Konzeption eines Repositories zur Unterstützung der Wiederverwendung von Software-Komponenten

Oliver Höß[+], Anette Weisbecker[+]

[+] *Fraunhofer-Institut für Arbeitswirtschaft und Organisation (IAO) und Institut für Arbeitswissenschaft und Technologiemanagement (IAT) der Universität Stuttgart, Competence Center Software-Management, Nobelstr. 12, 70569 Stuttgart, Deutschland, Tel.: +49 (711) 970 - 2409 bzw. - 2400, Fax: +49 (711) 970 - 2401, E-Mail:{Oliver.Hoess|Anette.Weisbecker}@iao.fhg.de, URL: http://www.sw-management.iao.fhg.de*

Zusammenfassung. Der Ansatz der komponentenbasierten Softwareentwicklung kann nur erfolgreich umgesetzt werden, wenn ein geeignet großer Vorrat an wiederverwendbaren Komponenten existiert und diese auch benutzt werden können. In der Praxis ist dies oft nicht der Fall. Der vorliegende Beitrag zeigt in einem Überblick Hemmnisse auf, die die Wiederverwendung von Software-Komponenten be- oder verhindern. Ein Komponenten-Repository, d.h. ein Werkzeug, mit dem Komponenten innerhalb eines Unternehmens gespeichert, verwaltet und wiedergefunden werden können, stellt einen wesentlichen Beitrag zur Lösung eines Teils der Problematik dar. Daher werden die unterschiedlichen fachlichen und technischen Anforderungen an ein derartiges Repository sowie ein möglicher Realisierungsansatz dargestellt.

Schlüsselworte: Wiederverwendung; Komponenten-Repository; Komponentenbasierte Softwareentwicklung; Wissensmanagement in der Softwareentwicklung

1 Einleitung

Sowohl in der Wissenschaft als auch in der Praxis hat sich die Erkenntnis durchgesetzt, dass die komponentenbasierte Softwareentwicklung das derzeit geeignete Paradigma ist, die gestiegenen Anforderungen an moderne Software hinsichtlich

- Entwicklungsdauer,
- Entwicklungskosten und
- Software-Qualität

erfüllen zu können (siehe u.a. auch [FWH+2000] oder [HöWe2001]).

Die Wiederverwendung von Software-Komponenten ist dabei einer der wichtigsten Aspekte bei diesem Ansatz [Same1997]. Nur wenn Komponenten entsprechend häufig wiederverwendet werden, lohnt sich der zusätzliche Aufwand, der benötigt wird, um eine Komponente wiederverwendbar zu machen. Dieser Zusatz-Aufwand kann bis auf das 3-fache des normalerweise üblichen Aufwands steigen [JaGJ1997]. Eine genaue wirtschaftliche Berechnung ist hierbei schwierig, da eine Vielzahl von unterschiedlichen, teilweise nicht genau quantifizierbaren Faktoren eine Rolle spielt.

Die Umsetzung der Wiederverwendung und damit des Komponenten-Ansatzes wird derzeit

durch mehrere Faktoren begünstigt. Zum einen wurden standardisierte Komponenten-Technologien entwickelt, zum anderen hat die Verbreitung des Internets die Entstehung von Komponenten-Märkten ermöglicht [Behl2000].

Derzeit stehen eine Reihe von Komponenten-Technologien und Architekturmodellen zur Verfügung [WAD+2000]. Die derzeit wichtigsten sind Sun's Java 2 Enterprise Edition (J2EE) mit den Komponenten-Technologien JavaBeans und Enterprise JavaBeans, Microsoft's .NET mit den Komponenten-Technologien ActiveX und COM+ sowie der CORBA 3.0-Standard mit dem CORBA Component Model[1]. Durch diese Komponenten-Technologien wird die Wiederverwendung zumindest auf technischer Ebene vereinfacht.

Im Umfeld der Komponenten-Märkte[2] fällt auf, dass dort vorwiegend technische, domänenunabhängig einsetzbare Komponenten angeboten werden. Fachliche, domänenspezifische Komponenten sind dort selten zu finden. Um jedoch eine Steigerung der Effizienz in der Entwicklung von betrieblichen Anwendungssystemen zu erreichen, ist ein ausreichendes Angebot an wiederverwendbaren Fachkomponenten von ausschlaggebender Bedeutung [WeHS2001].

Insgesamt ist trotz dieser begünstigenden Faktoren die Wiederverwendung von Software-Komponenten in der betrieblichen Praxis von Unternehmen im industriellen Bereich und im Dienstleistungssektor bei weitem noch nicht optimal umgesetzt. Dies wird auch durch aktuelle Studien belegt (z.B. [DiEs2001] und [GfK+2000]).

Das Fraunhofer IAO hat in einer Reihe von Forschungsprojekten mit Industriebeteiligung (z.B. PROMPT [WeGr1998]) sowie in Projekten im Auftrag von Software-Herstellern und software-produzierenden Einheiten in Unternehmen ähnliche Erfahrungen gemacht. Wenn man den Durchschnitt aller Unternehmen betrachtet, wird nach dem in Bild 1 dargestellten Klassifikationsschema von [JaGJ1997] nach diesen Erfahrungen die Stufe 4 nicht erreicht.

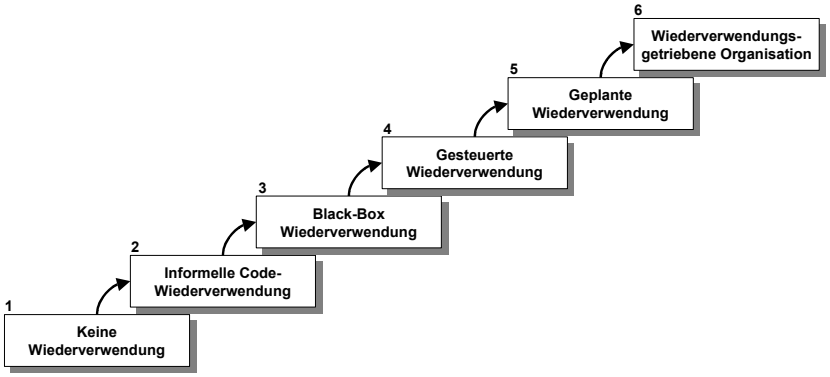

Bild 1: Reifestufen der Wiederverwendung (nach [JaGJ1997])

Dieser Beitrag zeigt nun in Abschnitt 2 eine Reihe von Hemmnissen auf, die die Wiederverwendung von Software-Komponenten be- oder verhindern. Ein wesentlicher Bestandteil einer Lösung zur Überwindung dieser Hemmnisse stellt dabei die Einführung und Nutzung eines

[1] Derzeit steht noch keine Implementierung des CORBA Component Models als Produkt zur Verfügung.

[2] Beispielsweise ComponentSource (www.componentsource.com) oder Flashline (www.flashline.com).

Komponenten-Repositories dar, das die Speicherung, Verwaltung und das Wiederfinden von gespeicherten Komponenten unterstützt. Basierend auf den Hauptprozessen der komponentenbasierten Softwareentwicklung (Abschnitt 3) werden in Abschnitt 4 Anforderungen an ein Komponenten-Repository formuliert sowie entsprechende Dienste definiert. Für die Umsetzung dieser Anforderungen und Dienste wird in Abschnitt 5 eine mögliche Systemarchitektur dargestellt. Der Beitrag wird dann mit einem Ausblick auf zukünftige Entwicklungen abgeschlossen.

2 Hemmnisse der Wiederverwendung von Software-Komponenten

Die Hemmnisse, die bei der Wiederverwendung von Software-Komponenten auftreten, sind von vielschichtiger Natur. In Bild 2 ist eine Übersicht der unterschiedlichen Kategorien dargestellt, die anschließend kurz erläutert werden. Dadurch soll die Vielfalt der Hemmnisse deutlich werden. Eine ausführlichere Darstellung ist in [WeHS2001] oder in [Reif1997] enthalten.

