Anforderungen an ein System zur automatisierten Behandlung inselübergreifenden Datenflusses

Interner Bericht

Markus Bon

Kaiserslautern, Mai/Juni 2002

1 Motivation

In großen Unternehmen wie DaimlerChrysler existieren eine Vielzahl verschiedener, oftmals sehr komplexer Prozesse. Bei der Entwicklung neuer Fahrzeuge sind unzählige Personen beteiligt, angefangen von verschiedenen Ingenieuren, die jeweils ihren Beitrag zum Gesamtfahrzeug liefern, über Werkzeugbauer, Personalverantwortliche oder auch Vorgesetzte.
Hinzu kommt, dass die Durchführung der Entwicklung nicht allein im Hause DaimlerChrysler durchgeführt wird. Aufgrund der immer kürzeren Entwicklungszeiten wird vielmehr immer öfter auf das Fachwissen spezialisierter Firmen zugegriffen, die ihren Beitrag zum Gesamtprodukt liefern.
In diesem Zusammenhang wird schnell klar, dass der Einsatz von Workflow-Management zur Verwaltung dieser doch sehr komplexen Prozesse großes Optimierungspotential bietet. Im Zuge des Projektes COW (Cross-Organizational-Workflows [KRS01] wurde bereits untersucht, inwieweit sich inselübergreifende Prozesse beschreiben lassen, und wie ein globaler Kontrollfluss umgesetzt werden kann. Ein weiterer hochinteressanter Aspekt ist der inselübergreifende Datenfluss [BHR01, BRH02]. Um sinnvoll zusammenarbeiten zu können, müssen natürlich auch Daten zwischen den beteiligten Gruppen ausgetauscht werden. Zwischen den an der Konstruktion beteiligten Ingenieuren müssen beispielsweise CAD-Geometrien ausgetauscht werden, um die Passgenauigkeit der Einzelteile zu garantieren. Für einen DMU-Check (den virtuellen Zusammenbau des Systems) müssen Gittermodelle aller Einzelteile zu einem Gesamtsystem zusammengefügt werden. Die Liste lässt sich beliebig fortführen.
In diesem Dokument wollen wir zunächst an einige typische Szenarien beschreiben, wie der Datenaustausch zwischen Inselsystemen durchgeführt wird. Dabei sollen techn. Randbedingungen ebenso betrachtet werden wie organisatorische. Da Firmendaten einen nicht unerheblichen Wert repräsentieren, müssen diese selbstverständlich vor unbefugtem Zugriff geschützt werden. Anschließend werden wir versuchen, die identifizierten Szenarios zu klassifizieren. Daraus sollen Anforderungen abgeleitet werden, die das zukünftige System erfüllen soll. Langfristig sollen hierfür Lösungskonzepte entwickelt werden, die schließlich zur Entwicklung eines Prototyps führen.

1.1 Das Projekt „Verteiltes Entwickeln“

In Zusammenarbeit mit der EADS wurde im Projekt „Verteiltes Entwickeln“ (VE) untersucht, inwieweit Daten von Ingenieuren auf verschiedenen Inseln genutzt werden können. Dieser Ansatz nutzte die Tatsache, dass die Daten der beteiligten Partner von Produktdatenverwaltungssystemen (PDVS) verwaltet und zur Verfügung gestellt werden. Um den verteilten Zugriff zu gewähren, wurde der gesamte Produktbaum in Teilbäume zerlegt. Für jeden dieser Teilbäume war einer der Partner verantwortlich. Es gab einen ausgezeichneten „Master“, der den Wurzelknoten und somit die Gesamtstruktur verwaltet. Der Übergang zwischen zwei Inseln wurde durch sogenannte Proxy-Objekte geregelt. Diese speziellen Knoten kennzeichnen die Stelle im Produktbaum, an der ein entfernter Teilbaum beginnt. Wird beim Navigieren durch die Gesamtstruktur ein solcher Knoten erreicht, muss der entsprechende Teilbaum übernommen werden. Mit Hilfe einer zentralen Informationsquelle, dem „Common Core“, wird der entsprechende Partner

kontaktiert und die Struktur-Information übertragen. Anschließend kann wie in einem lokalen Strukturbaum (bis zum nächsten Proxy-Objekt) navigiert werden.
Im Projekt VE wurde ein rein lesender Zugriff vorausgesetzt. Das Ändern von Teilen blieb bei dem jeweiligen Besitzer vorbehalten, Änderungswünsche eines Partners wurden über einen entsprechenden Änderungsprozess abgewickelt. Die Kooperationsdatenmenge ist durch die globale Struktur schon im Vorfeld ausgehandelt und ist eher statisch.

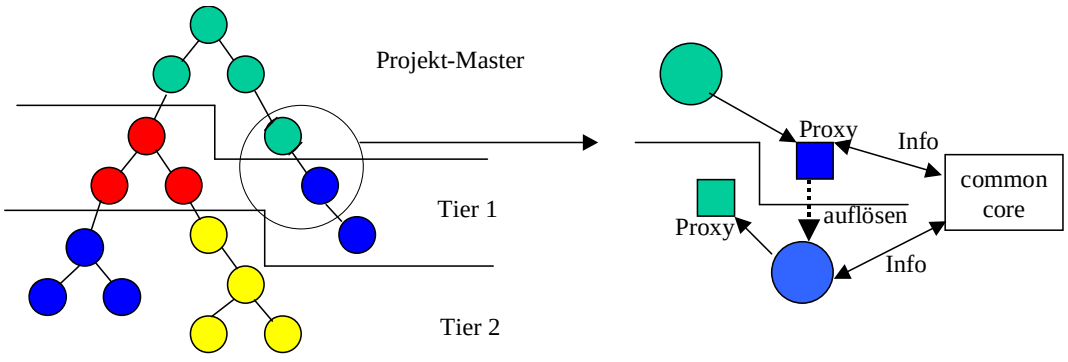

Abb. 1: Einsatz von Proxy-Objekten im Projekt VE

1.2 Projekt COW (EDAG)

Im Zuge des Projekts COW in Zusammenarbeit mit der Firma EDAG wurden verschiedene Szenarien näher beschrieben, von denen hier nur das Szenario „Konstruktion - Berechnung“ betrachtet werden soll. Im Zuge der gemeinsamen Entwicklung einer Autotür müssen zwischen DC und EDAG verschiedene Arten von Daten in beide Richtungen ausgetauscht werden. Nach Lieferung der CAD-Daten an DC wird ein Netzmodell des Gesamtfahrzeugs erzeugt und ein virtueller Crashtest durchgeführt. Die Ergebnisse dieser Berechnung werden wiederum an die EDAG zurückgeliefert. Dabei kann es sich um einfache Text-Dokumente, aber auch um Videofilme oder ähnliches handeln. Die Modulberechnungsergebnisse werden anschließend gespeichert, was bei der EDAG systemunterstützt und bei DC in der Verantwortung des zuständigen Mitarbeiters „per Hand“ geschieht.

