(19) Deutsches Patent- und Markenamt

(10) **DE 11 2013 005 688 T5** 2015.08.06

(12) **Veröffentlichung**

der internationalen Anmeldung mit der
(87) Veröffentlichungs-Nr.: **WO 2014/085717**
in deutscher Übersetzung (Art. III § 8 Abs. 2 IntPatÜG)
(21) Deutsches Aktenzeichen: **11 2013 005 688.7**
(86) PCT-Aktenzeichen: **PCT/US2013/072423**
(86) PCT-Anmeldetag: **27.11.2013**
(87) PCT-Veröffentlichungstag: **05.06.2014**
(43) Veröffentlichungstag der PCT Anmeldung
in deutscher Übersetzung: **06.08.2015**

(51) Int Cl.: ***G06F 15/76*** (2006.01)

(30) Unionspriorität:

US-61/730,939	**28.11.2012**	**US**
US-61/730,940	**28.11.2012**	**US**
US-61/749,224	**04.01.2013**	**US**
US-61/749,231	**04.01.2013**	**US**
US-61/874,078	**05.09.2013**	**US**
US-61/874,056	**05.09.2013**	**US**

(71) Anmelder:
NVIDIA Corporation, Santa Clara, Calif., US

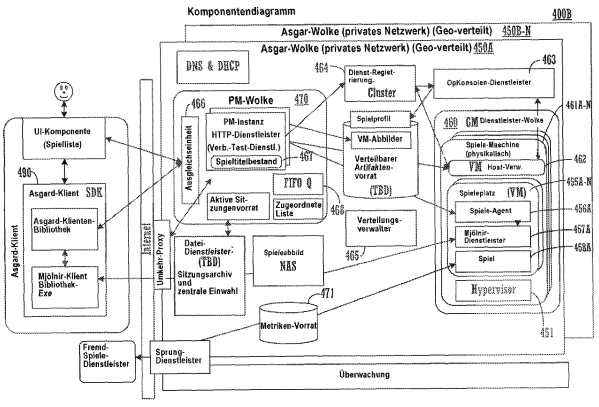
(74) Vertreter:
Dilg Haeusler Schindelmann Patentanwaltsgesellschaft mbH, 80636 München, DE

(72) Erfinder:
Diard, Franck, Mountain View, Calif., US; Huang, Jen-Hsun, Los Altos Hills, Calif., US; Apte, Atul, Cupertino, Calif., US; Putnam, Tom, Austin, Tex., US; Ahuja, Alok, San Jose, Calif., US; Lavoie, Matt, Raleigh, N.C., US; Harms, Dennis, 66953 Pirmasens, DE; Goeringer, Tyler, Sunnyvale, Calif., US; Wang, Yao-Tian, Emerald Hills, Calif., US; Matloff, Scott, Dublin, Calif., US; Holmes, John, Chris, Sunnyvale, Calif., US; Krishnamoorthy, Subu, Sunnyvale, Calif., US; Eckart, Stefan, 81669 München, DE; Vukojevic, Bojan, Dublin, Calif., US; Wang, Xun, San Jose, Calif., US

Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen

(54) Bezeichnung: **Verfahren und System für eine wolkenbasierte virtualisierte Graphikverarbeitung für Fernanzeigen**

(57) Zusammenfassung: Eine Vorrichtung zur Bereitstellung einer Grafikverarbeitung. Die Vorrichtung umfasst eine duale CPU-Sockel-Architektur mit einem ersten CPU-Sockel und einem zweiten CPU-Sockel. Die Vorrichtung umfasst mehrere GPU-Leiterplatten, die mehrere GPU-Prozessoren bereitstellen, die mit dem ersten CPU-Sockel und dem zweiten CPU-Sockel verbunden sind, wobei jede GPU-Leiterplatte zwei oder mehr der mehreren GPU-Prozessoren aufweist. Die Vorrichtung umfasst eine Kommunikationsschnittstelle, die den ersten CPU-Sockel mit einer ersten Teilgruppe aus einer oder mehreren GPU-Leiterplatten und den zweiten CPU-Sockel mit einer zweiten Teilgruppe aus einer oder mehreren GPU-Leiterplatten verbindet.



Beschreibung

QUERVERWEIS AUF VERWANDTE ANMELDUNGEN

[0001] Die vorliegende Anmeldung beansprucht die Priorität der vorläufigen US-Patentanmeldung mit der Seriennummer 61/730940 der gleichen Anmelderin mit dem Titel „WOLKENBASIERTE VIRTUALISIERTE GRAFIKVERARBEITUNG FÜR FERNANZEIGEN” mit Einreichungsdatum vom 28. November 2012 mit dem Anwaltsaktenzeichen NVID-P-SC-12-0727-US0, die hierin in ihrer Gesamtheit durch Bezugnahme mit eingeschlossen ist.

[0002] Die vorliegende Anmeldung beansprucht die Priorität der vorläufigen US-Patentanmeldung, die die gleiche Anmelderin wie die vorliegende Anmeldung hat, mit der Seriennummer 61/730939 mit dem Titel „WOLKENBASIERTE VIRTUALISIERTE GRAFIKVERARBEITUNG FÜR FERNANZEIGEN” mit Einreichungsdatum vom 28. November 2012 mit dem Anwaltsaktenzeichen NVID-P-SC-12-0728-US0, die hierin in ihrer Gesamtheit durch Bezugnahme mit eingeschlossen ist.

[0003] Die vorliegende Anmeldung beansprucht die Priorität der vorläufigen US-Patentanmeldung, die die gleiche Anmelderin wie die vorliegende Anmeldung hat, mit der Seriennummer 61/749224 mit dem Titel „AN NETZWERK ANGEBUNDENES GPU-GERÄT” mit dem Einreichungsdatum vom 4. Januar 2013 mit dem Anwaltsaktenzeichen NVID-P-SC-12-0814-US0, die hierin in ihrer Gesamtheit durch Bezugnahme mit eingeschlossen ist.

[0004] Die vorliegende Anmeldung beansprucht die Priorität der vorläufigen US-Patentanmeldung, die die gleiche Anmelderin wie die vorliegende Anmeldung hat, mit der Seriennummer 61/874056 mit dem Titel „THOR-SYSTEMARCHITEKTUR” mit Einreichungsdatum vom 5. September 2013 mit dem Anwaltsaktenzeichen NVID-P-SC-12-0470-US0, die hierin durch Bezugnahme in ihrer Gesamtheit mit eingeschlossen ist.

[0005] Die vorliegende Anmeldung beansprucht die Priorität der vorläufigen US-Patentanmeldung, die die gleiche Anmelderin wie die vorliegende Anmeldung hat, mit der Seriennummer 61/874078 mit dem Titel „THOR-SYSTEMARCHITEKTUR” mit Einreichungsdatum vom 5. September 2013 mit dem Anwaltsaktenzeichen NVID-P-SC-12-0814-USX, die hiermit in ihrer Gesamtheit durch Bezugnahme mit eingeschlossen ist.

[0006] Die vorliegende Anmeldung beansprucht die Priorität der vorläufigen US-Patentanmeldung, die die gleiche Anmelderin wie die vorliegende Anmeldung hat, mit der Seriennummer 61/874078 mit dem Titel „AN NETZWERK ANGEBUNDENES GPU-GERÄT” mit dem Einreichungsdatum vom 5. September 2013 mit dem Anwaltsaktenzeichen NVID-P-SC-12-0814-USX, die hierin in ihrer Gesamtheit durch Bezugnahme mit eingeschlossen ist.

[0007] Die vorliegende Anmeldung ist verwandt mit der ebenfalls anhängigen US-Patentanmeldung mit der Seriennummer 13/727357, „VIRTUALISIERTE GRAFIKVERARBEITUNG FÜR EINE FERNANZEIGE”, die am 26. Dezember 2012 mit dem Anwaltsaktenzeichen NVID-P-SC-09-0210-US1 eingereicht wurde und hiermit durch Bezugnahme für alle Zwecke mit eingeschlossen ist.

HINTERGRUND

[0008] In der Vergangenheit wurde eine Anwendung, etwa ein Videospiel, unter Anwendung eines Personalcomputers (PC) oder unter Anwendung einer Konsole, die mit einem Fernsehgerät verbunden war, ausgeführt (gespielt). Ein Anwender kaufte oder mietete ein Spiel, das auf dem PC geladen oder in die Spielekonsole eingefügt wurde und spielte dann in bekannter Weise.

[0009] In der jüngeren Vergangenheit ist das Online-Spielen bzw. interaktive Spiele populär geworden. Ein Online-Spiel wird über ein Netzwerk, etwa das Internet, gespielt. Das Spiel wird auf ein Gerät eines Anwenders herunter geladen, während weitere Software, die zum Spielen des Spieles benötigt wird, auf einem Server bzw. Dienstleistungsrechner bzw. Dienstleister liegt, auf den über das Netzwerk zugegriffen wird. Das Online-Spielen erlaubt es mehreren Anwendern, gegeneinander in der Spieleumgebung anzutreten, die durch die Software auf dem Dienstleister bereitgestellt wird.

[0010] Ferner wurde das mobile Spielen zunehmend populär. Beispielsweise kann ein Mobilgerät (beispielsweise Telefon) ein Videospiel für einen Anwender bereitstellen, das über beispielsweise die berührungsempfindlichen Steuerelemente des Mobiltelefons gesteuert werden kann. Diese Steuerelemente werden virtuell

erzeugt und auf dem berührungsempfindlichen Bildschirm angezeigt. Da derartige Mobiltelefone nicht speziell für das Spielen hergestellt werden, ist die Rechenleistung derartiger Mobiltelefone häufig für viele Spiele zu gering. Ein weiteres Problem bei Mobiltelefonen besteht darin, dass sie häufig nicht in der Lage sind, gewisse Spiele zu unterstützen bzw. zu ermöglichen, da derartige Spiele die Ausführung einer gewissen Betriebssystemsumgebung erfordern. Ferner nehmen die virtuellen Bedienknöpfe wertvollen Platz auf dem Bildschirm ein, wodurch die gesamte Anzeige des Spieles für den Anwender verringert wird.

[0011] Ferner stellen virtuelle Steuerungsknöpfe, die durch den berührungsempfindlichen Bildschirm der Anzeige simuliert werden, eine minderwertige Schnittstelle zwischen dem Anwender und dem Spiel bereit. Es ist schwierig, eine Interaktion auf Grundlage von Berührung mit einem virtuellen Knopf zu erreichen, da der Knopf auf einem flachen Bildschirm virtualisiert bzw. virtuell dargestellt wird. Ohne eine Berührungsreferenz besteht die einzige Möglichkeit sicherzustellen, dass der virtuelle Knopf betätigt wurde, darin, physikalisch den Finger und den virtuellen Knopf gleichzeitig zu beobachten. Dies kann die Beobachtung des Bildschirms durch den Spieler während einer entscheidenden Phase in einem Spiel verringern. Ferner sind die Knöpfe auf die vordere Oberfläche des Mobilgeräts beschränkt. Da die Knöpfe virtuell erzeugt werden, können diese Knöpfe lediglich auf der Anzeige mit berührungsempfindlichem Bildschirm dargestellt werden. Die konkurrierende Anforderung nach Platzbedarf auf dem Bildschirm kann zu einer Verringerung der Anzahl an Knöpfen führen oder kann dazu führen, dass diese als Bild so klein erzeugt werden, dass sie nur schwer nutzbar sind.

ÜBERBLICK

[0012] In Ausführungsformen der vorliegenden Erfindung ist eine Vorrichtung zur Bereitstellung einer Grafikverarbeitung beschrieben. Die Vorrichtung umfasst eine duale bzw. Doppel-Sockel-Architektur für eine zentrale Recheneinheit (CPU) mit einem ersten CPU-Sockel und einem zweiten CPU-Sockel. Die Vorrichtung umfasst mehrere Leiterplatten mit grafischer Verarbeitungseinheit (GPU), die mehrere GPU-Prozessoren bereitstellen, die mit dem ersten CPU-Sockel und dem zweiten CPU-Sockel verbunden sind, wobei jede GPU-Leiterplatte zwei oder mehr der mehreren GPU-Prozessoren aufweist. Die Vorrichtung umfasst eine Kommunikationsschnittstelle, die den ersten CPU-Sockel mit einer ersten Teilgruppe einer oder mehrerer GPU-Leiterplatten und den zweiten CPU-Sockel mit einer zweiten Teilgruppe einer oder mehreren GPU-Leiterplatten verbindet. In einer weiteren Ausführungsform ist eine mit einem Netzwerk verbundene GPU-Einrichtung beschrieben. Die mit Netzwerk verbundene GPU-Einrichtung umfasst mehrere Verarbeitungsleiterplatten, die mehrere virtuelle CPU-und GPU-Prozessoren bereitstellen. Jede der Verarbeitungsleiterplatten umfasst eine duale CPU-Sockel-Architektur mit einem ersten CPU-Sockel und einem zweiten CPU-Sockel. Jede Verarbeitungsleiterplatte umfasst mehrere GPU-Leiterplatten, die mehrere GPU-Prozessoren bereitstellen, die mit dem ersten CPU-Sockel und dem zweiten CPU-Sockel verbunden sind, wobei jede GPU-Leiterplatte zwei oder mehr der mehreren GPU-Prozessoren aufweist. Jede Verarbeitungsleiterplatte umfasst mehrere erste Kommunikationsbrücken, wovon jede eine entsprechende der GPU-Leiterplatten mit dem CPU-Sockel und dem zweiten CPU-Sockel verbindet. Jede Verarbeitungsleiterplatte umfasst eine Kommunikationsschnittstelle, die den ersten CPU-Sockel mit einer ersten Teilgruppe aus einer oder mehreren GPU-Leiterplatten und den zweiten CPU-Sockel mit einer zweiten Teilgruppe aus einer oder mehreren GPU-Leiterplatten verbindet.

