Ansätze zur Wiederverwendung von Software im Rahmen der Softwareindustrialisierung am Beispiel von Softwarekomponenten, serviceorientierten Architekturen und modellgetriebenen Architekturen

Die Autoren

Daniel Pfeiffer
Axel Winkelmann

Dipl.-Wirt.-Inf. Daniel Pfeiffer
Dr. Axel Winkelmann
European Research Center for Information Systems
Institut für Wirtschaftsinformatik
Leonardo-Campus 3
48149 Münster
{daniel.pfeiffer|axel.winkelmann}@ercis.de
http://www.ercis.de

Entwicklung der Softwareindustrialisierung

Erste Anstrengungen, die Softwareentwicklung effizienter zu gestalten, gehen bereits auf die 50er-Jahre des letzten Jahrhunderts zurück. Sie waren eine Konsequenz aus immer wiederkehrenden, gleichartigen Programmieraufgaben die nach dem Einsatz von abstrakteren Beschreibungsformen verlangten. In der Folge wurden die ersten höheren Programmiersprachen entwickelt, die über Konstrukte wie bedingte Anweisungen oder Schleifen verfügten und so halfen, den Programmieraufwand zu reduzieren. Das Gesamtsystem wurde in eine Menge wieder verwendbarer Teillösungen (Prozeduren) zerlegt. Ende der 60er Jahre wurde erstmals der Begriff der Softwarekomponente geprägt und als Voraussetzung für den Übergang hin zu einer industriellen Softwareentwicklung benannt [McIl69]. Zu diesem Zeitpunkt begann auch die Debatte um die so genannte Softwarekrise [Cox90; Dijk72]. Diese Diskussion, deren Auswirkungen noch bis heute zu spüren sind, charakterisiert eine Situation, in der Softwareentwicklung zu teuer ist, die Qualität der Anwendungen zu stark variiert und ihre Gestaltung nur unzureichend gesteuert werden kann. Dies führte in den 70er Jahren verstärkt zu Versuchen zur Wieder- und Mehrfachverwendung von Programmcode, um die Qualität der Anwendungen zu erhöhen und ihre Entwicklungskosten zu senken. Wiederverwendbare Software kann verstanden werden als „Code, der auf einfache Weise, unverändert oder mit nur wenigen Änderungen, in mehreren Programmen mit wechselnden Anforderungen möglicherweise auf verschiedenen Systemen genutzt werden kann“ [CaEl95]. In Analogie zu den vorgefertigten Bauteilen in der Automobilindustrie bestand das Ziel der Softwareentwicklung in der Kombination von bestehenden Artefakten, so genannten Komponenten, zu einem neuen Produkt. Der komponentenbasierten Gestaltung von Anwendungssystemen war jedoch zunächst nicht derselbe Erfolg beschert wie der modularen Produktionsweise innerhalb der Ingenieurdisziplinen. Nicht vernetzte Einzelplatzanwendungen und ein tief verankertes Selbstverständnis der Softwareentwickler als Künstler und Handwerker [Knut68] verhinderten die Verbreitung und Nutzung von Softwarekomponenten [Grif98].

In den 80er Jahren rückten Fragen über eine mögliche Automatisierung der Softwareentwicklung wieder stärker in das Blickfeld von Wissenschaft und Praxis. Es entstand die Idee des Computer-Aided Software Engineering (CASE), die Anfang der 90er Jahre ihren Höhepunkt erreichte. Softwarebasierte Werkzeuge wurden nun als eine Möglichkeit wahrgenommen, die Softwareentwicklung vorzubereiten und wirksam zu unterstützen. Die Begleitung der Planung, des Entwurfs und der Dokumentation der Arbeitsergebnisse durch eine CASE-Software hatte zum Ziel, die Entwicklung von Anwendungen zu beschleunigen und gleichzeitig zuverlässiger zu machen. Allerdings gelang es nicht, das ursprünglich angestrebte Potenzial von CASE in vollem Umfang auszuschöpfen. Gründe dafür waren beispielsweise technisch unausgereifte Produkte, fehlendes methodisches Know-how oder die Überforderung der Mitarbeiter und eine sich daraus entwickelnde Abwehrhaltung gegen den Ansatz [CoOn97; McCl89].

Ausgelöst durch eine zunehmende Vernetzung von Computersystemen und der Etablierung des objektorientierten Paradigma kam es Mitte der 90er Jahre zu einer Wiederbelebung der Komponentenidee. Gerade die Vermarktung von Softwarekomponenten als eigenständige Produkte entwickelte sich in diesem Zeitraum. Dabei wurde deutlich, dass formale Spezifikationen ein notwendiges Kriterium dafür darstellen, um Code sinnvoll wieder verwenden und mit anderen Programmteilen kombinieren zu können [SaYi06]. Gleichzeitig wurde erkannt, dass Wiederverwendung keineswegs uneingeschränkt möglich ist und häufig Anpassungen sowie kleinere Änderungen notwendig werden [Rost97]. Der Wiederverwendung von Komponenten wurden erhebliche Zeit- und Kostenvorteile bei der Gestaltung von Anwendungssystemen prognostiziert [GMYK06].