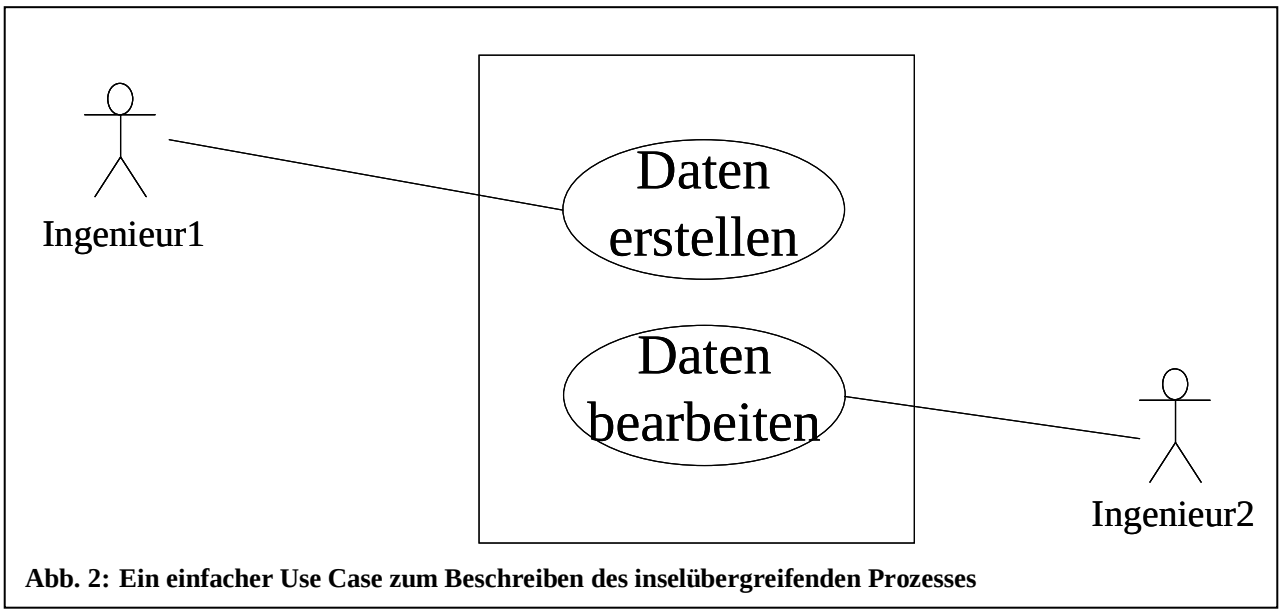

Abb. 2: Ein einfacher Use Case zum Beschreiben des inselübergreifenden Prozesses

Bei diesem Szenario ist der Prozess recht detailliert beschrieben. Die einzelnen Schritte werden terminlich zwischen DC und EDAG eng koordiniert, die vorgesehene Verteilung der Daten im globalen Datenfluss ist im Vorfeld schon recht klar. Große Flexibilität herrscht in der Art der ausgetauschten Dokumente und der Ablagestrategie im System.

2 Use Cases

In diesem Abschnitt werden wir eine Sammlung von Use Cases anlegen, die das Spektrum der bei inselübergreifenden Prozessen auftretenden Fälle abdecken sollen. Diese werden anschließend klassifiziert, was zu einer baumartigen Strukturierung führen wird.
Alle von uns betrachteten Prozesse haben die in Abb.2 gezeigte Form. Auf einem Inselsystem, d. h., einem System innerhalb einer Firma mit zugehörigen Ressourcen wie PDMS, DBMS, Entwicklungs-Tools, Applikationen, aber auch Mitarbeitern. Ausgangspunkt ist immer, dass Daten auf der einen Seite (der Quellinsel) erstellt bzw. auf einen neuen Stand gebracht werden. Diese Daten, die wir als *Kooperationsdaten* bezeichnen, müssen nun einem (oder auch mehreren) Bearbeitern auf einem anderen System (der Zielinsel) zur Verfügung gestellt werden.

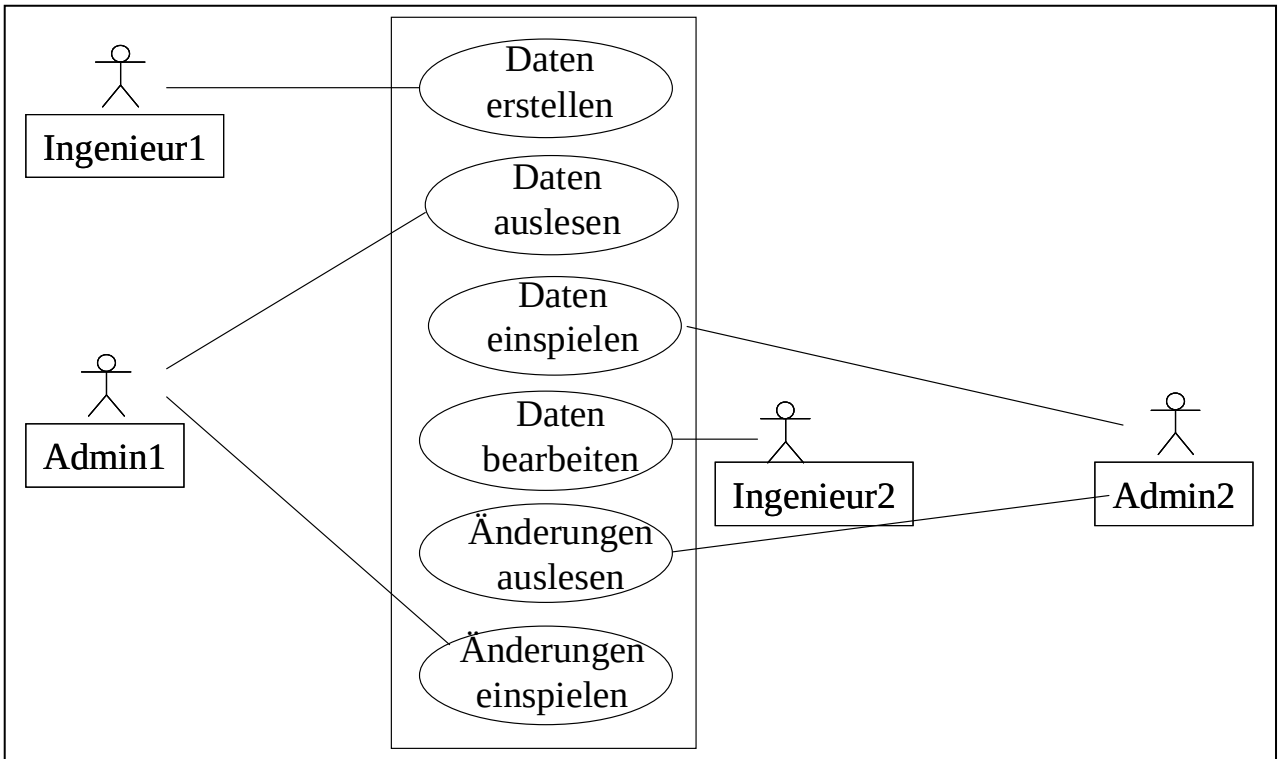

Abb. 3: Erweiterung des einfachen Use Cases um Datenübertragungsschritte

Zunächst wollen wir betrachten, welche weiteren Schritte zum Bereitstellen der Daten durchzuführen sind. Nach dem Erstellen der Kooperationsdaten müssen diese aus dem für sie zuständigen System (DBS, PDMS) ausgelesen und in eine übertragbare Form gebracht werden. Diese Aufgabe muss von einer dazu befähigten Person durchgeführt werden, die nicht nur weiß, was übertragen werden soll, sonder auch in welcher Form. Weiterhin muss diese Person auch über die notwendigen Rechte zum Auslesen der Kooperationsdaten verfügen.

