Auswahl geeigneter Technologien für betriebliche Integrationsszenarien

Jörg Friebe, Till Luhmann, Jürgen Meister

BTC AG, Escherweg 3, 26121 Oldenburg
OFFIS, Escherweg 2, 26121 Oldenburg
joerg.friebe@btc-ag.com, till.luhmann@btc-ag.com, juergen.meister@offis.de

Abstract: Integration ist eine wichtige Anforderung an Entwurf und Implementierung betrieblicher Informationssysteme. Am Markt gibt es ein breites Spektrum an anspruchsvollen, oft noch im Wandel begriffenen Werkzeugen und Technologien (z.B. ETL, Application Server, Web Services, EAI, WfMS, SOA) mit zum Teil ähnlicher Funktionalität zur Lösung der Integrationsaufgaben. Die spezifische Auswahl eines geeigneten Werkzeugs für ein konkretes Integrationsszenario gestaltet sich daher schwierig. Zur Unterstützung dieses Prozesses wird eine praxisorientierte Bewertungssystematik eingeführt: Diese beschreibt die Eignung der unterschiedlichen Werkzeugklassen für typische Integrationsaufgaben, die anhand praktischer Erfahrungen mit Integrationsprojekten gewonnen wurden. Durch die Anwendung solcher Orientierungshilfen kann der Praktiker eine schnelle Vorauswahl geeigneter Werkzeugklassen treffen, so dass im Folgeschritt für ein spezielles Integrationsszenario nur eine Teilmenge der in Frage kommenden Werkzeuge genauer auf ihre funktionale und technologische Eignung untersucht werden muss.

1 Einleitung

In der betrieblichen Praxis spielt die Integration bestehender Softwaresysteme eine wachsende Rolle. Dieser Trend wird sich fortsetzen, da sich die Software-Landschaften in den Unternehmen vor dem Hintergrund wachsender Kommunikationsbedürfnisse auch unternehmensübergreifend deutlich ausweiten, und vor allem aus Kostengründen mehr Bestehendes zu integrieren als Neues zu entwickeln sein wird. Ziel dieser Bestrebungen ist die möglichst weitgehende Automatisierung und Optimierung der Geschäftsprozesse innerhalb eines Unternehmens und entlang von Supply-Chains auch zwischen Unternehmen und anderen Akteuren (Öffentliche Verwaltung, Privatkunden, ...). Unterstützt werden diese Vorhaben durch eine Vielzahl von technologiespezifischen Integrationswerkzeugen mit zum Teil vergleichbaren Leistungsmerkmalen [Sto02]. In der Praxis stellt sich daher oftmals die Frage, wo die Stärken und Schwächen der einzelnen Technologien liegen, und welche dieser Ansätze für ein konkretes Integrationsszenario geeignet sind [Rit98].

Bei der Ermittlung passender Technologien müssen die Besonderheiten des betrachteten Integrationsszenarios berücksichtigt werden. Dies betrifft insbesondere die für das Szenario relevanten Integrationsebenen und die jeweils benötigte Funktionalität. Davon ausge-

hend können die Werkzeugklassen für die Integration ausgewählt werden. Eine zusätzliche Hilfestellung für die Technologieauswahl bieten die in diesem Beitrag eingeführten Heuristiken, die eine Bewertung der Technologien anhand praxisorientierter Merkmale erlauben. Neben den genannten Kriterien spielen ferner auch technologische und strategische Aspekte der aktuellen IT-Landschaft bei der Auswahl eine wichtige Rolle.

In Abschnitt 2 werden zunächst die Integrationsebenen sowie die dafür relevanten Funktionen und entsprechenden Werkzeugklassen vorgestellt. Aufgrund der vielfältigen funktionalen Überlappungen und Kombinationsmöglichkeiten einzelner Technologien empfiehlt es sich, bei der Betrachtung eines konkreten Szenarios anstelle einer funktionalen Sicht zunächst praktische Aspekte zu berücksichtigen. In Abschnitt 3 werden daher typische Merkmale von praktischen Integrationsszenarien diskutiert und anhand eines Fallbeispiels motiviert. Dabei werden Orientierungshilfen für die Auswahl geeigneter Integrationstechnologien. Diese werden in Abschnitt 4 verwendet, um eine praxisorientierte Bewertungssystematik zu erarbeiten.

2 Integrationsebenen und -technologien

Bei der Integration betrieblicher Softwaresysteme können mindestens die Daten-, die Anwendungs- und die Prozessebene unterschieden werden [BFGH02]. Gewachsene Anwendungslandschaften sind häufig durch eine proprietäre Punkt-zu-Punkt-Integration der Softwaresysteme auf den unterschiedlichen Ebenen gekennzeichnet. Die dabei praktizierte Ad-hoc-Integration wird durch systemnahe Entwicklungswerkzeuge unterstützt, die dem Entwickler dieselbe „Basis-Middleware“ zur Verfügung stellen, die auch Grundlage vieler fortgeschrittener Middleware-Technologien darstellt. Zu der Basis-Middleware zählen u.a.:

- Protokolle für die Realisierung der synchronen Kommunikation zwischen lokalen und verteilten Prozessen, z. B. Remote Procedure Calls (RPC) und Remote Methode Invocation (RMI),
- Messaging-Dienste für die asynchrone Kommunikation zwischen verteilten Prozessen und Anwendungen, z. B. JMS, MSMQ(T), MQ Series und Oracle Advanced Queueing,
- Frameworks für die Nutzung von Komponenten und Objekten in verteilten Systemen, z. B. CORBA, JavaBeans, DCOM/COM+ und .NET sowie
- Programmierschittstellen für den standardisierten Zugriff auf DBMS, z. B. ODBC, JDBC, OLE DB und ADO.NET.

Proprietäre Lösungen sind jedoch meist durch eine fehlende Durchgängigkeit, verbunden mit einem stetig wachsenden Integrationsaufwand, gekennzeichnet, so dass sich bald und bei fortschreitender Integration immer drängender die Frage nach dem Einsatz fortgeschrittener Integrationswerkzeuge und -technologien stellt.

Zur systematischen Vorbereitung dieses Themas werden in den Abschnitten 2.1, 2.2 und 2.3 die Integrationsebenen eingeführt und dafür typische Integrationsaufgaben und -tech-

nologien erläutert.

2.1 Datenintegration

Die Integration betrieblicher Daten hat das Ziel, die datenbankübergreifende Konsistenz der Daten für die zu integrierenden operativen Systeme sicherzustellen und/oder einen zentralen Datenbestand für das betriebliche Analyse- und Berichtswesen aufzubauen. Ein naheliegender Ansatz zur Lösung dieser Aufgaben ist die Einführung eines unternehmensweiten Datenschemas, das in einem zentralen Datenbankserver verwaltet wird. Die Schwierigkeit besteht nun darin, die Schemata und Daten der bestehenden Anwendungen mit einem zentralen (nicht notwendigerweise „an einem Ort zentralistisch“ verwalteten) Datenbestand konsistent zu halten. Sieht man von der Möglichkeit einer unmittelbaren Integration durch Re-Implementierung der Anwendungen ab, so muss die Integration der Datenbestände nachträglich erfolgen. Dabei werden zwei Arten von Integrationsansätzen unterschieden:

1. Bei der *abfragegetriebenen* Datenintegration werden die Daten erst zum Zeitpunkt einer Anfrage von den einzelnen Datenquellen angefordert. Integrationswerkzeuge und Datenquellen müssen eine verteilte Anfrageverarbeitung und verteilte Datenbanktransaktionen unterstützen. Ein Gesamtergebnis wird durch Transformation und Zusammenführung der Teilergebnisse in ein einheitliches Datenschema erzeugt.
2. Bei der *vorausschauenden* Datenintegration werden die Daten in regelmäßigen Abständen in einer zentralen Datenbank zusammengeführt. Datentransformation und -integration und das Datenqualitätsmanagement (DQM) stellen hier die Schlüsselfeatures dar, die durch Integrationswerkzeuge unterstützt werden müssen.

