kenntnisnahme des Quellcodes oder allein auf Grund der Kenntnis der Komponentenspezifikation (bzw. der Schnittstelle) in eine zu entwickelnde Anwendung eingebaut werden. In diesem Zusammenhang wird auch von White-Box-Wiederverwendung bzw. Black-Box-Wiederverwendung gesprochen. Insbesondere die White-Box-Wiederverwendung ist in der Komponentenorientierung umstritten, weil sie einem Konsumenten ermöglicht, die Implementierung einer Komponente zu studieren und dieses Spezialwissen bei der Konfiguration zu verwenden. Darüber hinaus ermöglicht sie im Prinzip auch die Anpassung der erhaltenen Implementierung (falls dies lizenzrechtlich gestattet ist). Gerade die Implementierung wird jedoch regelmäßig bei der Auslieferung neuer Versionen verändert und sollte daher nicht Gegenstand der Kenntnisnahme oder gar Anpassung durch einen Konsumenten sein [Szyp1998:34].

Andererseits führt die strikte Verwendung des Black-Box-Prinzips dazu, dass sich Konsumenten auf die (gekapselte) Implementierung eines Herstellers verlassen müssen. Damit geht in der Regel ein Vertrauensverlust für kleinere und unbekannte Komponentenhersteller einher, für die die Offenlegung ihrer Implementierung eine vertrauensschöpfende Maßnahme (vgl. z.B. auch Opensource Software) ist. Darüber hinaus wird in der Literatur gelegentlich die Offenlegung von Teilen der Implementierung gefordert, um die Eignung einer Komponente besser beurteilen zu können (sog. „Grey-Box-Wiederverwendung“ [BüWe1997]). Da die Diskussion hierüber noch nicht abgeschlossen ist, lohnt sich die Unterscheidung der beiden Ansätze.

Umfassende und übersichtliche Beschreibung der Außensicht. Die Auswahl von Komponenten erfolgt auf Grund ihrer Eignung für die geplante Anwendung. Diese lässt sich durch eine Analyse der Außensicht einer Komponente ermitteln, die vom Produzent in Form einer Komponentenspezifikation dokumentiert wird. Dabei ist darauf zu achten, dass die Dokumentation der Außensicht möglichst umfassend erfolgt und sich nicht alleine auf die technische Schnittstelle (d.h. die angebotenen Dienste) beschränkt.

Bei der Implementierung eines Komponentenkatalogs ist eine adäquate Spezifikation anzustreben, die die Bewertung der Eignung einer Komponente möglichst gut unterstützt. Dafür sind unterschiedliche (z.B. konzeptionelle und technische) Aspekte einer Komponente zu berücksichtigen. Eine geeignete Beschreibung sollte daher sowohl eine thematisch gegliederte als auch eine aspektorientierte Struktur besitzen, wodurch die Übersichtlichkeit bei einer gleichzeitigen flexiblen Erweiterbarkeit garantiert wird. Gleichzeitig sollte die Vergleichbarkeit von Komponentenspezifikationen durch die Vorgabe von Notationen, die bei einer Spezifikation zu verwenden sind, unterstützt werden.

Die Bewertung der Eignung einer Komponente umfasst vor allem eine Analyse möglicher Heterogenitäten, durch die eine Verwendung im Rahmen der Konfigurierung erschwert bzw. ausgeschlossen wird. Es lassen sich drei Arten von Heterogenitäten unterscheiden:

- Syntaktische (technische) Heterogenität tritt auf, wenn Komponenten für unterschiedliche Plattformen (Komponententechnologien, Betriebssysteme etc.) implementiert wurden. Sie kann durch die Verwendung plattformneutraler Konnektoren (Adapter), die in der Regel eine standardisierte Beschreibung der Schnittstelle einer Komponente beinhalten, kompensiert werden. Durch die Wahl von Komponenten, die ausnahmslos für dieselbe Plattform geschrieben wurden, kann sie verhindert werden.
- Semantische Heterogenität tritt auf, wenn Komponenten anwendungsspezifische Begriffe verschiedenartig implementieren (beispielsweise den Begriff „Konto“ jeweils unterschiedlich verstehen). Sie kann durch eine explizite Auflistung der Begriffe mitsamt den zugehörigen Definitionen messbar gemacht werden. Maßnahmen zur Überwindung basieren in der Regel auf dem Bau von Übersetzern, die begriffliche Verschiedenheiten ggf. kompensieren. Sie kann verhindert werden, falls bei der Konfigurierung einer Anwendung ausschließlich Komponenten verwendet werden, die den gleichen begrifflichen Standard implementieren.
- Pragmatische Heterogenität tritt auf, wenn Komponenten mit einem jeweils unterschiedlichen Verständnis über den anwendungsspezifischen (gemeinsamen, komponentenübergreifenden) Kontrollfluss implementiert wurden. So wäre es beispielsweise bei der Abwicklung einer Bestellung im Rahmen eines Geschäftsprozesses denkbar, dass eine Komponente A davon ausgeht, dass die Lagerbuchführung zu einem späteren Zeitpunkt informiert wird, während

eine nachfolgende Komponente B erwartet, dass dies zu einem vorhergehenden Zeitpunkt bereits erfolgt ist. Diese Form der Heterogenität ist nur schwierig zu beseitigen (in der Praxis erreicht man dies beispielsweise durch den Einsatz von Frameworks, die einen gemeinsamen Ablauf – Orchestrierung – implementieren). Eine entsprechende Heterogenität kann durch eine Dokumentation der von Komponenten jeweils implementierten (Teil-) Prozesse explizit gemacht werden.

