Effektive Architekturgestaltung auf Basis einer Typologie für Datenintegrationsarchitekturen

Reinhard Jung

Lehrstuhl für Wirtschaftsinformatik und Betriebliche Kommunikationssysteme
Universität Duisburg-Essen, Institut für Informatik und Wirtschaftsinformatik
Universitätsstraße 9, DE-45141 Essen
reinhard.jung@icb.uni-due.de

Abstract: Die Datenintegration ist eine der zentralen Voraussetzungen für die Durchführung von Integrationsvorhaben, sei es unternehmensintern oder -übergreifend. Der Entwurf einer Datenintegrationsarchitektur ist deshalb von großer Bedeutung. Der Aufsatz identifiziert fünf Merkmale, mit deren Hilfe sich Datenintegrationsarchitekturen beschreiben und unterscheiden lassen. Durch Kombination aller miteinander verträglicher Merkmalsausprägungen entstehen 22 grundsätzlich mögliche Datenintegrationsarchitekturtypen. Eine Gegenüberstellung von fachlichen Anforderungen an Daten, die beispielsweise aus Geschäftsprozessen ableitbar sind, mit den Eigenschaften der Datenintegrationsarchitekturtypen ermöglicht eine Vorauswahl geeigneter (effektiver) Typen.

1 Einleitung

Die häufige Änderung von Geschäftsmodellen und Leistungsportfolios sowohl von Unternehmen als auch von Behörden zwingt zu hoher Anpassungsfähigkeit (Flexibilität) im Bereich der Anwendungssysteme und der IT-Architekturen. In realen Szenarien kann zu diesem Zweck nicht das gesamte Anwendungsportfolio ersetzt werden. Stattdessen sind neue Anwendungssysteme mit den bestehenden zu integrieren, um die Effektivität des betrieblichen Informationssystems aufrecht zu erhalten oder herzustellen. Ein zentraler Schritt eines solchen Integrationsvorhabens ist die Integration der Daten aller beteiligten Systeme (Datenintegration).

Prinzipiell steht eine ganze Reihe von Architekturkonzepten zur Verfügung, die für eine Integration grundsätzlich geeignet erscheinen. Beispiele sind so unterschiedliche Architekturkonzepte wie Data-Warehouse-Systeme, Operational Data Stores und serviceorientierte Architekturen. Darüber hinaus ist auch im Bereich der Technologien inzwischen eine große Menge an Optionen verfügbar, beispielsweise Middleware-Produkte und -Standards, Integrations- oder Message-Broker und Web Services. Viele Architekturkonzepte sind allerdings nicht universell einsetzbar. Ein Data-Warehouse-System beispielsweise ist – zumindest in der „klassischen“ Sichtweise – aufgrund der Charakteristika von Datenextraktions- und Konsolidierungsprozessen nicht in der Lage, Echtzeit-Daten bereitzustellen oder etwa die Modifikation von Data-Warehouse-Daten in die

operativen Anwendungssysteme zu propagieren (eine grundsätzlich andere Sichtweise wird vertreten bspw. in [Ze03]).

Bevor eine konkrete Architektur gestaltet und eine Entscheidung für bestimmte Technologien und Software-Lösungen gefällt werden kann, muss eine genaue Spezifikation der Anforderungen erfolgen. Der vorliegende Beitrag widmet sich der Frage, wie anhand dieser Anforderungen effektive Architekturen für die Datenintegration gestaltet werden können. Zu diesem Zweck wird, nach einer Festlegung der begrifflichen Grundlagen (Kapitel 2), eine Menge von Merkmalen vorgestellt, mit deren Hilfe sich Datenintegrationsarchitekturen beschreiben und unterscheiden lassen (Kapitel 3). In Kapitel 4 wird die Merkmalsmenge genutzt, um eine Typologie für Datenintegrationsarchitekturen aufzubauen; die Überprüfung der Typologie erfolgt durch Anwendung der Merkmale auf bekannte Datenintegrationsarchitekturen. Ein abstraktes Anwendungsbeispiel für die Klassifikation findet sich in Kapitel 5. Die Typologie wird dabei genutzt, um auf Basis von Anforderungen effektive, also mit Blick auf die Anforderungen geeignete Architekturen auswählen zu können. Wirtschaftlichkeitsüberlegungen (Effizienz), die u.a. die erforderliche Anpassung von Altsystemen erfassen, werden in diesem Aufsatz ausgeklammert. Der Aufsatz schließt mit einer Zusammenfassung und einem Ausblick (Kapitel 6).

2 Begriffliche Grundlagen

Da Integration und insbesondere Datenintegration zu den traditionellen Analyse- und Gestaltungsgebieten der Informatik- und Wirtschaftsinformatik-Forschung gehören [Ro99], existiert eine Fülle von teilweise stark divergierenden Definitionen. Eine ausführliche Begriffsanalyse findet sich in [Ju06, 34 ff.]. Um zu tragfähigen Definitionen für diesen Beitrag zu gelangen, bietet es sich an, zunächst die in der aktuellen Literatur genannten Integrationsarten zu untersuchen:

- [RMB01, 17 ff.] unterscheiden Präsentations-, Daten- und Funktionsintegration.
- [FS01, 215 ff.] unterscheiden Daten-, Funktions- und Objektintegration.
- [Li00] unterscheidet neben Präsentations-, Daten- und Objektintegration auch die Integration von Applikationen durch Kopplung mit Hilfe von Schnittstellen.
- [BU02] unterscheiden Daten-, Applikations- und Prozessintegration, wobei Applikationsintegration als Kombination von Daten-, Funktions- und Präsentationsintegration aufgefasst wird.