Der letztgenannte Ansatz wird i. A. verwendet, um in einem sogenannten Data Warehouse (DWH) [BG04] den gesamten für Analyse- und Berichtsaufgaben relevanten Datenbestand zu integrieren. Mit Hilfe von Extraction-Transformation-Loading-(ETL-)Werkzeugen werden Daten periodisch aus den Quellsystemen extrahiert, in eine einheitliche Darstellung transformiert und anschließend in das DWH geladen. Während der Transformationsphase erfolgt die Identifikation und Zusammenführung von redundanten Datensätzen. Im Rahmen des ETL-Prozesses können darüber hinaus auch weitere Maßnahmen für die Qualitätsverbesserung des integrierten Datenbestandes stattfinden [Hin01].

Neben einfachen, mit dem jeweiligen DBMS mitgelieferten ETL-Werkzeugen, wie z. B. Oracles SQL*Loader oder Microsofts Data Transformation Services, gibt es auch komplette Suiten, z. B. Informatica Powercenter oder Ascential Data Stage, die eine große Auswahl an Adaptern für die Extraktion der Daten aus unterschiedlichen Anwendungssystemen bereitstellen. Darüber hinaus stellen solche Suiten in der Regel Notationen bereit, um komplexe Datenbewirtschaftungsprozesse visuell zu modellieren.

Für die abfragegetriebene Datenintegration existiert bereits seit Längerem eine Vielzahl von föderierten Architekturvarianten [SL90]. Föderierte Architekturen weisen i. A. eine Komponente auf, der die Rolle eines Mediators zugeordnet ist. Seine Aufgabe besteht darin, das föderierte Datenschema zu verwalten sowie die verteilte Anfrageverarbeitung

durchzuführen. Da in föderierten Systemen die Datenintegration unmittelbar während der Anfrageverarbeitung erfolgt, wird in der Regel auf die Identifizierung redundanter Daten verzichtet. Bei Vorhandensein überlappender Datenbereiche wird eine eindeutige Datenführerschaft bzw. ein „führendes System“ definiert, so dass der kritische Datenbereich nur über das führende System (z. B. eine Datenbank) abgefragt wird.

Viele DBMS-Anbieter ermöglichen es, Daten aus externen Datenquellen als pseudo-lokale Tabellen zu integrieren. Durch die Definition von Verweisen (Database Links) können Sichten auf andere Datenbanken in das Schema der lokalen Datenbank eingefügt werden, wodurch Benutzer in die Lage versetzt werden, Zugriffe über mehrere Datenbanken durchzuführen. Dieser Ansatz unterstützt i. A. die verteilte Anfrageverarbeitung und verteilte Transaktionen. Beispiele in verbreiteten DBMS sind: Oracle Database Link, Oracle Transparent Gateway, IBM Distributed Facility, Microsoft SQL Server 2000 Linked Server und andere.

Für Fälle, in denen die Performance der Anfragen eine wichtige Rolle spielt oder die Verfügbarkeit der Remote-Datenbanken zwischenzeitlich unterbrochen sein kann, können Replikationstechnologien verwendet werden. Während Database Links eine synchrone und ETL-Methoden eine asynchrone Datenintegration ermöglichen, realisieren Replikationsmechanismen je nach Strategie einen synchronen *oder* asynchronen Abgleich zwischen den verknüpften Datenbanken [NH02].

Bei der Realisierung komplexer Szenarien können sich diese Technologien als unzureichend erweisen, zumal wenn Systeme beteiligt sind, die keinen direkten Zugriff auf ihre Datenbestände erlauben. In diesen Fällen ist es notwendig, z. B. auf Basis von EAI-Werkzeugen (Enterprise Application Integration) individuelle Adapter zu implementieren, mit denen die Daten über Schnittstellen des Anwendungssystems zugreifbar gemacht werden.

2.2 Anwendungsintegration

Anwendungsintegration führt man durch, um

- Teile der vorhandenen Funktionalität der zu integrierenden Applikationen wiederzuverwenden (funktionale Integration),
- die anwendungsübergreifende Datenkonsistenz sicherzustellen und/oder
- die Benutzungsoberflächen der Systeme zu vereinheitlichen (GUI-Integration).

Bei einer funktionalen Integrationslösung stellen die integrierten Systeme Dienste bereit, die von anderen Systemen genutzt werden können. Der Nachrichtenaustausch zwischen den Systemen kann dabei synchron oder asynchron erfolgen. Die synchrone Kommunikation setzt einen hohen Verfügbarkeitsgrad der Diensteanbieter voraus, was z. B. aufgrund von Offline-Zeiten nicht immer zugesichert werden kann. Unter solchen Randbedingungen bietet sich die Verwendung asynchroner Kommunikation als pragmatische Lösung für die Erhöhung der Verfügbarkeit einer Systemlandschaft an.

Anwendungsintegration zur Sicherstellung der Datenkonsistenz kann als Sonderfall der funktionalen Integration wie auch der Datenintegration (vgl. Abschnitt 2.1) angesehen werden. In diesem Fall werden durch den Aufruf von Diensten in integrierten Systemen Datenänderungen in mehrere Anwendungen übernommen. Dies erfordert in der Regel den Einsatz von Transaktionsmechanismen.

Für eine erfolgreiche GUI-Integration ist eine saubere Trennung der Geschäftslogik von der Benutzungsschnittstelle entscheidend. Die Geschäftslogik sollte idealerweise in Form von Diensten definiert sein. Integrierte Benutzungsschnittstellen können im Rahmen eines Web-Portals oder mit Desktop-Technologien, wie z. B. MFC, Qt, Java Swing oder .NET Windows.Forms, realisiert werden.

Eine der Basistechnologien für die Integration auf Anwendungsebene sind die Applikationsserver (z. B. CORBA, J2EE, .NET). Diese bieten eine Laufzeitumgebung für ein anwendungsübergreifendes Objektmodell sowie folgende Dienste an:

- Funktionen zum Auffinden, Instanziieren und Aktivieren der gewünschten Dienste in verteilten Systemen,
- Dienste für die Bewältigung von Leistungsspitzen durch Connection Multiplexing, Load Balancing und Clustering sowie
- Transaktionsmechanismen zur Sicherstellung der anwendungsübergreifenden Datenkonsistenz.