Die sich daraufhin um die Jahrtausendwende entwickelnden Komponententechnologien entstanden vor dem Hintergrund des rasanten Wachstums des Internets und der damit verbundenen technischen Möglichkeiten. Die Fortschritte auf diesem Gebiet gaben den Auslöser zur Entwicklung

eines neuartigen Konzepts für Anwendungsarchitekturen. Anstatt von technologisch proprietären Komponenten sollten nun über das Internet erreichbare Webservices die Funktionalität von Anwendungen verteilt zur Verfügung stellen [McSZ01; Yang03]. Der Gedanke, die gesamte Anwendungslandschaft eines Unternehmens nach diesem Prinzip aufzubauen, mündete in der serviceorientierten Architektur (Service-Oriented Architecture, (SOA)) [PaGe03]. Gleichzeitig bedeuteten konkurrierende Komponentenstandards eine deutliche Zunahme der Komplexität für Anwendungssystementwicker. Der Wunsch, die Mehrfachentwicklung von Programmlogik für unterschiedliche Komponentenplattformen zu vermeiden, führte schließlich zur modellgetriebenen Architektur (Model Driven Architecture (MDA)). Die MDA verfolgt das Ziel, Entwurfswissen mit Hilfe von Modellen automatisiert auf unterschiedliche technologische Umgebungen zu übertragen [Uhl03].

Gegenstand dieses Beitrags ist es, wichtige Strömungen in der Softwareentwicklung ab Mitte der 90er Jahre aus der Perspektive der Softwareindustrialisierung zu betrachten und ihre Entstehung anhand von Belegen im Internet nachzuzeichnen. Industrialisierung wird dabei als ein Prozess verstanden, der insbesondere mit drei Effekten einhergeht: Standardisierung, Automatisierung und Wiederverwendung. Die Aspekte der Wiederverwendung werden anhand der Softwarekomponenten beschrieben. Die Automatisierung wird im Kontext der MDA betrachtet. SOA wird als ein Ansatz zur Standardisierung von Anwendungsarchitekturen vorgestellt. Zudem werden Blogs sowie Konferenzen und Kongresse beleuchtet, die sich mit dem Thema Softwareindustrialisierung auseinandersetzen.

Komponentenbasierte Softwareentwicklung

Eine Komponente wird in der Literatur je nach Betrachtungswinkel unterschiedlich definiert [CaLo00]. Unter einer Komponente kann ein unabhängiger, wieder verwendbarer Softwarebaustein verstanden werden, der eine Schnittstellenbeschreibung sowie eine Spezifikation seiner funktionalen und nicht-funktionalen Eigenschaften umfasst [AHLM03; CCGS2005; GrTh00]. Eine Komponente grenzt sich gegenüber ihrer Umgebung ab und stellt für ihre Anwender eine Black Box dar. Dies bedeutet, dass die interne Struktur der Komponente von außen nicht sichtbar ist und auf die Funktionalität der Komponente nur über ihre vertraglich vereinbarte Schnittstelle zugegriffen werden kann. Die Wiederverwendbarkeit von Komponenten ergibt sich aus ihrem Verzicht auf einen direkten Kontextbezug und der daraus resultierenden Abstraktion von spezifischen, singulären Einsatzbedingungen [DiEs00]. Sie können durch Kombination mit anderen Komponenten und zusätzlichem Verbindungscode zu vollständigen Softwaresystemen zusammengesetzt werden.

Die Begriffe Objekt und Komponente stehen in einer engen inhaltlichen Beziehung zueinander. Folgt man dem objektorientierten Paradigma, so besteht eine Komponente aus mehreren Objekten, zwischen denen Abhängigkeiten bestehen. Allerdings kann eine Komponente auch auf Grundlage anderer Implementierungsparadigmen, wie beispielsweise der funktionalen Programmierung, entwickelt werden. Eine notwendige Abhängigkeit zwischen Komponenten und Objektorientierung besteht also nicht, obgleich beide Ansätze häufig gemeinsam anzutreffen sind. Ein wesentlicher Unterschied zwischen komponentenbasierter und objektorientierter Softwareentwicklung besteht darin, dass erstere primär an Interfacegestaltung, Wiederverwendung von Black-Box-Elementen und Laufzeitverhalten ausgerichtet ist [CHJK02].

Eine zentrale Problematik der Komponentenorientierung ist die Wahl der Komponentengranularität. Je feingranularer die Komponenten, desto weniger Funktionalität enthalten diese, so dass sie ggf. nur gemeinsam sinnvoll wieder verwendet werden können. Bei zu grober Granularität entsteht eine Form der Modularisierung, die nur wenig Wiederverwendungspotential bietet und im Extremfall an den monolithischen Aufbau von Softwaresystemen erinnert.

Historisch lassen sich verschiedene Komponententechnologien identifizieren, die unter verschiedenen Blickwinkeln zur Softwareindustrialisierung beigetragen haben. Tabelle 1 beschreibt die wesentlichen Meilensteine dieser Entwicklung.

Microsofts Vorstellung des Component Object Models (COM) bildete den Ausgangspunkt der Wiederbelebung des Komponentenansatzes. Daraufhin wurden konkurrierende Vorschläge von u. a. Borland und SUN verbreitet. Vor dem Hintergrund der sich rasant entwickelten graphischen Benutzeroberflächen, fokussierten diese Ansätze zunächst auf eine effiziente Um-

Tabelle 1 Komponententechnologien in der Softwareentwicklung

Ursprungsjahr	Firma/Organisation	Produkt	Grundlage	URL
1990	Microsoft	Component Object Model (COM)	Object Linking and Embedding (OLE)	http://www.microsoft.com/com/
1990	Oberon microsystems	BlackBox Component Builder	Component Pascal	http://www.oberon.ch/blackbox.html
1991	Borland	Delphi	Pascal	http://www.borland.com/de/products/delphi/
1991	OMG	Common Object Request Broker Architecture (CORBA)	Interface Definition Language (IDL)	http://www.omg.org/corba/
1996	SUN	Java Beans	Java	http://java.sun.com/products/javabeans/
1999	SUN	Enterprise Java Beans (EJB)	Java, JEE	http://java.sun.com/products/ejb/
2000	Microsoft	.NET	(D)COM	http://www.microsoft.com/net/
2002	OMG	CORBA Component Model (CMM)	CORBA	http://www.omg.org/technology/documents/formal/components.htm