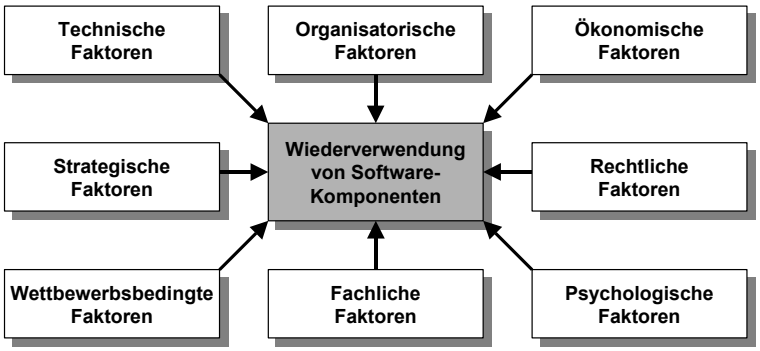

Bild 2: Hemmnisse bei der Wiederverwendung von Software-Komponenten

2.1 Technische Faktoren

Durch die in der Einleitung beschriebenen standardisierten Komponenten-Technologien (siehe auch [WAD+2000]) können viele Probleme auf technischer Ebene gelöst werden. Auch für den Fall, dass Komponenten unterschiedlicher Technologien miteinander kombiniert werden müssen, können durch Anwendung von „Wrapping"-Techniken Lösungen herbeigeführt werden.

Ebenfalls in den technischen Bereich fällt jedoch die meist nicht vorhandene Möglichkeit zur unternehmensweiten Speicherung, Verwaltung und Recherche von Komponenten in einem Komponenten-Repository. Dies wird auch in der Studie [DiEs2001] deutlich, bei der Punkte wie z.B.

- unzureichende Werkzeugunterstützung,
- eine unzureichende Verwaltung von Komponenten sowie
- Probleme beim Finden von Komponenten

als Hauptkritikpunkte angegeben werden.

2.2 Organisatorische Faktoren

Der wesentliche Faktor im Bereich der Organisation ist das häufige Fehlen entsprechender Stellen, die für die Wiederverwendung von Software-Komponenten verantwortlich sind. Die Hauptaufgabe eines derartigen „Reuse-Teams“ [WeGr1998] ist dabei sicherlich die Installation sowie die Pflege und Wartung des in Abschnitt 2.1 beschriebenen Komponenten-Repositories. Dies schließt auch die Evaluation und die Integration von extern erworbenen Komponenten mit ein.

Um auch die Integration in die Projekt-Struktur des jeweiligen Unternehmens sicherzustellen, sollten die Mitglieder dieses Teams auch in einer beratenden Funktion in den Projekten mitarbeiten um dort insbesondere in den Bereichen Wiederverwendung und Software-Architektur hohe Standards zu setzen. Außerdem können auf diese Weise in den Projekten wiederverwendbare Komponenten identifiziert werden und zur Wiederverwendung und Einstellung in das Repository vorbereitet werden.

2.3 Ökonomische Faktoren

Da das Management eines Unternehmens i.d.R. nach betriebswirtschaftlichen Grundsätzen denkt und handelt, ist es von entscheidender Bedeutung, dass der Nutzen der Wiederverwendung anhand von konkreten Zahlen deutlich wird. Diese Ermittlung von Kennzahlen im Bereich der Software-Entwicklung wird in den meisten Unternehmen nicht durchgeführt, was dazu führt, dass das Management nicht bereit ist, dauerhaft Mittel für den Betrieb eines Repositories sowie der entsprechenden Stellen für ein „Reuse-Team“ zur Verfügung zu stellen [Reif1997]. Daher sollte eine Ermittlung der entsprechenden Kennzahlen ebenfalls zu den Aufgaben dieses Teams gehören.

2.4 Rechtliche Faktoren

Die teilweise unklare Rechtslage, beispielsweise bei der Fehlfunktion eines Produkts durch die Fehlfunktion einer extern eingekauften Komponente, führt teilweise dazu, dass bereits bestehende Komponenten nochmals implementiert werden (siehe [SoSo1998]), um den rechtlichen Risiken in diesem Bereich aus dem Wege zu gehen.

Auch die in [Same1997] erwähnte Möglichkeit, Komponenten zur Verwendung auf eigenes Risiko freizugeben, mag in manchen Bereichen eine geeignete Lösung sein, ist jedoch nicht überall anwendbar.

2.5 Psychologische Faktoren

Auch psychologische Faktoren stellen einen häufig unterschätzten Hinderungsgrund bei der Wiederverwendung von Software-Komponenten dar. Häufig werden Komponenten aus Konkurrenzdenken den Kollegen nicht zur Verfügung gestellt, um zu verhindern, dass diese Erfolge auf Basis der eigenen Arbeiten erreichen.

In vielen Fällen ist es auch der Forscher-Drang, der Entwickler dazu verleitet, Funktionalitäten zu implementieren, ohne ausreichend zu untersuchen, ob diese evtl. bereits an anderer Stelle entwickelt wurden [Reif1997].

2.6 Fachliche Faktoren

Der in der Einleitung beschriebene Mangel an wiederverwendbaren fachlichen Komponenten ist nicht zuletzt darauf zurückzuführen, dass es äußerst schwierig ist, fachliche Funktionalitäten derart zu verallgemeinern, dass sie als „Fachkomponenten“ wiederverwendbar werden.

Erfolgversprechend sind in diesem Bereich unternehmensübergreifende, branchenspezifische Initiativen, die vereinheitlichte fachliche Objektmodelle entwickeln und diese für Software-Hersteller und –Anwender in der Branche zur Verfügung stellen. Ein Vorreiter auf diesem Gebiet ist der Gesamtverband der deutschen Versicherungswirtschaft, der ein vereinheitlichtes Fachkonzept auf UML-Basis für die gesamte Versicherungsbranche definiert hat [GdV1999].

2.7 Wettbewerbsbedingte Faktoren

In vielen modernen Unternehmen, wie z.B. bei Online-Finanzdienstleistern, stellt die verwendete Software das wesentliche Produktionsmittel dar. Derartige Unternehmen sind daher aus Gründen des Wettbewerbs in vielen Fällen nicht dazu bereit, standardisierte Komponenten einzusetzen, da sie ansonsten Alleinstellungsmerkmale aufgeben müssen, die sie von ihren Konkurrenten unterscheiden. Für die Überwindung dieses Hemmnisses ist daher eine sehr hohe Flexibilität und Konfigurierbarkeit der Komponenten notwendig.

2.8 Strategische Faktoren

Neben wettbewerbsbedingten Faktoren spielen in vielen Fällen auch strategische Überlegungen eine Rolle. Beispielsweise wird sich ein Unternehmen bei wichtigen Komponenten nur sehr ungern auf Komponenten eines kleinen Herstellers verlassen, da dessen Zukunft und somit die Pflege und Weiterentwicklung dieser Komponenten evtl. nicht gesichert ist. Dies wurde insbesondere in den letzten Jahren deutlich, als viele Startup-Unternehmen, darunter auch Komponenten-Hersteller, ihre Geschäftstätigkeit einstellen mussten.