Ferner stellen sie häufig auch GUI-Bibliotheken bereit. Für CORBA gibt es eine Vielzahl von kommerziellen und freien Implementierungen, wie z. B. Visibroker von Borland, ORBIX von Iona oder MICO. Auf dem CORBA-Standard basierende Applikationsserver sind für unterschiedliche Betriebsplattformen verfügbar. Beispiele für J2EE-Server sind BEA WebLogic, IBM WebSphere, Oracle OC4J und JBoss. Diese sind ebenfalls für verschiedene Rechnerplattformen erhältlich. Für das .NET-Objektmodell wird die Applikationsserver-Funktionalität zusammen mit dem Microsoft Windows 2003 Server ausgeliefert.

Applikationsserver leiten Nachrichten zwischen verteilten Komponenteninstanzen weiter und koordinieren somit deren Zusammenarbeit. Problematisch ist jedoch, dass die Applikationsserver jeweils unterschiedliche Nachrichtenformate über zum Teil unterschiedlichen Transportprotokollen verwenden. So wurde mit IIOP (Internet Inter-ORB Protokoll) ein Standard eingeführt, der die Kommunikation zwischen unterschiedlichen CORBA-Implementierungen ermöglicht. Im betrieblichen Umfeld verwenden Anwendungssysteme jedoch häufig nur proprietäre Nachrichtenformate und Protokolle.

Diese Problematik wird durch EAI-Plattformen adressiert, die auszutauschende Nachrichten vom Quell- in das Zielformat umwandeln und auch die notwendigen Protokolltransformationen vornehmen. Für asynchrone Aufrufe bieten EAI-Plattformen verschiedene technische Dienste an: So wird z. B. häufig die persistente Speicherung der auf ein Ergebnis wartenden asynchronen Anfragen unterstützt, um eine Überlastung des Systems zu vermeiden. EAI-Plattformen stellen eine Vielzahl von Adaptern für Standard-Anwendungssysteme bereit und bieten zusätzlich Mechanismen für die Implementierung individueller Anwendungsadapter an. Beispiele für EAI-Systeme sind Tibco BusinessWorks, webMethods BusinessIntegration, IBM WebSphere und Microsoft BizTalk. Ne-

ben der beschriebenen Basisfunktionalität verfügen diese Systeme über diverse weitere Features, u.a. zur Prozess- und Portalintegration.

Die jüngste Standardtechnologie für die Anwendungsintegration sind die Web Services [LRS02, ACKM04], die im Rahmen von Service-Orientierten Architekturen (SOA) genutzt werden können. Mit SOAP (Simple Object Access Protocol) als Kommunikationsprotokoll über Transportprotokollen wie HTTP oder SMTP steht eine Alternative für IIOP bereit, der die Integration von Systemen in heterogenen Umgebungen ermöglicht. Mit WSDL (Web Services Description Language) und UDDI (Universal Description, Discovery, and Integration) können Dienste beschrieben, verwaltet und entdeckt werden.

Da heute schon viele betriebliche Applikationen Web-Service-Technologie unterstützen, kann für die Integration auf Basis von SOA auf die Entwicklung von Adaptern und somit im Prinzip auf den Einsatz eines EAI-Systems verzichtet werden [Ley04]. Für die Verwaltung und Koordination von Nachrichten, insbesondere der asynchronen Nachrichten, muss jedoch auch weiterhin Funktionalität bereitgehalten werden, um die zuverlässige Weiterleitung von Aufrufen und deren Ergebnissen zwischen Geschäftsprozessen und ausführenden Diensten sicherzustellen. Die Business Process Execution Language for Web Services (BPEL4WS) [WB03] ermöglicht die Aggregation von Web Services. Mit Hilfe der BPEL4WS können Folgen von Web-Service-Aufrufen zu Microflows und Macroflows zusammengefasst werden. Beide repräsentieren sich nach außen als eigenständige Web Services. Microflows führen atomare Funktionseinheiten aus und ermöglichen damit die Implementierung einer Transaktionsunterstützung im SOA-Umfeld. Macroflows bieten darüber hinaus die Möglichkeit, langlaufende Prozesse (insb. Workflows mit Benutzerinteraktion) mit persistierten Zuständen zu implementieren. Der SOA-Ansatz findet breiten Anklang in der Industrie und wird durch viele große Softwarehersteller wie IBM, Microsoft, SAP und BEA unterstützt.

2.3 Prozessintegration

Ziel der betrieblichen Prozessintegration ist die Automatisierung von Geschäftsprozessen eines Unternehmens [STS+04]. Geschäftsprozesse werden als Folge von Aktivitäten definiert, die nacheinander, parallel oder nur unter bestimmten Bedingungen ausgeführt werden. Es kann zwischen automatischen und manuellen Aktivitäten unterschieden werden. Automatische Aktivitäten werden durch ein Anwendungssystem autonom ohne Benutzereingriffe ausgeführt. Manuelle Aktivitäten bedürfen einer Interaktion mit den Benutzern. Es kann sich dabei z. B. um eine abschließende Prüfung der Angaben in einer Maske oder einen durchzuführenden Telefonanruf handeln. Integrationswerkzeuge müssen daher Konzepte zur Entgegennahme von Benutzerinteraktionen anbieten. Auch müssen im Rahmen der Geschäftsprozesse häufig langlaufende Transaktionen realisiert werden, die sich aus einer Folge von aufeinander aufbauenden Teil-Transaktionen zusammensetzen. Im Rahmen der Prozessintegration muss in der Regel parallel zum Aktivitätenfluss auch ein Fluss von Daten und Dokumenten innerhalb des Geschäftsprozesses unterstützt werden.

Bei langlaufenden Geschäftsprozessen muss die Prozesssteuerungssoftware die Möglichkeit bieten, unterbrochene Prozesse, die auf Benutzereingaben oder Rückgabewerte von asynchron aufgerufenen Diensten warten, zwischenzuspeichern und nach nach Beendigung der wartenden Aktivität weiterzuführen. Ferner ist die Bereitstellung von Informationen über den aktuellen Bearbeitungszustand von Geschäftsprozessen sinnvoll, und es ist wünschenswert, einem Benutzer alle noch von ihm zu bearbeitenden Aktivitäten aufzuzeigen. Integrationswerkzeuge sollten zudem eine einfache Anpassung integrierter Prozesse an Veränderungen im Unternehmen ermöglichen.

Für die Prozessintegration können z. B. Workflow-Managementsysteme (WfMS) eingesetzt werden. WfMS steuern die Ausführung von Aktivitäten sowie den Daten- und Dokumentenfluss für einzelne Prozessinstanzen. Diese Systeme unterstützen schwerpunktmäßig interaktive Arbeitsabläufe, bei denen unstrukturierte Dokumente verarbeitet werden müssen. Das WfMS verwaltet dabei zentral die Workflow-Schema- und -Instanzdaten sowie die zugehörigen Akteure und Aktionen. Beispiele für WfMS sind FileNet, Staffware, SER Doxis und Carnot. Auch heutige EAI-Plattformen werden zunehmend durch Funktionalitäten zur Prozessintegration ergänzt. Der Schwerpunkt dieser Systeme liegt jedoch eher auf der Unterstützung vollautomatisierter Geschäftsprozesse ohne Benutzerinteraktion, z. B. im Bereich der Abwicklung automatisierter Bestellvorgänge.

