(19) Bundesrepublik Deutschland
Deutsches Patent- und Markenamt

(10) **DE 44 97 149 B4** 2005.02.10

(12) **Patentschrift**

(21) Deutsches Aktenzeichen: **P 44 97 149.4**
(86) PCT-Aktenzeichen: **PCT/US94/10093**
(87) PCT-Veröffentlichungs-Nr.: **WO 95/08809**
(86) PCT-Anmeldetag: **09.09.1994**
(87) PCT-Veröffentlichungstag: **30.03.1995**
(43) Veröffentlichungstag der PCT Anmeldung in deutscher Übersetzung: **17.10.1996**
(45) Veröffentlichungstag der Patenterteilung: **10.02.2005**

(51) Int Cl.[7]: **G06F 17/30**

Innerhalb von 3 Monaten nach Veröffentlichung der Erteilung kann Einspruch erhoben werden.

(30) Unionspriorität:
126586 24.09.1993 US

(71) Patentinhaber:
Oracle Corp., Redwood City, Calif., US

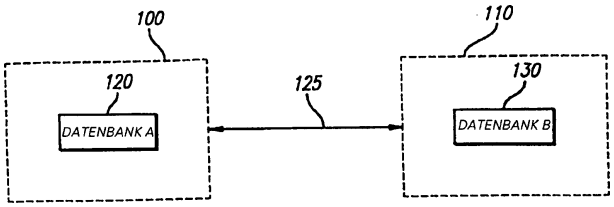
(74) Vertreter:
Betten & Resch, 80333 München

(72) Erfinder:
Jain, Sandeep, Belmont, Calif., US; Daniels, Dean, Fremont, Calif., US

(56) Für die Beurteilung der Patentfähigkeit in Betracht gezogene Druckschriften:
US 48 75 159 A
US 48 53 843 A
US 46 48 036 A
US 46 31 673 A
US 40 77 059 A
US 51 70 480
IBM Technical Disclosure Bulletin, Vol. 33, No.6B, Nov. 1990, S. 395/396;

(54) Bezeichnung: **Computerbezogenes Verfahren zur Datenreplikation in Peer-to-Peer-Umgebung**

(57) Hauptanspruch: Computerbezogenes Verfahren zur Datenreplikation in Peer-to-Peer-Umgebung (Umgebung mit gleichrangigen Arbeitsstationen), wobei das Verfahren folgende Schritte umfaßt:
Durchführen von Datenmodifikationen in einem ersten Computersystem und in einem zweiten Computersystem;
Erzeugen von Ausbreitungs- oder Weitergabeinformationen, die mit den Datenmodifikationen in dem ersten und zweiten Computersystem verknüpft sind, wobei die Ausbreitungs- bzw. Weitergabeinformationen wiederauffindbar und modifizierbar sind und Informationen einschließen, die eine in einem anderen Computersystem auszuführende Operation angeben; und
Durchführen einer bidirektionalen Replikation der Datenmodifikationen unter Verwendung der Ausbreitungs- bzw. Weitergabeinformationen, wobei die bidirektionale Replikation zwischen dem ersten Computersystem und dem zweiten Computersystem stattfindet und durch entweder das erste oder das zweite Computersystem initiiert wird, und wobei die bidirektionale Replikation in konsistenten Daten bezüglich des ersten und zweiten Computersystems resultiert,
Identifizieren von Ausnahmeereignissen während der Durchführung der Datenmodifikationen, wobei der Schritt des Durchführens von Datenmodifikationen von einer Prozedur durchgeführt wird und der Schritt des Identifizierens der Ausnahmeereignisse in...



Beschreibung

HINTERGRUNG DER ERFINDUNG

GEBIET DER ERFINDUNG

[0001] Diese Erfindung betrifft das Gebiet des Replizierens von Daten.

Stand der Technik

[0002] Weil Computersysteme an einem Ort mit anderen Orten kommunizieren können, greifen Computeranwendungen in wachsendem Maß auf Daten zu, die an vielen lokalen und entfernten Systemen angeordnet sind. In einem Nurlese-Modus (d.h. keine Datenmodifikationen), können Mehrfachkopien desselben Datenfeldes an mehreren Orten angeordnet sein, ohne daß Probleme bezüglich der Datenintegrität größer werden. Jedoch wird dann, wenn mehrere Benutzer an mehreren Systemorten beginnen, eine oder mehrere der Kopien derselben Datenfelder zu modifizieren, die Datenintegrität ein kritisches Problem.

[0003] Idealerweise sollte jeder Benutzer eine beliebige Kopie eines Datenfeldes modifizieren können, einhergehend mit der Möglichkeit, die Modifikation jeder Kopie desselben Datenfeldes automatisch zu einem beliebigen Ort weitergeben zu können. Bei Systemen nach dem Stand der Technik fehlt ein Bereitstellen dieser Art von Umgebung gleich behandelbarer Kopien (Peer-to-Peer-Umgebung).

[0004] Beispielsweise bieten einige Systeme eine "Master"-Kopie mit mehreren "Slave"-Kopien. Irgendwelche Abänderungen werden am "Master" durchgeführt und die "Slave"-Orte empfangen eine Kopie der abgeänderten Daten, nachdem die Abänderung am "Master" beendet ist. Somit muß ein Benutzer am "Slave"-Ort zum Abändern eines Datenfeldes auf die "Master"-Kopie zugreifen. Diese Technik bietet nicht die Möglichkeit, jede Kopie zu erneuern und die an jener Kopie durchgeführten Änderungen bzw. Modifikationen an alle anderen Kopien weiterzugeben.

[0005] Beispielsweise offenbart das US-Patent 4,077,059 von Cordi et al. ein Mehrfachverarbeitungssystem mit einem hierarchischen Speicher mit Protokollführung und Kopie-Hintergrundspeicher (copyback). Das hierarchische Speichersystem hat zwei Speichereinheiten auf jeder Ebene. Eine Haupteinheit enthält alle Daten für die Ebene und die andere Einheit, die Kopie-Hintergrundspeichereinheit, enthält die Änderungen, die an jenen Daten entweder durch eine Addition oder eine Modifikation durchgeführt worden sind. Die Haupteinheit hat eine Schnittstelle zu der nächsthöheren Ebene in der Hierarchie und ihrer Verarbeitungseinheit. Die Kopie-Hintergrundspeichereinheit überträgt die Datenänderungen zur niedrigeren Ebene, wenn die Haupteinheit der niedrigeren Ebene keine Schnittstelle zur nächsthöheren Ebene oder zur Verarbeitungseinheit bildet. Die Kopie-Hintergrundspeichereinheit ist kleiner als die Haupteinheit, um die nötigen Speichereinheiten auf jeder Ebene zu reduzieren. Da die Kopie-Hintergrundspeichereinheit kleiner als die Haupteinheit ist, ist es möglich, daß die Anzahl von Änderungen an den Daten der Haupteinheit die Kapazität des Kopie-Hintergrundspeichers übersteigt. Ein Überwachungsprogramm wird dazu verwendet, sicherzustellen, daß die Anzahl von Änderungen die Speicherkapazität der Kopie-Hintergrundspeichereinheit nicht übersteigt. Geeignete Maßnahmen werden unternommen, um die Anzahl von Änderungen im Kopie-Hintergrundspeicher zu reduzieren, wenn der Kopie-Hintergrundspeicher eine bestimmte Kapazität erreicht.

[0006] Das US-Patent 4,558,413 von Schmidt et al. offenbart ein Verwaltungssystem zum Verwalten von Softwareversionen, die erneuert bzw. upgedated sind und in bestimmten Speichervorrichtungen in einer verteilten Softwareumgebung eines lokalen Netzwerkes gespeichert sind. Das System sammelt und rekompiliert Versionen eines Softwareobjekts (d.h. Quellen- und Objektmodule), die an diesen Speichervorrichtungen im LAN angeordnet sind. Das kompilierte Programm wird in der verteilten Softwareumgebung verwendet. Das System enthält die zugehörigen Softwareobjekte: Version, eindeutiger Name, Erneuerungschronologie, Abhängigkeiten von (und Verbindungen zu) anderen Softwareobjekten und Standort.

[0007] Das Verwaltungssystem wird automatisch benachrichtigt, wenn Änderungen an einem Softwareobjekt durchgeführt werden.

[0008] Das US-Patent 4,631,673 von Haas et al. offenbart ein Verfahren zum Auffrischen von Tabellen mit mehreren Spalten in einer relationalen Datenbank. Haas zeigt ein Verfahren zum Auffrischen der Momentaufnahme (d.h. einer Nurlese-Kopie eines Basistabellenteils). Jeder Datensatz einer Basistabelle muß folgendes aufweisen: (1) einen Tupel-(d.h. eindeutigen)-Identifizierer TID, (2) die vorherigen Datensätze TID, PREVTID,

und (3) eine "Zeitmarkierung" UID der letzten Abänderung der Datensätze. Die Momentaufnahme enthält den TID des entsprechenden Basistabellen-Datensatzes. Beim Auffrischen sendet die Momentaufnahme-Stelle die höchste UID, die von der Momentaufnahme gesehen wird. Die Basistabellenstelle identifiziert Änderungen basierend auf Werten von TID, PREVTID und UID. Haas beschreibt eine Master-Slave-Umgebung, durch die Änderungen an einem Master an Kopien weitergegeben werden.

[0009] Das US-Patent 4,635,189 von Kendall beschreibt ein verteiltes Echtzeit-Datenbankverwaltungssystem, das in einem Speicher eines Prozessors Kopien der Variablen speichert, die zum Laufenlassen der Programme in jenem Prozessor benötigt werden. Wenn eine Variable erzeugt wird, wird ein Prozessor zu dem Prozessor bestimmt, der den Wert jener Variable festlegt. Jede Variablenkopie wird auf periodischer Basis oder beim Auftreten eines definierten Zustandes durch den aktuellen Wert des ursprünglichen Wertes erneuert. Kendall beschreibt ein Verfahren einer derartigen Datenmanipulation, daß ein erster Prozessor eine ursprüngliche Variable in einem zweiten Prozessor adressieren kann und den aktuellsten Wert jener Variable derart führen kann, daß er beim Auftreten eines Zustandes in einem vierten Prozessor in einem dritten Prozessor gespeichert wird. Dann kann eine Bestätigungsnachricht zu einem fünften Prozessor gesendet werden.

[0010] Das US-Patent 4,646,229 von Boyle beschreibt ein Datenbanksystem, das zukünftige Versionen der Datenbank zur Anwendung bei zeitorientierten Anwendungen enthält, wie beispielsweise einer Anwendung zum Planen der Verwendung derselben Einrichtungen für gegenwärtige und zukünftige Benutzer. In der Datenbank wird die gesamte Information gehalten, die zum Darstellen der Datenbank-Inhalte zu gewünschten zukünftigen Zeitpunkten erforderlich ist. Alle Transaktionen (z.B. logische Arbeitseinheiten) werden bezüglich der Zeit markiert, um einen Zugriff auf die richtige Version der Datenbank sicherzustellen.

[0011] Das US-Patent 4,648,036 von Gallant betrifft ein Verfahren zum Steuern einer Anfrage- und Erneuerungsverarbeitung in einem Datenbanksystem. Insbesondere beschreibt Gallant ein Verfahren zum Sicherstellen, daß auf eine Anfrage hin Information empfangen wird, die die Datenbank entweder vor oder nach einer Erneuerung darstellt, nicht aber Information, die einen Zustand darstellt, nachdem die Erneuerung beginnt aber noch nicht beendet ist. Transaktionsmodifikationen werden bezüglich einer zukünftigen Datenbankstruktur durchgeführt. Am Ende der Modifikationen bzw. Abänderungen wird eine Umschaltung von der aktuellen Datenbankstruktur zur zukünftigen Datenbankstruktur durchgeführt. Ein Anfrageprozeß greift auf die aktuelle Datenbankstruktur zu.

[0012] Das US-Patent 4,714,992 von Gladney et al. betrifft ein Verfahren zum Verwalten einer Überalterung von Replikationen von Datenobjekten in einem verteilten Verarbeitungssystem. Datenbankobjekte an einer Quellenstelle werden auf eine Replizierstelle repliziert. Wenn an der Quellenstelle gespeicherte Objekte geändert werden, werden entsprechende Objekte an der Replizierstelle veraltet. Eine Replizierstelle erzeugt eine Anfrage nach einer Liste veralteter Objekte von der Quellenstelle. Gladney beschreibt eine Einrichtung zum Identifizieren der veralteten Objekte, zum Weiterleiten der Identifizierung der veralteten Objekte und zum Entfernen der veralteten Objekte von der Replizierstelle. Gladney beschreibt eine Master-Slave-Umgebung, durch die Änderungen an einem Master zu den Replikationen weitergegeben werden.

[0013] Das US-Patent 4,853,843 von Ecklund et al. beschreibt eine Mehrfachversions-Datenbank, wobei jede Erneuerungsoperation eine neue Version der Datenbank erzeugt und die älteren Versionen verfügbar bleiben. Mehrere alternative Versionspfade werden zurückbehalten. Ecklund beschreibt ein Verfahren zum Ableiten einer minimalen Gruppe von alternativen Versionspfaden. Wenn ein Teil erneuert wird, werden die Erneuerungen an mehreren Stellen in den Teilen synchron durchgeführt. Eine aus den Datenbankvorgeschichten abgeleitete Änderungsliste und virtuelle Teiländerungs-Vorgeschichten werden dazu verwendet, das Auftreten von Konflikten festzustellen. Ecklund beschreibt ein System zum Verbinden mehrerer Versionen eines Datenobjekts in eine verteilte Datenbank, so daß jeder Erneuerungsteil auf seine eigene Version zugreifen kann.

[0014] Das US-Patent 4,875,159 von Carev et al. beschreibt ein System zum Synchronisieren zweier Versionen von Dateien in einem Mehrprozessorsystem. Beide Versionen enthalten ein Synchronisationsende-Steuerfeld und ein Synchronisation-in-Arbeit-Feld. Das Synchronisationsendefeld zeigt an, daß die zugehörige Version synchronisiert ist, wenn sie eingestellt ist. Das Synchronisation-in-Arbeit-Feld zeigt an, daß die zugehörige Version gerade synchronisiert wird, wenn sie eingestellt ist. Wenn das Synchronisationsendefeld in einer oder beiden der Versionen gelöscht wird, wird das Synchronisation-in-Arbeit-Feld in einer der Versionen eingestellt. Dann wird eine temporäre Datei erzeugt, und eine Kopie der einen Version wird zur temporären Datei übertragen. Das Synchronisation-in-Arbeit-Feld der einen Version wird untersucht, nachdem die Übertragung beendet ist. Wenn das Synchronisation-in-Arbeit-Feld eingestellt ist, wird das Synchronisationsendefeld in der temporären Version eingestellt. Der temporären Version wird der Name der anderen der Versionen gegeben, und das

Original dieser Version wird entfernt.

[0015] EPO 0,428,264 A2 von Boykin et al. offenbart ein Verfahren zum Erzeugen eines Zugriffsplans in einem Datenbanksystem, das einen Zweiebenencode zum Durchführen zuvor ausgewählter Beschränkungsprüfungen enthält. Datenbankzugriffsbefehle werden in Zugriffspläne kompiliert, die zur Laufzeit anstelle der Zugriffsbefehle zum Verbessern der Systemleistungsfähigkeit ausgeführt werden.

[0016] Roussopoulos, N. und Kang, H., "Principles and Techniques in the Design of ADMS±", IEEE, Dezember 1986, S. 19–25 beschreibt eine Technik zum Herunterladen von Datenbankobjekten von einem Großrechner zu einer Workstation, wenn die Workstation auf Daten im Großrechner zugreift. Ein Zugriff auf die lokale Datenbankuntergruppe wird an der Workstation durchgeführt. Datenbankobjekte, auf die von mehreren Workstations zugegriffen wird, sind global angeordnet. Erneuerungen am Großrechner werden protokolliert und inkrementell zugeführt, bevor eine Abfrage der involvierten Daten durchgeführt wird. Änderungen an den heruntergeladenen Objekten werden durch Beibehalten eines Sicherungsprotokolls durchgeführt, das aus Einträgen besteht, von denen jeder aus einer Operation (d.h. Einfügen oder Löschen), einem Tupel-Identifizierer und einem DI-Protokoll (d.h. entweder einem Zeiger auf eine Datei, die Löschungen enthält, oder einem Zeiger zu einem neu eingefügten Tupel) besteht.

[0017] IBM Technical Disclosure Bulletin, "Updating Loosely-Coupled SQL/DS Databases", Vol. 33, No. 6B, Nov. 1990, S. 395/396, beschreibt eine entfernte SQL/DS Update-Einrichtung, die einen Übertragungsmechanismus bereitstellt, der bei der Implementierung von verteilten Datenbankanwendungen Verwendung findet. Gemäß der Einrichtung werden Änderungen in eine Datenbank an einem Peer-Knoten zu den jeweils anderen Knoten innerhalb einer SQL-Peer-Gruppe weitergeleitet. Die SQL-Peer-Gruppe ist durch eine Tabelle in der SQL-Datenbank definiert, wobei ein Knoten zu mehreren SQL-Peer-Gruppen zugehörig sein kann. Die Mitglieder einer solchen Gruppe tauschen Informationen durch das Versenden von Updates über bestimmte Daten untereinander aus, wobei ein Datenbankserver Änderungen zu allen in der Tabelle gelisteten Mitgliedern weiterleitet. Die Einrichtung kann zwar eine Übergabe der Updates nicht garantieren, stellt jedoch eine günstige und bequeme Möglichkeit dar, Kommunikationspfade für Datenbank-Updates bereit zu stellen. Insbesondere verfügt die Einrichtung jedoch nicht über die Bereitstellung einer Mehrzahl von Konfliktroutinen, die nach einander aufrufbar sind.

[0018] Das US-Patent 5,170,480 von Mohan et al. bezieht sich auf die Erhaltung der Datenkonsistenz zwischen einer Ursprungs- und einer replizierten Datenbank durch Trennen von Redo-Datensätzen aus dem Transaktionsprotokoll der Ursprungsdatenbank in entsprechende Warteschlangen. Jede Warteschlange wird exklusiv mit jeweils einer anderen Warteschlange von parallelen Warteschlangen-Servern verbunden. Durch exklusive Abarbeitung der Redo-Datensätze in den Warteschlangen durch die replizierte Datenbank wird die replizierte Datenbank widerspruchsfrei zur Ursprungsdatenbank aufgrund eines blockierungsfreien Erneuerungsmechanismus, der die Seiten der replizierten Datenbank parallel verarbeitet. Es wird jedoch nicht die Bereitstellung von mehreren Konfliktroutinen zur Verhinderung Blockierungen offenbart.

Aufgabenstellung

ZUSAMMENFASSUNG DER ERFINDUNG

[0019] Es ist daher eine Aufgabe der vorliegenden Erfindung, ein computerbezogenes Verfahren zur Datenreplikation in Peer-to-Peer-Umgebung (Umgebung mit gleichrangigen Arbeitsstationen) bereit zu stellen, bei dem ein konsistenter Datenbestand in der Umgebung trotz unterschiedlicher Modifikationen desselben Datensatzes an verschiedenen Computersystemen sicher gewährleistet ist.

[0020] Zur Lösung der Aufgabe wird gemäß der vorliegenden Erfindung ein Verfahren gemäß Patentanspruch 1 vorgeschlagen.

[0021] Die vorliegende Erfindung schafft die Möglichkeit zum Replizieren von an einer lokalen Stelle durchgeführten Änderungen (z.B. Einfügen, Löschen oder Erneuern bzw. Updaten) an mehreren entfernten Stellen in einer Peer-to-Peer-Umgebung bzw. einer Umgebung gleichrangiger Arbeitsstationen. Informationen bezüglich dieser Änderungen sind in einer Gruppe von Repliziertabellen enthalten. Somit können Änderungen sofort nach der Festschreibung (commitment) der Änderungsübertragung der ursprünglichen Transaktion an andere Stellen asynchron dupliziert oder verzögert werden, bis die entfernte Stelle verfügbar ist.

[0022] Die Repliziertabellen der vorliegenden Erfindung enthalten eine Transaktionstabelle, eine Transakti-

onsknotentabelle, eine Aufruftabelle, eine Aufrufknotentabelle und eine Ausnahmetabelle. Die Transaktionstabelle enthält Informationen über eine Transaktion. Die Transaktionsknotentabelle enthält einen Eintrag für jede entfernte Stelle, bei der eine Transaktion anzuwenden ist. Die Aufruftabelle enthält einen Eintrag für jedes Verfahren (d.h. Transaktions- oder Nichttransaktionsfolgen logischer Schritte), der zu entfernten Stellen zu replizieren ist. Die Aufrufknotentabelle identifiziert Knoten, an denen ein Verfahren ausgeführt wird. Die Ausnahmetabelle enthält Informationen bezüglich einer Ausnahme, die auftritt, wenn ein Replizierfahren an einer Datenstelle verarbeitet wird (z.B. eine Konflikt- oder Informationsnachricht).

[0023] Die vorliegende Erfindung schafft eine wertorientierte Replikation auf Zeilenebene und Spaltenebene. Eine Replikation auf Zeilenebene ändert eine entfernte Stelle basierend auf den alten und neuen Werten, die in einer Reihe bzw. Zeile an einer Ursprungsstelle enthalten. sind. Eine Replikation auf Spaltenebene ändert eine entfernte Stelle basierend auf den Spaltenwerten an einer Ursprungsstelle.

[0024] Weiterhin schafft eine Replikation auf Zeilenebene und Spaltenebene die Möglichkeit, konkurrierende Änderungen an der entfernten Stelle zu identifizieren. Eine Triggerung bzw. Auslösung oder ein Start (d.h. ein Verfahren) wird jedesmal dann durchgeführt, wenn eine Abänderung an einem Datenfeld durchgeführt wird. Eine Triggerung (trigger) ordnet Einträge in den Repliziertabellen in Warteschlangen an. Die Tabelleneinträge halten Informationen zurück, so daß die ursprüngliche(n) Änderung(en), die zu einem Datenfeld gehört (gehören), zu entfernten Kopien derselben Datenfelder weitergegeben werden können. Die neuen Werte können mit den Werten verglichen werden, die in den entfernten Kopien enthalten sind, um alle Änderungen zu identifizieren, die verloren werden könnten, wenn die aktuelle Änderung zugefügt wird.

[0025] Die vorliegende Erfindung schafft weiterhin eine logik-orientierte Replikation auf Verfahrensebene. Das Replizieren auf Verfahrensebene ändert eine entfernte Stelle basierend auf den logischen Operationen, die zum Ändern von Daten an der Ursprungsstelle verwendet werden. Das Replizieren auf Verfahrensebene schafft genausogut die Möglichkeit zum Identifizieren kollidierender Erneuerungen (updates). Somit können dieselben logischen Operationen, die an einer Kopie eines Datenfeldes durchgeführt werden, zu allen anderen Kopien desselben Datenfeldes weitergegeben werden.

[0026] Informationen bezüglich Kollisionen, die durch die vorliegende Erfindung identifiziert werden, können in den Repliziertabellen zurückbehalten werden. Die in den Repliziertabellen enthaltene Information kann sofort oder nachfolgend zum Adressieren irgendwelcher durch die vorliegende Erfindung erfaßte Kollisionen verwendet werden. Das Replizieren auf Verfahrensebene schafft die Möglichkeit zum Adressieren von Kollisionen und anderen Ausnahmen in dem Verfahren, das zu den entfernten Datenstellen repliziert und bei ihnen ausgeführt wird. Die vorliegende Erfindung schafft weiterhin die Möglichkeit zum Wiederaufheben (rollback) irgendwelcher durchgeführter Änderungen, wenn eine Kollision bzw. ein Konflikt identifiziert wird.

Ausführungsbeispiel

KURZE BESCHREIBUNG DER ZEICHNUNGEN

[0027] **Fig.** 1 stellt eines oder mehrere Computersysteme an einer oder mehreren Stellen dar, die jeweils Kopien von Datenfeldern enthalten.

[0028] **Fig.** 2A stellt die Struktur der an einer Stelle gespeicherten Datenfelder dar.

[0029] **Fig.** 2B und 2C stellen eine normale Transaktion und durch die normale Transaktion erneuerte Abtastdaten dar.

[0030] **Fig.** 2D stellt eine Replikation der an mehreren Datenstellen verarbeiteten Transaktionen dar.

[0031] **Fig.** 3 stellt Repliziertabellen dar.

[0032] **Fig.** 4 stellt einen Verfahrensablauf einer Triggerung dar.

[0033] **Fig.** 5A–5C stellt einen Verfahrensablauf für Änderungsoperationen an lokalen und entfernten Kopien von Datenfeldern dar.

[0034] **Fig.** 6 und 7 stellen eine Technik zum Eingeben von Replizierinformation in die Repliziertabellen dar.

[0035] **Fig.** 8A–8B stellen Repliziertabellen mit einer Zeilenebenen-Replizierinformation dar.

[0036] **Fig.** 9A–9B stellen Verfahren dar, die ein Verfahrensebenen-Replizieren verwenden.

[0037] **Fig.** 10 stellt Repliziertabellen dar, die eine Verfahrensebenen-Replizierinformation enthalten.

[0038] **Fig.** 11A–11B stellen die Ausführung verzögerter entfernter Transaktions- und Nicht-Transaktionsverfahrensaufrufe dar.

[0039] **Fig.** 12 stellt einen Ausnahme-Arbeitsablauf dar.

[0040] **Fig.** 13 stellt einen Rekonstruktions-Arbeitsablauf dar.

DETAILLIERTE BESCHREIBUNG DER ERFINDUNG

[0041] Es wird ein Verfahren und eine Vorrichtung für ein Replizieren von Daten beschrieben. In der folgenden Beschreibung sind zahlreiche spezifische Einzelheiten gezeigt, um eine genauere Beschreibung der vorliegenden Erfindung zu liefern. Es ist einem Fachmann auf dem Gebiet jedoch klar, daß die vorliegende Erfindung ohne diese spezifischen Einzelheiten ausgeführt werden kann. Andererseits sind wohlbekannte Merkmale nicht detailliert beschrieben, damit die Erfindung nicht undeutlich wird.

[0042] In einer Netzwerkumgebung, die aus einer oder mehreren Stellen besteht (z.B. Datenbankserver oder Computerplätze), können duplizierte Kopien derselben Daten an mehr als einer Stelle vorhanden sein (z.B. an einem oder mehreren Datenbankservern oder Computersystemen). **Fig.** 1 zeigt ein Beispiel einer Netzwerkumgebung, die kopierte Daten enthält, die an mehreren Stellen vorhanden sind. Eine Datenstelle A **100** kann irgendeine Art von Computersystem sein (z.B. ein vernetzter Datenbankserver, ein Großrechnersystem oder ein Personalcomputer-System). Gleichermaßen kann eine Datenstelle B **110** irgendeine Art von Computersystem sein. Die Datenstelle A **100** enthält eine Datenbank A **120**. Die Datenstelle A **100** und die Datenstelle B **110** sind über eine Kommunikationsverbindung **125** miteinander verbunden.

[0043] Anfangs enthält die Datenbank B **130**, die an der Datenstelle B **110** angeordnet ist, eine duplizierte Kopie der Datenbank A **120**. Somit ist eine Kopie desselben Datenfeldes an beiden Stellen verfügbar. Das bedeutet, daß ein Benutzer, der auf die Datenbank A zugreift, ein Datenfeld (z.B. eine Anzahl von Besen im Lagerbestand) an der Datenstelle A **100** lesen kann, während ein anderer Benutzer auf die Anzahl verfügbarer Besen durch Lesen von Information zugreifen kann, die an der Datenstelle B **110** vorhanden ist. Solange ein Benutzer A und ein Benutzer B auf die verfügbare Menge in einem Datenfeld in einem Nurlese-Modus zugreifen, bleibt der Wert dieses Datenfeldes an beiden Stellen konstant und daher konsistent.

[0044] **Fig.** 2A stellt eine Datenbank A **120** dar. Die Datenbank A **120** enthält zwei Relationen oder Tabellen. Die Lagerbestandstabelle **202A** enthält zwei Felder. Ein Feld für Artikel **204A** und ein Feld für die verfügbare Menge (qoh = QuantityOnHand) **204B**. Die Felder für Artikel und die verfügbare Menge weisen die Information für jeden in der Lagerbestandstabelle **202A** enthaltenen Lagerbestand auf. Eine Bestell- bzw. Auftragstabelle **202B** enthält Bestell- bzw. Auftragsinformation, die zu einer Bestellung bzw. einem Auftrag eines Kunden gehören (z.B. Ursprungsort; Kunde, der die Bestellung aufgibt; bestellter Artikel; Menge eines bestellten Artikels; und ob die Bestellung durch den verfügbaren Lagerbestand erfüllt werden kann). Jede Tabelle enthält Einträge oder Reihen. Beispielsweise enthält die Lagerbestandstabelle **202A** drei Einträge **206A**–**206C**.