Bereits die Unterscheidung von Integrationsarten zeigt ein heterogenes Bild. Als weitgehender Konsens in der Literatur lassen sich allerdings die Abhängigkeiten zwischen den verschiedenen Arten festhalten: Datenintegration ist eine Voraussetzung für die Funktionsintegration, und die Objektintegration ist eine alternative Sichtweise auf die Daten- und Funktionsintegration. Schließlich wird die Daten- und Funktionsintegration bzw. die Objektintegration als Voraussetzung für die Prozessintegration gesehen. Da der Auslöser von Datenintegrationsmaßnahmen primär in fachlichen Anforderungen zu sehen ist, liegt diesem Aufsatz entsprechend eine eher betriebswirtschaftliche Definition zugrunde:

Datenintegration ist der „Zustand, bei dem Aufgabenträger innerhalb eines Untersuchungsbereichs Zugriff auf die Informationsobjekte haben, die für die Aufgabenerfüllung erforderlich sind. Die Informationsobjekte müssen dabei den aufgaben- und aufgabenträgerspezifischen Qualitätsanforderungen genügen." [Ju06, 45]. Dabei ist ein Informationsobjekt eine zweckbezogene Zusammenfassung von miteinander in Beziehung stehenden Daten. Informationsobjekte können eine einfache Struktur aufweisen (z.B. Informationsobjekttyp „Kunde") oder eine komplexe Struktur (z.B. Informationsobjekttyp „Umsatzvolumen nach Kunde, Region und Zeitpunkt"). Unsere Definition von Datenintegration schließt implizit alle denkbaren Verwendungen von Daten ein, also sowohl eine transaktionsorientierte und typischerweise modifizierende Nutzung als auch eine analytische, in der Regel nur lesende Nutzung. Ferner wird wie in [BU02, 416] davon ausgegangen, dass die Konsistenz des Gesamtdatenbestands zu erhalten ist.

Die bereits existierenden IKS werden zusammen als Applikationsarchitektur bezeichnet. Im Rahmen eines Integrationsvorhabens werden dieser Architektur zielgemäß (Ziel: Datenintegration) Integrationskomponenten hinzugefügt. Die Applikationsarchitektur bildet zusammen mit den Integrationskomponenten die Datenintegrationsarchitektur (DIA). Durch Abstraktion von konkreten Datenintegrationsarchitekturen erhält man über die Zwischenstufe von DIA-Modellen (vgl. dazu auch [Si04, 315]) schließlich DIA-Typen (vgl. dazu Abb. 1). Dabei richtet sich die Zusammenfassung zu einem DIA-Typ danach, inwieweit die betreffenden Datenintegrationsarchitekturen (oder auch DIA-Modelle) strukturelle Ähnlichkeit aufweisen.

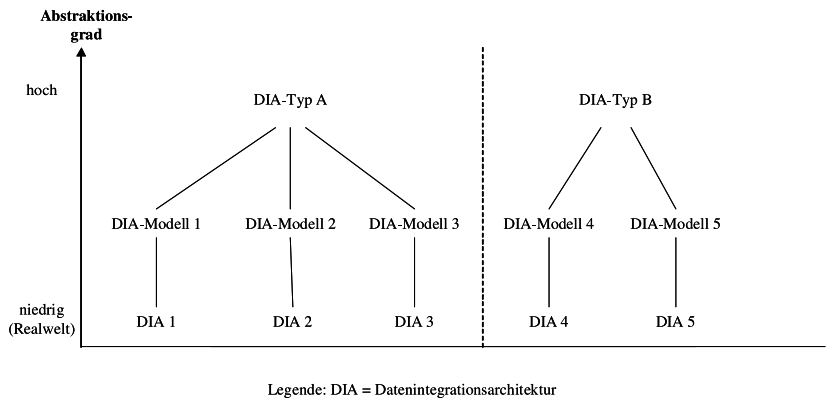

Abbildung 1: Abgrenzung der Begriffe „DIA", „DIA-Modell" und „DIA-Typ" anhand eines abstrakten Beispiels

Zum Zweck einer begrifflichen Differenzierung werden Daten, die für neue Aufgaben benötigt werden und durch Integration erzeugt werden, als globale Daten bezeichnet. Die Daten der bereits vorhandenen Applikationen werden hingegen als lokale Daten bezeichnet. Die Begriffe entstammen dem Forschungsgebiet „Schemaintegration" (vgl. z.B. [LNE89]).

Als Ergänzung zu den begrifflichen Grundlagen sei noch der Begriff Enterprise Application Integration (EAI) erwähnt, der hier thematische Relevanz besitzt, inzwischen aller-

dings zu Recht als Schlagwort bezeichnet wird [Ho03, 45]. Die Begriffsvielfalt sei anhand der folgenden Definitionen bekannter Vertreter der EAI-Szene aufgezeigt, die deutlich voneinander abweichen:

- Ruh et al. bezeichnen die Erstellung neuer Geschäftslösungen durch Kombination von (bestehenden) Applikationen unter Verwendung von Middleware als EAI [RMB01, 2].
- Linthicum definiert EAI als das uneingeschränkte Teilen von Daten und Geschäftsprozessen zwischen beteiligten Applikationen und Datenbasen innerhalb eines Unternehmens [Li00, 3].
- Stonebraker bezeichnet den Mechanismus zur Integration voneinander isolierter Informationen als EAI [St99, 2].

Der von uns vorgeschlagene Ansatz für den Entwurf der Datenintegration ist mit den genannten Interpretationen von EAI insoweit vereinbar, als er (a) bestehende Applikationen berücksichtigt und (b) auf die Integration („sharing“) von Daten zwischen Applikationen und Datenbanken gerichtet ist. Unser Ansatz entspricht damit weitgehend der EAI-Definition von Stonebraker. Wir sehen in der von uns angestrebten Datenintegration eine zentrale Voraussetzung für umfassendere Ansätze, die in den beiden anderen Definitionen implizit oder explizit zum Ausdruck kommen, insbesondere für die Prozessintegration.

3 Merkmale von Datenintegrationsarchitekturen

In diesem Abschnitt wird untersucht, mit Hilfe welcher Merkmale sich Datenintegrationsarchitekturen beschreiben und damit unterscheiden lassen; eine ausführliche Herleitung findet sich in [Ju06, 192 ff.].