[0013] In Ausführungsformen der vorliegenden Erfindung ist ein computerimplementiertes Verfahren zur Umschaltung von Video-Strömen, die an eine entfernte Anzeige bzw. Fernanzeige geleitet werden, offenbart. In anderen Ausführungsformen ist ein nicht-flüchtiges computerlesbares Medium offenbart mit von einem Computer ausführbaren Befehlen, die ein Computersystem veranlassen, ein Verfahren zum Umschalten von Video-Strömen auszuführen, die einer Fernanzeige zugeleitet werden. In noch anderen Ausführungsformen ist ein Computersystem offenbart, das einen Prozessor und einen mit dem Prozessor verbundenen Speicher aufweist, der darin gespeichert Befehle enthält, die, wenn sie von dem Computersystem ausgeführt werden, das Computersystem veranlassen, ein Verfahren zur Umschaltung von Video-Strömen auszuführen, die einer Fernanzeige zugeleitet werden. Das Verfahren umfasst eine Initialisierung einer Instanz einer Anwendung. Das Verfahren umfasst ferner die Ausführung einer Grafikerzeugung in mehreren Blöcke bzw. Bildblöcken, um einen ersten Video-Strom durch Ausführung der Anwendung zu erzeugen, wobei der erste Video-Strom die mehrere Bildblöcke enthält. Das Verfahren umfasst die sequenzielle Ladung der mehreren Bildblöcke in einen oder mehrere Blockpuffer. Das Verfahren umfasst die Ermittelung, wann eine erste Bitmap bzw. Bit-Zuordnung eines Bildblocks in einen entsprechenden Blockpuffer geladen wird und mit einer Anwendungssignatur übereinstimmt, die ein Abkömmling einer übergeordneten Bitmap bzw. Bit-Zuordnung ist, die zu einem Schlüsselbildblock des ersten Video-Stroms gehört.

[0014] In einer weiteren Ausführungsform ist ein System zur Umschaltung von Video-Strömen offenbart, die einer Fernanzeige zugeleitet werden. Das System umfasst einen Prozessor für die Initialisierung einer Instanz

einer Anwendung. Das System umfasst eine Grafikerzeugungseinheit zum Ausführen einer Grafikerzeugung in mehreren Bildblöcken, um einen ersten Video-Strom durch Ausführung der Anwendung zu erzeugen, wobei der erste Video-Strom die mehreren Bildblöcke enthält. Das System umfasst einen Blockpuffer für den Empfang einer Sequenz mehrerer Bildblöcke, die zu dem ersten Video-Strom gehören. Das System umfasst einen Komparator, der ausgebildet ist zu ermitteln, wann eine erste Bitmap bzw. Bit-Zuordnung eines Bildblocks in einen entsprechenden Blockpuffer geladen wird und mit einer Anwendungssignatur übereinstimmt, die ein Abkömmling einer übergeordneten Bitmap bzw. Bit-Zuordnung ist, die zu einem Schlüsselbildblock des ersten Video-Stroms gehört.

[0015] In Ausführungsformen der vorliegenden Erfindung ist ein computerimplementiertes Verfahren für die Netzwerk-Wolke-Ressourcenerzeugung bzw. Netzwerk-Cloud-Ressourcenerzeugung offenbart. Das Verfahren umfasst die Erzeugung einer Schablone einer virtuellen Maschine in einem wolkenbasierten bzw. Cloud basierten System. Das Verfahren umfasst die Erzeugung einer Instanz einer virtuellen Maschine für einen Endanwender durch Klonen der Schablone. Das Verfahren umfasst das Laden einer Anwendung, die von der virtuellen Maschine ausgeführt wird. Das Verfahren umfasst den Zugriff auf eine erste Information, die mit dem Endanwender assoziiert bzw. in Beziehung steht. Das Verfahren umfasst das Laden der ersten Information in eine Instanz der Anwendung.

[0016] In Ausführungsformen der vorliegenden Erfindung ist ein computerimplementiertes Verfahren zur Zuweisung beschrieben. In anderen Ausführungsformen ist ein nicht-flüchtiges computerlesbares Medium offenbart mit von einem Computer ausführbaren Befehlen, die ein Computersystem veranlassen, ein Verfahren zur Zuweisung auszuführen. Es ist ferner ein noch anderes Computersystem offenbart, das einen Prozessor und einen mit dem Prozessor verbundenen Speicher aufweist, der darin gespeichert Befehle enthält, die, wenn sie von dem Computersystem ausgeführt werden, das Computersystem veranlassen, ein Verfahren zur Zuweisung bzw. Reservierung auszuführen. Das Verfahren umfasst das Empfangen einer Anforderung zur Ausführung einer Anwendung von einem Klienten-Gerät, das mit einem Endanwender assoziiert ist. Das Verfahren umfasst die Ermittelung einer ersten Leistungsklasse für die Anwendung. Das Verfahren umfasst die Ermittlung einer ersten virtuellen Maschine der ersten Leistungsklasse, die verfügbar ist. Das Verfahren umfasst die Zuordnung der ersten virtuellen Maschine zum Ausführen der Anwendung in Verbindung mit dem Klienten-Gerät.

[0017] Diese und andere Aufgaben und Vorteile der diversen Ausführungsformen der vorliegenden Offenbarung ergeben sich für den Fachmann aus dem Studium der folgenden detaillierten Beschreibung der Ausführungsformen, die in den diversen Zeichnungen dargestellt sind.

KURZE BESCHREIBUNG DER ZEICHNUNGEN

[0018] Die begleitenden Zeichnungen, die in dieser Beschreibung enthalten sind und einen Teil davon bilden und in denen gleiche Bezugszeichen gleiche Elemente zeigen, stellen Ausführungsformen der vorliegenden Offenbarung dar und dienen zusammen mit der Beschreibung dazu, die Prinzipien der Offenbarung zu erklären.

[0019] Fig. 1 zeigt eine Blockansicht eines anschaulichen Computersystems, das zur Implementierung von Ausführungsformen gemäß der vorliegenden Offenbarung geeignet ist.

[0020] Fig. 2 ist eine Blockansicht eines Beispiels eines Klienten-Geräts, das zur Implementierung von Ausführungsformen gemäß der vorliegenden Erfindung in der Lage ist.

[0021] Fig. 3 ist eine Blockansicht eines Beispiels einer Netzwerkarchitektur, in der Klienten-Systeme und Server bzw. Dienstleister mit einem Netzwerk gemäß Ausführungsformen der vorliegenden Erfindung verbunden sein können.

[0022] Fig. 4A zeigt ein Grafiksystem **400A**, das zur Implementierung einer wolkenbasierten virtualisierten Grafikverarbeitung für Fernanzeige gemäß einer Ausführungsform der vorliegenden Offenbarung konfigurierbar ist.

[0023] Fig. 4B ist eine Darstellung einer Architektur **400B**, die ausgebildet ist, eine wolkenbasierte virtualisierte Grafikverarbeitung für Fernanzeigen gemäß einer Ausführungsform der vorliegenden Offenbarung zu implementieren.

[0024] Fig. 5A–B sind Darstellungen einer dualen Sockel-Architektur bzw. Doppel-Sockel-Architektur, die mit mehreren Chipsätzen aus Grafikprozessoren versehen ist, die eingerichtet sind, ein oder mehrere mit Netzwerk verbundene GPU-Einrichtungen gemäß einer Ausführungsform der vorliegenden Offenbarung bereitzustellen.

[0025] Fig. 6 ist eine Darstellung der Implementierung mehrerer Verarbeitungsleiterplatten, die für mehrere GPU geeignet sind, um eine wolkenbasierte virtualisierte Grafikverarbeitung für Fernanzeigen bereitzustellen.

[0026] Fig. 7 ist eine Blockansicht eines Systems, das zur Erfassung eines Schlüsselbildblocks gemäß einer Ausführungsform der vorliegenden Offenbarung konfiguriert ist.

[0027] Fig. 8A ist ein Flussdiagramm, das ein Verfahren zum Erkennen eines Schlüsselbildblocks während der Ausführung einer Anwendung in einer virtuellen Maschine zeigt, die von einer Wolke-Rechenplattformen bzw. Cloud-Rechenplattformen unterstützt wird, wodurch eine virtualisierte Grafikverarbeitung für Fernanzeigen gemäß einer Ausführungsform der vorliegenden Offenbarung bereitgestellt wird.

[0028] Fig. 8B ist ein Informationsflussdiagramm, das den Prozess zum Erkennen eines Schlüsselbildblocks gemäß einer Ausführungsform der vorliegenden Offenbarung zeigt.

[0029] Fig. 9 ist eine Blockansicht eines Ressourcenerzeugungssystems, das ausgebildet ist, neue virtuelle Maschinen unter Anwendung einer Schablone einer virtuellen Maschine zu erzeugen und jede Instanz einer virtuellen Maschine mit anwenderspezifischen Daten gemäß einer Ausführungsform der vorliegenden Offenbarung speziell auszulegen.

[0030] Fig. 10A ist ein Flussdiagramm, das ein Verfahren zur Ressourcenerzeugung für eine neue virtuelle Maschine zeigt, die eine wolkenbasierten virtualisierte Grafikverarbeitung für eine Fernanzeige gemäß einer Ausführungsform der vorliegenden Offenbarung bereitstellt.

[0031] Fig. 10B ist ein Flussdiagramm, das ein Verfahren zur Drosselung der Zuweisungen bzw. Reservierungen von Ressourcen zeigt, wenn eine neue virtuelle Maschine erzeugt wird, um ihren Einfluss auf die Funktion bestehender virtuellen Maschinen gemäß einer Ausführungsform der vorliegenden Offenbarung zu verringern.

[0032] Fig. 11 ist eine Blockansicht eines Systems, das in der Lage ist, eine Fensterverwaltung in einer Fernanzeige zum Zwecke der Minimierung der Sichtbarmachung eines Desktop-Betriebssystems auf einem Frontfenster der Fernanzeige gemäß einer Ausführungsform der vorliegenden Offenbarung auszuführen.

[0033] Fig. 12 ist ein Flussdiagramm, das ein Verfahren zur Ausführung einer Fensterverwaltung in einer Fernanzeige zum Zwecke der Minimierung der Sichtbarmachung eines Desktop-Betriebssystems auf einem Frontfenster der Fernanzeige gemäß einer Ausführungsform der vorliegenden Offenbarung zeigt.

[0034] Fig. 13A–O zeigen die Implementierung einer Spieleplattform, die eine wolkenbasierte virtualisierte Grafikverarbeitung für Fernanzeigen gemäß Ausführungsformen der vorliegenden Offenbarung bereitstellt.

[0035] Fig. 14A–H sind Diagramme, die ein System und ein Verfahren zum dynamischen Zuweisen und Zuordnen von Spielenamen in einer wolkenbasierten Spiele/Anwendungsumgebung in Ausführungsformen der vorliegenden Erfindung zeigen.

[0036] Fig. 15 ist ein Flussdiagramm, das ein Verfahren zur Zuweisung einer virtuellen Maschine zu einem Anwender-Klienten in einem wolkenbasierten Grafikverarbeitungssystem zeigt.

[0037] Fig. 16 ist ein Flussdiagramm **1600**, das ein computerimplementiertes Verfahren zur Implementierung einer globalen Verriegelung zeigt, um die Handhabung von Anforderungen gemäß einer Ausführungsform der vorliegenden Offenbarung zu ordnen.

[0038] Fig. 17A–F sind Darstellungen diverser Verfahren, die zur Platzzuweisung in einem wolkenbasierten Grafikverarbeitungssystem gemäß Ausführungsformen der vorliegenden Offenbarung implementiert sind.

DETAILLIERTE BESCHREIBUNG DER ERFINDUNG

[0039] Es sei nun detailliert auf die diversen Ausführungsformen der vorliegenden Offenbarung verwiesen, wovon Beispiele in den begleitenden Zeichnungen dargestellt sind. Obwohl die Beschreibung in Verbindung

mit diesen Ausführungsformen angegeben ist, ist zu beachten, dass diese Ausführungsformen nicht beabsichtigen, die Offenbarung auf diese Ausführungsformen einzuschränken.

[0040] Vielmehr beabsichtigt die Offenbarung, Alternativen, Modifizierungen und Äquivalente abzudecken, die innerhalb des Grundgedankens und des Schutzbereichs der Offenbarung liegen, wie sie durch die angefügten Patentansprüche definiert ist. Ferner sind in der folgenden detaillierten Beschreibung der vorliegenden Offenbarung zahlreiche spezielle Details angegeben, um ein gründlicheres Verständnis der vorliegenden Offenbarung zu ermöglichen. Es ist jedoch zu beachten, dass die vorliegende Offenbarung ohne diese speziellen Details in die Praxis umgesetzt werden kann. In anderen Fällen sind gut bekannte Verfahren, Prozeduren, Komponenten und Schaltungen nicht detailliert beschrieben, um nicht in unnötiger Weise die vorliegende Offenbarung zu verdunkeln.

[0041] Einige Teile der folgenden detaillierten Beschreibungen sind in Begriffen von Prozeduren, Logikblöcken, Verarbeitung und anderen symbolischen Darstellungen von Operationen an Datenbits innerhalb eines Computerspeichers angegeben. Diese Beschreibungen und Darstellungen sind Mittel, die von dem Fachmann verwendet werden, um in höchst effizienter Weise den Inhalt seiner Arbeit anderen Fachleuten zu vermitteln. In der vorliegenden Anmeldung ist eine Prozedur, ein Logikblock, ein Prozess oder dergleichen als eine selbstkonsistente Sequenz aus Schritten oder Befehlen zu verstehen, die zu einem gewünschten Ergebnis führen. Die Schritte sind solche, die physikalische Änderungen an physikalischen Größen verwenden. Für gewöhnlich, ohne dass dies jedoch erforderlich ist, nehmen diese Größen die Form elektrischer oder magnetischer Signale an, die in einem Computersystem gespeichert, übertragen, kombiniert, verglichen und anderweitig verarbeitet werden können. Es hat sich gelegentlich als günstig erwiesen, insbesondere aus Gründen der üblichen Nutzung, diese Signale als Transaktionen, Bits, Werte, Elemente, Symbole, Zeichen, Abtastwerte, Pixel oder dergleichen zu bezeichnen.

[0042] Es sollte jedoch beachtet werden, dass alle diese und ähnliche Begriffe mit den geeigneten physikalischen Größen zu verknüpfen sind und lediglich bequeme Namen darstellen, die diesen Größen verliehen werden. Sofern dies nicht anderweitig in den folgenden Erläuterungen angegeben ist oder aus diesen hervorgeht, ist zu beachten, dass durchgängig in der vorliegenden Offenbarung Erläuterungen unter Verwendung von Computerbegriffen, etwa „ausführen", „empfangen", „verbinden", „navigieren", „ermöglichen", „installieren", oder dergleichen, Aktionen und Prozesse eines Computersystems oder einer ähnlichen elektronischen Recheneinrichtung oder eines Prozessors bezeichnen (beispielsweise in den Flussdiagrammen **Fig.** 8A, **Fig.** 10A–B, **Fig.** 12, **Fig.** 15, **Fig.** 16 und **Fig.** 17A–F der vorliegenden Anmeldung). Das Computersystem oder die ähnliche elektronische Recheneinrichtung verarbeitet und transformiert Daten, die als physikalische (elektronische) Größen innerhalb der Speicher, Register oder anderen Informationsspeichern, Übertragungsgeräten oder Anzeigegeräten des Computersystems dargestellt sind.

