Über die Verteilung von Daten in einer verteilten Organisation[1]

Konzept und Nutzung eines Hardware-/Softwaresystems

Prof. Dr. Thorsten Spitta
Universität Bielefeld, Fakultät für Wirtschaftswissenschaften
Postfach 100131, D-33501 Bielefeld

Zusammenfassung. *Es wird das Konzept und die evolutionäre Weiterentwicklung eines verteilten, heterogenen Hardware-/Softwaresystems beschrieben. Kern des Systems mit einem sehr frühzeitigen Einsatz von PCs als Client ist eine Protokoll-Komponente, die die Verteilung von Datenbeständen durch automatische, zeitnahe Replikation erlaubt. Die Komponente wurde im Laufe der bisher sechsjährigen Nutzung zum Replikations-Server weiterentwickelt, der heute die Einbindung von Fertigsoftware-Bausteinen in eine existierende Softwarelandschaft ermöglicht. Im vorliegenden Fall sind dies Funktionen aus SAP R/3. Die Replikations-Technologie erlaubt auch einen ausfallsicheren Datenverbund mit ausländischen Werken, die auf keine stabile Kommunikations-Infrastruktur zurückgreifen können. Der Fall wird dazu benutzt, generelle Aussagen zur Verteilung von Datenbeständen in Industrieunternehmen zu diskutieren, insbesondere synchrone, transaktionale und asynchrone, zeitnahe Buchungen.*

1 Überblick

Verteilte Unternehmensstrukturen gibt es nicht erst, seit man von *verteilten Systemen* spricht, im Gegenteil. Unternehmen mit regional verteilten Standorten sind ab einer gewissen Größenordnung fast die Regel. Sie entstehen in kleineren Firmen weniger durch Auslagerung, als durch Eingliederung bei Fusionen. Hierdurch werden meist neue Informationssysteme erforderlich, die standortübergreifend arbeiten können, also eine verteilte Datenhaltung erfordern. Als Beispiel dient hier ein verteiltes PPS-System. Es entstand in einer Zeit, in der es noch keine Produkte für heterogene Systeme gab (1989/90). Es erwies sich über die ursprünglichen Intentionen hinaus als langfristig so tragfähig, daß es allen organisatorischen Veränderungen des Firmenverbundes über inzwischen 6 Jahre folgen konnte. Dies war vor allem möglich durch eine frühzeitige lose Kopplung verteilter Datenbestände mit Hilfe einer Replikations-Komponente, die es ermöglichte, beliebige Systeme asynchron miteinander zu verbinden. Inzwischen ist sowohl die datentechnische Integration ausländischer Produktionen mit der Zentrale als auch die Einbindung von Fertigsoftware Praxis.

Die Replikations-Komponente koppelt beliebige Datenhaltungssysteme an verschiedenen Standorten auf der Ebene der Anwendung miteinander. Die erfolgreiche Nutzung dieser Technik ist Anlaß zu einer verallgemeinernden Betrachtung, wie die Verteilung der wichtigsten Datenbestände eines Industriebetriebes organisiert sein sollte. Das Ergebnis ist die (sicherlich provokante) These, daß ein Industrieunternehmen synchron gekoppelte verteilte Datenbanken mit Zwei-Phasen-Commit [BEKK84] über Standortgrenzen hinweg gar nicht braucht, dies sogar wegen der Starrheit einer realtime-Kopplung organisatorisch kontraproduktiv ist. Damit wird z. B. die Philosophie des so erfolgreichen Systems *R/3*[2] (vgl. [Sin95]) als langfristig nicht mehr sinnvoll in Frage gestellt, da sie der Idee des verteilten Unternehmens zuwiderläuft. Es muß nicht jeder Geschäftsvorfall realtime gebucht werden, um Organisationen zu integrieren. Eine *zeitnahe* Buchung ist allerdings wünschenswert.

Die Abschnitte 2 bis 6 stellen das Konzept, den Einsatz und die evolutionäre Weiterentwicklung des verteilten PPS-Systems dar. Abschnitt 7 versucht, die Aspekte der Datenverteilung und die dazu erforderliche Technik zu generalisieren.

---

[1] Erschienen: Fachtagung MOBIS, Rundbrief GI-Fachausschuß 5.2, Oktober 1997, 26-30

[2] „R" heißt *realtime* und war auf Großrechner-Plattformen sicher ein großer Fortschritt in der Integration betrieblicher Anwendungssoftware. Die SAP AG arbeitet nach Presseberichten mit Hochdruck an einer Entkopplung.

