Datenreplikationstechniken für stationäre und mobile Informationssysteme im Krankenhaus

Heiko Niemann[1], Wilhelm Hasselbring[2]

1. OFFIS
   Escherweg 2
   26121 Oldenburg
   heiko.niemann@offis.de

2. Carl von Ossietzky Universität Oldenburg
   Fachbereich Informatik, Abt. SE
   26111 Oldenburg
   hasselbring@informatik.uni-oldenburg.de

1 Einleitung

Sowohl für stationäre als auch mobile Informationssysteme wurden Datenreplikationstechniken entwickelt. Stationär bedeutet hier, dass die beteiligten Systeme ihre Lokalisationen nicht ändern und i.A. jederzeit über eine Netzanbindung kommunizieren können. Demgegenüber zeichnen sich mobile Systeme dadurch aus, dass sie ihre Lokalisationen leicht ändern können und nicht permanent über eine Netzanbindung mit anderen Systemen interagieren. Durch Netzpartitionierung oder Nichtverfügbarkeit spezieller Systeme tritt im stationären Umfeld ebenfalls das Verbindungsproblem der mobilen Systeme auf. Daher wurden in der Literatur an verschiedenen Stellen stationäre und mobile Systeme verglichen, mit dem Tenor, dass viele Probleme gleich sind, aber verschiedene Lösungen benötigt werden, siehe z.B. [DH95].
In diesem Papier werden Analogien und auch Unterschiede zwischen stationären und mobilen Systemen diskutiert. Bei dem Vergleich wird der Schwerpunkt auf die Datenreplikation gelegt. Krankenhausinformationssysteme (KIS), die in der Praxis immer aus vielen verschiedenen Systemen bestehen, dienen uns als Anwendungsszenario. Gerade hier werden hohe Ansprüche an die Replikationsverfahren gestellt, da einerseits wegen der Konsistenzanforderung eine enge Kopplung der beteiligten Systeme nötig ist, andererseits darf wegen der geforderten Autonomie die Kopplung nicht zu eng sein [Ha97]. Da auch in Kliniken mobile Geräte Einzug halten [Bu98], wollen wir die in diesem Kontext diskutierten Replikationsstrategien mit den Techniken der mobilen Datenbanken kombinieren.

2 Datenreplikation

Hinsichtlich Datenreplikation soll einerseits betrachtet werden, welche Daten repliziert werden, und andererseits, mit welchen Verfahren sie aktualisiert werden. Die Verfahren (Beschreibung z.B. in [Da96]) unterteilen wir in synchrone und asynchrone Replikation.

	Stationäre Systeme	**Mobile Systeme**
Datenreduktion	kaum	domänenspezifisch
Synchrone Replikation	hohe Konsistenzanforderung	im Online-Modus
Asynchrone Replikation	hohe Autonomieanforderung	im Online- und Offline-Modus

Tabelle 1 Datenreplikation bei stationären und mobilen Systemen

In stationären Systemen werden die gemeinsamen Daten i.d.R. komplett repliziert, d.h. die Datenreduktion spielt keine Rolle. Da die Ressourcen, wie z.B. Speicherplatz oder Rechenleistung, von mobilen Systemen häufig eingeschränkt sind, wird dort eher selten der komplette Datenbestand repliziert, sondern es wird auf die aktuell benötigten Daten für das jeweilige mobile Gerät reduziert [Lu00]. Für unser Anwendungsszenario KIS bedeutet dies, dass die stationären Systeme mehr oder weniger die kompletten Patientendaten beherbergen, während die mobilen Systeme mit einem Teil des Datenbestandes ausgestattet sind, z.B. die aktuellen Daten der Patienten einer Station.
Synchrone Replikation, d.h. die zeitgleiche Änderung der notwendigen Replikate, bietet Konsistenz durch Verfahren wie das ROWA-Verfahren (Read One, Write All), das Primary-Copy-Verfahren oder die Quorum- bzw. Votierungsverfahren. Demgegenüber bietet die asynchrone Replikation, d.h. die zeitversetzte Aktualisierung der Replikate, eine höhere Autonomie bei zumindest temporärer Inkonsistenz. Beispielsweise können beim Peer-To-Peer-Verfahren alle Kopien geändert werden, die Aktualisierung wird asynchron vorgenommen und es werden somit bewusst Konflikte in Kauf genommen. Bei der Zusammenführung müssen dann die möglicherweise aufgetretenen Konflikte behandelt werden.
Während bei stationären Systemen in Abhängigkeit der Anforderungen zwischen synchroner und asynchroner Replikation gewechselt werden kann, hängt die Strategie bei mobilen Systemen in erster Linie davon ab, ob das System „online“ oder „offline“ ist. Im Online-Modus können grundsätzlich beide Varianten zum Einsatz kommen, während im Offline-Modus nur die asynchrone Propagierung in Frage kommt.
Die stationären Systeme eines KIS stellen sowohl Konsistenz als auch Autonomie als Anforderung, so dass eine Kombination der synchronen und asynchronen Replikation von Interesse ist (siehe [NH02]). Bei zusätzlichem Einsatz mobiler Geräte in Kliniken muss der asynchronen Replikation ein erhöhter Stellenwert zugewiesen werden. Zwar sind in dieser Anwendungsdomäne die mobilen Systeme durch drahtlose Verbindungstechniken wie z.B. Bluetooth in der Lage, häufig online zu sein, aber durch Störungen des Funkverkehrs oder einfaches Abschalten des mobilen Gerätes ist des öfteren eine Nichterreichbarkeit zu erwarten.