Tabelle 2 Plattformanbieter für Komponententechnologien

Hersteller	Produkt	Technologie	Lizenz	URL
	JacORB	CORBA	OS	http://www.jacorb.org/
	MICO	CORBA	OS	http://www.mico.org/
	OpenORB	CORBA	OS	http://openorb.sourceforge.net/
	OmniORB	CORBA	OS	http://omniorb.sourceforge.net/
	Orbit	CORBA	OS	http://orbit-resource.sourceforge.net/
	QEDO	CCM	OS	http://qedo.berlios.de/
	Mono	.NET	OS	http://www.mono-project.com/
Apache	Geronimo	JEE	OS	http://geronimo.apache.org/
Bea Systems	WebLogic Application Server	JEE	CS	http://de.bea.com/products/weblogic/
Borland	VisiBroker	CORBA	CS	http://www.borland.com/visibroker/
IBM	WebSphere Application Server	JEE	CS	http://www.ibm.com/software/websphere/
IONA	Orbix	CORBA	CS	http://www.iona.com/products/orbix/
Microsoft	.NET	.NET	CS	http://www.microsoft.com/net/
ObjectWeb	OpenCCM	CCM	OS	http://openccm.objectweb.org/
ObjectWeb	JOnAS	JEE	OS	http://jonas.objectweb.org/
Oracle	Application Server	JEE	CS	http://www.oracle.com/appserver/
Red Hat	JBoss Application Server	JEE	OS	http://labs.jboss.com/portal/jbossas
SAP	Netweaver	JEE	CS	http://www.sap.com/germany/plattform/netweaver/
SUN	Java System Application Server	JEE	CS	http://www.sun.com/software/products/appsrvr/

setzung von Benutzerschnittstellen. Die Java Beans von SUN sind eine Technologie, die primär auf die Spezifikation von GUI-Komponenten ausgerichtet ist. Komponenten erwiesen sich jedoch nicht nur auf der Client-Seite als nützlich, sondern eroberten allmählich auch den Server-Bereich. Die Enterprise Java Beans (EJB) wurden von SUN als ein Ansatz zur komponentenbasierten Realisierung von verteilten Geschäftsanwendungen vorgestellt. Auch Microsoft baute seinen COM-Ansatz über verschiedene Zwischenstufen zu einer für den Unternehmenseinsatz geeigneten Komponententechnologie aus. Diese Bestrebungen mündeten schließlich in der .NET-Plattform.

Parallel zu Unternehmen wie Borland, Microsoft und SUN beschäftigte sich auch die Object Managament Goup (OMG) mit der Realisierung von auf objektorientierten Technologien basierenden Geschäftsanwendungen. Mit der Common Object Request Brocker Architecture (CORBA) stellte sie einen Standard zur programmier- und plattformunabhängigen Entwicklung verteilter Anwendungen vor [Wesk99]. Zentraler Ausgangspunkt dieser Spezifikation ist der Object Request Broker (ORB) als Kommunikationszentrale für Anwendungen. Die vom Server zur Verfügung gestellte Schnittstellstellenbeschreibung der Dienste wird durch die Interface Definition Language (IDL) realisiert. Auf Grundlage dieser Spezifikationen in der IDL lassen sich, in Abhängigkeit von der zur Implementierung der Funktionalität verwendeten Programmiersprache, die jeweiligen Gerüste für die zu implementierenden Klassen generieren. Zudem werden in der CORBA-Spezifikation zahlreiche Dienste, z. B. zur Transaktionsverwaltung oder Identifikation von Diensten, definiert, die eine CORBA-Plattform zu realisieren hat. Das Corba Component Model (CMM) erweitert diese Spezifikation um den Aspekt einer Laufzeitumgebung von Komponenten und stellt damit eine verallgemeinerte Form des Enterprise JavaBeans (EJB)-Ansatzes zur Verfügung.

Der Markt für komponentenbasierte Geschäftanwendungen wird derzeit von Microsoft mit .NET und SUN mit JEE dominiert. CORBA hat im Bereich von Altsystemen noch große Bedeutung, und es existieren einige frei verfügbare CCM-Implementierungen. Tabelle 2 listet bedeutende Produkte bzw. Plattformanbieter zu den verschiedenen Komponententechnologien auf und gibt an, ob es sich um Closed Source- (CS) oder Open Source- (OS) Lösungen handelt.

Serviceorientierte Softwareentwicklung

Die serviceorientierte Softwareentwicklung basiert auf einem Architekturkonzept, bei dem die Geschäftslogik gekapselt als Dienst für Anwendungen zur Verfügung gestellt wird. Sie basiert auf der serviceorientierten Architektur (SOA), die in ihrer Interpretation als Managementansatz eine an den Geschäftsprozessen ausgerichtete IT-Infrastruktur anstrebt [CSDS03]. Die serviceorientierte Softwareentwicklung greift in großen Umfang – aber nicht ausschließlich – auf das Konzept des Web Services zurück [PaGe03].

Die Standards, die zur Charakterisierung von Web Services dienen, basieren hauptsächlich auf der Extensible Markup Language (XML). Ein Web Service stellt eine klar definierte Leistung zur Verfügung, die von einem Dienstanbieter offeriert und einem oder mehreren Dienstnutzern verwendet werden kann. Die Schnittstelle und funktionale Spezifikation eines Web Services sind vertraglich vereinbart. Die Beschreibung der Diensteschnittstelle erfolgt in der Web Services Description Language (WSDL). Über zentrale Verzeichnisdienste werden die in der Universal Description, Discovery and Integration (UDDI)-Sprache charakterisierten Dienste für potenzielle Service-Nutzer auffindbar gemacht. Zum Austausch von Nachrichten nutzen Web Services das Simple Object Access Protocol (SOAP). Das Zusammenspiel unterschiedlicher Web Services, mit dem Ziel die Funktionalität vollständiger Anwendungen zu erreichen, wird durch die Business Process Execution Language (BPEL) beschrieben.