[0043] **Fig.** 8A, **Fig.** 10A–B, **Fig.** 12, **Fig.** 15, **Fig.** 16 und **Fig.** 17A–F sind Flussdiagramme von Beispielen von computerimplementierten Verfahren zur Bereitstellung einer Cloud basierten bzw. wolkenbasierten virtualisierten, das heißt virtuell erzeugten, Grafikverarbeitung für entfernte Anzeigen bzw. Fernanzeige gemäß Ausführungsformen der vorliegenden Erfindung. Obwohl spezielle Schritte in den Flussdiagrammen offenbart sind, sind derartige Schritte nur anschaulicher Natur. Das heißt, Ausführungsformen der vorliegenden Erfindung sind gut geeignet, um diverse andere Schritte oder Variationen der Schritte, die in den Flussdiagrammen genannt sind, auszuführen.

[0044] Andere hierin beschriebene Ausführungsformen werden gegebenenfalls im allgemeinen Zusammenhang von Befehlen, die von einem Computer ausführbar sind, erläutert, die in einer gewissen Form eines computerlesbaren Speichermediums abgelegt sind, etwa in Programmmodulen, die von einem oder mehreren Computern oder anderen Geräten ausgeführt werden. Beispielsweise, ohne einschränken zu wollen, umfassen computerlesbare Speichermedien nicht-flüchtige Computerspeichermedien und Kommunikationsmedien. Generell umfassen Programmmodule Routinen, Programme, Objekte, Komponenten, Datenstrukturen etc., die spezielle Aufgaben ausführen oder spezielle abstrakte Datentypen implementieren. Die Funktion der Programmmodule kann kombiniert oder verteilt sein entsprechend dem Bedarf in diversen Ausführungsformen.

[0045] Zu Computerspeichermedien gehören flüchtige und nicht-flüchtige, entfernbare und nicht-entfernbare Medien, die in beliebigen Verfahren oder Technologien zur Speicherung von Information eingerichtet sind, etwa computerlesbare Befehle, Datenstrukturen, Programmmodule oder andere Daten. Zu Computerspeichermedien gehören, ohne Einschränkung, ein Speicher mit wahlfreiem Zugriff (RAM), einen Nur-Lese-Speicher (ROM), elektrisch löschbare programmierbare ROM (EEPROM), ein Flash-Speicher oder eine andere Speichertechnologie, Kompaktdisketten-ROM (CD-ROM), digitale Vielseitigkeitsdisketten (DVDs) oder andere op-

tische Speicher, Magnetkassetten, Magnetbänder, ein magnetischer Plattenspeicher oder andere magnetische Speichereinrichtungen oder ein anderes Medium, das verwendet werden kann, um die gewünschte Information zu speichern und auf das zugegriffen werden kann, um diese Information abzurufen.

[0046] Kommunikationsmedien können von einem Computer ausführbare Befehle, Datenstrukturen und Programmmodule verkörpern und umfassen beliebige Informationsbereitstellungsmedien. Beispielsweise, ohne Einschränkung, gehören zu Kommunikationsmedien verdrahtete Medien, etwa ein verdrahtetes Netzwerk oder eine direkt verdrahtete Verbindung, und drahtlose Medien, etwa akustische, Hochfrequenz-(HF-), Infrarot-oder andere drahtlose Medien. Auch Kombinationen beliebiger Medien der obigen Medien können im Bereich der computerlesbaren Medien mit eingeschlossen sein.

[0047] **Fig.** 1 ist eine Blockansicht eines Beispiels eines Rechensystems **100**, das in der Lage ist, Ausführungsformen der vorliegenden Offenbarung zu implementieren. Das Rechensystem **100** repräsentiert allgemein eine Recheneinrichtung oder ein Rechensystem mit einzelnem Prozessor oder mehreren Prozessoren, das in der Lage ist, computerlesbare Befehle auszuführen. Zu Beispielen des Rechensystems **100** gehören, ohne Einschränkung, Arbeitsplatzrechner, Klapprechner, klientenseitige Endgeräte, Server bzw. Dienstleistungsrechner bzw. Dienstleister, verteilte Rechensysteme, tragbare Geräte oder ein beliebiges anderes Rechensystem oder eine Einrichtung. In der grundlegendsten Konfiguration kann das Rechensystem **100** zumindest einen Prozessor **105** und einen Systemspeicher **110** aufweisen.

[0048] Zu beachten ist, dass das Computersystem **100**, das hierin beschrieben ist, eine beispielhafte Konfiguration einer funktionsfähigen Plattform zeigt, in der Ausführungsformen vorteilhaft implementiert werden können. Dennoch können auch andere Computersysteme mit unterschiedlichen Konfigurationen anstelle des Computersystems **100** innerhalb des Schutzbereichs der vorliegenden Erfindung angewendet werden. D. h., das Computersystem **100** kann Elemente enthalten, die sich von denen unterscheiden, die in Verbindung mit **Fig.** 1 beschrieben sind. Ferner können Ausführungsformen in einem beliebigen System umgesetzt werden, das konfiguriert werden kann, um dies zu ermöglichen, wobei dies Computersysteme sein können, die nicht gleich dem Computersystem **100** sind. Zu beachten ist, dass Ausführungsformen in vielen unterschiedlichen Arten von Computersystemen **100** umgesetzt werden können. Das System **100** kann beispielsweise als ein Tischrechner-Computersystem oder als ein Dienstleister-Computersystem implementiert werden mit einer leistungsfähigen CPU für Allgemeinzwecke, die mit einer speziellen grafikerzeugenden GPU verbunden ist. In einer derartigen Ausführungsform können Komponenten mit eingeschlossen sein, die Peripheriebusse, spezielle Audio/Video-Komponenten, I/O-Geräte und dergleichen hinzufügen. In ähnlicher Weise kann das System **100** als ein Handgerät bzw. tragbares Gerät (beispielsweise Funktelefon, etc.) oder als eine Fernsehgerät gestützte Videospiel-Konsole, beispielsweise Xbox®, die von der Microsoft Corporation, Redmond, Washington erhältlich ist, oder der PlayStation3®, die von der Sony Computer Entertainment Corporation, Tokio, Japan erhältlich ist, implementiert sein. Das System **100** kann ferner als ein „System auf einem Chip" implementiert sein, in der die Elektronik (beispielsweise die Komponenten **105**, **110**, **115**, **120**, **125**, **130**, **150** und dergleichen) einer Recheneinrichtung vollständig in einem einzelnen integrierten Schaltungschip enthalten sind. Zu Beispielen gehören ein Handinstrument mit einer Anzeige, ein Fahrzeugnavigationssystem, ein tragbares Unterhaltungssystem, und dergleichen.

[0049] In dem Beispiel der **Fig.** 1 umfasst das Computersystem **100** eine zentrale Recheneinheit (CPU) **105**, um Software-Anwendungen und optional ein Betriebssystem auszuführen. Ein Speicher **110** speichert Anwendungen und Daten zur Verwendung durch die CPU **105**. Ein Speicher **115** liefert nicht flüchtigen Speicherplatz für Anwendungen und Daten und kann stationäre Plattenlaufwerke, entfernbare Pattenlaufwerke, Flash-Speichergeräte und CD-ROM, DVD-ROM oder andere optische Speichereinrichtungen umfassen. Die optionale Anwendereingabeeinrichtung **120** umfasst Geräte bzw. Einrichtungen, die Anwendereingaben von einem oder mehreren Anwendern an das Computersystem **100** übertragen und es können dazu gehören: Tastaturen, Mäuse, Joysticks, berührungsempfindliche Bildschirme und/oder Mikrofone.

[0050] Die Kommunikationsschnittstelle oder Netzwerkschnittstelle **125** ermöglicht es dem Computersystem **100**, mit anderen Computersystemen über ein elektronisches Kommunikationsnetzwerk in Verbindung zu treten, wozu verdrahtete und/oder drahtlose Kommunikation einschließlich des Internets gehört. Das optionale Anzeigegerät **150** kann ein beliebiges Gerät sein, das in der Lage ist visuelle Information in Reaktion auf ein Signal aus dem Computersystem **100** anzuzeigen. Die Komponenten des Computersystems **100** einschließlich der CPU **105**, dem Speicher **110**, dem Datenspeicher **115**, den Anwender-Eingabegeräten **120**, der Kommunikationsschnittstelle **125** und dem Anzeigegerät **150** können über einen oder mehrere Datenbusse **160** verbunden sein.

[0051] In der Ausführungsform der **Fig.** 1 kann ein Grafiksystem **130** mit dem Datenbus **160** und den Komponenten des Computersystems **100** verbunden sein.

[0052] Das Grafiksystem **130** kann eine physikalische Graphikverarbeitungseinheit (GPU) **135** und einen Grafikspeicher aufweisen. Die GPU **135** erzeugt Pixeldaten für Ausgabebilder aus bilderzeugenden Befehlen. Die physikalische GPU **135** kann als mehrere virtuelle GPU konfiguriert sein, die parallel (gleichzeitig) von einer Anzahl von Anwendungen, die parallel ausgeführt werden, verwendet werden können.

[0053] Ein Grafikspeicher kann einen Anzeigespeicher **140** enthalten (beispielsweise einen Blockpuffer), der zur Speicherung von Pixeldaten für jedes Pixel in einem Ausgabebild verwendet wird. In einer weiteren Ausführungsform kann der Anzeigespeicher **140** und/oder ein zusätzlicher Speicher **145** ein Teil des Speichers **110** sein und kann gemeinsam mit der CPU **105** verwendet werden. Alternativ können der Anzeigespeicher **140** und/oder der zusätzliche Speicher **145** ein oder mehrere separate Speicher sein, die ausschließlich zur Verwendung durch das Grafiksystem **130** vorgesehen sind.

[0054] In einer weiteren Ausführungsform umfasst das Grafikverarbeitungssystem **130** eine oder mehrere zusätzliche physikalische GPU **155** ähnlich zu der GPU **135**. Jede zusätzliche GPU **155** kann ausgebildet sein, parallel mit der GPU **135** zu arbeiten. Jede zusätzliche GPU **155** erzeugt Pixeldaten für Ausgabebilder aus bilderzeugenden Befehlen. Jede zusätzliche physikalische GPU **155** kann als mehrere virtuelle GPU konfiguriert sein, die parallel (gleichzeitig) von einer Reihe von Anwendungen, die parallel ausgeführt werden, verwendet werden. Jede zusätzliche GPU **155** kann in Verbindung mit der GPU **135** arbeiten, um gleichzeitig Pixeldaten für unterschiedliche Bereiche eines Ausgabebilds zu erzeugen, oder um gleichzeitig Pixeldaten für unterschiedliche Ausgabebilder zu erzeugen.

[0055] Jede zusätzliche GPU **155** kann auf der gleichen Leiterplatte bzw. Leiterplatte wie die GPU **135** angeordnet sein, wodurch eine Verbindung zu dem Datenbus **160** gemeinsam mit der GPU **135** benutzt wird, oder jede zusätzliche GPU **155** kann auf einer weiteren Leiterplatte angeordnet sein, die separat mit dem Datenbus **160** verbunden ist. Jede zusätzliche GPU **155** kann ferner in das gleiche Modul oder das gleiche Chipgehäuse wie die GPU **135** integriert sein. Jede zusätzliche GPU **155** kann zusätzlichen Speicher ähnlich zu dem Anzeigespeicher **140** und dem zusätzlichen Speicher **145** aufweisen, oder diese kann die Speicher **140** und **145** gemeinsam mit der GPU **135** benutzen.

[0056] **Fig.** 2 ist eine Blockansicht eines Beispiels eines Endanwender-Geräts oder Klienten-Geräts **200**, in welchem Ausführungsformen gemäß der vorliegenden Erfindung eingerichtet werden können. In dem Beispiel aus **Fig.** 2 umfasst das Klienten-Gerät **200** eine CPU **205**, um Software-Anwendungen und optional ein Betriebssystem auszuführen. Das Anwender-Eingabegeräte **220** umfasst Geräte, die Anwendereingaben von einem oder mehreren Anwendern weitergeben und wozu Tastaturen, Mäuse, Joysticks, berührungsempfindliche Bildschirme und/oder Mikrofone gehören.

[0057] Die Kommunikationsschnittstelle **225** ermöglicht es dem Klienten-Gerät **200**, mit anderen Computersystemen (beispielsweise dem Computersystem **100** aus **Fig.** 1) über ein elektronisches Kommunikationsnetzwerk zu kommunizieren, wozu eine verdrahtete und/oder drahtlose Kommunikation gehört und wozu das Internet gehört. Der Dekodierer **255** kann ein beliebiges Gerät sein, das in der Lage ist, Daten zu dekodieren (dekomprimieren), die kodiert (komprimiert) sein können. Beispielsweise kann der Dekodierer **255** ein H.264-Dekodierer sein. Das Anzeigegerät **250** kann ein beliebiges Gerät sein, das in der Lage ist, visuelle Information anzuzeigen, wozu Information gehört, die von dem Dekodierer **255** erhalten wird. Das Anzeigegerät **250** kann verwendet werden, um diese Information, die zumindest teilweise von dem Klienten-Gerät **200** erzeugt wurde, anzuzeigen. Jedoch kann das Anzeigegerät **250** auch verwendet werden, um visuelle Information anzuzeigen, die von dem Computersystem **100** erhalten wird. Die Komponenten des Klienten-Geräts **200** können über einen oder mehrere Datenbusse **260** miteinander verbunden sein. Ferner können die Komponenten physikalisch innerhalb des Gehäuses des Klienten-Geräts **200** angeordnet sein oder auch nicht. Beispielsweise kann die Anzeige **250** ein Monitor sein, mit dem das Klienten-Gerät **200** über ein Kabel oder drahtlos kommuniziert.