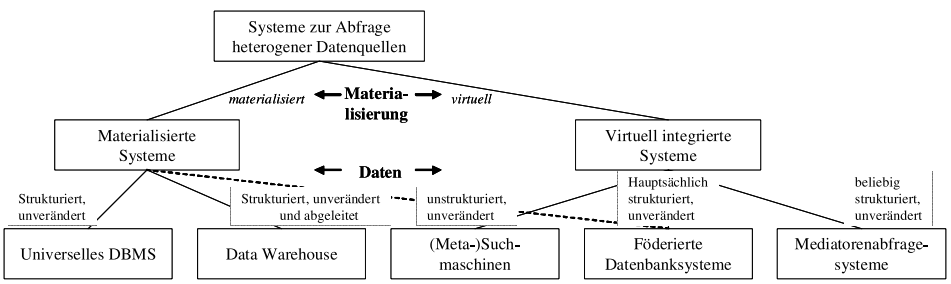

Abbildung 2: Klassifikation von Systemen zur Abfrage heterogener Datenquellen nach [DD99, 65]

Als Ausgangspunkt zur Identifikation von Merkmalen lässt sich die von [DD99] vorgeschlagene Klassifikation von Systemen zur Abfrage heterogener Datenquellen (vgl. Abb. 2) nutzen. Ein erstes Gliederungskriterium ist dort die Materialisierung der integrierten Daten mit den Ausprägungen „materialisiert“ und „virtuell“. Auf einer zweiten Stufe wird dann nach der Beschaffenheit der integrierten (globalen) Daten hinsichtlich Strukturierung und Modifikation gegenüber den lokalen Daten unterschieden. Obwohl dieses

zweite Merkmal für die Beschreibung einer Datenintegrationsarchitektur nicht wesentlich ist, führen die resultierenden Ausprägungen zur Identifikation eines weiteren wichtigen Merkmals, nämlich der Architekturtopologie, denn die auf der untersten Ebene in der Abbildung unterschiedenen Architekturen weichen in ihrer Topologie stark voneinander ab. In den beiden folgenden Abschnitten werden die Merkmale „Architekturtopologie“ und „Materialisierung“ dargestellt.

3.1 Architekturtopologie

Bezogen auf das Merkmal „Architekturtopologie“ lassen sich drei Formen unterscheiden: multilaterale Kopplung, Föderation und Fusion. Die Unterscheidung bezieht sich insbesondere darauf, wie zentral oder dezentral bestimmte Aufgaben innerhalb der Architektur organisiert sind. Zur Veranschaulichung dieser drei Formen wird im Folgenden von drei vorhandenen Applikationen (vA) und einer neuen Applikation (nA) ausgegangen, wobei die nA der Unterstützung neuer Aufgaben dient und damit der „Nachfrager“ von integrierten Daten ist.

Multilaterale Kopplung (Peer-to-peer)

Das entscheidende Merkmal der multilateralen Kopplung liegt darin, dass Komponenten eines betrachteten Systems mit allen übrigen Komponenten des Systems verbunden werden (können). Eng verbunden mit der Idee der Kopplung ist der Begriff „Peer-to-peer“, der im Sinne eines Netzwerks aus Komponenten (Peers) wie folgt charakterisiert wird [SF02, 587]:

- Jede Komponente kann Server- und Client-Funktionalität aufweisen, d.h. Anfragen sowohl stellen als auch beantworten.
- Die Kommunikation zwischen den Komponenten erfolgt direkt, d.h. ohne Zwischenschaltung einer zentralen Instanz.
- Die Komponenten behalten vollständig ihre Autonomie.

Im vorliegenden Kontext bringt es die Autonomie der Applikationen und das Fehlen einer zusätzlichen Komponente mit sich, dass Erweiterungen oder Modifikationen der zu koppelnden vA erforderlich werden. So ist insbesondere die Transaktionsverwaltung eine zusätzlich durchzuführende Aufgabe. Dabei ist festzulegen, ob die vA die Transaktionen auf dem eigenen Datenbestand überwachen, oder ob diejenige Applikation die Überwachungsfunktion übernimmt, welche die Transaktion ausgelöst hat. Eine weitere Aufgabe bei der Integration der zu koppelnden Applikationen ist die Datenformatkonvertierung.

Föderation

Der Begriff „Föderation“ entstammt dem Bereich der Datenbanktechnik. Dort wird ein Multidatenbanksystem, dessen Komponentensysteme (Datenbanken) eine gewisse Selbständigkeit (Autonomie) behalten, als föderiertes Datenbanksystem bezeichnet [Co97, 41]. Entsprechend lässt sich im Kontext dieses Beitrags der Begriff „Föderation“

verwenden. Unter einer Föderation wird eine Architekturtopologie verstanden, in der bestimmte Aufgaben im Zusammenhang mit der Datenintegration (z.B. Transaktionsverwaltung, Datenformatkonvertierung) von einer zentralen, koordinierenden Komponente ausgeführt werden. Gleichzeitig behalten die an der Föderation beteiligten Komponenten ihre Autonomie.

Fusion

Bei der Fusion existiert eine zentrale Komponente, welche die nA bedient und dabei eine vollständige Kontrolle über die involvierten Datenbestände ausübt; insbesondere die Transaktionskontrolle wird dabei zentralisiert. Dies kann durch zwei Varianten erreicht werden:

- Faktische Fusion: Die lokalen Daten werden in eine zentrale Datenbasis migriert, die der koordinierenden Komponente zugeordnet ist. Nachfolgend werden die vA an die veränderte Datenhaltung angepasst oder – falls ihre Funktionalität anderweitig abgedeckt wird – ausser Betrieb gesetzt.
- Virtuelle Fusion: Die vA werden in einen Verbund unter zentraler Kontrolle durch die koordinierende Komponente eingebunden, wobei die vA ihre Autonomie vollständig verlieren. Auch hier ist ein globales Schema zu erstellen.

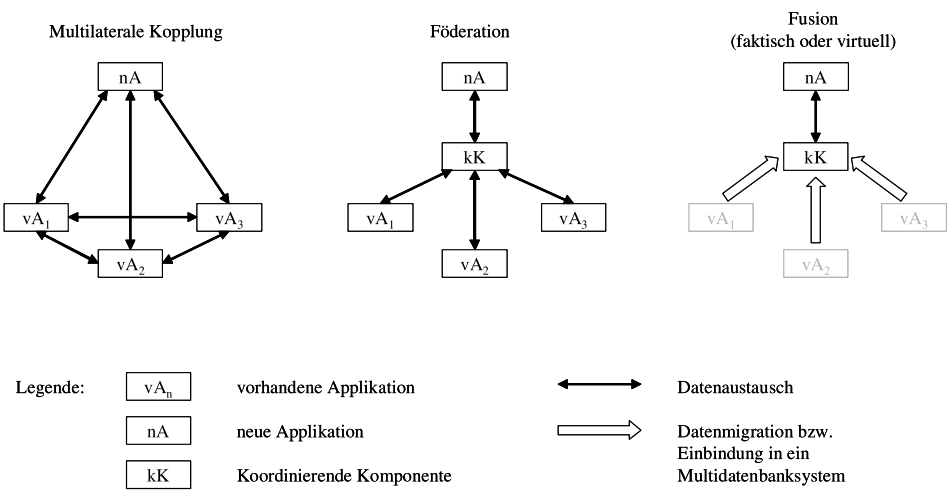

Abbildung 3: Architekturtopologien

In Abb. 3 sind die drei möglichen Architekturtopologien veranschaulicht. Bei der dritten Variante sind die vA schattiert dargestellt, weil sie bei der faktischen Fusion ggfs. entfallen.