[0045] In **Fig.** 2B enthält eine Datenbank B **130** wie die Datenbank A eine Lagerbestands- und eine Bestell- bzw. Auftragstabelle. Weiterhin enthalten die Datenbank A und die Datenbank B identische Einträge und Werte für jeden Eintrag. **Fig.** 2B zeigt weiterhin eine Gruppe von Schritten, die dazu verwendet werden kann, die in entweder der Datenbank A oder der Datenbank B enthaltenen Daten zu ändern.

[0046] Diese Gruppe von Schritten, nämlich eine typische Bestell- bzw. Auftragstransaktion (d.h. Aufgabe einer Bestellung bzw. Vergabe eines Auftrags), besteht aus Schritten zum Prüfen der verfügbaren Menge, um die Menge des bestellten Arikels im Lager festzustellen, zum Erneuern der verfügbaren Menge in der Lagerbestandstabelle (d.h. dann, wenn die verfügbare Menge > die bestellte Menge ist), und zum Einfügen eines Eintrags in die Bestelltabelle zum Anzeigen der Bestellung. Dieses Verfahren ist in den Schritten 1 und 2 der in **Fig.** 2B gezeigten grundsätzlichen Bestelltransaktion gezeigt.

[0047] Der Endschritt in der grundsätzlichen Bestelltransaktion, der Übertragungsschritt, ist bei der Transak-

tionsverarbeitung ein impliziter Schritt. Er liefert die Möglichkeit, irgendwelche Änderungen vorzunehmen, die an den Tabellen für Dauer durchgeführt worden sind. Vor einem Ausführen einer Übertragung (d.h. einem Dauerhaftmachen der Tabellenänderungen), können die an den Lagerbestands- und Auftragstabellen durchgeführten Änderungen zurückgenommen werden.

[0048] **Fig.** 2B stellt den Anfangszustand der Tabellen in zwei Datenbanken dar. Jedoch können die Benutzer A und B die Tabellen unter Verwendung der grundsätzlichen Auftragstransaktion erneuert. Das bedeutet, daß die Auftragstransaktion dann, wenn der Benutzer A einen Auftrag bzw. eine Bestellung von einem Kunden (z.B. ein Kunde **10** bestellt fünfzig Produkte) und die Auftragstransaktion an der Stelle A (d.h. der Datenbank A) aufruft, die Lagerbestands- und die Auftragstabelle in der Datenbank A erneuert. Die Datenbank B wird unverändert bleiben. Somit ist nach einer Auftragsverarbeitungstransaktion durch den Benutzer A die Datenbank B nicht länger identisch zur Datenbank A.

[0049] **Fig.** 2C stellt die resultierenden Datenbanken dar, nachdem beide Benutzer A und B die Auftragstransaktion verwenden, um einen Auftrag zu verarbeiten, der an ihren jeweiligen Stellen empfangen wird. Beispielsweise empfängt der Benutzer A einen Auftrag für fünfzig Produkte vom Kunden **10**. Der Benutzer A ruft die Auftragstransaktion auf. Die Auftragstransaktion erneuert die Lagerbestandstabelle (d.h. das Feld für die verfügbare Menge, das zum Lagerbestandsartikel "Produkt" gehört) und die Auftragstabelle (d.h. fügt einen Eintrag zum Anzeigen des Auftrags hinzu) in der Datenbank A. Gleichermaßen ruft der Benutzer B die Auftragstransaktion zum Verarbeiten des Auftrags auf, der von einem Kunden **11** an der Stelle B empfangen wird (d.h. für vierzig "Produkte"). Die von B aufgerufene Transaktion erneuert die Lagerbestandstabelle und die Auftragstabelle in der Datenbank B.

[0050] Beide Aufträge bestanden aus einer Bestellung von Produkten. Der Kunde **10** bestellte fünfzig Produkte und der Kunde **11** bestellte vierzig Produkte. Daher wurden insgesamt neunzig Produkte verkauft. Jedoch zeigt die Datenbank A die Bestellung des Kunden **11** nicht an, und die Datenbank B zeigt die Bestellung des Kunden **10** nicht an. Somit zeigt jede Datenbank nur die Bestellungen an, die durch einen der Benutzer (d.h. A oder B) verarbeitet werden, und zeigt daher nicht alle Bestellungen, die an allen Orten verarbeitet worden sind.

[0051] Somit gibt es eine Notwendigkeit dafür, eine lokale Änderung zu allen entfernten Kopien desselben Datenfeldes weiterzugeben. Die vorliegende Erfindung schafft diese Möglichkeit zum Replizieren der Datenänderungen, die an einer Stelle gemacht worden sind, zu anderen Stellen. Somit werden die in **Fig.** 2C dargestellten Datenbanken an andere Orte repliziert, so daß Änderungen aufgrund einer Auftragstransaktion an einer Stelle zu anderen Stellen geführt werden und die Datenfelder wieder konsistent sind.

[0052] **Fig.** 2D stellt den Zustand der zwei Datenbanken dar, bevor und nachdem die Replizierfähigkeiten der vorliegenden Erfindung benutzt werden. Vor einem Replizieren zeigen die zwei Datenbanken nur die lokal verarbeiteten Auftragstransaktionen an (d.h. die Datenbank A zeigt die Auftragstransaktion des Benutzers A und die Datenbank B zeigt die Auftragstransaktion des Benutzers B) und nicht die bei einer entfernten Datenbank verarbeiteten Auftragstransaktionen (z.B. die Datenbank B ist für den Benutzer A eine entfernte Datenbank).

[0053] Nach dem Replizieren von der Datenbank B zur Datenbank A und von der Datenbank A zur Datenbank B werden die Datenänderungen durch den Benutzer A an der lokalen Datenbank des Benutzers A (d.h. der Datenbank A) und an der Datenbank B (d.h. der entfernten Stelle) gezeigt. Gleichermaßen werden die Datenänderungen durch den Benutzer B sowohl an der lokalen als auch an den entfernten Stellen gezeigt. Der Wert der verfügbaren Menge an Produkten in beiden Datenbanken zeigt die gesamte Abnahme der verfügbaren Menge, die zu beiden Bestellungen gehört. Weiterhin zeigen die Auftragstabellen in beiden Datenbanken beide empfangene Bestellungen an.

[0054] Die vorliegende Erfindung schafft die Möglichkeit zum Replizieren von Daten auf der Zeilenebene und auf der Verfahrensebene. Das Replizieren nach der Zeilenebene (d.h. das wertorientierte Replizieren) wird durch Verbinden einer Triggerung mit einer Tabelle (z.B. der Lagerbestandstabelle) erreicht. Eine Triggerung ist ein Verfahren, das dann ausgeführt wird, wenn eine Änderung (z.B. eine Erneuerung, ein Einfügen oder ein Löschen) bei einer Reihe bzw. Zeile in einer Tabelle auftritt. Eine Triggerung identifiziert einen verzögerten entfernten Verfahrensaufruf (DRPC), der als seine Argumente die alten Werte, die neuen Werte und die Operation (z.B. Einfügen, Löschen oder Erneuern) hat.

[0055] Ein Replizieren auf Verfahrenebene gibt eher die Operation als die Zeilenwerte weiter (z.B. logikorientiert). Nachdem ein Verfahren an der Ursprungsstelle ausgeführt ist, verzögert das Verfahren einen Aufruf zu

sich selbst an einer anderen Stelle. Der DRPC wendet die logische Erneuerung des ursprünglichen Verfahrens an einer entfernten Stelle an. Das Replizieren auf Verfahrenebene erfordert weniger Netzverkehr als das Replizieren auf Zeilenebene, da ein DRPC für mehrere Tabellen verwendet werden kann.

Änderung, Identifikation und Zurückbehaltung

[0056] Zum Weitergeben der Änderungen, die an Datenfeldern in einer Datenbank durchgeführt werden, zu denselben Datenfeldern in einer anderen Datenbank, ist es nötig, die Änderungen zurückzuhalten, bis sie an den anderen Stellen durchgeführt werden können. Verfahren nach dem Stand der Technik verwenden ein Transaktions-Wiedergewinnungsprotokoll (d.h. ein Protokoll zum nochmaligen Durchführen) zum Zurückhalten und Identifizieren der Datenbankänderungen für eine Ausbreitung bzw. Weitergabe. Jedoch war das Protokoll zur nochmaligen Durchführung ursprünglich für die Anwendung von Transaktionen in einem einzelnen Datenbanksystem gedacht und enthält eine "Undo"-Information (= Information zum Nicht-Ausführen) (d.h. eine Information, die dazu verwendet werden kann, an Daten durchgeführte Änderungen aufzuheben, nachdem ein Ereignis wie beispielsweise ein Systemausfall auftritt).

[0057] Beispielsweise kann ein "Redo"-Protokoll (= Protokoll zur nochmaligen Ausführung) bei einer einzelnen Kopie der Daten verwendet werden, um bezüglich einer Datenbank durchgeführte Änderungen durch eine oder mehrere Transaktionen ungeschehen zu machen, wenn ein System- oder ein Anwendungsfehler auftritt. Wenn auf einen solchen Fehler gestoßen wird, müssen die Transaktionserneuerungen, die vor dem Fehler durchgeführt sind (und irgendwie mit dem Fehler in Zusammenhang stehen), ungeschehen gemacht werden, um die Datenintegrität beizubehalten, die vor den Erneuerungen existiert. Jedoch wurde das Protokoll zum nochmaligen Ausführen nicht zum Fangen von Änderungsinformation für eine Ausbreitung zu einem zweiten Datenbanksystem entwickelt. Somit tritt dann, wenn das Protokoll für seinen beabsichtigten Zweck wie auch als Mittel zum Replizieren von Daten verwendet wird, ein Speicherverwaltungsproblem auf, weil ein Protokoll für ein nochmaliges Ausführen, das zum Zurückhalten von Ausbreitungsinformation verwendet wird, niemals in den Offline-Betrieb umgeschaltet werden kann (d.h. dazu gebracht, daß das Datenbanksystem nicht darauf zugreifen kann).

[0058] Im Gegensatz zu den Systemen nach dem Stand der Technik bietet die vorliegende Erfindung eine Ausbreitungs- bzw. Weitergabe-Identifikationsfähigkeit, die auf dieselbe Weise wie jede andere Tabelle oder Relation verwaltet werden kann, die durch einen Datenbankverwalter verwaltet wird. Die vorliegende Erfindung bietet die Möglichkeit, die Ausbreitungsinformation in Tabellen im Datenbanksystem zu codieren. Die in diesen Tabellen gespeicherten Informationen können wie alle anderen Daten innerhalb des Datenbanksystems wiedergewonnen werden, und auf sie kann jederzeit zugegriffen werden.

[0059] Die Tabellen enthalten die zum Replizieren einer Datenänderung zu anderen Datenstellen nötige Information. Beispielsweise enthalten die Tabellen auf einen DRPC bezogene Information, ihren Replizierzielort, die Transaktion, von der der DRPC ein Teil ist, die Reihenfolge, in der ein DRPC innerhalb einer Verzögerungs-Transaktion ausgeführt wird, die Reihenfolge, bei den Transaktionen relativ zu anderen Transaktionen ausgeführt werden, und die durch jeden DRPC verwendeten Argumente. Die vorliegende Erfindung verwendet folgende Tabellen: Transaktions-, Transaktionsknoten-, Aufruf-, Aufrufknoten- und Ausnahmetabellen. **Fig.** 3 stellt eine Zusammensetzung dieser Tabellen dar. Aufgrund der Vielseitigkeit der vorliegenden Erfindung (z.B. wird eine Replizierinformation in Relationen gespeichert) kann zusätzliche Information zu diesen Relationen hinzugefügt werden.

- Transaktionstabelle -

[0060] Die Transaktionstabelle enthält Informationen über Transaktionen, die an den Daten durchgeführt werden und die auf irgendeine Weise verzögerte entfernte Verfahrensaufrufe (d.h. DRPC) verwenden. Die Transaktionstabelle besteht aus folgenden Feldern: einem Transaktionsidentifizierer, einer Lieferreihenfolgennummer (DON), einer Startzeit, einem Verzögerungs-Benutzeridentifizierer und einer Zielortliste. Der Transaktionsidentifizierer ("Transaktions_Id") ist ein eindeutiger Identifizierer, der jeder Transaktion zugeordnet wird. Der Transaktions_Id identifiziert weiterhin eindeutig die ursprüngliche Datenbank für verzögerte Transaktionen. Der Transaktions_Id ist der primäre Schlüssel für die Transaktionstabelle (d.h. den Wert, der einen Eintrag in dieser Relation eindeutig identifiziert). Die Zielortliste steuert, ob Zielorte für eine Transaktion beschrieben sind, durch die Aufrufknotentabelle oder externe Führungstabellen.

[0061] Die DON ist eine Abstraktion einer Systemänderungsnummer. Eine DON zeigt die Übertragungsablauffolge einer Transaktion in Relation zu allen anderen Transaktionen, die in der Transaktionstabelle aufgelis-

tet sind. Die DON berücksichtigt die Teil-Reihenfolge der Übertragung der Transaktionen in der Verzögerungs-Datenbank. Somit ist dann, wenn eine Transaktion Eins T1 dieselben Daten wie eine Transaktion Zwei T2 betrifft und T2 vor T1 übertragt, die DON der Transaktion Zwei (D2) kleiner als die DON von T1 (D1).

[0062] Das Zeitfeld zeigt die Zeit, zu der die Transaktion begonnen wurde, und das Verzögerungs-Benutzerfeld identifiziert den Benutzer, der den Verfahrensaufruf der verzögerten Transaktion initiierte. Diese Information kann dazu verwendet werden, einen Zugriff zu überwachen und zu steuern (z.B. aus Sicherheitsgründen). Beispielsweise kann vor einem Durchführen irgendwelcher Änderungen an den Daten an einer entfernten Stelle eine Überprüfung durchgeführt werden, um zu bestimmen, ob der Verzögerungs-Benutzer die Zugriffsprivilegien hat, die die Änderung an der entfernten Stelle zulassen würden.

[0063] Die durch die vorliegende Erfindung geschaffene Replizierfähigkeit bietet die Fähigkeit zum Ändern aller oder eines beliebigen Teils der durch eine Transaktion geänderten Daten durch eine Stellenspezifikation. Das bedeutet, daß eine verzögerte Transaktion aus einer Untergruppe der Aufrufe bestehen kann, die durch die ursprüngliche Transaktion für eine beliebige gegebene Zielortstelle verzögert sind. Beispielsweise kann eine Auftragstransaktion an einer Stelle A eine Auftrags- und eine Lagerbestandstabelle erneuern, und die replizierte Transaktion an der Stelle B kann nur die Auftragstabelle erneuern.

[0064] Eine Aufrufknotentabelle kann dazu verwendet werden, Transaktionsaufrufe zu definieren, die an einer gegebenen Stelle angewendet werden. Zusätzlich kann ein beliebiger Führungsmechanismus verwendet werden. Beispielsweise kann eine Abbildung von Aufrufen zu Zielorten gemäß dem Namen der Verfahren definiert werden, die gerade verzögert werden. Das Zielortlistenfeld der Transaktionstabelle zeigt den verwendeten Abbildungsmechanismus (z.B. eine Aufrufknotentabelle oder einen anderen Führungsmechanismus).

- Transaktionsknotentabelle -

[0065] Die Transaktionsknotentabelle (d.h. Zielorttabelle) identifiziert die Knoten oder entfernten Stellen, an denen die in der Transaktionstabelle enthaltenen Transaktionen auszuführen sind. Die Transaktionsknotentabelle enthält einen Eintrag für jeden Knoten (z.B. eine entfernte Stelle), bei dem eine Transaktion auszuführen ist.

[0066] Der Transaktionsidentifizierer ("Transaktions_Id") hat dieselbe Definition wie dasselbe Feld in der Transaktionstabelle. Der Zielortknoten ("Zielort_Knoten") identifiziert die Knoten (d.h. die entfernten Datenbanken), bei denen die Transaktion auszuführen ist, um die Änderungen zu replizieren, die durch die Transaktion an der lokalen Datenstelle durchgeführt sind. Somit kann derselbe Transaktions_Id zum Zugreifen auf einen Eintrag in der Transaktionstabelle und auf irgendeinen entsprechenden Eintrag oder entsprechende Einträge in der Transaktionsknotentabelle verwendet werden. Weiterhin identifiziert ein Eintrag in der Transaktionsknotentabelle einen der entfernten Orte, zu denen die Transaktionsänderung auszubreiten bzw. weiterzugeben ist. Eine Kombination aus Transaktionsidentifizierer und Zielortknoten kann einen Eintrag in der Transaktionsknotentabelle eindeutig identifizieren.

- Aufruftabelle -

[0067] Wie es in dem zuvor gezeigten Auftragsverarbeitungsbeispiel dargestellt ist, können Transaktionen (d.h. eine logische Arbeitseinheit) aus Schritten oder Verfahren (z. B. Auftragsvergabe und Lagerbestandsprüfung) zusammengesetzt sein. Bei einer Software, die diese Schritte codiert, kann jeder von ihnen als einzelnes Verfahren zum Schaffen einer zusätzlichen Struktur für eine Anwendung angesehen werden. Weiterhin können Verfahren definiert werden, ohne daß sie ein Teil einer Transaktion sind. Informationen bezüglich jedes Verfahrenstyps sind in der Aufruftabelle zurückgehalten. Die Aufruftabelle enthält einen eindeutigen Identifizierer, einen Aufrufidentifizierer ("Aufruf_Id"), der einen Aufruf innerhalb einer Transaktion bestellen kann, oder nicht ausführbare DRPC in Relation zu allen anderen.

[0068] Wie die Transaktions- und Transaktionsknotentabellen enthält die Aufruftabelle einen Transaktionsidentifizierer. Für Transaktions-DRPCs hat der Transaktionsidentifizierer dieselbe Definition wie das Transaktionsidentifiziererfeld in den Transaktions- und Transaktionsknotentabellen. Für Nicht-Transaktions- bzw. stationäre DRPCs wird das Transaktionsidentifiziererfeld nicht verwendet.

[0069] Der Verzögerungsverfahrensidentifizierer ("Verfahrens_Id") identifiziert das Verfahren oder den Aufruf (d.h. eine Reihe von Programmschritten), der an einem entfernten Ort bzw. einer entfernten Stelle auszuführen ist. Beispielsweise kann der Verzögerungsverfahrensidentifizierer eine Zeichenkette sein, die den Namen des

Verfahrens enthält. Er sollte auch ein im System vorgesehener eindeutiger Identifizierer sein, der die Stelle (z.B. Speicheradresse) des Verfahrens enthält. Die Parameterzahl identifiziert die Anzahl von Parametern (Werte, die in das Verfahren zur Anwendung während der Ausführung des Verfahrens geführt werden) für ein Verfahren.

[0070] Das Parameterfeld (Parameter) ist eine langreihige Bytekette, die die Parameter für den Eintrag in die Aufruftabelle enthält. Das Format des Feldes ist wie folgt:

<tc_1><$Länge_1$><$Wert_1$><tc_2><$Länge_2$><$Wert_2$><...<tc_n><$Länge_n$><$Wert_n$><0>

[0071] Wobei:
<tc_i> der Parametertypcode für den i-ten Parameter ist (d.h. ob der Parameter vom Typ einer Nummer, eines Zeichens, eines Datums, eines Reihenidentifizierers oder Null ist);
<$Länge_i$> der binäre ganzzahlige Wert von zwei Bytes der Länge des Wertes $_i$ ist (Länge von Null zeigt einen Parameterwert von Null);
<$Wert_i$> der Parameterwert ist;
<0> ein Einzelbyte-Wert ist, der das Ende der Kette anzeigt.

- Aufrufknotentabelle -

[0072] Die Aufrufknotentabelle enthält eine Reihe für jeden Zielort jedes verzögerten Aufrufs, wenn der Zielort nicht durch eine externe Führungstabelle definiert ist. Die Aufrufknoten ermöglichen die Fähigkeit, eine Transaktion zu replizieren, die aus mehreren Verfahrensaufrufen besteht, durch Spezifizieren der Ausführung eines oder aller Aufrufe an einem gegebenen Ort. Wenn Aufrufzielorte nicht durch eine externe Führungsstruktur definiert sind, werden Aufrufknoten durch den Verzögerungs-Benutzer spezifiziert, und zwar entweder mit dem verzögerten Aufruf als Transaktionsfehler-Zielorte oder als vom System bestimmte Zielorte. Wenn eine verzögerte Transaktion zu einem Zielort gesendet wird, wird die Aufrufknotentabelle gefragt, jene Aufrufe auszuwählen, die als Teil der verzögerten Transaktion am Zielort auszuführen sind.

[0073] Der Transaktionsidentifizierer und der Aufrufidentifizierer sind dieselben wie jene in der Aufruftabelle. Das Zielortknotenfeld identifiziert den Knoten, bei dem die Ausführung eines Verfahrens verzögert wird.

- Parametertabelle -

[0074] Bei einem anderen Ausführungsbeispiel kann eine Tabelle dazu verwendet werden, die Parameter für einen Aufruf zurückzuhalten, anstatt die Parameter in der Aufruftabelle zu speichern. Bei diesem Ausführungsbeispiel enthält die Parametertabelle einen Eintrag für jeden Parameter, der durch einen Eintrag in der Aufruftabelle verwendet wird. Das bedeutet, daß es für jeden Aufruf, der Parameter enthält, einen oder mehrere Einträge in der Parametertabelle gibt. Jeder Eintrag in der Parametertabelle enthält einen Parameter für einen Eintrag in der Aufruftabelle.

[0075] Die Parametertabelle enthält einen Aufrufidentifizierer ("Aufruf_Id"). Wie die Aufruftabelle identifiziert der Aufrufidentifizierer einen Verfahrensaufruf. Ein Verfahrensaufruf mit mehr als einem Parameter enthält eine Reihenfolgeliste von Parametern. Die Parameterzahl ("Parameter_Zahl") kann daher eine Stelle in einer Reihenfolgeliste von Parametern des Verfahrensaufrufs identifizieren. Ein Aufrufidentifizierer- und Parameterzahl-Paar kann einen Eintrag in der Parametertabelle eindeutig identifizieren. Das Typenfeld enthält einen Code, der den Parametertyp anzeigt. Das bedeutet, daß das Typenfeld anzeigt, ob der Parametertabelleneintrag eine Zahl, ein Zeichen, ein Datum oder ein Reihenidentifizierer ist.

[0076] Nur eines der übrigen Felder (d.h. Zahl, Zeichen, Daten, Reihenidentifizierer) wird für jeden Eintrag in der Tabelle verwendet. Das bedeutet, daß dann, wenn der Parameter eine Zahl ist, der Wert des Parameters im Zahlenfeld bzw. Nummernfeld gespeichert wird. Gleichermaßen wird dann, wenn der Parameter vom Zeichentyp ist, der Wert des Parameters im Zeichenfeld gespeichert. Ein Datum-Parameterwert wird im Datumfeld gespeichert. Eine Reihenidentifiziererinformation (d.h. Identifizierer, der eine Reihe innerhalb einer Tabelle spezifiziert) wird im Reihenidentifiziererfeld gespeichert.

- Ausnahmetabelle -

[0077] Die Ausnahmetabelle wird dazu verwendet, zu irgendeiner Ausnahme oder irgendeinem Ereignis während Ausführungen einer verzögerten Transaktion gehörende Information zu speichern. Diese Information

kann nachfolgend geprüft werden, und das außergewöhnliche Ereignis kann adressiert werden. Beispielsweise können mehrere kollidierende Erneuerungen bezüglich unterschiedlicher Kopien replizierter Daten auftreten. Somit kann eine Transaktion T1 eine Kopie eines Datensatzes A, nämlich C1, erneuern, und eine zweite Transaktion T2 kann eine zweite Kopie des Datensatzes A, nämlich C2, erneuern. Wenn T1 zu C2 weitergegeben wird, kann T1 die Erneuerung von T2 überschreiben und umgekehrt. Die vorliegende Erfindung erfaßt diesen Ausnahmetyp und auch andere und hält Information für jede Ausnahme zurück.

[0078] Die Ausnahmetabelle enthält ein Transaktionsidentifizierer-("Transaktions_Id")-Feld, das dieselbe Definition wie das Transaktionsidentifiziererfeld in den Transaktions-, Transaktionsknoten und Aufruftabellen hat. Der Aufrufidentifizierer hat dieselbe Definition wie das Aufrufidentifiziererfeld in der Aufruftabelle. Der Zielortknoten ("Zielort_Knoten") identifiziert den Knoten, an dem die Ausnahme auftrat. Im Beispiel des vorherigen Absatzes würde das Knotenfeld die Identifikation des Knotens enthalten, die C2 speichert. Das Fehlercodefeld (Fehler_Code) enthält einen Code zum Identifizieren des Fehlers oder einer Ausnahme, dem begegnet wird (z.B. einer Überschreibmöglichkeit für eine Erneuerung von T2 von C2). Weiterhin enthält das Fehlerkettenfeld eine zusätzliche Beschreibung des Fehlers. Eine Kombination aus Transaktionsidentifizierer und Zielortknoten kann einen Eintrag in dieser Tabelle eindeutig identifizieren.

Bestands-Repliziertabellen

[0079] Das durch die vorliegende Erfindung geschaffene Änderungsreplizieren ist asynchron. Das bedeutet, daß die Replizierverfahren, die die entfernten Datenkopien ändern (z.B. <Tabellenname>_Einfügen) nicht als Teil der Änderungsoperation ausgeführt werden, die am lokalen ursprünglichen Ort durchgeführt wird. Vielmehr kann eine Änderung entfernter Datenkopien verzögert werden, bis die entfernten Kopien verfügbar sind.

[0080] Das Verfahren zum Verzögern der Änderungsoperation am entfernten Ort wird bei der vorliegenden Erfindung durch Speichern der Information für jede verzögerte Änderungsoperation in den Repliziertabellen und nachfolgendes Durchführen der Änderungsoperation, die in den Repliziertabellen identifiziert wird, an den entfernten Orten bzw. Stellen erreicht.

[0081] In bezug auf die Datenbank A (DB_A) in **Fig.** 2C können die Änderungen der Lagerbestands- und der Bestelltabelle beispielsweise zur Datenbank B (DB_B) in **Fig.** 2C durch Replizieren der Änderung der Datenbank A zur Lagerbestands- und zur Auftragstabelle der Datenbank B repliziert werden. Somit können die Erneuerung, die bezüglich der Lagerbestandstabelle der Datenbank A durchgeführt ist, und der Eintrag, der in der Auftragstabelle der Datenbank A eingefügt ist, an den Ort der Datenbank B repliziert werden, und zwar durch Replizieren der Änderungen der Grund-Auftragstransaktion, die bezüglich der Daten in der Datenbank A durchgeführt sind.

[0082] Die Grund-Auftragstransaktion wird bei der Datenbank B durch Bringen der Transaktion in den Repliziertabellen in eine Warteschlange und durch nachfolgendes Anwenden der in der Transaktion enthaltenen Änderungen auf die Daten in der Datenbank B gemäß der in den Repliziertabellen enthaltenen Information repliziert.

[0083] **Fig.** 6 stellt einen Verfahrensablauf zum Bilden einer Warteschlange dar, und zwar aus den Transaktions-, Transaktionsknoten- und Aufrufzielorttabellen für einen Transaktions-, einen Transaktionszielortknoten- und irgendeinen Transaktions- oder Nicht-Transaktions-Aufruf. **Fig.** 6 kann mehrfach aufgerufen werden, um eine beliebige Anzahl von Aufrufen in eine Warteschlange zu bringen. Beispielsweise kann das Aufrufprogramm einen Schleifenbildungsmechanismus enthalten, der einen Warteschlangentransaktions-DRPC für jeden Transaktions- oder Nicht-Transaktions-Aufruf aufrufen kann.

[0084] Beim Entscheidungsblock **601** (d.h. "erster Aufruf?") geht das Verfahren weiter zum Block **616**, wenn dies nicht der erste Aufruf zum Bilden einer Warteschlange dieser Transaktion ist. Wenn es der erste Aufruf ist, geht die Verarbeitung zum Block **602** weiter. Beim Verarbeitungsblock **602** wird der in den Tabellen zu speichernden Transaktion ein aktueller Transaktionsidentifizierer zugeordnet (z.B. die Grund-Auftragstransaktion der Datenbank A). Dem aktuellen Transaktionsidentifizierer wird ein Wert zugeordnet, der den Tabelleneintrag eindeutig identifizieren wird. Beim Verarbeitungsblock **604** wird ein Eintrag in die Transaktionstabelle eingefügt. Dem Transaktionsidentifiziererfeld wird der Wert des aktuellen Transaktionsidentifizierers zugeordnet.

