(19) österreichisches patentamt

(10) **AT 500 050 A2 2005-10-15**

(12) Österreichische Patentanmeldung

(21) Anmeldenummer: **A 1332/2002**

(22) Anmeldetag: **05.09.2002**

(43) Veröffentlicht am: **15.10.2005**

(51) Int. Cl.[7]: **G06F 11/00**

(73) Patentanmelder:

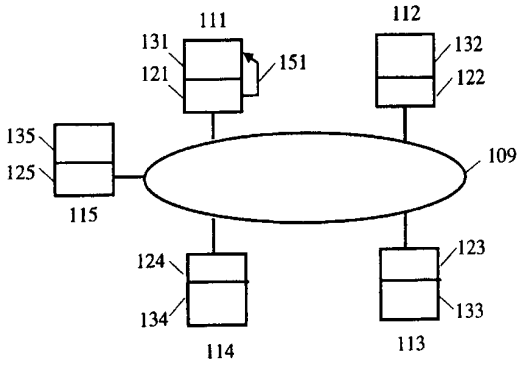
FTS COMPUTERTECHNIK GES.M.B.H.
A-2500 BADEN-SIEGENFELD (AT)

(72) Erfinder:

KOPETZ HERMANN
BADEN-SIEGENFELD (AT)

(54) VERFAHREN UND APPARAT ZUR FEHLERERKENNUNG IN EINEM VERTEILTEN ECHTZEITCOMPUTERSYSTEM

(57) Ein verteiltes Echtzeitcomputersystem besteht aus einer Anzahl von Knotenrechnern, die über ein Echtzeitkommunikationssystem Nachrichten austauschen und die über lokale Schnittstellen einen technischen Prozess beobachten und beeinflussen. Solche verteilte Echtzeitcomputersysteme werden zunehmend zur Steuerung sicherheitskritischer Anwendungen eingesetzt. Beispiele für solche Anwendungen sind die Steuerung der Bremsen in einem Kraftfahrzeug oder ein Flugleitsystem in einem Flugzeug. In diesen Anwendungen ist es von großer Wichtigkeit, dass ein auftretender Fehler, der zum Verlust der Datenkonsistenz im verteilten System führen kann, schnell erkannt wird. Beispiele für solche Fehler sind der Verlust von Nachrichten, der Ausfall eines Knotenrechners, oder ein Fehler in der dem Knotenrechner zugeordneten Prozessperipherie. Die vorliegende Erfindung beschreibt ein neues differenziertes Verfahren zur schnellen Fehlererkennung ein einem verteilten Echtzeitcomputersystem, das die Datenkonsistenz innerhalb einer Funktionsgruppe überwacht. Dieses neue Verfahren kann auch in der Hardware des Kommunikations- Controllers implementiert werden.



AT 500 050 A2 2005-10-15

ZUSAMMENFASSUNG

Ein verteiltes Echtzeitcomputersystem besteht aus einer Anzahl von Knotenrechnern, die über ein Echtzeitkommunikationssystem Nachrichten austauschen und die über lokale Schnittstellen einen technischen Prozess beobachten und beeinflussen. Solche verteilte Echtzeitcomputersysteme werden zunehmend zur Steuerung sicherheitskritischer Anwendungen eingesetzt. Beispiele für solche Anwendungen sind die Steuerung der Bremsen in einem Kraftfahrzeug oder ein Flugleitsystem in einem Flugzeug. In diesen Anwendungen ist es von großer Wichtigkeit, das ein auftretender Fehler, der zum Verlust der Datenkonsistenz im verteilten System führen kann, schnell erkannt wird. Beispiele für solche Fehler sind der Verlust von Nachrichten, der Ausfall eines Knotenrechners, oder ein Fehler in der dem Knotenrechner zugeordneten Prozessperipherie. Die vorliegende Erfindung beschreibt ein neues differenziertes Verfahren zur schnellen Fehlererkennung ein einem verteilten Echtzeitcomputersystem, das die Datenkonsistenz innerhalb einer Funktionsgruppe überwacht. Dieses neue Verfahren kann auch in der Hardware des Kommunikationskontrollers implementiert werden.

Einreichkopie, 5.9. 2002

Verfahren und Apparat zur Fehlererkennung in einem verteilten Echtzeitcomputersystem

Zitierte Patente:

[1] US 5694542 issued on Dec. 12, 1989: A loosely coupled distributed computer system with node synchronization for precision in real time.

[2] EP 0 658 257 erteilt am 18.12.1996: Kommunikationskontrolleinheit und Verfahren zur Übermittlung von Nachrichten.

[3] US 5887143 issued on March 23, 1999: Time-Triggered Communcation Control Unit and Communication.

[4] DE 19753288 eingereicht am 3. Dezember 1996: Effizientes Quittungsverfahren

Sonstige Literatur:

[5] OMG Smart Transducer Interface Standard: URL: http://omg.org

[6] Kopetz, H. (1997). *Real-Time Systems, Design Principles for Distributed Embedded Applications; ISBN: 0-7923-9894-7.* Boston. Kluwer Academic Publishers.