Nach der Übertragung zur Zielinsel müssen die Daten wieder eingespielt werden. Dies muss durch eine Person erfolgen, die Kenntnis hat, in welchem System die Daten abzulegen sind, und natürlich auch die dazu notwendigen Rechte besitzt. Anschließend können die Daten lokal bearbeitet werden. Sollen hierbei durchgeführte Änderungen propagiert werden, so ist das Spiel in umgekehrter Reihenfolge zu wiederholen: Auslesen der geänderten Daten, Übertragen und Wiedereinspielen ins System. Diesen erweiterten Prozess zeigt Abb.3.
Da das Auslesen, Übertragen und Wiedereinspielen der Daten einen sehr zeitintensiven Vorgang darstellt, soll an dieser Stelle durch eine weitestgehende Automatisierung der Bereitstellung der Prozessablauf optimiert werden. Der daraus resultierende Prozess ist in Abb.4 zu sehen. Die Schritte zum Auslesen, Übertragen und Wiedereinspielen werden zu einem einzelnen Schritt „Daten handhaben“ zusammengefasst, was beispielsweise durch geeignete Integrationsarchitekturen oder einer Form des externen Datenmanagements (als logisches Gegenstück zum internen Datenmanagement des lokalen Systems). Dieser Schritt umfasst somit genau den Aufgabenbereich, den unser zu erstellendes System später abdecken soll.

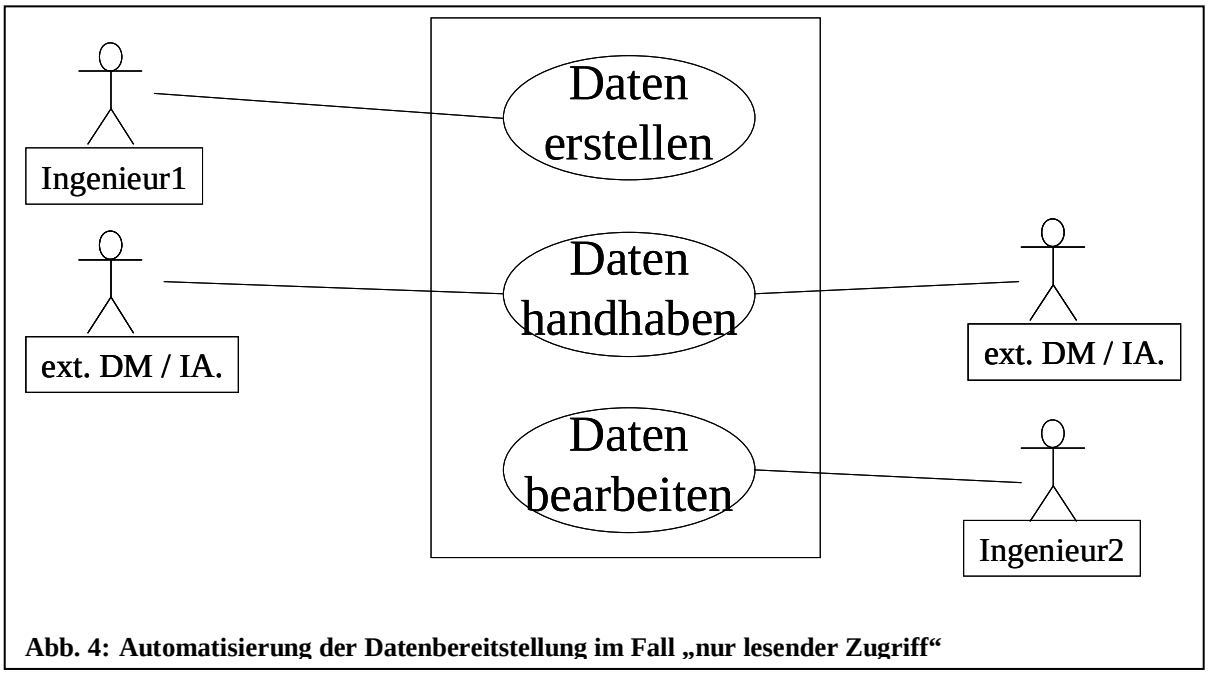

Abb. 4: Automatisierung der Datenbereitstellung im Fall „nur lesender Zugriff“

2.1 Zugriffscharakteristik

Eine wichtige Rolle für die Automatisierung des Zugriffs spielen offensichtlich Zeitpunkt und Häufigkeit des Zugriffs. Die Spannweite erstreckt sich von spontanem, ungeplanten Zugriff bis hin zu hochgradig vorgeplantem Datenaustausch. Letzterer ermöglicht die „einfachste“ Kontrolle durch das System, jedoch lassen sich in der Praxis nicht alle Prozesse (insbesondere bei der Ingenieurstätigkeit) auf diese Art und Weise planen.

2.1.1 „Ad hoc Zugriff“

Oftmals kommt es vor, dass auf bestimmte Informationen einmalig und ungeplant zugegriffen werden muss. Beispielsweise kann ein Ingenieur auf ein ähnliches, schon bestehendes Bauteil zurückgreifen wollen, um sich ein Bild über bisherige Entwicklungen zu machen. Der Besitzer des erwünschten Daten wird informiert, stellt die

Daten als Kopie zur Verfügung und überträgt sie an den Empfänger. Dieser kann dann lokal in seinem System darauf zugreifen. Der Schutz der Daten gegen unbefugten Zugriff von Seiten Dritter obliegt in diesem Fall dem Empfänger.

2.1.2 Ad hoc Zugriff auf Originaldaten

Eine Variante des in 2.1. geschilderten Falls besteht darin, keine Kopie der Daten anzufertigen, sondern den Zugriff auf die Originaldaten zu ermöglichen, beispielsweise durch Bereitstellen in einem freigegebenen Verzeichnis. Hierbei spielt auch das gewählte Sicherheitskonzept eine große Rolle, da zwar dem Anfrager Zugriff auf die Daten gewährleistet werden soll, aber keine weiteren, unbefugten Personen darauf zugreifen sollen.

2.1.3 Vorgeplante Übertragung von Datenreferenzen

Oftmals kann der Zeitpunkt der Datenbereitstellung auch vorgeplant werden, wenn etwa bestimmte Liefertermine vereinbart sind, oder in einer Iteration eine vordefinierte Qualitätsstufe erreicht wurde. In diesem Fall ist die Kooperation zwischen den Partnern in Form einer Prozessdefinition beschrieben, insbesondere können hier auch Datenabhängigkeiten explizit beschrieben werden. Nach Fertigstellen der Daten in dem entsprechenden Prozessschritt wird dem Empfänger eine Referenz zugestellt, mit deren Hilfe er die Kooperationsdaten lokalisieren kann. Sind ausreichende Zugriffsmöglichkeiten und Rechte vorhanden, kann er mit Hilfe dieser Referenz auf die Originaldaten zugreifen.

2.1.4 Vorgeplante Übertragung von Datenreferenzen mit anschließender Replikation

Auch in diesem Fall werden für den inselübergreifenden Datenfluss die Kooperationsdaten durch Angabe der Referenzen beschrieben. Wird jedoch über diese Referenz auf die gewünschten Daten zugegriffen, werden diese auf der Zielinsel repliziert und stehen zur lokalen Verarbeitung bereit. Dadurch werden nur Daten repliziert, die auch tatsächlich benutzt werden, jedoch verlängert sich die Zugriffszeit beim Erstzugriff.