[0085] Der ursprünglichen Transaktion (z.B. der Auftragstransaktion der Datenbank A) wird eine DON zugeordnet, wenn der Übertragungsschritt in der Auftragstransaktion erfolgreich durchgeführt ist. Die DON schafft die Möglichkeit zum Anordnen der Transaktionen basierend auf der Reihenfolge, in der sie die Daten geändert

haben. Somit kann dort, wo die Reihenfolge der Änderungen kritisch ist, die DON dazu verwendet werden, die Änderungsreihenfolge zurückzuhalten und dadurch eine Datenintegrität zu behalten. Dem DON-Feld wird der Wert der DON der ursprünglichen Transaktion zugeordnet. Das Zeitfeld des neuen Transaktionstabelleneintrags wird auf die aktuelle Zeit eingestellt. Dem Verzögerungs-Benutzeridentifizierer wird der Wert des Benutzers zugeordnet, der die ursprüngliche Transaktion begann.

[0086] Ein Zielort (d.h. ein Fern-Datenkopie-Identifizierer) wird beim Verarbeitungsblock **606** identifiziert. Beim Verarbeitungsblock **608** wird ein Eintrag in der Transaktionsknotentabelle erzeugt, um anzuzeigen, daß die gerade in die Repliziertabelle eingegebene Transaktion am identifizierten Zielort durchzuführen ist. Somit wird dem Transaktionsidentifizierer derselbe Wert zugeordnet, wie dasselbe Feld in der Transaktionstabelle, und der Zielortknoten wird auf den identifizierten Zielort eingestellt.

[0087] Für jeden identifizierten Zielort sollte ein Eintrag gemacht werden. Somit geht beim Entscheidungsblock **610** (d.h. "Andere entfernte Zielorte?") dann, wenn zusätzliche Zielorte existieren, die Verarbeitung weiter zum Block **606**, um den nächsten Zielort zu identifizieren. Weiterhin wird ein Transaktionsknotentabelleneintrag für jeden derartigen Zielort erzeugt. Wenn beim Entscheidungsblock **610** (d.h. "Andere entfernte Zielorte?") bestimmt wird, daß alle entfernte Zielorte in die Transaktionsknotentabelle eingegeben worden sind, geht die Verarbeitung zum Entscheidungsblock **616** (d.h. "Alle Aufrufe in der Transaktion verarbeitet?").

[0088] Wie es in der in **Fig.** 2B gezeigten Grund-Auftragstransaktion dargestellt ist, kann eine Transaktion aus vielen Schritten aufgebaut sein. Bei der Grund-Auftragstransaktion waren die Schritte: Prüfen des Lagerbestands und Auftragsverarbeitung. Diese Schritte können softwaremäßig als einzelne Verfahrensschritte entwickelt und codiert sein. In diesem Fall kann die Grund-Auftragstransaktion einen Lagerbestandsprüfungs-Verfahrensschritt enthalten, der die Schritte ausübt, die bei einer Lagerbestandsprüfung enthalten sind. Gleichermaßen kann die Grund-Auftragstransaktion den Auftragsort- und den Übertragungsverfahrensschritt enthalten. Jeder dieser Aufrufe, den die Transaktion aufweist, kann dann in die Aufruftabelle der vorliegenden Erfindung eingegeben werden.

[0089] Beim Verarbeitungsblock **616** wird ein eindeutiger Aufrufidentifizierer erzeugt und ihm wird ein aktueller Aufrufidentifizierer zugeordnet. Beim Verarbeitungsblock **618** werden Warteschlangen-Aufrufargumente dazu aufgerufen, die Aufrufinformation in die Repliziertabellen einzugeben. Nachdem die geeignete Aufrufinformation zu den Repliziertabellen hinzugefügt worden ist, springt die Verarbeitung zurück zum Block **620**.

[0090] **Fig.** 7 stellt einen Verarbeitungsablauf zum Bringen einer Aufrufinformation in eine Warteschlange dar. Beim Verarbeitungsblock **700** wird ein Eintrag in der Aufruftabelle für den Verfahrensschrittaufruf erzeugt, der gerade verarbeitet wird. Das Eintrags-Aufrufidentifiziererfeld wird auf den aktuellen Aufrufidentifizierer eingestellt. Derselbe Wert, der den Transaktionsidentifiziererfeldern in den Transaktions- und Transaktionsknotentabellen zugeordnet ist, wird für das Transaktionsidentifiziererfeld in der Aufruftabelle verwendet. Dem Verfahrensschritt-Identifiziererfeld wird ein Wert zugeordnet, der den gerade verzögerten Verfahrensschritt eindeutig identifizieren kann. Dieser Wert kann irgendeiner sein, der den Verfahrensschritt identifiziert, wie beispielsweise ein Verfahrensschrittname oder die Speicherstelle des Verfahrensschritts.

[0091] Während seiner Ausführung kann ein Verfahrensschritt Werte (Parameter) verwenden, die extern erzeugt werden (d.h. außerhalb des Verfahrensschritts definiert werden) und in den Verfahrensschritt geführt werden, oder die intern erzeugt werden. Beim bevorzugten Ausführungsbeispiel werden die Parameter im Parameterfeld der Aufruftabelle in eine Warteschlange gebracht.

[0092] Beim Verarbeitungsblock **702** wird eine Parameterzahl zu Null initialisiert. Beim Entscheidungsblock **704** (d.h. "Alle Parameter im Aufruf verarbeitet?") springt dann, wenn alle Parameter eines Verfahrensschritts zur Parametertabelle hinzugefügt worden sind oder der Verfahrensschritt keine weiteren zugehörigen Parameter aufweist, die Verarbeitung zurück zum Verarbeitungsblock **714**. Beim Block **714** wird der Wert der Parameterzahl dazu verwendet, das Parameterzahlenfeld des Aufrufeintrags in der Aufruftabelle zu erneuern.

[0093] Beim Entscheidungsblock **716** (d.h. "Aufruf-Zielortknoten nicht durch einen externen Führungsmechanismus definiert und Zielortknoten durch einen Benutzer mit DRPC definiert?") springt dann, wenn die Aufruf-Zielortknoten (d.h. Ausführungsknoten) durch einen externen Führungsmechanismus definiert sind, die Verarbeitung zurück zum Block **724**. Wenn die Aufruf-Ausführungsknoten durch den Benutzer mit dem DRPC definiert sind und nicht durch einen externen Führungsmechanismus, geht die Verarbeitung weiter beim Entscheidungsblock **718**. Beim Entscheidungsblock **718** (d.h. "Alle Ausführungsknoten verarbeitet?") springt dann, wenn alle Ausführungsknoten in die Aufrufknotentabelle eingegeben worden sind, die Verarbeitung zu-

rück zum Block **724**.

[0094] Wenn nicht alle Ausführungsknoten verarbeitet worden sind, geht die Verarbeitung weiter beim Block **720**, um den nächsten Ausführungsknoten zu bekommen, der durch den Benutzer spezifiziert ist. Beim Verarbeitungsblock **722** wird ein Eintrag in der Aufrufknotentabelle für den aktuellen Ausführungsknoten erzeugt. Die Verarbeitung geht weiter beim Entscheidungsblock **718**, um alle übrigen Ausführungsknoten zu verarbeiten.

[0095] Wenn beim Entscheidungsblock **704** (d.h. "Alle Parameter im Aufruf verarbeitet?") zu verarbeitende Parameter zurückbleiben, geht die Verarbeitung weiter beim Verarbeitungsblock **710**. Beim Verarbeitungsblock **710** werden der Parametertyp (d.h. Datentyp), die Länge und der Wert an jeden existierenden Wert im Parameterfeld des aktuellen Aufruftabelleneintrags angehängt. Beim Block **712** wird die Parameterzahl um Eins inkrementiert. Die Verarbeitung geht weiter beim Block **704**, um alle übrigen Parameter zu verarbeiten.

[0096] Bei einem anderen Ausführungsbeispiel können Aufrufparameter in einer separaten Tabelle, nämlich einer Parametertabelle (siehe **Fig.** 3), gespeichert werden. Bei dem anderen Ausführungsbeispiel erzeugt ein Block **710** einen separaten Eintrag in einer Parametertabelle für jeden Parameter. Somit enthält eine Parametertabelle einen Eintrag für jeden Parameter, der zu einem Verfahrensschritt gehört. Jeder Eintrag enthält den Aufrufidentifizierer, eine Parameterzahl (d.h. die Zahl des Parameters in bezug zu anderen Parametern im Aufruf) und den Datentyp des Parameters. Der Wert des Parameters wird in einem der Wertfelder (d.h. Zahl, Zeichen, Daten oder Reihenidentifizierer) basierend auf dem Datentyp gespeichert. Beispielsweise dann, wenn der Lagerbestandsprüfungs-Verfahrensschritt in der Grund-Auftragstransaktion drei Parameter enthielt, würden drei Einträge zur Parametertabelle hinzugefügt werden.

Triggerungen

[0097] Triggerungen bieten eine Alternative zum Initiieren des Bestandes der Repliziertabellen. Eine Triggerung ist ein Verfahrensschritt, der dann ausgeführt wird, wenn eine beliebige Änderung (z.B. eine Erneuerung, eine Einfügung oder ein Löschen) bezüglich eines Eintrags in einer Tabelle am lokalen Ort durchgeführt wird. **Fig.** 4 zeigt ein Beispiel des Verfahrensablaufs einer Triggerung bei der vorliegenden Erfindung.

[0098] Ein Entscheidungsblock **402** (d.h. "Feuert diese Triggerung als Ergebnis einer replizierten Änderung?") stellt eine Ausgabe dar, die bei der vorliegenden Erfindung durch eine Triggerung adressiert ist. Weil eine Triggerung initiiert wird, wenn eine beliebige Änderungsoperation bezüglich einer Tabelle durchgeführt wird, wird eine Operation, die bezüglich eines Fern-Dateneintrags durchgeführt wird (d.h. eine Änderungsoperation), im Initiieren einer zweiten Triggerung resultieren.

[0099] Bis nicht eine Triggerung einen Mechanismus zum Unterscheiden zwischen einer als Ergebnis einer ursprünglichen Änderungsoperation durchgeführten Änderung und einer Änderung, die eine replizierte Änderung ist (d.h. das Ergebnis einer früher gefeuerten Triggerung) enthält, wird die replizierte Änderung selbst eine Triggerung erzeugen und die ursprüngliche Änderungsoperation könnte mehrere Male unnötigerweise an eine Datenstelle repliziert werden. Dies würde die Integrität der Daten an lokalen und entfernten Orten bzw. Stellen gefährden.

[0100] Somit bestimmt der Entscheidungsblock **402** (d.h. "Ist diese Triggerung ein Ergebnis einer früher gefeuerten Triggerung?"), ob die Tabellenänderung, die die Triggerung erzeugte, eine Folge einer ursprünglichen oder einer replizierten Änderung ist. Wenn die Änderung ein Ergebnis einer ursprünglichen Änderung ist, ist die Triggerung nicht das Ergebnis einer früher gefeuerten Triggerung. Jedoch dann, wenn die Änderung das Ergebnis einer ursprünglichen Änderung ist, die zu einer entfernten Tabelle repliziert worden ist, sollte keine zweite Triggerung in den Repliziertabellen in eine Warteschlange gebracht werden.

[0101] Daher endet dann, wenn beim Entscheidungsblock **402** (d.h. "Ist diese Triggerung ein Ergebnis einer früher gefeuerten Triggerung?") eine Triggerung das Ergebnis einer replizierten Änderung ist, eine Verarbeitung beim Block **406**, und der Änderungs-Verfahrensschritt wird nicht in eine Warteschlange gebracht. Wenn die Triggerung beim Entscheidungsblock **402** als Ergebnis einer ursprünglichen Änderungen erzeugt wurde, fährt die Verarbeitung beim Verarbeitungsblock **404** fort. Beim Verarbeitungsblock **404** wird ein Änderungseintrag in die Repliziertabellen eingefügt. Dann endet die Verarbeitung beim Block **406**.

[0102] Der Mechanismus zum Identifizieren duplizierender Replikationen kann bei der vorliegenden Erfindung auf verschiedene Weise implementiert werden. Beim bevorzugten Ausführungsbeispiel kann eine dupli-

zierende Replikation auch durch Einstellen einer globalen Variablen (d.h. einer Variablen, auf die durch einen Triggerungs- oder einen Replizier-Verfahrensschritt zugegriffen werden kann) erfaßt werden, bevor eine replizierte Änderung durchgeführt wird. Wenn die replizierte Änderung durchgeführt wird, kann die Triggerung (beim Entscheidungsblock **402** in **Fig.** 4) die globale Variable prüfen, um zu bestimmen, ob die Änderung das Ergebnis eines Replizier-Verfahrensschritts oder eine ursprüngliche Änderung ist. Diese Alternative eines Einstellens der globalen Variablen im Replizier-Verfahrensschritt ist nachfolgend ausführlicher in Verbindung mit einem Reihenebenen-Replizieren dargestellt.

[0103] Bei einem alternativen Ausführungsbeispiel kann ein duplizierendes Replizieren durch Zuordnen von Datenänderungen zu einem Benutzer erfaßt werden. Somit ist es möglich, eine ursprüngliche Änderung durch ihren Benutzer zu identifizieren und eine replizierte Änderung durch einen anderen Benutzer zu identifizieren (d.h. Änderungs-Verfahrensschritte können durch sich unterscheidende Benutzer ausgeführt werden). Somit kann der Entscheidungsblock **402** in **Fig.** 4 den Namen des Benutzers prüfen, der die Änderung aufrief. Wenn der Benutzer ein sich unterscheidender Benutzer ist, wurde die Triggerung als Ergebnis einer früher gefeuerten Triggerung erzeugt. Wenn der Benutzer kein sich unterscheidender Benutzer ist, wurde die Triggerung als Ergebnis einer ursprünglichen Änderung erzeugt.

[0104] Diese Alternativen bieten Beispiele für Techniken zum Erzeugen duplizierende Änderungen. Irgendwelche Mittel können dazu verwendet werden, duplizierende Änderungen zu erfassen, ohne vom Schutzbereich der Erfindung abzuweichen.

Reihenebenen-Replizier-Verfahrensschritte

[0105] Ein Zeilenebenen- bzw. Reihenebenen-Replizieren ist ein Merkmal der vorliegenden Erfindung, die Triggerungen zum Replizieren von wertorientierten Reihenebenen-Änderungen verwendet. Das bedeutet, daß ein Reihenebenen-Replizieren eine Möglichkeit schafft, um an Werten in einer Reihe durchgeführte Änderungen zu replizieren. Zu einem Reihenebenen-Replizieren gehört eine Triggerung mit einer Tabelle (z.B. der Lagerbestand der Datenbank A in **Fig.** 2A), so daß irgendwelche Änderungen, die an einem oder mehreren Werten in einem lokalen Tabelleneintrag (z.B. Feld für verfügbare Menge in der Lagerbestandstabelle der Datenbank A der **Fig.** 2A) durchgeführt werden, eine gleiche Änderung an entfernten Kopien der geänderten Werte (z.B. Feld für verfügbare Menge in der Lagerbestandstabelle der Datenbank B der **Fig.** 2A) triggern bzw. veranlassen werden.

[0106] Eine Triggerung führt dazu, daß die Information, der ein Verfahrensschritt zugeordnet ist, der zum Replizieren der Wertänderungen in der lokalen Kopie zu einer entfernten Kopie verwendet wird, in den Repliziertabellen gespeichert wird. Der Verfahrensschritt, nämlich ein verzögerter entfernter Verfahrensschrittaufruf (DRPC), kann nachfolgend an entfernten Orten bzw. Stellen zum Replizieren der Datenänderungen) ausgeführt werden, die bezüglich lokaler Daten durchgeführt ist (sind). Der Name des DRPC entspricht der Tabelle, die gerade geändert wird, und der Operation, die gerade bezüglich lokaler Daten durchgeführt wird (z.B. <Tabellenname>_Erneuern). Der DRPC hat als seine Argumente (d.h. Parameter) im allgemeinen die alten Werte der lokalen Daten und die neuen Werte der lokalen Daten. Die alten Werte, oder eine Untergruppe davon, identifizieren eindeutig die Reihe, die das Ziel der Änderung ist. Die Verwendung dieser Argumente ist für die Operation spezifisch, die am lokalen Ort bzw. an der lokalen Stelle durchgeführt wird.

[0107] Beispielsweise dann, wenn eine Erneuerungsoperation bezüglich lokaler Daten durchgeführt wird, würden die alten Werte verwendet, um Kollisionen zu erfassen. Das bedeutet, daß ein Unterschied zwischen den alten Werten der lokalen Daten und den aktuellen Werten am entfernten Ort anzeigen kann, daß eine separate Operation bezüglich entfernter Daten durchgeführt worden ist, die bei der aktuellen Erneuerung gelöscht werden kann. Die neuen Werte können dazu verwendet werden, die entfernten Daten zu erneuern.

[0108] Bei einer Einfügeoperation (d.h. bei einem Einfügen einer Reihe von Werten oder einem Feldwert) gibt es keine alten Werte, und daher sind im Aufruf keine alten Werte enthalten. Weiterhin gibt es keine Notwendigkeit, alte Werte zu verwenden, um eine Prüfung bezüglich Kollisionen (d.h. Ausnahmen) auf der Reihenebene durchzuführen, wenn eine neue Reihe oder ein Feldwert eingefügt wird. Die neuen Werte, oder eine Untergruppe davon, identifizieren eindeutig einen neuen Fern-Tabelleneintrag.

[0109] Wenn die Operation ein Löschen ist (d.h. ein Löschen einer Reihe oder eines Feldes), gibt es keine neuen Werte. Jedoch können, wie bei einer Erneuerungsoperation, die alten Werte dazu verwendet werden, mögliche Kollisionen zu erfassen. Weiterhin können die alten Werte, oder eine Untergruppe davon, dazu verwendet werden, den zu löschenden Fern-Tabelleneintrag eindeutig zu identifizieren.

[0110] Ein DRPC-Name kann den Namen der zu ändernden Tabelle und der durchzuführenden Operation enthalten. In **Fig.** 2A können die Triggerungen für die Lagerbestandstabelle beispielsweise folgende Namen haben: Lagerbestandseinfügung, Lagerbestandserneuerung und Lagerbestandslöschen. Diese Namensgebungskonvention hilft beim Identifizieren der Tabelle und der Operation, die im Replizierverfahren enthalten sind. Jedoch kann bei der vorliegenden Erfindung irgendeine Namensgebungskonvention verwendet werden.

- Reihenebenen-Einfügung -

[0111] Die **Fig.** 5A–5C zeigen ein Beispiel des Verfahrensablaufs für folgende DRPCs: Tabellenname>_Einfügen, <Tabellenname>_Löschen und <Tabellenname>_Erneuern. Diese DRPCs führen an Orten aus, die entfernt von der ursprünglichen Änderung sind. Daher beziehen sie sich dann, wenn sie "Ferntabellen" genannt werden, auf Tabellen, die am Ort der DRPC-Ausführung und entfernt von der ursprünglichen Änderung sind. **Fig.** 5A zeigt ein Beispiel des Verarbeitungsablaufs eines <Tabellenname>_Einfüge-DRPC. Wie es zuvor erörtert ist, kann eine globale Variable als Alternative zum Identifizieren replizierter Änderungen verwendet werden. Eine solche globale Variable wird beim Verarbeitungsblock **502** eingestellt. Beim Verarbeitungsblock **504** wird unter Verwendung des Wertes (der Werte), der (die) in dem (den) Neuwert-Parameter(n) bereitgestellt ist (sind), in den <Tabellennamen> eingefügt.

[0112] Zum weiteren Darstellen der Notwendigkeit zum Testen von duplizierenden Änderungen würde die replizierte Einfügeoperation, die durch den Prozeß der **Fig.** 5A durchgeführt wird, eine Triggerung erzeugen (d.h. irgendeine Tabellenänderung initiiert eine Triggerung). Somit wird das Triggerverfahren der **Fig.** 4 aufgerufen, wenn die Einfügeoperation des Verarbeitungsblocks **504** (**Fig.** 5A) durchgeführt wird. Weil die globale Variable eingestellt wurde, um eine duplizierende Änderung anzuzeigen (beim Verarbeitungsblock **502** der **Fig.** 5A) kann die Triggerung bestimmen (beim Verarbeitungsblock **402** der **Fig.** 4), daß die Änderung eine replizierte Änderung ist, und ein DRPC wird nicht in eine Warteschlange für die replizierte Änderung gebracht (d.h. beim Verarbeitungsblock **404** in **Fig.** 4).

[0113] Beim Fortfahren mit dem Verarbeitungsablauf der **Fig.** 5A wird, nachdem die Einfügeoperation bezüglich der Ferntabelle durchgeführt ist, die globale Repliziervariable beim Verarbeitungsblock **506** zurückgesetzt. Die Verarbeitung endet beim Block **508**.

- Reihenebenen-Erneuerung -

[0114] **Fig.** 5B zeigt ein Beispiel des Verarbeitungsablaufs eines Erneuerungs-DRPC (z.B. eine Erneuerung des Tabellennamen). Eine globale Repliziervariable wird beim Verarbeitungsblock **522** gesetzt. Beim Verarbeitungsblock **524** wird der Ferntabelleneintrag unter Verwendung der alten Werte oder einer Untergruppe der alten Werte identifiziert. Beim Entscheidungsblock **526** (d.h. "Reihe an entferntem Ort gefunden?") fährt dann, wenn der Ferntabelleneintrag nicht gefunden werden kann, die Verarbeitung beim Verarbeitungsblock **532** fort, um eine Ausnahme in den Repliziertabellen zu protokollieren. Weiterhin wird die globale Repliziervariable beim Verarbeitungsblock **548** rückgesetzt und die Verarbeitung endet beim Block **550**. Wenn der Ferntabelleneintrag gefunden wird, fährt die Verarbeitung beim Entscheidungsblock **530** fort.

[0115] Bevor eine Erneuerung zu einem entfernten Ort durchgeführt wird, kann eine Überprüfung durchgeführt werden, um zu bestimmen, ob eine Änderung zu den entfernten Daten durchgeführt worden ist, die unabhängig von der aktuellen Erneuerungsoperation ist, die gelöscht werden könnte, wenn die aktuelle Erneuerung bezüglich der entfernten Daten durchgeführt wird. Dies könnte beispielsweise dann auftreten, wenn eine Änderung (die eine andere als die aktuelle replizierte Operation ist) am entfernten Ort entstanden sein könnte, deren Replizieren den Ort nicht erreicht hat, der die aktuelle Replizererneuerung aufrief. Wenn die aktuelle replizierte Erneuerung die Ferntabelleneintragswerte mit den Neuwert-Parametern überschreibt, werden die aktuellen Werte des Ferntabelleneintrags verloren, und daher wird die ursprüngliche Änderung der Ferntabelle verloren.

[0116] Alternativ dazu können bei einigen Anwendungen fortlaufende Änderungen zum Trennen von Gruppen nichtprimärer Feldwerte zugelassen sein. Beispielsweise muß eine Änderung eines Ausgleichs eines Kunden nicht mit einer Änderung einer Kundenadresse kollidieren. Wenn Erneuerungen auf nichtprimäre Felder auf einer Feld-zu-Feld-Basis angewendet werden können, werden keine fortwährenden Erneuerungen verloren. Beim Entscheidungsblock **530** (d.h. "Verhinderung des Typs einer verlorenen Erneuerung?") wird bestimmt, ob die verlorene Erneuerung eine Reihenebenen-Erneuerung oder eine Spaltenebenen-Erneuerung ist. Wenn sie eine Reihenebenen-Erneuerung ist, fährt die Verarbeitung beim Entscheidungsblock **536** fort. Wenn sie eine Spaltenebenen-Erneuerung ist, fährt die Verarbeitung beim Entscheidungsblock **534** fort.

[0117] Beim Entscheidungsblock **534** (d.h. "Gleicht jeder Feldwert seinem entsprechenden Altwert-Parameter, wo der entsprechende Altwert-Parameter nicht gleich dem entsprechenden Neuwert-Parameter ist?") fährt dann, wenn die alten Werte gleich ihrem entsprechenden Altwert-Parameter sind, wo der Altwert-Parameter ungleich dem Neuwert-Parameter ist, die Verarbeitung beim Verarbeitungsblock **544** fort, um die Felder zu erneuern, die geändert worden sind, und die Verarbeitung fährt beim Block **548** fort. Wenn nicht, fährt die Verarbeitung beim Entscheidungsblock **538** (d.h. "Sollten verlorene Erneuerungen verhindert werden?") fort, wenn verlorene Erneuerungen verhindert werden sollten, und die Verarbeitung fährt beim Block **540** fort, um Ausnahmen auszurufen, und dann fährt die Verarbeitung fort beim Block **548**.

[0118] Beim Entscheidungsblock **536** (d.h. "Gleicht jeder Feldwert in der Reihe seinem entsprechenden Altwert-Parameter?") fährt dann, wenn die alten Werte gleich ihrem entsprechenden Altwert-Parameter sind, die Verarbeitung beim Verarbeitungsblock **546** fort, um jedes Feld in der Reihe mit ihrem entsprechenden Neuwert-Parameter zu erneuern, und die Verarbeitung fährt beim Block **548** fort. Wenn nicht, fährt die Verarbeitung beim Entscheidungsblock **542** (d.h. "Sollten verlorene Erneuerungen verhindert werden?") fort, wenn verlorene Erneuerungen verhindert werden sollten, und die Verarbeitung fährt beim Block **540** fort, um Ausnahmen aufzurufen, und dann fährt die Verarbeitung beim Block **548** fort.

[0119] Beim Verarbeitungsblock wird die globale Repliziervariable rückgesetzt. Die Verarbeitung endet beim Block **550**.

- Reihenebenen-Löschen -

[0120] **Fig.** 5C zeigt ein Beispiel des Verarbeitungsablaufs eines Lösch-DRPC (z.B. ein Tabellennamen-Löschen). Eine globale bzw. Gesamt-Repliziervariable wird beim Verarbeitungsblock **562** gesetzt. Beim Verarbeitungsblock **564** wird der Ferntabelleneintrag unter Verwendung der alten Werte oder einer Untergruppe der alten Werte identifiziert. Beim Entscheidungsblock **566** (d.h. "Reihe an entferntem Ort gefunden?") fährt dann, wenn der Ferntabelleneintrag nicht gefunden werden kann, die Verarbeitung beim Verarbeitungsblock **572** fort, um eine Ausnahme in den Repliziertabellen zu protokollieren. Weiterhin wird die globale Repliziervariable beim Verarbeitungsblock **576** rückgesetzt und die Verarbeitung endet beim Block **578**. Wenn der Ferntabelleneintrag gefunden wird, fährt die Verarbeitung beim Entscheidungsblock **528** fort.

[0121] Wie beim Erneuerungs-DRPC-Verfahren wird eine Überprüfung durchgeführt, um zu bestimmen, ob eine Überprüfung bezüglich verlorener Erneuerungen (d.h. Änderungen) durchgeführt werden sollte. Somit bestimmt der Entscheidungsblock **568** (d.h. "Sollten verlorene Erneuerungen verhindert werden?"), ob auf potentielle verlorene Erneuerungen (d.h. verlorene Erneuerungen) getestet werden soll. Wenn nicht, fährt die Verarbeitung beim Verarbeitungsblock **574** fort, und der Ferntabelleneintrag wird von der Ferntabelle gelöscht. Nachdem die Löschoperation bezüglich des Ferntabelleneintrags durchgeführt ist, wird die globale Repliziervariable beim Verarbeitungsblock **576** rückgesetzt, und die Verarbeitung endet beim Block **578**.

[0122] Wenn beim Entscheidungsblock **568** (d.h. "Sollten verlorene Erneuerungen verhindert werden?") existierende Änderungen aufbewahrt werden sollten, fährt die Verarbeitung beim Entscheidungsblock **570** fort. Beim Entscheidungsblock **530** (d.h. "Ist jeder Feldwert in der Reihe gleich seinem entsprechenden Altwert-Parameter?") fährt dann, wenn irgendeiner der Feldwerte des Ferntabelleneintrags nicht gleich seinem entsprechenden Altwert-Parameter ist, die Verarbeitung beim Verarbeitungsblock **572** fort. Beim Verarbeitungsblock **572** wird eine Ausnahme in den Repliziertabellen protokolliert. Die Verarbeitung fährt beim Block **576** fort, wo die globale Repliziervariable rückgesetzt wird, und die Verarbeitung endet beim Block **578**.