[0058] Im Vergleich zu dem Computersystem **100** kann das Klienten-Gerät **200** in dem Beispiel aus **Fig.** 2 weniger Komponenten und weniger Funktionen aufweisen und kann damit als ein schlanker Klient bezeichnet werden. Generell kann das Klienten-Gerät **200** eine beliebige Art eines Geräts bzw. einer Einrichtung sein, die Anzeigefähigkeiten, die Fähigkeit zum Dekodieren (Dekomprimieren) von Daten und die Fähigkeit besitzt, Eingaben von einem Anwender zu empfangen und derartige Eingaben an das Computersystem **100** zu senden. Jedoch kann das Klienten-Gerät **200** weitere Eigenschaften besitzen, die über die genannten Eigenschaften

hinausgehen. Das Klienten-Gerät **200** kann beispielsweise ein Personalcomputer, ein Tablett-Computer, ein Fernsehgerät, ein tragbares Spielesystem oder dergleichen sein.

[0059] **Fig.** 3 ist eine Blockansicht eines Beispiels einer Netzwerkarchitektur **200**, in der Klienten-Systeme **310**, **320** und **330** und Server bzw. Dienstleister **340** und **345** mit einem Netzwerk **350** verbunden sind. Die Klienten-Systeme **310**, **320** und **330** repräsentierten generell eine beliebige Art oder Form einer Recheneinrichtung oder eines Rechensystems, etwa des Rechensystems **110** aus **Fig.** 1 und/oder des Klienten-Geräts **200** aus **Fig.** 2.

[0060] In ähnlicher Weise repräsentieren die Dienstleister **340** und **345** generell Recheneinrichtungen oder Rechensysteme, etwa Anwendungen-Dienstleister, GPU-Dienstleister oder Datenbank-Dienstleister, die ausgebildet sind, diverse Datenbankdienstleistungen und/oder gewisse Software-Anwendungen auszuführen. Das Netzwerk **350** repräsentiert generell ein beliebiges Telekommunikationsnetzwerk oder Computernetzwerk, was beispielsweise ein internes Netz, ein Weitbereichsnetzwerk (WAN), ein lokales Netzwerk (LAN), ein Netzwerk im persönlichen Bereich (PAN) oder das Internet mit einschließt.

[0061] In Verbindung mit dem Rechensystem **100** aus **Fig.** 1 kann eine Kommunikationsschnittstelle, etwa eine Kommunikationsschnittstelle **125**, verwendet werden, um eine Verbindung zwischen jedem Klienten-System **310**, **320** und **330** und dem Netzwerk **350** herzustellen. Die Klienten-Systeme **310**, **320** und **330** können in der Lage sein, auf Information auf dem Dienstleister **340** oder **345** unter Anwendung beispielsweise einer Netz-Suchanwendungen bzw. Browser-Anwendung oder einer anderen Klienten-Software zu zugreifen. Auf diese Weise sind die Klienten-Systeme **310**, **320** und **330** konfigurierbar, um auf die Dienstleister **340** und/oder **345** zuzugreifen, die Grafikverarbeitungsfähigkeiten bieten, wodurch die Grafikverarbeitung auf die nachgeordneten Dienstleister **340** und/oder **345** übertragen wird, um eine Anzeige in den vorgeordneten Klienten-Systemen **310**, **320** und **330** bereitzustellen. Ferner gestattet es eine derartige Software, dass die Klienten-Systeme **310**, **320** und **330** auf Daten zugreifen, die auf dem Dienstleister **340**, dem Dienstleister **345**, den Speichereinrichtungen **360(1)**–(L), Speichereinrichtungen **370(1)**–(N), Speichereinrichtungen **390(1)**–(M) oder einem intelligenten Speicherarray **395** bewahrt werden. Obwohl **Fig.** 3 die Verwendung eines Netzwerks (etwa das Internet) für den Austausch von Daten zeigt, sind die hierin beschriebenen Ausführungsformen nicht auf das Internet oder eine spezielle netzwerkbasierte Umgebung beschränkt.

[0062] In einer Ausführungsform sind eine oder mehrere der hierin offenbarten beispielhaften Ausführungsformen in ihrer Gesamtheit oder teilweise als ein Computerprogramm kodiert und werden in den Dienstleister **340**, den Dienstleister **345**, die Speichereinrichtungen **360(1)**–(L), die Speichereinrichtungen **370(1)**–(N), die Speichereinrichtungen **390(1)**–(M), das intelligente Speicherarray **395**, oder Kombinationen davon geladen und von diesem ausgeführt. Eine oder mehrere der hierin offenbarten anschaulichen Ausführungsformen können in ihrer Gesamtheit oder teilweise ebenfalls als ein Computerprogramm kodiert sein, die in dem Dienstleister **340** gespeichert sind, von dem Dienstleister **345** ausgeführt und auf die Klienten-Systeme **310**, **320** und **330** über das Netzwerk **350** verteilt werden.

Verfahren und Systeme für eine NETZ-Architektur, die wolkenbasierte virtualisierte

Grafikverarbeitung für Fernanzeigen bereitstellt

[0063] **Fig.** 4A zeigt ein Grafiksystem **400A**, das zum Implementieren einer wolkenbasierten virtualisierten Grafikverarbeitung für Fernanzeigen gemäß einer Ausführungsform der vorliegenden Offenbarung konfigurierbar ist. Wie gezeigt, umfasst das Grafikverarbeitungssystem **400A** eine physikalische GPU **135** aus **Fig.** 1, obwohl das System **400A** zusätzliche physikalische GPU **155** enthalten kann, wie dies zuvor beschrieben ist.

[0064] Gemäß Ausführungsformen der vorliegenden Erfindung ist die physikalische GPU **135** für die gleichzeitige Nutzung durch eine Anzahl N von Anwendungen 1, 2, ..., N ausgelegt. Insbesondere ist die physikalische GPU **135** als eine Anzahl M virtueller GPU **415A**, **415B**, ..., **415M** konfiguriert, die gleichzeitig von den Anwendungen 1, 2, ..., N verwendet werden. Jede der zusätzlichen GPU **155** kann in ähnlicher Weise wie mehrere virtuelle GPU konfiguriert sein. In einer Ausführungsform sind die GPU **135** und die zusätzlichen GPU **155** mit einer Speicherverwaltungseinheit **420** (MMU; beispielsweise eine Eingabe/Ausgabe-MMU) verbunden, die wiederum mit dem Grafikspeicher verbunden ist, der in Verbindung mit **Fig.** 1 beschrieben ist.

[0065] In einer Ausführungsform sind die Anwendungen 1, 2, ..., N Videospiele-Anwendungen; jedoch ist die Erfindung nicht darauf eingeschränkt. D. h., die Anwendungen 1, 2, ..., N können eine beliebige Art von Anwendung sein. Beispielsweise kann die Anwendung Finanzdienstleistungen, Dienstleistungen für computergestützte Gestaltung (CAD), etc. bereitstellen. In einer noch weiteren Beispiel kann die Anwendung eine Pro-

grammieranleitung sein, die in Tabellenform eine Liste diverser Programme bietet, die auf unterschiedlichen Fernsehkanälen in unterschiedlichen Zeitfenstern verfügbar sind, und das Klienten-Gerät kann eine digitale Fernsehbox (Kabel oder Satellit) sein.

[0066] **Fig.** 4B ist eine Darstellung einer Architektur **400B**, die eingerichtet ist, eine wolkenbasierte virtualisierte Grafikverarbeitung für Fernanzeigen gemäß einer Ausführungsform der vorliegenden Offenbarung zu implementieren. Die Grafikarchitektur **400B** stellt Netzwerkressourcen bereit, die von einem oder mehreren Betriebssystemen eines übergeordneten Rechners verwaltet werden, wobei die Netzwerkressourcen (beispielsweise CPU, GPU, etc.) virtuell erzeugt bzw. virtualisiert und in einer oder mehreren virtuellen Maschinen (VM) und externen Kommunikationsnetzwerken gemeinsam genutzt werden. Insbesondere ist ein Cloud-System bzw. Wolke-System **45A**–N gezeigt, wobei alle Wolke-Systeme **450A**–N in Kooperation sind, um mehrere Rechnerressourcen über ein Kommunikationsnetzwerk miteinander zu verbinden. Damit sind die Rechnerressourcen über die Architektur **400B** hinweg verteilt. In Ausführungsformen der vorliegenden Erfindung liefern die Wolke-Systeme **450A**–N vollständige, virtualisierte Spielesysteme, die auch ausgelegt sind, eine Grafikverarbeitung mit hoher Leistung bereitzustellen.

[0067] Beispielsweise stellt das Wolke-System **450A** verteilte Rechnerressourcen bereit und ist repräsentativ für jedes der Wolke-Systeme **450A**–N, das eine wolkenbasierte virtualisierte Grafikverarbeitung für Fernanzeigen bereitstellt. Insbesondere umfasst das Wolke-System **450A** ein Spiele-Maschinen-(GM)Dienstleister-Wolke-System **460**, das mehrere physikalische Dienstleister oder Spielemaschinen **461A**–N umfasst. Das GM-System **461A** ist hierin beschrieben und ist repräsentativ für jede der GM **461A**–N. Insbesondere stellt die physikalische GM **461A** ein übergeordnetes System oder ein Host-System bereit, das mehrere virtuelle Maschinen unterstützt bzw. bereitstellt. Zur Implementierung der virtuellen Maschinen **455A**–N in der GM **461A** beinhaltet ein VM-Host-Verwalter **462** in Verbindung mit dem Hyper-Revisor **451** Software, Firmware oder Hardware, die die virtuellen Maschinen (auch als „Spieleplätze" bezeichnet) **455A**–N erzeugen, verwalten und ausführen. Insbesondere tritt der VM-Host-Verwalter **462** mit einem Hyper-Revisor **451** in Wechselwirkung, um die Spieleplätze **455A**–N in den GM-System **461A** zuzuweisen. Genauer gesagt, der VM-Host-Verwalter ist in der Lage, die physikalischen Ressourcen zu binden und eine Verbindung zu diesen herzustellen (beispielsweise ein oder mehrere CPU-Kerne und die physikalischen GPU), die einem speziellen Spieleplatz (beispielsweise **455A**) zugewiesen sind.

[0068] Der Spieleplatz (beispielsweise VM) **455A** ist repräsentativ für die Spieleplätze **455A**–N innerhalb des physikalischen Dienstleisters oder der GM **461A**. Jeder virtuelle Spieleplatz **455A** arbeitet innerhalb seiner eigenen virtuellen Umgebung, die mittels eines entsprechenden Betriebssystems ausgeführt wird, etwa ein virtualisiertes Windows-Betriebssystem. Beispielsweise verleiht das Betriebssystem dem Spieleplatz **455A** die Fähigkeit, die Spieleanwendung **458A** auszuführen, diese in einen Video-Strom umzuwandeln und diesen aus einem geeigneten Anschluss und einem Kanal zu einem Empfänger in dem Klienten-Dienstleister **490** zu senden. Der Klienten-Dienstleister **490** umfasst eine beliebige elektronische Einrichtung, die ausgebildet ist, mit dem Wolke-System **450A**–N für eine Instantiierung einer virtuellen Maschine zu kommunizieren. Beispielsweise umfasst der Klienten-Dienstleister **490** einen schlanken Klienten, einen nicht intelligenten Dienstleister, ein Mobilgerät, einen Klapprechner, einen Personalcomputer, etc..

[0069] Ein Spiele-Agent **446A** instantiiert den Spieleplatz **455A** und hilft bei der Verwaltung und der Koordination der mehreren Spieleplätze innerhalb des Wolke-Systems **450A**. Beispielsweise wirken der Spiele-Agent **456A** zusammen mit all den anderen Spiele-Agenten zusammen mit einem Bereitstellungsverwalter **471** einer Bereitstellungsverwalter-Wolke **470**, um die notwendigen Spieleplätze innerhalb des Wolke-Systems **450A**–N zu erzeugen, zu verwalten und den Endklienten-Systemen **490** zuzuführen. Beispielsweise arbeitet der Bereitstellungsverwalter **470** mit dem Klienten-System **490** zusammen, um einen entsprechenden Spieleplatz bereitzustellen (beispielsweise **455A**). Ferner enthält der Bereitstellungsverwalter Information, die mit jeder der Spieleanwendungen (beispielsweise **458**) in Beziehung steht, und die in dem Spieletitel-Bestands-Block **467** enthalten ist. Damit ist der Spiele-Agent **456A** in der Lage, Information zu empfangen, die mit der Spieleanwendung **458** in Beziehung steht, die in dem Spieleplatz **455A** instantiiert ist, etwa eine beschreibende Informationen, empfohlene Konfigurationseinstellungen auf der Grundlage der Eigenschaften des Spieleplatzes **455A**, etc.. Eine zuerst-hinein-zuerst-heraus-(FIFO-)Komponente **468** ordnet die eintreffenden Anforderungen für Spieleplätze. In einer Ausführungsform handhabt die FIFO-Komponente Anforderungen in der FIFO-Reihenfolge.

[0070] Ein Mjölnir-Dienstleister **457A** stellt die Kodierung und die Paketbildung für Informationen bereit. Beispielsweise ist der Mjölnir-Dienstleister **457A** ausgebildet, grafische Videodaten zu kodieren und zu Pakete zu formen, die von dem Spieleplatz (VM) **455A** durch die Ausführung einer Spieleanwendung **458A** erzeugt

werden. Wie gezeigt, wird die Instantiierung der Spieleanwendung **458A** von einem Metrikspeicher **471** bereitgestellt, der mit einem Dritt-Partei-Inhalte-Dienstleister bzw. Fremd-Inhlate-Dienstleister verbunden ist, der Zugriff zu einer Instanz der Spieleanwendung **458A** bereitstellt.

[0071] Ein OpKonsole-Dienstleister **463** koordiniert die diversen VM-Host-Verwalter (beispielsweise **462**) an den Knoten der physikalischen Dienstleister **461A**–N. Insbesondere dient der OpKonsole-Dienstleister **463** als ein Cluster-Verwalter für die Cluster aus Dienstleistern oder GM **461A**–N. Beispielsweise informiert der OpKonsole-Dienstleister **463** die Dienstleister-Registrierung **464** oder die Platzregistrierung, welche Spieleplätze funktionsbereit sind, welche Spieleplätze verwendet werden, welche Spieleplätze abgeschaltet sind, welche Spieleplätze in der Warteschlange sind, welche Spieleplätze zurückgesetzt werden, welche Spieleplätze gewartet werden müssen, etc..