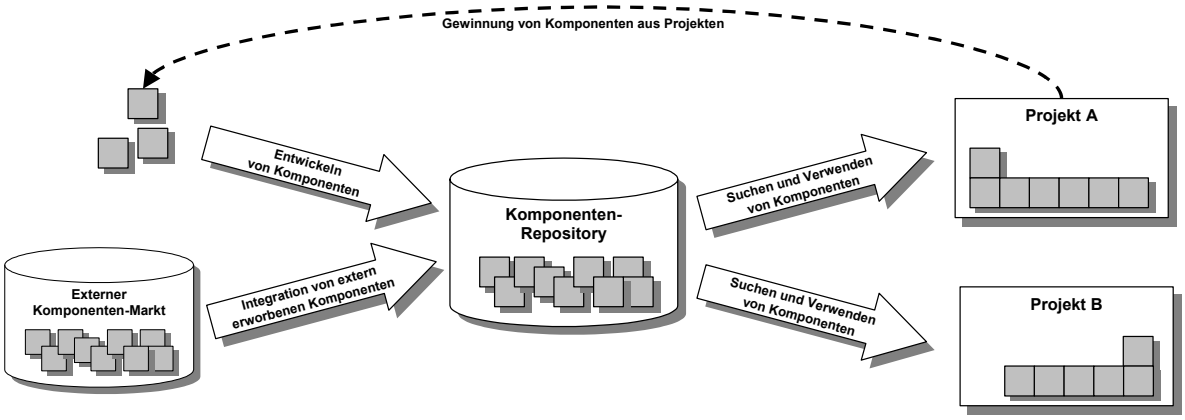

Bild 3: Hauptprozesse bei der komponentenbasierten Softwareentwicklung

3 Durch das Repository zu unterstützende Prozesse

Innerhalb der komponentenbasierten Softwareentwicklung existieren eine Reihe von Prozes-

sen, die durch ein Komponenten-Repository (vgl. Abschnitt 2.1) geeignet unterstützt werden müssen, um einen Nutzen zu erzielen. Die wichtigsten Teilprozesse, die auch eng mit den im Projekt REBOOT definierten „development *for* reuse“ und „development *with* reuse“ korrespondieren [Karl1995], sowie die Beteiligung des Reuse-Teams (vgl. Abschnitt 2.2) sind in Bild 3 dargestellt und werden in den folgenden Abschnitten näher erläutert (vgl. auch [Bull2000]). Die dargestellten Prozesse sind aus Gründen der Übersichtlichkeit stark vereinfacht und sollen nur die wesentlichen Aspekte verdeutlichen. Die Notation, die bei den graphischen Prozessdarstellungen verwendet wird, ist dabei an die Methode der ereignisgesteuerten Prozessketten angelehnt (siehe auch [KeNS1992]).

3.1 Entwicklung von neuen Komponenten bzw. Versionen

Neue Komponenten entstehen im Unternehmen i.d.R. auf zwei unterschiedliche Arten. Der einfache Fall ist, dass sie durch das Reuse-Team in Hinblick auf die spätere Wiederverwendung entwickelt werden. Die auf diese Art und Weise entstehenden Komponenten werden daher meistens die Anforderungen an eine spätere Wiederverwendung erfüllen.

Der weitaus häufigere Fall ist jedoch der, dass Software innerhalb von aktuellen Projekten entsteht und dabei von den Projektmitarbeiten oder durch das in beratender Funktion beteiligte Reuse-Team erkannt wird, dass sich die Software zur Wiederverwendung eignet. In diesem Fall ist jedoch i.d.R. ein weiterer Schritt für die Verallgemeinerung durch das Reuse-Team der meist an die konkreten Projektbelange angepasste Software notwendig, bevor man sie wirklich „Komponente“ nennen kann.

In beiden Fällen wird die Komponente anschließend durch das Reuse-Team in das Repository eingestellt. Dieser Teilprozess wird in Abschnitt 3.2 beschrieben.

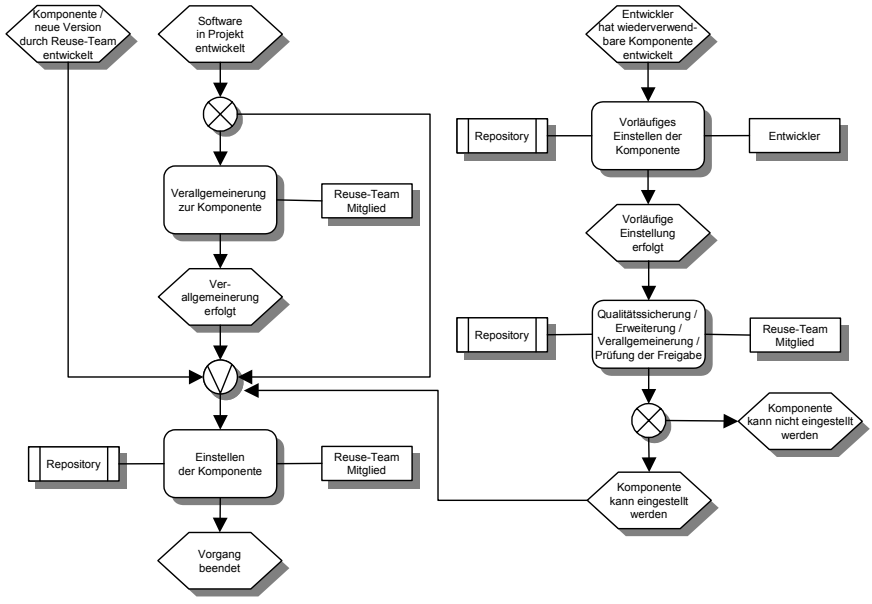

Bild 4: Entwicklung von neuen Komponenten bzw. Versionen

Es soll den Entwicklern auch selbst die Gelegenheit gegeben werden, Komponenten in das Repository einzustellen. Dies soll auf eine möglichst einfache Art und Weise möglich sein. Es

muss jedoch danach noch eine Qualitätssicherung mit einer Prüfung der Freigabe durch das Reuse-Team erfolgen. Nur wenn diese Freigabe erfolgt ist, kann die Komponente endgültig klassifiziert und eingestellt werden.

3.2 Einstellen von Komponenten bzw. Versionen

Vor der eigentlichen Einstellung einer Komponente in das Repository muss die dazugehörige Dokumentation bzw. Spezifikation ergänzt werden. Dabei bietet es sich an, sich an eine standardisierte Dokumentations- bzw. Spezifikationsmethodik anzulehnen. Ein Beispiel dafür ist der Vorschlag des GI-Arbeitskreises „Komponentenorientierte betriebliche Anwendungssysteme“ (siehe [ABC+2002]).

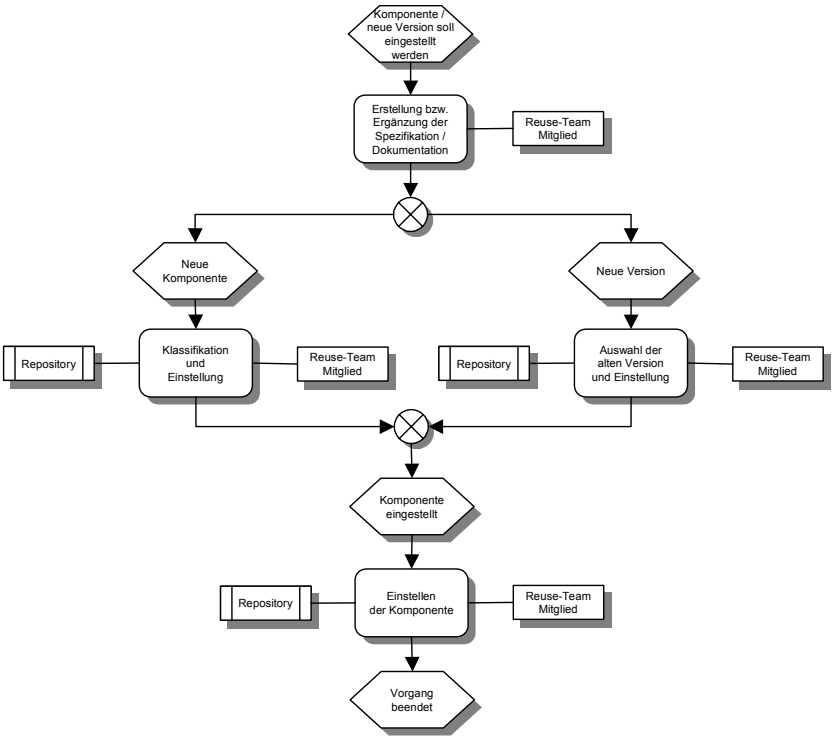

Bild 5: Einstellen von Komponenten bzw. Versionen

Wenn es sich um eine neue Version einer schon im Repository vorhandenen Komponente handelt, muss an dieser Stelle die vorherige Version der Komponente gesucht werden und die neue Version eingestellt werden.