Auf Basis der Standards BPEL4WS, WS-Coordination [WC03] und WS-Transactions [WT02] werden derzeit von unterschiedlichen Herstellern Plattformen für die Prozessintegration auf der Basis von SOA implementiert, die die Aggregation von Web Services zu sogenannten Macroflows ermöglichen. Macroflows unterstützen die Modellierung von Geschäftsprozessen mit synchronen und asynchronen Aufrufen, manuellen Aktivitäten und langlaufenden Transaktionen. Für die Implementierung der Rollback-Funktionalität ist in diesem Zusammenhang ein Konzept auf der Basis von kompensierenden Aktivitäten eingeführt worden. Diese Funktionalitäten werden zukünftig in Integrationsprodukten von z. B. IBM, Oracle (Collaxa) oder Microsoft zu finden sein.

3 Eigenschaften praktischer Integrationsszenarien

Oft lassen sich die in konkreten Projekten auftretenden Integrationsszenarien nicht genau einer der Integrationsebenen zuordnen. In diesem Abschnitt werden daher als weitere charakteristische Merkmale praktischer Integrationsszenarien

- das genutzte Kommunikationsparadigma (Abschnitt 3.1),
- die Strukturiertheit der Daten (Abschnitt 3.2),
- die Datenredundanz und Anforderungen an die Aktualität der Daten (Abschnitt 3.3),
- der Automatisierungsgrad der Geschäftsprozesse (Abschnitt 3.4),
- die Transaktionsunterstützung (Abschnitt 3.5) sowie
- weitere Rahmenbedingungen, hier spezifisch die Anzahl und Heterogenität der zu integrierenden Systeme (Abschnitt 3.6),

vorgeschlagen und diskutiert, die sich in verschiedenen Integrationsprojekten als wesentliche Aspekte erwiesen haben. In den einzelnen Abschnitten wird zur Erläuterung auf folgendes Fallbeispiel Bezug genommen:

In einem Unternehmen sollen ein ERP-, ein CRM- und ein Abrechnungssystem integriert werden, wobei alle drei Systeme jeweils Produktdaten und Stammdaten verwalten. Das ERP-System ist für CRM-Anwendungen nicht ausreichend verfügbar (geplante Offline-Zeiten), CRM-Anwendungen sollen den Außendienstmitarbeitern auch unterwegs ohne eine Datenverbindung zu den operativen Systemen bereitstehen. Ferner dürfen nur Produkte und Dienste verkauft werden, die auch abgerechnet werden können.

3.1 Kommunikationsparadigma

Bei der Auswahl eines Kommunikationsparadigmas für ein Integrationsszenario muss beachtet werden, dass die synchrone Kommunikation eine hohe Verfügbarkeit bei den zu integrierenden Systemen voraussetzt. Synchrone Kommunikation ermöglicht eine enge Kopplung der Anwendungssysteme, was unter Umständen die Performance der Integrationslösung steigern kann. Die asynchrone Kommunikation ermöglicht hingegen eine fehlertolerante Integration, so dass dieser Ansatz insbesondere für die Integration von Systemen mit geringer Verfügbarkeit geeignet ist.

Für die Kommunikation mit Altsystemen muss häufig auf eine dateibasierte Form der *asynchronen* Kommunikation zurückgegriffen werden. Die Quellanwendung schreibt ihre Daten als Datei in ein Austauschverzeichnis. Diese Dateien werden in regelmäßigen Abständen von der Zielanwendung bzw. einem Importskript aus dem Austauschverzeichnis abgeholt („push and poll“). Durch EAI-Plattformen können solche Systeme über individuell konfigurierte File-Adapter angekoppelt werden.

Verfügt eine Anwendung über ein Remote API oder handelt es sich um eine datenbankgestützte Anwendung, so gibt es unterschiedliche Mechanismen für die Realisierung einer *synchronen* Kommunikation: Die Anwendung kann z.B. über einen proprietären Adapter der Integrationsplattform angesprochen werden. Alternativ dazu kann das Remote API als Web Service z.B. im Rahmen einer SOA angesprochen werden. Werkzeuge für Java und .NET-Sprachen ermöglichen hierfür die Generierung einer Web-Service-Schnittstelle inkl. der WSDL-Beschreibungen. Bei einer datenbankgestützten Anwendung ohne eigenes API kann die synchrone Kommunikation durch direkte Datenbankzugriffe implementiert werden. Der zugehörige individuell konfigurierte Datenbankadapter kann seine Funktionsschnittstelle plattformseitig ebenfalls über Web Services anbieten.

Mit der Anzahl der integrierten Systeme steigt auch der Kommunikationsaufwand an. In Fällen, bei denen die gleiche Nachricht an mehrere Empfänger verschickt wird, kann das *Publish-Subscribe*-Pattern eingesetzt werden, das auch in EAI-Plattformen mit Message-Broker-Architektur systemintern für die Nachrichtenverteilung genutzt wird. Dabei melden sich nur interessierte Anwendungen beim Verteiler bestimmter, von der Plattform definierter Nachrichten an. Nützlich ist dieses Verfahren z.B. bei komplexen Koordinierungsaufgaben, wie dem oben erwähnten Stammdatenabgleich.

Im Fallbeispiel fällt die Entscheidung zugunsten asynchroner Kommunikation, damit kundenbezogene Geschäftsprozesse mit geringen Auswirkungen durch Nichtverfügbarkeit des ERP- oder Abrechnungssystems durchgeführt werden können. Sämtliche Änderungen in den Stammdaten werden asynchron beauftragt und quittiert. Falls Änderungen nicht durch das ERP-System akzeptiert werden, so wird dies dem CRM-System asynchron angezeigt, woraufhin dort die Änderungen im Rahmen einer manuellen Aktivität auf der Basis der Arbeitskorb-Metapher korrigiert werden.

3.2 Strukturiertheit der Daten

Ein wichtiger Aspekt der Integration ist die Struktur bzw. der Strukturierungsgrad der zu integrierenden Daten. Für Geschäftsprozesse können strukturierte, semi-strukturierte und unstrukturierte Daten relevant sein. Zur Optimierung der Geschäftsprozesse ist es sinnvoll, den Anteil der strukturierten Daten zu erhöhen, weil strukturierte und teilweise auch semi-strukturierte Daten automatisiert verarbeitet werden können. Unstrukturierte Daten erfordern i. A. eine manuelle Bearbeitung durch einen Benutzer, um z.B. den weiteren Verlauf eines Geschäftsprozesses nach einer Interpretation des Dokuments zu beeinflussen.

In Datenbanken abgelegte Daten sind in der Regel strukturiert. Bei der Integration von Dateien können hingegen alle genannten Varianten vorkommen: Typisierte Dateien (zur Aufnahme von Mengen von Datensätzen gleichen Typs, z. B. CSV) beinhalten strukturierte Daten, die sich innerhalb einer Integrationsplattform einfach verarbeiten lassen. Semistrukturierte Daten haben eine komplexe Struktur, wie z. B. EDI-Formate. Heute werden semi-strukturierte Daten bevorzugt in XML-Formaten verarbeitet, weil dies die Nutzung entsprechender XML-Bibliotheken (z. B. SAX) ermöglicht und der individuelle Entwicklungsaufwand für das Einlesen und Parsen der Dateien damit weitgehend entfällt. Auch verschiedene Integrationsplattformen nutzen XML als Nachrichtenformat, so dass Synergieeffekte (Wegfallen von Konvertierung) entstehen können. Das automatisierte Verarbeiten von Dokumenten in proprietären semi-strukturierten Formaten, wie z.B. Word- oder Excel-Dateien, erfordert den Einsatz spezifischer Adapter, so dass diese Formate häufig wie alle unstrukturierte Daten nur als Ganzes weitergeleitet und archiviert werden.