3 Konfliktmanagement

Die asynchrone Replikation, ob im stationären oder mobilen Fall, birgt die Gefahr der Konflikte. Bei der Zusammenführung, die auch Synchronisation genannt wird, müssen diese Konflikte behandelt werden. Beim Konfliktmanagement fließt i.A. semantisches Wissen aus dem Anwendungsbereich ein.

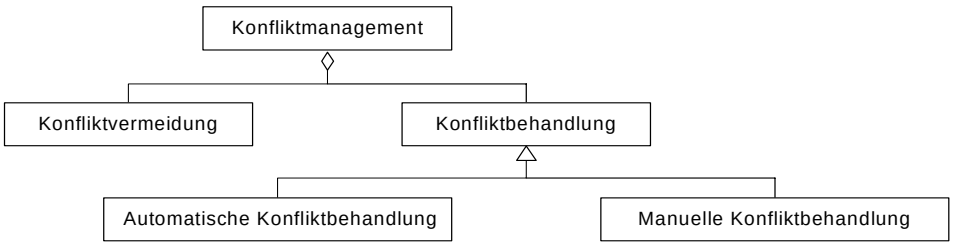

Abbildung 1 Konfliktmanagement (dargestellt in der UML in Anlehnung an [Ha99])

In Abbildung 1 wird gezeigt, dass sich Konfliktmanagement aus Konfliktvermeidung und Konfliktbehandlung zusammensetzt. Wenn Konflikte erkannt werden, so müssen diese Konflikte entweder automatisch oder manuell behandelt werden.

3.1 Konfliktvermeidung

Eine Vermeidung von Konflikten kann dadurch erreicht werden, dass

- das ändernde System die anderen Kopien während der Änderung und Propagierung sperrt bzw. an einer zentralen Stelle eine Sperrung beantragt (nur im „Online"-Zustand möglich) oder
- ein ausgewähltes System ein „Token" erhält (auch im „Offline"-Zustand möglich; Wechsel des Token nur im „Online"-Zustand).

Die erste Variante ist für mobile Systeme kaum praktikabel, während die zweite Variante durchaus auch in mobilen Datenbanken angewandt wird. Hierunter fällt z.B. eine Sperrung der entsprechenden Objekte auf der zentralen Datenbank, solange ein mobiles Gerät diese Objekte heruntergeladen hat. Diese mitunter langanhaltenden Sperrungen müssen natürlich für die Anwendung akzeptabel sein (vgl. auch langanhaltende Transaktionen). Für stationäre Systeme, insbesondere wenn es sich um autonome Systeme handelt, spielt die Konfliktvermeidung eine eher untergeordnete Rolle, da das autarke Arbeiten eingeschränkt wird.

3.2 Konfliktbehandlung

Wenn nicht alle Konflikte vermieden werden können, so müssen sie nach ihrer Erkennung, wofür die Synchronisationskomponente verantwortlich zeichnet, behandelt werden. Diese Konfliktbehandlung kann automatisch z.B. über die Synchronisationskomponente oder manuell erfolgen.