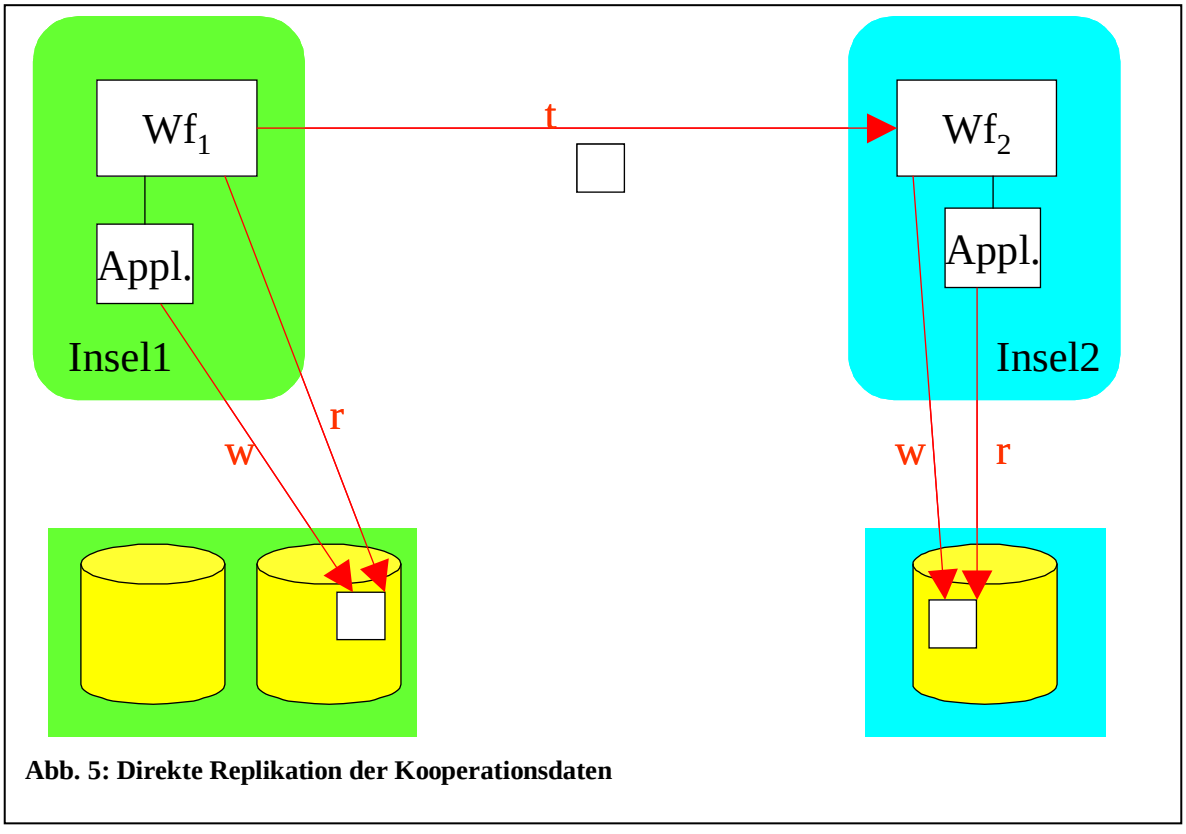

Abb. 5: Direkte Replikation der Kooperationsdaten

2.1.5 Direkte Replikation

Nach Fertigstellen der Kooperationsdaten werden automatisch Kopien an alle daran interessierten Zielsysteme geschickt. Nach Einspielen dieser Kopien können diese lokal bearbeitet werden. Der Übertragungsaufwand liegt hierbei höher, da auch Daten übertragen werden, die nicht unbedingt wirklich verwendet werden, jedoch ist der lokale Zugriff sehr schnell, da die Kooperationsdaten schon vor dem ersten Zugriff lokal bereitgestellt werden.

2.1.6 Replikation auf neutralem System

Die Kooperationsdaten werden in einem neutralen, vertrauenswürdiges Drittsystem zur Verfügung gestellt. Nach Dem Erstellen der Daten auf der Quellinsel wird eine Kopie auf das neutrale Drittsystem transferiert. Alle an diesen Daten interessierten Zielsysteme können dort auf diese zugreifen („Marktplatz“). Die Verantwortung für den Schutz der Daten liegt beim Drittsystem. Einen Vorteil bietet dieser Ansatz durch die erleichterte Konsistenzhaltung, da nun auf einem System alle Daten auf einmal vorhanden sind und auch nur auf einem System gepflegt werden müssen.

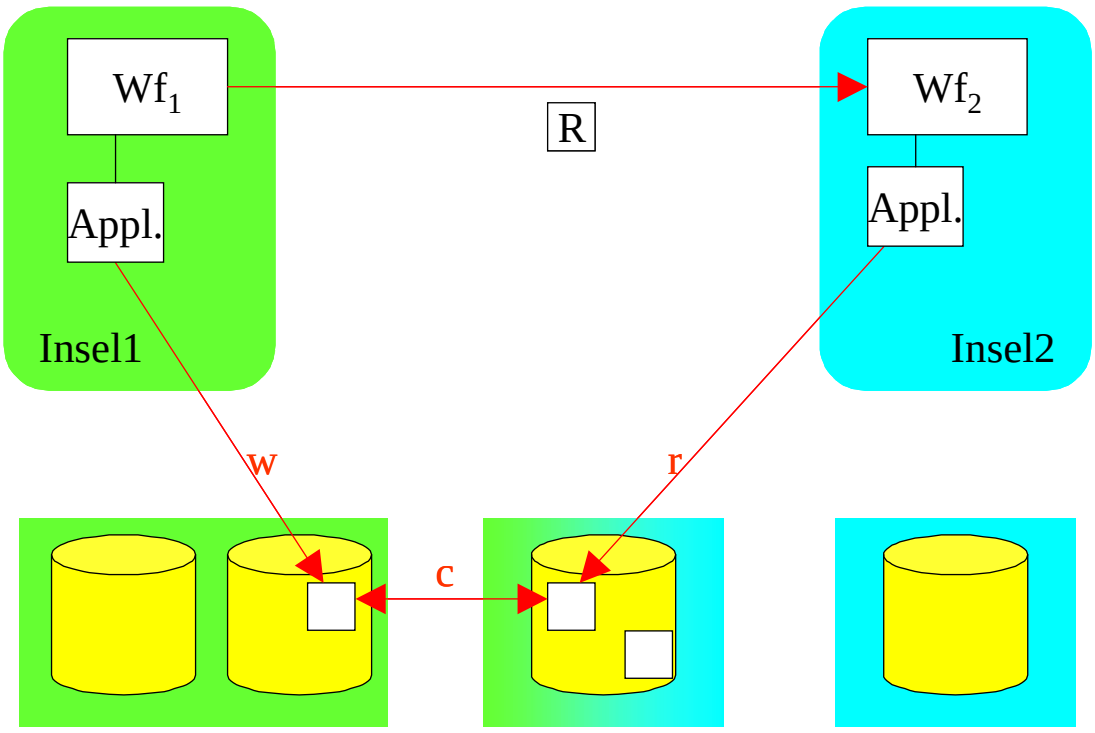

Abb. 6: Replikation der Daten in neutralem System („Marktplatz“)

2.1.7 Einordnung

Die beschriebenen Zugriffsformen lassen sich in einer Baumstruktur darstellen, wie sie in Abb. 7 zu sehen ist. Zunächst soll nur der lesende Zugriff betrachtet werden. Wir können zunächst zwischen ungeplantem und vorgeplantem Zugriff unterscheiden. Der Zugriff kann entweder auf die Originaldaten erfolgen, oder aber auf ein Replikat.
Bei der Vorplanung ist die Art des Zugriffs schon im Vorfeld bekannt, beispielsweise durch entsprechende Modellierung des zu Grunde liegenden Prozesses. Eine Möglichkeit besteht nun darin, dem Empfänger der Daten eine Referenz darauf zu übermitteln, die ihm