2 Verteilte Organisation als Anlaß für verteilte Systeme

Das schon erwähnete Unternehmensgebilde entstand Ende 1987 in der Textilbranche (Strümpfe) durch Fusion zweier vorheriger Konkurrenten. Die Logistik des neugebildeten Konzerns mit nunmehr 400 MioDM Jahresumsatz sah sich mit einer hochgradig verteilten Organisation konfrontiert, die ohne integrierende Informationssysteme nicht zu steuern war:

- vier inländische Produktionsstandorte,
- drei Versandstandorte
- zwei Vertriebsorganisationen,
- acht ausländische Produktionsstandorte.

Die Vertriebslogistik war gegen Störungen der Warenversorgung empfindlich, da sie bundesweit über 20.000 Lebensmittel-Läden im Streckengeschäft zu versorgen hatte.

In Großunternehmen damals übliche Zentralrechner-Lösungen mit Stand- und Wählleitungen sollten zum Ende der achziger Jahre nicht noch neu entwickelt werden, da die Konzepte verteilter Systeme bereits bekannt waren (vgl. z.B. [Niss82]) und sich Hardwareentwicklungen abzeichneten, die das ermöglichen würden. Werksrechner-Lösungen mit täglichem Filetransfer waren bereits verbreitet, hier allerdings zu starr, da die Bewegungen in den Daten aufgrund des Massengeschäftes während des Tages sehr groß waren. So lagen die täglichen (!) Bestandsschwankungen im Bereich von 1 bis 1,5 Mio Paar Feinstrumpfhosen. Die Anfang 1989 am Markt verfügbare Informationstechnik wurde den betriebswirtschaftlichen Anforderungen nicht gerecht. Es mußte ein verteiltes Hardware-/Softwaresystem aus möglichst vielen Standard-Komponenten entwickelt werden. Die Frage war nicht, *wie kann man verteilte Systeme in der Organisation nutzen?* (vgl. z. B. [NiRe90]), sondern es war ein verteiltes System gesucht, ohne das die Unternehmensziele einer effizienteren Logistik nicht hätten verfolgt werden können.

3 Anforderungen und Konzept

Benötigt wurde eine für die Massenfertigung und -distribution geeignete Logistik-Software, die in einem ersten Schritt mindestens eine variantenfähige Stammdatenverwaltung, Bestandsführung und Produktionsauftragssteuerung umfassen mußte. Auf dieser sollte später als Kernstück der Logistik-Unterstützung eine die Produktion und den Vertrieb integrierende Steuerung von der Bedarfsermittlung bis zur Warenverteilung aufgebaut werden. Die einzige hierfür in Frage kommende Standard-Software war R/2 von SAP, das allerdings wegen zu hoher Mengengerüste und fehlenden Varianten-Konzeptes nicht einsetzbar war. Eine von SAP angebotende Pilotanwendung des in Entwicklung befindlichen Systems R/3 erschien allein aufgrund des notwendigen Einsatztermins (1991) zu riskant. Es wurde eine Eigenentwicklung entschieden, die mit knappen Budgets folgende strategischen Anforderungen erfüllen mußte:

- Unterstützung einer zentralen Logistik, unter anderem durch aktuelle Bestände,
- Unterstützung der Produktionssteuerung mit komplizierten Belegdruck-Funktionen für die Auftragsverfolgung in den Werken,
- verläßliche Warenausgangsbuchungen (Materialbeistellungen) und -eingangsbuchungen (Halb- und Fertigfabrikate) zu nicht computerunterstützten Werken im Ausland,
- Online-Buchungen in der Produktion in drei Schichten (24 h),
- vom Zentralrechner unabhängige Werksrechner, so daß ein zentraler Ausfall die Produktion nicht stören konnte,
- Einstieg in den Ausstieg aus der Host-Technologie.

Da es durch eine organisatorische Entscheidung („Zentraler Versand“) nicht mehr notwendig war, den Versand ebenfalls auf dedizierten Rechnern abzuwickeln, konnte sich das Konzept für ein verteiltes System im ersten Schritt auf dezentrale Funktionen für die *Produktion* beschränken. Dies

zeigt Bild 1 durch räumliche Zuordnung von Funktionen. Eine später aus dem Host herauszulösende *Versandsteuerung* sollte dem gleichen Konzept folgen.