[0072] Ferner interagiert der OpKonsole-Dienstleister **463** mit der Dienstleistungsregistrierung **464** derart, dass der aktuelle Status des gesamten Clusters an Spieleplätzen bereitgestellt wird. Der Status von Spieleplätzen wird in der Dienstleistungsregistrierung **464** bewahrt. Ferner ist der Verteilungsverwalter **465** ausgebildet, einen Benachrichtigungsdienst für jeden der Spieleplätze bereitzustellen. Beispielsweise ist der Verteilungsverwalter **465** in der Lage, Informationen über und Instantiierungen von neuen Spielern oder Information über neue Spiele, neue Software-Aktualisierungen, neue Versionen von Software, etc. weiterzugeben. Auf diese Weise stellt der Verteilungsverwalter **465** eine skalierbare Methode zur Benachrichtigung des Spiele-Agenten und jedes Spieleplatzes über relevante Information.

[0073] Der Metrik-Vertrieb bzw. Store **471** ist ausgebildet, Daten, die die Funktion der Spieleanwendungen innerhalb entsprechender Spieleplätze betreffen, zu sammeln. Beispielsweise sammelt der Metrik-Vertrieb **471** Bit-Raten, Qualität der Dienstleistung (QoS), etc.. Die Lastausgleichseinheit **466** gleicht die Last bzw. Auslastung für Verbindungsanfragen für Spieleplätze aus, die von den Klienten-Dienstleister **490** eingehen.

Visueller Computerapparat (VCA) zur Einrichtung einer
wolkenbasierten virtualisierten Grafikverarbeitung für Fernanzeigen

[0074] **Fig.** 5A ist eine Darstellung einer symmetrischen NETZ-Systemarchitektur bzw. GRID-Systemarchitektur **500A**, die eine Grafikverarbeitung gemäß einer Ausführungsform der vorliegenden Offenbarung bereitstellt. Beispielsweise ist die Architektur als eine Leiterplatte eingerichtet, die einen oder mehrere integrierte Chips aufweist, die miteinander kommunizierend verbunden sind, um grafische Verarbeitungsfähigkeiten bereitzustellen. Die NETZ-Architektur **500A** ist in der wolkenbasierten Architektur **400B** implementiert, um in einer Ausführungsform die wolkenbasierte virtualisierte Grafikverarbeitung für Fernanzeigen zu implementieren.

[0075] Die NETZ-Systemarchitektur **500A** enthält eine Doppel-Sockel-Architektur, die einen ersten CPU-Sockel **501A** und einen zweiten CPU-Sockel **501B** enthält. In einer Implementierung ist jeder CPU-Sockel ausgebildet, elektrische Verbindungen zwischen einem Mikroprozessor, der CPU-Fähigkeiten bereitstellt, und einer darunter liegenden Leiterplatte bereitzustellen. Wie in **Fig.** 5A gezeigt ist, ist ein Mehrkern-Prozessor kommunizierend mit jeweils dem ersten und dem zweiten CPU-Sockel verbunden. Beispielsweise enthält eine integrierte Schaltung oder ein „Chipsatz" Mehrkern-Prozessoren, die Hardware-Komponenten eines Rechensystems als Kombination bilden, das ausgebildet ist, Befehle eines Computerprogramms, das entsprechend implementiert ist, durch Operationen eines Betriebssystems auszuführen. In einer Ausführungsform umfasst der Mehrkern-Prozessor einen XEON E 2670-Prozessor. Wie in **Fig.** 5A gezeigt ist, ist ein XEON E 2670-Prozessor mit dem ersten Sockel **501A** und ein XEON E 2670-Prozessor ist mit dem zweiten Sockel **501B** verbunden.

[0076] In einer Ausführungsform ist die Doppel-Sockel-Arearchitektur **500A** als eine Sandy-Brücke-Prozessor-Architektur mit Mehrkern-Prozessoren konfiguriert. Beispielsweise ist eine Intel-Schnellpfad-Zwischenverbindung (QPI) zwischen dem ersten und dem zweiten CPU-Sockel vorgesehen, um eine Punkt-Zu-Punkt-Frontseiten-Busschnittstelle zwischen den beiden Sockeln und den CPU-Prozessoren, die mit den Sockeln verbunden sind, bereitzustellen. Auf diese Weise können die beiden CPU-Prozessoren in dem ersten CPU-Sockel **501A** und dem zweiten CPU-Sockel **501B** als eine Maschine unter einem einzelnen Betriebssystem arbeiten. D. h., ein einzelnes Betriebssystem verwaltet die CPU-Mehrkern-Prozessoren, die mit dem ersten CPU-Sockel **501A** und dem zweiten CPU-Sockel **501B** gekoppelt sind, wie dies durch die QPI-Verbindung unterstützt wird.

[0077] Ferner ist die Doppel-Sockel-Architektur **500A** mit mehreren Grafikprozessoren (beispielsweise integrierte Schaltungen, Chipsätze, etc.) versehen. Insbesondere umfasst die Architektur **500A** mehrere GPU-Leiterplatten **510A**, **510B**, ..., **510N**, die mehrere GPU-Prozessoren und/oder Chipsätze bereitstellen, wobei die GPU-Prozessoren mit dem ersten CPU-Sockel und dem zweiten CPU-Sockel verbunden sind. Jede der

GPU-Leiterplatten ist mit einem entsprechenden CPU-Sockel über eine Kommunikationsbusschnittstelle verbunden, die den ersten CPU-Sockel **501A** mit einer ersten Teilgruppe aus einer oder mehreren GPU-Leiterplatten verbindet, und der zweite CPU-Sockel **501B** ist mit einer zweiten Teilgruppe aus einer oder mehreren GPU-Leiterplatten verbunden. In einer Ausführungsform umfasst die Kommunikationsbusschnittstelle eine Kommunikationsbrücke **505A**–N, etwa eine PCI-Express-(PCIe)Brücke in einer Ausführungsform. In dieser Konfiguration arbeitet eine PCIe-Steuerung derart, dass eine Kommunikation zwischen einer entsprechenden GPU-Leiterplatte und dem CPU-Sockel über eine entsprechende PCIe-Brücke ermöglicht wird. Die Doppel-Sockel-Architektur **500A** umfasst daher mehrere Kommunikationsbrücken, wobei jede Kommunikationsbrücke kommunizierend eine entsprechende GPU-Leiterplatte mit einem entsprechenden ersten und zweiten CPU-Sockel verbindet. Wie beispielsweise in **Fig.** 5A gezeigt ist, verbinden die Brücken **505A**–D jeweils die GPU-Leiterplatten **510A**–D mit entweder dem ersten CPU-Sockel **501A** oder dem zweiten CPU-Sockel **501B**.

[0078] In einer Ausführungsform sind die mehreren GPU-Leiterplatten symmetrisch über die duale CPU-Sockel-Architektur **500A** verteilt. Beispielsweise ist eine gleiche Anzahl an GPU-Leiterplatten mit jedem CPU-Sockel verbunden oder ist an diesem angeschlossen. Wie in **Fig.** 5A gezeigt ist, umfasst die duale CPU-Sockel-Architektur **500A** vier der GPU-Leiterplatten, die mit dem ersten CPU-Sockel **501A** und dem zweiten CPU-Sockel **501B** verbunden sind. Insbesondere sind in einer Konfiguration mit vier GPU-Leiterplatten eine erste GPU-Leiterplatte **510A** und eine zweite GPU-Leiterplatte **510B** jeweils mit dem ersten CPU-Sockel **501A** verbunden, und eine dritte GPU-Leiterplatte **510C** und eine vierte GPU-Leiterplatte **510D** sind jeweils mit dem zweiten CPU-Sockel **501B** verbunden.

[0079] Alle GPU-Leiterplatten **510A**–D sind in einer Ausführungsform jeweils identisch aufgebaut. In anderen Ausführungsformen können die GPU-Leiterplatten **510A**–D unterschiedlich konfiguriert sein, wobei eine variierende Anzahl von GPU-Prozessoren in jeder GPU-Leiterplatte enthalten ist. Wie gezeigt, enthält jede GPU-Leiterplatte **510A**–D je zwei oder mehr der mehreren GPU-Prozessoren. Wie in **Fig.** 5A gezeigt ist, enthalten die GPU-Leiterplatten **510A**–D jeweils zwei der GPU-Prozessoren, obwohl eine größere Anzahl an GPU-Prozessoren in anderen Ausführungsformen verwendet ist. Zum Zwecke der Darstellung ist die GPU-Leiterplatte **510A**–D mit einem NVIDIA®-GK107-Grafikprozessorchip versehen. Jede GPU-Leiterplatte **510A**–D enthält einen Brückenteiler, der mit einer entsprechenden Kommunikationsbrücke verbunden ist. Der Brückenteiler lenkt eine eintreffende Kommunikation zu der geeigneten GPU-Leiterplatte.

[0080] **Fig.** 5B ist eine Darstellung einer unsymmetrischeren NETZ-System-Architektur **500B**, die eine grafische Verarbeitung gemäß einer Ausführungsform der vorliegenden Offenbarung bereitstellt. D. h., die mehreren GPU-Leiterplatten **550A**–C sind unsymmetrisch über eine Doppel-Sockel-Architektur **500B** verteilt. Generell ist die Doppel-Sockel-NETZ-System-Architektur **500B** in ähnlicher Weise konfiguriert wie die NETZ-System-Architektur **500A**, mit zwei CPU-Sockeln **560A** und **560B**, die jeweils kommunizierend mit einer oder mehreren GPU-Leiterplatten verbunden sind.

[0081] Beispielsweise umfasst die unsymmetrische Konfiguration eine ungleiche Anzahl an GPU-Leiterplatten, die jeweils mit dem CPU-Sockel **560A** oder **560B** verbunden sind. Wie in **Fig.** 5B gezeigt ist, enthält die duale CPU-Sockel-Architektur **500B** drei GPU-Leiterplatten, die mit dem ersten CPU-Sockel **560A** und dem zweiten CPU-Sockel **560B** verbunden sind. Genauer gesagt, in einer Konfiguration mit drei GPU-Leiterplatten sind eine erste GPU-Leiterplatte **550A** und eine zweite GPU-Leiterplatte **550B** jeweils mit dem ersten CPU-Sockel **560A** verbunden. Andererseits ist nur eine der GPU-Leiterplatte **550C** mit dem zweiten CPU-Sockel **560B** verbunden.

[0082] Alle GPU-Leiterplatten **550A**–C sind in einer Ausführungsform identisch aufgebaut. In anderen Ausführungsformen können die GPU-Leiterplatten **510** unterschiedlich ausgebildet sein, wobei eine variierende Anzahl an GPU-Prozessoren in jeder GPU-Leiterplatte enthalten ist. Wie gezeigt, enthält jede der GPU-Leiterplatte **550A**–C zwei oder mehr der mehreren GPU-Prozessoren. Wie in **Fig.** 5B gezeigt ist, beinhalten die GPU-Leiterplatten **550A**–C jeweils vier GPU-Prozessoren, obwohl auch eine andere Anzahl an GPU-Prozessoren in anderen Ausführungsformen verwendet wird. Zum Zwecke der Darstellung ist etwa die GPU-Leiterplatte **550A**–C mit einem NVIDIA®-GK107-Grafikprozessorchip versehen.

[0083] In einer Ausführungsform ermöglichen die duale bzw. Doppel-CPU-Sockel-Architektur (beispielsweise **500A**) und die mehreren GPU-Leiterplatten eine Vielzahl virtueller Maschinen (beispielsweise durch eine entsprechend basierte Struktur), wovon jede Teile eines oder mehrerer CPU-Kerne und einen oder mehrere GPU-Prozessoren enthält. In einer weiteren Ausführungsform ist die duale Sockel-Architektur **500A** so konfiguriert, dass ein einzelner GPU-Prozessor eine Instanz einer virtuellen Maschine unterstützt, entsprechend der Besetzung in der dualen Sockel-Architektur **500A**. In einer weiteren Ausführungsform werden mehrere mehrere

GPU-Prozessoren durch die mehreren physikalischen GPU-Prozessoren ermöglicht, wobei eine virtualisierte GPU eine virtuelle Maschine ermöglicht bzw. unterstützt. In diesem Falle kann eine physikalische GPU oder Teile einer physikalischen GPU eine beliebige Anzahl virtueller Maschinen ermöglichen.

[0084] In einer Ausführungsform sind die duale CPU-Sockel-Architektur und die mehreren GPU-Leiterplatten in einen Pseudo-System virtueller Maschinen implementiert, die unter einem einzelnen Betriebssystem arbeiten. D. h., eine Software-Schicht ermöglicht die Verwendung eines einzelnen Betriebssystems, um mehrere Anwendungen, etwa Spieleanwendungen, als Datenstrom zu übertragen. D. h., die Software-Schicht (beispielsweise Middleware) instantiiert eine pseudo-virtuelle Maschine, die mehrere End-Klienten unterstützen kann, wobei die pseudo-virtuelle Maschine mehrere Anwendungen für einen oder mehrere Endanwender abarbeitet und/oder ausführt.

[0085] In einer Ausführungsform ist die NUMI-Festlegungs-Software-Konfiguration der Bereitstellung des korrekten CPU-zu-GPU-Verhältnisses und zur Zuweisung von Hardware-Ressourcen wie folgt beschrieben:
CPUFestlegung bzw. Pinning = [„0,1,2,3,4,5,6,7”, „8,9,10,11,12,13,14,15”, „8,9,10,11,12,13,14,15”, „16,17,18,19,20,21,22,23”, „24,25,26,27,28,29,30,31”, „24,25,26,27,28,29,30,31”]}

[0086] In einer weiteren Ausführungsform ist die NUMI-Festlegungs-Konfiguration zur Implementierung des korrekten CPU-zu-GPU-Verhältnisses beschrieben wie folgt:
CPU Festlegung bzw. Pinning = [„0,1,2,3,4,5,6,7”, „0,1,2,3,4,5,6,7”, „8,9,10,11,12,13,14,15”, „8,9,10,11,12,13,14,15”, „16,17,18,19,20,21,22,23”, „16,17,18,19,20,21,22,23”, „24,25,26,27,28,29,30,31”, „24,25,26,27,28,29,30,31”]}

[0087] **Fig.** 6 ist eine Darstellung einer Implementierung von Verarbeitungsleiterplatten mit der Fähigkeit für mehrere GPU, um eine wolkenbasierte virtualisierte Grafikverarbeitung für Fernanzeigen bereitzustellen. Insbesondere zeigt **Fig.** 6 einen am Netzwerk angeschlossenen GPU-Apparat bzw. Gerät **620**, der eine wolkenbasierte Grafikverarbeitung für Fernanzeigen bereitstellt. In einer Ausführungsform stellt der GPU-Apparat **620** eine Grafikverarbeitung in einer PC-emulierten Umgebung bereit, und damit arbeitet der GPU-Apparat **620** als ein visueller Computerapparat (VCA).