TECHNISCHES UMFELD

Diese Erfindung betrifft ein Verfahren und einen Kommunikationskontroller zur konsistenten Erkennung von Fehlern in einem verteilten Echtzeitcomputersystem, in dem eine Vielzahl von Knotenrechnern mittels Nachrichten kommunizieren.

HINTERGRUND DIESER ERFINDUNG

Ein verteiltes Echtzeitcomputersystem besteht aus einer Anzahl von Knotenrechnern, die über ein Echtzeitkommunikationssystem Nachrichten austauschen und die über lokale Schnittstellen einen technischen Prozess beobachten und beeinflussen. Solche verteilte Echtzeitcomputersysteme werden zunehmend zur Steuerung sicherheitskritischer Anwendungen eingesetzt. Beispiele für solche Anwendungen sind die Steuerung der Bremsen in einem Kraftfahrzeug oder ein Flugleitsystem in einem Flugzeug. In diesen Anwendungen ist es von großer Wichtigkeit, dass ein auftretender Fehler, der zum Verlust der Datenkonsistenz im verteilten System führen kann, schnell erkannt wird.

Beispiele für solche Fehler sind der Verlust von Nachrichten, der Ausfall eines Knotenrechners, oder ein Fehler in der dem Knotenrechner zugeordneten Prozessperipherie.

Eine besonders häufige und damit wichtige Fehlerklasse in verteilten Echtzeitsystemen ist die Klasse der Crash/Omission (CO) Fehler. Die Fehlerklasse der CO-Fehler umfasst den Verlust von Nachrichten oder den Totalausfall von Knotenrechnern. Viele der bekannten Kommunikationsprotokolle enthalten Mechanismen, um CO Fehler zu erkennen [6]. Eine Verfahren das CO Fehler in einem verteilten Echtzeitsystem erkennen kann wurde in den Patentschriften [1-4] offengelegt. Dieses Verfahren geht davon aus, dass alle Rechner eines Subsystems (Clusters) an allen Funktionen des Systems beteiligt sind und die Inkonsistenz, die durch einen CO Fehler eines Kontenrechners verursacht wird, dadurch behoben wird, dass der betroffene Knotenrechner keine weiteren Nachrichten sendet bis er sich wieder erfolgreich reintegriert hat. Diese Art der Fehlererkennung und Fehlerbehandlung wird in einigen Anwendungsfällen als zu restriktiv angesehen, da auch der Ausfall einer einzigen Funktion eines Knotenrechners, der an mehreren Funktionen beteiligt ist, zum Ausfall des gesamten Knotenrechners führt.

Dieser Nachteil der auf einen Rechnerknoten bezogenen Fehlererkennung und Fehlerbehandlung wird durch die vorliegende Erfindung behoben. Entsprechend dieser Erfindung wird die Fehlererkennung funktionsbezogen durchgeführt, wobei ein Knotenrechner an mehreren Funktionen beteiligt sein kann. Die Datenkonsistenz wird innerhalb von Funktionsgruppen sichergestellt. Zum Beispiel können in einem Fahrzeug die vier Knotenrechner, die die Steuerung der Bremsen an den vier Rädern vornehmen, eine solche Funktionsgruppe bilden. In dieser Anwendung ist es wichtig, dass alle vier Radknotenrechner eine konsistente Sicht des Bremssystems haben. Ein Radknotenrechner kann jedoch auch an einer anderen Funktion beteiligt sein, die mit der unmittelbaren Bremsfunktion nichts zu tun hat. Ein Fehler in dieser „anderen" Funktion soll nicht zu einem Fehler der Bremsfunktion führen. Aus der Sicht der Anwendung ist es wichtig, dass innerhalb einer Funktionsgruppe ein Konsistenzverlust schnell erkannt wird, der Betrieb der anderen unabhängigen Funktionsgruppen, an denen ein Knotenrechner auch noch beteiligt ist, jedoch nicht beeinträchtigt wird.

Die hier offengelegte Erfindung bringt folgende Vorteile gegenüber dem bekannten Stand der Technik der verteilten Fehlererkennung, wie er z.B. im TTP/C Protokoll [1-4] realisiert ist:

- Es wird eine differenzierte Mitgliedschaft eingeführt, die es ermöglicht Knoten weiter zu betreiben, die in Teilfunktionen inkonsistent sind.
- Durch die Einführung getrennter Funktionsgruppen zur Überwachung der Kommunikation und zur Überwachung eines verteilten Anwendungssystems kann zwischen Fehlern in der Kommunikation und Fehlern in der Anwendung unterschieden werden.
- Der Mitgliedschaftsalgorithmus wird aus dem Übertragungsprotokoll gelöst.
- Durch die harmonische Anbindung der Periodizität an die globale Zeit lassen sich synchrone Multiclustersysteme einfach realisieren.