[0123] Wenn beim Entscheidungsblock **570** (d.h. "Ist jeder Feldwert in der Reihe gleich seinem entsprechenden Altwert-Parameter?") alle Feldwerte im Ferntabelleneintrag gleich ihren entsprechenden Altwert-Parametern sind, fährt die Verarbeitung beim Verarbeitungsblock **574** fort. Beim Verarbeitungsblock **574** wird der Ferntabelleneintrag von der Ferntabelle gelöscht. Nachdem die Löschoperation an der Ferntabelle durchgeführt ist, wird die globale Repliziervariable beim Verarbeitungsblock **576** rückgesetzt, und die Verarbeitung endet beim Block **578**.

Reihenebenen-Replizierbeispiel

[0124] Das in **Fig.** 2C dargestellte Replizieren (d.h. DB_B=>DB_A und DB_A=>DB_B) kann durch Verwendung von Triggerungen und eines Reihenebenen-Replizierens erreicht werden. Die Bestellung von fünfzig Produkten am Ort der Datenbank A resultierte im Aufrufen einer Grund-Auftragstransaktion zum Erneuern der Lagerbestandstabelle und zum Anordnen einer Bestellung in der Auftragstabelle. Jede dieser Änderungen wird

veranlassen, daß eine zu jeder Tabelle gehörende Triggerung ausgeführt wird.

[0125] Beispielsweise dann, wenn das Feld für eine verfügbare Menge der Lagerbestandstabelle der Datenbank A durch Subtrahieren der bestellten Menge erneuert wird, ruft eine Triggerung, die zur Lagerbestandstabelle gehört (und in **Fig.** 4 dargestellt ist), die Verfahrensschritte der **Fig.** 6 und 7 auf, um die Repliziertabellen mit einem DRPC zu füllen. In diesem Fall kann ein Lagerbestandserneuerungs-DRPC, der gleich dem Tabellennamen-Erneuerungs-DRPC, der in **Fig.** 5B dargestellt ist, zum Replizieren der Änderungen bezüglich der Lagerbestandstabelle der Datenbank A zur Datenbank B verwendet werden.

[0126] Gemäß **Fig.** 6 wird dann, wenn dies der erste Aufruf ist, ein Transaktionsidentifizierer für den DRPC (d.h. die Tabellennamen-Erneuerung) beim Verarbeitungsblock **602** erzeugt. Beim Verarbeitungsblock **604** wird ein Eintrag in die Transaktionstabelle eingefügt, wie es in **Fig.** 8A dargestellt ist. Ein Transaktionsidentifizierer (z.B. 1), eine DON, eine Zeit und ein Verzögerungs-Benutzeridentifizierer werden der Lagerbestands-Erneuerungstransaktion zugeordnet.

[0127] Bei den Blöcken **606** bis **610** wird die Transaktionsknotentabelle gefüllt. In diesem Fall ist nur eine Fernkopie der Daten bei der Datenbank B angeordnet. Daher wird ein Eintrag in die Transaktionsknotentabelle eingefügt, wo der Transaktionsidentifizierer derselbe wie dasselbe Feld in der Transaktionstabelle ist, und der Zielortknoten wird auf die Datenbank B gesetzt.

[0128] Ein Eintrag wird in der Aufruftabelle für jeden Aufruf in der Transaktion erzeugt. Gemäß **Fig.** 7 wird ein Eintrag in die Aufruftabelle eingefügt, um den Lagerbestandserneuerungs-DRPC zu zeigen. Der Aufrufidentifizierer ist ein eindeutiger Identifizierer für den Lagerbestandserneuerungs-DRPC. Der Transaktionsidentifizierer hat denselben Wert wie dasselbe Feld in den Transaktions- und Transaktionsknotentabellen. Der Identifizierer für ein verzögertes Verfahren kann ein beliebiger Wert sein, der den DRPC identifiziert. In diesem Fall wird der Name des DRPC verwendet.

[0129] Wenn kein externer Führungsmechanismus verwendet wird, kann die Aufrufknotentabelle mit den Einträgen gefüllt werden, die die benutzerspezifizierten Zielortknoten identifizieren, an denen eine Ausführung des DRPC verzögert ist (d.h. den Ausführungsknoten). Gemäß **Fig.** 8A zeigt das Zielortlistenfeld an, daß die Zielortknoten nicht durch einen externen Führungsmechanismus spezifiziert sind. In diesem Fall können die Zielortknoten für den einen DRPC der Transaktion (d.h. die Lagerbestandserneuerung) durch den Eintrag in der Transaktionsknotentabelle bestimmt werden. Jedoch kann die Aufrufknotentabelle dazu verwendet werden, die Datenbank B als den Zielortknoten für eine Ausführung des Lagerbestandserneuerungs-DRPC zu identifizieren.

[0130] Alle zu einem Aufruf gehörenden Parameter sind im Aufrufeintrag in der Aufruftabelle gespeichert. Gemäß **Fig.** 5B verwendet ein Erneuerungs-Verfahrensschritt die alten Lagerbestandswerte und neue Lagerbestandswerte für Produkt. Der Produkt-Eintrag in der Lagerbestandstabelle wird eindeutig durch den alten Wert "Produkt" identifiziert. Weil jeder Eintrag in der Tabelle zwei Felder enthält, wird es zwei alte Werte und zwei neue Werte geben. Daher gehören vier Argumente zum Lagerbestandserneuerungs-Verfahrensschritt. Die Argumente (einschließlich eines Beispiels ihrer Attribute) sind folgende:

Alte Werte			Neue Werte		
Typ	Länge	Wert	Typ	Länge	Wert
2	06	Produkt	2	06	Produkt
1	03	400	1	03	350

[0131] Das Parameterfeld im Aufruftabelleneintrag, dem dieser Aufruf zugeordnet ist, enthält eine Kette von Parameterinformationen. Ein Beendigungswert (z.B. "0") wird an das Ende der Kette gesetzt. Die resultierende Kette ist folgende:
"206Produkt103400206Produkt1033500".

[0132] Wenn eine Lagerbestandserneuerung nicht als Transaktions-DRPC angesehen wird, könnte das Verfahren zum Einfügen von Einträgen in die Transaktions- und Transaktionsknotentabellen umgangen werden. Zum Zuteilen der Zielorte zum Nicht-Transaktions-DRPC, bei denen der DRPC auszuführen ist, kann die Aufrufknotentabelle verwendet werden. Ein Eintrag kann in der Aufrufknotentabelle für jede Stelle angeordnet werden, an der der Aufruf auszuführen ist.

[0133] **Fig.** 8B stellt den Zustand der Repliziertabellen nach ihrer Auffüllung bei einem Nicht-Transaktions-DRPC dar. Die Transaktions- und Transaktionsnotentabellen sind nicht verwendet. Die Aufruf- und Aufrufknotentabellen sind mit Ausnahme des Transaktionsidentifiziererfeldes dieselben wie in **Fig.** 8A. Da es keine zugehörige Transaktion gibt, gibt es keinen Eintrag in den Transaktions- und Transaktionsknotentabellen, und keinen Transaktionsidentifiziererwert.

Spaltenebenen-Replizieren

[0134] Ein Spaltenebenen-Replizieren ist eine Abänderung einer Verhinderung einer verlorenen Erneuerung für ein Reihenebenen-Replizieren. Ein Spaltenebenen-Replizieren wird nur bei Erneuerungsoperationen verwendet. Beim Spaltenebenen-Replizieren können fortlaufende Erneuerungen durchgeführt werden, um Gruppen nichtprimärer Schlüsselspalten zu trennen. Verlorene Erneuerungen werden nur dann verhindert, wenn die Erneuerungen auf Spalten bezogen sind, deren Werte geändert worden sind (d.h. jene Spalten, die am ursprünglichen Ort geändert sind, wie es durch eine Differenz zwischen den Altwert- und Neuwert-Parametern angezeigt wird).

[0135] Ein Spaltenebenen-Replizieren verwendet denselben <Tabellennamen>-Erneuerungs-Verfahrensschritt wie ein Reihenebenen-Replizieren. **Fig.** 5B stellt die Unterschiede bezüglich der Logik zum Erfassen verlorener Erneuerungen und zum Anwenden von Erneuerungen für Reihenebenen- und Spaltenebenen-Replizierschemata dar. Ein Reihenebenen-Replizierschema bestimmt (beim Entscheidungsblock **536**), daß aktuelle Werte an einen entfernten Ort mit alten Werten für alle Spalten in einem Tabelleneintrag übereinstimmen, und zwar vor einem Anwenden einer Erneuerung an einem entfernten Ort. Das Spaltenebenen-Replizieren prüft (beim Entscheidungsblock **534**) nur jene Spalten, die durch die ursprüngliche Erneuerung geändert wurden (wie es durch eine Differenz zwischen den Werten der entsprechenden alten und neuen Parameter angezeigt ist). Wenn die Altwert-Parameter der geänderten Spalten gleich ihrer entsprechenden Werte am entfernten Ort sind, wird der Verlust einer dazwischenliegenden Erneuerung unwahrscheinlich, und die Erneuerungsoperation kann beendet werden. Wenn die Altwert-Parameter der geänderten Spalten nicht dieselben wie ihre entsprechenden Werte sind und verlorene Erneuerungen zu verhindern sind (beim Entscheidungsblock **538**), wird eine Ausnahme hervorgerufen (Verarbeitungsblock **540**), und die Erneuerungsoperation wird nicht durchgeführt. Wenn verlorene Erneuerungen nicht zu verhindern sind (beim Entscheidungsblock **538**), kann die Erneuerungsoperation durchgeführt werden.

[0136] Eine Reihenebenen-Erneuerungsoperation enthält alle Spalten in einem Tabelleneintrag (beim Verarbeitungsblock **546**). Jedoch enthält eine Spaltenebenen-Erneuerungsoperation nur jene Spalten, die durch die ursprüngliche Erneuerung geändert sind (wie es durch eine Differenz zwischen den Werten der entsprechenden alten und neuen Parameter angezeigt ist). Somit werden bei einer Spaltenebenen-Erneuerungsoperation nur jene Spalten, die durch die ursprüngliche Erneuerung geändert sind, mit ihrem entsprechenden Wert in den Neuwert-Parametern erneuert (Verarbeitungsblock **544**).

Verfahrensschrittebenen-Replizieren

[0137] Ein Verfahrensschrittebenen-Replizieren bietet eine andere Alternative zum Replizieren von Datenänderungen. Wie es zuvor erörtert ist, ist ein Reihenebenen-Replizieren wertorientiert. Das bedeutet, daß die Werte, die das Ergebnis einer Operation sind, zu entfernten Kopien repliziert werden. Gegensätzlich dazu bietet ein Verfahrensschrittebenen-Replizieren die Möglichkeit, die logische Operation an entfernte Orte zu replizieren. Das bedeutet, daß der Verfahrensschritt, der die lokale Kopie änderte, an die entfernten Orte repliziert werden kann. Somit erzeugt der Verfahrensschritt nach der Ausführung eines Verfahrensschrittes am Ursprungsort einen DRPC, so daß der Verfahrensschritt selbst zu einem anderen Ort verzögert, um die logische Erneuerung auf eine entfernte Kopie anzuwenden.

[0138] Das Verfahrensschrittebenen-Replizieren bietet zusätzliche Flexibilität darin, daß eine Anwendung (z.B. das Auftragsverarbeitungsbeispiel in den **Fig.** 2A–2C) bestimmen kann, wie ein Replizieren auszubreiten ist, und wie Replizierkonflikten (z.B. mehreren kollidierenden Erneuerungen bezüglich desselben Datenfeldes) zu begegnen ist. Das bedeutet, daß ein DRPC seine Ausbreitung entwickeln kann, und das Verfahren, das aufzurufen ist, wenn Replizierkonflikte identifiziert werden.

[0139] Die **Fig.** 9A–9B zeigen ein Beispiel einer Anwendung, das ein Verfahrensschrittebenen-Replizieren zum Replizieren seiner logischen Operationen zu anderen Orten verwendet. Die Anwendung nimmt die Relationen (d.h. Tabellen) an, die in den **Fig.** 2A–2D beschrieben sind. Weiterhin müssen irgendwelche Änderungen, die an den Tabellen am Ort der Datenbank A durchgeführt sind, an den Ort der Datenbank B repliziert

werden, und umgekehrt.

[0140] Bei der Grund-Auftragstransaktion, die in den **Fig.** 2B–2C dargestellt ist, wird dann, wenn einmal eine Bestellung eines Kunden empfangen ist, die Lagerbestandstabelle erneuert, um ein Absinken des Lagerbestandes um die bestellte Menge zu zeigen, und ein Eintrag wird in der Bestelltabelle eingefügt, um Information über die Bestellung zurückzuhalten. Wenn die Bestellung bei der Datenbank A empfangen und verarbeitet wird, wird dieselbe Auftragsverarbeitung zur Datenbank B repliziert, wie es in **Fig.** 2D dargestellt ist.

[0141] **Fig.** 9A stellt die ursprüngliche Auftragsverarbeitung dar, die durchgeführt wird, wenn eine Bestellung an einem der Orte (z.B. der Datenbank A) empfangen wird. Zum Zusammenfassen der Auftragsverarbeitung bei der Datenbank A wird beim Verarbeitungsblock **902** der bestellte Artikel in der Lagerbestandstabelle gefunden, und die Lagerbestandsmenge wird im Feld für die verfügbare Menge gespeichert. Wenn der Eintrag nicht gefunden wird, wird beim Block **906** eine Ausnahme erzeugt. Wenn die Lagerbestandsmenge (d.h. die verfügbare Menge) größer als die bestellte Menge ist, wird die Lagerbestandsmenge um die bestellte Menge verringert, und die Bestellung wird bei den Blöcken **908**, **912** und **914** als erfüllt angesehen. Wenn die Lagerbestandsmenge nicht größer als die bestellte Menge ist, wird die Bestellung als zurückgestellt angesehen. In jedem Fall wird die Bestellung beim Verarbeitungsblock **916** in die Auftragstabelle eingegeben.

[0142] Wenn die Bestellung einmal bei der Datenbank A verarbeitet ist, wird die Auftragsverarbeitung an entfernten Orten bzw. Stellen durch Eingeben des replizierten Verfahrensschritts in den Repliziertabellen repliziert, und nachfolgend wird der replizierte Verfahrensschritt an den entfernten Orten ausgeführt. Beim Verarbeitungsblock **918** wird die zum replizierten Verfahrensschritt gehörende Information in den Repliziertabellen durch Aufrufen eines Warteschlangentransaktions-DRPC gespeichert. Wie es früher beschrieben ist, speichert der Warteschlangentransaktions-DRPC die replizierte Information in den Tabellen. Der Verarbeitungsblock **918** stellt weiterhin etwas von der Information dar, die in den Repliziertabellen gespeichert ist.

[0143] **Fig.** 10 stellt den Zustand der Repliziertabellen dar, nachdem ein Warteschlangentransaktions-DRPC die Information des replizierten Verfahrensschritts verarbeitet hat. Die Transaktionstabelle wird mit einem Transaktionsidentifizierer versehen, um die Auftragsverarbeitungstransaktion eindeutig zu identifizieren, sowie mit der Transaktions-DON, der Übertragungszeit und dem Verzögerungs-Benutzer. Die Transaktionsknotentabelle hat einen Eintrag für die entfernte Kopie der Lagerbestands- und Auftragstabellen, die in der Datenbank B angeordnet sind.

[0144] Die Aufruftabelle enthält einen Eintrag zum Identifizieren des Verfahrensschritts für den entfernten Auftragsort für diese Auftragsverarbeitungstransaktion (d.h. 4), und die Parameterzahl wird auf Fünf gesetzt. Das Parameterfeld in der Aufruftabelle enthält die Parameterinformation (d.h. bestellter Artikel, Ursprungsort, Kunde, bestellte Menge und den Auftragsstatus in der Datenbank A). Die Aufrufknotentabelle enthält einen Eintrag zum Identifizieren des Knotens, an dem der DRPC für den Ort der entfernten Bestellung auszuführen ist.

[0145] **Fig.** 9B zeigt ein Beispiel der entfernten Auftragsverarbeitung bei der Datenbank B. Außer für die Änderung bei der Datenkopie, die gerade geändert wird, sind die Verarbeitungsblöcke **902** bis **916** dieselben wie die entsprechenden Blöcke in **Fig.** 9A. Wie es zuvor angegeben ist, bietet das Verfahrensschrittebenen-Replizieren die Möglichkeit, eine Anwendung zum Bearbeiten von Ausnahmen zuzulassen, die als Ergebnis des Replizierens des Verfahrensschritts an anderen Orten auftreten. Beim vorliegenden Beispiel wird eine Prüfung durchgeführt, um zu bestimmen, ob es eine Diskrepanz bezüglich der Lagerbestandszahl des bestellten Artikels bei der Datenbank A und bei der Datenbank B gibt. Ein Weg, um dies zu bestimmen, besteht im Feststellen, ob die Bestellung erfüllt werden kann, und zwar basierend auf der Information über die verfügbare Menge in beiden Datenbanken. Somit wird beim Entscheidungsblock **920** in **Fig.** 9B (d.h. "Status = erfüllt?") der Auftragsstatus bei der Datenbank B gegenüber dem Auftragsstatus bei der Datenbank A geprüft. Wenn sie nicht gleich sind, kann eine Ausnahme für eine spätere Überprüfung hervorgerufen werden.

[0146] Anstelle des Hervorrufens einer Ausnahme für eine spätere Überprüfung können andere Verfahren zum Begegnen dieser Diskrepanz im Verfahrensschritt enthalten sein. Beim vorliegenden Beispiel könnte der Verfahrensschritt derart entwickelt worden sein, daß er die Bestellung entweder bei der Datenbank A oder bei der Datenbank B ändert. In jedem Fall bietet die vorliegende Erfindung die Möglichkeit, eine Anwendung zuzulassen, um Ausnahmen unter Verwendung des Verfahrensschrittebenen-Replizierens zu verarbeiten. Unabhängig davon, ob beim Entscheidungsblock **920** eine Diskrepanz erfaßt wird oder nicht, endet das Verfahren bei **924**.

Verzögerte Fernverfahrensschritt-Aufrufinitiierung

[0147] Wie es zuvor gezeigt ist, bietet die vorliegende Erfindung dann, wenn ein DRPC einmal in den Repliziertabellen angeordnet ist, die Möglichkeit, den DRPC an einem entfernten Ort aufeinanderfolgend auszuführen. **Fig.** 11A stellt ein Verfahren zum Initiieren verzögerter Transaktions-DRPC dar, die in den Repliziertabellen enthalten sind.

[0148] Die Auswahl der Transaktions-DRPC kann unter Verwendung einer Anzahl unterschiedlicher Kriterien durchgeführt werden. Beispielsweise können sie basierend auf dem Transaktionsidentifizierer, dem Transaktionszielort oder einer Kombination der beiden ausgewählt werden. Welches Auswahlkriterium auch immer verwendet wird, werden die zu verarbeitenden Transaktionen beim Verarbeitungsblock **1102** ausgewählt. Die Auswahltransaktionen werden für eine Ausführung gemäß dem DON-Feld in den Transaktionstabelleneinträgen in eine Reihe gebracht. Beim Entscheidungsblock **1104** (d.h. "Alle ausgewählten Transaktionen verarbeitet?") endet dann, wenn alle ausgewählten Transaktionen ausgeführt worden sind, die Verarbeitung bei **1132**.

[0149] Wenn es übrige Transaktionen gibt, wird die nächste Transaktion erhalten, und ihr Identifizierer wird im aktuellen Transaktionsidentifizierer beim Verarbeitungsblock **1106** gespeichert. Die bei einer Transaktion durchgeführten Änderungsoperationen können unterlassen werden, bevor sie bestätigt werden. Beim Verarbeitungsblock **1108** ist eine Stelle (d.h. eine Schutzstelle) vorgesehen, um den Status der Daten vor den Änderungen der aktuellen Transaktionen zu identifizieren.

[0150] Beim Entscheidungsblock **1110** (d.h. "Alle Aufrufe verarbeitet?") fährt dann, wenn alle Aufrufe in der aktuellen Transaktion verarbeitet worden sind, die Verarbeitung fort beim Verarbeitungsblock **1112**. Beim Block **1112** werden der Eintrag in der Transaktionsknotentabelle, der der verarbeiteten Transaktion entspricht, und die geänderte entfernte Kopie aus der Transaktionsknotentabelle gelöscht. Beim Entscheidungsblock **1114** (d.h. "Gibt es noch mehr entfernte Kopien, auf die der DRPC anzuwenden ist?") fährt dann, wenn es eine Notwendigkeit zum Zurückhalten des Eintrags der aktuellen Transaktion in den Repliziertabellen gibt, die Verarbeitung fort beim Verarbeitungsblock **1118**. Wenn es keine Notwendigkeit zum Zurückhalten des Eintrags in den Repliziertabellen gibt, wird der Eintrag aus den Repliziertabellen beim Verarbeitungsblock **1116** gelöscht. Beim Verarbeitungsblock **1118** werden die Änderungen der aktuellen Transaktion bestätigt, was ihre Änderungen dauerhaft macht. Die Verarbeitung fährt beim Entscheidungsblock **1104** (d.h. "Sind alle ausgewählten Transaktionen verarbeitet?") fort, um die übrigen Transaktionen durchzuführen.

[0151] Wenn beim Entscheidungsblock **1110** (d.h. "Alle Aufrufe verarbeitet?") nicht alle DRPC bei der aktuellen Transaktion ausgeführt worden sind, fährt die Verarbeitung beim Verarbeitungsblock **1120** fort. Beim Verarbeitungsblock **1120** wird ein Aufrufknotentabelleneintrag für den Zielortknoten verarbeitet. Der Verarbeitungsblock **1121** identifiziert einen Aufruftabelleneintrag für den im Verarbeitungsblock **1120** identifizierten Aufruf.

[0152] Beim Verarbeitungsblock **1122** wird die DRPC-Aufrufkette wiederhergestellt. Eine DRPC-Aufrufkette ist eine Technik zum Identifizieren des auszuführenden DRPC. Andere Einrichtungen zum Identifizieren des DRPC können verwendet werden. Beispielsweise kann ein DRPC durch eine interne Darstellung der Aufrufkette identifiziert werden. In diesem Fall kann der DRPC durch eine optimierte Systemschnittstelle niedriger Ebene unter Verwendung der inneren Darstellung ausgeführt werden. Somit kann der Aufruf ohne Wiederherstellen eines DRPC-Aufrufs in seiner Gesamtheit aufgerufen werden.

[0153] Wenn der DRPC unter Verwendung einer herkömmlichen DRPC-Aufrufkette identifiziert und ausgeführt wird, ist die wiederhergestellte Aufrufkette für den Verfahrensschrittaufruf für den entfernten Auftragsort in der Aufruftabelle in **Fig.** 10 folgendermaßen: entfernter Auftragsort (Produkt, DB_A, 10, 50, erfüllt). Wenn der entfernte Ort in dem Aufruf enthalten ist, ist der Aufruf folgendermaßen:
Entfernte_Auftrags_Vergabe, @DB_B (Produkt, DB_A, 10, 50, erfüllt).

[0154] Gemäß **Fig.** 11A wird beim Verarbeitungsblock **1122** eine DRPC-Aufrufkette unter Verwendung des Verfahrensschrittidentifizierers und der Parameterfelder aus der Aufruftabelle wiederhergestellt. Die Parameter für einen Aufruf werden unter Verwendung des Parameterfeldes in der Aufruftabelle wiederhergestellt. **Fig.** 13 stellt einen Verfahrensablauf zum Aufteilen des Parameterfeldes dar. Beim Verarbeitungsblock **1302** wird ein Byte aus dem Parameterfeld extrahiert. Dieses Byte zeigt den Type des aktuellen Parameters im Parameterfeld.

[0155] Beim Entscheidungsblock **1304** (d.h. "Typ = 0?") endet dann, wenn das aus dem Parameterfeld extra-

hierte Byte gleich einem Endwert ist, die Verarbeitung beim Block **1306**. Wenn das Byte kein Endwert ist, fährt die Verarbeitung beim Verarbeitungsblock **1308** fort, um "Länge" auf die nächsten zwei Bytes des Parameterfeldes zu setzen. Beim Verarbeitungsblock **1310** wird "Wert" auf die nächsten "Länge"-Bytes aus dem Parameterfeld gesetzt. Beim Verarbeitungsblock **1312** werden die Inhalte von "Wert" und "Länge" zu einer Aufrufbildungseinrichtung geführt, um diese Parameterinformation in den wiederhergestellten Aufruf einzubauen.

[0156] Die Verarbeitung fährt beim Block **1302** fort, um irgendwelche übrigen Parameter im Parameterfeld zu verarbeiten.

[0157] Gemäß **Fig.** 11A wird beim Verarbeitungsblock **1124** der Verfahrensschritt am entfernten Ort ausgeführt. Beim Entscheidungsblock **1126** (d.h. "Erfolgreiche Ausführung?") fährt dann, wenn der Verfahrensschritt erfolgreich ausgeführt wurde, die Verarbeitung beim Verarbeitungsblock **1127** fort, um den ausgewählten Aufrufknotentabelleneintrag zu löschen. Die Verarbeitung fährt beim Entscheidungsblock **1110** fort, um auf zusätzliche zu verarbeitende Aufrufe zu prüfen.

[0158] Wenn die Verfahrensschrittausführung nicht erfolgreich war, werden die Änderungen, die durchgeführt sind, da die Schutzstelle zuvor erzeugt wurde, beim Verarbeitungsblock **1128** unterlassen. Beim Verarbeitungsblock **1130** werden die Ausnahmen aufgerufen, um einen Eintrag in der Ausnahmetabelle zu erzeugen, um Information zurückzuhalten, die das nicht erfolgreiche Beenden des Verfahrensschritts betrifft. Die Ausnahmetabelle kann an einer beliebigen Stelle gespeichert werden (z.B. einem Ursprung, einem Zielort oder an beiden Orten). Beim bevorzugten Ausführungsbeispiel ist die Ausnahmetabelle auf der Seite des Zielortes gespeichert. Die Verarbeitung fährt beim Entscheidungsblock **1104** mit irgendwelchen übrigen Transaktionen fort.

- Ausnahme-Verarbeitungsablauf -

[0159] Eine Ausnahme kann in einer Relation gespeichert werden, die entweder am Ort des Ursprungs oder auf der Seite des Zielortes oder an beiden angeordnet ist. **Fig.** 12 stellt einen Verarbeitungsablauf zum Speichern einer Ausnahme dar. Beim Block **1202** werden Einträge in den Fehler- und Transaktionstabellen in den Zielort-Repliziertabellen basierend auf den Werten in den Tabellen des Ursprungsortes erzeugt. Beim Verarbeitungsblock **1204** werden Einträge in der Aufruftabelle und der Aufrufknotentabelle in den Zielort-Repliziertabellen basierend auf den Werten in den Tabellen am Ursprungsort erzeugt. Beim Block **1206** endet die Verarbeitung.

[0160] **Fig.** 11A stellt ein Verfahren zum Ausführen von Transaktions-DRPC-Einträgen dar, die in den Repliziertabellen enthalten sind. Zum Verarbeiten der Repliziertabelleneinträge unter Verwendung der Replizierfähigkeiten der vorliegenden Erfindung können andere Verfahren verwendet werden. Beispielsweise bietet die vorliegende Erfindung die Möglichkeit, Nicht-Transaktions-DRPC-Einträge auszuführen, die in den Repliziertabellen enthalten sind. **Fig.** 11B stellt ein Verfahren zum Ausführen von Nicht-Transaktions-DRPC-Einträgen dar.