Im Fallbeispiel sind sowohl unstrukturierte Dokumente (Auftragsdokumente, Auftragsbestätigungen, Störungsmeldungen) als auch strukturierte und semi-strukturierte Dokumente (Kunden, Adressen, Bankverbindungen, Konten, Rechnungen) zu verwalten und zwischen den zu integrierenden Systemen auszutauschen.

3.3 Datenredundanz und Anforderungen an die Aktualität der Daten

Häufig kommt es vor, dass die zu integrierenden Anwendungen überlappende Datenbestände aufweisen. Sollen diese Daten in einem konsistenten Zustand gehalten werden, so müssen Verfahren zur Behandlung der Datenredundanz entwickelt werden. Hierbei müssen folgende Aspekte berücksichtigt werden:

1. Häufigkeit der Datenänderungen,
2. Unterscheidung zwischen Datenproduzenten und -nutzern,
3. Anforderungen an die Aktualität der Daten in den einzelnen Systemen sowie
4. beim Datenabgleich entstehender Prozess- und Kommunikationsaufwand.

Für die Auflösung der Datenredundanz gibt es zwei grundsätzliche Strategien: Datenführerschaft und Datenintegration im engeren Sinne. Bei der Datenführerschaft wird der gesamte Datenbestand des integrierten Systems in disjunkte Datenbereiche aufgeteilt. Anschliessend wird für jeden Datenbereich genau ein führendes Teilsystem ausgewählt, das für die Konsistenz der ihm zugeordneten Bereiche verantwortlich ist. Die restlichen Teilsysteme haben nur lesenden Zugriff auf diesen Datenbereich. Bei der Datenintegration im engeren Sinne wird keine klare Verantwortlichkeit für überlappende Datenbereiche definiert, so dass beim Zugriff auf solche Bereiche die redundanten Datensätze identifiziert und verknüpft werden müssen.

Die Entscheidung für den einen oder anderen Ansatz zur Auflösung der Datenredundanz wird in der Praxis häufig durch die Anforderungen an die Aktualität der Daten grundlegend beeinflusst. Falls die Daten in „Echtzeit“ vorliegen müssen, muss es für den gesamten Datenbestand eindeutige Führerschaften geben, weil die Identifikation redundanter Daten und ihre Integration i. A. nicht in Echtzeit durchgeführt werden kann.

Im Fallbeispiel stellen die Kundenstammdaten einen besonderen Wert dar. Sie sollen aus historischen Gründen vom ERP-System führend verwaltet werden. Datenführerschaft bedeutet hier, dass das ERP-System die gültige Sicht der Kundenstammdaten verwaltet und jede Änderung der Kundenstammdaten über das ERP-System autorisiert werden muss. Zusätzlich müssen Änderungen der Kundenstammdaten jedoch jederzeit beauftragt werden können, insbesondere in Offline-Zeiten des ERP-Systems und im mobilen Einsatz des CRM-Systems. Auftragsprozesse werden ebenfalls vom (ggf. mobil offline arbeitenden) CRM-System initiiert. Diese können jedoch nur abgeschlossen werden, wenn die beauftragten Dienste und Produkte auch abgerechnet werden können. Diese Entscheidung kann nur vom Abrechnungssystem getroffen werden.

3.4 Automatisierungsgrad der Geschäftsprozesse

Zwei komplementäre Aspekte im Bereich der Prozessintegration sind der Grad der Automatisierbarkeit von Prozessaktivitäten und der Grad der benötigten Benutzerinteraktion. Ein Ziel der Optimierung von Geschäftsprozessen ist typischerweise, den Anteil automatisierter Bearbeitungsschritte zu erhöhen und damit den Anteil manueller Bearbeitungsschritte zu verringern. Ein hoher Automatisierungsgrad ermöglicht, auf synchrone Kopplungen mit einem online wartenden Benutzer zu verzichten. Im Idealfall führen nur Ausnahmen (Exceptions) im Prozess zur Benachrichtigung eines Prozessverantwortlichen, der dann ggf. interaktiv in den Prozess eingreift.

In vielen Geschäftsprozessen kann jedoch nicht auf Benutzerinteraktionen verzichtet werden. Hierzu zählt u. a. das Prüfen von wichtigen Zwischenergebnissen in ansonsten au-

tomatisierten Prozessen, die Erfassung zusätzlicher Daten über Bildschirmformulare oder die manuelle Bearbeitung von Dokumenten, wie z.B. Word-Dateien.

Zur Behandlung von interaktiven Aktivitäten kommt häufig die Arbeitskorb-Metapher zum Einsatz, die es ermöglicht, Synchronisationspunkte bei der zum großen Teil asynchronen Ausführung von Geschäftsprozessen zu definieren. Ist nach der Ausführung von automatisierten Aktivitäten eine Bearbeitung durch den Benutzer notwendig, so wird für den Benutzer (respektive seine Rolle) in dem Arbeitskorb eine manuelle Aktivität abgelegt. Nach der interaktiven Bearbeitung dieser Aktivität durch den Anwender startet die Prozesssteuerung die nachfolgenden Prozessschritte. Ein Integrationswerkzeug, mit dem interaktive Prozesse abgebildet werden sollen, muss ein Framework zur Entwicklung und Einbindung von entsprechenden Formularen bereitstellen. Weiterhin sind clientseitige Schnittstellen zu Anwendungen notwendig, mit denen bestimmte Aktivitäten, wie z.B. Grafikbearbeitung, durchgeführt werden.

Im Fallbeispiel enthalten die Geschäftsprozesse (z.B. der Auftragsprozess) sowohl manuelle (z.B. Erfassen von Produktdetails, Identifizieren von Kunden) als auch automatische (z.B. Bonitätsprüfung, Prüfung der technischen Realisierbarkeit) Aktivitäten.

3.5 Transaktionsunterstützung

Ziel der Prozessintegration ist es, getrennt ablaufende Einzelaktivitäten oder Microflows zu größeren Ablaufeinheiten oder Macroflows zusammenzuführen. Für die Koordination derart integrierter Prozesse ist die Mitführung bestimmter Zustandsinformationen erforderlich. In vielen innerbetrieblichen Arbeitsabläufen wird beispielsweise der Bearbeitungsstand manueller Prozesse in der Regel über Auftrags- oder Laufzettel erfasst und fortgeschrieben.