In Tabelle 3 werden die wesentlichen technologischen Grundlagen für die serviceorientierte Softwareentwicklung dargestellt.

Tabelle 3 Sprachen auf dem Weg zur SOA

Ursprungsjahr	Firma/Organisation	Sprache	Grundlage	URL
1998	W3C	Extensible Markup Language (XML)	Standard Generalized Markup Language (SGML)	http://www.w3.org/XML/Core/
1999	W3C/IBM/Microsoft u. a.	SOAP	XML	http://www.w3.org/TR/soap/
2000	OASIS	Universal Description, Discovery and Integration (UDDI)	XML	http://www.uddi.org/
2001	W3C/Ariba/IBM/Micosoft	Web Services Description Language (WSDL)	XML	http://www.w3.org/TR/wsdl.html
2002	IBM/BEA/Microsoft/ SAP/Siebel/OASIS	Business Process Execution Language (BPEL)	XML, WSDL	http://www.oasis-open.org/committees/ tc_home.php?wg_abbrev=wsbpel

Tabelle 4 Anbieter von SOA-basierten Lösungen

Hersteller	Produkt	Kat.	Lizenz	URL
Amberpoint	SOA Management System	SOA	CS	http://www.amberpoint.com/solutions/
Bea Systems	Aqualogic	PA	CS	http://de.bea.com/products/aqualogic/
Cape Clear	ESB Platform	SOA	CS	http://www.capeclear.com/products/
IBM	Websphere	PA	CS	http://www.ibm.com/software/websphere/
InterSystems	Ensemble	EAI	CS	http://www.intersystems.de/ensemble/
Inubit	inubit IS	EAI/BPM	CS	http://www.inubit.com/index.php?id=3&sub=1&lang=de
IONA	Artix	PA	CS	http://www.iona.com/solutions/soa/
iWay Software	iWay Suite	EAI	CS	http://www.iwaysoftware.com/products/
Magic	Ibold	EAI	CS	http://www.magicsoftware.com/ibolt/
Microsoft	.NET	PA	CS	http://msdn.microsoft.com/architecture/soa/
Oracle	SOA Suite	PA	CS	http://www.oracle.com/technologies/soa/soa-suite.html
Progress	Actional, Sonic	SOA	CS	http://www.progress.com/products/
Red Hat	JBoss	PA	OS	http://labs.jboss.com/portal/jbossesb/
SAP	Netweaver	PA	CS	http://www.sap.com/germany/plattform/enterprisesoa
Seagull	LegaSuite	EAI	CS	http://www.seagullsoftware.com/products/legasuite.html
SOA Software	Infrastructure Suite	SOA	CS	http://www.soa.com/
Software AG	Crossvision	EAI	CS	http://www.softwareag.com/de/solutions/soa
SUN	Java CAPS	PA	CS	http://developers.sun.com/prodtech/javacaps
Tibco	BusinessWorks/iProcess Suite	EAI/BPM	CS	http://www.tibco.com/solutions/soa/
Vitra	BusinessWare	EAI/BPM	CS	http://www.vitria.com/BusinessWare/
Webmethods	Fabric	EAI	CS	http://www.webmethods.com/fabric

Tabelle 5 Standards der MDA

Ursprungsjahr	Organisation	Sprache/Produkt	Grundlage	URL
1997	OMG	Unified Modeling Language (UML)	OOSE, OMT, OPEN/OML, u. a.	http://www.uml.org/
2000	OMG	Meta Object Facility (MOF)	UML	http://www.omg.org/mof/
2003	OMG	Model Driven Architecture (MDA)	UML, MOF	http://www.omg.org/mda/
2004	OMG	UML Profile for Enterprise Distributed Object Computing (EDOC)	UML	http://www.omg.org/technology/documents/formal/edoc.htm

Ein Vergleich der Tabellen 2 und 4 zeigt deutlich, dass der Bereich der serviceorientierten Softwareentwicklung von Herstellern dominiert wird, die gleichzeitig auch im Komponententechnologiemarkt aktiv sind. Als wichtiger Grund dafür kann die enge technologische Verwandtschaft von Komponenten und Diensten sowie der korrespondierenden Architekturen angeführt werden. Viele der Plattformanbieter (PA) haben ihre bestehenden Produkte leicht modifiziert oder erweitert, um sich im SOA-Kontext mit ihren Lösungen positionieren zu können. Zudem stoßen in dieses neue Marktsegment auch Unternehmen, die sich zuvor hauptsächlich mit Enterprise Application Integration (EAI) und Business Process Management (BPM) beschäftigt haben. Der Anteil der reinen SOA-Produkte ist vergleichsweise gering. Tabelle 4 enthält wichtige Produkte aus dem Umfeld der service-orientierten Softwareentwicklung.