Der zu implementierende Komponentenkatalog sollte (unter anderem) die Beurteilung evtl. auftretender Heterogenitäten ermöglichen, indem er jeweils angemessene Dokumentationen zur Verfügung stellt.

Trennung der Komponenten- und Schnittstellenspezifikation. Komponenten stellen ihre Dienste definitionsgemäß über eine Schnittstelle zur Verfügung, die gleichzeitig eine wichtige Grundlage für die Auswahl durch einen Konsumenten bildet. Austauschbare Komponenten implementieren dabei in der Regel dieselbe (oder eine kompatible) Schnittstelle. Auf Grund dieser wichtigen Eigenschaft sollte die Spezifikation einer Schnittstelle von der Spezifikation einer Komponente, die diese implementiert, getrennt werden. Auf diese Weise wird es möglich, Schnittstellen als eigene Objekte (der Standardisierung) zu betrachten und die Auswahl geeigneter Komponenten auf Basis einer Schnittstellendefinition vorzunehmen.

Die Schnittstellenspezifikation als Auswahlkonzept geht dabei deutlich über die Angabe der technischen Schnittstelle hinaus und beinhaltet alle für die Bewertung notwendigen Angaben (insbesondere auch Angaben über die Semantik und Performance einer Komponente [Szyp1998:43-46], [GrPf1998]). Komponentenspezifikationen verweisen gegebenenfalls auf diese Schnittstellenspezifikationen (sofern sie diese Schnittstellen implementieren).

Berücksichtigung variierender Funktionalität. Komponentenorientierte Anwendungen zeichnen sich (neben anderen Kriterien) vor allem durch eine verbesserte Unterstützung der Anpassung aus. Dies kann (wenn eine Black-Box-Wiederverwendung vorausgesetzt wird) zum einen durch den Austausch entsprechender Komponenten gegen Varianten mit der gewünschten Funktionalität (die sie über die gleiche Schnittstelle anbieten wie die ausgetauschte Komponente) geschehen. Falls Komponenten die Steuerung ihrer Funktionalität über entsprechende Parameter zulassen, wäre es andererseits auch denkbar, eine Anpassung durch Veränderung der Parametrisierung zu erreichen. Bei der Implementierung eines Komponentenkatalogs ist daher darauf zu achten, dass allein stehende Varianten geeignet zu Baureihen zusammengefasst bzw. vorhandene Möglichkeiten der Parametrisierung dokumentiert werden.

Darüber hinaus sollte ein Komponentenkatalog verschiedene Versionen einer Komponente unterscheiden können, da sich durch einen Versionswechsel ggf. Inkompatibilitäten in Form einer geänderten Schnittstelle bzw. einer geänderten Implementierung ergeben können. Bei der Auswahl geeigneter Komponenten kann daher die Version (einer Komponente bzw. der von ihr implementierten Schnittstelle) von entscheidender Bedeutung sein.

Kompatibilität zu etablierten Komponentenkatalogen. Als letzte wichtige Anforderung ist die geforderte Kompatibilität zu etablierten Komponentenkatalogen zu nennen. Hierdurch ergibt sich die Möglichkeit, dass evtl. vorhandene Werkzeuge, die für solche Kataloge bereits implementiert wurden, auch auf den zu implementierenden Komponentenkatalog zugreifen können (und sich mit geringem Aufwand optimal auf dessen vollständige Nutzung anpassen lassen).

Die Kompatibilität zu etablierten Komponentenkatalogen steht allerdings unter dem Vorbehalt, dass diese ausgereift genug sind und den zuvor genannten Kriterien genügen (bzw. so erweitert werden können, dass dies der Fall ist; dann spricht man von Abwärtskompatibilität). Aus diesem Grund ist explizit eine Kompatibilität zu den bereits am Markt befindlichen Komponentenmarktplätzen (beispielsweise ComponentSource, www.componentsource.com, und Flashline, www.flashline.com) nicht verlangt. In der Tat existieren derzeit kaum ausgereifte Komponentenkataloge oder Spezifikationsrahmen, die insbesondere den vorgenannten Kriterien gerecht werden [GrPf1998], [ABC+2001]. Ein sorgfältig entwickelter Standard für die Katalogisierung von XML Web-Services (die sich – wie bereits diskutiert – als spezielle Komponenten auffassen lassen), ist UDDI – Universal Description, Discovery, and Integration [UDDI2000]. Dieser wird gegenwärtig von einem Industriekonsortium getragen und zur Weiterentwicklung an ein neutrales Standardisierungsgremium abgegeben. Der Standard wird bereits von zahlreichen Werkzeugen unterstützt und sollte deshalb in Kompatibilitätserwägungen einbezogen werden.

Ein weiterer Standard, der bei der Entwicklung eines Komponentenkatalogs zu berücksichtigen ist, ist der Vorschlag zur vereinheitlichten Spezifikation von Fachkomponenten, der von einer Arbeitsgruppe der Gesellschaft für Informatik (im Austausch zwischen Universitäten und Industrieunternehmen) entwickelt wurde [Turo2002]. Auf Grund der unterschiedlichen beteiligten Interessengruppen ist davon auszugehen, dass dieser Standard sowohl Bedeutung an Universitäten (beispielsweise in der Lehre) als auch in der Industrie erlangen wird.