[0161] Beim Verarbeitungsblock **1152** werden die auszuführenden Nicht-Transaktions-DRPC ausgewählt. Beim Entscheidungsblock **1154** (d.h. "Alle ausgewählten Aufrufe verarbeitet?") endet dann, wenn alle ausgewählten DRPC verarbeitet worden sind, die Verarbeitung beim Block **1174**. Wenn es übrige Aufrufe gibt, fährt die Verarbeitung beim Verarbeitungsblock **1156** fort, um den nächsten Aufruf auszuwählen und ihn durch seinen Aufrufidentifizierer zu identifizieren. Wie in **Fig.** 11A enthält die Verarbeitung eines Nicht-Transaktions-DRPC in **Fig.** 11B ein Bilden einer Schutzstelle (beim Block **1158**), ein Bilden eines DRPC-Aufrufs (beim Block **1160**) und ein Ausführen des DRPC an einem entfernten Ort (beim Block **1162**). Wenn die Ausführung nicht erfolgreich ist, werden die Änderungen wegen der Schutzstelle unterlassen (beim Block **1172**), und eine Ausnahme wird hervorgerufen (beim Block **1174**). Wenn die Ausführung erfolgreich ist und keine Ausnahmen während der Ausführung hervorgerufen werden, wird der Datensatz für diesen Zielort aus der Aufrufknotentabelle gelöscht (beim Block **1168**), und die Verarbeitung fährt beim Entscheidungsblock **1169** fort. Beim Entscheidungsblock **1169** (d.h. "Zusätzliche Zielorte für einen Aufruf?") fährt dann, wenn es zusätzliche Zielorte für den aktuellen Aufruf gibt, die Verarbeitung beim Entscheidungsblock **1154** fort, um irgendwelche übrigen Aufrufe zu verarbeiten. Wenn es keine zusätzlichen Zielorte für einen Aufruf gibt, werden die Änderungen beim Block **1170** bestätigt.

Konflikte

[0162] Die vorliegende Erfindung bietet die Möglichkeit zum Identifizieren kollidierender Änderungen. Beispielsweise können Erneuerungen, die bei einer entfernten Kopie der Daten aufgetreten sind, verloren werden,

wenn die replizierte Änderung die aktuellen Werte in der entfernten Kopie überschreibt. Somit ist es wichtig, irgendwelche Konflikte zu erfassen. Weiterhin bietet die vorliegende Erfindung dann, wenn ein Konflikt erfaßt wird, die Möglichkeit, eine Ausnahme zu übertragen, irgendwelche Änderungen an einer Datenkopie aufzuheben, nachdem eine Ausnahme erfaßt ist, und eine Ausnahmebearbeitung in einem Anwendungsprogramm einzubauen. Ausnahmen und Konfliktinformation können am Ursprungsort, dem Zielort oder beiden gespeichert werden.

- Konflikterfassung -

[0163] Wie es zuvor angegeben ist, hat ein Reihenebenen-replizierter verzögerter, entfernter Verfahrensschrittaufruf sowohl die alten als auch die neuen Werte als Teil seiner Parameter. Somit kann ein potentieller Konflikt durch Vergleichen der alten Werte der Reihe am ursprünglichen Erneuerungsort mit dem aktuellen Wert der Reihe am Zielort für die replizierte Änderung erfaßt werden. Wenn die Werte unterschiedlich sind, existiert ein Konflikt zwischen den lokalen und den entfernten Datenkopien.

[0164] Wie es zuvor dargestellt ist, bietet die vorliegende Erfindung die Möglichkeit für eine Anwendung, wie beispielsweise das hierin beschriebene Auftragsverarbeitungsbeispiel, mit enthaltener Fehlererfassung. Wie es beim Auftragsverarbeitungsbeispiel dargestellt ist, enthält das Verfahren für eine Bestellung an einem entfernten Ort eine Untersuchung der lokalen und der entfernten Kopien des Auftragsstatus. Somit bietet die vorliegende Erfindung die Möglichkeit für eine Anwendung zum Identifizieren von Konflikten innerhalb eines ihrer Verfahrensschritte.

- Behandlung von Konflikten -

[0165] Wenn ein Konflikt erfaßt wird, kann Information bezüglich des Konflikts identifiziert und in der Ausnahmetabelle gespeichert werden (siehe **Fig.** 3). Die Felder der Ausnahmetabelle bieten einen Fehlercode und eine beschreibende Fehlerkette. Zusätzlich bieten die Ausnahmetabellen Schlüssel in andere Repliziertabellen. Dies schafft die Möglichkeit zum Zugreifen auf die Information, die in den Repliziertabellen gespeichert ist, die zu einem DRPC gehören, in dem die Ausnahme hervorgerufen wird.

[0166] Beispielsweise kann die Ausnahmetabelle als Schlüssel in die Transaktionstabelle einen Transaktionsidentifizierer enthalten, der der gerade verarbeiteten aktuellen Transaktion entspricht. Zum Zugreifen auf den zugehörigen Eintrag in der Aufruftabelle enthält die Ausnahmetabelle weiterhin den Aufrufidentifizierer des gerade verarbeiteten aktuellen Aufruftabelleneintrags und den Zielortknoten (d.h. die entfernte Kopienstelle). Auf die Aufruftabelle kann unter Verwendung eines Verfahrensschritt-Aufrufidentifizierers zugegriffen werden.

[0167] Zusätzlich zum Zurückhalten von Information bezüglich eines Konflikts bietet die vorliegende Erfindung die Möglichkeit, eine verzögerte Transaktion nicht auszuführen, so daß alle Erneuerungen, die durch die ursprüngliche Transaktion verzögert werden, unterlassen werden. Die Ausführung verzögerter Aufrufe ist abhängig von einer erfolgreichen Bestätigung einer Verzögerungs-Transaktion. Wenn die Verzögerungs-Transaktion wiederaufgehoben wird, wird die Warteschlange der verzögerten Aufrufe, die beim Replizieren codiert ist, wiederaufgehoben.

[0168] Die vorliegende Erfindung bietet weiterhin die Möglichkeit zum Beinhalten einer Fehlerverarbeitung bei einer Anwendung. Somit kann ein Fehler verarbeitet werden, sobald ein Fehler erfaßt wird, oder für eine spätere Verarbeitung verzögert werden. Die Ausnahmetabelle bietet die Möglichkeit, nach einer normalen Verarbeitung einer Anwendung irgendwelche Fehler zu adressieren. Die folgende Fehlerverarbeitung kann mit verschiedenen Stufen der Intervention eines Bedieners und einer Automatisierung durchgeführt werden. Die vorliegende Erfindung bietet die Flexibilität, zuzulassen, daß eine Anwendung den Typ der Fehlerbearbeitung adressiert.

[0169] Mehrere Konfliktprogramme können vorgesehen sein, die dazu zu verwenden sind, einen Konflikt zu lösen, wenn er auftritt. Sie können in einer Reihenfolge aufgerufen werden, bis einer von ihnen einen erfolgreichen Rücksprungwert zurückbringt. Wenn keines der Auflösungsprogramme erfolgreich ist, wird die Ausnahme als Ausnahme zurückgehalten.

[0170] Somit ist ein Verfahren und ein Gerät zum Datenreplizieren geschaffen worden.

Patentansprüche

1. Computerbezogenes Verfahren zur Datenreplikation in Peer-to-Peer-Umgebung (Umgebung mit gleichrangigen Arbeitsstationen), wobei das Verfahren folgende Schritte umfaßt:
Durchführen von Datenmodifikationen in einem ersten Computersystem und in einem zweiten Computersystem;
Erzeugen von Ausbreitungs- oder Weitergabeinformationen, die mit den Datenmodifikationen in dem ersten und zweiten Computersystem verknüpft sind, wobei die Ausbreitungs- bzw. Weitergabeinformationen wiederauffindbar und modifizierbar sind und Informationen einschließen, die eine in einem anderen Computersystem auszuführende Operation angeben; und
Durchführen einer bidirektionalen Replikation der Datenmodifikationen unter Verwendung der Ausbreitungs- bzw. Weitergabeinformationen, wobei die bidirektionale Replikation zwischen dem ersten Computersystem und dem zweiten Computersystem stattfindet und durch entweder das erste oder das zweite Computersystem initiiert wird, und wobei die bidirektionale Replikation in konsistenten Daten bezüglich des ersten und zweiten Computersystems resultiert,
Identifizieren von Ausnahmeereignissen während der Durchführung der Datenmodifikationen, wobei der Schritt des Durchführens von Datenmodifikationen von einer Prozedur durchgeführt wird und der Schritt des Identifizierens der Ausnahmeereignisse in der Prozedur enthalten ist, wobei ferner die Prozedur die Ausnahmeereignisse durch die Spezifizierung mehrerer Konfliktroutinen bearbeitet, die der Reihe nach aufgerufen werden können, bis eine von ihnen einen erfolgreichen Wert zurückgibt.

2. Verfahren nach Anspruch 1, bei dem der Schritt des Erzeugens der Ausbreitungs- bzw. Weitergabinformationen ferner folgende Schritte umfaßt:
Erhalten von Informationen bezüglich einer logischen Arbeitseinheit;
Erhalten von Informationen bezüglich der Zielorte, an denen die logische Arbeitseinheit durchzuführen ist;
Erhalten von Informationen bezüglich Verfahrensschritten oder Prozeduren zur Replikation der logischen Arbeitseinheit; und
Erhalten von Informationen bezüglich von Zielorten, an denen die Verfahrensschritte oder Prozeduren durchzuführen sind.

3. Verfahren nach Anspruch 2, welches ferner umfaßt:
Erhalten von Informationen bezüglich von Ausnahmebedingungen, die während der Modifikationen hervorgerufen werden.

4. Verfahren nach Anspruch 2, bei dem der Schritt des Erhaltens von Informationen bezüglich von Zielorten, an denen die Verfahrensschritte oder Prozeduren durchzuführen sind, den Schritt des Erhaltens von Informationen zur Identifizierung eines einer Vielzahl von in einem Netzwerk verbundenen Computern, in dem die Verfahrensschritte oder Prozeduren durchzuführen sind, umfaßt.

5. Verfahren nach Anspruch 2, bei dem der Schritt des Identifizierens von Ausnahmebedingungen die Ermittlung, ob die Datenmodifikationen in dem ersten und dem zweiten Computersystem miteinander in Konflikt stehen, umfaßt.

6. Verfahren nach Anspruch 1, welches ferner den Schritt des Speicherns der Ausbreitungs- bzw. Weitergabeinformationen und der Ausnahmeereignisse in einer oder mehreren Tabellen umfaßt.

7. Verfahren nach Anspruch 2, bei dem der Schritt des Erhaltens von Informationen bezüglich einer logischen Arbeitseinheit einen Schritt des Erhaltens von Informationen bezüglich einer logischen Arbeitseinheit einen Schritt des Erhaltens von Informationen bezüglich von Modifikationen eines Datenwerts umfaßt.

8. Verfahren nach Anspruch 7, bei dem der Schritt des Erhaltens von Informationen bezüglich der Modifikation eines Datenwertes den Schritt des Erhaltens von Informationen bezüglich der Modifikation eines Datenwertes einer Datenbasis umfaßt.

Es folgen 26 Blatt Zeichnungen

Anhängende Zeichnungen

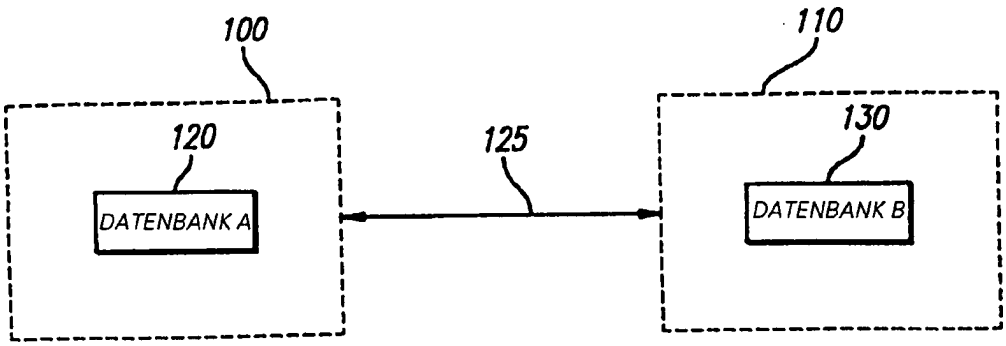


FIG. 1

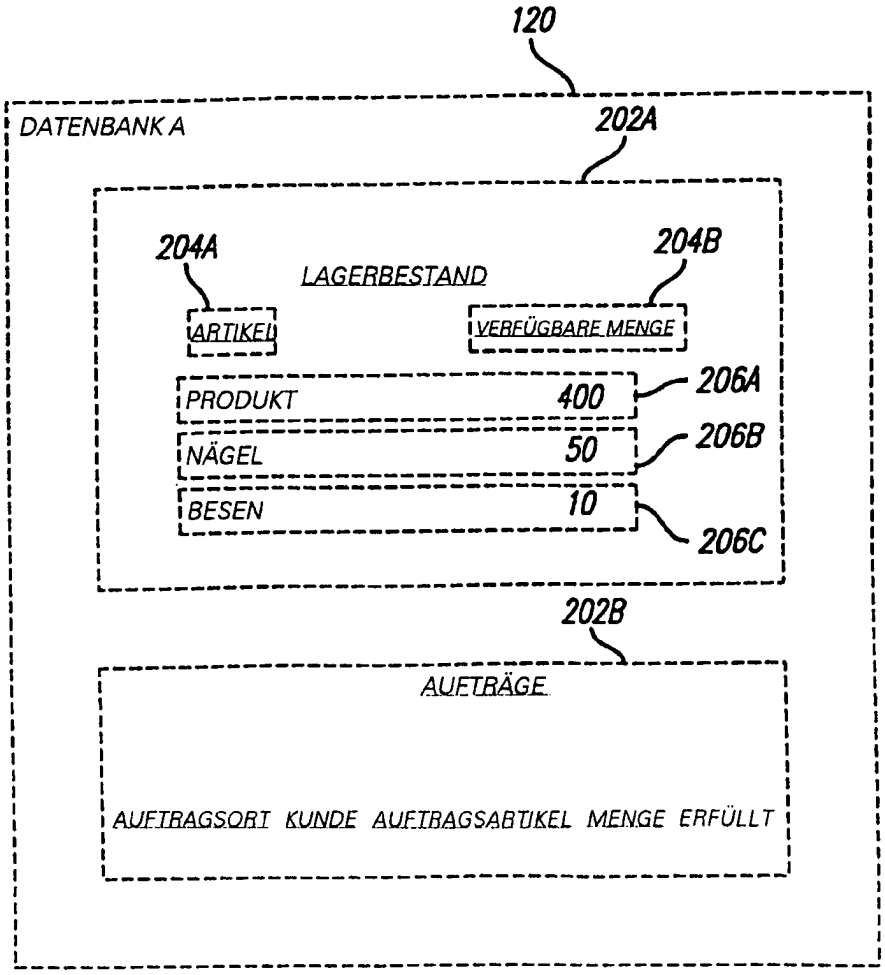


FIG. 2A

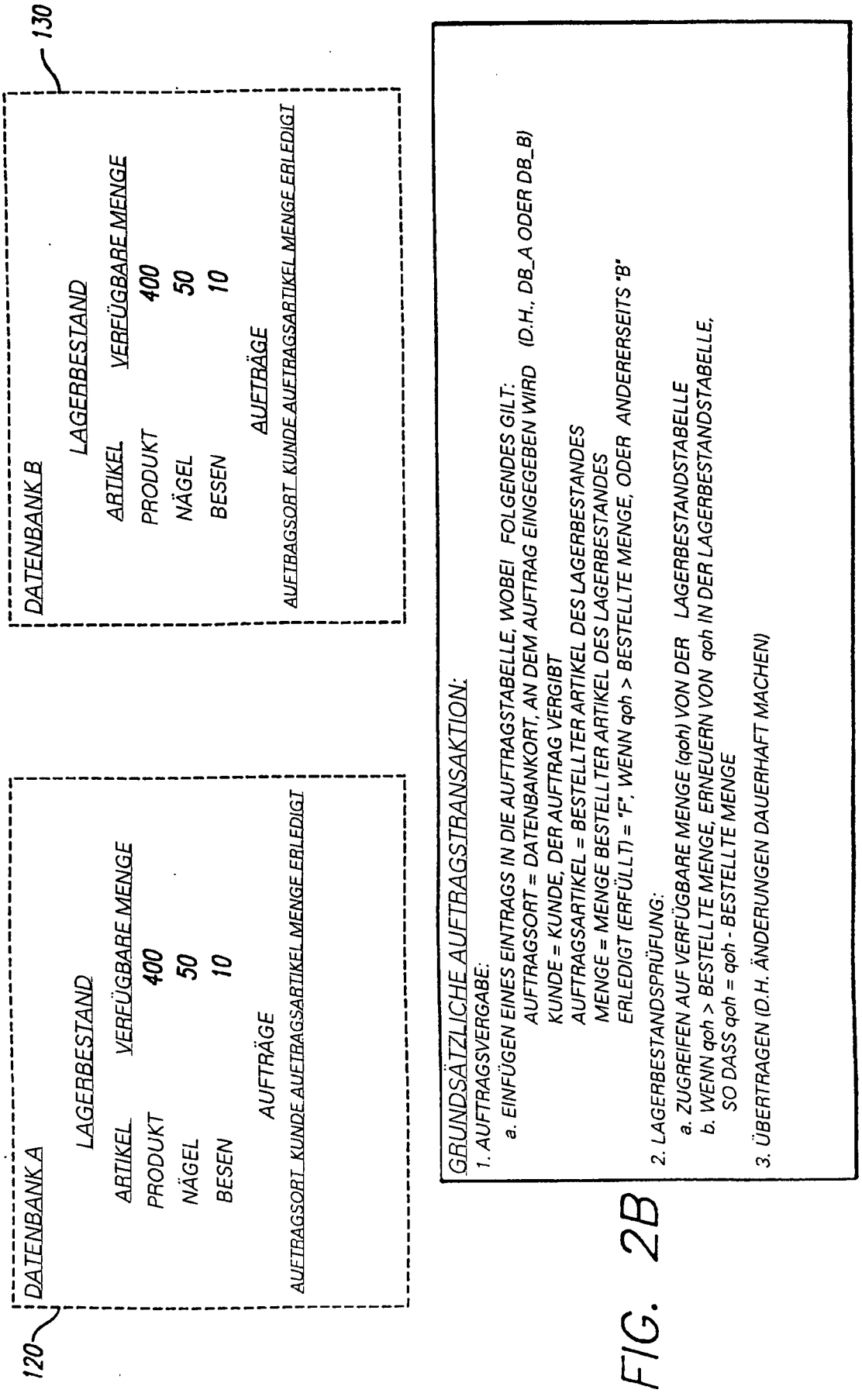

120
DATENBANK A
LAGERBESTAND
ARTIKEL VERFÜGBARE MENGE
PRODUKT 400
NÄGEL 50
BESEN 10
AUFTRÄGE
AUFTRAGSORT KUNDE AUFTRAGSARTIKEL MENGE ERLEDIGT
130
DATENBANK B
LAGERBESTAND
ARTIKEL VERFÜGBARE MENGE
PRODUKT 400
NÄGEL 50
BESEN 10
AUFTRÄGE
AUFTRAGSORT KUNDE AUFTRAGSARTIKEL MENGE ERLEDIGT
GRUNDSÄTZLICHE AUFTRAGSTRANSAKTION:
1. AUFTRAGSVERGABE:
a. EINFÜGEN EINES EINTRAGS IN DIE AUFTRAGSTABELLE, WOBEI FOLGENDES GILT:
AUFTRAGSORT = DATENBANKORT, AN DEM AUFTRAG EINGEGEBEN WIRD (D.H., DB_A ODER DB_B)
KUNDE = KUNDE, DER AUFTRAG VERGIBT
AUFTRAGSARTIKEL = BESTELLTER ARTIKEL DES LAGERBESTANDES
MENGE = MENGE BESTELLTER ARTIKEL DES LAGERBESTANDES
ERLEDIGT (ERFÜLLT) = "F", WENN qoh > BESTELLTE MENGE, ODER ANDERERSEITS "B"
2. LAGERBESTANDSPRÜFUNG:
a. ZUGREIFEN AUF VERFÜGBARE MENGE (qoh) VON DER LAGERBESTANDSTABELLE
b. WENN qoh > BESTELLTE MENGE, ERNEUERN VON qoh IN DER LAGERBESTANDSTABELLE,
SO DASS qoh = qoh - BESTELLTE MENGE
3. ÜBERTRAGEN (D.H. ÄNDERUNGEN DAUERHAFT MACHEN)

FIG. 2B

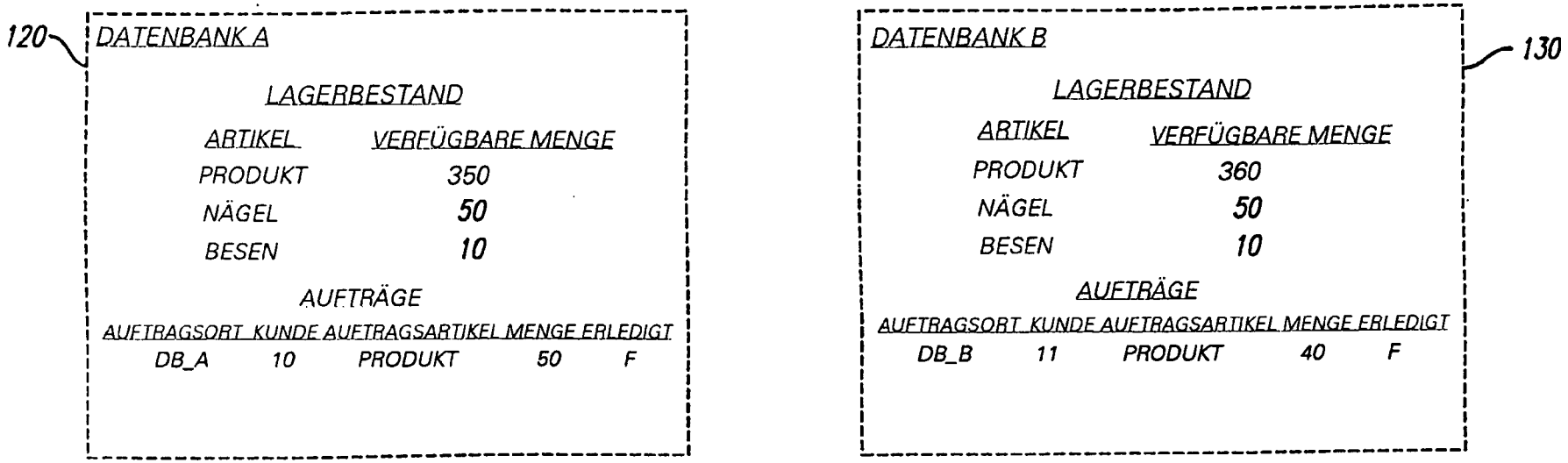


FIG. 2C

GRUNDSÄTZLICHE AUFTRAGSTRANSAKTION:

1. AUFTRAGSVERGABE:
   a. EINFÜGEN EINES EINTRAGS IN DIE AUFTRAGSTABELLE, WOBEI FOLGENDES GILT:
      AUFTRAGSORT = DATENBANKORT, AN DEM AUFTRAG EINGEGEBEN WIRD (D.H., DB_A ODER DB_B)
      KUNDE = KUNDE, DER AUFTRAG VERGIBT
      AUFTRAGSARTIKEL = BESTELLTER ARTIKEL DES LAGERBESTANDES
      MENGE = MENGE BESTELLTER ARTIKEL DES LAGERBESTANDES
      ERLEDIGT (ERFÜLLT) = "F", WENN qoh > BESTELLTE MENGE, ODER ANDERERSEITS "B"
2. LAGERBESTANDSPRÜFUNG:
   a. ZUGREIFEN AUF VERFÜGBARE MENGE (qoh) VON DER LAGERBESTANDSTABELLE
   b. WENN qoh > BESTELLTE MENGE, ERNEUERN VON qoh IN DER LAGERBESTANDSTABELLE, SO DASS qoh = qoh - BESTELLTE MENGE
3. ÜBERTRAGEN (D.H. ÄNDERUNGEN DAUERHAFT MACHEN)

DE 44 97 149 B4 2005.02.10

VOR EINEM REPLIZIEREN

DATENBANK A

LAGERBESTAND

ARTIKEL	VERFÜGBARE MENGE
PRODUKT	350
NÄGEL	50
BESEN	10

AUFTRÄGE

AUFTRAGSORT	KUNDE	AUFTRAGSARTIKEL	MENGE	ERLEDIGT
DB_A	10	PRODUKT	50	F

DATENBANK B

LAGERBESTAND

ARTIKEL	VERFÜGBARE MENGE
PRODUKT	360
NÄGEL	50
BESEN	10

AUFTRÄGE

AUFTRAGSORT	KUNDE	AUFTRAGSARTIKEL	MENGE	ERLEDIGT
DB_B	11	PRODUKT	40	F

NACH EINEM REPLIZIEREN VON DB_B => DB_A

DATENBANK A

LAGERBESTAND

ARTIKEL	VERFÜGBARE MENGE
PRODUKT	310
NÄGEL	50
BESEN	10

AUFTRÄGE

AUFTRAGSORT	KUNDE	AUFTRAGSARTIKEL	MENGE	ERLEDIGT
DB_A	10	PRODUKT	50	F
DB_B	11	PRODUKT	40	F

DATENBANK B

LAGERBESTAND

ARTIKEL	VERFÜGBARE MENGE
PRODUKT	360
NÄGEL	50
BESEN	10

AUFTRÄGE

AUFTRAGSORT	KUNDE	AUFTRAGSARTIKEL	MENGE	ERLEDIGT
DB_B	11	PRODUKT	40	F

FIG. 2D(1)

DE 44 97 149 B4 2005.02.10

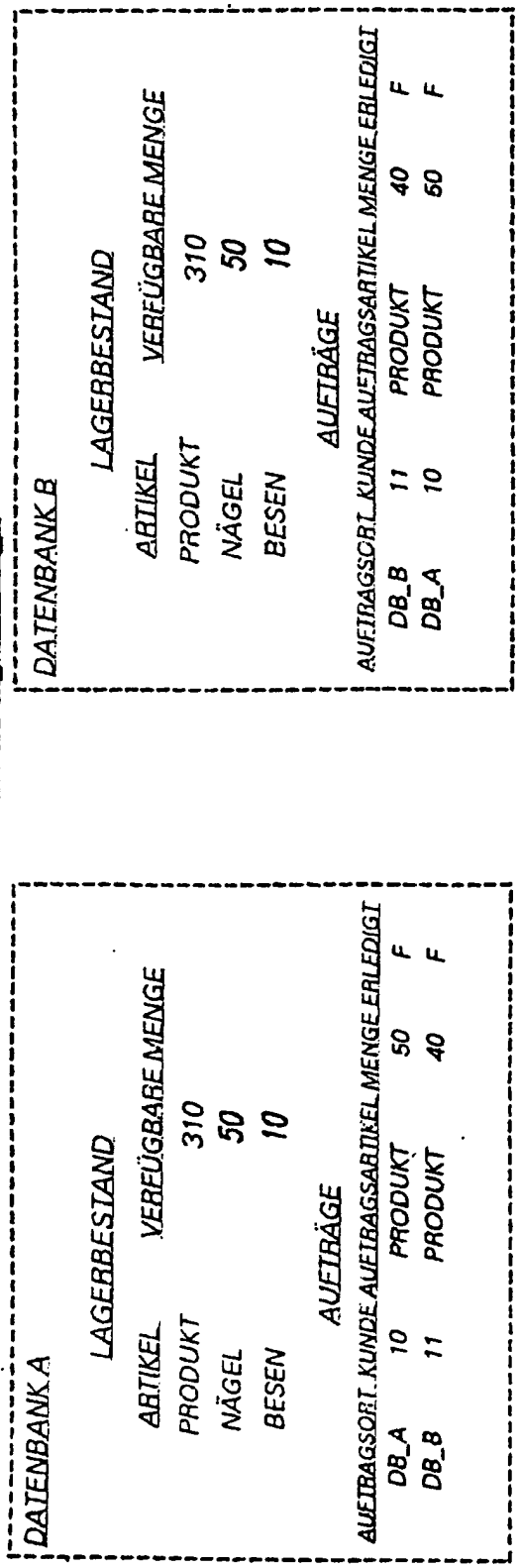

NACH EINEM REPLIZIEREN VON DB_A => DB_B
DATENBANK A
LAGERBESTAND
ARTIKEL VERFÜGBARE MENGE
PRODUKT 310
NÄGEL 50
BESEN 10
AUFTRÄGE
AUFTRAGSORT KUNDE AUFTRAGSARTIKEL MENGE ERLEDIGT
DB_A 10 PRODUKT 50 F
DB_B 11 PRODUKT 40 F
DATENBANK B
LAGERBESTAND
ARTIKEL VERFÜGBARE MENGE
PRODUKT 310
NÄGEL 50
BESEN 10
AUFTRÄGE
AUFTRAGSORT KUNDE AUFTRAGSARTIKEL MENGE ERLEDIGT
DB_B 11 PRODUKT 40 F
DB_A 10 PRODUKT 60 F