Bei der automatisierten Steuerung integrierter Prozesse muss die eingesetzte Integrationslösung u. a. folgende Arten von Status- bzw. Zustandsinformationen mitführen:

- den globalen Prozesszustand (z.B. aktiv, wartet, beendet), eine Referenz auf die aktuelle Bearbeitungsinstanz (User, System, Dienst) und Referenzen auf die aktuell in Bearbeitung befindliche und die letzte erfolgreich abgeschlossene Prozessaktivität,
- den Status von Prozessvariablen, die sich auf die aktuelle Prozessinstanz beziehen (beliebige lokale Variablen),
- die Inhalte von Eingangsdokumenten und -nachrichten sowie prozessintern erzeugte oder veränderte Dokumente und Nachrichten und
- ggf. ein Protokoll der durchgeführten Aktivitäten und der beteiligten Akteure (Systeme, Bearbeiter) zu Zwecken der Historienfortschreibung (z.B. aus vertraglichen bzw. gesetzlichen Anforderungen).

Die Zustandsinformationen sollten durch das Integrationssystem automatisch instanziiert, initialisiert und aktualisiert werden. Zur Gewährleistung der Transaktionssicherheit der im Rahmen einer Integrationslösung bereitgestellten Prozesse ist die sichere Vorhaltung der Informationen notwendig. Gleichgültig ist hierbei, ob ein Prozess z. B. aufgrund ma-

nueller Interaktionen wie im Fallbeispiel als langlaufend zu betrachten ist, oder ob er möglicherweise auf einer extrem kurzen Zeitskala abläuft. Dem Prozessdesigner ist es zudem freigestellt, die Transaktionssicht entweder grobgranular auf den Gesamtprozess oder feingranular auf jede Einzelaktivität zu beziehen. Wichtig ist es dabei, dass die Integrationsplattform die Persistierung automatisch vornimmt und hierfür keine eigenen Aktivitäten implementiert werden müssen.

Ziel der prozessbezogenen Zustandshaltung ist es, einen Prozess für den Fall einer technischen Unterbrechung geregelt zurückzurollen und an einem Synchronisationspunkt wiederaufzunehmen. In der Regel kann dabei nicht auf hochwertige Transaktionsmechanismen wie 2-Phasen-Commit zurückgegriffen werden, da dies nur von den wenigsten Systemen unterstützt wird. Dieser Mangel wird heute pragmatisch durch geschicktes Prozessdesign (passender Einsatz kompensierender Aktivitäten) in Kombination mit organisatorischen Richtlinien für die parallele Prozessausführung, so gut es geht, behoben.

Im Fallbeispiel ist sicherzustellen, dass im Rahmen des Auftragsprozesses ein Auftrag (initiiert durch das CRM-System) nur komplett oder gar nicht durchgeführt wird. Eine beauftragte Leistung darf nur dann abgerechnet werden, wenn sie auch erbracht wurde. Andererseits darf auch keine beauftragte Leistung erbracht werden, die nicht abgerechnet werden kann.

3.6 Anzahl und Heterogenität der zu integrierenden Systeme

Aus praktischer Sicht bemisst sich die Komplexität eines Integrationsszenarios an der Anzahl und Verschiedenheit der von der Integration betroffenen Systeme. Wichtiges Ziel der Integrationslösung ist es dabei oftmals, eine existierende, durch individuelle Punkt-zu-Punkt-Verbindungen zwischen Anwendungen gekennzeichnete Systemwelt in eine plattformzentrische Anwendungslandschaft zu überführen.

Ein Argument für die Einführung einer Integrationsplattform ist dabei häufig die mögliche Aufwandsreduktion durch eine Reduzierung der Schnittstellenanzahl, zumal zwischen n Systemen bis zu n*(n-1)/2 Kopplungen bestehen können. In der Praxis fällt aber auf, dass bei wenigen Systemen häufig mehr Schnittstellen von und zu der Integrationsplattform zu implementieren sind als ursprünglich Punkt-zu-Punkt-Verbindungen existieren. In der Regel ist nämlich nur ein Teil aller möglichen Punkt-zu-Punkt-Schnittstellen sinnvoll und daher auch realisiert. Als Voraussetzung für die Integration müssen aber Kopplungen aller Systeme zu der zentralen Plattform erstellt werden. Das Argument der bloßen Aufwandsreduktion ist somit häufig erst ab einer größeren Anzahl von Systemen wirksam. Dennoch überwiegen die Vorteile der Plattformkopplung den zusätzlichen Aufwand, sofern mit der plattformgestützten Systemanbindung und Prozessabwicklung Plattformdienste wie z. B. Überwachung, Transaktionen oder Sicherheitsmechanismen genutzt werden können, die einen echten Mehrwert gegenüber den originären Systemlandschaften darstellen. Daher kann eine plattformgestützte Integrationslösung bereits bei nur zwei zu koppelnden Systemen sinnvoll sein, insbesondere wenn diese Systeme komplex sind und über mehrere gegenseitige Schnittstellenanbindungen unterschiedlicher Funktionsmodule verfügen. Im

Fallbeispiel sind mit ERP-System, CRM-System und Abrechnungssystem drei komplexe Systeme zu integrieren, die über mehrere gegenseitige Schnittstellen verfügen.

In den allermeisten Fällen besitzen die zu koppelnden Systeme unterschiedliche Datenmodelle, die im Rahmen einer Systemkopplung ineinander transformiert werden müssen. Hierbei können aus Sicht der Datenintegration zwei Strategien verfolgt werden: Im einfacheren Fall implementiert die Integrationsplattform eine direkte Abbildung der Datenschnittstellen zwischen zwei Anwendungen. Diese Konstruktion ist als plattformgestützte Punkt-zu-Punkt-Kopplung anzusehen, die immerhin die Vorteile der Nutzbarkeit der o.g. Plattformdienste bietet. Im komplexeren Fall wird zwischen allen beteiligten Anwendungen ein gemeinsames (kanonisches) Datenmodell vereinbart, das auf einem Standardinformationsmodell (Ontologie) basieren kann (z. B. CIM für Energiemanagementsysteme [IEC03]). Die zentrale Integrationsplattform übernimmt in diesem Fall die Transformationen zwischen den individuellen Datenmodellen und dem kanonischen Datenmodell.

Werden sehr viele Systeme integriert, so muss beachtet werden, dass mit der Anzahl der Systeme sich in der Regel auch deren Schnittstellentechnologien immer stärker diversifizieren. Entsprechend mächtig und vielfältig müssen die von der Integrationsplattform bereitgestellten Adapter bzw. Adapter-Entwicklungswerkzeuge sein. Zur Vermeidung proprietärer Schnittstellenlösungen kann die Strategie verfolgt werden, Standardtechnologien einzusetzen. Hier ist in den vergangenen Jahren die Technologie der Web Services in den Mittelpunkt gerückt, mit der eine plattform- und sprachenunabhängige Anwendungskonnektivität realisiert werden kann. Dabei ist allerdings zu bedenken, dass die Web-Service-Unterstützung eines Legacy-Systems zunächst selbst entwickelt werden will. Die Mechanismen von Integrationsplattformen sehen hierzu vor, eine Anwendung zunächst über einen plattformproprietären Adapter anzukoppeln, der dann seinerseits automatisiert eine Web-Service-Schnittstelle bereitstellt. Man muss also logisch zwischen der *proprietären* Plattformanbindung und der *standardisierten* Verknüpfung von Anwendungen über Web Services unterscheiden.