Durch die hier offengelegte Erfindung ergeben sich somit folgende signifikante wirtschaftliche Vorteile:

- Durch die Einführung von Funktionsgruppen wird die Konstruktion von Knotenrechnern, die an mehreren Funktionsgruppen beteiligt sind, erleichtert. Damit wird eine Reduktion der Anzahl der Knotenrechner in einem System und damit eine Kostenreduktion ermöglicht.
- Die Zuverlässigkeit einer gegebenen Funktion wird dadurch erhöht, dass der Ausfall einer anderen unabhängigen Funktion in einem Knotenrechner keinen Einfluss auf die gegebene Funktion hat.
- Die Trennung der Fehlererkennungsfunktion von der unmittelbaren Protokollfunktion ermöglicht den Aufbau von fehlertoleranten Systemen mit Standardprotokollen, wie, z.B. ETHERNET, da das Fehlerekennungsprotokoll in einer Protokollschicht über dem Standardprotokoll der Datenkommunikation implementiert werden kann.

ZUSAMMENFASSUNG

Ein verteiltes Echtzeitcomputersystem besteht aus einer Anzahl von Knotenrechnern, die über ein Echtzeitkommunikationssystem Nachrichten austauschen und die über lokale Schnittstellen einen technischen Prozess beobachten und beeinflussen. Solche verteilte Echtzeitcomputersysteme werden zunehmend zur Steuerung sicherheitskritischer Anwendungen eingesetzt. Beispiele für solche Anwendungen sind die Steuerung der Bremsen in einem Kraftfahrzeug oder ein Flugleitsystem in einem Flugzeug. In diesen Anwendungen ist es von großer Wichtigkeit, das ein auftretender Fehler, der zum Verlust der Datenkonsistenz im verteilten System führen kann, schnell erkannt wird. Beispiele für solche Fehler sind der Verlust von Nachrichten, der Ausfall eines Knotenrechners, oder ein Fehler in der dem Knotenrechner zugeordneten Prozessperipherie. Die vorliegende Erfindung beschreibt ein neues differenziertes Verfahren zur schnellen Fehlererkennung ein einem verteilten Echtzeitcomputersystem, das die Datenkonsistenz innerhalb einer Funktionsgruppe überwacht. Dieses neue Verfahren kann auch anwendungsunabhängig in der Hardware des Kommunikationskontrollers implementiert werden.

KURZE BESCHREIBUNG DER ABBILDUNGEN

Das vorab beschriebene Ziel und andere neue Eigenschaften der vorliegenden Erfindung werden in den angeführten Abbildungen erläutert.

Fig. 1 zeigt die Struktur eines verteilten Computersystems.

Fig. 2. zeigt den Aufbau eines Funktionsgruppenmitgliedfelds.

Fig. 3 zeigt den Aufbau einer Konfigurationsnachricht.

Fig. 4 zeigt die Struktur des Zeitfeldes.

BESCHREIBUNG EINER REALISIERUNG

Im folgenden Abschnitt wird eine konkrete Realisierung des neuen Verfahrens an einem möglichen Beispiel mit fünf Knotenrechnern, die zwei Funktionsgruppen bilden, gezeigt. Die Erfindung ist nicht auf dieses eine konkrete Realisierungsbeispiel beschränkt. Der in diesem Beispiel vorgestellte Fehlerkerkennungsalgorithmus setzt voraus, dass die Datenübertragung in Kommunikationsrunden periodisch abläuft, dass jeder korrekte sendende Knoten Zugriff auf eine global synchronisierte Zeitbasis hat, dass jeder korrekte sendende Knoten exakt einmal pro Kommunikationsrunde eine Nachricht sendet und dass maximal ein Fehler innerhalb von zwei Übertragungsrunden auftritt.

Fig. 1 zeigt ein Subsystem, ein Cluster, mit fünf Knotenrechner **111**, **112**, **113**, **114**, und **115** die über mindestens je einen Kommunikationskontroller **121**, **122**, **123**, **124**, und **125** und über je einen Host Computer **131**, **132**, **133**, **134**, und **135**, verfügen, und die über ein Echtzeitkommunikationssystem **109** bestehend aus einem oder mehreren Kommunikationskanälen Nachrichten austauschen. Der Kommunikationskontroller exekutiert das Kommunikationsprotokoll. Der Host Computer führt die Applikationssoftware aus. Ein Echzeitkommunikationssystem **109** ist ein Kommunikationssystem dass Nachrichten innerhalb eines bekannten Echtzeitintervalls, der *Transportlatenz*, überträgt. Um die Zuverlässigkeit zu erhöhen, kann das Kommunikationssystem Ressourcenredundanz oder Zeitredundanz zur Anwendung bringen, d.h., für jede logische vom Kommunikationskontroller **121** zur Verfügung gestellte Nachricht können gleichzeitig auf mehreren parallelen Kommunikationskanälen oder nacheinander auf einem Kommunikationskanal mehrere redundante Instanzen der logischen Nachricht, d.h. mehrere physische Nachrichten, gesendet werden. Es wird angenommen, dass dem Empfänger *a priori* der Sendezeitpunkt einer Nachricht bekannt gegeben wurde. Diese Bekanntgabe der Sendezeitpunkte kann unmittelbar vor dem Senden dynamisch erfolgen. Der Empfänger kann sich aus dem Sendezeitpunkt und der bekannten Transportlatenz des Kommunikationssystems ausrechnen, wann die letzte korrekte Instanz einer Nachricht spätestens beim Empfänger einzutreffen hat. Wenn bis zu diesem Zeitpunkt keine Nachricht vom Sender eingetroffen ist, dann ist aus der Sicht des Empfängers ein Fehler aufgetreten (Nachrichtenverlust).