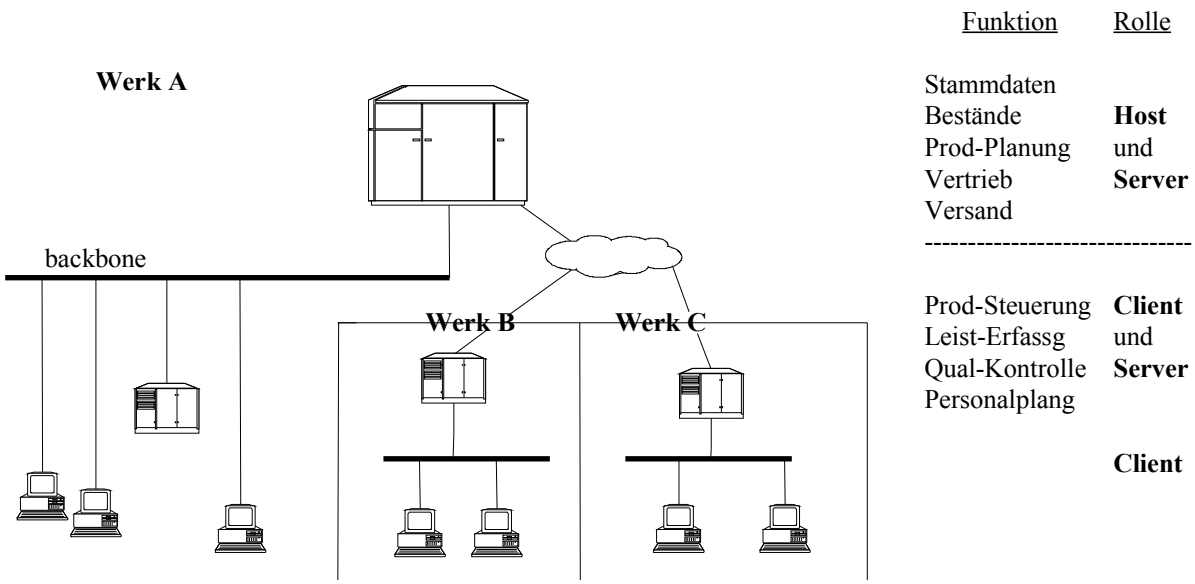

Bild 1: Client-Server-System als verteilte Rechner- und Funktionen-Hierarchie

Damit sich eine zentralen Logistik realisieren ließ, wurden rechnerübergreifende Schnittstellen mit zeitnahem, automatisiertem Datenaustausch benötigt. Mit der Freigabe von Produktionsaufträgen mußten Daten auf die Werksrechner geladen und mit Produktionsende wieder zurückgeladen werden. Die Daten- und Funktionsverteilung zeigt Bild 2.