[0088] Wie in **Fig.** 6 gezeigt ist, umfasst der VCA **620** eine oder mehrere NETZ-Systemarchitekturen oder Verarbeitungsleiterplatten **610**. Beispielsweise sind die NETZ-Architekturen **610** ähnlich in der Funktionskonfiguration wie die NETZ-Systemarchitekturen **500A** und **500B**, die allgemein in den **Fig.** 5A–B beschrieben sind. Insbesondere umfasst die NETZ-Architektur einen doppelten CPU-Sockel **615**. In einer Implementierung ist jeder CPU-Sockel ausgebildet, elektrische Verbindungen zwischen einem Mikroprozessor, der CPU-Eigenschaften bietet, und einer zu Grunde liegenden Leiterplatte bereitzustellen. Ein Mehrkern-Prozessor ist in einer Ausführungsform kommunizierend mit jedem der CPU-Sockel verbunden. Ferner umfasst die doppelte Sockel-NETZ-Architektur **610** mehrere GPU-Leiterplatten, die einen oder mehrere Grafik-Prozessoren (nicht gezeigt) enthalten, die mehrere GPU-Prozessoren und/oder Chipsätze bereitstellen, die mit dem CPU-Sockel **615** verbunden sind. D. h., jede der GPU-Leiterplatten ist mit einem entsprechenden CPU-Sockel über eine Kommunikationsbus Schnittstelle verbunden.

[0089] Bislang sind der VCA **620** und die GPU-Architektur **610** unterhalb der Linie A-A eingerichtet, die beabsichtigt ist, um die Hardware von der Software-Schicht zu trennen, wenn eine wolkenbasierte virtualisierte Grafikverarbeitung für entfernte Anzeigen implementiert wird. Somit ist über der Linie A-A eine Wolke-Rechenplattform **630** gezeigt, die mehrere virtuelle Maschinen erzeugt und verwaltet, wovon jede so gestaltet ist, dass sie eine Grafikverarbeitung mit hoher Leistung bereitstellt. Die Wolke-Rechenplattform **630** stellt die gleichen Dienste und Eigenschaften wie die Wolke-Systeme **450A**–N aus **Fig.** 4B bereit. In einer Ausführungsform dienen die virtuellen Maschinen, die in der Wolke-Rechenplattform-Schicht **630** erzeugt und verwaltet sind, als PC-Emulatoren, die Anweisungsbefehle von Endanwender-Klienten-Geräten (beispielsweise schlanke Klienten) nehmen, die Befehle verarbeiten und dann die Ergebnisse an den schlanken Klienten zurückgeben.

[0090] Die Wolke-Rechenplattform **630** kann, wie dargestellt, für die Klienten-Geräte viele Formen annehmen. Beispielsweise gibt die Schicht **640** an, dass die Wolke-Rechenplattform als ein Cluster bzw. Ansammlung von Rechenressourcen präsentiert werden kann. Beispielsweise kann der Cluster die Form eines Datencenter-Clusters oder eines virtuellen Spiele-Maschinen-Clusters annehmen.

TABELLE 1

AUFLISTUNG VON ANSPRÜCHEN

1. Eine Vorrichtung zur Bereitstellung einer Grafikverarbeitung mit: einer dualen bzw. Doppel-CPU-Sockel-Architektur mit einem ersten CPU-Sockel und einem zweiten CPU-Sockel; mehreren GPU-Leiterplatten, die mehrere GPU-Prozessoren bereitstellen, die mit dem ersten CPU-Sockel und dem zweiten CPU-Sockel verbunden sind, wobei jede GPU-Leiterplatte zwei oder mehr der mehreren GPU-Prozessoren aufweist; und einer Kommunikationsbusschnittstelle, die den ersten CPU-Sockel mit einer ersten Teilgruppe aus einer oder mehreren GPU-Leiterplatten und den zweiten CPU-Sockel mit einer zweiten Teilgruppe aus einer oder mehreren GPU-Leiterplatten verbindet.
2. Die Vorrichtung nach Anspruch 1, die ferner umfasst: einen Mehrkern-Prozessor, der mit dem ersten CPU-Sockel verbunden ist.
3. Die Vorrichtung nach Anspruch 2, wobei der Mehrkern-Prozessor einen XEON E 2670-Prozessor umfasst, der mit dem ersten CPU-Sockel verbunden ist.
4. Die Vorrichtung nach Anspruch 1, wobei die Kommunikationsbusschnittstelle umfasst: mehrere Kommunikationsbrücken, die jeweils eine entsprechende GPU-Leiterplatte mit dem ersten CPU-Sockel und dem zweiten CPU-Sockel verbinden.
5. Die Vorrichtung nach Anspruch 4, wobei mindestens eine der Kommunikationsbrücken eine PCIe-Brücke umfasst.
5. Die Vorrichtung nach Anspruch 5, wobei jede der mehreren GPU-Leiterplatten umfasst: einen Brückenteiler, der mit einer entsprechenden PCIe-Brücke verbunden ist; und zwei oder mehr der mehreren GPU-Prozessoren.
6. Die Vorrichtung nach Anspruch 5, wobei die mehreren GPU-Leiterplatten symmetrisch auf die duale GPU-Architektur verteilt sind.
7. Die Vorrichtung nach Anspruch 6, die ferner umfasst: eine erste GPU-Leiterplatte und eine zweite GPU-Leiterplatte, die jeweils mit dem ersten CPU-Sockel verbunden sind; und eine dritte GPU-Leiterplatte und eine vierte GPU-Leiterplatte, die jeweils mit dem zweiten CPU-Sockel verbunden sind; wobei jeweils die erste, die zweite, die dritte und die vierte GPU-Leiterplatte eine Zwei-GPU-Prozessor-Konfiguration umfassen.
8. Die Vorrichtung nach Anspruch 5, wobei die mehreren GPU-Leiterplatten unsymmetrisch über die duale CPU-Sockel-Architektur verteilt sind.
9. Die Vorrichtung nach Anspruch 1, die ferner umfasst: ein Frontseiten-Busnetzwerk, das den ersten CPU-Sockel mit dem zweiten CPU-Sockel verbindet;
10. Die Vorrichtung nach Anspruch 1, wobei das Frontseiten-Busnetzwerk umfasst: eine QPI-Verbindung, die den ersten CPU-Sockel kommunizierend mit dem zweiten CPU-Sockel verbindet; und ein einzelnes Betriebssystem, das CPU-Prozessoren, die mit dem ersten CPU-Sockel und dem zweiten CPU-Sockel verbunden sind, wie sie von der QPI-Verbindung unterstützt sind, verwaltet.
11. Die Vorrichtung nach Anspruch 1, wobei mindestens einer der mehreren Grafikprozessoren einen NVIDIA GK107-Grafikprozessorchip umfasst.
12. Die Vorrichtung nach Anspruch 1, wobei ein GPU-Prozessor eine virtuelle Maschine in einer 1-zu-1-Zuordnung unterstützt.
13. Die Vorrichtung nach Anspruch 1, die ferner umfasst: mehrere virtualisierte GPU-Prozessoren, die von den mehreren GPU-Prozessoren unterstützt werden, wobei eine virtualisierte GPU eine virtuelle Maschine unterstützt bzw. ermöglicht.
14. Die Vorrichtung nach Anspruch 1, wobei die duale CPU-Sockel-Architektur in einer Sandy-Brückenkonfiguration konfiguriert ist.
15. Die Vorrichtung nach Anspruch 1, wobei die mehreren GPU-Leiterplatten identisch sind.

16. Die Vorrichtung nach Anspruch 1, wobei die duale CPU-Sockel-Architektur und die mehreren GPU-Leiterplatten mehrere virtuelle Maschinen ermöglichen, wovon jede Teile eines oder mehrerer CPU-Kerne und eines oder mehrerer GPU-Prozessoren umfasst.

17. Die Vorrichtung nach Anspruch 1, wobei die duale CPU-Sockel-Architektur und die mehreren GPU-Leiterplatten in einen pseudo-virtuellen Maschinensystem implementiert sind, das unter einem einzelnen Betriebssystem arbeitet und mehrere Anwendungen für einen oder mehrere Endanwender ausführt.

18. Eine am Netzwerk angeschlossene GPU-Einrichtung mit:
mehreren Verarbeitungsleiterplatten, die mehrere virtuelle CPU-und GPU-Prozessoren bereitstellen, wobei jede der Verarbeitungsleiterplatten umfasst:
eine duale CPU-Sockel-Architektur mit einem ersten CPU-Sockel und einem zweiten CPU-Sockel;
mehreren GPU-Leiterplatten, die mehrere GPU-Prozessoren bereitstellen, die mit dem ersten CPU-Sockel und dem zweiten CPU-Sockel verbunden sind, wobei jede GPU-Leiterplatte zwei oder mehr der mehreren GPU-Prozessoren aufweist;
mehreren erste Kommunikationsbrücken, die jeweils eine entsprechende GPU-Leiterplatte mit dem ersten CPU-Sockel und dem zweiten CPU-Sockel verbinden; und
einer Kommunikationsschnittstelle, die den ersten CPU-Sockel mit einer ersten Teilgruppe aus einer oder mehreren GPU-Leiterplatten und den zweiten CPU-Sockel mit einer zweiten Teilgruppe aus einer oder mehreren GPU-Leiterplatten verbindet.

19. Die Vorrichtung nach Anspruch 18, wobei jede der mehreren GPU-Leiterplatten umfasst:
einen Brückenteiler, der mit einer entsprechenden Kommunikationsbrücke verbunden ist; und
zwei oder mehr der mehreren GPU-Prozessoren.

20. Die Vorrichtung nach Anspruch 18, wobei die mehreren GPU-Leiterplatten symmetrisch über die doppelte CPU-Sockel-Architektur verteilt sind, so dass eine erste GPU-Leiterplatte und eine zweite GPU-Leiterplatte jeweils mit dem ersten CPU-Sockel verbunden sind, und eine dritte GPU-Leiterplatte und eine vierte GPU-Leiterplatte jeweils mit dem zweiten CPU-Sockel verbunden sind, wobei jeweils die erste, die zweite, die dritte und die vierte GPU-Leiterplatte eine Zwei-GPU-Prozessor-Konfiguration aufweisen.

21. Die am Netzwerk angeschlossene GPU-Einrichtung nach Anspruch 18, die ferner umfasst:
mehrere Recheneinrichtungsemulatoren, die von den mehreren Verarbeitungsleiterplatten unterstützt werden.

Erkennung eines Schlüsselbildblocks bei Ausführung einer Anwendung in einem wolkenbasierten System, das eine virtualisierte Grafikverarbeitung für entfernte Dienstleister bereitstellt

[0091] **Fig.** 7 ist eine Blockansicht eines Systems **700**, das ausgebildet ist, einen Schlüsselbildblock gemäß einer Ausführungsform der vorliegenden Offenbarung zu erkennen. In einer Ausführungsform ist das System **700** in dem Grafiksystem **400A** aus **Fig.** 4A und/oder der Architektur **400B** aus **Fig.** 4B implementiert, die konfiguriert sind, eine wolkenbasierte virtualisierte Grafikverarbeitung für Fernanzeigen einzurichten. Die Erfassung eines vorgewählten Schlüsselbildblocks ist wichtig, um zu ermitteln, wann auf ein übertragenes Video von der Anwendung zurück zu schalten ist, nachdem ein zweites Video, etwa eine Werbung, übertragen wird.

[0092] Wie in **Fig.** 7 gezeigt ist, umfasst das System **700** einen Prozessor **701**, der zur Initialisierung einer Instanz einer Anwendung konfiguriert ist. In einer Ausführungsform ist der Prozessor **701** eine virtuelle Maschine, die von einem wolkenbasierten Grafikverarbeitungssystem unterstützt bzw. ermöglicht wird, die zum Teil eine virtualisierte Grafikbilderzeugung und Verarbeitung für entfernte Anzeigen bzw. Fernanzeigen bereitstellt. Die Anwendung, die in der virtuellen Maschine instantiiert wird, durchläuft einen Ladeprozess, um die Anwendung zu initialisieren. In einer Ausführungsform umfasst der Ladeprozess die Bestimmung der geeigneten Konfigurationseinstellung für die virtuelle Maschine, wenn die Anwendung ausgeführt wird. Die Konfigurationseinstellung kann die Ressourcenfähigkeiten der virtuellen Maschine sowie die Ressourcenfähigkeiten des End-Klienten-Geräts berücksichtigen.

[0093] Das System **700** umfasst eine Grafikbilderzeugungseinheit **705** zur Ausführung einer grafischen Bilderzeugung, um mehrere Bildblöcke zu erzeugen, die die Basis eines ersten Video-Stroms bilden. Die grafische Bilderzeugung wird durch die Ausführung der Anwendung durchgeführt, wobei der erste Video-Strom die mehreren Bildblöcke enthält.

[0094] In einer Ausführungsform umfasst das System **700** optional einen Video-Kodierer/Dekodierer **710**, der das erzeugte Video in ein komprimiertes Format kodiert, bevor der kodierte Video-Strom einer Fernanzeige

zugeleitet wird. In der vorliegenden Ausführungsform wird der kodierte Video-Bildblock verwendet, um einen Schlüsselbildblock zu erkennen.