3 Das Memorandum zur vereinheitlichten Spezifikation von Fachkomponenten als Grundlage eines Komponentenkatalogs

Der entwickelte Komponentenkatalog beinhaltet als zentrales Architekturelement ein Metaschema, das die Struktur der zu speichernden Spezifikationen einer Komponente vorgibt. Dieses Metaschema wurde auf der Basis eines Spezifikationsrahmens entwickelt, der somit von zentraler Bedeutung ist. Die in Kapitel 2 genannten Anforderungen an den Komponentenkatalog sind daher insbesondere auch an den zugrunde liegenden Spezifikationsrahmen zu stellen.

Im Rahmen des Entwicklungsprojekts wurden sowohl der in UDDI enthaltene Spezifikationsrahmen [UDDI2000] als auch der Vorschlag zur vereinheitlichten Spezifikation von Fachkomponenten des Arbeitskreises 5.10.3 der Gesellschaft für Informatik [Turo2002] herangezogen. Dabei ergab sich zunächst das Problem, dass der in UDDI enthaltene Spezifikationsrahmen für XML Web-Services getrennt von dem diesen umfassenden Spezifikationsrahmen zur Dokumentation von Unternehmen zu betrachten war, was im Standard jedoch nicht vorgesehen ist (dort wird ein integrierter Spezifikationsrahmen sowohl zur Beschreibung von Unternehmen als auch der von diesen angebotenen Dienste – XML Web-Services – vorgegeben [Cera2002:158]).

Zusammenfassend lässt sich jedoch festhalten, dass UDDI zur Spezifikation eines XML Web-Services allgemeine Informationen über diesen in Form eines Namens und einer informalen Beschreibung umfasst (die als White-Pages bezeichnet werden). Darüber hinaus kann der angebotene Dienst durch Angabe einer Anwendungsdomäne klassifiziert werden (Yellow-Pages). Schließlich wird in den sog. Green-Pages die Schnittstelle des angebotenen Dienstes sowie deren Ort (in Form einer Internet-Adresse) erfasst [CCC+2001:187-198].

Der Spezifikationsrahmen von UDDI verfügt über eine thematische Gliederung der Spezifikationen.

White-Pages beinhalten allgemeine Angaben, Yellow-Pages fassen klassifizierende Merkmale zusammen, und Green-Pages befassen sich schließlich mit technischen Spezifikationen (zum Aufruf des angebotenen Dienstes). Darüber hinaus ist es möglich, variierende Funktionalität durch die Spezifikation von Web-Service-Gruppen in einer gemeinsamen Spezifikation abzubilden [CCC+2001:190]. Als Grundlage für die Implementierung eines Komponentenkataloges kann dieser Spezifikationsrahmen dennoch nicht ohne umfangreiche Modifikationen herangezogen werden, da er die in Kapitel 2 genannten Anforderungen nur in Teilen erfüllt. So unterstützt er insbesondere nicht die verschiedenen Wiederverwendungsarten, sondern konzentriert sich auf physisch wieder zu verwendende XML Web-Services. Darüber hinaus ist die im Rahmen von UDDI vorgeschlagene Spezifikation wenig umfassend und erlaubt insbesondere nicht die Beurteilung der Semantik (also der implementierten Begriffe) bzw. Pragmatik (also der implementierten Prozesse) einer Komponente.

Das Memorandum des Arbeitskreises „Komponentenorientierte betriebliche Anwendungssysteme“ (vgl. [Turo2002]) beinhaltet einen detaillierten Spezifikationsrahmen zur Dokumentation von Fachkomponenten. Sein Ziel ist die möglichst vollständige Beschreibung der Außensicht einer Komponente, womit bereits eine der im vorigen Kapitel genannten Anforderungen erfüllt wird. Durch seinen ebenenorientierten (aspektorientierten) Ansatz ist der Spezifikationsrahmen zudem flexibel erweiterbar, etwa durch die Einführung neuer Ebenen (durch Betreiber eines Komponentenkatalogs etc.). Er gliedert sich in die Ebenen „Vermarktung“, „Qualität“, „Aufgabe“, „Terminologie“, „Abstimmung“, „Verhalten“ und „Schnittstelle“, die in jedem Werkzeug vorhanden sein müssen (vgl. Abb. 2). Im Rahmen der Vermarktungsebene werden zunächst allgemeine Angaben über Fachkomponenten (z.B. der Name, Hersteller, Version etc.) zusammengefasst. Die Qualitätsebene konzentriert sich auf die Spezifikation nicht-funktionaler Eigenschaften (z.B. Laufzeitverhalten, Antwortzeit etc.), die für die Auswahl einer Komponente entscheidend sein können.