Die gewissermaßen hohe Schule der Integration vieler Anwendungen besteht dann darin, auf Basis eines kanonischen Datenmodells ein gemeinsames Kommunikationsmodell unter Nutzung von Web Services zu etablieren. In diesem Modell nimmt die Integrationsplattform die Rolle eines Datenbus' ein, auf dem alle Anwendungen ihre Daten publizieren und sie anderen Anwendungen zur Verfügung stellen.

4 Auswahl von Integrationstechnologien

Die Entscheidung für eine konkrete Integrationsplattform ist abhängig von den charakteristischen Merkmalen des Integrationsszenarios. In diesem Artikel werden die Werkzeugklassen

- Datawarehouses und ETL-Werkzeuge,
- FDBS und DBMS-Gateways,
- Datenreplikation,

- Applikationsserver,
- klassische EAI-Werkzeuge,
- Workflow-Management-Systeme,
- SOA und BPEL4WS sowie
- klassische Middleware-Technologien

betrachtet. Die Prüfung der charakteristischen Merkmale in der Analysephase eines Integrationsprojekts gibt entscheidende Hinweise für die Auswahl einer Klasse. Erst im Anschluss an diesen Auswahlprozess sollten konkrete Integrationsprodukte untersucht werden, und erst dann wird die Unterstützung differenzierter Basistechnologien (Messagebus-Architektur, Objektmodell, ...) relevant.

	ETL/DWH	FDBS	Replikation	Appl. Server	klass. EAI	klass. WfMS	SOA + BPEL	Basis-Middl.	propr. Integr.
3.1 Kommunikation									
Push-and-Poll	+	–	–	o	++	o	–	o	+
Synchron	–	++	o	++	++	+	++	++	++
Asynchron	++	n. a.	++	++	++	++	++	++	o
Publish-Subscribe	–	n. a.	++	++	++	o	o	+(+)	–
3.2 Daten									
Strukturiert	++	++	++	++	++	o	o	++	++
Semi-strukt. (XML)	+	(+)	(+)	++	++	o	++	o	o
Unstrukturiert	o	(o)	(o)	o	o	++	(o)	o	o
3.3 Redundanz									
Datenführerschaft	n. a.	++	++	o	+	n. a.	–	o	o
Merging/DQM	++	–	o	–	+	o	–	o	o
Hohe Aktualität	–	++	o	++	++	o	+	+	+
3.4 Automatisierung									
Realtime-Aktivitäten	–	++	+	++	++	o	+	++	++
Batch-Aktivitäten	++	+	++	o	++	o	–	o	o
Manuelle Aktivitäten	–	+	+	++	o	++	(+)	–	o
3.5 Transaktionen									
ACID	++	+	o	++	+	–	(+)	o	–
Langläufer	n. a.	o	o	–	o	++	o	–	o
Zustandshaltung	n. a.	+	–	+	+	++	o	o	o
3.6 Anwendungssysteme									
Vielzahl	++	+	+(+)	++	++	++	++	++	–
Heterogenität	++	o	+	(+)	++	+	++	+	–
Globaler Datenbus	n. a.	+	o	+	+	–	++	–	–

Tabelle 1: Eignung von Integrationstechnologien für typische Merkmale von Integrationsszenarien

Tabelle 1 gibt einen Überblick über die verschiedenen Werkzeugklassen und zeigt, welche Integrationscharakteristika diese unterstützen. Zum Vergleich wird in der Tabelle zusätzlich die proprietäre Integration über individuelle Punkt-zu-Punkt-Schnittstellen den übrigen Integrationsansätzen gegenübergestellt.

ETL/DWH Datawarehouses bzw. ETL-Werkzeuge dienen im Wesentlichen der vorausschauenden Integration zur Schaffung einer Analyseplattform auf konsolidierten Daten. Eine Vielzahl heterogener Datenquellen kann integriert werden. Für OLTP-Zwecke und automatisierte Geschäftsprozesse sind diese Ansätze zwar weniger geeignet, sie können aber für die initiale Integration externer Datenbestände in eine führende Datenbank genutzt werden, weil sie die Möglichkeit zur qualitätsgesicherten Zusammenführung überlappender Datenbestände bieten.

FDBS und Replikation FDBS sind zur Automatisierung von Geschäftsprozessen und für OLTP-Zwecke besser geeignet, wobei hier die Integration ebenfalls auf Datenschemaebene erfolgt. Während in FDBS Daten „online“ zusammengeführt werden, geschieht dies bei Replikation im Vorfeld, wodurch eine bessere Verfügbarkeit des Gesamtsystems erreicht werden kann. Sind Systeme zu integrieren, deren Datenschemata nicht offenliegen, so eignen sich diese Ansätze schlecht.

Applikationsserver Applikationsserver stellen ähnlich wie klassische Middleware eine Plattform zur Integration auf Anwendungsebene dar und bieten (insb. im J2EE-Umfeld) eine Vielzahl von technischen Diensten an. Die konkrete Anbindung von Systemen ist jedoch in der Regel individuell zu programmieren.

EAI-Werkzeuge EAI-Produkte hingegen bieten neben der Integrationsplattform in der Regel auch Adaptoren für populäre Drittsysteme (etwa Siebel, SAP). EAI-Werkzeuge ermöglichen die Integration existierender Anwendungen auf Daten-, Anwendungs- und Prozessebene. Bei der Integration mittels eines EAI-Werkzeugs ist typischweise für jede Anwendung, die integriert werden soll, ein Adapter zu erstellen, der die Transformation der Daten und der entfernten Systemaufrufe realisiert. Diese Adapter basieren in der Regel auf proprietären Klassenbibliotheken.

Neben der reinen Transformation der Daten und Systemaufrufe stellen die meisten EAI-Werkzeuge Funktionalität zur Lastverteilung, Ausfallsicherheit und zum Monitoring der EAI-Aktivitäten zur Verfügung. Dabei wird der Bearbeitungsstand der einzelnen Aktivitäten vom EAI-Werkzeug verwaltet.

Mit wachsender Popularität der Web Services haben auch die EAI-Anbieter das Potential dieser Technologie erkannt. So bieten heute viele EAI-Werkzeuganbieter die Integration nicht mehr nur mittels proprietärer Adapter an, sondern ermöglichen es den Integratoren, Anwendungen über Web Services an das EAI-Werkzeug anzubinden.

EAI-Werkzeuge eignen sich in erster Linie für vollautomatisierte Arbeitsabläufe, die auf

strukturierten Daten arbeiten. Die Unterstützung für manuelle Aktivitäten und unstrukturierte Daten ist derzeit nur rudimentär vorhanden.

Workflowmanagementsysteme WfMS haben gerade im Bereich manueller Aktivitäten und unstrukturierter Daten ihre Stärken, wenngleich auch sie zunehmend Mechanismen zur Durchführung automatischer Aktivitäten mitbringen. Bei diesen Systemen ist weiterhin die Unterstützung des Prozessmanagements, insbesondere für langlaufende Prozesse, am besten ausgeprägt. WfMS konzentrieren sich auf Arbeitsabläufe, die unstrukturierte Dokumente verarbeiten und deren Interpretation zumeist den Eingriff eines Menschen erfordert.