Im Rahmen des asynchronen Peer-To-Peer-Replikationsverfahrens wurden die Möglichkeiten der automatischen Konfliktbehandlung bei der Zusammenführung diskutiert, z.B. in [DGS83; Le97]. Grundsätzlich können folgende Methoden unterschieden werden:

- Ermittlung einer überlebenden Änderung: An Hand spezieller Kriterien wie Zeitstempel, Versionskontrolle oder Prioritäten wird die Änderung bestimmt, die gültig bleiben soll, während die anderen Änderungen verworfen werden.
- Berechnung einer resultierenden Änderung: Nach speziellen Methoden wird auf Basis aller Änderungen eine gemeinsame Änderung berechnet.

Nicht alle Konflikte können automatisch behoben werden, so dass u.U. ein manueller Eingriff nötig ist. In diesem Fall greift z.B. der Anwender oder ein Administrator ein, um das System wieder in einen konsistenten Zustand zu überführen. Das Konfliktmanagementsystem kann dabei z.B. folgendermaßen unterstützen:

- An Hand eines Log-Protokolls wird der Konflikt gemeldet und manuell korrigiert.
- Die auf einem mobilen Gerät vorgenommen Änderungen werden zunächst provisorisch vorgenommen. Bei der Synchronisation werden diese Änderungen dann gültig gesetzt oder mit einer entsprechenden Meldung an den Anwender abgewiesen (vgl. das Zwei-Schichten-Verfahren in [GHO96]).
- Die in Konflikt stehenden Daten werden zunächst ins System übernommen und besonders gekennzeichnet. Anschließend nimmt der Anwender eine Korrektur vor.

4 Synchronisation über einen Replikationsmanager

Wie in Abschnitt 2 dargestellt, stellen Krankenhausinformationssysteme besondere Ansprüche an die Datenreplikation. In [NH02] wird u.a. für diesen Kontext eine adaptive Replikationsstrategie für stationäre Informationssysteme vorgestellt, die von einem Replikationsmanager realisiert wird. Der Replikationsmanager kann auf Grund parametergesteuerter Kriterien wie Verfügbarkeit, Tageszeit, Last oder Performanz zwischen synchroner und asynchroner Replikation wechseln. Da auch der Einsatz mobiler Geräte in großen Universitätskliniken denkbar ist oder schon praktiziert wird, z.B. können Ärzte während der Visiten auf den Stationen Daten über PDAs pflegen [Bu98], kann die Synchronisation dieser mobilen Geräte mit den anderen Systemen des KIS über den Replikationsmanager erfolgen, wie in Abbildung 2 angedeutet.

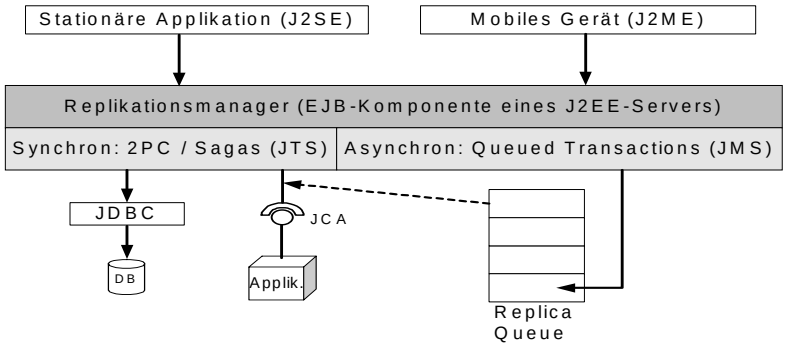

Abbildung 2 Replikationsmanager

Für die Implementierung des Replikationsmanagers kann die Java-Technologie J2EE verwendet werden. Der Replikationsmanager selbst wird als EJB-Komponente auf einem EJB- bzw. J2EE-Server realisiert. Für die synchrone Replikation kann entweder das Transaktionskonzept 2-Phasen-Commit (2PC), sofern die beteiligten Systeme dieses in Form des XA-Protokolls unterstützen, oder das offen-geschachtelte Transaktionskonzept Sagas unter Ausnutzung des Java Transaction Service (JTS) genutzt werden. Die asynchrone Replikation kann transaktional durch Queued Transactions unterstützt werden, wobei der Java Message Service (JMS) für die Umsetzung herangezogen wird. Hierbei wird eine Replikationsanforderung zunächst in die „Replica Queue“ gestellt, von wo sie der Replikationsmanager zu einem späteren Zeitpunkt abruft und die entsprechenden Systeme aktualisiert (durch den gestrichelten Pfeil angedeutet).
Falls bei der Aktualisierung direkt auf die Datenbank einer Anwendung zugegriffen werden darf, kann dieser Zugriff über JDBC erfolgen. Muss über Schnittstellen auf eine Anwendung zugegriffen werden, wie z.B. bei SAP R/3, kann die Java Connector Architecture (JCA) zum Einsatz kommen. In diesem Fall wird ein Ressourcenadapter für die jeweilige Applikation benötigt. Die Anwendungen, die eine Replikationsanforderung an den Replikationsmanager stellen, können im Fall stationärer Systeme mittels J2SE und im Fall mobiler Systeme wegen der eingeschränkten Ressourcen mittels J2ME realisiert werden.
Da beim Einsatz mobiler Geräte die Konfliktgefahr grundsätzlich erhöht wird, sollte der Replikationsmanager über ein ausgereiftes Konfliktmanagement verfügen. An dieser