FIG. 2D(2)

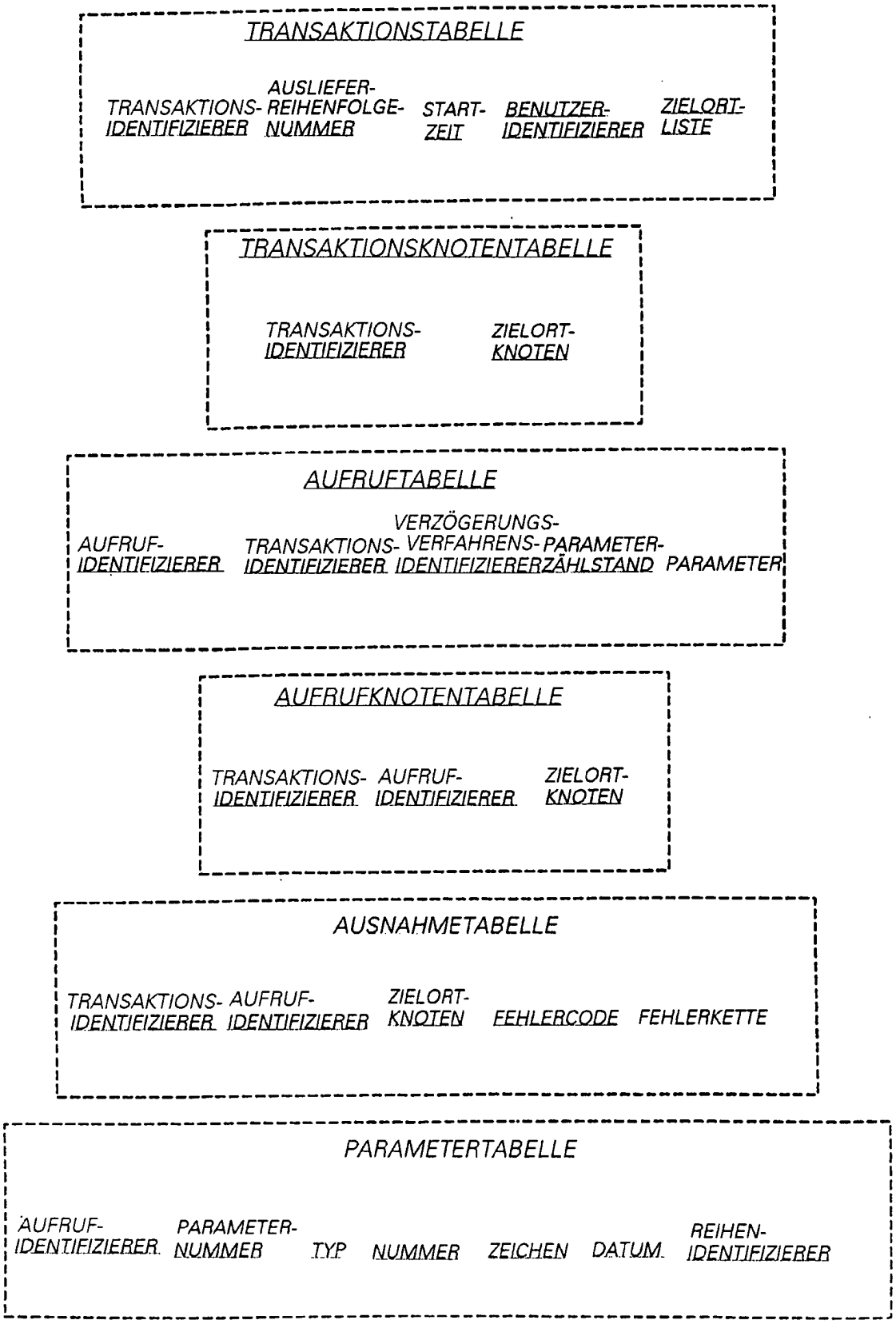

TRANSAKTIONSTABELLE
TRANSAKTIONS-IDENTIFIZIERER
AUSLIEFER-REIHENFOLGE-NUMMER
START-ZEIT
BENUTZER-IDENTIFIZIERER
ZIELORT-LISTE
TRANSAKTIONSKNOTENTABELLE
TRANSAKTIONS-IDENTIFIZIERER
ZIELORT-KNOTEN
AUFRUFTABELLE
AUFRUF-IDENTIFIZIERER
TRANSAKTIONS-IDENTIFIZIERER
VERZÖGERUNGS-VERFAHRENS-IDENTIFIZIERER
PARAMETER-ZÄHLSTAND
PARAMETER
AUFRUFKNOTENTABELLE
TRANSAKTIONS-IDENTIFIZIERER
AUFRUF-IDENTIFIZIERER
ZIELORT-KNOTEN
AUSNAHMETABELLE
TRANSAKTIONS-IDENTIFIZIERER
AUFRUF-IDENTIFIZIERER
ZIELORT-KNOTEN
FEHLERCODE
FEHLERKETTE
PARAMETERTABELLE
AUFRUF-IDENTIFIZIERER
PARAMETER-NUMMER
TYP
NUMMER
ZEICHEN
DATUM
REIHEN-IDENTIFIZIERER

FIG. 3

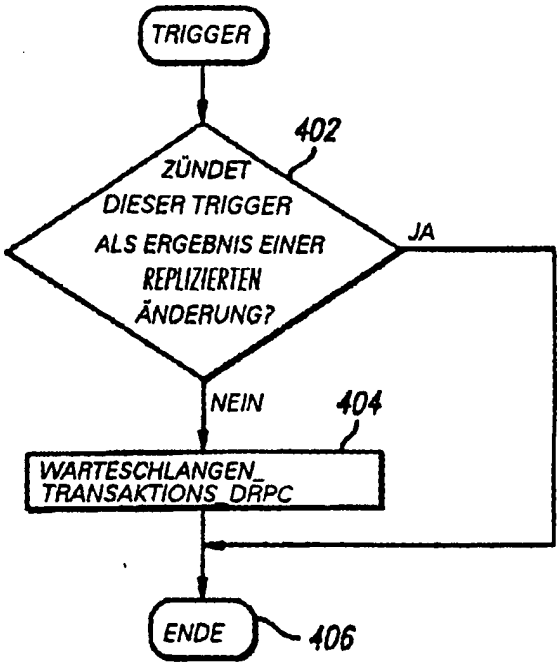

TRIGGER
402
ZÜNDET DIESER TRIGGER ALS ERGEBNIS EINER REPLIZIERTEN ÄNDERUNG?
JA
NEIN
404
WARTESCHLANGEN_TRANSAKTIONS_DRPC
ENDE
406

FIG. 4

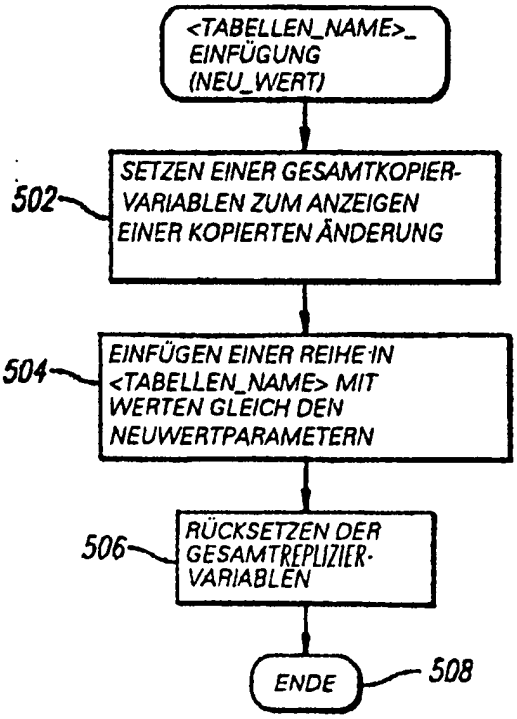

<TABELLEN_NAME>_EINFÜGUNG (NEU_WERT)
502
SETZEN EINER GESAMTKOPIER-VARIABLEN ZUM ANZEIGEN EINER KOPIERTEN ÄNDERUNG
504
EINFÜGEN EINER REIHE IN <TABELLEN_NAME> MIT WERTEN GLEICH DEN NEUWERTPARAMETERN
506
RÜCKSETZEN DER GESAMTREPLIZIER-VARIABLEN
ENDE
508

FIG. 5A

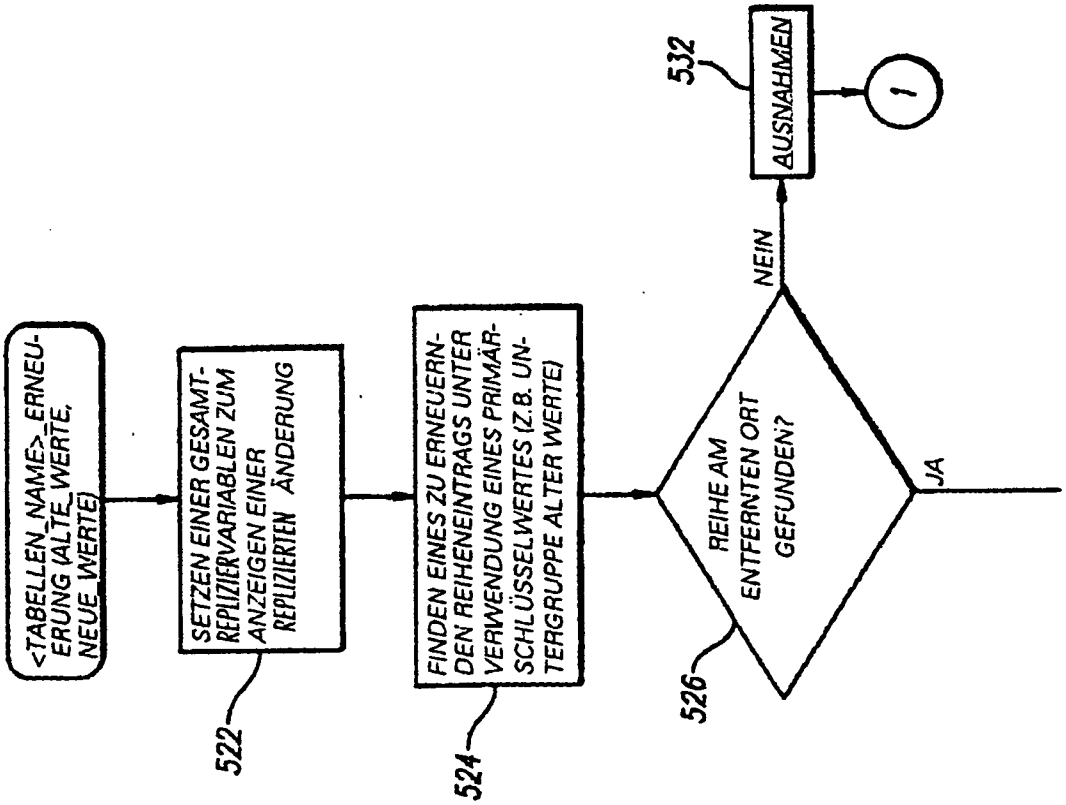

<TABELLEN_NAME>_ERNEUERUNG (ALTE_WERTE, NEUE_WERTE)
SETZEN EINER GESAMT-REPLIZIERVARIABLEN ZUM ANZEIGEN EINER REPLIZIERTEN ÄNDERUNG
522
FINDEN EINES ZU ERNEUERNDEN REIHENEINTRAGS UNTER VERWENDUNG EINES PRIMÄRSCHLÜSSELWERTES (Z.B. UNTERGRUPPE ALTER WERTE)
524
REIHE AM ENTFERNTEN ORT GEFUNDEN?
526
NEIN
JA
AUSNAHMEN
532
1

FIG. 5B(1)

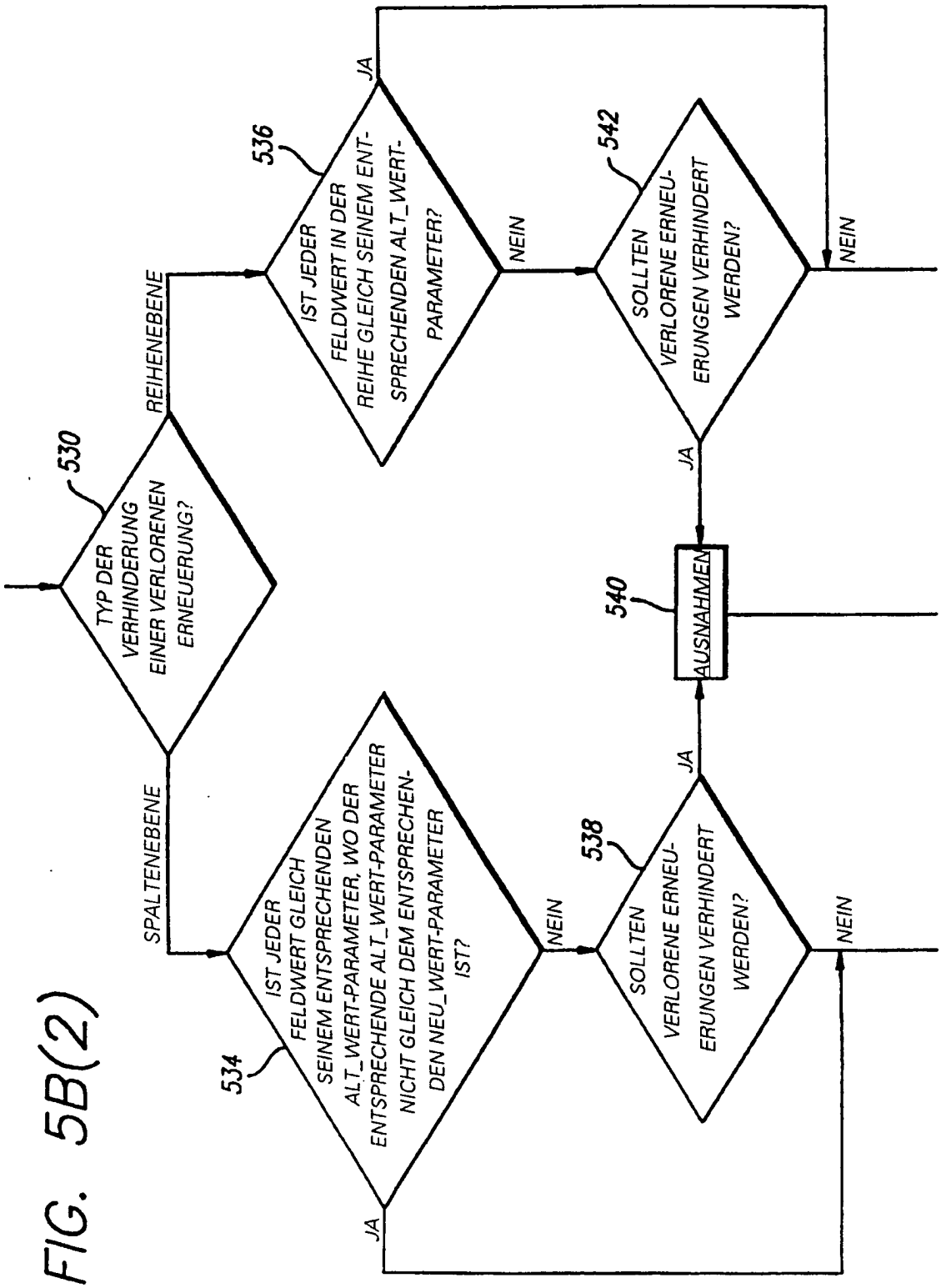

530
TYP DER VERHINDERUNG EINER VERLORENEN ERNEUERUNG?
REIHENEBENE
SPALTENEBENE
536
IST JEDER FELDWERT IN DER REIHE GLEICH SEINEM ENTSPRECHENDEN ALT_WERT-PARAMETER?
JA
NEIN
542
SOLLTEN VERLORENE ERNEUERUNGEN VERHINDERT WERDEN?
JA
NEIN
540
AUSNAHMEN
534
IST JEDER FELDWERT GLEICH SEINEM ENTSPRECHENDEN ALT_WERT-PARAMETER, WO DER ENTSPRECHENDE ALT_WERT-PARAMETER NICHT GLEICH DEM ENTSPRECHENDEN NEU_WERT-PARAMETER IST?
JA
NEIN
538
SOLLTEN VERLORENE ERNEUERUNGEN VERHINDERT WERDEN?
JA
NEIN

FIG. 5B(2)

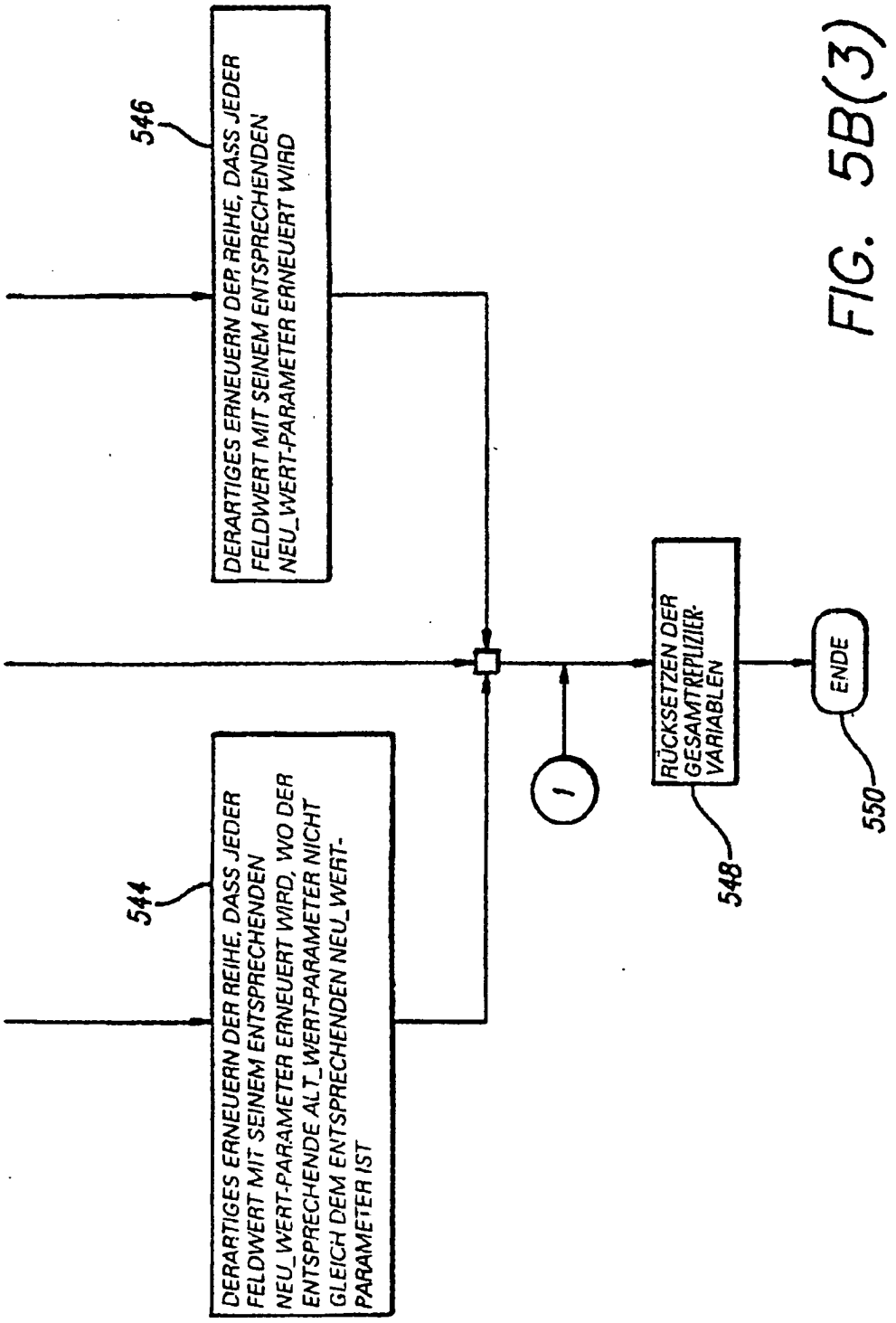

546
DERARTIGES ERNEUERN DER REIHE, DASS JEDER FELDWERT MIT SEINEM ENTSPRECHENDEN NEU_WERT-PARAMETER ERNEUERT WIRD
544
DERARTIGES ERNEUERN DER REIHE, DASS JEDER FELDWERT MIT SEINEM ENTSPRECHENDEN NEU_WERT-PARAMETER ERNEUERT WIRD, WO DER ENTSPRECHENDE ALT_WERT-PARAMETER NICHT GLEICH DEM ENTSPRECHENDEN NEU_WERT-PARAMETER IST
1
RÜCKSETZEN DER GESAMTREPLIZIER-VARIABLEN
548
ENDE
550

FIG. 5B(3)

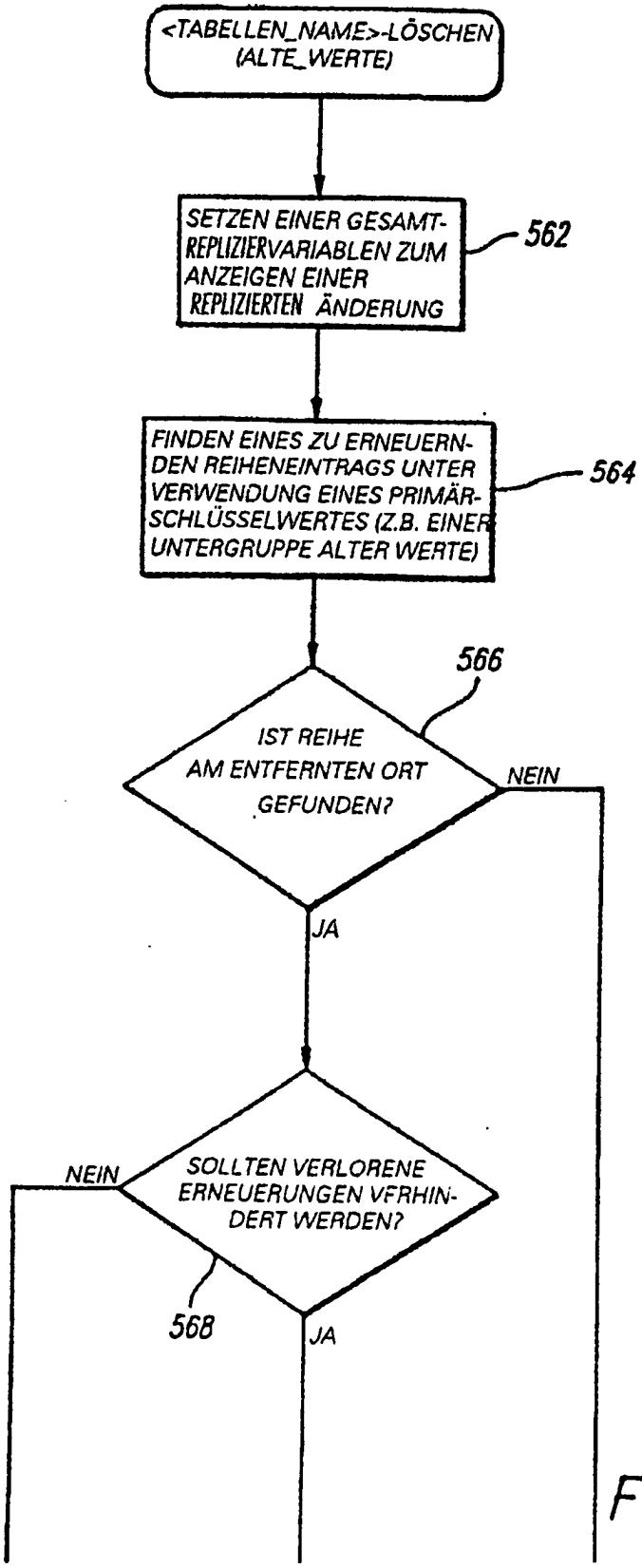

<TABELLEN_NAME>-LÖSCHEN (ALTE_WERTE)
SETZEN EINER GESAMT-REPLIZIERVARIABLEN ZUM ANZEIGEN EINER REPLIZIERTEN ÄNDERUNG
562
FINDEN EINES ZU ERNEUERNDEN REIHENEINTRAGS UNTER VERWENDUNG EINES PRIMÄRSCHLÜSSELWERTES (Z.B. EINER UNTERGRUPPE ALTER WERTE)
564
566
IST REIHE AM ENTFERNTEN ORT GEFUNDEN?
NEIN
JA
SOLLTEN VERLORENE ERNEUERUNGEN VERHINDERT WERDEN?
NEIN
568
JA

FIG. 5C(1)

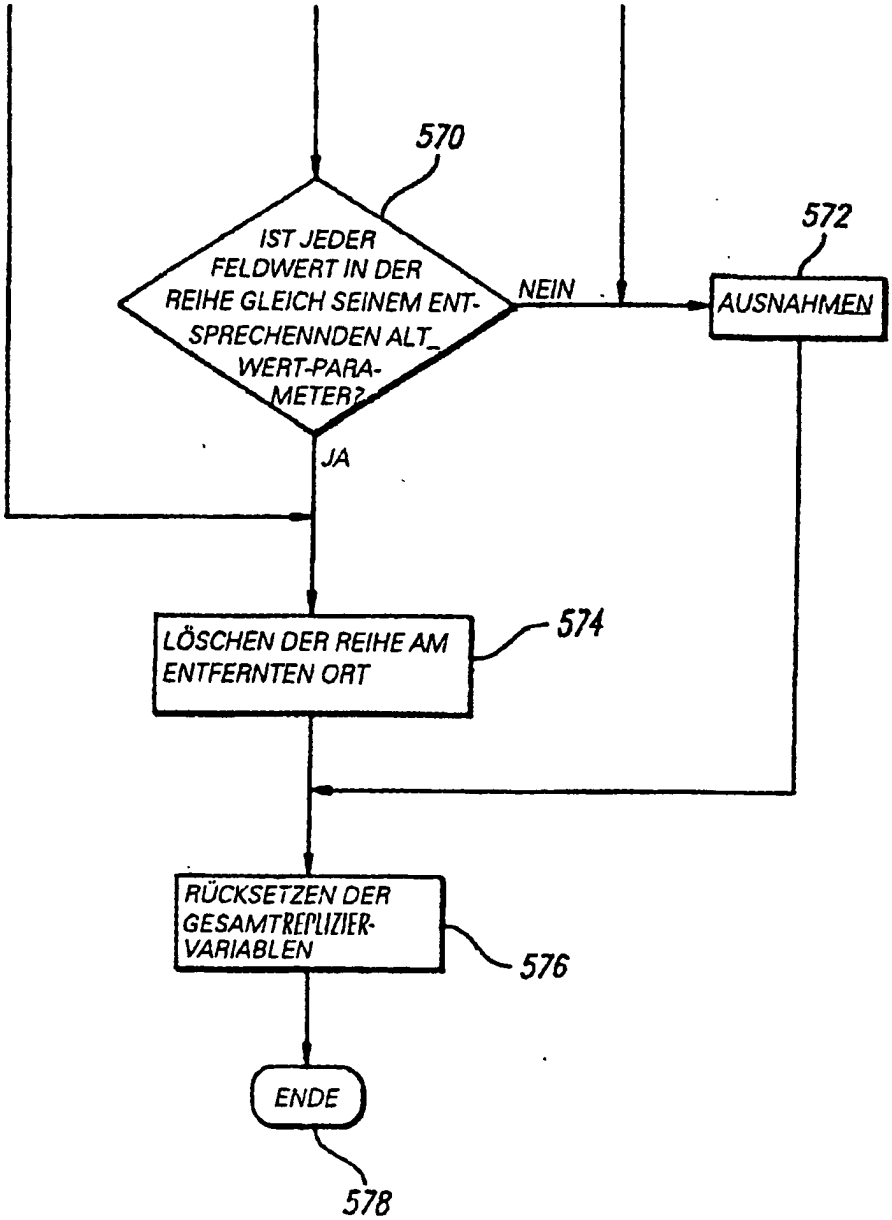
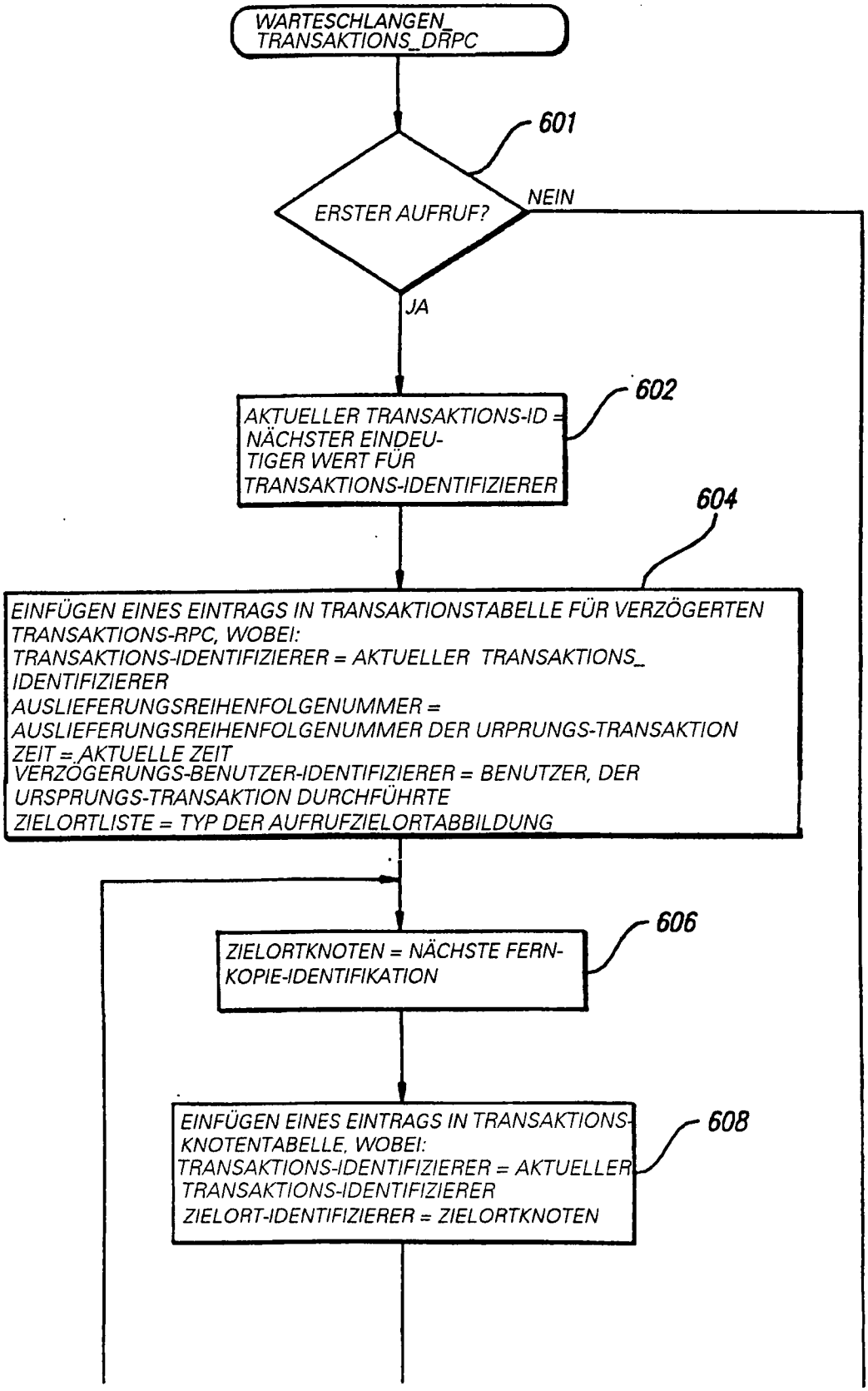