SOA + BPEL4WS Web Services in Kombination mit Ausführungseinheiten für Prozessbeschreibungssprachen (etwa BPEL4WS) stellen einen vielversprechenden Ansatz dar, der die Bereiche funktionale Integration und Prozessintegration gut abdeckt. Zur Spezifikation von Geschäftsprozessen sind in den letzten Jahren eine Reihe von Standards wie BPEL4WS, BMPL und XPDL entstanden, wobei es sich um XML-basierte Sprachen handelt [WB03, ACKM04, WC03, WT02]. BPEL4WS (Business Process Execution Language for Web Services) ist speziell auf die Koordination von Web Services ausgerichtet. Die Kombination aus BPEL4WS und Web Services bietet eine Alternative zu klassischen EAI-Werkzeugen. Lediglich die Unterstützung für Transaktionen und unstrukturierte Daten ist schwächer ausgeprägt und zur Datenintegration ist dieser Ansatz weniger geeignet. Von Nachteil ist heute ebenfalls die geringe Verfügbarkeit von Systemen, die diesen Ansatz unterstützen.

Basis-Middleware und proprietäre Integration Basis-Middleware wie CORBA oder COM+ kann lediglich als Grundlage einer Integrationsplattform dienen. Da sowohl besondere Unterstützung für das Prozessmanagement als auch für das Datenmanagement fehlt, ist der Einsatz einer Basis-Middleware ohne weitere Komponenten für technische Dienste wenig sinnvoll. Gleiches gilt für die proprietäre Integration, wobei hier zusätzlich Fragen der Interprozesskommunikation, Synchronisation usw. jeweils separat zu addressieren sind.

Im Fallbeispiel wird der Unterstützung manueller Aktivitäten und unstrukturierter Dokumente ein großer Wert zugesprochen. Dies führt zu der Entscheidung, die Integrationsproblematik durch den Einsatz eines WfMS zu lösen. Da sowohl ERP- als auch CRM-System über eine integrierte WfM-Komponente verfügen, die zu unterstützenden Prozesse jedoch zum Großteil den Kunden einbeziehen, fällt die Wahl auf die im CRM-System integrierte WfM-Komponente. Zur asynchronen Kopplung zwischen CRM-, ERP- und Abrechnungssystem fällt die Wahl auf einen J2EE-Applikationsserver, der die benötigten Messaging-Dienste (JMS) bereitstellt. Synchrone Kommunikation stellt der Applikationsserver als Web Services (über HTTP) bereit.

5 Zusammenfassung und Ausblick

Dieser Beitrag führt eine praxisorientierte Bewertungssystematik für Integrationsprojekte ein. Ziel ist, die Vorauswahl von Werkzeugen und Technologien innerhalb eines nahezu unüberschaubaren Marktes von Herstellern und Produkten zu erleichtern. Der Artikel empfiehlt, die Vorauswahl einer Technologie zunächst anhand praxisrelevanter Merkmale durchzuführen. Erst im Anschluss daran sollte eine genauere Betrachtung der gewählten Technologien bzw. Werkzeugklassen hinsichtlich ihrer funktionalen Eignung erfolgen.

Nach Einführung der Integrationsebenen wurden Merkmale praktischer Integrationsszenarien herausgearbeitet. Anschliessend wurde die grundsätzliche Eignung der vorgestellten Werkzeugklassen für die einzelnen Merkmalsausprägungen auf der Basis praktischer Erfahrungen bewertet.

Die Autoren hoffen, dass die vorgestellte Systematik und die darauf basierende Bewertung in konkreten Integrationsprojekten von Nutzen ist. Über einzelne Bewertungen von Werkzeugklassen kann sicherlich noch im Detail diskutiert werden, zumal bestimmte Produkte mit ihren individuellen Stärken und Schwächen signifikant vom Durchschnitt abweichen können. Aufgrund der schnellen Entwicklung bei einzelnen Technologien, wie z. B. SOA und BPEL4WS, ist es zudem sinnvoll, die Bewertungen in bestimmten zeitlichen Abständen zu überprüfen.

Literatur

[ACKM04] Gustavo Alonso, Fabio Casati, Harumi Kuno und Vijay Machiraju. *Web Services – Concepts, Architectures and Applications*. Springer, 1. Auflage, 2004.

[BFGH02] B. Bunjes, J. Friebe, R. Götze und A. Harren. Integration von Daten, Anwendungen und Prozesse am Beispiel des Telekommunikationsunternehmens EWE TEL. *Wirtschaftsinformatik*, 44(5):415–423, 2002.

[BG04] Andreas Bauer und Holger Günzel. *Data Warehouse Systeme – Architektur, Entwicklung, Anwendung*. dpunkt, 2. Auflage, 2004.

[Hin01] Holger Hinrichs. Datenqualitätsmanagement in Data Warehouse-Umgebungen. In Springer, Hrsg., *Datenbanksysteme in Büro, Technik und Wissenschaft, 9. GI-Fachtagung BTW 2001, Oldenburg*, Seiten 187–206. A. Heuer and F. Leymann and D. Priebe, Februar 2001.

[IEC03] IEC. Energy Management System API - Part 301: Common Information Model (CIM) Base, November 2003.

[Ley04] Frank Leymann. The Influence of Web Services on Software: Potentials and Tasks. In Peter Dadam und Manfred Reichert, Hrsg., *34. Jahrestagung der Gesellschaft für Informatik e.V. (GI), Ulm, 20.-24. September 2004*, Jgg. 50 of *LNI*, Seiten 14–25. GI, 2004.

[LRS02] Frank Leymann, Dieter Roller und Marc-Thomas Schmidt. Web Services and Business Process Management. *IBM Systems Journal*, 41:198–211, 2 2002.

[NH02] H. Niemann und W. Hasselbring. Adaptive Replikationsstrategien für heterogene, autonome Informationssysteme. In G. Weber, Hrsg., *14. GI-Workshop über Grundlagen von Datenbanken*, Seiten 81–85, Fischland, 5 2002. http://wwwdb.informatik.uni-rostock.de/wsgvdb02/.

[Rit98] David Ritter. The middleware muddle. *SIGMOD Rec.*, 27(4):86–93, 1998.

[SL90] Amit P. Sheth und James A. Larson. Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. In *ACM Computing Surveys*, Jgg. 22, Seiten 183–236. ACM, 1990.

[Sto02] Michael Stonebreaker. Too Much Middleware. *ACM SIGMOD Record*, 31(1):97–106, March 2002.

[STS+04] August-Wilhelm Scheer, Oliver Thomas, Christian Seel, Gunnar Martin und Bettina Kaffai. Geschäftsprozessorientierte Software-Architekturen: Revolution auf dem Software-Markt? In Peter Dadam und Manfred Reichert, Hrsg., *34. Jahrestagung der Gesellschaft für Informatik e.V. (GI), Ulm, 20.-24. September 2004*, Jgg. 50 of *LNI*, Seiten 2–13. GI, 2004.

[WB03] WS-BPEL. Business Process Execution Language for Web Services Version 1.1. http://www-128.ibm.com/developerworks/library/ws-bpel/, May 2003.

[WC03] WS-C. Web Services Coordination. http://www-128.ibm.com/developerworks/library/ws-coor/, September 2003.

[WT02] WS-T. Web Services Transaction. http://www-128.ibm.com/developerworks/library/ws-transpec/, August 2002.