Stelle muss semantisches Wissen der Domäne einfließen. Z.B. die Forderung nach Autonomie von KIS verbietet langes Sperren, so dass Konfliktvermeidung nicht in Frage kommt. Eine automatische Konfliktbehandlung könnte hier durch die Ermittlung der überlebenden Änderung geschehen, wobei diejenige Änderung als gültig gesetzt wird, die z.B. von dem behandelndem Arzt durchgeführt wurde. Falls aus Sicherheitsgründen ein manueller Eingriff bevorzugt wird, so kann bei der Synchronisation der Konflikt vom Replikationsmanager angezeigt werden, worauf der Anwender die überlebende Änderung, ggf. nach Rücksprache, bestimmt.

5 Zusammenfassung und Ausblick

An Hand der Datenreplikation in Krankenhäusern werden die Unterschiede und Gemeinsamkeiten von stationären und mobilen Systemen diskutiert, wobei ein Schwerpunkt auf das für die asynchrone Replikation relevante Konfliktmanagement gesetzt wird. Als Konsequenz kann festgehalten werden, dass sich die Forschungsbereiche der mobilen Systeme und der verteilten Datenbanken ergänzen. Für Krankenhausinformationssysteme, die sowohl stationäre als auch mobile Systeme aufweisen können, wird diese Synergie am Beispiel eines adaptiven Replikationsmanagers demonstriert. Für die Implementierung des Replikationsmanagers bietet sich die Java-Technologie J2EE mit den entsprechenden Diensten sowie J2SE und J2ME an.
In diesem Beitrag werden primär mobile Geräte betrachtet, die über keine permanente Netzanbindung verfügen (z.B. PDAs). Im Krankenhaus ist aber auch eine drahtlose Vernetzung möglich, so dass die mobilen Geräte auch online auf das KIS zugreifen können. Eine Kombination synchroner und asynchroner Datenreplikationstechniken erlaubt es, flexibel auf solche Möglichkeiten zu reagieren.

Literaturverzeichnis

[Bu98] Buchauer, A.: Integration mobiler Informationswerkzeuge in heterogene Krankenhausinformationssysteme. Ruprecht-Karls-Universität Heidelberg, Institut für Medizinische Biometrie und Informatik, Abteilung Medizinische Informatik, 1998

[Da96] Dadam, P.: Verteilte Datenbanken und Client/Server-Systeme. Springer-Verlag, 1996

[DGS83] Davidson, S.; Garcia-Molina, H.; Dale, S.: Consistency in Partitioned Networks. In: ACM Computing Surveys 17 (3), 1985, S.341-370

[DH95] Dunham, M.; Helal, A.: Mobile Computing and Databases: Anything New? In: SIGMOD Record 24 (4), 1995, S.5-9

[GHO96] Gray, J.; Helland, P.; O'Neil, P. et al.: The Dangers of Replication and a Solution. In:ACM Sigmod Conference, Montreal, 1996 S.173-182

[Ha97] Hasselbring, W.: Federated integration of replicated information within hospitals. In: International Journal on Digital Libraries 1 (3), 1997, S.192-208

[Ha99] Hasselbring, W.: On Defining Computer Science Terminology. In: Communications of the ACM 42 (2), 1999, S.88-91

[Le97] Lenz, R.: Adaptive Datenreplikation in verteilten Systemen. Teubner, Leipzig 1997

[Lu00] Lubinski, A.: Replizieren und Reduzieren von Daten für ressourcenbegrenzte Umgebungen. In:12. GI-Workshop Grundlagen von Datenbanken, Plön, 2000 S.5

[NH02] Niemann, H.; Hasselbring, W.: Adaptive Replikationsstrategie für heterogene, autonome Informationssysteme. In:14. Workshop über Grundlagen von Datenbanken, Fischland, 2002