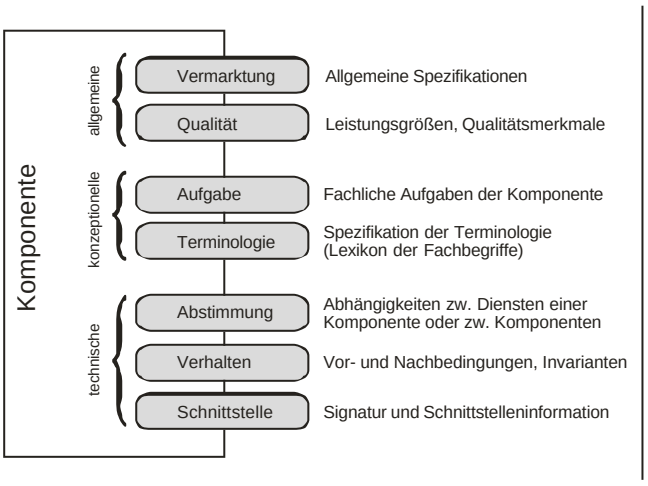

Abbildung 2 Standardisierter Spezifikationsrahmen

Die Aufgabenebene erfasst (auf konzeptionellem Niveau) die von einer Komponente unterstützten Aufgaben und ihre Zerlegung in Teilaufgaben, zielt also im Prinzip auf die Beschreibung des Prozesses (z.B. eines Geschäftsprozesses), der einer Implementierung zugrunde liegt. Die Terminologieebene dient schließlich der Erfassung relevanter (konzeptioneller) Begriffe und ihrer Definitionen, die bei

der Implementierung einer Komponente zugrunde gelegt wurden – sie erfasst also die fachliche Semantik.

Die Abstimmungsebene beschreibt schließlich einzuhaltende Reihenfolgen zwischen den Diensten einer Komponente bzw. Aufrufe zu anderen Komponenten, die bei der Benutzung einer Komponente einzuhalten sind. Auf diese Weise wird der durch die Aufgabenebene abstrakt (losgelöst von der Implementierung) beschriebene Ablauf (der konzeptionelle Prozess) durch die Angabe des zugehörigen technischen Ablaufs konkretisiert. Die Verhaltensebene beschreibt darüber hinaus die (technische) Semantik einer Fachkomponente durch Angabe von Vor- und Nachbedingungen zu den einzelnen an der Schnittstelle angebotenen Diensten. Diese stimmen mit den Definitionen der zugehörigen Begriffe der Terminologieebene überein bzw. lassen sich aus diesen ableiten. Insofern lässt sich die Beschreibung der technischen Semantik zumindest teilweise auf die Beschreibung der fachlichen Semantik zurückführen. Die Schnittstellenebene schließlich spezifiziert die angebotenen Dienste einer Fachkomponente und deren Signaturen für den Aufruf durch Dritte.

Bei der Beschreibung der einzelnen Ebenen fällt zunächst auf, dass die insgesamt sieben Ebenen des Spezifikationsrahmens nicht explizit thematisch gegliedert wurden, wodurch das Verständnis (und darüber hinaus auch die Beantwortung der Frage: „Welche Informationen richten sich an wen?“) für den Betrachter einer Komponentenspezifikation zunächst (auf Grund der Fülle der Spezifikationen) erschwert wird. Eine solche Gliederung ist jedoch durchaus möglich: So lassen sich die Ebenen „Vermarktung“ und „Qualität“ dem thematischen Bereich „allgemeine und nicht-funktionale Spezifikationen“ zuordnen, die als Kurzübersicht auch für einen „Laien“ (Nicht-Konfigurierer) verständlich sind. Die Ebenen „Aufgabe“ und „Terminologie“, die sich unter dem Thema „konzeptionelle Spezifikationen“ zusammenfassen lassen, wenden sich hingegen vor allem an Experten, die die fachliche Eignung einer Komponente beurteilen können. Und schließlich dienen die Ebenen „Abstimmung“, „Verhalten“ und „Schnittstelle“, die als „technische Spezifikationen“ zusammengefasst werden können, vor allem der Beurteilung der technischen (algorithmischen) Eignung einer Komponente bzw. des entsprechenden Konfigurationsaufwands.

Positiv fällt ebenfalls auf, dass der Spezifikationsrahmen des Memorandums die Vorgabe von Notationen für die einzelnen Spezifikationen anstrebt, wodurch eine gewisse Homogenität erreicht wird. Allerdings wird dieses Ziel im Rahmen der Qualitätsebene nicht erreicht, weil dort weder zu ersehen ist, was zu spezifizieren ist, noch mit welchen Notationen dies erfolgen könnte. Weiterhin ist anzumerken, dass der vorgeschlagene Spezifikationsrahmen zunächst einem sehr allgemeinen Ansatz zur Spezifikation von Komponenten folgt, der durchaus für alle in Kapitel 2 genannten Komponentenarten verwendet werden könnte [Over2002]. Insbesondere lassen sich auch XML Web-Services durch diesen Spezifikationsrahmen umfassender beschreiben als durch den von UDDI vorgeschlagenen (der de facto eine Teilmenge des hier diskutierten Memorandums darstellt [OvTh2002]). Dennoch beschränkt sich das Memorandum auf die Spezifikation von Fachkomponenten (noch dazu mit einer rein betrieblichen Ausrichtung, vgl. [Turo2002:1]).

Auch die verschiedenen Arten der Wiederverwendung können im Spezifikationsrahmen des Memorandums nicht explizit unterschieden werden, weshalb man an dieser Stelle auf die Angabe in der allgemeinen Komponentenbeschreibung vertrauen (bzw. hoffen) muss. Zur Zeit ist eine Spezifikation variierender Funktionalität ebenfalls nicht möglich – allerdings muss bei der Beurteilung darauf verwiesen werden, dass ein entsprechender Ansatz zumindest für parametrisierbare Fachkomponenten derzeit in einem Forschungsprojekt, das an der Universität Augsburg angesiedelt ist, erarbeitet wird.