Handelt es sich um eine neue, d.h. bisher nicht im Repository vorhandene Komponente, muss sie vor der Einstellung entsprechend klassifiziert werden, um später auch wieder auffindbar zu sein. In der Wissenschaft wurden schon eine Reihe von Ansätzen für die Klassifikation von Software-Komponenten entwickelt, wie z.B. die Klassifikation mit Facetten [Börs1995]. Derzeit besteht jedoch noch keine allgemein akzeptierte Klassifikationsmethodik für Komponenten. Ein Ansatz könnte sein, die Klassifikationsmethodik von bereits bestehenden Software-Märkten im Internet zu verwenden. Beispielsweise hat der Komponentenmarkt Flashline seine Klassifikationsschematik in Form einer XML-DTD veröffentlicht (siehe [Flas2002]).

Diese ist jedoch noch nicht sehr ausgereift und es besteht an dieser Stelle im Bereich der Definition eines vereinheitlichten Komponentenklassifikationsschemas definitiv noch Forschungs- bzw. Standardisierungsbedarf.

Ist die Komponente eingestellt, versendet das Repository in beiden Fällen automatisch Benachrichtigungen an die Anwender, die sich dafür registriert haben, bei neuen Versionen einer Komponente bzw. neuen Komponenten benachrichtigt zu werden (vgl. Abschnitt 3.4).

3.3 Integration von extern entwickelten Komponenten

Eine der Aufgaben des Reuse-Teams ist das regelmäßige Durchsuchen von externen Komponenten-Märkten und –Anbietern. Werden dort Komponenten gefunden, die für die Wiederverwendung geeignet sind, sollten diese nach einer Evaluationsphase in das Repository eingepflegt werden, d.h. es wird der COTS-Ansatz (Commercial Off-The-Shelf) unterstützt [Voas1998].

In vielen Fällen müssen jedoch vorher noch Anpassungen an die Spezifika des Unternehmens durchgeführt werden. Dabei ist darauf zu achten, dass diese Anpassungen bei Erscheinen einer neuen Version nicht verloren gehen, sondern übernommen werden können.

Der Optimalfall ist daher, dass keine Anpassungen notwendig sind. Dies ist jedoch bei komplexeren Komponenten eher selten der Fall.

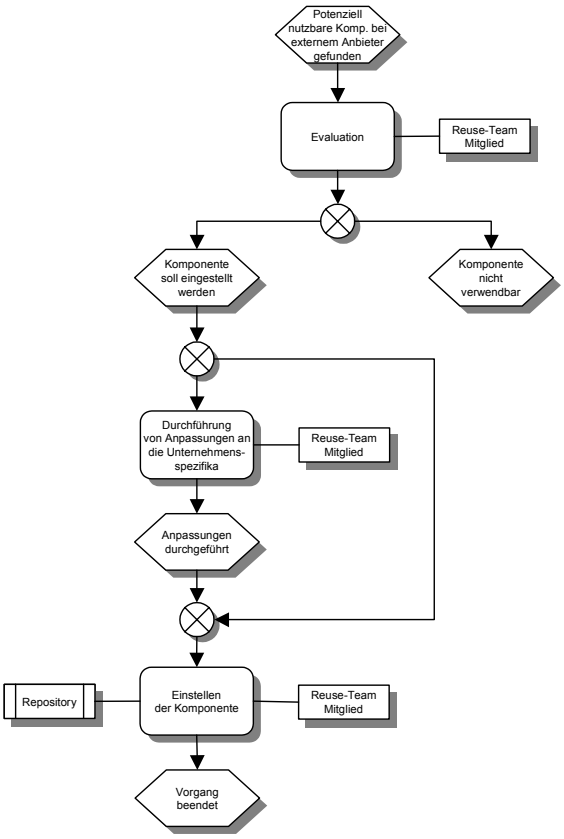

Bild 6: Integration von extern entwickelten Komponenten

3.4 Suchen und Verwenden von Komponenten

Der Verwendung einer Komponente können 2 unterschiedliche Ereignisse vorangehen:

- Eine Suche aufgrund des Bedarfs einer konkreten Komponente.
- Eine Benachrichtigung des Systems, dass eine neue Version einer Komponente existiert bzw. dass eine neue Komponente eingestellt wurde.

Im Falle einer Suche muss der Anwender die vorhandenen Komponenten auf Basis des verwendeten Klassifikationsschemas durchsuchen. Dabei sind unterschiedliche Suchmöglichkeiten, wie z.B. eine schlagwortbasierte Suche oder eine Baumsuche, denkbar (vgl. 4.1.2).

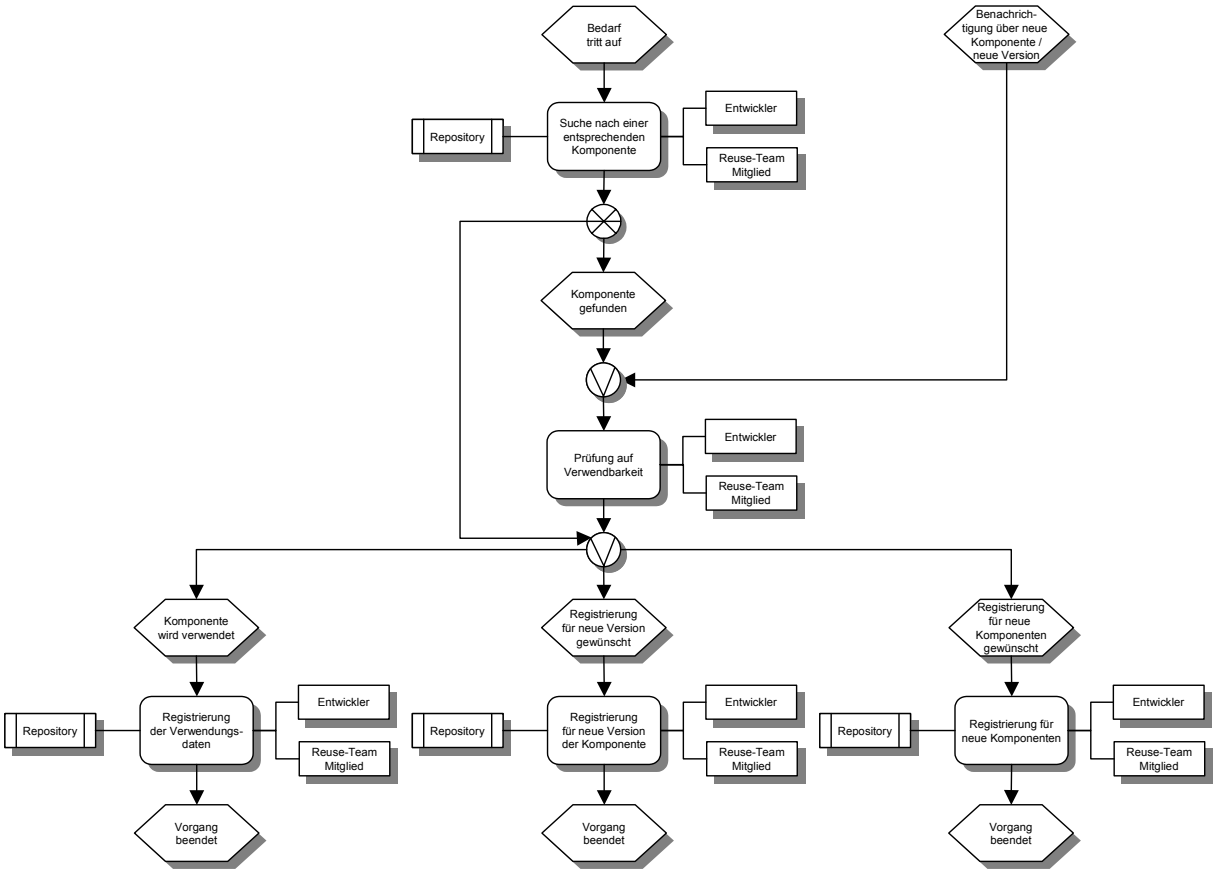

Bild 7: Entwicklung von neuen Komponenten bzw. Versionen

Wenn eine Komponente aufgrund der Klassifikationskriterien den Anforderungen entspricht, kann deren Dokumentation bzw. Spezifikation mit den Detailanforderungen abgeglichen werden. Dies wird umso einfacher, je mehr die Art der Dokumentation bzw. Spezifikation standardisiert ist, z.B. nach der Methode, die in [ABC+2002] beschrieben wird.