3.2 Materialisierung

Das Merkmal „Materialisierung“ bezieht sich auf die Frage, ob die zur Deckung des Informationsbedarfs erforderlichen Daten in einer integrierten Form vorgehalten werden (materialisiert, „fetch in advance“) oder aber erst zum Zeitpunkt der Anfrage (virtuell, „fetch on demand“) konsolidiert und weitergegeben werden [LB02, 76; SH01, 556].

Materialisierung ist bei den Architekturtopologien „Föderation“ und „multilaterale Kopplung“ möglich. Im ersten Fall beinhaltet die koordinierende Komponente eine zentrale Datenbasis, welche die replizierten Daten aufnimmt. Im zweiten Fall werden bei jeder Applikation zusätzlich zu den „eigenen“ Daten Replikate von Daten anderer Applikationen abgelegt.

Eine Materialisierung bringt Vorteile mit sich, da ein Datenzugriff dann unabhängig von einer eingeschränkten Verfügbarkeit der Kommunikationsverbindungen und der Applikation erfolgen kann, welche die lokalen Daten führt. Gleichzeitig entstehen aber auch Nachteile [NI02, 431], beispielsweise bei der Konsistenzsicherung zwischen Replikaten bei Nicht-Verfügbarkeit der beteiligten Komponenten.

Ein Spezialfall ergibt sich, wenn bestimmte Daten in eine separate Datenbank migriert werden. In [Br01, 220 f.] wird dieser Fall als „Shared data solution“ bezeichnet: Daten, die von mehreren Applikationen benötigt werden, werden in eine zentrale Datenbank migriert; die übrigen (lokalen) Daten verbleiben bei den jeweiligen Applikationen. Da mit der zentralen Datenbank auch eine koordinierende Komponente in Form eines zugehörigen Datenbankmanagementsystems erforderlich wird, ist diese Variante der Architekturtopologie „Föderation“ zuzurechnen.

Es ergeben sich die folgenden Merkmalsausprägungen für die Materialisierung: „materialisiert“, „virtuell“ und „teilweise migriert“. Die Ausprägung „teilweise migriert“ repräsentiert den von Britton beschriebenen Ansatz.

3.3 Transaktionstyp

Einen Hinweis auf ein weiteres Merkmal, das zur Unterscheidung von Architekturen für die Datenintegration geeignet ist, liefert [St99] im Zusammenhang mit der Klassifikation von EAI-Technologien. Stonebraker unterscheidet u.a. danach, welcher Zweck mit dem Technologieeinsatz verfolgt wird. Zum einen wird die Aktualisierung von Daten und zum anderen das Lesen genannt.

Als Merkmal einer Datenintegrationsarchitektur wird in diesem Zusammenhang im Folgenden von dem Transaktionstyp gesprochen, der innerhalb der Architektur auf globalen Daten zulässig ist. Unterschieden werden typischerweise – wie auch bei verteilten Datenbanksystemen [CA98, 280, Hu97, 52 ff.] – Architekturen, die ausschließlich lesende Zugriffe auf globale Daten zulassen, und solche, die zusätzlich auch schreibende Zugriffe auf diese Daten ermöglichen. Ein schreibender Zugriff ist dabei ein Zugriff, der Daten erzeugt, verändert oder löscht. Sofern schreibende Zugriffe auf globale Daten

zugelassen werden, ergibt sich implizit die Anforderungen, die zugehörigen lokalen Daten zu aktualisieren (vgl. dazu den folgenden Abschnitt).

3.4 Aktualisierungskontrolle

Zwischen lokalen Daten und dem Ort des Datenzugriffs (globale Daten), bei dem es sich im Fall der Materialisierung um eine dedizierte Datenbank handeln kann, ergeben sich zwei Aktualisierungsperspektiven:

- *Aktualisierungskontrolle lokal-zu-global*: Nach der initialen Materialisierung von globalen Daten kommt es häufig vor, dass die zugrunde liegenden lokalen Daten modifiziert werden. Es stellt sich daher die Frage, nach welchen Regeln und in welchem Zeitrahmen die Aktualisierung zwischen lokalen und globalen Daten erfolgen soll. Mit Blick auf den Modifikationszeitpunkt der lokalen Daten werden die beiden folgenden Varianten unterschieden: sofortige (synchrone) und verzögerte/spätere (asynchrone) Aktualisierung der globalen Daten.
- *Aktualisierungskontrolle global-zu-lokal*: Bei dieser Aktualisierungsperspektive wird davon ausgegangen, dass globale Daten (virtuelle oder materialisierte) verändert wurden. Auch hier stellt sich die Frage, wie die Aktualisierung der lokalen Daten entsprechend der Modifikation globaler Daten erfolgen soll; es ergeben sich wiederum die oben beschriebenen Möglichkeiten einer synchronen oder asynchronen Aktualisierung.

4 Datenintegrationsarchitekturtypen

Im vorhergehenden Abschnitt wurden Architekturtopologie, Materialisierung, Transaktionstyp und Aktualisierungskontrolle (lokal-zu-global, global-zu-lokal) als Merkmale vorgestellt, mit denen sich Datenintegrationsarchitekturen beschreiben lassen.

4.1 Typisierung

Um DIA-Typen identifizieren zu können, sind die Konstruktionsmerkmale zunächst in eine Reihenfolge zu bringen, welche die wechselseitigen Abhängigkeiten berücksichtigt:

- Die Global-zu-lokal-Aktualisierung ist nur relevant, wenn auf die (materialisierten oder virtuellen) globalen Daten auch schreibend zugegriffen wird.
- Die Lokal-zu-global-Aktualisierung ist nur dann erforderlich, wenn die globalen Daten materialisiert vorliegen.
- Wird als Architekturtopologie die Fusion gewählt, dann sind das Merkmal „Materialisierung“ sowie die davon abhängigen Merkmale irrelevant.