Modellgetriebene Softwareentwicklung

Traditionelle Ansätze der Softwareentwicklung trennen zwischen fachkonzeptioneller Erfassung der (betriebswirtschaftlichen) Anforderungen sowie deren Umsetzung in Software. Der Quellcode einer Anwendung wird in Abhängigkeit von den fachlichen Voraussetzungen und den spezifischen technologischen Rahmenbedingungen entwickelt. Die losgelöste Pflege von Fachkonzept und Implementierung hat sich im Zeitablauf jedoch als problematisch erwiesen. Neue fachliche Anforderungen, Änderungen und Anpassungen des Systems führen zu einer hohen Komplexität in der Softwareentwicklung. Häufig entfällt aus Zeitgründen eine gründliche fachkonzeptuelle Spezifikation und somit zugleich Dokumentation der Anforderungen und des Systems [Fett04]. Die Anforderungsanalyse wird nicht selten zu Gunsten der schnellen Realisierung einzelner Programmmodule vernachlässigt. Konflikte zwischen unterschiedlichen Nutzergruppen fallen damit allerdings erst im späteren Nutzungsstadium auf, da Widersprüche vorweg nicht ausreichend geklärt und ausgeräumt werden. Die Existenz unterschiedlicher Komponententechnologien und Programmiersprachen sorgt dafür, dass für jede dieser Plattformen eine separate manuelle Quellcode-Erstellung erforderlich ist. Eine Überführung der bereits als Software vorliegenden betriebswirtschaftlichen Prozesse in andere Programmiersprachen und Technologien ist daher nicht ohne größeren Anpassungsaufwand möglich und mit einem aufwändigen Reverse-Engineering verbunden.

Ziel der MDA ist es, den Entwicklungsprozess selbst auf ein höheres Abstraktionsniveau zu bringen [AtKü03]. Statt die fachkonzeptuellen Modelle lediglich für die Aufnahme des organisatorischen Ist-Zustandes und der Spezifikation der Softwareanforderungen zu verwenden, bilden diese nun die entscheidende Grundlage des gesamten Entwicklungszykluses. Sie lösen damit den Quelltext in seiner Rolle als wichtigstes Artefakt der Softwareentwicklung ab. Die Anwendungssystemgestaltung im Kontext der MDA orientiert sich deutlich stärker am fachkonzeptuellen Modell und beschränkt sich nicht auf technische Aspekte [Dewa05]. Im Idealfall sollen betriebswirtschaftliche Fachkonzepte mittels MDA (teil-)automatisiert in Quellcode überführt werden. Damit soll die Trennung von Anwender- und Entwicklerwissen durch gemeinsames Erstellen der konzeptionellen Modelle überbrückt werden. Das von der OMG maßgeblich vorangetriebene MDA-Konzept verspricht, Anwendungswissen von Softwareentwurfsentscheidungen loszulösen und das in fachlichen Modellen festgehaltene Wissen soweit wie möglich automatisiert in technische, plattformunabhängige Modelle zu überführen. Diese plattformunabhängigen Modelle lassen sich dann in Abhängigkeit von den zur Verfügung stehenden Technologien in plattformspezifische Modelle und schließlich in Code transformieren [Uhl03].

Tabelle 5 enthält die wesentlichen Standards, die der MDA zugrunde liegen.

Eine wichtige Klasse von MDA-Lösungen stellen die UML Modellierungstools (UML MT) dar. Diese Produkte verfügen sehr häufig über die notwendige Funktionalität, um aus UML-Diagrammen die grundlegende Klassenstruktur der Anwendung für verschiedene Sprachen, wie Java, C# oder C++, zu generieren und bestehenden Quellcode in Form von UML-Diagrammen zu dokumentieren. Metamodellierungstools (MMT) bieten zum großen Teil eine Programmierschnittstelle an, um Modelle aus beliebigen Modellierungssprachen in eine alternative Form zu überführen. MDA-Generatoren (MDAG) sind auf das Einlesen und Transformieren von Modellen anhand von festgelegten Regeln spezialisiert. Tabelle 6 listet aktuelle Produkte aus dem MDA- und Modellierungsumfeld auf.

Informationsquellen zu Themen der Softwareindustrialisierung

Blogs, Foren und Informationsportale

Informationsportale und Online-Tagebücher (Blogs) einzelner Autoren sind ebenso wie Foren ergiebige Quellen für Informationen über aktuelle Themen der Softwareindustrialisierung [HiWi05]. Anders als Bücher können sie tagesaktuell neue Ereignisse und Technologien kommentieren. Meist findet hierbei eine Vermischung verschiedener Kommunikationsarten statt, da in den Gästebüchern einzelner Blogs nicht selten interessante Diskussionen zu einzel-