Wenn der Suchende eine Komponente findet, die seinen Anforderungen entspricht und er sich dafür entscheidet, sie zu verwenden, sollte das System automatisch die Daten der Verwendung (Adressdaten, Projektname, ...) erfassen, um eine Übersicht über die Verwendungen der Komponente zu erhalten und bei neuen Versionen eine Benachrichtigung generieren zu kön-

nen.

In jedem Fall, d.h. insbesondere auch im Fall einer nicht erfolgreichen Suche, kann sich der Benutzer für eine Benachrichtigung bei neuen Komponenten bzw. bei neuen Versionen von Komponenten registrieren. Dies ist nicht nur bei einzelnen Komponenten, sondern auch für ganze Gruppen von Komponenten, z.B. auf Grund des Klassifikationsschemas, möglich.

Bei der Wiederverwendung von bestehenden Komponenten ist insbesondere auch die Frage der Integration der Wiederverwendung in die Entwicklungsprozesse von großer Bedeutung, dabei insbesondere die Fragestellung, ob das Anwendungsdesign vor der Auswahl der geeigneter Komponenten erfolgt, oder ob das Anwendungsdesign abhängig von den vorhandenen Komponenten durchgeführt werden sollte (vgl. [PoBu2001]).

4 Benötigte Dienste eines Repositories

Um die in Abschnitt 3 beschriebenen Prozesse zu unterstützen, muss das Repository eine Reihe von Diensten bereitstellen. Die Dienste ergeben sich dabei hauptsächlich aus den funktionalen Anforderungen und können in Basis-Dienste und Mehrwert-Dienste unterteilt werden.

Basis-Dienste sind dabei Dienste, die Grundfunktionalitäten realisieren, die zur Funktion des Repositories unbedingt notwendig sind. Mehrwert-Dienste sind Dienste, die zwar nicht unbedingt notwendig sind, aber den Anwendern oder den Betreibern eines Repositories einen Mehrwert bringen.

Außerdem bestehen noch eine Reihe von technischen Anforderungen, die für den erfolgreichen Einsatz eines Repositories ausschlaggebend sind (siehe Bild 8).

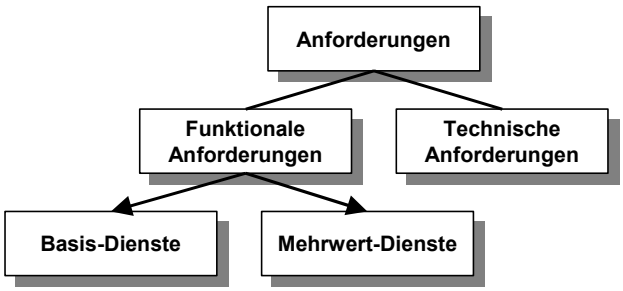

Bild 8: Anforderungen und Dienste

4.1 Basis-Dienste

Die Basis-Dienste beschränken sich im wesentlichen auf die Realisierung der Funktionalitäten für das Einstellen und Suchen von Komponenten sowie die Realisierung eines Benutzer- bzw. Rollenkonzepts.

4.1.1 Einstellen von Komponenten

Das Repository muss den Anwendern die Funktionalität anbieten, Komponenten jeglicher Art, d.h. beliebige Dateien, einzustellen. Im Falle von Komponenten, die z.B. aus lizenzrechtlichen Gründen nicht im Repository gespeichert werden können, muss ein Verweis auf die

Bezugsquelle eintragbar sein.

Die einzustellenden Komponenten müssen dabei mit entsprechenden Meta-Daten, wie z.B. Name, Kurzbeschreibung oder Autor versehen werden können. Ein besonders wichtiges Meta-Datum ist hierbei die Dokumentation bzw. Spezifikation der Komponente. Je nach Güte des Repositories kann diese nur als „Black Box“ oder in einer strukturierten und später auch auswertbaren Form abgelegt werden. In jedem Fall sollte eine standardisierte Spezifikations- und Dokumentationsmethodik verwandt werden, die obligatorisch bei allen abgelegten Komponenten eingehalten werden muss.

Desweiteren muss die Komponente in ein Klassifikationsschema eingeordnet werden. Hierbei sind auch domänenabhängige Klassifikationskriterien vorzusehen. Desweiteren können Komponenten nach unterschiedlichen Kriterien klassifiziert werden.

Um die Evolution der Komponenten zu unterstützen, muss eine Versionierungsfunktionalität vorhanden sein. Dabei können durchaus unterschiedliche Versionsnummern, z.B. interne und externe Versionsnummern, verwendet werden.

Um eine Qualitätssicherung zu unterstützen und um einen Wildwuchs zu vermeiden, sollten neu eingestellte Komponenten durch eine Clearing-Stelle, die durch das Reuse-Team gebildet wird, einer Prüfung unterzogen werden, bevor sie für die allgemeine Verwendung freigegeben werden.

4.1.2 Suchfunktionalitäten

Grundsätzlich sollten in einem Repository mindestens zwei Suchvarianten möglich sein:

- Eine schlagwortbasierte Suche.
- Eine baumbasierte Suche.

Die schlagwortbasierte Suche erlaubt nach Eingabe bzw. Auswahl von gewissen Suchkriterien eine Suche auf den Metadaten der Komponente. Die eingegebenen Kriterien werden durch die bekannten logischen Operatoren „UND“ und „ODER“ miteinander verknüpft. Bei der Formulierung bzw. Definition der Suchanfragen können beispielsweise auch Features wie z.B. eine Thesaurusunterstützung integriert werden (siehe auch [GCO+2000]). Die Suche wird solange verfeinert, bis die gewünschte Komponente gefunden ist bzw. die Suche aufgegeben wird.

Die baumbasierte Suche ermöglicht es dem Anwender, durch den Baum, der durch die Klassifikationsmerkmale aufgespannt wird, zu navigieren („browsen“) und sich auf jeder Ebene jeweils alle vorhandenen Komponenten anzeigen zu lassen. Dabei können dynamisch unterschiedliche Bäume aufgebaut werden, die je nach Art der Navigation unterschiedlich sind.

Die baumbasierte Suche ist dann besser geeignet, wenn der Anwender noch keine konkrete Vorstellung besitzt, durch die Eingabe welcher Schlagworte er die gesuchte Komponente finden kann, sondern er sich eher eine Übersicht über den Vorrat an Komponenten verschaffen möchte.

4.1.3 Unterstützung eines Benutzer- / Rollenkonzepts

Für den erfolgreichen Einsatz des Repositories ist es von großer Bedeutung, dass unterschiedliche Nutzergruppen unterstützt werden, die jeweils auch unterschiedliche Rechte besitzen.