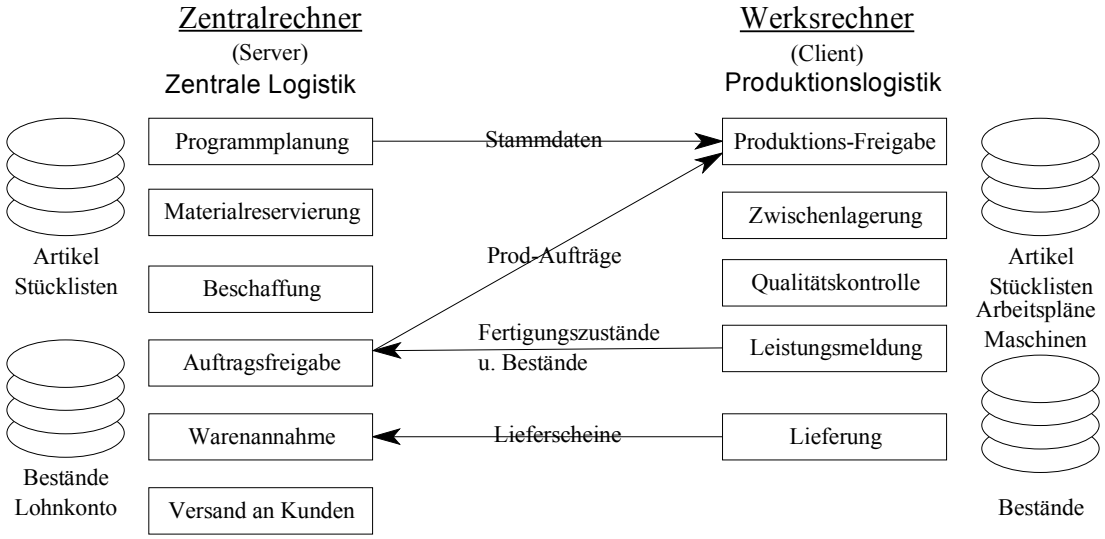

Bild 2: Die Verteilung von Daten und Funktionen auf Rechner und Standorte

Die verteilten Datenbestände in dieser Konfiguration sind Artikel- und Arbeitsplan-Stammdaten, Produktionsaufträge, Material- und Halbfabrikate-Bestände. Für die zentrale Logistik waren vor allem die sich kurzfristig verändernden Produktionsaufträge und die Bestände wichtig, wohingegen als *organisatorischer Schwachpunkt* ausgemacht wurde, daß Stammdaten für bereits eröffnete Produktionsaufträge geändert werden sollten. Die Verwaltung der *zentralen* Stammdaten wurde organisatorisch zentralisiert. Demgegenüber sollten alle Datenzustände *während* der Produktion ausschließlich in der Verantwortung des ausführenden Werkes liegen. Die Bestände wiederum mußten in der Konsolidierungsphase, in der sich der Konzern befand, auch zentral sichtbar gemacht

werden. Zu Beginn der Entwicklung des Systems wurden Bestände von fast 50 Mio DM im Konzern nur einmal im Jahr durch Inventur erfaßt. Die drei zu verteilenden Datenbestände machen folgende grundsätzlichen Aspekte deutlich:

1. Es gibt Datenbestände, bei denen eine **Verteilung** betriebswirtschaftlich überhaupt **nicht wünschenswert** ist. Hier genügt eine Replikation dezentral benötigter Daten. Dies trifft auf die meisten Stammdaten zu wie etwa ARTIKEL oder KUNDE.
2. Andere Datenbestände sollten **überwiegend dezentral** gehalten und gepflegt werden, um eine organisatorische Entzerrung zu erreichen. Es muß aber eine Kopplung zu zentralen, davon abhängigen Daten geben. Diese Kopplung sollte *lose* sein, da eine feste Kopplung über online-Transaktionen das verteilte System schwerfällig macht. Dies gilt vor allem im *Fehlerfall*, etwa dem Ausfall von Rechnern oder Leitungen. Die lose Kopplung war für die Produktionsaufträge sinnvoll und für werkspezifische Stammdaten, z. B. Arbeitspläne. Solche lose gekoppelten Daten werden von nur genau *einem* Klienten gepflegt.
3. Eine wieder andere Kategorie bilden die Bestände. Hier sind bestimmte **abgeleitete Daten zentral** in möglichst aktueller Form erforderlich, etwa für die Materialbedarfsplanung. **Nicht zentral** von Interesse sind jedoch die sehr granularen **primäre**n **Daten**, hier also alle Buchungen und schon gar nicht die Details stellplatzverwalteter Läger. Da sich jedoch die zentralen Daten bei jeder Buchung verändern können, ist hier eine engere Kopplung als im Fall 2 notwendig. Dies gilt auch aus Konsistenzgründen, da die zentralen Daten von *mehreren* Klienten konkurrierend geändert werden können. Dieser Fall wird hier *partiell verteilt* genannt.

In allen drei Fällen ist allerdings die feste (*synchrone*) Kopplung einer verteilten Datenbank mit Mehr-Phasen-Commit-Protokoll [VoGH93] betriebswirtschaftlich nicht erforderlich. Statt dessen kann am Beispiel der dargestellten Fallstudie gezeigt werden, daß weniger feste (*asynchrone*) Kopplungen für die Verteilung von Anwendungen den größeren Nutzen haben, allerdings nur dann, wenn automatische Protokolle ablaufen. Die Lösung in Form einer *Protokoll-Komponente* wurde realisiert von Wolfgang Steinbauer in einem Münchener Softwarehaus [Stei91]. Sie wurde stark beeinflußt von den Konzepten von Jablonski [Jabl90]. Das Kommunikationsprotokoll muß eine Lösung auf der *Anwendungsebene* bieten, nicht nur auf der Transportebene. Es muß die Definition logischer Transaktionen für zusammenhängend zu übertragende Daten ermöglichen und in der Lage sein, geänderte *Sätze* zu replizieren, nicht nur Dateien. Doch was macht dieses System noch heute interessant und was kann aus dieser Entwicklung verallgemeinert werden? Dies sind drei Punkte, von denen hier nur der letzte ausgeführt wird:

1. Die Verwendung einer ausgereiften **Standard-Architektur** auf allen Plattformen (vgl. [Spit96]).
2. Ein **objekttyporientierter Entwurf** ohne direkte Datenschnittstellen (vgl. [Spit89, Kap. 4.3]).
3. Eine **Protokoll-Komponente** für die ausfallsichere Synchronisation miteinander verknüpfter heterogener Datenbanken auf verteilten Rechnern. Hierfür gibt es bis heute keine Produkte.

4 Technische Realisierung

Aus den organisatorischen ergeben sich folgende technischen Anforderungen:

- automatische, operatorlose Replikation geänderter zentraler Daten,
- download dezentral zu pflegender Bestandteile eines verteilten Datenbestandes,
- upload einzelner Attributwerte von Datensätzen (z. B. Fertig- und Leistungsmeldungen),
- relativ zeitnahes Aktualisieren von Sätzen eines partiell verteilten Datenbestandes,
- Ausfallsicherheit der rechnerübergreifenden Replikationen und Buchungen.

Vor allem der letzte Punkt machte ein durchdachtes Protokoll erforderlich, das erheblich über einen „intelligenten Filetransfer“ hinausgeht. Es mußte sichergestellt werden, daß jeder Satz nur *genau einmal* in der Client-Datenbank geändert wird, auch im Fehlerfall. Über Parameter muß dem System mitgeteilt werden können, welche Datenbankzugriffe logische Transaktionen bilden.

Kernstück der Protokoll-Komponente ist ein Zustandsautomat in Form einer Tabelle, in der alle Zustandsübergänge festgehalten werden. Der down- bzw. upload-Vorgang läuft uhrzeitgesteuert automatisch ab, kann aber auch manuell gestartet werden. Das Intervall ist parametrisiert, da das Laufzeitverhalten stark vom tatsächlichen Datenvolumen abhängt. Der Zustandsautomat residiert auf jedem Werksrechner. Er steuert alle Übertragungsvorgänge und die hierfür definierten Transaktionen. Er kennt die logischen Verknüpfungen zwischen lose gekoppelten und partiell verteilten Datenbeständen. Die Module der zentralen Datenbestände konnten aufgrund des objekttyporientierten Entwurfs problemlos um die Operationen *download* bzw. *upload* erweitert werden.

5 Betrieb und Nutzung

Das System wurde im Dezember 1991 in Betrieb genommen und läuft seitdem störungsfrei. Während auf der Host-Seite eine gut ausfallgesicherte und performante Datenbank zur Verfügung stand, konnten die Unsicherheiten des Werksrechner-Datenhaltungssystems (Stand 1989!) im wesentlichen durch das Netzbetriebssystem (Plattenspiegelung) abgefangen werden. Im praktischen Betrieb haben sich DBASE als PC-Datenbasis und Clients unter MS-DOS als genügend zuverlässig erwiesen; die Host-Datenbank war ADABAS. Da die Funktionalität neben dem Aufbereiten von Belegdrucken im wesentlichen aus Buchungen und einer Host-Emulation bestand, reichten Personalcomputer (80286!) als Buchungsplätze für den praktischen Betrieb vollkommen aus.

In der Wartung zeigte sich der Nutzen einer qualitativ hochwertigen Entwicklung. Von 1993 bis Mitte 1996 betrug der gesamte Wartungsaufwand unter 10 % des Entwicklungsaufwandes. Der Anteil fehlerbehebender Wartung an diesem Aufwand betrug unter 5%. Die meisten Änderungen sind der evolutionären Weiterentwicklung des Systems zuzurechnen.

Der für die Einführung in den Werken zentrale Erfolgsfaktor war die Protokoll-Komponente. Gerade in der Anbindung ausländischer Werke mit schlechter nationaler Kommunikations-Infrastruktrur zeigte das Konzept der losen Kopplung seine Stärken. Nach anfänglich halbstündiger Replikation zwischen Zentrale und Werk erwies sich eine Frequenz von fünf Minuten als hochperformant, selbst mit relativ langsamen Wählleitungen. Das System war sicher genug, um auch bei ausfallenden oder gestörten Leitungen oder Servern noch automatisch zu funktionieren. Dies wäre mit einer starr gekoppelten verteilten Datenbank nicht möglich gewesen.