Die fünf Knotenrechner **111**, **112**, **113**, **114**, und **115** können mehrere Funktionen ausführen und daher unterschiedlichen Funktionsgruppen angehören. In einem konkreten Anwendungsfall können die vier Knotenrechner **111**, **112**, **113**, und **115** die Funktionsgruppe-1 bilden, und die vier Knotenrechner **111**, **112**, **113**, und **114** die Funktionsgruppe-2 bilden. Zum Beispiel realisiert in einer bestimmten Anwendung Funktionsgruppe-1 die Funktion einer verteilten Uhrensynchronisation [6] und Funktionsgruppe-2 die Funktion eines verteilten Bremssystems mit vier Bremsen in einem Kraftfahrzeug, die von den Knotenrechner **111**, **112**, **113**, und **114** gesteuert werden. Wenn ein Fehler in einer Bremse auftritt, so ist die im Rechnerknoten parallel ablaufende Funktion der Uhrensynchronisation von diesem Fehler nicht unmittelbar betroffen. Entsprechend der vorliegenden Erfindung wird jeder dieser Funktionsgruppen ein eigenes Funktionsgruppenmitgliedfeld (Fig. 2) zugeordnet, dessen Bitlänge mindesten der Anzahl der Knotenrechner in der Funktionsgruppe entspricht und wo jedem in einer Funktionsgruppe beteiligten Knotenrechner ein

bestimmtes Bit, z.B. **201, 202, 203, 205**, des Funktionsgruppenmitgliedfeldes eindeutig zugeordnet ist. In diesem Beispiel bedeutet z.B. der Wert WAHR des Bits **201** dass die Funktion-1 im Knotenrechner **111** korrekt ausgeführt wird und der Wert FALSCH dass die Funktion-1 im Knotenrechner **101** nicht korrekt ausgeführt wird. Jeder Knotenrechner **111** verwaltet eine lokale Version des Funktionsgruppenmitgliedfeldes jeder Funktionsgruppe, an der er beteiligt ist.

Es wird davon ausgegangen, dass die Nachrichten periodisch in Runden gesendet werden und dass die zeitliche Sendeordnung der Nachrichten innerhalb einer Funktionsgruppe *a priori* festgelegt wird. Diese Festlegung kann über eine Konfigurationsnachricht (Fig. 3) erfolgen, die angibt, welche Periodizität **301** eine Funktionsgruppe hat und wann **312** jeder Knoten **310** welche Nachrichten **311** zu senden hat. Innerhalb einer Funktionsgruppe sind alle Nachrichten zeitlich geordnet. Da ein Echtzeitkommunikationssystem eine bekannte Transportlatenz haben muss, ist somit auch festgelegt, wann eine Nachricht beim Empfänger einzutreffen hat. Es ist vorteilhaft, die Konfigurationsnachrichten zu verschlüsseln, um zu verhindern, dass ein fehlerhafter oder eine böswilliger Rechnerknoten, der dazu nicht berechtigt ist, eine falsche Konfigurationsnachricht an richtige Rechnerknoten verteilen kann.

Die Rundenperiode **301** kann direkt von der global zur Verfügung stehenden Zeit abgeleitet werden. Fig. 4 zeigt ein von der Object Management Group (OMG) standardisierte Zeitdarstellung [5], die aus 8 Byte besteht. Diese Zeitdarstellung **401** verwendet das Binärformat, wobei die volle physikalische Sekunde **402** den zentralen Bezugswert darstellt. Mit diesem Zeitformat **401** lässt sich jeder Zeitpunkt innerhalb der nächsten zehntausend Jahre mit einer Genauigkeit von ca. 60 Nanosekunden darstellen. Jedem Bit dieses Zeitformats kann eine Periodizität zugeordnet werden, die sich aus der Stellung des Bits im Zeitformat ergibt. Wenn die Periodizität einer Runde über ein so ausgewähltes Periodizitätsbit **403** spezifiziert wird, so stehen alle Perioden in allen Subsystemen zueinander in einem harmonischen Verhältnis. Im Beispiel der Fig. 4 ist das Periodizitätsbit **403** das 6. Bit rechts von dem vollen Sekundenbit **402**. Das Periodizitätsbit **403** bestimmt somit eine Rundenfrequenz von $1/2^6$ Sekunden, das entpricht 64 Hz fest. Der Periodenbeginn ist definiert als der Zeitpunkt, zu dem alle Bits rechts vom Periodizitätsbit **403** im Zeitformat **401** nicht gesetzt sind. Der Sendezeitpunkt eines Knotenrechners kann durch die Angabe des Offsets **410** vom Periodenbeginn präzise spezifiziert werden. Wenn die Periodizität jeder Runde eine Zweierpotenz der kleinsten Runde entspricht, dann ist es möglich große synchrone Systeme, die über eine Vielzahl von Clustern verfügen, einfach aufzubauen. In den unterschiedlichen Clustern können auch unterschiedliche Kommunikationsprotokolle eingesetzt werden.