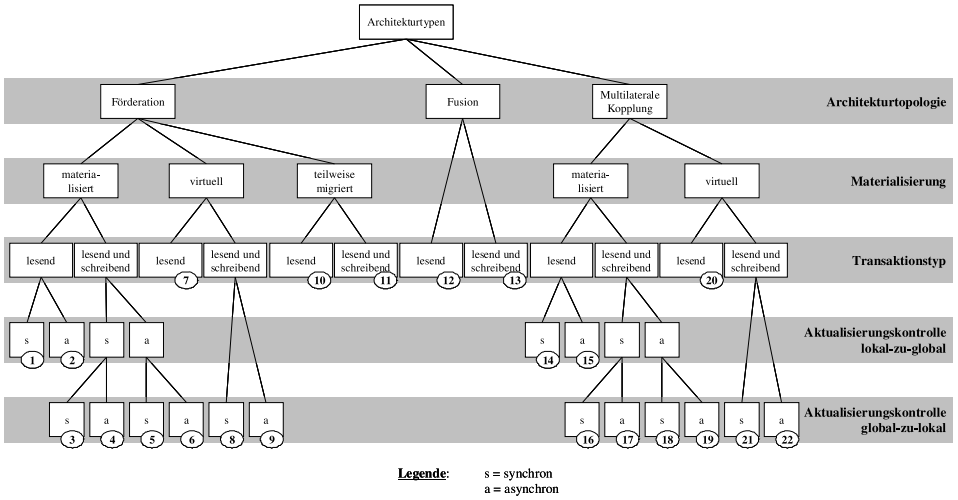

Abbildung 4: Merkmalsbaum für Datenintegrationsarchitekturen [Ju06, 222]

Aus den Abhängigkeiten lässt sich eine baumförmige Strukturierung der Merkmalsausprägungen ableiten. Innerhalb dieser Baumstruktur ist jede Merkmalsausprägung auf einer Ebene mit den möglichen Merkmalsausprägungen der darunter liegenden Ebene verbunden (vgl. Abb. 4).

Durch diese Anordnung entstehen Pfade von der Wurzel des Baums bis zu den verschiedenen Blattelementen, so dass jeder Pfad einen von insgesamt 22 möglichen DIA-Typen repräsentiert.

4.2 Überprüfung der Datenintegrationsarchitekturtypen

Die Überprüfung der Merkmalsmenge und damit der hergeleiteten DIA-Typen umfasst zwei Aspekte. Es ist zu untersuchen, ob sich die Merkmalsmenge einerseits vollständig und andererseits disjunkt verhält. Da sich Vollständigkeit in diesem Kontext nicht beweisen lässt, unternehmen wir den Versuch, die Merkmalsmenge zu falsifizieren. Zu diesem Zweck erfolgt eine Beschreibung der aus Literatur und Praxis bekannten Datenintegrationsarchitekturen mit Hilfe der Merkmalsmenge. Auf diese Weise kann auch die Disjunktheit überprüft werden: Unterschiedliche Datenintegrationsarchitekturen sollten unterschiedliche Kombinationen von Merkmalsausprägungen aufweisen. Einen weiteren Überprüfungsschritt stellt die Einordnung anderer Klassifikationsansätze in unseren Ansatz dar (vgl. Abschnitt 6).

Data-Warehouse-Systeme

Die Konsolidierung von Daten aus unterschiedlichen betrieblichen Funktionsbereichen und ihre Weitergabe in – bezogen auf die Hierarchie – vertikaler Richtung und damit für dispositive Zwecke stellt eine wesentliche Facette der Datenintegration dar [PRW01, 181]. Aus verschiedenen Gründen ist es allerdings nicht möglich bzw. nicht praktikabel,

„auf“ den operativen Applikationen eine Datenintegration zu realisieren, die lediglich „virtuell“ ist und bei Bedarf ad hoc erzeugt wird. Beispiele sind eine für dispositive Zwecke zu „schmale“ Datenhistorie in den operativen Applikationen und mögliche Laufzeitengpässe, die durch umfangreiche Abfragen auf den operativen Daten entstehen könnten.

Der typische Lösungsansatz in diesem Kontext ist ein Data-Warehouse-System (DW), also eine themenorientierte, integrierte, zeitorientierte und nicht-volatile Datenbasis, die zur Unterstützung von Managemententscheidungen eingesetzt werden kann [In92]. Die in einem DW verfügbaren Daten weisen einige besondere Eigenschaften auf:

- *Redundanz*: Die Daten sind Kopien von lokalen Daten oder aus diesen abgeleitete Daten.
- *Stabilität des Datenbestands in definierten Zeitintervallen*: Da die Daten in regelmäßigen Abständen aus den operativen Datenbanken in das DW kopiert (extrahiert) werden, ist der Datenbestand im DW in den Zeitintervallen zwischen zwei Extraktionszeitpunkten stabil.
- *Eingeschränkte Datenaktualität*: In den vorgenannten Zeitintervallen nimmt die Aktualität der Daten – verglichen mit den lokalen Daten – ab.
- *Informationscharakter der Daten*: Der Datenfluss innerhalb eines DW ist unidirektional, d.h., die Daten besitzen lediglich Informationscharakter. Ihre Modifikation mit dem Ziel, damit durch Propagierung auch die lokalen Daten zu verändern, ist grundsätzlich nicht vorgesehen.

In Abb. 5 ist die Klassifikation eines DW mit Hilfe eines morphologischen Kastens dargestellt.