Ein einfaches Benutzerkonzept kann mit folgenden vier Rollen realisiert werden:

- Gäste: Können nur lesend auf das Repository zugreifen, aber keine Komponenten einstellen.
- Angemeldete Nutzer: Können zusätzlich auch Komponenten einstellen.
- Reuse-Administratoren: Sind für die Pflege des Repositories sowie für die Freigabe von durch andere Nutzer eingestellten Komponenten zuständig.
- System-Administratoren: Besitzen alle Rechte und verwalten die Benutzer. Sie sind für den unterbrechungsfreien technischen Betrieb des Repositories zuständig.

Das verwendete Benutzer- / Rollenkonzept kann selbstverständlich noch beliebig detailliert werden. Es ist jedoch zu bedenken, dass auch ein entsprechender Pflegeaufwand einkalkuliert werden muss.

4.2 Mehrwertdienste

Zusätzlich zu den Basis-Diensten kann bzw. sollte ein Repository noch sog. Mehrwert-Dienste besitzen, die für den Anwender oder den Betreiber einen Mehrwert bieten. Im folgenden werden einige mögliche Dienste vorgestellt, der Kreativität sind an dieser Stelle jedoch kaum Grenzen gesetzt.

4.2.1 Benachrichtigungsmechanismen

Das Repository sollte eine Reihe von Benachrichtigungsdiensten anbieten. Alle Anwender, die eine bestimmte Komponente in ihrem Projekt nutzen, müssen sich im System registrieren, so dass sie automatisch benachrichtigt werden können, wenn eine neue Version dieser Komponente verfügbar wird (vgl. Abschnitt 3.4). Durch diese Verfahrensweise haben das Reuse-Team sowie andere Anwender auch einen Überblick, wer welche Komponenten in welchen Projekten nutzt.

Auch Benutzer, die die Komponente nicht nutzen, können sich für Benachrichtigungen bei neuen Versionen einer Komponente registrieren, da die neue Version evtl. den Anforderungen entspricht und dann genutzt werden kann.

Diese Registrierungsmöglichkeit kann auch auf ganze Kategorien von Komponenten ausgedehnt werden, so dass man sich z.B. dafür registrieren kann, benachrichtigt zu werden, wenn eine neue Komponente im Bereich „GUI-Komponenten“ eingestellt wird.

4.2.2 Einsatznachweise, Erfahrungsberichte und Bewertungen

Die Dokumentation der Wiederverwendung von aus dem Repository bezogenen Komponenten sollte obligatorisch sein, da diese Einsatznachweise eine wichtige Grundlage für die Berechnung von Kennzahlen (4.2.3) und die Implementierung von Anreizsystemen (4.2.4) darstellt.

Es sollte zusätzlich die Möglichkeit bestehen, dass die Anwender zu jeder Komponente Erfahrungsberichte und Bewertungen abgeben können. Auf diese Weise können Anwender von den Erfahrungen anderer Anwender profitieren.

Wichtig ist in diesem Zusammenhang ebenfalls eine Clearingstellen-Funktion der Reuse-Administratoren. Diese müssen neue Beiträge inhaltlich überprüfen und anschließend freige-

ben um eine hohe Qualität der Beiträge zu gewährleisten.

4.2.3 Unterstützung der Ermittlung und Auswertung von Kennzahlen

Um den ökonomischen Nutzen der Wiederverwendung von Software-Komponenten zu dokumentieren (oder im negativen Fall zu widerlegen) sollte das Repository eine Unterstützung für die Ermittlung und Auswertung von Kennzahlen, wie z.B. die Häufigkeit der Wiederverwendung einer Komponente, bereitstellen.

Somit soll das Reuse-Team befähigt werden, dem Management gegenüber Aussagen zum Kosten / Nutzen –Verhältnis von einzelnen Komponenten, zur Pflege und zum Betrieb des Repositories und somit zur ökonomischen Berechtigung des gesamten Reuse-Teams zu machen.

Eine besondere Bedeutung kommt dabei den in 4.2.2 beschriebenen Einsatznachweisen zu, da die Häufigkeit der Wiederverwendung einer Komponente für eine Reihe von Kennzahlen, z.B. aus dem Kosten / Nutzen-Bereich, eine wichtige Ausgangsgröße darstellt.

4.2.4 Implementierung von Anreizsystemen

Um die in Abschnitt 2.5 beschriebenen psychologischen Hemmnisse zu überwinden, ist es sinnvoll, Anreizsysteme in das Repository zu integrieren. Die Verwendung von Anreizsystemen ist ein häufig genutztes Mittel im Bereich des Wissensmanagements und kann daher auch für das Wissensmanagement im Bereich der Softwareentwicklung angewandt werden.

Der erste Ansatz dabei ist, den Einsteller einer Komponente zu belohnen. Diese Belohnung kann materieller, finanzieller oder auch ideeller Art sein. Der Nachteil bei dieser Vorgehensweise ist jedoch, dass die Belohnung auch dann erfolgt, wenn eine Wiederverwendung der eingestellten Komponente nicht erfolgt.

Daher ist ein anderer Ansatz der, dass die Belohnung erst dann wirksam wird, wenn die Komponente durch einen Dritten wiederverwendet wurde. Es besteht zusätzlich auch noch die Möglichkeit, statt dem Einsteller denjenigen zu belohnen, der eine Komponente wiederverwendet, um nicht nur die Einstellung sondern auch die Wiederverwendung und insbesondere auch die Dokumentation des Wiederverwendungseinsatzes zu belohnen. Auch in diesem Fall ist der Erfassung der Einsatznachweise eine wichtige Grundvoraussetzung (vgl. 4.2.2).

4.2.5 Portalfunktionalität

Das Repository sollte ebenfalls eine gewisse Portalfunktionalität anbieten, über die weitergehende Informationen unterschiedlichster Art abgerufen werden können. Beispiele hierfür sind:

- Eine strukturierte Linksammlung externer Komponentenmärkte und –anbieter.
- Ein Überblick über aktuelle Entwicklungsprojekte im Unternehmen.
- Eine Darstellung des Reuse-Teams (Personen und Aufgaben).
- Eine Übersicht über neu eingestellte Komponenten.

Dabei sollte eine Personalisierung der dargestellten Inhalte möglich sein, da dies ja auch eines der definierenden Merkmale eines Portals darstellt.

4.2.6 „Schwarzes Brett“

Da es in umfangreichen Datenbeständen nicht unbedingt garantiert ist, dass ein Anwender die gesuchte Komponente auch wirklich findet, ist es sinnvoll, eine Funktionalität anzubieten, die der eines „Schwarzen Bretts“ entspricht. Anwender können dort Suchanzeigen für Komponenten mit einer bestimmten Funktionalität aufgeben oder andere Fragen stellen.

Die Fragen können dann durch das Reuse-Team oder andere erfahrene Anwender beantwortet werden. Auf diese Weise können auch Anregungen für extern zu beschaffende Komponenten an das Reuse-Team herangetragen werden (vgl. Abschnitt 3.3).

4.3 Technische Anforderungen

Neben den in den Abschnitten 4.1 und 4.2 beschriebenen Basis- und Mehrwertdiensten existieren auch noch eine Reihe von technischen Anforderungen, die im folgenden beschrieben werden.

4.3.1 Verfügbarkeit über Intranet / Internet - Technologien

Da eine möglichst breite Verfügbarkeit gewährleistet werden soll, ist es notwendig, dass das Repository eine HTML-basierte Oberfläche besitzt. Somit können die Nutzer über ihre Standard-Webbrowser auf die Funktionalitäten des Repositories zugreifen, ohne zusätzliche Software installieren zu müssen.

Für die Administratoren ist zusätzlich ein Administrations-Client notwendig, da sich einige Funktionalitäten einer guten Benutzungsoberfläche (wie z.B. dynamische Baumdarstellungen) nur sehr unbefriedigend lösen lassen, dies aber für Administrations-Tätigkeiten oft notwendig ist.