Unmittelbar vor dem Senden einer Nachricht setzt der sendende Kontenrechner (der Sender) **111** das ihm zugeordnete Bit in seinem lokalen Funktionsgruppenmitgliedfeld **201** auf WAHR wenn der Sender seine Funktionsfähigkeit in dieser Funktionsgruppe als gegeben ansieht. Vor dem Senden einer Nachricht setzt der Sender **111** das ihm zugeordnete Bit in seinem lokalen Funktionsgruppenmitgliedfeld **201** auf FALSCH wenn der Sender seine Funktionsfähigkeit in dieser Funktionsgruppe als nicht gegeben ansieht. Im vorangegangenen Beispiel mit der defekten Bremse wird der Sender sein Bit im Funktionsgruppenmitgliedfeld-1, der Uhrensynchronisation auf WAHR und sein Bit im Funktionsgruppenmitgliedfeld-2, dem Bremssystem, auf FALSCH setzen wenn die lokale Bremse fehlerhaft ist.

Unmittelbar vor dem Senden einer Nachricht muss der Sender alle seine lokalen Funktionsgruppenmitgliedsfelder in die zu sendende Nachricht kopieren und mit der Nachricht an die Empfänger senden.

Der Empfänger einer Nachricht setzt das dem Sender zugeordnete Bit in jedem seiner lokalen Funktionsgruppenmitgliedfelder unmittelbar nach dem erwarteten Empfangszeitpunkt einer Nachricht von einem Sender wie folgt:

(1) Wenn bis zu diesem Zeitpunkt keine der redundanten Nachrichten des Senders syntaktisch korrekt eingetroffen ist, dann setzt der Empfänger den Wert des dem Sender zugeordneten Bits in all seinen betroffenen lokalen Funktionsgruppenmitgliedsfeldern auf FALSCH. Eine Nachricht ist syntaktisch korrekt, wenn die CRC Überprüfung der Nachricht keinen Fehler anzeigt.

(2) Wenn das in der empfangenen Nachricht dem Nachrichtensender zugeordnete Bit in einem Funktionsgruppenmitgliedfeld auf FALSCH gesetzt ist, dann setzt der Empfänger das entsprechende Bit in seinem entsprechenden lokalen Funktionsgruppenmitgliedsfeld auf FALSCH.

(3) Wenn die Summe des Sendezeitpunktes der Nachricht und der bekannten Transportlatenz von dem vom Empfänger beobachteten Empfangszeitpunkt signifikant abweicht, dann setzt der Empfänger den Wert des dem Sender zugeordneten Bits in all seinen lokalen Funktionsgruppenmitgliedsfeldern auf FALSCH. Der beobachtete Empfangszeitpunkt weicht vom erwarteten Empfangszeitpunkt signifikant ab, wenn die Differenz zwischen diesen beiden Zeitpunkten, die in den aktiven Intervallen einer *sparse timebase* (siehe[6], p. 56) liegen müssen, größer ist als die Dauer eines aktiven Zeitintervalls der *sparse timebase*.

(4) In allen anderen Fällen setzt der Empfänger die entsprechenden Bits in seinen lokalen Funktionsgruppenmitgliedfeldern auf WAHR.

Jedes Mitglied einer Funktionsgruppe kann eine funktionsgruppenbezogene Konsistenzvariable verwalten, die unmittelbar nach dem Senden auf den Wert WAHR gesetzt wird und die auf den Wert FALSCH gesetzt wird, sobald das in einer empfangenen Nachricht enthaltenen Funktionsgruppenmitgliedsfeld eines Nachfolgers des Senders vom Inhalt des entsprechenden lokalen Funktionsgruppenmitgliedsfeldes abweicht. Durch diese Regel wird eine Inkonsistenz im System entdeckt, die innerhalb der letzten Runde beobachtet werden konnte.

Jedes Mitglied einer Funktionsgruppe kann eine funktionsgruppenbezogene Quittungsvariable verwalten, die unmittelbar nach dem Senden auf FALSCH gesetzt wird und die auf den Wert WAHR gesetzt wird, sobald das in einer empfangenen Nachricht enthaltene auf den Sender bezogene Bit des Funktionsgruppenmitgliedsfeldes in einer empfangenen Nachricht eines Nachfolgers des Senders auf WAHR gesetzt ist. Durch diese Regel wird überprüft ob irgendein Empfänger die Nachricht des Senders erhalten hat.