Tabelle 6 Anbieter von MDA-orientierten Produkten

Hersteller	Produkt	Kategorie	Lizenz	URL
	Eclipse Modeling Project	UML MT/MDAG	OS	http://www.eclipse.org/modeling/
	StarUML	UML MT	OS	http://staruml.sourceforge.net/en/
	ArgoUML	UML MT	OS	http://argouml.tigris.org/
	Graphical Modeling Framework	MMT	OS	http://www.eclipse.org/gmf/
	AndromedaMDA	MDAG	OS	http://www.andromda.org/
	MOFScript	MDAG	OS	http://www.modelbased.net/mofscript/
	openArchitectureWare	MDAG	OS	http://www.openarchitectureware.org/
	OpenMDX	MDAG	OS	http://www.openmdx.org/
Altova	UModel	UML MT	CS	http://www.altova.com/products/umodel/uml_tool.html
ARTiSAN	Studio	UML MT	CS	http://www.artisansw.com/products/
BOC	Adonis	MMT	CS	http://www.boc-eu.com/
Borland	Together	UML MT	CS	http://www.borland.com/de/products/together/
Compuware	OptimalJ	MDAG	CS	http://www.compuware.com/products/optimalj/
Embarcadero	Describe	UML MT	CS	http://www.embarcadero.com/products/describe/
Gentleware	Poseidon	UML	CS	http://www.gentleware.com/products.html
IBM	Rational Software Architect/ Modeller	UML MT/MDAG	CS	http://www.ibm.com/software/awdtools/modeler/swmodeler/
IBM	Model Transformation Framework	MDAG	OS	http://www.alphaworks.ibm.com/tech/mtf
I-Logix	Rhapsody	UML MT/MDA G	CS	http://www.ilogix.com/sublevel.aspx?id=53
innoQ	iQGen	MDAG	CS	http://www.innoq.com/iqgen/
Integranova	OlivaNova	MDAG	CS	http://www.integranova.de/produkte.php
Interactive Objects	Arcstyler	MDAG	CS	http://www.interactive-objects.com/products/arcstyler
MDWorkbench	MDWorkbench	MDAG	CS	http://www.mdworkbench.com/products.php
Metacase	MetaEdit+	MMT	CS	http://www.metacase.com/mep/
MIA-Software	Model-in-Action	MDAG	CS	http://www.mia-software.com
NoMagic	MagicDraw	UML MT	CS	http://www.magicdraw.com/
Objecteering	Objecteering	UML MT	CS	http://www.objecteering.com/
OMONDO	EclipseUML Studio	UML MT	CS	http://www.omondo.com/
Semture	Cubetto Toolset	MMT	CS	http://www.semture.de/cubetto
Sparx Systems	Enterprise Architect	UML MT	CS	http://www.sparxsystems.com/
SUN	Java Studio Enterprise	UML MT	CS	http://developers.sun.com/prodtech/javatools/jsenterprise/
Telelogic's	System Architect	UML ML	CS	http://www.telelogic.com/products/systemarchitect/
University of Kent	Kent Modelling Framework	MDAG	OS	http://www.cs.kent.ac.uk/projects/kmf/
Visual Paradigm	Visual Paradigm for UML	UML ML	CS	http://www.visual-paradigm.com/product/vpuml/
Xactium	XMF Mosaic	MDAG	CS	http://albini.xactium.com

nen Themen geführt werden oder Fremdautoren Aufsätze beisteuern. Häufig bloggen Insider wie Nicolas Carr („IT doesn't matter"), Matt Cutts (Google Webspam-Verantwortlicher) oder Ed Brill (IBM Market Management Lotus und Domino) ihre Ansichten zu bestimmten Technologien oder Entwicklungen. Die Übersicht in Tabelle 7 gibt aufgrund der Vielzahl an Blogs und Foren nur einen ausgewählten Teil wieder.

Tabelle 7 Blogs zu Themen der Softwareindustrialisierung

Name	URL	Inhalt
Blogs, Foren und Informationsportale zur Softwareindustrialisierung		
Adding Simplicity – An Engineering Mantra	http://www.addsimplicity.com/	Ebay-Architekt Dan Pritchett berichtet über Software- und Hardware-Design-Themen
Bernds Management-Welt	http://www.heise.de/ix/blog/2/	Bernd Oestereich berichtet über strategische IT-Themen und Objektorientierung
Ed Brill – Collaboration, technology, travel, and more	http://www.edbrill.com/	Blog des IBM-Verantwortlichen für Lotus Notes und Domino
Entwicklerforum	http://entwickler-forum.de/	Diskussionsforen mit über 100.000 Beiträgen zu Entwicklungsthemen wie Java, .Net, Datenbanken, Eclipse usw.
Joel on Software	http://www.joelonsoftware.com/	Allgemeiner Blog von Joel Spolsky zu Technik- und Management-Themen der Softwareindustrialisierung
Lawblog – Udo Vetter	http://www.lawblog.de/	Juristisches Blog mit vielen Themen aus dem IT-Bereich
Matt Cutts: Gadgets, Google, and SEO	http://www.mattcutts.com/blog/	Technische und strategische Informationen gibt Googles Webspam-Entwicklungschef Matt Cutts
Microsoft-Blogs	http://blogs.msdn.com/	Knapp 100 Entwickler des Konzerns informieren in Blog-Form über technische Details
Rough Type	http://www.roughtype.com/	Blog des Erfolgsautors Nicholas G. Carr über strategische Aspekte der Softwareindustrialisierung
The Industrialization of Software	http://softwareindustrialization.com/	Ausführlicher Blog mit rund 80 Artikeln von Mitch Barnett zur Frage, was Softwareindustrialisierung ist
Blogs, Foren und Informationsportale zu Komponententechnologien		
Der Dotnet-Doktor	http://www.heise.de/ix/blog/1/	Holger Schwichtenberg schreibt über Entwicklungen im .Net-Umfeld
JavaSPEKTRUM Blogosphäre	http://www.sigs.de/blog/js/	Java-Berichte aus dem Umfeld der Zeitschrift JavaSPEKTRUM
OrbZone – CORBA Community	http://www.orbzone.org/	Blog und Nachrichtenportal für CORBA-Entwickler
TheServerSide.com	http://www.theserverside.com/	Diskussionen und Blogeinträge zu serverseitigem Java
Blogs, Foren und Informationsportale zu SOA		
InformationWeek – SOA	http://www.informationweek.com/software/soa/	Von der Redaktion der InformationWeek aufgearbeitete Informationen zu SOA
Moderierter SOA-Marktplatz	http://www.soa-forum.net/	Informationsplattform für Hersteller und Anwender zu SOA-Informationen und -Software
Service-Oriented Architecture @ ZDNet.com	http://blogs.zdnet.com/service-oriented/	Verschiedene Autoren diskutieren über technische und strategische Aspekte von SOA
SOA – IT-Blog	http://itblog.eckenfels.net/categories/14-SOA	SOA- und IT-Themen diskutiert von Bernd Eckenfels
SOA Blog	http://www.explore-soa.de/soa/de/soablog	Ein von T-Systems-Mitarbeitern betriebener Blog zu SOA-Themen
SOA-Expertenrat	http://www.computerwoche.de/soa-expertenrat/	Deutschsprachige Expertendiskussionen zu SOA im Rahmen des Internetangebots der Computerwoche
SOA-Trends	http://www.computerwoche.de/soa-trends/	Redaktionell von der Computerwoche aufgearbeitete Hintergrundinformationen, Softwareübersichten und Architekturerläuterungen
Blogs, Foren und Informationsportale zu MDA		
H.S. Lahman on OO and MDA	http://pathfinderpeople.blogs.com/hslahman/	Blog mit technischen Details zur Objektorientierung und MDA
Modelbased.net	http://www.modelbased.net/	Informationsseite zu Themen der Model-Driven System-Entwicklung, hervorgegangen aus dem EU-Projekt ModelWare
The Enterprise Model-Driven Solutions Set	http://mda-soa.blogspot.com/	Blog-Beiträge zur Diskussion um Model-Driven Architectures