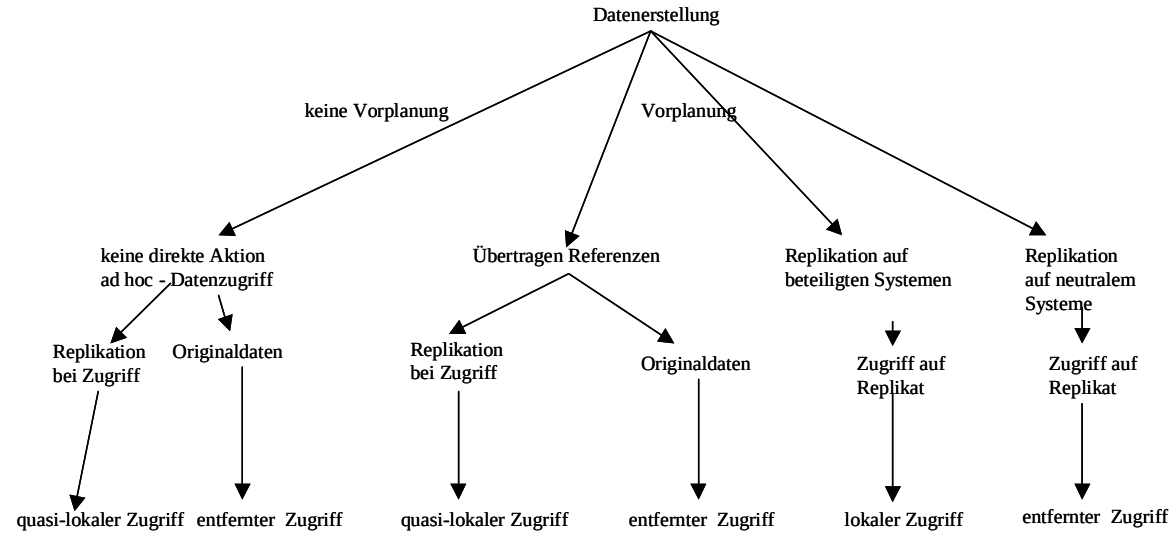

Abb. 7: Pfad-Darstellung der möglichen Zugriffsformen (nur lesend)

den Datenzugriff ermöglicht, oder aber direkt eine Kopie an die Zielinsel(n) zu übermitteln. Eine Variation davon ist die Übertragung der Daten an einen verlässlichen, wohlbekannten Platz, an dem die Daten von allen Interessierten (und Berechtigten) gelesen werden können.

Falls Änderungen, die auf der Zielinsel vorgenommen wurden, propagiert werden sollen, gibt es im wesentlichen die in Abb. 8 gezeigten Fälle. Wird mit den Originaldaten gearbeitet, können die Änderungen direkt eingebracht werden, sofern die entsprechenden Rechte gegeben sind. Dies kann jedoch bekanntermaßen zu großen Problemen und inkonsistenten Daten führen. Daher werden wohl in den meisten Fällen die Änderungen indirekt eingebracht, d. h., eine entsprechende Kontrolle auf der Quellinsel sorgt für ein einwandfreies Anpassen der Daten.

Das gleiche Problem entsteht auch bei der Arbeit mit Repliken, ein direktes Zurückschreiben der geänderten Kooperationsdaten macht daher selten Sinn. Vielmehr kommen hier meist sehr genau definierte Änderungsprozesse ins Spiel, die ein indirektes Einbringen ermöglichen.

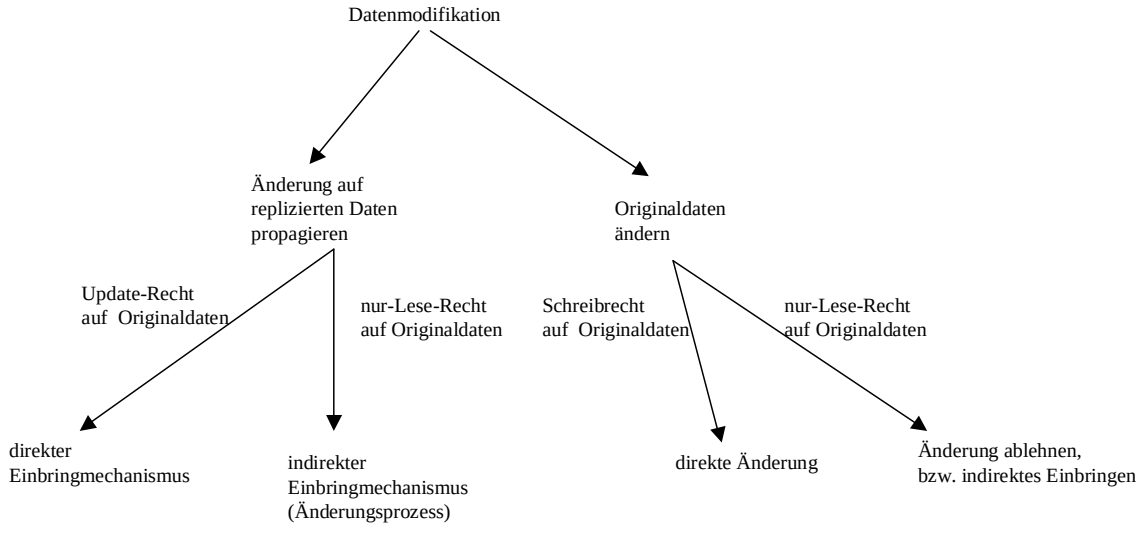

Abb. 8: Pfad-Darstellung für Änderungspropagierung

2.2 Datenzugriff

Die in einem Unternehmen vorhandenen Daten zu Produkten und Prozessen stellen einen hohen Wert dar und müssen dementsprechend vor unbefugtem Zugriff geschützt werden. Um dies zu erreichen sind verschiedene Ansätze denkbar, die im folgenden näher betrachtet werden sollen.

2.2.1 Indirekter Zugriff

Eine Möglichkeit, den Zugriff auf eigene Daten abzusichern, ist, ihn kurzerhand nicht zuzulassen. Falls Daten weitergegeben werden sollen, werden sie von einem Mitarbeiter ausgelesen, zur Übertragung vorbereitet und beispielsweise auf Band oder CD verschickt. Analog kann das Einspielen von fremden Daten geschehen. Nach Erhalt des Datenträgers kann der eigene Mitarbeiter die Daten einspielen und ggf. prüfen.

2.2.2 Direkter Lesezugriff

Falls der Partner Zugang direkten zum Quellsystem hat, kann ihm auch direkter Lesezugriff zu den Kooperationsdaten angeboten werden. In diesem Fall müssen Mechanismen zur Authentifizierung des Lesers gegenüber dem System und zur Autorisierung angeboten werden. Dem Leser müssen dann auf der für ihn relevanten Datenmenge entsprechende Leserechte eingeräumt werden.

2.2.3 Direkter Schreibzugriff

Dem Partner kann auch direkter Schreibzugriff auf die Kooperationsdaten eingeräumt werden. Neben den Mechanismen für Authentifizierung und Autorisierung des Lesers gibt es hier jedoch das zusätzliche Problem des konkurrenten Zugriffs, der zu Inkonsistenzen führen kann.

2.2.4 Zugriff über Funktionsschnittstelle

Anstatt dem Partner einen direkten Zugang zum System zu ermöglichen, kann ihm auch eine entsprechende Schnittstelle angeboten werden. Von außen wird (von einem autorisierten Leser) eine Funktion aufgerufen, die ihm die gewünschten Daten zurückliefert. Systeminterna können auf diese Art und Weise vollkommen verborgen werden.

2.3 Datenübertragung

Zum tatsächlichen physischen Übertragen der Daten gibt es ebenfalls verschiedene Möglichkeiten, die den folgenden Fällen zu Grunde liegen.