Wenn der Kommunikationskontroller **121** des Senders einer Nachricht das dem Sender zugeordnete Bit in allen lokalen Funktionsgruppenmitgliedfeldern des Senders unmittelbar vor dem Senden auf FALSCH setzt, im Falle dass innerhalb der letzten Kommunikationsrunde das dem Sender zugeordnete Bit des

Funktionsgruppenmitgliedfeldes vom Hostcomputer **131** nicht beschrieben wurde, dann wird der Hostcomputer **131** gezwungen, in jeder Runde in jedem Funktionsgruppenmitgliedfeld das dem Sender zugeordnete Bit erneut auf WAHR zu setzten, sofern die betreffende Funktion noch immer erfüllt wird. Durch diese Regel wird verhindert, dass *ein fail-silent* Ausfall einer Funktion des Hostrechners **131** von den Kommunikationspartnern innerhalb einer Funktionsgruppe nicht beobachtet werden kann.

Wenn in einem Anwendungsfall gewünscht wird zu unterscheiden, ob ein Fehler in dem Kommunikationssystem **109** oder in einem Hostcomputer **131** aufgetreten ist, so können zwei verschiedene Funktionsgruppen mit entsprechenden Funktionsgruppenmitgliedfeldern eingeführt werden. Die eine Funktionsgruppe überwacht die Funktion der Kommunikation, während die andere Funktionsgruppe die Funktion der Anwendung im Host überwacht.

Um den Host-Rechner zu entlasten und die Verarbeitung zu beschleunigen kann das offengelegte Verfahren zur konsistenten Fehlererkennung in einem verteilten Computersystem auch in der Hardware des Kommunikationskontrollers **121** verwirklicht werden. In diesem Fall kann es zweckmäßig sein, den Hostcomputer **131** über eine Interruptleitung **151** zu signalisieren (Fehlerinterrupt), wenn ein Bit eines lokalen Funktionsgruppenmitgliedfeldes des Kommunikationskontrollers **121** vom Zustand WAHR in den Zustand FALSCH wechselt. Wenn kein Fehlerinterrupt auftritt kann der Hostcomputer **131** annehmen, dass die Kommunikation innerhalb der Funktionsgruppen des Hostcomputers fehlerfrei funktioniert.

Der Kommunikationskontroller **121** kann den Hostprozessor **131** über eine Interruptleitung **151** auch signalisieren, wenn in der vergangenen Runde keiner der korrekten Empfänger die letzte gesendete Nachricht dieses Knotenrechners quittiert hat oder wenn in der letzten Runde eine Inkonsistenz beobachtet wurde.

PATENTANSPRÜCHE

(1) Verfahren zur konsistenten Erkennung von Fehlern in einem verteilten Echtzeitcomputersystem, in dem eine Vielzahl von Knotenrechnern, z.B. die fünf Knotenrechner **111, 112, 113, 114,** und **115** die über mindestens je einen Kommunikationskontroller **121, 122, 123, 124,** und **125** und über je einen Hostcomputer **131, 132, 133, 134,** und **135**, verfügen, und die über ein rundenbasiertes Echtzeitkommunikationssystem **109** bestehend aus einem oder mehreren Kommunikationskanälen Nachrichten austauschen, wobei für jede logische Nachricht eine oder mehrere redundante physische Nachrichten transportiert werden können, und wo alle Knotenrechner über einen Zugriff auf eine global synchronisierte Zeit verfügen **dadurch gekennzeichnet, dass** die Knotenrechner Funktionsgruppen bilden, wobei ein Knotenrechner Mitglied mehrerer Funktionsgruppen sein kann, und wo jeder Funktionsgruppe ein dediziertes Funktionsgruppenmitgliedfeld (Fig. 2) zugeordnet ist dessen Bitlänge mindesten der Anzahl der Knotenrechner in dieser Funktionsgruppe entspricht und wo jedem in einer Funktionsgruppe beteiligten Knotenrechner **111** ein spezifiziertes Bit **201** des Funktionsgruppenmitgliedfeldes eindeutig zugeordnet ist, und wo jeder Knotenrechner eine lokale Version des Funktionsgruppenmitgliedfeldes jeder Funktionsgruppe, an der er beteiligt ist, verwaltet, und wo vor dem Senden einer Nachricht der sendende Knoten das ihm zugeordnete Bit in seinem lokalen Funktionsgruppenmitgliedfeld auf WAHR setzt wenn der sendende Knoten seine Funktionsfähigkeit in dieser Funktionsgruppe als gegeben ansieht und wo vor dem Senden einer Nachricht der sendende Knoten das ihm zugeordnete Bit in seinem lokalen Funktionsgruppenmitgliedfeld auf FALSCH setzt wenn der sendende Knoten seine Funktionsfähigkeit in dieser Funktionsgruppe als nicht gegeben ansieht und wo der sendende Knoten alle seine lokalen Funktionsgruppenmitgliedsfelder in eine zu sendende Nachricht kopiert, um sie an die Empfänger zu übertragen.