Tabelle 8 Konferenzen und Kongresse zu Themen der Softwareindustrialisierung

Name	URL	Inhalt
Kongresse und Konferenzen zur Softwareindustrialisierung		
EKON	http://entwickler-konferenz.de/	Verschiedene Themen für Softwareentwickler und -architekten, Datenbankexperten und IT-Manager
ENASE	http://www.enase.org/index.htm	Internationale Konferenz zur Evaluation von neuen Ansätzen für das Software Engineering
EuroDevCon	http://eurodevcon.com/	Developer-Konferenz parallel zur EKON
International Conference on the Quality of Software-Architectures	http://qosa.ipd.uka.de/	Konferenz zu Themen von Architekturdesign, -bewertung und -management
The Enterprise Computing Conference	http://edoc.mitre.org/	Internationale Konferenz mit Fokus auf Technologien für Unternehmensanwendungen
Kongresse und Konferenzen zu Komponententechnologien		
Euromicro-Konverenz-Track	http://www.idt.mdh.se/ecbse/2007/	Track zum komponentenbasierten Software Engineering im Rahmen der Euromicro-Konferenz
European Conference on Object-Oriented Programming	http://2007.ecoop.org/	Konferenz behandelt verschiedene Themen der Objektorientierten Programmierung
International ACM SIGSOFT Symposium on Component-Based Software Engineering	http://www.csse.monash.edu.au/~hws/CBSE10/index.shtml	Symposium für komponentenorientierte Softwareentwicklung
Net.Object Days	http://www.netobjectdays.org/	Internationale Konferenz mit Themenschwerpunkt u. a. im Bereich Komponententechnologie
Tools – Europe	http://tools.ethz.ch/	Konferenz über Objekte, Modelle, Komponenten, Patterns
Kongresse und Konferenzen zu SOA		
Business Integration Forum	http://www.eai-forum.de/	Konferenz zu dynamischen und event-orientierten End-to-End Geschäftsprozessen durch SOA, EAI, BPM, EDA und BAM
Eclipse-Forum	http://eclipseforumeurope.com/	Diskussion von Ansätzen im Rahmen der Eclipse-Plattform
Enterprise Architektur Konferenz	http://eakon.de/	Diskussion neuer IT-Architekturen und -Geschäftsmodelle, insbesondere SOA
Konferenz für Java, Enterprise Architekturen und SOA	http://jax.de/	JAX zählt zu den größten Konferenzen für Enterprise-IT-Technologien in Europa mit über 150 Sessions und 10 Workshops
OMG Information Days	http://www.sigs-datacom.de/sd/kongresse/infodays/2007/01/index.htm	Deutsche SOA-Tagung
SOA Days Business Conference	http://domains.euroforum.com/soa0307/	Konferenz zu SOA-Themen aus Managementperspektive
SOA-Kongress	http://www.soa-kongress.de/	Deutscher SOA-Kongress
Kongresse und Konferenzen zu MDA		
European Conference on Model Driven Architecture Foundations and Applications	http://www.haifa.il.ibm.com/conferences/ecmda2007/index.html	Europäische MDA-Konferenz
International Conference on Model Driven Engineering Languages and Systems	http://www.modelsconference.org/	Internationale Konferenz zu MDA-Themen und -Technologien.
MDEIS	http://www.iceis.org/workshops/mdeis/mdeis2007-cfp.html	MDA-Workshop im Rahmen der International Conference on Enterprise Information Systems
MoDSE	http://www.sciences.univ-nantes.fr/MoDSE2007/	MDA-Workshop im Rahmen der European Conference on Software Maintenance and Reengineering

Konferenzen und Kongresse

Die Anzahl an Konferenzen und Kongressen zu Themen der Softwareindustrialisierung ist sowohl auf wissenschaftlicher als auch praktischer Seite sehr groß. Eine Liste von allgemeinen, eher wissenschaftlich orientierten Konferenzen zu relevanten Themen findet sich auf der Webseite der AISWorld Community unter http://www.isworld.org/forthcoming/conferences.asp. In Tabelle 8 sind einige Konferenzen und Kongresse genannt, die sich schwerpunktmäßig mit Themen der Softwareindustrialisierung, insbesondere SOA, MDA und Komponententechnik beschäftigen.