570
IST JEDER FELDWERT IN DER REIHE GLEICH SEINEM ENT-SPRECHENNDEN ALT_ WERT-PARA-METER?
NEIN
572
AUSNAHMEN
JA
LÖSCHEN DER REIHE AM ENTFERNTEN ORT
574
RÜCKSETZEN DER GESAMTREPLIZIER-VARIABLEN
576
ENDE
578

FIG. 5C(2)

FIG. 6(a)


WARTESCHLANGEN_TRANSAKTIONS_DRPC
601
ERSTER AUFRUF?
NEIN
JA
602
AKTUELLER TRANSAKTIONS-ID = NÄCHSTER EINDEUTIGER WERT FÜR TRANSAKTIONS-IDENTIFIZIERER
604
EINFÜGEN EINES EINTRAGS IN TRANSAKTIONSTABELLE FÜR VERZÖGERTEN TRANSAKTIONS-RPC, WOBEI:
TRANSAKTIONS-IDENTIFIZIERER = AKTUELLER TRANSAKTIONS_IDENTIFIZIERER
AUSLIEFERUNGSREIHENFOLGENUMMER = AUSLIEFERUNGSREIHENFOLGENUMMER DER URPRUNGS-TRANSAKTION
ZEIT = AKTUELLE ZEIT
VERZÖGERUNGS-BENUTZER-IDENTIFIZIERER = BENUTZER, DER URSPRUNGS-TRANSAKTION DURCHFÜHRTE
ZIELORTLISTE = TYP DER AUFRUFZIELORTABBILDUNG
606
ZIELORTKNOTEN = NÄCHSTE FERNKOPIE-IDENTIFIKATION
608
EINFÜGEN EINES EINTRAGS IN TRANSAKTIONSKNOTENTABELLE, WOBEI:
TRANSAKTIONS-IDENTIFIZIERER = AKTUELLER TRANSAKTIONS-IDENTIFIZIERER
ZIELORT-IDENTIFIZIERER = ZIELORTKNOTEN

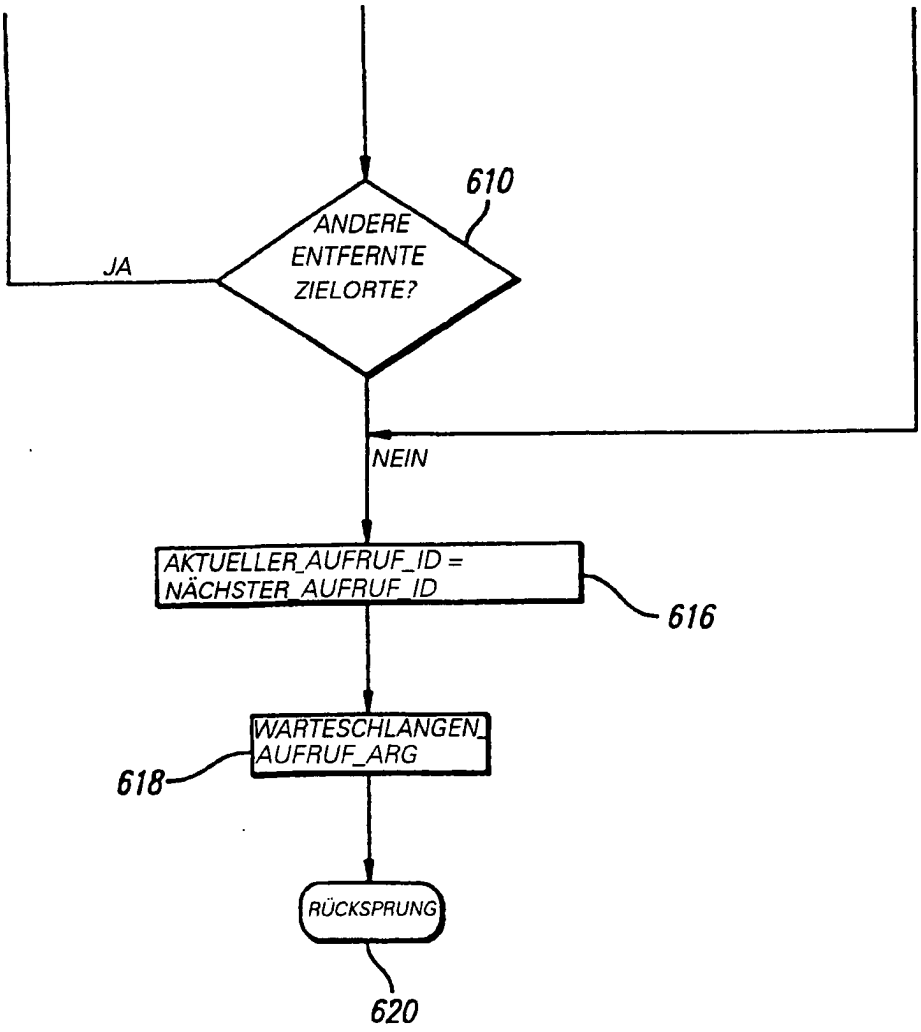

610
ANDERE ENTFERNTE ZIELORTE?
JA
NEIN
AKTUELLER_AUFRUF_ID = NÄCHSTER_AUFRUF_ID
616
WARTESCHLANGEN_AUFRUF_ARG
618
RÜCKSPRUNG
620

FIG. 6(b)

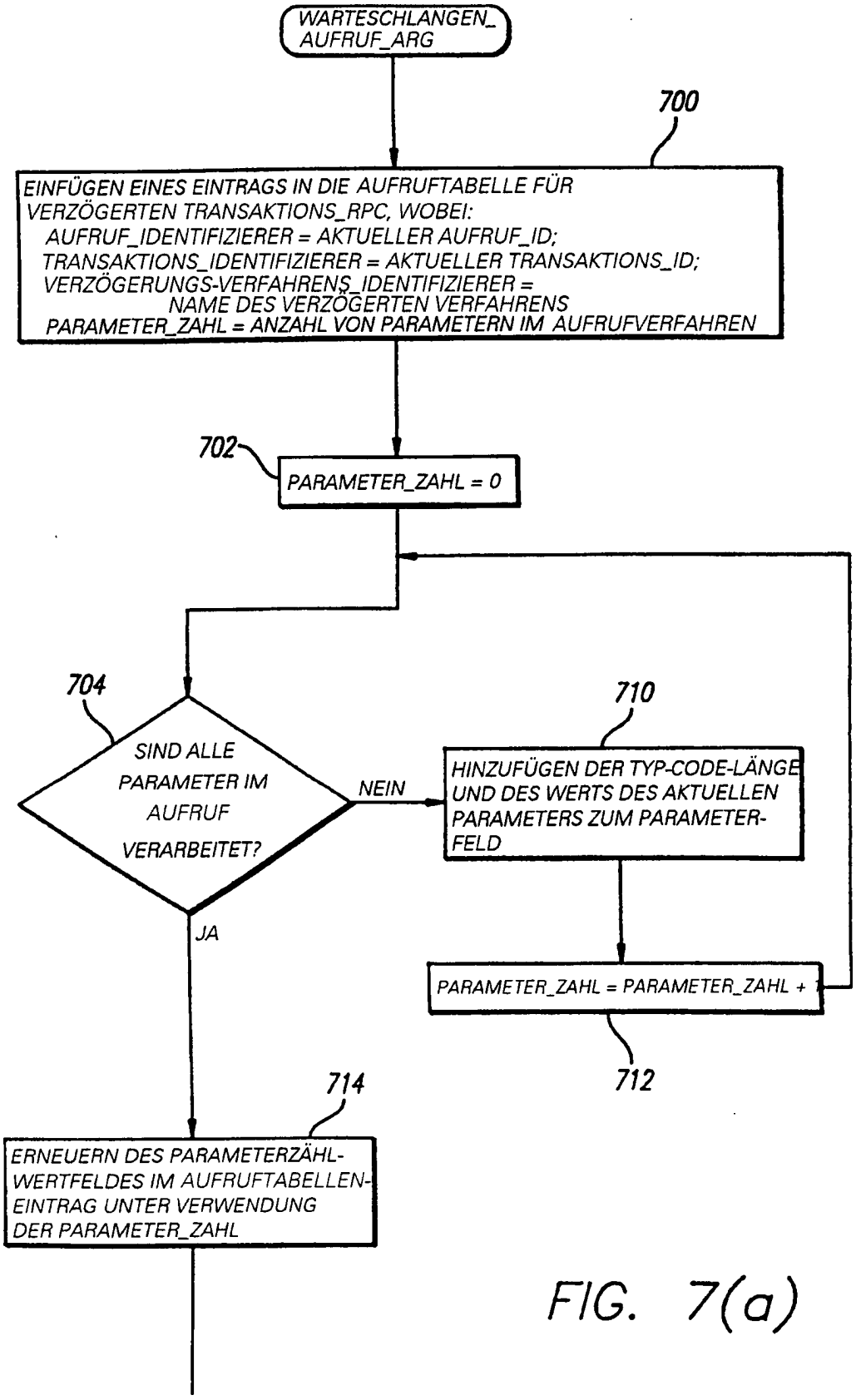

WARTESCHLANGEN_
AUFRUF_ARG
700
EINFÜGEN EINES EINTRAGS IN DIE AUFRUFTABELLE FÜR
VERZÖGERTEN TRANSAKTIONS_RPC, WOBEI:
AUFRUF_IDENTIFIZIERER = AKTUELLER AUFRUF_ID;
TRANSAKTIONS_IDENTIFIZIERER = AKTUELLER TRANSAKTIONS_ID;
VERZÖGERUNGS-VERFAHRENS_IDENTIFIZIERER =
NAME DES VERZÖGERTEN VERFAHRENS
PARAMETER_ZAHL = ANZAHL VON PARAMETERN IM AUFRUFVERFAHREN
702
PARAMETER_ZAHL = 0
704
SIND ALLE PARAMETER IM AUFRUF VERARBEITET?
NEIN
710
HINZUFÜGEN DER TYP-CODE-LÄNGE UND DES WERTS DES AKTUELLEN PARAMETERS ZUM PARAMETER-FELD
PARAMETER_ZAHL = PARAMETER_ZAHL + 1
712
JA
714
ERNEUERN DES PARAMETERZÄHL-WERTFELDES IM AUFRUFTABELLEN-EINTRAG UNTER VERWENDUNG DER PARAMETER_ZAHL

FIG. 7(a)

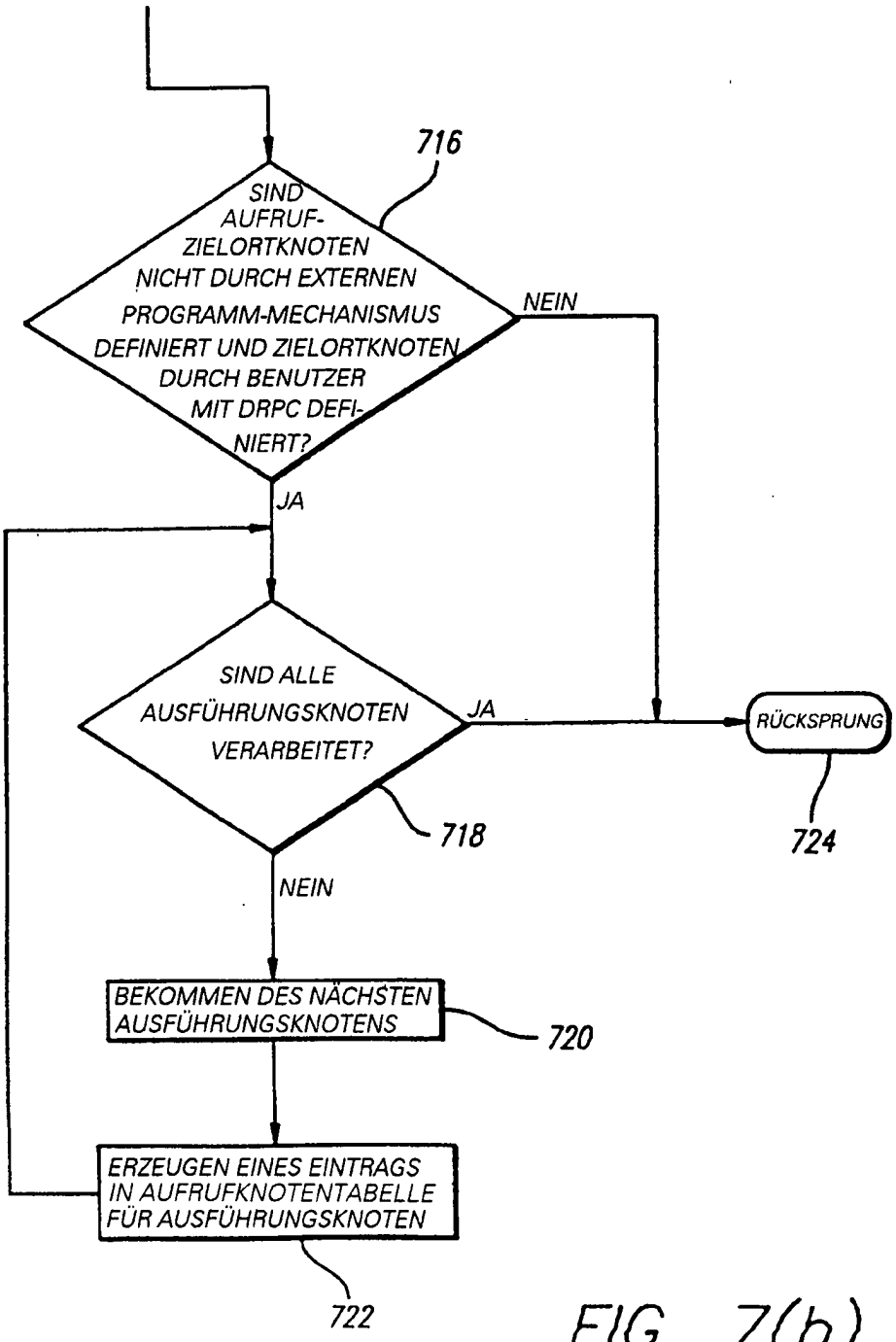

716
SIND AUFRUF-ZIELORTKNOTEN NICHT DURCH EXTERNEN PROGRAMM-MECHANISMUS DEFINIERT UND ZIELORTKNOTEN DURCH BENUTZER MIT DRPC DEFINIERT?
NEIN
JA
SIND ALLE AUSFÜHRUNGSKNOTEN VERARBEITET?
JA
RÜCKSPRUNG
718
724
NEIN
BEKOMMEN DES NÄCHSTEN AUSFÜHRUNGSKNOTENS
720
ERZEUGEN EINES EINTRAGS IN AUFRUFKNOTENTABELLE FÜR AUSFÜHRUNGSKNOTEN
722

FIG. 7(b)

TRANSAKTIONSTABELLE

TRANSAKTIONS-IDENTIFIZIERER	AUSGABE-REIHEN-FOLGE-NUMMER	AUS-FÜHRUNGS-ZEIT	VERZÖGERUNGS-BENUTZER-IDENTIFIZIERER	ZIELORT-LISTE
105	283	10:00	BENUTZER A	D

TRANSAKTIONSKNOTENTABELLE

TRANSAKTIONS-IDENTIFIZIERER	ZIELORT-KNOTEN
105	DB_B

AUFRUFTABELLE

AUFRUF-IDENT.	TRANSAKTIONS-IDENTIFIZIERER	VERZÖGERUNGS-VERFAHRENS-IDENTIFIZIERER	PARAMETER-ZÄHLWERT	PARAMETER
10056	105	LAGERBESTANDS-ERNEUERUNG	4	206PRODUKT103400206PRODUKT1033500

AUFRUFKNOTENTABELLE

TRANSAKTIONS-IDENTIFIZIERER	AUFRUF-IDENTIFIZIERER	ZIELORT-KNOTEN
105	10056	DB_B

FIG. 8A

TRANSAKTIONSTABELLE

TRANSAKTIONS-IDENTIFIZIERER	AUSGABE-REIHEN-FOLGE-NUMMER	AUS-FÜHRUNGS-ZEIT	VERZÖGERUNGS-BENUTZER-IDENTIFIZIERER	ZIELORT-LISTE

TRANSAKTIONSKNOTENTABELLE

TRANSAKTIONS-IDENTIFIZIERER	ZIELORT-KNOTEN

AUFRUFTABELLE

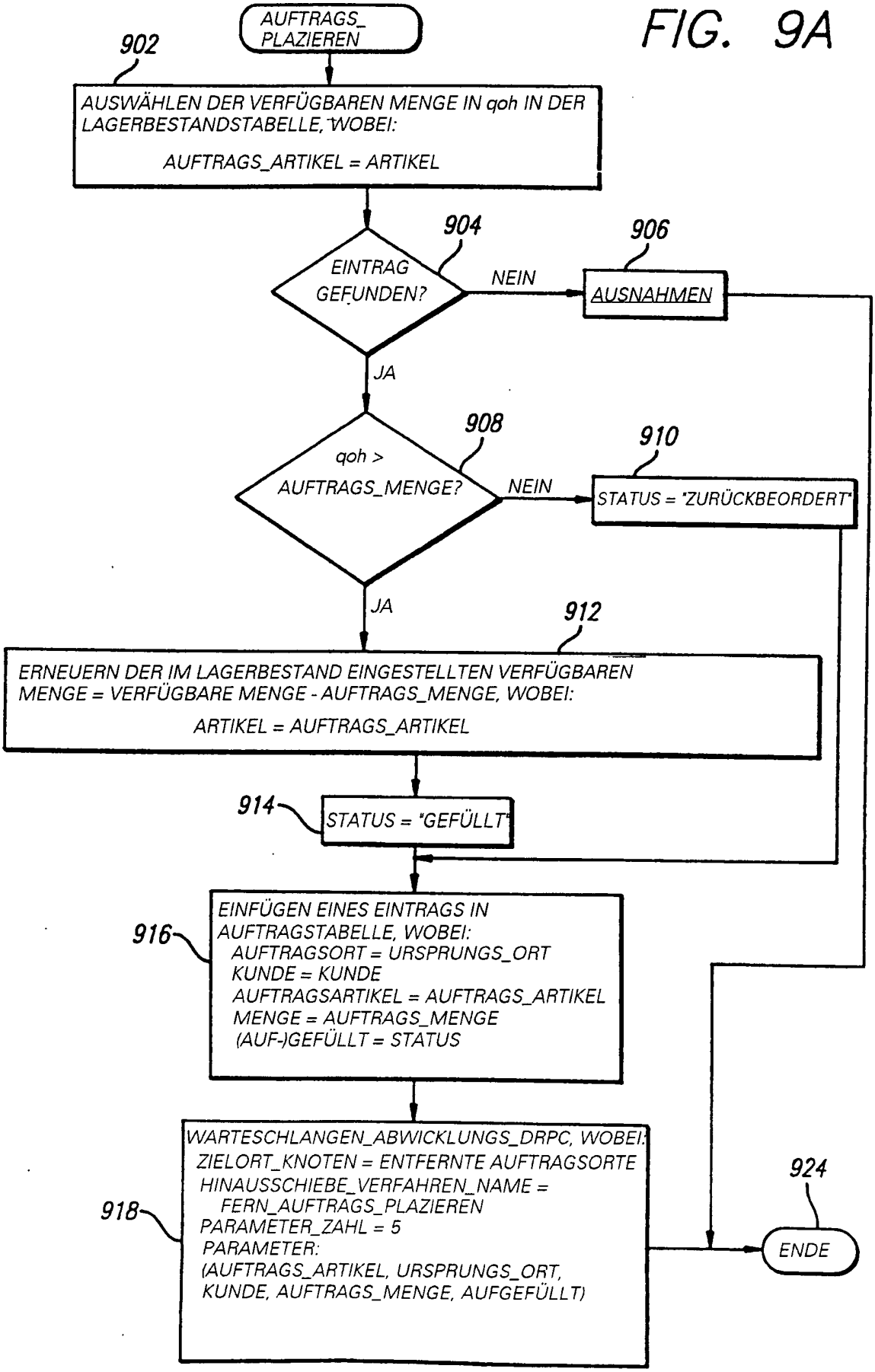
AUFRUF-IDENT.	TRANSAKTIONS-IDENTIFIZIERER	VERZÖGERUNGS-VERFAHRENS-IDENTIFIZIERER	PARAMETER-ZÄHLWERT	PARAMETER
10056		LAGERBESTANDS-ERNEUERUNG	4	206PRODUKT103400206PRODUKT1033500

AUFRUFKNOTENTABELLE

TRANSAKTIONS-IDENTIFIZIERER	AUFRUF-IDENTIFIZIERER	ZIELORT-KNOTEN
	10056	DB_B

FIG. 8B

FIG. 9A


AUFTRAGS_ PLAZIEREN
902
AUSWÄHLEN DER VERFÜGBAREN MENGE IN qoh IN DER LAGERBESTANDSTABELLE, WOBEI:
AUFTRAGS_ARTIKEL = ARTIKEL
904
EINTRAG GEFUNDEN?
NEIN
906
AUSNAHMEN
JA
908
qoh > AUFTRAGS_MENGE?
NEIN
910
STATUS = "ZURÜCKBEORDERT"
JA
912
ERNEUERN DER IM LAGERBESTAND EINGESTELLTEN VERFÜGBAREN MENGE = VERFÜGBARE MENGE - AUFTRAGS_MENGE, WOBEI:
ARTIKEL = AUFTRAGS_ARTIKEL
914
STATUS = "GEFÜLLT"
916
EINFÜGEN EINES EINTRAGS IN AUFTRAGSTABELLE, WOBEI:
AUFTRAGSORT = URSPRUNGS_ORT
KUNDE = KUNDE
AUFTRAGSARTIKEL = AUFTRAGS_ARTIKEL
MENGE = AUFTRAGS_MENGE
(AUF-)GEFÜLLT = STATUS
918
WARTESCHLANGEN_ABWICKLUNGS_DRPC, WOBEI:
ZIELORT_KNOTEN = ENTFERNTE AUFTRAGSORTE
HINAUSSCHIEBE_VERFAHREN_NAME = FERN_AUFTRAGS_PLAZIEREN
PARAMETER_ZAHL = 5
PARAMETER:
(AUFTRAGS_ARTIKEL, URSPRUNGS_ORT, KUNDE, AUFTRAGS_MENGE, AUFGEFÜLLT)
924
ENDE

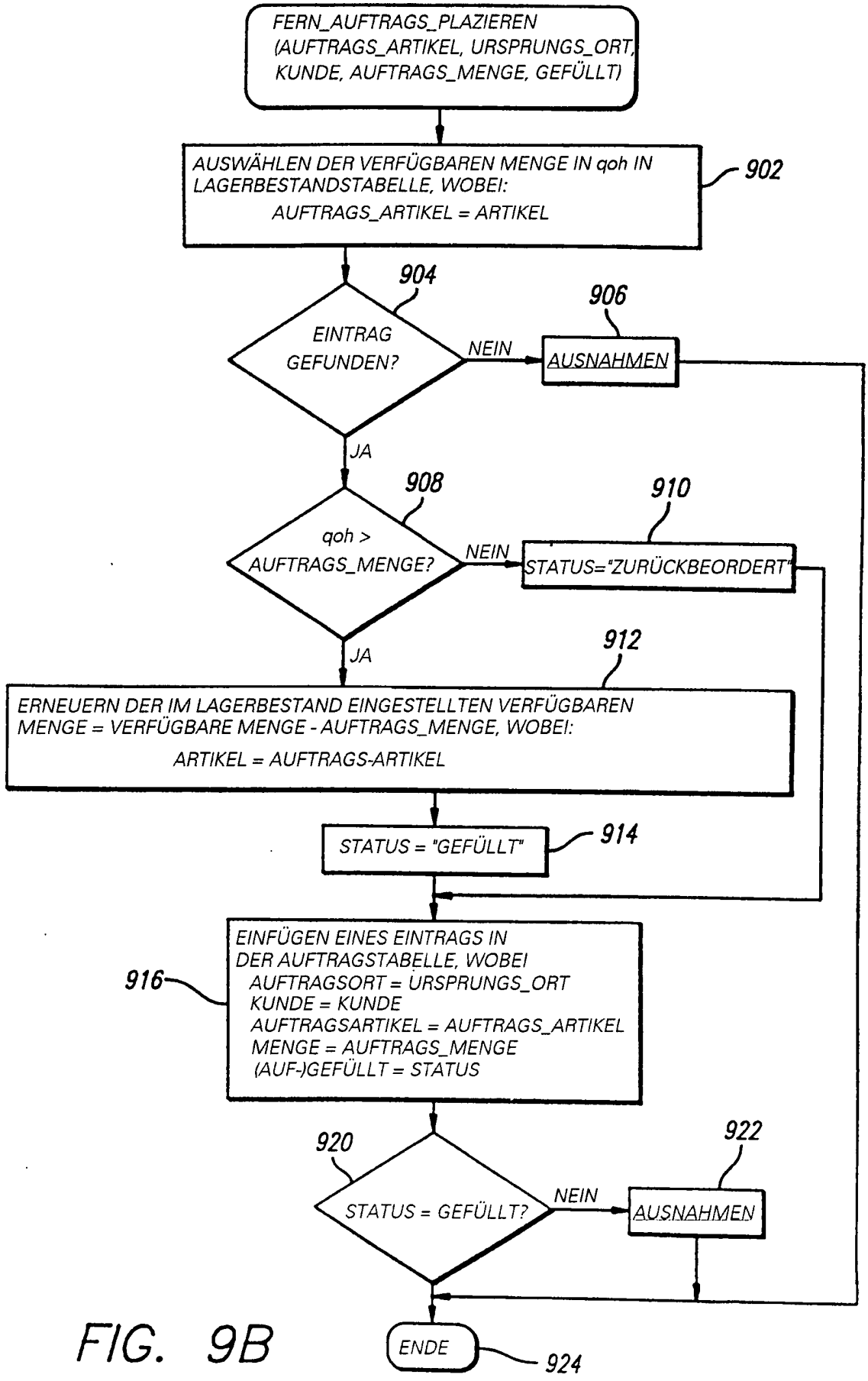

FERN_AUFTRAGS_PLAZIEREN (AUFTRAGS_ARTIKEL, URSPRUNGS_ORT, KUNDE, AUFTRAGS_MENGE, GEFÜLLT)
AUSWÄHLEN DER VERFÜGBAREN MENGE IN qoh IN LAGERBESTANDSTABELLE, WOBEI:
AUFTRAGS_ARTIKEL = ARTIKEL
902
904
EINTRAG GEFUNDEN?
NEIN
906
AUSNAHMEN
JA
908
qoh > AUFTRAGS_MENGE?
NEIN
910
STATUS="ZURÜCKBEORDERT"
JA
912
ERNEUERN DER IM LAGERBESTAND EINGESTELLTEN VERFÜGBAREN MENGE = VERFÜGBARE MENGE - AUFTRAGS_MENGE, WOBEI:
ARTIKEL = AUFTRAGS-ARTIKEL
STATUS = "GEFÜLLT"
914
EINFÜGEN EINES EINTRAGS IN DER AUFTRAGSTABELLE, WOBEI
AUFTRAGSORT = URSPRUNGS_ORT
KUNDE = KUNDE
AUFTRAGSARTIKEL = AUFTRAGS_ARTIKEL
MENGE = AUFTRAGS_MENGE
(AUF-)GEFÜLLT = STATUS
916
920
STATUS = GEFÜLLT?
NEIN
922
AUSNAHMEN
ENDE
924