2.3.1 Binäre Protokolle

Eine Möglichkeit der Datenübertragung besteht im Einsatz eines einfachen Nachrichtenprotokolls. Im Fall der Übertragung über das Internet bietet sich dabei beispielsweise TCP/IP oder (eher ungewöhnlich) UDP an. Auf Quell-Seite muss hierzu ein geeignetes Sender-Modul bereitstehen, auf Zielseite ein dazu passendes Empfänger-Modul. Beide müssen für die Übertragung gleichzeitig aktiviert werden. Der Aufwand, der dabei entsteht, ist jedoch nicht zu unterschätzen. Die Kommunikation findet auf sehr niedrigem Niveau statt. Die Applikationen sind dafür zuständig, sich um die Verbindung zu kümmern, sich auf ein Nachrichtenformat zu einigen, die Kooperationsdaten sendefähig aufzubereiten und sie schließlich in Nachrichten zu ver- bzw. aus Nachrichten zu entpacken. Die Wartung einer solchen Sende-/Empfangsapplikation ist alles andere als einfach. Zusätzlich müssen auf der Firewall Ports freigeschaltet werden, um die

Kommunikation zu ermöglichen, was nur in Ausnahmefällen möglich ist. Weiterhin kann auch nicht davon ausgegangen werden, dass beide Systeme rund um die Uhr dazu bereit sind, synchron miteinander Daten auszutauschen.

2.3.2 FTP

Als weiteres Protokoll kommt das *File Transfer Protocol* (FTP) in Frage. Die Kooperationsdaten werden nach Fertigstellung auf der Quellseite auf einem FTP-Server zur Verfügung gestellt, der von Seiten der Zielinsel angesprochen werden kann (beim Einsatz von Firewalls z. B. durch Auslagern in die DMZ). Nach Information des zuständigen Benutzers (beispielsweise per E-Mail) können die Kooperationsdaten heruntergeladen werden.

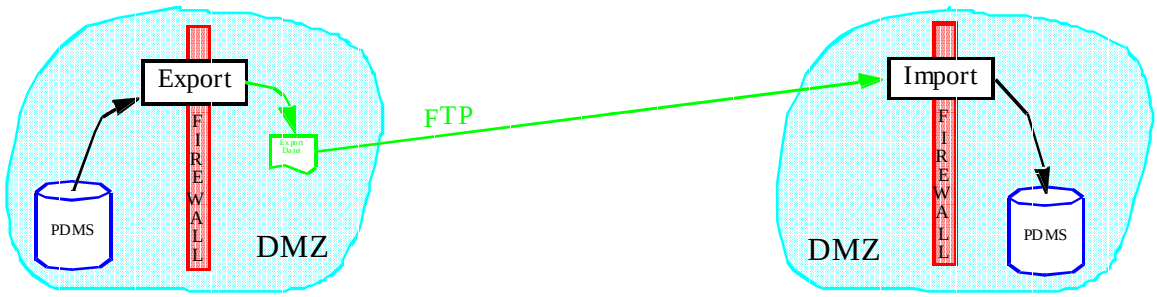

Abb. 9: Kooperationsdatenübertragung mit Hilfe von FTP

2.3.3 RPC

Ein inzwischen oft benutzter Mechanismus zur Kommunikation ist auch der Einsatz von *Remote Procedure Calls* (RPC), dem Aufruf von Methoden, die an entfernter Stelle angeboten werden. In diesem Zusammenhang wird oft die *CORBA*-Middleware eingesetzt; im JAVA-Umfeld wird mit RMI ein etwas schlankerer Dienst angeboten. Immer mehr Anklang findet auch das auf XML basierende *Simple Object Access Protocol* (SOAP), mit dessen Hilfe die Firewall-Problematik, wie später noch beschrieben wird, elegant umgangen werden kann.

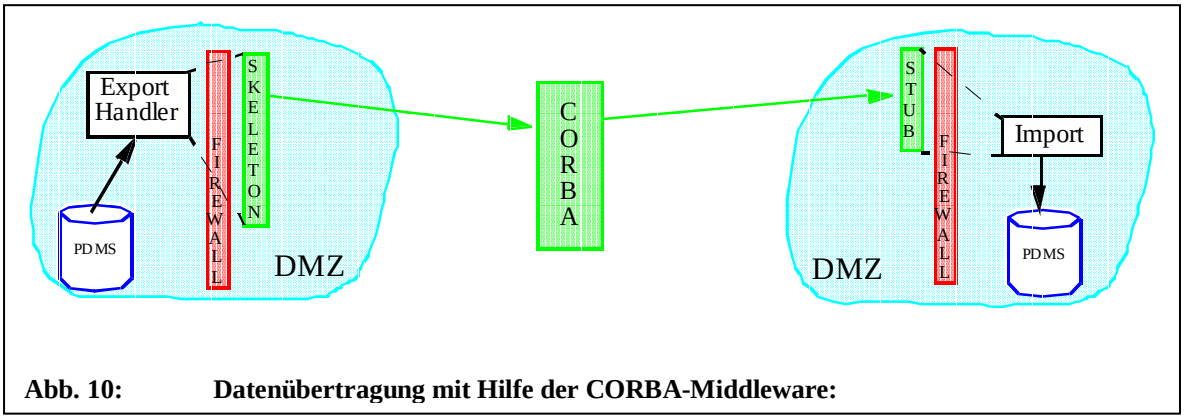

Abb. 10: Datenübertragung mit Hilfe der CORBA-Middleware:

2.3.3.1 CORBA/RMI

Beim Einsatz von RPCs werden lokal Methoden aufgerufen, die auf einem entfernten System ausgeführt werden. Dies wird (grob vereinfacht) möglich, in dem auf dem rufenden System nur Stellvertreter der eigentlichen Methoden (Stubs) aufgerufen werden. Die Parameter werden von der Middleware verpackt und übertragen, auf dem Zielsystem von einer entsprechenden Methode (Skeleton) entpackt und aufbereitet, bevor die eigentliche

Methode ausgeführt wird. Das Ergebnis wird dann auf gleiche Art und Weise zurückgeliefert. Leider erfordert der Einsatz dieser Middleware wieder „Lücken" in der Firewall, was den Einsatz in der Praxis erschwert.

2.3.3.2 SOAP

SOAP geht einen etwas anderen Weg. SOAP-Nachrichten werden in der Regel mittels HTTP übertragen. Die Verarbeitung geschieht über ein auf einem Applikationsserver bereitgestelltes Servlet. Der Applikationsserver kann sich an einen gängigen Web-Server angliedern (beispielsweise lässt sich TOMCAT über ein Erweiterungsmodul leicht an den APACHE-Web-Server koppeln). Die SOAP-Nachricht kann somit über den auf allen Firewalls üblicherweise freigeschalteten Port 80 das System erreichen und wird entsprechend an den Applikationsserver bzw. das zuständige Servlet (den SOAP Message Handler) weitergeleitet.
Die Kooperationsdaten werden auf Quellseite für den Transport vorbereitet, von einem geeigneten Servlet als ‚Nutzfracht" in eine SOAP-Nachricht verpackt und an die Zielinsel verschickt. Hier wird die Nachricht wiederum von einem Servlet zerlegt, die Kooperationsdaten können anschließend weiterverarbeitet werden.
Das Verschicken von Nachrichten mit SOAP ist deutlich langsamer als der Einsatz von CORBA oder RMI [GSC+00]. SOAP hat jedoch den großen Vorteil, dass es beim Einsatz von Firewalls die wenigsten Probleme bereitet, da es den im Allgemeinen freigeschalteten Port 80 benutzen kann.