Zusammenfassung

In diesem Beitrag wurden mit Komponenten, SOA und MDA wesentliche technologische Ansätze zur Softwareindustrialisierung vorgestellt. Jedes dieser Konzepte verspricht erhebliche Einsparungs- und Automatisierungspotenziale im Rahmen der Softwareentwicklung. Aufgrund der gegenüber von anderen Wirtschaftszweigen bislang vergleichsweise geringen Erfolge der Softwareindustrialisierung ist jedoch ein gesundes Maß an Skepsis angebracht, ob diese Ansätze die von ihnen versprochenen Effizienzsteigerungen voll realisieren können. Der hohe Umfang an verfügbaren Produkten zeigt jedoch, dass es seitens der Wirtschaft erhebliches Interesse gibt, die Softwareindustrialisierung weiter voran zu treiben. Gleichsam reflektieren die große Anzahl an Konferenzen und Kongressen das starke wissenschaftliche Interesse an dem Thema. Die enge Zusammenarbeit von Wissenschaft und Praxis ist Vorraussetzung dafür, um das Ziel einer weitgehend automatisierten Softwareentwicklung zu erreichen.

Literatur

[AHLM03] *Apperly, H.; Hoffman, R.; Latchem, S.; Maybank, B.; Piper, D.; Simons, C.; McGibbon, B.:* Service- and Component-Based Development. Kent 2003.

[AtKü03] *Atkinson, C.; Kühne, T.:* Model-Driven Development: A Metamodeling Foundation. In: IEEE Software 20 (2003) 5, S. 36–41.

[CaEl95] *Carroll, M. D.; Ellis, M. A.:* Designing and Coding Reusable C++. Reading, Mass. 1995.

[CaLo00] *Carney, D.; Long, F.:* What Do You Mean By COTS? Finally a Useful Answer. In: IEEE Software 17 (2000) 2, S. 83–86.

[CCGS2005] *Cooper, K.; Cangusu, J. W.; Ganessan Sankaranarayanan, R. L.; Soundararadjane, R.; Wong, E.:* An Empirical Study on the Specification of Components Using Fuzzy Logic. In: Proceedings of the 8th International ACM SIGSOFT Symposium on Component-based Software Engineering (CBSE). St. Louis, Missouri 2005, S. 155–170.

[CHJK02] *Crnkovic, I.; Hnkh, B.; Jonsson, T.; Kizilfan, Z.:* Specification, Implementation, and Deployment of Components. In: Communications of the ACM 45 (2002) 10, S. 35–40.

[CoOn97] *Coupe, R. T.; Onodu, N. M.:* Evaluating the impact of CASE: an empirical comparison of retrospective and cross-sectional survey approaches. In: European Journal of Information Systems 6 (1997) 1, S. 15–24.

[Cox90] *Cox, B. J.:* Planning the software industrial revolution. In: IEEE Software 7 (1990) 6, S. 25–33.

[CSDS03] *Casati, F.; Shan, E.; Dayal, U.; Shan, M.-C.:* Business-oriented management of Web services. In: Communications of the ACM 46 (2003) 10, S. 55–60.

[Dewa05] *Dewanto, L.:* Anwendungsentwicklung mit Model Driven Architecture – dargestellt anhand vollständiger Finanzpläne. Dissertation. Münster 2005.

[DiEs00] *Dietzsch, A.; Esswein, W.*: Komponentenbasierte Softwareentwicklung. In: Wirtschaftsinformatik 42 (2000) 1, S. 93–98.

[Dijk72] *Dijkstra, E. W.:* The humble programmer. In: Communications of the ACM 15 (1972) 10, S. 859–866.

[Fett04] *Fettke, P.:* Model Driven Architecture (MDA). In: WISU 35 (2004) 4, S. 460–464.

[GMYK06] *Gao, T.; Ma, H.; Yen, I.; Khan, L.; Bastani, F.:* A Repository for Component-Based Embedded Software Development. In: International Journal of Software Engineering & Knowledge Engineering 16 (2006) 4, S. 523–552.

[Grif98] *Griffel, F.:* Componentware: Konzepte und Techniken eines Softwareparadigmas. Heidelberg 1998.

[GrTh00] *Gruhn, V.; Thiel, A.:* Komponentenmodelle: DCOM, JavaBeans, Enterprise JavaBeans, CORBA. München 2000.

[HiWi05] *Hippner, H.; Wilde, T.:* Social Software. In: Wirtschaftsinformatik 47 (2005) 6, S. 441–444.

[Knut68] *Knuth, D. E.:* The art of computer programming. Reading, Mass. 1968.

[McCl89] *McClure, C.:* The CASE experience. In: Byte.com 14 (1989) 4, S. 235–242.

[McIl69] *McIlroy, M. D.:* Mass Produced Software Components. In: *Naur, P., Randell, B.* (Hrsg.): Software Engineering, Report on a conference sponsored by the NATO SCIENCE COMMITTEE. Garmisch, Germany 1968, S. 138–151.

[McSZ01] *McIlraith, S. A.; Son, T. C.; Zeng, H.:* Semantic Web Services. In: IEEE Intelligent Systems 16 (2001) 2, S. 46–53.

[PaGe03] *Papazoglou, M. P.; Georgakopoulos, D.:* Service-Oriented Computing. In: Communications of the ACM 46 (2003) 10, S. 25–28.

[Rost97] *Rost, J.:* Wiederverwendbare Software. In: Wirtschaftsinformatik 39 (1997) 4, S. 357–365.

[SaYi06] *Sadaoui, S.; Yin, P.:* Generalization and Instantiation for Component Reuse. In: International Journal of Software Engineering & Knowledge Engineering 16 (2006) 2, S. 175–200.

[Uhl03] *Uhl, A.:* Model Driven Architecture Is Ready for Prime Time. In: IEEE Software 20 (2003) 5, S. 70–72.

[Wesk99] *Weske, M.:* Business-Objekte: Konzepte, Architekturen, Standards. In: Wirtschaftsinformatik 41 (1999) 1, S. 4–11.

[Yang03] *Yang, J.:* Web service componentization. In: Communications of the ACM 46 (2003) 10, S. 35–40.