Die Protokoll-Komponente ist inzwischen ohne Änderung der ursprünglich entwickelten Software, insbesondere des Zustandsautomaten, zum *Replikations-Server* geworden, der auch zentral für jede Art von Datenreplikation eingesetzt werden kann. Hierdurch lassen sich vielfältige Verknüpfungen heterogener Systeme vornehmen wie der folgende Abschnitt zeigt.

6 Partielle Einbeziehung von Standardsoftware

Während die erste Entwicklungsstufe - **Fertigungssteuerung in verteilten Produktionsstätten bei variantenreicher Massenfertigung** - auch heute noch nicht zufriedenstellend mit R/3 geleistet werden kann, ergab sich für die zweite Stufe ein anderes Bild. Sie sollte der Logistik eine **Plankomponente** liefern. Diese beginnt beim Absatzplan und verfeinert diesen über den periodengerechten Artikelplan zum variantenbezogenen Produktions-Feinplan. Bekanntlich erfordert ein solches System eine Nettobedarfsermittlung, eine Standardaufgabe der Produktionsplanung. Die Module PP-MRPII mit PP-SOP aus SAP R/3 leisten dies. Die Stammdaten von R/3 lassen sich über den Replikations-Server versorgen, R/3 ist also hier *Sekundärsystem*. Bild 3 zeigt die eben beschriebene Technologie des aus Eigen- und Fremdsoftware verteilten Systems. Es ergeben sich folgende Schnittstellen:

1. Holen Daten aus Eigensoftware oder Nicht-SAP-Fremdsystemen.
2. Senden Daten an SAP.
3. Senden Daten an Werksrechner.

4. Datenaustausch Planer-Client mit SAP-Server.

Das Datenverteilungskonzept mit Hilfe des Replikations-Servers erlaubt einen unkonventionellen und für viele Unternehmen kostengünstigen Weg, funktionale Bestandteile einer integrierten Fertigsoftware zu nutzen. Es besteht *nicht* der sehr aufwendige Zwang, die Stammdatenverwaltung des Fremdsystems in Betrieb zu nehmen. Vielmehr kann man relativ schnell die gewünschten Funktionen nutzen, indem man die Eigensoftware als Primärsystem weiterverwendet. Im konkreten Fall ist dies eine ebenfalls 1989-1991 entwickelte variantenfähige Stammdatenverwaltung.

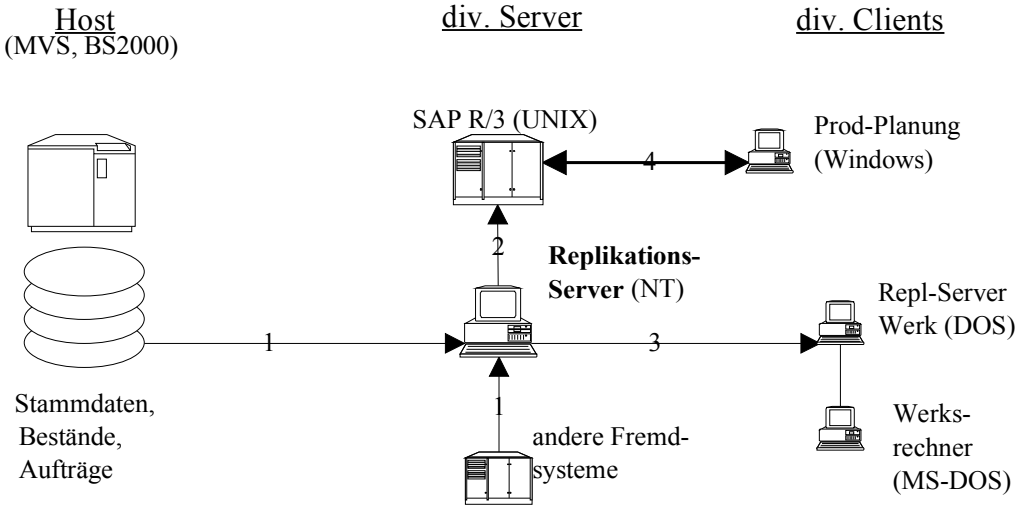

Bild 3: Der Replikations-Server zur Einbindung von Fremdsoftware-Komponenten

Es ist unschwer zu erkennen, daß sich durch den Einsatz von Windows-Clients in den Werken auch andere R/3-Funktionen nutzen lassen, ohne daß die vorhandene DOS-Software neu implementiert werden muß, oder daß *Fremdsystem* z.B. der SAP-Konkurrent BAAN sein kann.