2.3.4 Warteschlangen

Der Einsatz von Warteschlangen ermöglicht das Entkoppeln von Senden und Empfangen. Die Quellseite stellt die Kooperationsdaten in eine beiden Partnern bekannte Warteschlange. Aus dieser können sie von der Zielseite wieder ausgelesen werden. Beide Systeme sind (abgesehen von den im Workflow definierten Abhängigkeiten) zeitlich voneinander unabhängig. Die Warteschlangen-API ist im Allgemeinen recht komfortabel, zusätzlich gewinnt das Gesamtsystem durch die persistente Speicherung der übertragenen Daten auch an Robustheit [SZ98]. Einstellen und Auslesen in die Warteschlange können (zumindest bei „nicht paranoid" eingestellten Firewalls) durch Applikationen innerhalb der Sicherheitszone ausgeführt werden. Von Nachteil ist lediglich, dass mit dem Warteschlangen-Service ein weiteres System ins Spiel kommt, das hohe Leistung und Verfügbarkeit gewährleisten muss.

3 Anforderungen

In diesem Abschnitt sollen nun die Anforderungen an das zu erstellende System beschrieben werden. Neben der Beschreibung der funktionalen Anforderungen werden wir auch nicht-funktionale Aspekte wie Modularität, Integrationsfreundlichkeit und selbstverständlich Performanz beachten. Aus den in Kapitel 2 beschriebenen Gegebenheiten lassen sich eine Reihe von Anforderungen ableiten.

3.1 Funktionale Anforderungen

Zunächst sollen die Anforderungen beschrieben werden, die die Funktionalität des zu entwickelnden Systems betreffen.

3.1.1 Beschreibung globaler Datenflussabhängigkeiten

In dem zu entwickelnden System sollen globale Datenflussabhängigkeiten modelliert werden können. Hierzu muss es möglich sein, die auf den beteiligten Inseln betroffenen Workflow-Typen auszuwählen, bzw. die am Datenaustausch beteiligten Aktivitäten zu identifizieren. Das Zusammenspiel zwischen Typ- und Instanzen-Ebene ist in Abb.11 zu sehen. Während die Definition der DfA bereits zur Definitionszeit geschieht, muss zur Laufzeit dafür gesorgt werden, dass auch die richtigen Instanzen miteinander kommunizieren.

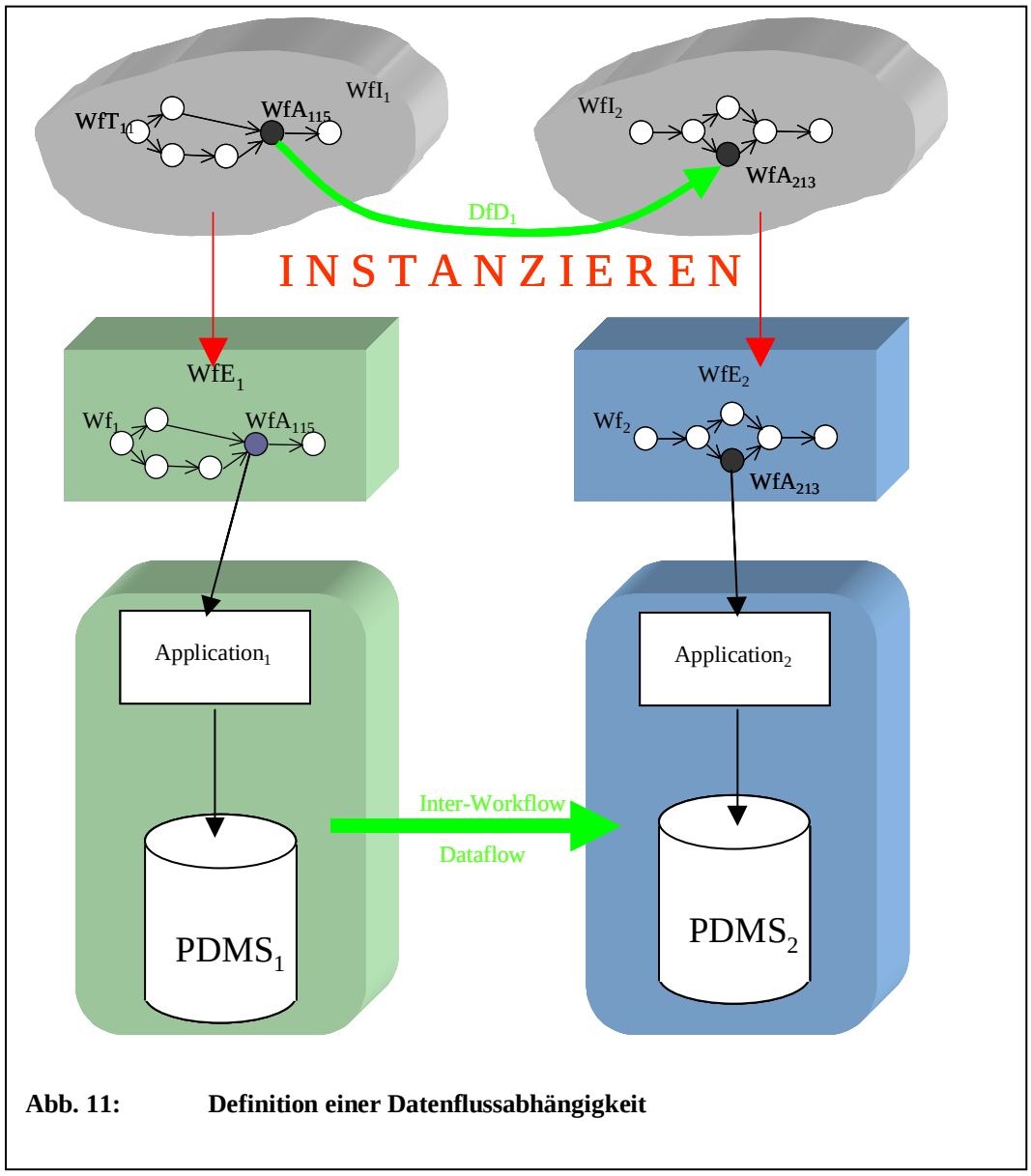

Abb. 11: Definition einer Datenflussabhängigkeit

3.1.2 Freie Beschreibung von Quell- und Zielsystem

Wie wir an den Fallbeispielen gesehen haben, können bei der Abarbeitung globaler Workflows sehr unterschiedliche Systeme zum Einsatz kommen. Es muss daher ermöglicht werden, jedes teilnehmende System in einer Art und Weise zu registrieren, die es ermöglicht, alle für die Zusammenarbeit benötigten Komponenten zur Verfügung zu stellen, um beispielsweise auf Daten eines PDMS, eines DBMS, eines Dokumentenservers

oder aber einer übergeordneten Integrationsarchitektur zugreifen zu können. Weiterhin ist die Angabe eines entsprechenden Zugriffsmechanismus denkbar, der zum externen Datenzugriff genutzt werden kann.

3.1.3 Neutrale Beschreibung der Kooperationsdaten

Ein wichtiger Punkt bei der Beschreibung der Datenflussabhängigkeit besteht im Festlegen der zu Übertragenden Kooperationsdaten. Das System muss die Möglichkeit bieten, die Kooperationsdaten so zu beschreiben, dass Quell- und Zielsystem die Übertragung und Weiterverarbeitung der „richtigen“ Daten durchführen können, wozu ein neutrales Beschreibungsformat zur Verfügung gestellt werden muss. Weiterhin muss die Heterogenität der Systeme muss berücksichtigt werden, da die Kooperationsdaten zum einen verschiedenartig modelliert sein können, zum anderen sogar in verschiedenartigen Systemen gespeichert sein können. Es muss folglich eine Mapping-Komponente zur Verfügung gestellt werden, mit deren Hilfe eine Abbildung vom auf der Quellinsel verwendetet Format auf das der Zielinsel ermöglicht wird.