(2) Verfahren nach Anspruch 1 **dadurch gekennzeichnet, dass** der Empfänger einer Nachricht das einem sendenden Knoten zugeordnete Bit in jedem seiner lokalen Funktionsgruppenmitgliedfelder unmittelbar nach dem erwarteten Empfangszeitpunkt einer Nachricht wie folgt bestimmt: (1) Wenn keine der redundanten Nachrichten des sendenden Knotens syntaktisch korrekt eingetroffen ist, dann setzt der Empfänger die entsprechenden Werte der dem sendenden Knoten zugeordneten Bits in allen seinen lokalen Funktionsgruppenmitgliedsfeldern auf FALSCH, (2) Wenn das in der empfangenen Nachricht den Nachrichtensender betreffende Bit eines Funktionsgruppenmitgliedfelds auf FALSCH gesetzt ist, dann setzt der Empfänger das entsprechende Bit in seinem entsprechenden lokalen Funktionsgruppenmitgliedfeld auch auf FALSCH, (3) Wenn die Summe des Sendezeitpunktes der Nachricht und der bekannten Transportlatenz von dem vom Empfänger beobachteten Empfangszeitpunkt signifikant abweicht, dann setzt der Empfänger die dem Sender zugeordneten Bits in allen seinen entsprechenden lokalen Funktionsgruppenmitgliedsfeldern auf FALSCH. In allen anderen Fällen setzt der Empfänger die dem Sender entsprechenden Bits in allen seinen lokalen Funktionsgruppenmitgliedfeldern auf WAHR.

(3) Verfahren nach einem oder mehreren der Ansprüche 1 bis 2 **dadurch gekennzeichnet, dass** jedes Mitglied einer Funktionsgruppe eine

funktionsgruppenbezogene Konsistenzvariable verwaltet, die unmittelbar nach dem Senden auf WAHR gesetzt wird und die auf den Wert FALSCH gesetzt wird, sobald das in einer empfangenen Nachricht enthaltenen Funktionsgruppenmitgliedsfeld eines Nachfolgers des sendenden Knotens vom Inhalt des entsprechenden lokalen Funktionsgruppenmitgliedsfeldes abweicht.

(4) Verfahren nach einem oder mehreren der Ansprüche 1 bis 3 **dadurch gekennzeichnet, dass** jedes Mitglied einer Funktionsgruppe eine funktionsgruppenbezogene Quittungsvariable verwaltet, die unmittelbar nach dem Senden auf den Wert FALSCH gesetzt wird und die auf den Wert WAHR gesetzt wird, sobald das in einer empfangenen Nachricht enthaltene auf den ursprünglich sendenden Knoten bezogene Bit des Funktionsgruppenmitgliedsfeldes in einer empfangenen Nachricht eines Nachfolgers des ursprünglich sendenden Knotens auf WAHR gesetzt ist.

(5) Verfahren nach einem oder mehreren der Ansprüche 1 bis 4 **dadurch gekennzeichnet, dass** den an einer Funktionsgruppe beteiligten Knotenrechnern über eine oder mehrere Konfigurationsnachrichten bekannt gegeben wird, welche Knotenrechner eine Funktionsgruppe bilden, und wann welche Nachrichten von welchen Knotenrechnern zu erwarten sind.

(6) Verfahren nach einem oder mehreren der Ansprüche 1 bis 5 **dadurch gekennzeichnet, dass** die Konfigurationsnachrichten verschlüsselt sind.

(7) Verfahren nach einem oder mehreren der Ansprüche 1 bis 6 **dadurch gekennzeichnet, dass** die Periodizität der Übertragung innerhalb einer Funktionsgruppe durch die Angabe eines Periodizitätsbit **403** in dem in **401** vorgestellten Zeitformat spezifiziert wird und der Periodenbeginn jeweils zum Zeitpunkt stattfindet, zu dem alle Bits des Zeitwertes rechts vom Periodizitätsbit **403** nicht gesetzt sind.

(8) Verfahren nach einem oder mehreren der Ansprüche 1 bis 7 **dadurch gekennzeichnet, dass** der periodische Sendezeitpunkt einer Nachricht durch die Angabe des Offsets **410** zum Periodenbeginn bestimmt wird.

(9) Verfahren nach einem oder mehreren der Ansprüche 1 bis 8 **dadurch gekennzeichnet, dass** der Kommunikationskontroller **121** des sendenden Knotens einer Nachricht das ihm zugeordnete Bit in allen lokalen Funktionsgruppenmitgliedfeldern des sendende Knotens unmittelbar vor dem Senden auf FALSCH setzt, wenn innerhalb der letzten Kommunikationsrunde das dem sendenden Knoten zugeordnete Bit des Funktionsgruppenmitgliedfeldes vom Host-Computer nicht beschrieben wurde.