Die Erweiterung um R/3-Funktionen wurde im Oktober 1996 in Betrieb genommen und funktioniert technisch aufgrund der langjährigen Erfahrungen reibungslos. Die vollständige organisatorische Einführung in der zentralen Logistik wird sicher Zeit benötigen, da jetzt ein System zur Verfügung steht, das Produktion und Vertrieb eines modischen Massenfertigers transparent miteinander verbindet. Dies integriert eine sehr konfliktträchtige organisatorische Schnittstelle (Schuldzuweisungen bei Störungen in der Warenversorgung). Es sind erhebliche Effekte in Richtung auf eine Verringerung der Durchlaufzeiten und der Bestände zu erwarten. Damit wurde praktisch bewiesen, daß die mit primitiven Mitteln 1989 begonnene Entwicklung zu Recht als strategischer Einstieg in verteilte Systeme bezeichnet werden kann.

7 Generalisierung

Die Fallstudie ist Anlaß zum Versuch einer Generalisierung. Kann die Behauptung allgemein aufrechterhalten werden, daß synchron gekoppelte verteilte Datenbanken für den normalen industriellen Anwender nicht erforderlich bzw. sogar organisatorisch schädlich sind?

Zweifellos bringt die asynchrone (Replikations-)Technik die Gefahr von Inkonsistenzen mit sich, die bei konkurrierenden Änderungen von Datenbeständen auftritt. In Platzbuchungssystemen oder bei der Führung von Bankkonten mag dies in der Tat ein ernsthaftes Problem sein. In Industrieunternehmen müssen solche Fälle jedoch zunächst *organisatorisch* durchleuchtet werden.

Eine konkurrierende **Stammdatenverwaltung** (s. Abschnitt 3) ist organisatorisch unsinnig. Sie sollte entweder nur zentral erfolgen - mit zeitnaher Replikation der Änderungen - oder dezentral, z. B. bei werkspezifischen Arbeitsplänen, Maschinendaten und Stücklisten. Sollte bei paralleler Fertigung in mehreren Werken nach Produktionsfreigabe eine Änderung notwendig werden, braucht

man entweder ein Versionskonzept der Stammdaten oder eine organisatorische, bewußt nicht maschinelle Abstimmung (etwa bei Totalausfall eines Lieferanten). Eine komplexe verteilte Datenbank mit Mehr-Phasen-Commit löst dieses Problem nicht.

Selbst für den größten dem Autor vorstellbaren Stammdatenbestand, die Adreßdaten einer monopolistischen Telefongesellschaft, wäre es abwegig, einen Änderungskonflikt für Adressen anzunehmen, auch wenn zentrale und lokale Änderungen gleichzeitig erlaubt werden sollten.

Genauer betrachten muß man jedoch **partiell verteilte Datenbestände**. Dies sind Zentrallager-, Werks- und Außendienstbestände, deren Korrektheit und Transparenz für jede Art von Logistik elementar ist. Hier geschehen zwei Dinge: Die Bewegungsdaten (primär) werden dezentral gebucht, da Werksbestände organisatorisch und technisch sinnvoll sind. Sie müssen aber ebenfalls zeitnah in zentrale Bestände gebucht werden, da auf diese häufig lesend zugegriffen wird. Wenn dezentrale Bestände durch Replikation in zentrale verdichtet werden, können relativ große Fehler entstehen, etwa beim Lagerabgang einer großen Lieferung. Hier muß parallel zum Buchen der dezentralen Bestände auch in die zentralen auf der Ebene der Bewegung gebucht werden, um Fehler zu vermeiden. Doch auch dies ist mit Replikation möglich und erfordert keine synchronen Transaktionen.

Die Überlegungen zu partiell verteilten Datenbeständen gelten übrigens prinzipiell gerade auch für sog. *Datawarehouse*-Lösungen informativer, meist hoch verdichteter Datenbanken. Hier tritt das Problem der *update-Anomalien* verschärft auf (vgl. [HMD97], dort insbes. [TrRy97] und [LSH97]). Bei solchen Systemen verbieten sich synchrone Transaktionen allein schon aufgrund der Laufzeiten der Verdichtungen.