FIG. 9B

TRANSAKTIONSTABELLE

TRANSAKTIONS-IDENTIFIZIERER	AUSGABE-REIHENFOLGE-NUMMER	ÜBER-TRAGS-ZEIT	VERZÖGERUNGS-BENUTZER-IDENTIFIZIERER	ZIELORT-LISTE
4	10	1:00	BENUTZER A	D

TRANSAKTIONSKNOTENTABELLE

TRANSAKTIONS-IDENTIFIZIERER	ZIELORT-KNOTEN
4	DB_B

AUFRUFTABELLE

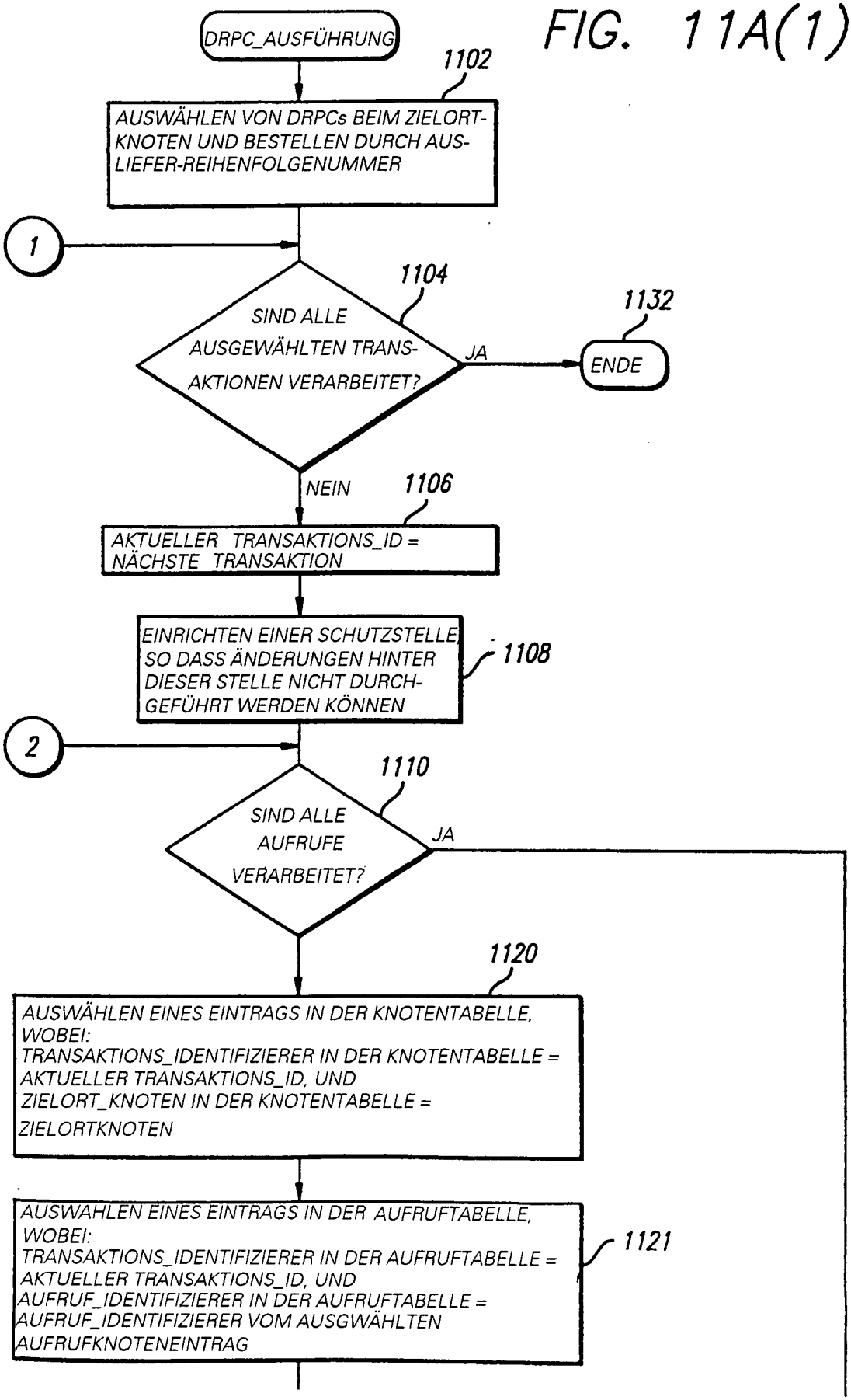
AUFRUF-IDENT.	TRANSAKTIONS-IDENTIFIZIERER	VERZÖGERUNGS-VERFAHRENS-IDENTIFIZIERER	PARAMETER-ZÄHLWERT	PARAMETEB
1	4	FERNBESTELLUNGS-PLAZIERUNG	4	206PRODUKT203DB_A1025010250206ERFÜLLT0

AUFRUFKNOTENTABELLE

TRANSAKTIONS-IDENTIFIZIERER	AUFRUF-IDENTIFIZIERER	ZIELORT-KNOTEN
4	1	DB_B

FIG. 10

FIG. 11A(1)


DRPC_AUSFÜHRUNG
1102
AUSWÄHLEN VON DRPCs BEIM ZIELORT-KNOTEN UND BESTELLEN DURCH AUS-LIEFER-REIHENFOLGENUMMER
1
1104
SIND ALLE AUSGEWÄHLTEN TRANS-AKTIONEN VERARBEITET?
JA
1132
ENDE
NEIN
1106
AKTUELLER TRANSAKTIONS_ID = NÄCHSTE TRANSAKTION
EINRICHTEN EINER SCHUTZSTELLE, SO DASS ÄNDERUNGEN HINTER DIESER STELLE NICHT DURCH-GEFÜHRT WERDEN KÖNNEN
1108
2
1110
SIND ALLE AUFRUFE VERARBEITET?
JA
1120
AUSWÄHLEN EINES EINTRAGS IN DER KNOTENTABELLE, WOBEI:
TRANSAKTIONS_IDENTIFIZIERER IN DER KNOTENTABELLE = AKTUELLER TRANSAKTIONS_ID, UND
ZIELORT_KNOTEN IN DER KNOTENTABELLE = ZIELORTKNOTEN
AUSWAHLEN EINES EINTRAGS IN DER AUFRUFTABELLE, WOBEI:
TRANSAKTIONS_IDENTIFIZIERER IN DER AUFRUFTABELLE = AKTUELLER TRANSAKTIONS_ID, UND
AUFRUF_IDENTIFIZIERER IN DER AUFRUFTABELLE = AUFRUF_IDENTIFIZIERER VOM AUSGWÄHLTEN AUFRUFKNOTENEINTRAG
1121

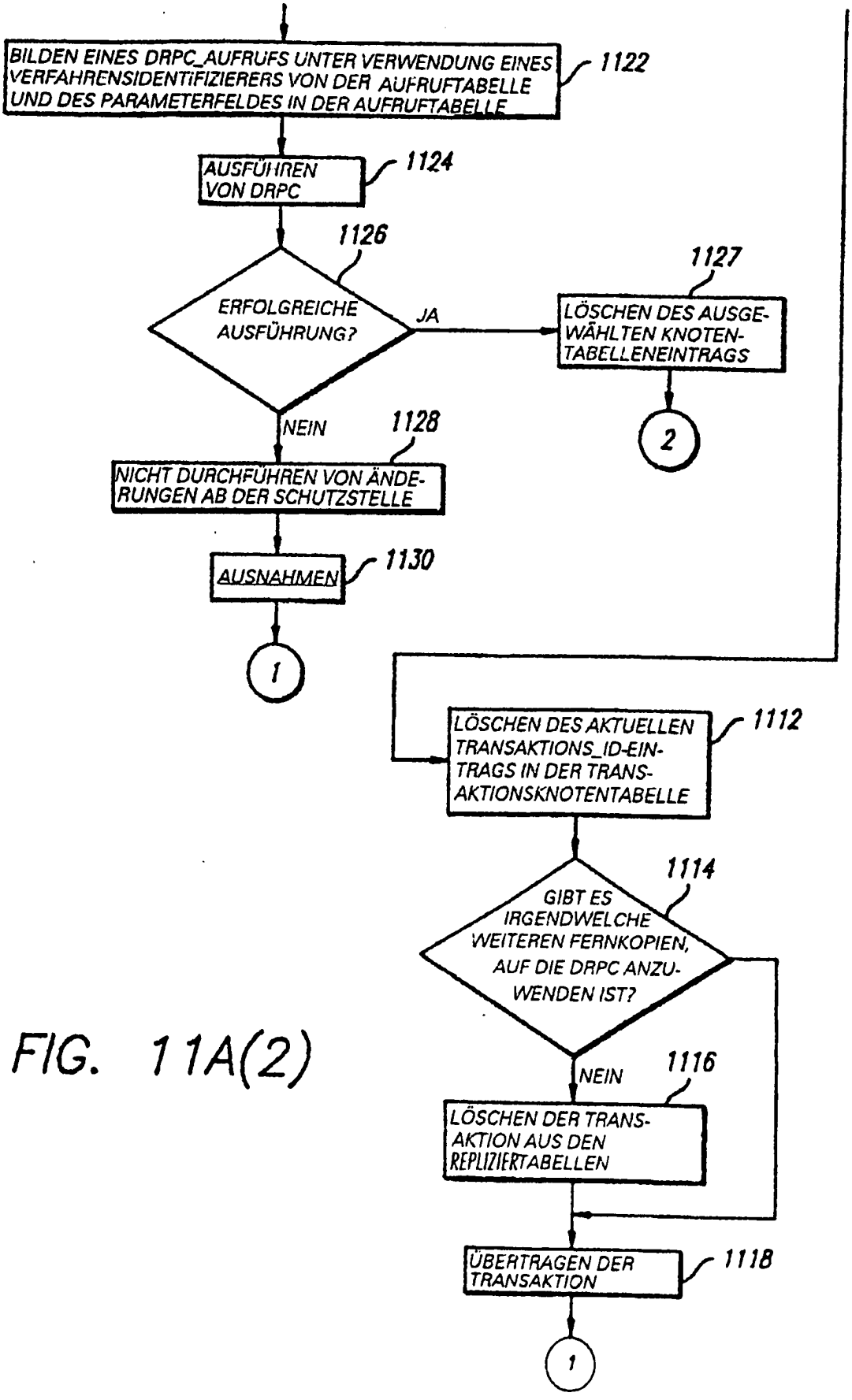

BILDEN EINES DRPC_AUFRUFS UNTER VERWENDUNG EINES VERFAHRENSIDENTIFIZIERERS VON DER AUFRUFTABELLE UND DES PARAMETERFELDES IN DER AUFRUFTABELLE
1122
AUSFÜHREN VON DRPC
1124
1126
ERFOLGREICHE AUSFÜHRUNG?
JA
1127
LÖSCHEN DES AUSGEWÄHLTEN KNOTEN-TABELLENEINTRAGS
2
NEIN
1128
NICHT DURCHFÜHREN VON ÄNDERUNGEN AB DER SCHUTZSTELLE
AUSNAHMEN
1130
1
LÖSCHEN DES AKTUELLEN TRANSAKTIONS_ID-EINTRAGS IN DER TRANSAKTIONSKNOTENTABELLE
1112
1114
GIBT ES IRGENDWELCHE WEITEREN FERNKOPIEN, AUF DIE DRPC ANZUWENDEN IST?
NEIN
1116
LÖSCHEN DER TRANSAKTION AUS DEN REPLIZIERTABELLEN
ÜBERTRAGEN DER TRANSAKTION
1118
1

FIG. 11A(2)

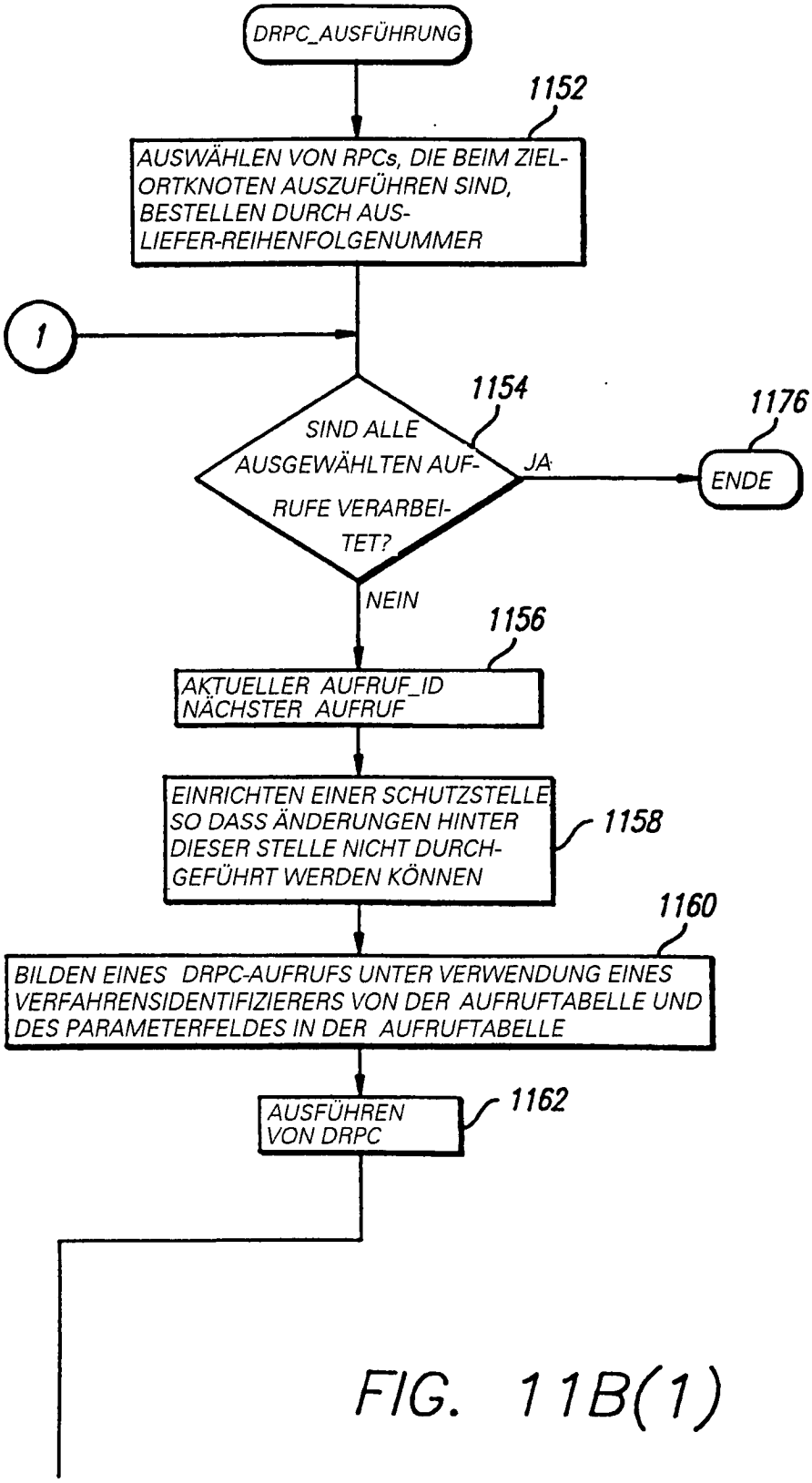

DRPC_AUSFÜHRUNG
1152
AUSWÄHLEN VON RPCs, DIE BEIM ZIEL-ORTKNOTEN AUSZUFÜHREN SIND, BESTELLEN DURCH AUS-LIEFER-REIHENFOLGENUMMER
1
1154
SIND ALLE AUSGEWÄHLTEN AUF-RUFE VERARBEI-TET?
JA
1176
ENDE
NEIN
1156
AKTUELLER AUFRUF_ID NÄCHSTER AUFRUF
EINRICHTEN EINER SCHUTZSTELLE SO DASS ÄNDERUNGEN HINTER DIESER STELLE NICHT DURCH-GEFÜHRT WERDEN KÖNNEN
1158
1160
BILDEN EINES DRPC-AUFRUFS UNTER VERWENDUNG EINES VERFAHRENSIDENTIFIZIERERS VON DER AUFRUFTABELLE UND DES PARAMETERFELDES IN DER AUFRUFTABELLE
AUSFÜHREN VON DRPC
1162

FIG. 11B(1)

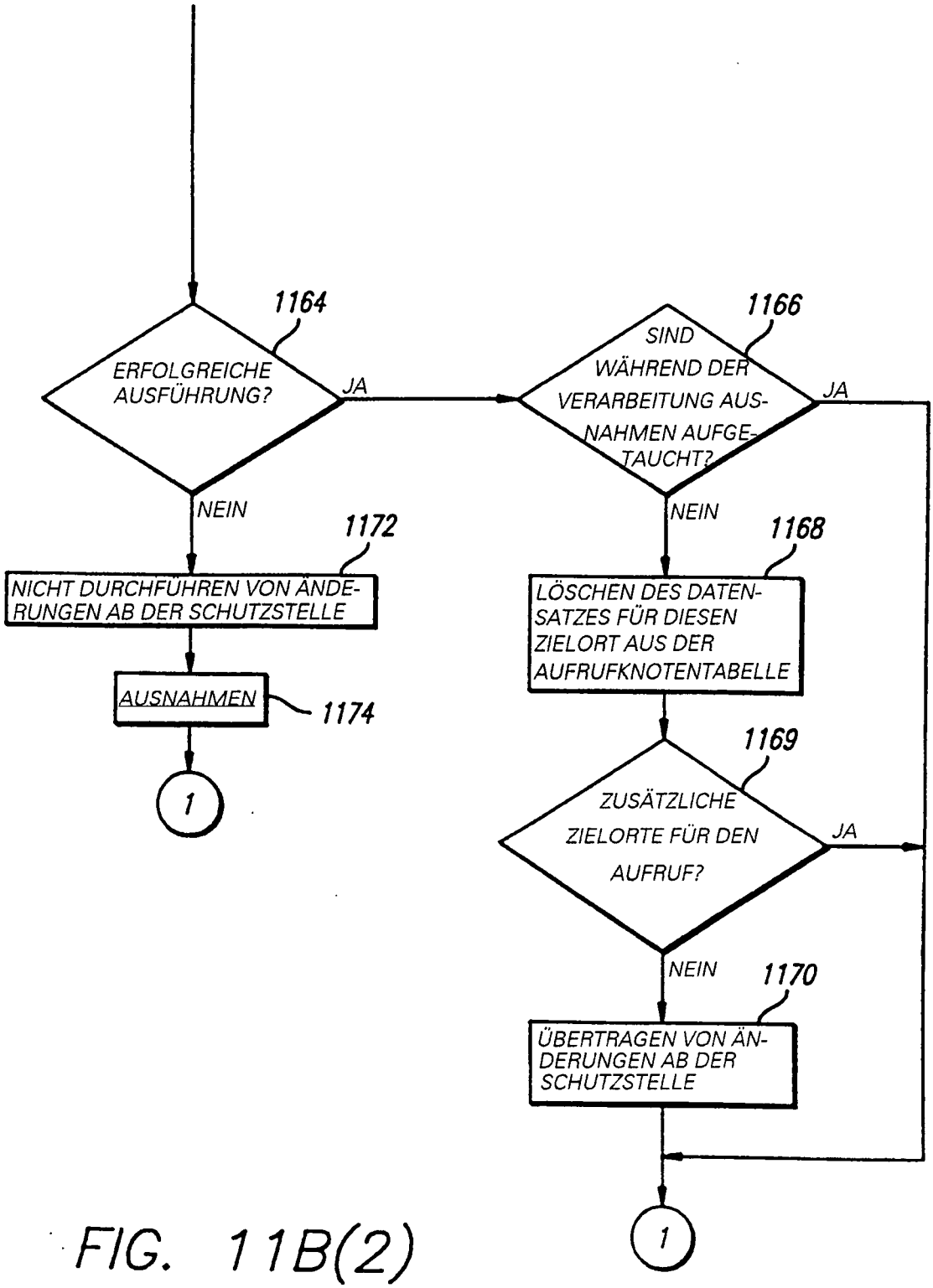

1164
ERFOLGREICHE AUSFÜHRUNG?
JA
1166
SIND WÄHREND DER VERARBEITUNG AUSNAHMEN AUFGETAUCHT?
JA
NEIN
1172
NICHT DURCHFÜHREN VON ÄNDERUNGEN AB DER SCHUTZSTELLE
AUSNAHMEN
1174
1
NEIN
1168
LÖSCHEN DES DATENSATZES FÜR DIESEN ZIELORT AUS DER AUFRUFKNOTENTABELLE
1169
ZUSÄTZLICHE ZIELORTE FÜR DEN AUFRUF?
JA
NEIN
1170
ÜBERTRAGEN VON ÄNDERUNGEN AB DER SCHUTZSTELLE
1

FIG. 11B(2)

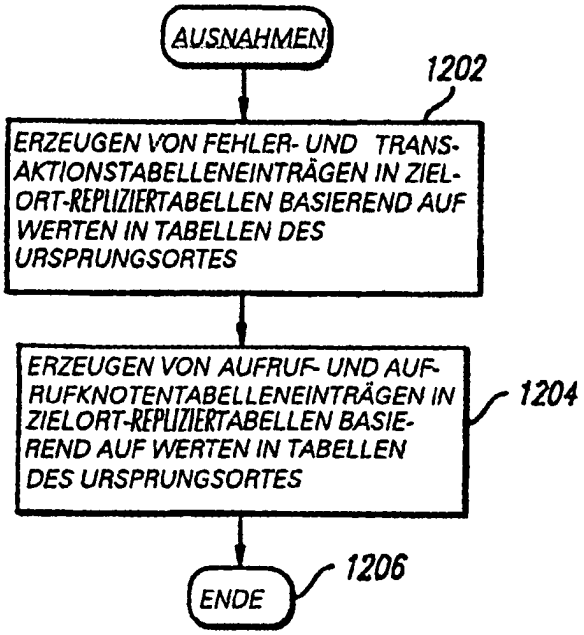

AUSNAHMEN
1202
ERZEUGEN VON FEHLER- UND TRANS-AKTIONSTABELLENEINTRÄGEN IN ZIEL-ORT-REPLIZIERTABELLEN BASIEREND AUF WERTEN IN TABELLEN DES URSPRUNGSORTES
ERZEUGEN VON AUFRUF- UND AUF-RUFKNOTENTABELLENEINTRÄGEN IN ZIELORT-REPLIZIERTABELLEN BASIE-REND AUF WERTEN IN TABELLEN DES URSPRUNGSORTES
1204
ENDE
1206

FIG. 12

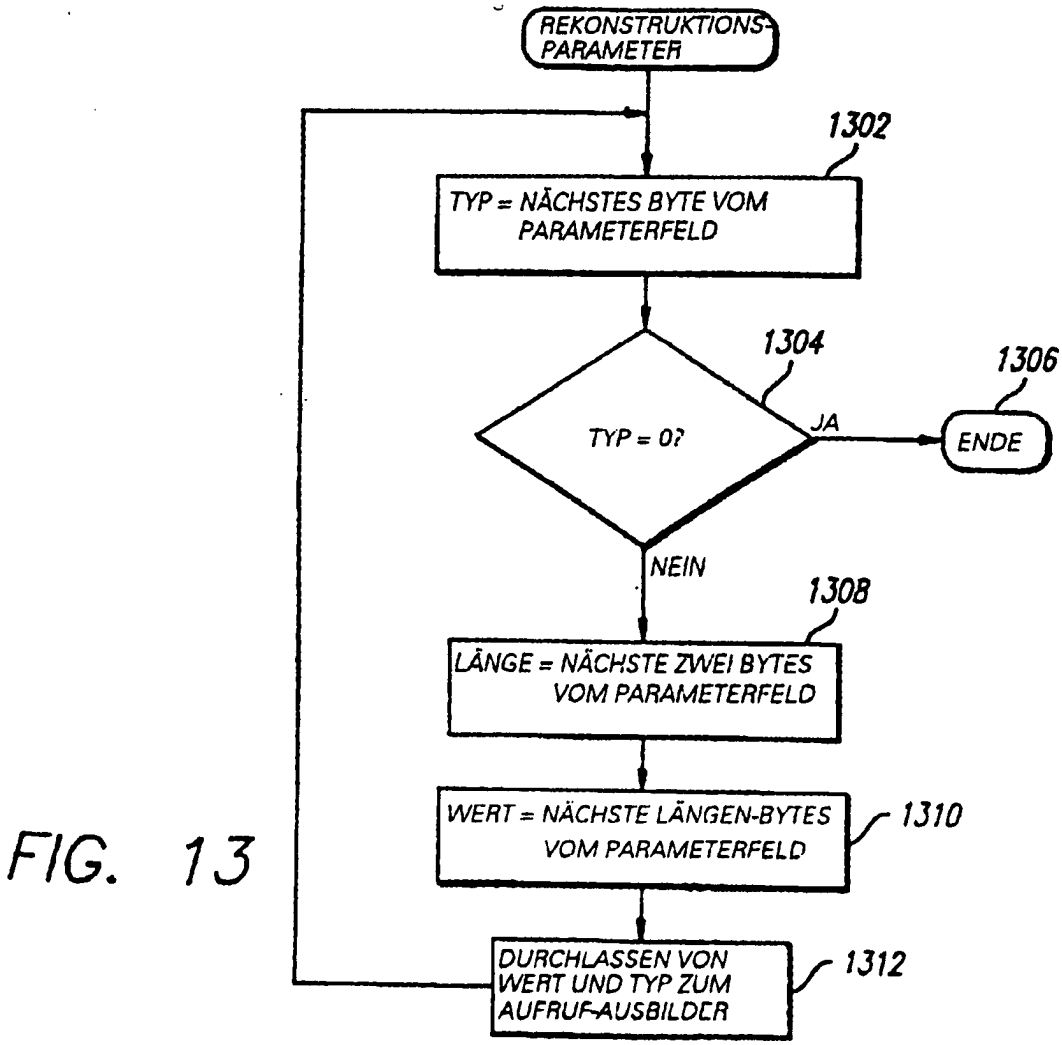

REKONSTRUKTIONS-PARAMETER
1302
TYP = NÄCHSTES BYTE VOM PARAMETERFELD
1304
TYP = 0?
JA
1306
ENDE
NEIN
1308
LÄNGE = NÄCHSTE ZWEI BYTES VOM PARAMETERFELD
WERT = NÄCHSTE LÄNGEN-BYTES VOM PARAMETERFELD
1310
DURCHLASSEN VON WERT UND TYP ZUM AUFRUF-AUSBILDER
1312

FIG. 13