(10) Kommunikationskontroller **121** eines Rechnerknotens eines verteilten Computersystems **dadurch gekennzeichnet, dass** einer oder mehrere der Verfahrensansprüche 1 bis 9 in der Hardware des Kommunikationskontrollers **121** implementiert werden.

(11) Kommunikationskontroller **121** eines Rechnerknotens eines verteilten Computersystems nach Anspruch 10 **dadurch gekennzeichnet, dass** der Kommunikationskontroller **121** dem Hostcomputer **131** über eine Interruptleitung **151** signalisiert, wenn ein Bit eines lokalen Funktionsgruppenmitgliedfelds in den Zustand FALSCH wechselt.

(12) Kommunikationskontroller **121** eines Rechnerknotens eines verteilten Computersystems nach Anspruch 10 **dadurch gekennzeichnet, dass** der Kommunikationskontroller **121** dem Hostcomputer **131** über eine Interruptleitung **151** unmittelbar vor dem Senden einer Nachricht signalisiert, dass die Quittungsvariable entsprechend Anspruch (5) den Wert FALSCH hat.

(13) Kommunikationskontroller **121** eines Rechnerknotens eines verteilten Computersystems nach Anspruch 10 **dadurch gekennzeichnet, dass** der Kommunikationskontroller **121** dem Hostcomputer **131** über eine Interruptleitung **151** unmittelbar vor dem Senden einer Nachricht signalisiert, dass die Konsistenzvariable entsprechen Anspruch (4) den Wert FALSCH hat.

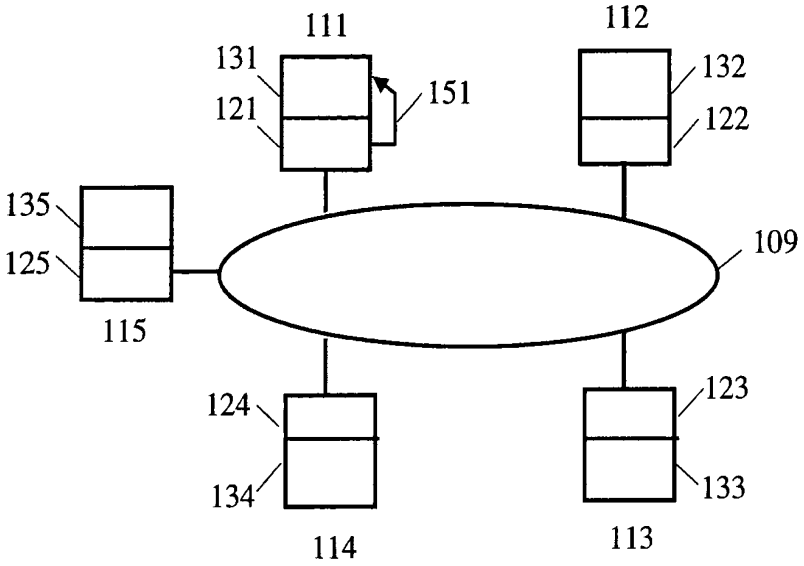


FIG. 1

Group-1 Membership Bit for Node 111	201
Group-1 Membership Bit for Node 112	202
Group-1 Membership Bit for Node 113	203
Group-1 Membership Bit for Node 115	205

FIG. 2

Specification of Period Bit	301
Name of Sender 1	310
Name of Message 1	311
Offset of Message 1	312
Name of Sender 2	
Name of Message 2	
Offset of Message 2	
Name of Sender 3	
Name of Message 3	
Offset of Message 3	
Name of Sender 4	
Name of Message 4	
Offset of Message 4	

FIG. 3

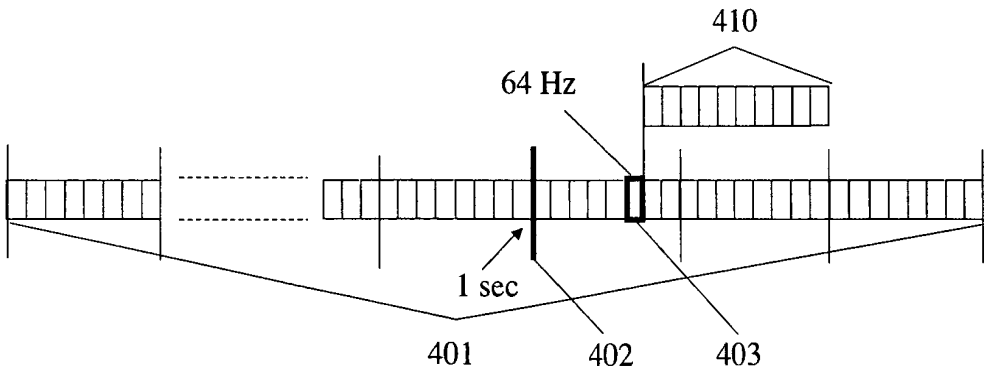


FIG. 4