Architekturtopologie	Förderation	Fusion	Multilaterale Kopplung	
Materialisierung	materialisiert	virtuell	teilweise migriert	nicht anwendbar
Transaktionstyp	lesend	lesend und schreibend		
Aktualisierungskontrolle lokal-zu-global	synchron	asynchron	nicht anwendbar	
Aktualisierungskontrolle global-zu-lokal	synchron	asynchron	nicht anwendbar	

Abbildung 5: Morphologischer Kasten für ein Data-Warehouse-System (DIA-Typ 2)

Operational Data Stores

Wie auch das DW weist ein Operational Data Store (ODS) eine dedizierte Datenbank auf, die lokale Daten in replizierter Form enthält. Das Ziel ist hier allerdings nicht die Unterstützung von Entscheidungsprozessen, sondern die Bereitstellung von integrierten Daten für operative Geschäftsprozesse, beispielsweise für ein Kundenselbstbedienungssystem (z.B. Online-Banking). Aus dieser Zielsetzung resultieren von einem DW abwei-

chende Anforderungen und spezifische technische Eigenschaften des ODS [Wi00, 135 f.]:

- Ein ODS beinhaltet in der Regel keine historisierten, sondern integrierte aktuelle Daten. Die Daten sind mit Blick auf operative Geschäftsprozesse zu Datenobjekten zusammengefasst, d.h. ihre Struktur ist auf die Verwendung in Transaktionen ausgerichtet.
- Neben dem lesenden Zugriff bedingen Transaktionen in Geschäftsprozessen in der Regel auch einen schreibenden Zugriff auf die integrierten (globalen) Daten, mit der Konsequenz, dass die lokalen Daten möglichst umgehend synchronisiert (aktualisiert) werden müssen.
- Da die Daten in einem ODS aufgrund des Verwendungszwecks wesentlich aktueller als die in einem DW sein müssen, ist eine Aktualisierung in längeren Zeitintervallen nicht sinnvoll. Stattdessen sind die globalen Daten möglichst umgehend zu aktualisieren, wenn sich die lokalen Daten ändern.

Abb. 6 zeigt die Klassifikation eines ODS.

Architekturtopologie	Förderation	Fusion	Multilaterale Kopplung
Materialisierung	materialisiert	virtuell	teilweise migriert	nicht anwendbar
Transaktionstyp	lesend	lesend und schreibend	
Aktualisierungskontrolle lokal-zu-global	synchron	asynchron	nicht anwendbar
Aktualisierungskontrolle global-zu-lokal	synchron	asynchron	nicht anwendbar

Abbildung 6: Morphologischer Kasten für einen Operational Data Store (DIA-Typen 3 bis 6)

Enterprise-Resource-Planning-Systeme

Architekturtopologie	Förderation	Fusion	Multilaterale Kopplung
Materialisierung	materialisiert	virtuell	teilweise migriert	nicht anwendbar
Transaktionstyp	lesend	lesend und schreibend	
Aktualisierungskontrolle lokal-zu-global	synchron	asynchron	nicht anwendbar
Aktualisierungskontrolle global-zu-lokal	synchron	asynchron	nicht anwendbar

Abbildung 7: Morphologischer Kasten für ein Enterprise-Resource-Planning-System (DIA-Typ 13)

Föderierte Datenbanksysteme

Föderierte Datenbanksysteme (FDBS) [HM79] gehören zu den so genannten Multidatenbanksystemen; ein FDBS ist also ein Zusammenschluss mehrerer Datenbanksysteme [SL90]. Ein charakteristisches Merkmal dabei ist, dass die Komponenten(datenbank)systeme eines FDBS weitgehend ihre Selbstständigkeit (Autonomie) behalten [Co97, 41]. In diesem Zusammenhang sind insbesondere die Ausführungs- und die Kommunikationsautonomie zu nennen [BBE99, 5]. Ausführungsautonomie besagt, dass die Komponentensysteme selbst „entscheiden", wann sie Aufträge von anderen Komponenten ausführen. Kommunikationsautonomie besagt analog, dass sie selbst entscheiden, wann sie mit anderen Komponenten kommunizieren.

Die zentrale Komponente eines FDBS ist der so genannte *Föderierungsdienst*. Seine Aufgaben lassen sich grob wie folgt beschreiben:

- Auswertung und Weiterleitung von Anfragen der globalen Applikation(en) an die beteiligten Datenbankmanagementsysteme sowie Konsolidierung der Anfrageergebnisse und Rücksendung an die globalen Applikation(en).
- Auswertung von Schreibzugriffen der globalen Applikation(en) und Durchführung der entsprechenden Transaktionsverwaltung, was unter der Prämisse der Autonomieerhaltung bei den Komponenten und mit Blick auf verteilte Transaktionen durchaus zu Schwierigkeiten führen kann.

Ein FDBS ist somit ein datenintegrierendes System, denn der Föderierungsdienst (als einzige ergänzte Komponente) realisiert für globale Applikationen, also solche mit Datenintegrationsbedarf, „eine eigene Datenbankfunktionalität [...], so dass sich das Gesamtsystem dem globalen Nutzer, der über den Föderierungsdienst auf die Komponentensysteme zugreift, als ein Datenbanksystem darstellt" [Co97, 50]. Ein FDBS weist bezüglich der integrierten Daten spezifische Eigenschaften auf:

- *Redundanzfreiheit*: Ein FDBS stellt lediglich einen Dienst „auf" vorhandenen lokalen Daten zur Verfügung; die angeforderten Daten werden ad hoc integriert und weitergeleitet, sodass keine Redundanz durch Materialisierung entsteht.
- *Hohe Datenaktualität*: Sofern eine Anfrage sowie die spätere Auswertung und Weiterleitung des Ergebnisses durch den Föderierungsdienst in einem sehr kurzen Zeitraum ausgeführt werden kann, ist von einer – verglichen mit der Datenaktualität in den operativen Datenbanken – ähnlich hohen Datenaktualität auszugehen.
- *Eingeschränkter Zugriff*: Da die Komponenten-Datenbanksysteme eines FDBS ihre Autonomie in der Regel behalten, kann es dazu kommen, dass Datenzugriffe nicht immer (sofort) ausgeführt werden.

– *Schreibende Transaktionen*: Im Gegensatz zu einem DW sind bei einem FDBS auch schreibende Transaktionen auf globalen Daten mit Propagierung in die lokalen Datenbestände vorgesehen.

In Abbildung 8 ist die Klassifikation eines FDBS dargestellt.