4.3.2 Integration in die Entwicklungsumgebung

Um den Zugriff auf das Repository möglichst einfach zu gestalten, ist es notwenig, eine Integration des Repositories in die jeweiligen Entwicklungsumgebungen der Entwickler zu gewährleisten. Aufgrund der Vielfalt der Entwicklungsumgebungen und der fehlenden Standards in diesem Bereich ist eine Integration jedoch um so schwerer, je enger sie sein soll. Eine lose Integration, d.h. über einen Link, ist jedoch in den meisten Fällen zu realisieren.

4.3.3 Qualität der Benutzungsoberfläche

Wie bei jedem Wissensmanagement-System ist auf eine qualitativ hochwertige Gestaltung der Benutzungsoberfläche großen Wert zu legen, da die Motivation der Einsteller, also der Knackpunkt eines jeden Wissensmanagement-Systems, i.d.R. stark von der Qualität der Benutzungsoberfläche abhängt. Insbesondere ist in diesem Zusammenhang auch auf kurze Antwortzeiten bei der Einstellung, aber auch bei der Suche, zu achten.

4.3.4 Verfügbarkeit der Funktionalität über Web-Services

Um die Funktionalitäten des Repositories auch für andere Systeme verfügbar zu machen und um insbesondere die in Abschnitt 4.3.2 beschriebene Integration in die Entwicklungsumgebungen zu realisieren, ist es sinnvoll, die Funktionalitäten des Repositories auch über sog.

Web-Services anzubieten. Diese Web-Services nutzen das SOAP Protokoll, das derzeit durch das Word Wide Web Consortium standardisiert wird [W3C2002], um unterschiedliche Anwendungen auf eine plattform-unabhängige Art und Weise miteinander zu integrieren.

Unter Verwendung dieser Technologie könnten auch mehrere Repositories über ein Meta-Repository miteinander kombiniert werden oder es könnten die Daten von externen Komponenten-Märkten integriert werden, wenn diese ebenfalls eine derartige Schnittstelle bereitstellen. Diese Technologien spielen als Basis-Technologien für die Weiterentwicklung des Internet zu einem „Semantic Web“ eine immer größere Rolle.

5 Konzeption einer System-Architektur

Um die in diesem Beitrag beschriebenen Anforderungen und Dienste umzusetzen, muss eine System-Architektur gewählt werden, die die Funktionalität über unterschiedliche Wege den unterschiedlichen Nutzergruppen zur Verfügung stellt. Ein derartiges Komponenten-Repository sollte unter Verwendung von derzeit verfügbaren Komponenten-Technologien, wie z.B. die J2EE- oder die .NET-Technologien realisiert werden. In Bild 9 ist eine mögliche System-Architektur dargestellt.

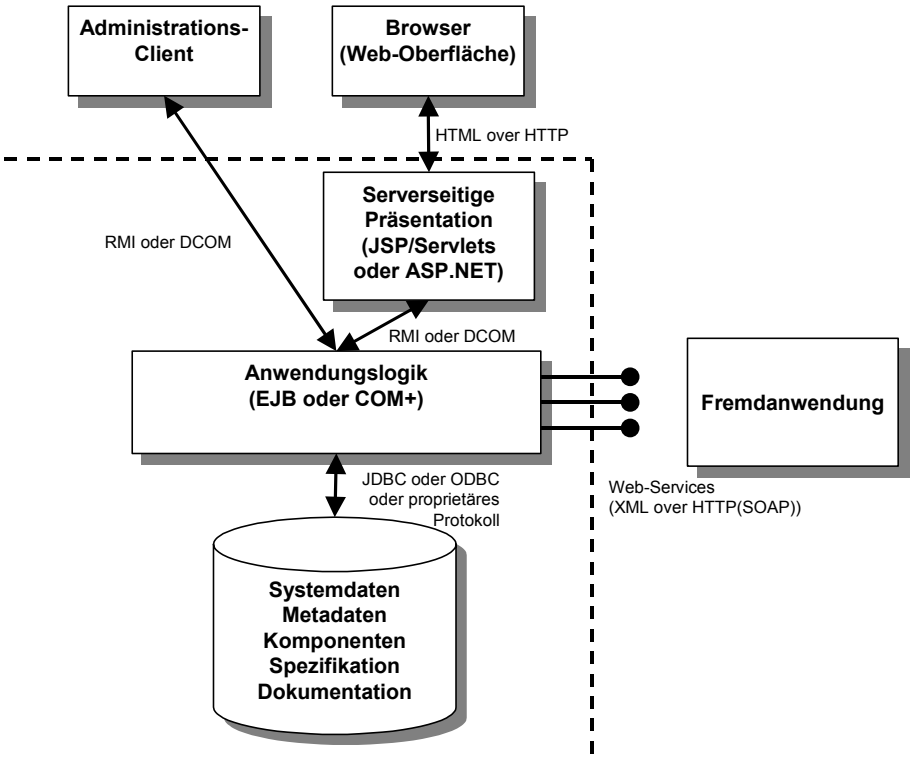

Bild 9: Mögliche System-Architektur eines Repositories

Für die Datenhaltung, d.h. für die Systemdaten, die Metadaten, die eigentlichen Komponenten sowie deren Spezifikation bzw. Dokumentation, können beispielsweise relationale Datenbank-Systeme eingesetzt werden. Falls die Dokumentation / Spezifikation XML-basiert ist, eignen sich in vielen Fällen auch XML-Datenbanken als Speichermedium.

Die Anwendungslogik sollte durch Komponenten der Technologien EJB oder COM+ realisiert werden. Die dort bereitgestellten Dienste können dann über drei unterschiedliche Kanäle zur Verfügung gestellt werden:

- Für die Endanwender werden die Daten durch eine serverseitige Präsentationsschicht in das HTML-Format aufbereitet, so dass sie überall verfügbar sind, wo ein entsprechender Browser vorhanden ist.
- Für Administratoren werden die Funktionalitäten über einen Administrations-Client zur Verfügung gestellt, da eine reine HTML-Oberfläche nicht den notwendigen Komfort für umfangreiche administrative Tätigkeiten bietet.
- Für die Integration in Fremdanwendungen, wie z.B. Entwicklungsumgebungen, werden die Dienste als Web-Services über das SOAP-Protokoll (Simple Object Access Protocol) zur Verfügung gestellt (siehe auch [W3C2002]).

Im Rahmen des durch das BMBF geförderten Projekts Adaptive READ wurde ein erster Prototyp eines Repositories in der beschriebenen Architektur erstellt. Dieser Prototyp wird derzeit insbesondere mit dem Fokus auf den Mehrwertdiensten weiterentwickelt.

6 Zusammenfassung und Ausblick

Im vorliegenden Beitrag wurden eine Reihe von Hemmnissen aufgezeigt, die die Wiederverwendung von Software-Komponenten behindern. Basierend auf den Hauptprozessen der komponentenbasierten Softwareentwicklung wurden die Anforderungen an ein Komponenten-Repository definiert, das einen Teil der angeführten Probleme lösen oder mildern kann. Kombiniert mit der Einführung eines Reuse-Teams, das für den Betrieb des Repositories verantwortlich ist, sowie von Anreiz- und Kennzahlensystemen können insbesondere Probleme im technischen, organisatorischen, psychologischen und ökonomischen Bereich adressiert werden.

Für den effektiven Einsatz eines derartigen Komponenten-Repositories ist es von enormer Bedeutung, einheitliche Klassifikations- und Spezifikationsmethoden für Software-Komponenten zu entwickeln. Die Klassifikationsmethoden sind dabei vor allem für das Einstellen und spätere Wiederfinden von Bedeutung, die Spezifikationsmethoden vor allem für die Evaluation und Anwendung von Komponenten.

Für einen Teil der angesprochenen Probleme, z.B. aus dem fachlichen, strategischen oder rechtlichen Bereich, kann ein Repository keine Lösung bieten. Teilweise sind diese Probleme in ihrem Grundsatz auch nur sehr schwer und auch in eher in einem anderen Kontext zu lösen.