Zusammenfassend lässt sich festhalten, dass zur Implementierung eines Komponentenkatalogs, der den oben genannten Anforderungen genügen soll, zunächst keiner der beiden analysierten Spezifikationsrahmen direkt verwendet werden konnte. Jedoch sind beide Ansätze von ihrer Struktur her kompatibel und können sich gegenseitig zum Vorteil befruchten. So beinhaltet der Spezifikationsrahmen des Arbeitskreises die in UDDI vorgesehenen Spezifikationen und kann somit als dessen Obermenge betrachtet werden. Andererseits lässt sich die in UDDI vorhandene thematische Gliederung auf den Spezifikationsrahmen des Arbeitskreises übertragen, wodurch eine bessere Übersichtlichkeit erreicht wird. Beide Spezifikationsrahmen berücksichtigen jedoch nicht die getrennte Spezifikation von Komponenten (Implementierungen) und Schnittstellen, die eine wichtige Anforderung darstellt.

4 Vorschläge zur Weiterentwicklung des Memorandums

In diesem Kapitel sollen zunächst einige strukturerhaltende Erweiterungen für den Spezifikationsrahmen des Arbeitskreises vorgeschlagen werden, die bei der Implementierung des Komponentenkatalogs vorgenommen wurden. Dabei wird auf die Trennung von Spezifikationen, die sich auf Schnittstellen bzw. auf Komponenten beziehen, die diese implementieren, zunächst verzichtet – hierdurch würde die Struktur des Spezifikationsrahmens tief greifend verändert.

Stattdessen werden im Folgenden allgemeine Änderungen, Erweiterungen zur Spezifikation verschiedener Komponentenarten und die Einführung einer thematischen Gliederung beschrieben.

4.1 Allgemeine Änderungsvorschläge

Während der Implementierung des Komponentenkatalogs wurden einzelne Änderungen vollzogen, die sich vor allem auf die Vermarktungsebene und die Qualitätsebene des Spezifikationsrahmens auswirken. Im Rahmen der Vermarktungsebene wurde der Spezifikation der Version einer Komponente zunächst ein Spezifikationsbereich für die Dokumentation variierender Funktionalität an die Seite gestellt (sei es durch die Angabe einer Parametrisierung oder einer Bezeichnung, die eine allein stehende Variante spezifiziert). Darüber hinaus wurde die Angabe mehrerer Ansprechpartner erlaubt und das Spezifikationsfeld „Vertriebskanäle“ eingeführt, das mehrere Vertriebskanäle sowie die jeweils davon abhängigen Preise und unterstützten Bezahlverfahren beinhaltet.

Beim Entwurf des Metaschemas wurde auf die Übernahme der Qualitätsebene auf Grund der bereits in Kapitel 3 angesprochenen Mängel verzichtet. Da dort weder die zu spezifizierenden Sachverhalte verbindlich benannt noch Notationen für die Spezifikation vorgegeben wurden, erweist sich diese Ebene als nur schwer implementierbar. Dennoch soll auf die Spezifikation wichtiger nicht-funktionaler Eigenschaften nicht verzichtet werden. Zur Angabe des Laufzeitverhaltens einer Komponente wurde daher die Leistungs-Ebene eingeführt, die sich mit Angaben über die zu erwartende Antwortzeit, die Servicequalität sowie den Datendurchsatz befasst. Diese können unter Verwendung komplexitätstheoretischer Maße (beispielsweise der bekannten „O-Notation“ [Szyp1998:46]) in einer einheitlichen (plattformunabhängigen) Notation spezifiziert werden.

Neben dieser Ebene wurde die Sicherheitsebene eingeführt, die Spezifikationen über die Integrität und die Zugriffssteuerung einer Komponente zusammenfasst. Zunächst lässt sich dort angeben, wie eine Komponente generell mit (kritischen) Daten umgeht (Datenschutzpolitik, Verschlüsselung etc.) und die Authentifizierung von Benutzern gewährleistet. Damit wird dem vor allem bei Konsumenten

vorhandenem Bedürfnis nach Dokumentation von Sicherheitsaspekten als Voraussetzung zur Verwendung von Fremdsoftware Rechnung getragen. Die aktuelle Diskussion (die z.B. auch bei Microsoft unter dem Schlagwort „Trustworthy Computing“ geführt wird) zeigt die Bedeutung dieser Spezifikationen, die ggf. auch zertifiziert werden können. Darüber hinaus kann in der Sicherheits-Ebene hinterlegt werden, welche Benutzer auf bestimmte Funktionen (beispielsweise zur Administration einer Komponente) zugreifen dürfen (Benutzerrechte etc.). Damit wird eine bessere Steuerung der Benutzerzugriffe ermöglicht.