3.1.4 Freie Wahl der Kommunikationskanäle

Wie in Abschnitt 2.3 beschrieben, gibt es eine Vielzahl von Möglichkeiten zur physischen Übertragung der Daten. Da die gewählte Übertragungsart von den teilnehmenden Systemen bestimmt wird, müssen wir Möglichkeiten vorsehen, wie wir alle diese Möglichkeiten auch unterstützen können.

3.1.5 Kommunikations-Gateway

Durch die freie Wahl der Kommunikationskanäle kann es zu Mischformen kommen. Beispielsweise kann die Quellinsel die Daten über TCP/IP verschicken, während die Zielinsel auf eine Warteschlange zugreifen will. In diesem Fall muss der Koordinator als eine Art Gateway eingreifen. Im genannten Fall spielt er den (synchronen) Empfänger für die sendende Quellinsel, um dann die übertragenen Daten in die entsprechende Warteschlange der Zielinsel einzustellen.

3.1.6 Monitoring

Monitoring spielt aus zweierlei Gründen eine wichtige Rolle. Zum einen soll natürlich die Möglichkeit geboten werden, den Verlauf des globalen Prozesses verfolgen zu können. Zum anderen soll unser System globale Datenflussabhängigkeiten erkennen und Auflösen können. Da jedoch kein globaler Kontrollfluss modelliert werden soll, müssen zu diesem Zweck die lokalen Prozesse beobachtet werden. Die Monitoring-Komponente übernimmt quasi die Kontrolle des Gesamtforschritts und ersetzt somit eine explizite Kontrollflusskomponente.

3.1.7 Schutz vor unbefugtem Zugriff

Wissen um interne Abläufe und natürlich das Datenmaterial an sich stellen für eine Firma ein beträchtliches Kapital dar. Daher muss sowohl für eine gesicherte Übertragung der Kooperationsdaten gesorgt werden (soweit dies mit den gewünschten Kommunikationsmitteln erreichbar ist), weiterhin muss aber auch ein

3.2 Nichtfunktionale Anforderungen

Neben den Anforderungen an die angebotene Funktionalität gibt es noch eine weitere Reihe von Punkten, die bei der Entwicklung des Systems nicht außer Acht gelassen werden sollten.

3.2.1 Gute Integrierbarkeit bestehender Systeme

Viele der Prozesse, die am globalen Gesamtprozess teilnehmen sollen, existieren bereits und werden durch lokale WfMS ausgeführt. Die lokalen Prozessbeschreibungen sollen durch die zusätzliche Definition der globalen Dfa nicht oder nur minimal verändert werden. Es muss folglich eine Möglichkeit gefunden werden, globale DfAs zu modellieren, ohne die lokalen Schemata anzufassen.

3.2.2 Einfache Bedienbarkeit

Wie bei allen komplexen Systemen ist ein Einsatz in der Praxis nur zu erwarten, wenn der Umgang damit sich möglichst unkompliziert gestaltet. Die Definition einer DFA, die Beschreibung der Kooperationsdaten, die Auswahl der Kommunikationskanäle und die Beschreibung des Inselsystems muss auf eingängige, intuitive Weise geschehen und muss durch entsprechende Werkzeuge unterstützt werden.

3.2.3 Hohe Verfügbarkeit

Die ungestörte Ausführung von Geschäfts- und Ingenieursprozessen ist eine für das Unternehmen überlebenswichtiger Aspekt. Daher muss ein System zur Unterstützung globaler Prozesse rund um die Uhr verfügbar sein. Es darf zu keiner Zeit zu Behinderungen der Abläufe durch Ausfälle der globalen Komponente kommen.

3.2.4 Gute Skalierbarkeit

Globale Prozesse spielen eine immer größere Rolle, so dass mit einem enormen Zuwachs an globalen Workflows und teilnehmenden Inselsystemen zu rechnen ist. Das globale System ist so zu gestalten, dass es einen solchen Zuwachs problemlos verkraften kann.

3.2.5 Performanz

Das System muss gewissen Ansprüchen bzgl. der Performanz genügen. Da eine deutliche zeitliche Verbesserung beim Bereitstellen von Kooperationsdaten erzielt werden soll, muss der durch die Automatisierung auftretende Zeitverlust (System-Overhead) klein gehalten werden.

3.2.6 Modularer Aufbau

Die an den globalen Abläufen beteiligten Systeme können prinzipiell eine beliebige Mischform der in Abschnitt 2 beschriebenen Anwendungsfälle bilden. Daher ist es nicht möglich, ein starres System zur Steuerung des Datenflusses zu benutzen. Vielmehr muss es ein ganzes Angebot von Funktionalität angeboten werden, aus dem auf Grund der Systembeschreibungen die entsprechenden Bausteine ausgewählt und kombiniert werden. Durch diesen modularen Aufbau lässt sich flexibel auf die Heterogenität der Systeme eingehen, weiterhin bietet dieser Ansatz die Möglichkeit einer relativ einfachen Erweiterbarkeit, indem neue Module mit der neugeforderten Funktionalität hinzugefügt werden.

4 Zusammenfassung und Ausblick

In diesem Dokument wurde beschrieben, welche Fälle bei der Verarbeitung inselübergreifender Datenflussabhängigkeiten auftreten können. Neben der Unterscheidung zwischen lesendem und schreibendem Zugriff waren dies vor allem auch die Vorplanbarkeit, das Verwenden der Originaldaten oder von Replikaten, der Zeitpunkt der Replizierung, die Art der Datenübertragung und die Vorgehensweise beim Propagieren von Änderungen. Anschließend wurde eine Liste mit Anforderungen an ein System zur automatisierten Behandlung globaler Datenflussabhängigkeiten aufgestellt.
In einem weitern Schritt sollen nun Konzepte entwickelt werden, die dieses Anforderungen erfüllen. Weiterhin ist ein Bewertungskonzept zu entwickeln, um diese Konzepte auf ihre Anwendbarkeit zu überprüfen.

5 Literatur

[BRZ00] Bon, M., Ritter, N., Zimmermann, J.: Interoperabilität heterogener Workflows, Proc. Grundlagen von Datenbanken, 2000: 11-15

[BHR01] Bon, M., Härder, T., Ritter, N.:
Produktdaten-Verwaltung in heterogenen Workflow-Umgebungen, Interner Bericht, Dez. 2001

[BRH02] Bon, M., Ritter, N., Härder, T.:
Sharing Product Data among Heterogeneous Workflow Environments, in Proc. Int. Conf. CAD 2002 - Corporate Engineering Research, Dresden, March 2002, pp. 139-149.

[KRS01] Kulendik, Ottokar; Rothermel, Kurt; Siebert, Reiner: Cross-organizational workflow management - General Approaches and their Suitability for Engineering Processes. In: Schmid, Beat (ed.); Stanoevska-Slabeva, Katarina (ed.); Tschammer, Volker (ed.): Proceedings of the First IFIP-Conference on E-Commerce, E-Business, E-Government : I3E 2001 ; Zuerich, Switzerland, October 3-5, 2001

[Step92] Subcommittee 4 of ISO Technical Committee 184, Product Data Representation and Exchange - Part 11: The EXPRESS Language Reference Manual, ISO Dokument, ISO DIS 10303-11, August 1992

[SZ98] Steiert, H.-P., Zimmermann, J.: JPMQ - An Advanced Persistent Message Queuing Service, in: Advances in Databases, Proc. 16th Nat. British Conf. on Databases (BNCOD16), LNCS 1405, Springer, 1998: 1-18

[VDI99] Verein Deutscher Ingenieure, VDI-Richtlinien VDI2219, Datenverarbeitung in der Konstruktion – Einführung und Wirtschaftlichkeit von EDM/PDM Systemen, November 1999, Beuth Verlag GmbH, 10772 Berlin