8 Fazit

Mit der vorliegenden Fallstudie wurden drei Grundsatzprobleme verteilter Systeme angesprochen. **Erstens** entstehen nutzenstiftende Systeme dieser Art nicht technik-getrieben, sondern aus organisatorischen, unternehmensstrategischen Gründen. Dies ist im Zeichen immer schnellerer, teilweise stark marketing-getriebener Produktzyklen der Informationstechnik nicht überall selbstverständlich. **Zweitens** gibt es starke Indizien, daß lose gekoppelte Systeme, in denen Replikationstechnik eine große Rolle spielt, nützlicher und dazu noch billiger sind als synchron gekoppelte. In dem hier beschriebenen Umfeld ist kein Anwendungsfall aufgetaucht, der ein synchrones Buchen erfordert hätte. Unser Eindruck ist, daß die meisten industriellen Anwender verteilte Datenbanken, die mit der Fähigkeit eines Zwei-Phasen-Commit beworben werden, gar nicht brauchen, es sei denn, aus Performance-Gründen in *einer* Lokation bei sehr großen Installationen. Auch hier sollte der Anwender *erst* seinen Bedarf prüfen und *danach* die Marketing-Aussagen der Anbieter. **Drittens** kann nur eine gut entworfene Software-Architektur die Qualitätsanforderung *Erweiterbarkeit* erfüllen, was durch die Beobachtung einer langjährigen Nutzung gezeigt werden konnte.

Literatur

[BEKK84] *Bayer, R.; Elhard, K.; Kießling, W.; Killer, D.*: Verteilte Datenbanksysteme. In: Informatik-Spektrum 7(1984) 1, 1-19.

[BeDa96] *Beuter, T.; Dadam, P.*: Prinzipien der Replikationskontrolle in verteilten Datenbanksystemen. In: Informatik F&E 11(1996) 4, 203-212.

[Geih93] *Geihs, K.*: Infrastrukturen für heterogene verteilte Systeme. In: Informatik-Spektrum 16(1993) 1, 11-23.

[HMD97] *Handbuch moderne Datenverarbeitung HMD:* Themenheft Data Warehouse, 34(1997) H. 195.

[Jabl90] *Jablonski, S.:* Datenverwaltung in verteilten Systemen. Informatik-Fachberichte 233, Springer, Berlin - Heidelberg et al. 1990.

[LSH97] *Lufter, J.; Schaarschmidt, R.; Küspert, K.:* Aktive Datenbankmechanismen: Stand in Forschung, Produkten und Entwicklung. In: [HMD97], 102-127.

[NiRe90] *Niemeier, J.; Reim, F.:* Welchen Einfluß haben verteilte Informationssysteme auf die betriebliche Organisation? In: Bullinger, H.-J. (Hrg): Verteilte, offene Informationssysteme in der betrieblichen Anwendung, IAO-Forum, Springer, Berlin - Heidelberg et al. 1990.

[Niss82] *Nissing, Th.:* Beitrag zur Entwicklung eines dezentralen Produktionsplanungs- und Steuerungssystems auf der Basis verteilter Datenbestände. Dissertation TH Aachen 1982.

[Sin95] *Sinzig, W.*: Neue Trends bei der Architektur von Standardsoftware am Beispiel Controlling. In: Wirtschaftsinformatik 37(1995) 2, 105-116.

[Spit89] *Spitta, Th.:* Software Engineering und Prototyping. Springer, Berlin - Heidelberg et.al. 1989.

[Spit96] *Spitta, Th.:* „CASE“ findet im Kopf statt - Der Einsatz von Konzepten und Werkzeugen. In: INFORMATIK/INFORMATIQUE 3(1996) 3, 17-25.

[Stei91] *Steinbauer, W.:* Entwicklung eines Systems zur Integration von Anwendungen im Rahmen eines PPS-Konzepts mit zentraler Produktionsauftragsverwaltung und dezentraler Werkstattsteuerung in einem Unternehmen der Textilindustrie. Diplomarbeit FH Würzburg-Schweinfurt, FB Informatik 1991, betreut und durchgeführt bei: Condat GmbH Ottobrunn.

[TrRy97] *Tresch, M.: Rys, M.:* Data Warehouse Architektur für Online Analytical Processing. In: [HMD97], 56-75.

[VoG93] *Vossen, G.; Groß-Hardt, M.:* Grundlagen der Transaktionsverarbeitung. Addison-Wesley, Bonn - Paris et al. 1993.