Neben diesen Änderungen wurde vor allem die Aufgabenebene dahingehend angepasst, dass nicht mehr nur Aufgaben und deren Zerlegung in Teilaufgaben, sondern jeweils die vollständigen einer Komponente bzw. Schnittstelle zugrunde liegenden konzeptionellen Prozesse (z.B. Geschäftsprozesse) spezifiziert werden, die unter anderem auch eine Aufgabenzerlegung beinhalten. Somit beinhaltet eine Komponentenspezifikation neben einer konzeptionellen (Terminologieebene) und technischen (Verhaltensebene) Dokumentation der Semantik auch eine konzeptionelle (Aufgabenebene) und technische (Abstimmungsebene) Spezifikation der Pragmatik. Hierdurch ist die Beurteilung eventuell auftretender semantischer und pragmatischer Heterogenitäten jeweils auf konzeptioneller und technischer Ebene möglich.

4.2 Erweiterungen zur Spezifikation verschiedener Komponentenarten

Der im Memorandum enthaltene Spezifikationsrahmen erlaubt auf Grund seines umfassenden Ansatzes ohne größere Veränderung seiner Struktur (also der einzelnen Ebenen) die Dokumentation verschiedener Komponentenarten.

Ein erstes Anliegen ist dabei die (vereinheitlichte) Spezifikation von Fach- und Systemkomponenten sowie Anwendungs- und System-Frameworks. Hierzu wurde beim Entwurf des Komponentenkatalogs zunächst das Attribut „Komponentenart“ mit dem entsprechenden Wertebereich in der Vermarktungsebene eingeführt. Dadurch wird es möglich, die genannten Komponentenarten mit einem einheitlichen Rahmen zu spezifizieren und in einem gemeinsamen Komponentenkatalog abzulegen. Durch diese Erweiterung wurde allerdings die bislang im Memorandum vorhandene Konzentration auf Fachkomponenten (mit betrieblicher Konzeption) aufgegeben, wodurch weitere Anpassungen am Spezifikationsrahmen des Arbeitskreises notwendig wurden.

So wird im Rahmen der Aufgabenebene nicht mehr von betrieblichen Aufgaben einer Fachkomponente, sondern nur noch allgemein von Aufgaben einer Komponente bzw. dem einer Komponente zugrunde liegenden konzeptionellen Prozess gesprochen. Eine allgemeine Unterscheidung von Aufgabenklassen kann wie folgt gemacht werden: Systemkomponenten und System-Frameworks realisieren anwendungsunabhängige (generische) Aufgaben. Im Falle eines Datenbank-Management-Systems (einer typischen Systemkomponente) ist dies zum Beispiel die Aufgabe „Persistente Speicherung von Informationen“. Hingegen implementieren Fachkomponenten und Anwendungs-Frameworks anwendungsspezifische (fachliche) Aufgaben, beispielsweise eine Bilanzierung nach den Vorgaben des Handelsgesetzbuches.

Diese Verallgemeinerung von betrieblichen Aufgaben hin zu einer generellen Einteilung von allgemeinen Aufgaben beeinflusst auch den Wertebereich des Spezifikationsfelds „Domäne“ in der Vermarktungsebene, der nun nicht mehr auf die einzelnen Funktionalbereiche eines Industriebetriebs beschränkt werden kann. Stattdessen wird eine mehrstufige aufgabenbezogene Gliederung von Domä-

nen vorgeschlagen, und zwar ausgehend von der Unterscheidung generischer und fachlicher Domänen. Generische Domänen in diesem Sinne wäre beispielsweise „Speicherung (von Informationen)“, „Steuerung (von Prozessen)“, „Folgerung (von Sachverhalten)“, „Erwägung (von Zusammenhängen)“ etc. Fachliche Domänen wären „Betriebswirtschaft“, „Gesundheitswesen“, „Kredit- und Versicherungsgewerbe“ (wie im Spezifikationsrahmen beispielsweise unter „Wirtschaftszweig“ und „Domäne“ genannt) etc.

Neben diesen Erweiterungen wurden die Attribute „Wiederverwendungsart“ (Wertebereich: logisch, physisch) und „Anpassungsart“ (Wertebereich: White-Box, Black-Box) in die Vermarktungsebene aufgenommen, wodurch weiteren Anforderungen aus Kapitel 2 genügt wird. Die Wiederverwendungsart gibt darüber Auskunft, ob eine Komponente physisch oder logisch wieder verwendet werden kann, während die Anpassungsart angibt, ob eine Komponente durch Veränderung ihres Programmcodes (white-box) oder lediglich durch Parametrisierung bzw. Auswahl einer geeigneten Variante (black-box) angepasst werden kann.

4.3 Einführung einer thematischen Gliederung und Anpassung an UDDI

Nach den vorgenommenen Erweiterungen bzw. Änderungen besteht der implementierte Spezifikationsrahmen aus bislang acht Ebenen, die sich den drei Themenkomplexen „allgemeine und nicht-funktionale“, „konzeptionelle“ und „technische Spezifikationen“ zuordnen lassen. Aus Gründen der besseren Übersichtlichkeit wurde bei der Entwicklung des Komponentenkatalogs eine thematische Gliederung konzipiert, die obige Themenkomplexe beinhaltet und darüber hinaus kompatibel zu UDDI ist.