Insgesamt kann zusammenfassend gesagt werden, dass der Bereich der Wiederverwendung von Software-Komponenten immer noch große Potenziale in sich verbirgt und noch enorme Anstrengungen in der Wissenschaft und in der anwendungsorientierten Forschung notwendig sind, um diese Potenziale nutzbar zu machen. Dies wird auch dadurch deutlich, dass im Bereich „Software Engineering“ des Arbeitsprogramms „IT 2006“ des BMBF (siehe [BMBF2002]) die Themen „Produktivitätserhöhung mittels Komponentenorientierung und Wiederverwendung“ sowie „Wissensmanagement in der Softwareentwicklung“ als Hauptthemen vertreten sind.

Literatur

[ABC+2002] *Ackermann, J.; Brinkop, F.; Conrad, S.; Fettke, P.; Frick, A.; Glistau, E.; Jaekel, H.; Klein, U.; Kotlar, O.; Loos, P.; Mrech, H.; Ortner, E.; Overhage, S.; Sahm, S.; Schmietendorf, A.; Teschke, T; Turowski, K.*: Vorschlag zur Vereinheitlichung der Spezifikation von Fachkomponenten. Memorandum des GI-Arbeitskreises 5.10.3: Komponentenorientierte betriebliche Anwendungssysteme, Version Februar 2002. http://wi2.wiso.uni-augsburg.de/gi-files/MEMO/Memorandum-Februar-2002_2.34.pdf, Abruf am 2002-02-28.

[Behl2000] *Behle, A.*: Wiederverwendung von Softwarekomponenten im Internet. Dissertation, Technische Hochschule Aachen, Dt. Univ.-Verlag, Wiesbaden 2000.

[BMBF2002] *Bundesministerium für Bildung und Forschung (BMBF)*: IT-Forschung 2006, Förderprogramm Informations- und Kommunikationstechnik. http://www.it2006.de/it-forschung_2006.pdf, Abruf am 2002-02-28.

[Börs1995] *Börstler, J.*: Feature-Oriented Classification for Software Reuse. In Proceedings SEKE 95, The 7th international conference on Software Engineering. Rockville, MD, USA, 22.-24.7.1995, S. 204-201.

[Bull2000] *Bullinger, H.-J. (Hrsg.)*: KoSPuD – Komponentenbasierte Software für Produkte und Dienstleistungen. Tagungsband, Fraunhofer IRB Verlag, ISBN 3-8167-5555-0, Stuttgart 2000.

[DiEs2001] *Dietzsch, A.; Esswein, W.:* Gibt es eine „Softwarekomponenten Industrie“ ? Ergebnisse einer empirischen Untersuchung. In: *Buhl, H. U.; Huther, A.; Reitwiesner, B. (Hrsg.):* Information Age Economy, 5. Internationale Tagung Wirtschaftsinformatik 2001, Tagungsband, 19.-21.9.2001, Augsburg, Physika-Verlag, ISBN 3-7908-1427-X , Heidelberg 2001, S. 697 – 710.

[Flas2002] *Flashline*: Component DTD, Beschreibung der Eigenschaften von Komponenten zur Einstellung in das Flashline-Archiv. http://www.componentregistry.com/dtd.jsp, Abruf am 2002-02-27.

[FWH+2000] *Fähnrich, K.-P.; Weisbecker, A.; Höß, O.; Kunsmann, J.:* Componentware – Vom Trend zum industriellen Einsatz. In: *Bullinger (Hrsg.)*: KoSPuD – Komponentenbasierte Software für Produkte und Dienstleistungen. Tagungsband, Fraunhofer IRB Verlag, ISBN 3-8167-5555-0, Stuttgart 2000, S. 1-37.

[GdV1999] *Gesamtverband der deutschen Versicherungswirtschaft e.V.:* Die Anwendungsarchitektur der Versicherungswirtschaft. Edition 1999. http://www.gdv-online.de/vaa/vaa_html/vaa3_0.htm, Abruf am 2002-02-27.

[GfK+2000] *GfK Marktforschung GmbH; Fraunhofer-Institut für Experimentelles Software Engineering (IESE); Fraunhofer-Institut für Systemtechnik und Innovationsforschung (ISI)*: Analyse und Evaluation der Softwareentwicklung in Deutschland. Abschlussbericht einer Studie für das Bundesministerium für Bildung und Forschung, Dezember 2000. http://www.dlr.de/IT/IV/Studien/evasoft_abschlussbericht.pdf, Abruf am 2002-02-25.

[GCO+2000] *Gibb, F.; McCartan, C.; O'Donnell, R.; Sweeney, N.; Leon, R.*: The integration of information retrieval techniques within a software reuse environment. In Journal of Information Science, Ausgabe 26 / 4, 2000.

[HöWe2001] *Höß, O.; Weisbecker, A.*: Komponentenbasierte Software für Produkte und Dienstleistungen (KoSPuD): Ergebnisse und Erfahrungen eines praxisorientierten Verbundforschungsprojekts. In: *Turowski, K. (Hrsg.)*: 3. Workshop komponentenorientierte betriebliche Anwendungssysteme (WKBA 3), Universität der Bundeswehr München und Gesellschaft für Informatik, Tagungsband, Frankfurt 2001, S. 1-13.

[JaGJ1997] *Jacobson, I.; Griss, M.; Johnson, P.*: Software Reuse – Architecture, Process And Organization For Business Success. Addison-Wesley Longman, New York 1997.

[Karl1995] *Karlsson, E.-A.*: Software Reuse – A Holistic Approach. Wiley, Chichester 1995.

[KeNS1992] *Keller, G.; Nüttgens, M.; Scheer, A.-W.*: Semantische Prozeßmodellierung auf der Grundlage "Ereignisgesteuerter Prozeßketten (EPK)". Veröffentlichung des Instituts für Wirtschaftsinformatik Heft 89, Saarbrücken 1992.

[PoBu2001] *Pohthong, A; Budgen, D.*: Reuse Strategies in software development: an empirical study. In: Information and Software Technology, 43 (2001), S. 561-575. 2001.

[Reif1997] *Reifer, D. J.*: Practical Software Reuse. Wiley, New York 1997.

[Same1997] *Sametinger, J.*: Software Engineering with Reusable Components. Springer, Berlin 1997.

[SoSo1998] *Sodhi, J.; Sodhi, P.*: Software Reuse – Domain Analysis and Design Processes. McGraw-Hill, New York 1998.

[Voas1998] *Voas, J. M.*: The Challenges of Using COTS Software in Component-Based Development. In IEEE Computer, June 1998, S. 53-59.

[WAD+2000] *Weisbecker, A.; Appl, J.; Drawehn, J.; Höß, O.; Schuster, E.*: Stand der Technik. In: *Weisbecker, A. (Hrsg.)*: KoSPuD – Komponentenbasierte Software für Produkte und Dienstleistungen. Abschlussbericht, Fraunhofer IRB Verlag, ISBN 3-8167-5556-9, Stuttgart 2000, S. 13-61.

[W3C2002] *World Wide Web Consortium (W3C):* Web Services Activity. http://www.w3.org/2002/ws/, Abruf am 2002-02-28.

[WeGr1998] *Weisbecker, A.; Groh, G. (Hrsg.)*: PROMPT – Organisationsgestaltung und Methoden für menschengerechte Software-Entwicklungsprozesse. Abschlussbericht, Fraunhofer IRB Verlag, ISBN 3-8167-5176-8, Stuttgart 1998

[WeHS2001] *Weisbecker, A.; Höß, O.; Strauß, O.*: Organisatorische und technische Maßnahmen zur Wiederverwendung von Komponenten. Deliverable D2.7, Projekt Adaptive READ, gefördert durch das BMBF, Stuttgart 2001.