Architekturtopologie	Förderation		Fusion		Multilaterale Kopplung
Materialisierung	materialisiert	virtuell	teilweise migriert		nicht anwendbar
Transaktionstyp	lesend		lesend und schreibend		
Aktualisierungskontrolle lokal-zu-global	synchron		asynchron		nicht anwendbar
Aktualisierungskontrolle global-zu-lokal	synchron		asynchron		nicht anwendbar

Abbildung 8: Morphologischer Kasten für ein föderiertes Datenbanksystem (DIA-Typen 7 bis 9)

5 Beispielhafte Anwendung

In diesem Abschnitt wird anhand eines abstrakten Beispiels aufgezeigt, wie die Klassifikation von Datenintegrationsarchitekturen verwendet werden kann, um auf Basis fachlicher Anforderungen zu einer Entscheidung hinsichtlich effektiver Datenintegrationsarchitekturen zu gelangen. In [Ju06, 212 ff.] wurden Merkmale des Informationsbedarfs aus Integrationssicht hergeleitet und daraufhin untersucht, inwiefern sie durch strukturelle Eigenschaften einer Datenintegrationsarchitektur beeinflusst werden (können). Zur Veranschaulichung wird hier auf eine Teilmenge der Merkmale zurückgegriffen (vgl. auch [Ju05]):

– *Antwortzeit*: Die Antwortzeit ist ein Maß für die maximale Zeit, die zwischen Anforderung und Bereitstellung von Daten vergehen darf.

– *Aktualität*: Die Aktualität drückt aus, inwieweit Daten den gegenwärtigen Zustand des zugrunde liegenden Realweltobjekts bzw. der zugrunde liegenden Realweltobjekte beschreiben sollen. In vielen Publikationen werden für besonders restriktive Anforderungen Ausprägungen wie „real-time“ und „near real-time“ verwendet.

– *Vollständigkeit*: Die Vollständigkeit drückt aus, inwieweit bei Datenbereitstellungen fehlende Werte akzeptabel sind. Die Forderung nach einer maximalen Vollständigkeit bei einer Verkaufsstatistik auf Basis von Monaten würde beispielsweise bedeuten, dass der Zugriff auf alle Monatsumsätze in einer bestimmten Periode erforderlich ist.

– *Verwendungsform*: Grundsätzlich sind zwei Verwendungsformen von Daten unterscheidbar. Zum einen können Daten für rein informative Zwecke eingesetzt werden, zum anderen ist es in manchen Szenarien erforderlich, Daten auch

verändern zu können. Im ersten Fall sprechen wir von der Verwendungsform „lesend“, im zweiten von „lesen und schreibend“.

Im Beispiel gehen wir davon aus, dass minimale Antwortzeiten sowie eine hohe Aktualität und Vollständigkeit der Daten gefordert werden sowie eine „lesende und schreibende“ Verwendung vorgesehen ist. Die folgenden Ausführungen beziehen sich auf Abb. Die Architekturtypen 10 und 12 sind nur von theoretischer Bedeutung und werden deshalb ausgeklammert. Bei beiden Typen ist eine Modifikation der integrierten Daten, die nur noch zentralisiert vorliegen, nicht möglich, sodass eine Einsetzbarkeit in der Praxis nicht gegeben ist.

Die vorgesehene Verwendungsform im Beispiel schränkt die Menge der DIA-Typen zunächst auf die Architekturtypen 3 bis 6, 8, 9, 11, 13, 16 bis 19, 21 und 22 ein, da nur diese für das Merkmal Transaktionstyp die Ausprägung „lesend und schreibend“ aufweisen. Die geforderte hohe Vollständigkeit schließt weiterhin Architekturtypen aus, die ausschließlich auf vorhandenen Datenressourcen aufsetzen und folglich Daten nur im Rahmen des ursprünglichen Verwendungszwecks vorhalten. Damit scheiden die Architekturtypen aus der Menge aus, die für das Merkmal „Materialisierung“ die Ausprägung „virtuell“ aufweisen, sodass nur noch die Architekturtypen 3 bis 6, 11, 13, 16 bis 19 in Frage kommen. In einem weiteren Selektionsschritt führt die geforderte hohe Aktualität bei den Architekturtypen mit replizierten Daten (Merkmal „Materialisierung“, Ausprägung „materialisiert“) zum Ausschluss aller Architekturtypen, die bei der „Aktualisierungskontrolle lokal-zu-global“ die Ausprägung „asynchron“ aufweisen. Das schränkt die Menge an noch zu überprüfenden Architekturtypen auf 3, 4, 11, 13, 16 und 17 ein. In einem letzten Schritt sind die Auswirkungen der geforderten kurzen Antwortzeit zu bedenken. Da die sechs in der Auswahl verbliebenen Architekturtypen über zusätzliche Datenbanken (mit globalen Daten) verfügen, ist die Antwortzeit bei lesenden Zugriffen selbst bei Nicht-Verfügbarkeit der lokalen Daten sicherlich als kurz einzustufen. Bei schreibenden Zugriffen auf die globalen Daten sind allerdings Aktualisierungen der lokalen Daten durchzuführen. Mit Blick auf die systemübergreifende Datenkonsistenz ist zu fordern, dass auch die hier durchzuführende Aktualisierung in Richtung der lokalen Datenbestände synchron erfolgt (Merkmal „Aktualisierungskontrolle global-zu-lokal“, Ausprägung „synchron“). Dadurch erfolgt ein letzter Eliminierungsschritt, als dessen Ergebnis lediglich die Architekturtypen 3, 11, 13 und 16 als effektiv (d.h. die Anforderungen erfüllend) einzustufen sind.

6 Zusammenfassung und Ausblick

Eine zentrale Voraussetzung für den zielgerichteten Entwurf von Architekturen ist die Möglichkeit, den Lösungsraum (hier: DIA-Typen) beschreiben zu können. Zu diesem Zweck wurde eine Menge von Merkmalen und deren Abhängigkeiten vorgestellt. Das Ergebnis sind 22 DIA-Typen, die jeweils spezifische Eigenschaften aufweisen. Eine Gegenüberstellung von fachlichen Anforderungen an Daten, die beispielsweise aus Geschäftsprozessen ableitbar sind, mit den Eigenschaften der DIA-Typen ermöglicht effektive (sachziel-konforme) Entwurfsentscheidungen (vgl. das abstrakte Beispiel in Kapitel 5). Die Validität der Merkmalsmenge sowie des Architekturselektionsprozesses aus dem

vorhergehenden Abschnitt konnte in drei Fallstudien aus dem Finanzdienstleistungsbereich bestätigt werden (vgl. [Ju06, 247 ff.]); in dieser Richtung erscheint eine weitere Überprüfung – auch in anderen Branchen – sinnvoll. Darüber hinaus ist zu überprüfen, ob die zusätzliche Berücksichtigung neuer Technologien und Konzepte, z.B. Data Streams, eine Modifikation der Merkmalsmenge und in der Folge der DIA-Typen bedingt. Einen weiteren Schwerpunkt zukünftiger Forschungsarbeit muss die Betrachtung der Effizienz bzw. der jeweils zu erwartenden Kosten (Formalziel) bilden, weil in einem Szenario mit mehreren als effektiv eingestuften DIA-Typen eine Entwurfsentscheidung auf Basis von Wirtschaftlichkeitsüberlegungen erfolgen muss.