[0095] Das System **700** umfasst ferner einen Anwendungssignaturgenerator, der ausgebildet ist, die übergeordnete Signatur zu erzeugen, die mit einem vorgewählten erzeugten Schlüsselbildblock der Anwendung verknüpft ist, der verwendet wird, um zu bestimmen, wann eine Instanz der ausgeführten Anwendung den gleichen Schlüsselbildblock erreicht hat, wie dies nachfolgend in Verbindung mit den **Fig.** 8A–B ausführlicher beschrieben ist. Insbesondere wird die Anwendungssignatur erzeugt, indem auf eine übergeordnete Bitmap bzw. Bit-Zuordnung aus einem entsprechenden Bildblock zugegriffen wird, der mit dem besagten Schlüsselbildblock aus dem übergeordneten Video-Strom verknüpft ist. In einer Ausführungsform wird die übergeordnete Bit-Zuordnung mit Prüfsumme versehen unter Anwendung eines Prüfsummenalgorithmus, um eine mit Prüfsumme versehene Bitmap bzw. Bit-Zuordnung des Schlüsselbildblocks zu erzeugen, die die Anwendungssignatur der Anwendung enthält.

[0096] Das System **700** umfasst einen Blockpuffer **725** zum Empfang einer Sequenz mehrerer Bildblöcke, die zu dem ersten Video-Strom gehören. In einer Ausführungsform wird die grafische Bilderzeugung von der virtuellen Maschine in dem wolkenbasierten Grafikbilderzeugungssystem ausgeführt, wobei der Video-Strom des erzeugten Videos dann einer Fernanzeige zugeleitet wird. Der Blockpuffer umfasst einen oder mehrere Blockpuffer, die ausgebildet sind, die erzeugten Video-Bildblöcke zu empfangen. Beispielsweise kann eine Grafik-Pipeline ihr erzeugtes Video an einen entsprechenden Blockpuffer ausgeben. In einem parallelen System gibt jede Pipeline eines Grafikprozessors mit mehreren Pipelines ihr erzeugtes Video an einen entsprechenden Blockpuffer aus.

[0097] Das System **700** umfasst einen Komparator **730**, der ausgebildet ist zu ermitteln, wann eine erste Bitmap bzw. Bit-Zuordnung eines Bildblocks in einen entsprechenden Blockpuffer geladen wird und eine Anwendungssignatur übereinstimmt mit einem Abkömmling einer übergeordneten Bit-Zuordnung, die mit einem Schlüsselbildblock des ersten Video-Stroms verknüpft ist.

[0098] **Fig.** 8A–B zeigen zusammen ein Verfahren zum Ausführen einer Anwendung und zum Ermitteln, wann ein vorbestimmter Schlüsselbildblock bei dieser Ausführung erzeugt wird. Die Erkennung des Schlüsselbildblocks ist aus vielen Gründen wichtig, wobei einer der Gründe ist, dass verstanden werden muss, wann zwischen dem ersten Video-Strom, der von der Ausführung der Anwendung erzeugt wird, und einem zweiten Video-Strom, der andere Information bietet, die möglicherweise keine Beziehung zu der Anwendung hat (beispielsweise Werbung, etc.) umzuschalten ist.

[0099] Insbesondere ist **Fig.** 8A ein Flussdiagramm **800A**, das ein Verfahren zum Erkennen eines Schlüsselbildblocks während der Ausführung einer Anwendung in einer virtuellen Maschine zeigt, die von einer Wolke-Rechenplattform unterstützt wird, die eine virtualisierte Grafikverarbeitung für Fernanzeigen gemäß einer Ausführungsform der vorliegenden Offenbarung bereitstellt. In einer noch weiteren Ausführungsform zeigt das Flussdiagramm **800A** ein computerimplementiertes Verfahren zur Erfassung eines Schlüsselbildblocks während der Ausführung einer Anwendung in einer virtuellen Maschine, die von einer Wolke-Rechenplattform unterstützt ist, die wiederum eine virtualisierte Grafikverarbeitung für Fernanzeigen bereitstellt. In einer weiteren Ausführungsform ist das Flussdiagramm **800A** in einem Computersystem implementiert mit einem Prozessor und einem mit dem Prozessor verbundenen Speicher, der darin gespeichert Befehle enthält, die, wenn sie von dem Computersystem ausgeführt werden, das System veranlassen, ein Verfahren zur Erkennung eines Schlüsselbildblocks während der Ausführung einer Anwendung in einer virtuellen Maschine auszuführen, die von einer Wolke-Rechenplattform unterstützt wird, die wiederum virtualisierte Grafikverarbeitung für Fernanzeigen bereitstellt. In einer noch weiteren Ausführungsformen sind Befehle zur Ausführung eines Verfahrens, wie es in dem Flussdiagramm **800A** dargestellt ist, auf einem nicht-flüchtigen computerlesbaren Speichermedium gespeichert mit von einem Computer ausführbaren Befehlen, um ein Computersystem zu veranlassen, ein Verfahren zur Erfassung eines Schlüsselbildblocks während der Ausführung einer Anwendung in einer virtuellen Maschine auszuführen, die von einer Wolke-Rechenplattform unterstützt ist, die wiederum eine virtualisierte Grafikverarbeitung für Fernanzeigen bereitstellt. In Ausführungsformen ist das im Flussdiagramm **800A** dargestellte Verfahren durch eine oder mehrere Komponenten des Computersystems **100** oder des Klienten-Geräts **200** aus den **Fig.** 1 und **Fig.** 2 sowie des Systems **700** aus **Fig.** 7 implementierbar.

[0100] Bei **850** beinhaltet das Verfahren die Initialisierung einer Instanz einer Anwendung. Wenn beispielsweise eine virtuelle Maschine innerhalb einer wolkenbasierten Plattform für virtuelle Grafikverarbeitung instantiiert wird auf Anforderung eines Endanwenders hin, wird eine Anwendung (beispielsweise eine Spieleanwendung) instantiiert. D. h., der Endanwender gibt an, dass er in der Anwendung spielen will, und die wolkenbasierte

Plattform für virtuelle Grafikverarbeitung erzeugt eine virtuelle Maschine zur Ausführung der Anwendung. Die Initialisierung der Anwendung führt zum Laden der Anwendung in die virtuelle Maschine und kann eine beliebige Sequenz an Operationen beinhalten, die einen Ausgabe-Video-Strom erzeugen können oder auch nicht. Beispielsweise zeigen einige Anwendungen ein sich drehendes Stundenglas über einem leeren oder abgedunkelten Bildschirm, während die Anwendung geladen wird.

[0101] In einer Ausführungsform überträgt die wolkenbasierte Plattform für virtuelle Grafikverarbeitung einen zweiten Video-Strom zu dem Endanwender, während die Anwendung geladen wird. Der zweite Video-Strom ist nicht Teil der Anwendung. Beispielsweise enthält der zweite Video-Strom zusätzliche Information, etwa Werbung, Benachrichtigungen, etc.. An einem vorbestimmten Zeitpunkt unterbricht die grafische Verarbeitungsplattform die Übertragung des zweiten Video-Stroms und schaltet zurück zu dem Video-Strom, der von der Anwendung erzeugt wird. Ausführungsformen der vorliegenden Erfindung ermöglichen die Erkennung eines speziellen und vorbestimmten Schlüsselbildblocks, der angibt, wann auf den Video-Strom zurück zu schalten ist, der von der Anwendung erzeugt wird.

[0102] Bei **855** beinhaltet das Verfahren die Ausführung einer grafischen Bilderzeugung in mehreren erzeugten Bildblöcken durch die Ausführung der Anwendung. Ein erster Video-Strom, der erzeugt wird, enthält die mehreren erzeugten Bildblöcke. **Fig.** 8B ist ein Informationsflussdiagramm **800B**, das den Prozess zum Erkennen eines Schlüsselbildblocks gemäß einer Ausführungsform der vorliegenden Erfindung zeigt. Wie gezeigt, beinhaltet die Ausführung einer Anwendung in einem beliebigen Prozessor, einschließlich eines Prozessors einer virtuellen Maschine, die Sendung von Befehlen zu einer virtuellen GPU **415**, wie dies zuvor in Verbindung mit den virtuellen GUP **415A**–M aus **Fig.** 4B beschrieben ist. Beispielsweise kann die GPU **415** eine beliebige der virtuellen GPU **415A**–M sein. Die GPU **415** gibt erzeugte Bildblöcke in geordneter Reihenfolge aus, wie sie in einer oder mehreren Grafik-Pipelines ausgeführt werden. Beispielsweise erzeugt die GPU **4151**–N Bildblöcke.

[0103] Bei **865** beinhaltet das Verfahren die Ermittelung, wann eine erste Bitmap bzw. Bit-Zuordnung eines Bildblocks, der in einen entsprechenden Blockpuffer geladen wird, mit einer Anwendungssignatur übereinstimmt. Wie in **Fig.** 8B gezeigt ist, führt ein Schattierungs-Modul den Vergleich zwischen der Bit-Zuordnung eines erzeugten Bildblocks aus der ausgeführten Anwendung und einer übergeordneten Bitmap bzw. Bit-Zuordnung aus, die die Anwendungssignatur **820** bildet. Sobald beispielsweise ein erzeugten Bildblock (beispielsweise Bildblock 1) von der GPU **415** erzeugt ist, und eine entsprechende Bitmap bzw. Bit-Zuordnung in einen entsprechenden Blockpuffer geladen ist, wird der erzeugte Bildblock oder seine zugeordnete Bit-Zuordnung in die Schattierungseinheit **830A** geladen und wird mit einer Anwendungssignatur **820** verglichen, die speziell für diese Anwendung ist und ebenfalls in die Schattierungseinheit **830A** geladen wird. Die Schattierungseinheit **830A** führt den Vergleich zwischen der Anwendungssignatur **820** und der zugehörigen Bitmap-Information, die in den Blockpuffer 1 geladen ist, durch.

[0104] Die Anwendungssignatur wird vorläufig in einer Ausführungsform aus einer übergeordneten Kopie bzw. Vorlagenkopie der Anwendung erzeugt. Insbesondere enthält in einer Ausführungsform die Anwendungssignatur einen Abkömmling einer Vorlagen-Bitmap, die mit einem Schlüsselbildblock des ersten Video-Stroms assoziiert ist. D. h., es wird eine Vorlagen-Anwendung ausgeführt und enthält mehrere Vorlagen-Bildblöcke, die erzeugt werden. Einer der erzeugten Bildblöcke wird aus einem Vorlagen-Video-Strom als der Schlüsselbildblock ausgewählt. Beispielsweise kann der Schlüsselbildblock eine spezielle Szene oder ein spezielles Logo sein, das in einem geeigneten Moment in der Ausführung, Initialisierung und/oder Ladung einer Instantiierung der Anwendung erzeugt wird. Die Vorlagen-Bitmap des Schlüsselbildblocks, die in einen Blockpuffer zum Anzeigen geladen wird, wird dann bearbeitet, um einen Abkömmling des als Vorlage dienenden Schlüsselbildblocks zu erzeugen und bildet die Anwendungssignatur. Beispielsweise beinhaltet die Erzeugung der Anwendungssignatur den Zugriff auf eine Vorlagen-Bitmap und die Ausführung eines Prüfsummen-Algorithmus an der Vorlagen-Bitmap, um eine Schlüsselbildblock-Bitmap mit Prüfsumme zu erzeugen. Die Schlüsselbildblock-Bitmap mit Prüfsummen enthält nunmehr die Anwendungssignatur **820** der Anwendung. In einer nachfolgenden Instantiierung der Anwendung mit geeigneter Bearbeitung des erzeugten Bildblocks (beispielsweise Ausführung einer Prüfsummen) sind Ausführungsformen der vorliegenden Erfindung in der Lage zu ermitteln, wann ein erzeugter Bildblock, ob modifiziert oder nicht modifiziert, mit dem Schlüsselbildblock oder einem Abkömmling des Schlüsselbildblocks übereinstimmt.

[0105] Wenn ermittelt wird, dass es eine Übereinstimmung zwischen dem erzeugten Bildblock und dem Vorlagen-Bildblock gibt, beinhaltet das Verfahren den Zugriff auf eine erste Bit-Zuordnung des Blocks, der in den entsprechenden Blockpuffer geladen wird. Diese erste Bit-Zuordnung bzw. Bitmap wird während der Ausführung der Anwendung erzeugt, die beispielsweise in einer entsprechenden virtuellen Maschine als Instanz auf-

gerufen wird. Welche Bearbeitung auch immer an dem Vorlagen-Bildblock oder der Bitmap des Vorlagen-Bildblocks durchgeführt wurde, wird auch an der ersten Bitmap ausgeführt. Wenn beispielsweise die Vorlagen-Schlüsselbildblock-Bitmap mit Prüfsummen versehen wurde, um die Anwendungssignatur zu erzeugen, dann beinhaltet das Verfahren die Ausführung eines Prüfsummen-Algorithmus an der ersten Bitmap, um eine erste Bitmap mit Prüfsumme zu erzeugen. Selbstverständlich können statt der Ausführung eines Prüfsummen-Algorithmus auch andere Bearbeitungen in anderen Ausführungsformen ausgeführt werden.

[0106] Wenn das Beispiel fortgesetzt wird, in dem die Vorlagen-Schlüsselbildblock-Bitmap mit Prüfsummen versehen wird, um eine Anwendungssignatur zu erzeugen, beinhaltet das Verfahren den Vergleich der mit Prüfsumme versehenen Schlüsselbildblock-Bitmap (beispielsweise Anwendungssignatur) mit der mit Prüfsumme versehenen ersten Bitmap, um zu bestimmen, ob es eine Übereinstimmung gibt. In einer Ausführungsform gibt es eine Toleranz im Prüfsummen-Algorithmus dahingehend, dass die erste Bitmap und die Vorlagen Bitmap des Schlüsselbildblocks innerhalb einer Toleranz liegen können. Solange sie innerhalb der Toleranz sind (beispielsweise eine Übereinstimmung mit 95%), stimmen die nachfolgenden Prüfsummen, die aus der Vorlagen-Bitmap und der ersten Bitmap erzeugt werden, überein.

[0107] Das Verfahren beinhaltet die Bestimmung, wann die Anwendung bei Ausführung den Schlüsselbildblock erreicht. Dies geschieht, wenn ermittelt wird, dass der mit Prüfsumme versehene Schlüsselbildblock und die mit Prüfsumme versehene erste Bitmap übereinstimmen, wie dies zuvor beschrieben ist. In einer Ausführungsform wird die Bestimmung, wann die Anwendung den Schlüsselbildblock erreicht, verwendet, um zwischen Video-strömen umzuschalten. In diesem Falle wird, wenn eine Übereinstimmung ermittelt wird, eine Benachrichtigung über die Übereinstimmung aufwärts im Software-Stapel ausgegeben, der für die Zuleitung eines kodierten Videos zu dem Klienten-Gerät verantwortlich ist, so dass eine Umschaltung ausgeführt werden kann. Während beispielsweise die Anwendung geladen wird, beinhaltet das Verfahren das Senden eines zweiten Video-Stroms zu einem Klienten-Gerät des anfordernden Endanwenders, um angezeigt zu werden. Der zweite Video-Strom enthält Information, die nicht mit der Anwendung in Beziehung steht, und/oder die unabhängig von der Anwendung erzeugt ist. In einem Beispiel enthält der zweite Video-Strom eine Werbung.