Die Kompatibilität zu UDDI wurde auf Grund der in Kapitel 3 genannten Anforderungen an den zu implementierenden Spezifikationsrahmen berücksichtigt und ist darüber hinaus ohne Änderungen in der Struktur des Spezifikationsrahmens des Arbeitskreises herstellbar: Hierzu musste lediglich eine weitere Anpassung vorgenommen werden, die in der Auslagerung der Domänenspezifikation in eine eigenständige Domänenebene bestand. Dann kann folgende Analogie zwischen dem Spezifikationsrahmen des Arbeitskreises und dem von UDDI hergestellt werden:

- Die technischen Spezifikationen der Schnittstellen-, der Verhaltens- und der Abstimmungsebene lassen sich den Green-Pages von UDDI zuordnen. Hierdurch wird der Spezifikationsrahmen von UDDI, der bislang lediglich aus der Schnittstellenebene bestand, (abwärtskompatibel) um Angaben zur technischen Semantik und Pragmatik erweitert.
- Die konzeptionellen Spezifikationen der (modifizierten) Aufgaben- und Terminologieebene lassen sich als Blue-Pages in UDDI einordnen. Bislang fehlen konzeptionelle Spezifikationen in UDDI vollständig. Somit stellen die beiden genannten Ebenen eine strukturelle Erweiterung dar.
- Die neu entstandene Domänenebene lässt sich den Yellow-Pages von UDDI zuordnen und besitzt die dort auch bislang bereits vorhandene Funktionalität.
- Die allgemeinen und nicht-funktionalen Spezifikationen der Vermarktungs-, Leistungs- und Sicherheitsebene lassen sich den White-Pages von UDDI zuordnen, die bislang nur wenige allgemeine Spezifikationen umfassen. Somit werden auch diese Spezifikationen erheblich erweitert.

Die Abbildung des Spezifikationsrahmens des Arbeitskreises auf den in UDDI beinhalteten wird detailliert in [OvTh2002] besprochen. Auf eine erneute Ausarbeitung soll daher an dieser Stelle verzichtet und statt dessen auf Abb. 3 verwiesen werden, die den entstandenen weiterentwickelten Spezifikationsrahmen noch einmal grafisch zusammenfasst.

Es ist zu überlegen, ob es nicht sinnvoll ist, die durch Einführung einer thematischen Struktur (white, yellow, blue, green pages) erzielte Kompatibilität zu UDDI auch im Memorandum explizit zu verankern. Hierdurch würden die Investitionen der Software-Industrie bei der Implementierung neuer Werkzeuge gesichert und gleichzeitig eine mögliche Weiterentwicklung hin zu einer vereinheitlichten Spezifikation zahlreicher Komponenten aufgezeigt.

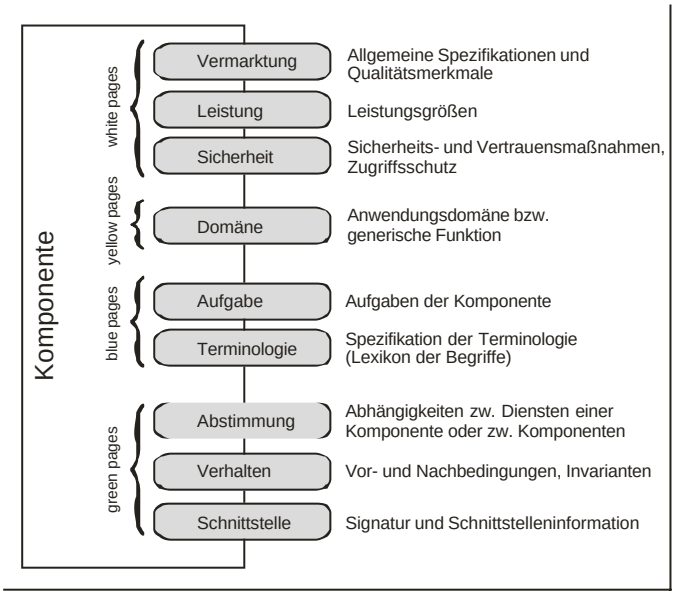

Abbildung 3 Weiterentwickelter Spezifikationsrahmen

5 Bewertung und Ausblick

Durch die Heranziehung der Spezifikationsrahmen des Arbeitskreises und des UDDI-Standards konnten bei der Implementierung des Komponentenkatalogs fast alle Anforderungen (die in Kapitel 2 zusammengefasst sind) erfüllt werden. So ist es möglich, in einem Katalog verschiedene Komponentenarten zu spezifizieren, abzulegen und darüber hinaus ihre Wiederverwendungs- und Anpassungsmöglichkeiten anzugeben. Die Auswahl wird durch eine vollständige Spezifikation der Außensicht einer Komponente unterstützt, die insbesondere die Beurteilung möglicher Heterogenitäten zwischen verschiedenen Komponenten berücksichtigt.

Schließlich sind im Komponentenkatalog abgelegte Spezifikationen übersichtlich thematisch gegliedert und (abwärts-) kompatibel zu dem international standardisierten Spezifikationsrahmen von UDDI. Nicht angesprochen wurde bislang jedoch die Trennung der Komponentenspezifikationen von den Spezifikationen der Schnittstellen, die von Komponenten implementiert werden. Eine solche Trennung lässt sich erreichen, indem große Teile des Spezifikationsrahmens (der auf die Spezifikation von Komponenten zielt) für die Spezifikation von Schnittstellen „umgewidmet“ werden. Dies gilt ins-

besondere für die Green-Pages, Blue-Pages und Teile der White-Pages. Eine Komponentenspezifikation erhält dann einen oder mehrere Verweise auf diejenigen Schnittstellenspezifikationen, die von der zugehörigen Komponente implementiert werden.