7 Literaturverzeichnis

[BBE99] Bouguettaya, A.; Benatallah, B.; Elmagarmid, A.: An Overview of Multidatabase Systems: Past and Present. In (Elmagarmid, A.; Rusinkiewicz, M.; Sheth, A. Hrsg.): Heterogeneous Autonomous Database Systems. Morgan Kaufman, San Francisco, 1999; S. 1-32.

[Br01] Britton, C.: IT Architectures and Middleware. Addison-Wesley, Boston et al., 2001.

[Bu02] Bunjes, B.; Friebe, J.; Götze, R.; Harren, A.: Integration von Daten, Anwendungen und Prozessen am Beispiel des Telekommunikationsunternehmens EWE TEL. In: WIRTSCHAFTSINFORMATIK 44 (2002) 5; S. 415-423.

[CA98] Calvanese, D.; De Giacomo, G.; Lenzerini, M.; Nardi, D.; Rosati, R.: Information Integration: Conceptual Modeling and Reasoning Support. In (Halper, M. Hrsg.): Proceedings of the 6th International Conference on Cooperative Information Systems (CoopIS'98), IEEE Computer Society, New York, 1998; S. 280-291.

[Co97] Conrad, S.: Föderierte Datenbanksysteme. Springer, Berlin et al., 1997.

[DD99] Domenig, R.; Dittrich, K.R.: An Overview and Classification of Mediated Query Systems. In: ACM SIGMOD Record 28 (1999) 3; S. 63-72.

[FS01] Ferstl, O.K.; Sinz, E.J.: Grundlagen der Wirtschaftsinformatik, Band 1, 4. Aufl. München, Wien, Oldenbourg, 2001.

[HM79] Hammer, M.; McLeod, D.: On Database Management System Architecture, Technical Report MIT/LCS/TM-141. Massachusetts Institute of Technology, Cambridge, 1979.

[Ho03] Holten, R.: Integration von Informationssystemen. WIRTSCHAFTSINFORMATIK 45 (2003) 1; S. 41-52.

[Hu97] Hull, R.: Managing Semantic Heterogeneity in Databases: A Theoretical Perspective. In (Mendelzon, A.; Özsoyoglu, Z.M. Hrsg.): Proceedings of the sixteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems. ACM, New York, 1997; S. 51-61.

[In92] Inmon, W.H.: Building the Data Warehouse. Wiley, New York et al., 1992.

[Ju05] Jung, R.: Anforderungen an Informationsobjekttypen als Basis von Architekturentscheidungen bei der Datenintegration. In (Lenz, R.; Hasenkamp, U.; Hasselbring, W.; Reichert, M. Hrsg.): EAI-Workshop 2005 – Enterprise Application Integration. GITO-Verlag, Berlin, 2005; S. 82-89.

[Ju06] Jung, R.: Architekturen zur Datenintegration – Gestaltungsempfehlungen auf der Basis fachkonzeptueller Anforderungen. DUV, Wiesbaden, 2006.

[LB02] Lehner, W.; Bauer, A.: Data-Warehouse-Systeme – derzeitiger Stand und aktuelle Entwicklungen. In: Datenbank-Spektrum 2 (2002) 4; S. 76-78.

[Li00] Linthicum, D.S.: Enterprise Application Integration. Addison-Wesley, Reading 2000.

[LNE89] Larson, J.A.; Navathe, S.B.; Elmasri, R.: A Theory of Attribute Equivalence in Databases with Application to Schema Integration. In: IEEE Transactions on Software Engineering 15 (1989) 4; S. 449-463.

[NI02] Niemann, H.; Hasselbring, W.; Wendt, T.; Winter, A.; Meierhofer, M.: Kopplungsstrategien für Anwendungssysteme im Krankenhaus. In: WIRTSCHAFTSINFORMATIK 44 (2002) 5; S. 425-434.

[PRW01] Picot, A.; Reichwald, R.; Wigand, R.T.: Die grenzenlose Unternehmung, 4. Aufl. Gabler, Wiesbaden, 2001.

[RMB01] Ruh, W.A.; Maginnis, F.X.; Brown, W.J.: Enterprise Application Integration – A Wiley Tech Brief. Wiley, New York et al., 2001.

[Ro99] Rosemann, M.: Gegenstand und Aufgaben des Integrationsmanagements. In (Scheer, A.-W.; Rosemann, M.; Schütte, R. Hrsg.): Integrationsmanagement, Arbeitsbericht Nr. 65; Institut für Wirtschaftsinformatik der Westf. Wilhelms-Universität Münster, Münster, 1999, S. 5-18.

[SF02] Schoder, D.; Fischbach, K.: Peer-to-Peer. In: WIRTSCHAFTSINFORMATIK 44 (2002) 6; S. 587-589.

[SH01] Stonebraker, M.; Hellerstein, J.M.: Content Integration for E-Business. In (Sellis, T. Hrsg.): Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data. ACM Press, New York, 2001; S. 552-560.

[Si04] Sinz, E.J.: Unternehmensarchitekturen in der Praxis – Architekturdesign am Reißbrett vs. situationsbedingte Realisierung von Informationssystemen. In: WIRTSCHAFTSINFORMATIK 46 (2004) 4; S. 315-316.

[SL90] Sheth, A.; Larson, J.: Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. In: ACM Computing Surveys 22 (1990) 3; S. 183-236.

[St99] Stonebraker, M.: Integrating Islands of Information. In: EAI Journal (1999) September/October; S. 1-5.

[Wi00] Winter, R.: Zur Positionierung und Weiterentwicklung des Data Warehousing in der betrieblichen Applikationsarchitektur. In (Jung, R.; Winter, R. Hrsg.): Data Warehousing Strategie. Springer, Berlin et al., 2000; S. 127-139.

[Ze03] Zeh, T.: Data Warehousing als Organisationskonzept des Datenmanagements – Eine kritische Betrachtung der Data-Warehouse-Definition von Inmon. Informatik – Forschung und Entwicklung 18 (2003); S. 32-38.