Eine vollständige Komponentenspezifikation besteht sodann aus der Komponentenspezifikation im engeren Sinn (also einer Spezifikation, die sich auf die Implementierung bezieht), sowie der Summe aller von einer Komponente implementierten Schnittstellenspezifikationen. Diese Spezifikationen werden vom Komponentenkatalog als einheitliche Spezifikation angezeigt, wodurch beim Anzeigen von Komponenten die Konformität zum Spezifikationsrahmen des Arbeitskreises weitgehend gewahrt bleibt. Beim Einstellen von Komponenten müssen jedoch zuvor die implementierten Schnittstellen separat spezifiziert worden sein.

Literatur

[ABC+2001] *Apperly, H.; Booch, G.; Councill, B.; Griss, M.; Heineman, G. T.; Jacobson, I.; Latchem, S.; McGibbon, B.; Norris, D.; Poulin, J.:* The Near-Term Future of Component-Based Software Engineering. In: *Councill, W. T.; Heineman, G. T. (eds.):* Component-Based Software Engineering: Putting the Pieces Together. Addison Wesley, Upper Saddle River, New Jersey 2001.

[Bett2001] *Bettag, U.:* Web-Services. In: Informatik Spektrum 24 (2001) 5, S. 302 – 304.

[BüWe1997] *Büchi, M.; Weck, W.:* A Plea for Grey-Box Components, TUCS Technical Report 122, 1997.

[CCC+2001] *Cauldwell, P.; Chawla, R.; Chopra, V.; Damschen, G.; Dix, C.; Hong, T.; Norton, F.; Ogbuji, U.; Olander, G.; Richmann, M. A.; Saunders, K.; Zaev, Z.:* Professional XML Web Services. Wrox Press, Birmingham 2001.

[Cera2002] *Cerami, E.:* Web Services Essentials. O'Reilly, Sebastopol (California) 2002.

[Fett2002] *Fettke, P.:* Aktuelle Übersicht über Komponentenmärkte. http://www.tu-chemnitz.de/wirtschaft/wi2/projects/components/, Abruf am 10.8.2002.

[Griss2001] *Griss, M. L.:* CBSE Success Factors: Integrating Architecture, Process, and Organization. In: *Councill, W. T.; Heineman, G. T. (Hrsg.):* Component-Based Software Engineering: Putting the Pieces Together. Addison Wesley, Upper Saddle River, New Jersey 2001.

[GrPf1998] *Gruntz, D.; Pfister, C.:* Komponentensoftware und ihre speziellen Anforderungen an Komponentenstandards. In: Object Spektrum (1998) 4, S. 38 – 42. http://www.cs.fh-aargau.ch/~gruntz/papers/OBJEKTSpektrum98/, Abruf am 10.8.2002

[IBM2000] *IBM Corporation (Hrsg.):* IBM San Francisco Overview. IBM Corporation, 2000. http://www.ibm.com/software/ad/sanfrancisco, Abruf am 10.8.2002.

[LORS2001] *Luyten, D.; Ortner, E.; Rzehak; K., Sahm, S.:* Muster, Frameworks und Fachkomponenten in der Anwendungsentwicklung. Arbeitsberichte des Fachgebiets Wirtschaftsinformatik I, TU Darmstadt 2001.

[Ortn1999] *Ortner, E.:* Repository Systems. Teil 1: Mehrstufigkeit und Entwicklungsumgebung. In: Informatik Spektrum 22 (1999) 4, S. 235 – 251. Teil 2: Aufbau und Betrieb eines Entwicklungsrepositoriums. In: Informatik Spektrum 22 (1999) 5, S. 351 – 363.

[Over2002] *Overhage, S.:* Die Spezifikation – kritischer Erfolgsfaktor der Komponentenorientierung. In: Turowski, K. (Hrsg.): Tagungsband 4. Workshop Komponentenorientierte betriebliche Anwendungssysteme. Universität Augsburg, Juni 2002, S. 1 – 17.

[OvTh2002] *Overhage, S.; Thomas, P.:* On Specifying XML Web Services Using UDDI Improvements. Zur Veröffentlichung angenommen: NetObjectdays 2002, Erfurt

[RaTu2001] *Rautenstrauch, C.; Turowski, K.:* Common Business Component Model (COBCOM): Generelles Modell komponentenbasierter Anwendungssysteme. In: *Buhl, H. U. et al (Hrsg.):* Information Age Economy. Physica Verlag, Heidelberg 2001.

[Szyp1998] *Szyperski, C.:* Component Software: Beyond Object-Oriented Programming. Addison-Wesley, Harlow 1998.

[Szyp2001] *Szyperski, C.:* Components and Web Services. In: Software Development Magazine 9 (2001) 8. http://www.sdmagazine.com/documents/sdm0108c/, Abruf am 10.8.2002.

[Turo2002] *Turowski, K. (Hrsg.):* Vereinheitlichte Spezifikation von Fachkomponenten. Gesellschaft für Informatik (GI), Arbeitskreis 5.10.3, Augsburg, 2002. http://www.fachkomponenten.de, Abruf am 20.4.2002.

[UDDI2000] *UDDI Organization (Hrsg.):* UDDI Technical White Paper. UDDI Standards Organization, September 2000. http://www.uddi.org, Abruf am 1.3.2002.