[0108] Wenn daher die ausgeführte Anwendung einen Punkt erreicht, an welchem erzeugte Bildblöcke mit dem Schlüsselbildblock übereinstimmen, gibt dies an, dass die Anwendung, die instantiiert wird, ebenfalls den Schlüsselbildblock erreicht hat. Auf diese Weise beinhaltet das Verfahren das Umschalten auf den ersten Video-Strom bei Erkennung des Schlüsselbildblocks, wobei der erste Video-Strom die mehreren erzeugten Videos (kodiert oder nicht kodiert) repräsentiert, die von der Anwendung erzeugt wurden. Das Verfahren beinhaltet auch das Senden des ersten Video-Stroms zu dem Klienten-Gerät und die Unterbrechung der Zuleitung des zweiten Video-Stroms (beispielsweise Werbung). In einer Ausführungsform wird der erste Video-Strom beginnend mit dem Bildblock, der mit dem Schlüsselbildblock übereinstimmt, an das Klienten-Gerät zum Anzeigen geleitet. In einer weiteren Ausführungsform wird der erste Video-Strom beginnend mit einem Bildblock nach dem Bildblock, der mit dem Schlüsselbildblock übereinstimmt, zu dem Klienten-Gerät zum Anzeigen übertragen.

TABELLE 2

AUFLISTUNG VON ANSPRÜCHEN

1. Ein nicht-flüchtiges computerlesbares Medium mit von einem Computer ausführbaren Befehlen, um ein Computersystem zu veranlassen, ein Verfahren zum Umschalten auszuführen mit: Initialisieren einer Instantiierung bzw. Instanz einer Anwendung; Ausführen einer grafischen Bilderzeugung, um mehrere erzeugte Bildblöcke durch Ausführung der Anwendung zu erzeugen, um einen ersten Video-Strom zu erzeugen, der die mehreren erzeugten Bildblöcke enthält; sequenzielles Laden der mehreren erzeugten Bildblöcke in einen oder mehrere Blockpuffer; und Ermitteln, wann eine erste Bitmap eines Bildblocks, der in einen entsprechenden Blockpuffer geladen wird, mit einer Anwendungssignatur übereinstimmt, die einen Abkömmling einer Vorlagen-Bitmap enthält, die mit einem Schlüsselbildblock des ersten Video-Stroms verknüpft ist.
2. Das computerlesbare Medium nach Anspruch 1, wobei die Initialisieren einer Instantiierung einer Anwendung in dem Verfahren umfasst: Initialisieren der Instantiierung der Anwendung in einer virtuellen Maschine eines wolkenbasierten grafischen Verarbeitungssystems für einen Endanwender.

3. Das computerlesbare Medium nach Anspruch 1, wobei in dem Verfahren die Anwendung eine Spieleanwendung umfasst.

4. Das computerlesbare Medium nach Anspruch 1, wobei das Verfahren ferner umfasst:
Senden eines zweiten Video-Stroms zu einem Klienten-Gerät des Endanwenders zum Anzeigen;
Umschalten auf den ersten Video-Strom bei Erkennung des Schlüsselbildblocks; und
Senden des ersten Video-Stroms beginnend mit dem Bildblock, der mit dem Schlüsselbildblock übereinstimmt, zu dem Klienten-Gerät zum Anzeigen.

5. Das computerlesbare Medium nach Anspruch 4, wobei in dem Verfahren der zweite Video-Strom eine Werbung enthält.

6. Das computerlesbare Medium nach Anspruch 1, wobei das Verfahren ferner umfasst:
Senden eines zweiten Video-Stroms zu einem Klienten-Gerät des Endanwenders für die Anzeige;
Umschalten auf den ersten Video-Strom bei Erkennung des Schlüsselbildblocks; und
Senden des ersten Video-Stroms beginnend mit einem Bildblock nach einem Bildblock, der mit dem Schlüsselbildblock verknüpft ist, an das Klienten-Gerät für die Anzeige.

7. Das computerlesbare Medium nach Anspruch 1, wobei das Verfahren ferner umfasst:
Auswählen des Schlüsselbildblocks, der aus einem Vorlagen-Video-Strom einer Vorlagenkopie der Anwendung genommen ist, zum Zwecke des Erzeugens der Anwendungssignatur.

8. Das computerlesbare Medien nach Anspruch 7, wobei die Bestimmung, wann eine erste Bitmap in dem Verfahren ferner umfasst:
Zugreifen auf eine Vorlagen-Bitmap von einem entsprechenden Bildblock, der mit dem Schlüsselbildblock von dem Vorlagen-Video-Strom verknüpft ist; und
Ausführen eines Prüfsummen-Algorithmus an der Vorlagen-Bitmap, um eine mit Prüfsumme versehene Schlüsselbildblock-Bitmap zu erzeugen, die die Anwendungssignatur der Anwendung enthält.

9. Das computerlesbare Medium nach Anspruch 8, wobei das Verfahren ferner umfasst:
Zugreifen auf eine erste Bitmap des Bildblocks, die in einen entsprechenden Blockpuffer geladen wird;
Ausführen des Prüfsummen-Algorithmus an der ersten Bitmap, um eine mit Prüfsumme versehene erste Bitmap zu erzeugen;
Vergleichen des mit Prüfsumme versehenen Schlüsselbildblocks und der mit Prüfsumme versehenen ersten Bitmap, um zu bestimmen, ob diese übereinstimmen; und
Ermitteln, wann die Anwendung bei Ausführung den Schlüsselbildblock erreicht, wenn der mit Prüfsumme versehene Schlüsselbildblock und die mit Prüfsumme versehene erste Bitmap übereinstimmen.

10. Ein Computersystem mit:
einem Prozessor; und
einem mit dem Prozessor verbundenen Speicher, in welchem Befehle gespeichert sind, die, wenn sie von einem Computersystem ausgeführt werden, das Computersystem veranlassen, ein Verfahren zum Umschalten auszuführen, mit:
Initialisieren einer Instantiierung einer Anwendung;
Ausführen einer grafischen Bilderzeugung, um mehrere erzeugte Bildblöcke durch Ausführung der Anwendung zu erzeugen, um einen ersten Video-Strom zu erzeugen, der die mehreren Bildblöcke enthält;
sequenzielles Laden der mehreren erzeugten Bildblöcke in einen oder mehrere Blockpuffer; und
Ermitteln, wann eine erste Bitmap eines erzeugten Bildblocks, die in einen entsprechenden Blockpuffer geladen wird, mit einer Anwendungssignatur übereinstimmt, die einen Abkömmling einer Vorlagen-Bitmap enthält, die mit einem Schlüsselbildblock des ersten Video-Stroms verknüpft ist.

11. Das Computersystem nach Anspruch 10, wobei die Initialisierung einer Instantiierung einer Anwendung in dem Verfahren umfasst:
Initialisieren der Instantiierung der Anwendung in einer virtuellen Maschine eines wolkenbasierten grafischen Verarbeitungssystems für einen Endanwender.

12. Das Computersystem nach Anspruch 10, wobei in dem Verfahren die Anwendung eine Spieleanwendung umfasst.

13. Das Computersystem nach Anspruch 10, wobei das Verfahren ferner umfasst:
Senden eines zweiten Video-Stroms zu einem Klienten-Gerät des Endanwenders für die Anzeige;
Umschalten auf den ersten Video-Strom bei Erkennung des Schlüsselbildblocks; und
Senden des ersten Video-Stroms beginnend mit dem Bildblock, der mit dem Schlüsselbildblock übereinstimmt, zu dem Klienten-Gerät für die Anzeige.

14. Das Computersystem nach Anspruch 13, wobei in dem Verfahren der zweite Video-Strom eine Werbung enthält.

15. Das Computersystem nach Anspruch 10, wobei das Verfahren ferner umfasst:
Senden eines zweiten Video-Stroms zu einem Klienten-Gerät des Endanwenders zur Anzeige;
Umschalten auf den ersten Video-Strom bei Erkennung des Schlüsselbildblocks; und
Senden des ersten Video-Stroms beginnend mit einem Bildblock nach einem Bildblock, der zu dem Schlüsselbildblock gehört, zu dem Klienten-Gerät zur Anzeige.

16. Das Computersystem nach Anspruch 10, wobei das Verfahren ferner umfasst:
Auswählen des Schlüsselbildblocks, der aus einem Vorlagen-Video-Strom einer Vorlagenkopie der Anwendung zum Zwecke der Erzeugung der Anwendungssignatur genommen ist.

17. Das Computersystem nach Anspruch 16, wobei die Bestimmung, wann eine erste Bitmap bzw. Bit-Zuordnung in dem Verfahren ferner umfasst:
Zugreifen auf eine Vorlagen-Bitmap von einem entsprechenden Bildblock, der mit dem Schlüsselbildblock aus dem Vorlagen-Video-Strom verknüpft ist; und
Ausführen eines Prüfsummen-Algorithmus an der Vorlagen-Bitmap, um eine mit Prüfsumme versehene Schlüsselbildblock-Bitmap zu erzeugen, die die Anwendungssignatur der Anwendung enthält.

18. Das Computersystem nach Anspruch 17, wobei das Verfahren ferner umfasst:
Zugreifen auf eine erste Bitmap des Bildblocks, der in einen entsprechenden Blockpuffer gespeichert wird;
Ausführen des Prüfsummen-Algorithmus an der ersten Bitmap, um eine mit Prüfsumme versehene erste Bitmap bzw. Bit-Zuordnung zu erzeugen;
Vergleichen des mit Prüfsumme versehene Schlüsselbildblocks mit der mit Prüfsumme versehene ersten Bitmap bzw. Bit-Zuordnung, um zu bestimmen, ob sie übereinstimmen; und
Ermitteln, wann die Anwendung bei Ausführung den Schlüsselbildblock erreicht, wenn der mit Prüfsumme versehene Schlüsselbildblock und die mit Prüfsumme versehene Bitmap bzw. Bit-Zuordnung übereinstimmen.

19. Ein System zum Umschalten mit:
einem Prozessor, der ausgebildet ist, eine Instantiierung einer Anwendung zu initialisieren;
einer Grafikerzeugungseinheit, die ausgebildet ist, eine Grafikerzeugung an mehreren Bildblöcke auszuführen, um einen ersten Video-Strom durch Ausführung der Anwendung zu erzeugen, wobei der erste Video-Strom die mehreren Bildblöcke enthält;
einem Blockpuffer, der für den Empfang einer Sequenz mehrerer Bildblöcke, die zu dem ersten Video-Strom gehören, ausgebildet ist;
einem Komparator, der ausgebildet ist, um zu bestimmen, wann eine erste Bitmap bzw. eine Bitzuordnung eines Bildblocks in einen entsprechenden Blockpuffer geladen wird und mit einer Anwendungssignatur übereinstimmt, die einen Abkömmling einer Vorlagen-Bitmap bzw. Bit-Zuordnung enthält, die zu einem Schlüsselbildblock des ersten Video-Stroms gehört.

20. Das System nach Anspruch 19, das ferner umfasst:
ein wolkenbasierte Grafikverarbeitungssystem;
eine virtuelle Maschine des wolkenbasierten Grafikverarbeitungssystem, wobei die virtuelle Maschine zur Ausführung der Anwendung konfiguriert ist.

Schnelles Klonen virtueller Maschinen

[0109] Fig. 9 ist eine Blockansicht eines Ressourcenerzeugungssystems **900** gemäß einer Ausführungsform der vorliegenden Offenbarung, das ausgebildet ist, neue virtuelle Maschinen unter Anwendung einer Schablone einer virtuellen Maschine zu erzeugen und jede Instanz eine virtuelle Maschine mit anwenderspezifischen Daten kundenspezifisch zuzuschneiden. Das System **900** ist in der Wolke-Architektur **400B** aus **Fig.** 4B implementierbar und wird verwendet, neue virtuelle Maschinen zu erzeugen. In einer Ausführungsform ist das Ressourcenerzeugungssystem **900** innerhalb des VM Host-Verwalters **462** des GM-Dienstleisters **461A** konfiguriert.

[0110] Wie gezeigt, enthält das System **901** ein schnelles Klon-Modul **910**, das ausgebildet ist, eine neue Instanz einer virtuellen Maschine zu erzeugen. Insbesondere erzeugt das schnelle Klon-Modul **910** eine neue virtuelle Maschine aus einer Schablone einer virtuellen Maschine. Die neue virtuelle Maschine mit spezifisch auf einen speziellen Kunden zugeschnitten, indem die Schablone mit Aktualisierungen modifiziert wird. Auf diese Weise müssen anstelle der Speicherung eines gesamten Abbilds der kundenspezifischen virtuellen Maschine lediglich die Aktualisierungen, die kundenspezifisch für den anfordernden Anwender sind, gespeichert

werden. Die Aktualisierungen werden dann verwendet, um die Schablone zum Erzeugen einer neuen Instanz einer virtuellen Maschine zu modifizieren, die für den Endanwender kundenspezifisch ausgelegt ist.

[0111] Das System **900** umfasst ferner eine Anwendungslade-Einheit **920**. In einer Ausführungsform ist ein Endanwender, der auf einen wolkenbasierten Dienst zugreift, der virtualisierte Grafikverarbeitung für entfernte Dienstleister bereitstellt, an der Ausführung einer Anwendung interessiert. Die virtuelle Maschine wird instantiiert, um diese Anwendung auszuführen. Beispielsweise kann eine wolkenbasierte grafische Verarbeitungsplattform geschaffen werden, um ein wolkenbasiertes Spielerlebnis bereitzustellen, wobei Spieleanwendungen für die wolkenbasierte Plattform gespeichert und ausgeführt werden. Endanwender würden typischerweise anfordern, eine spezielle Spieleanwendung zu spielen, und es wird eine virtuelle Maschine instantiiert, um diese Anforderung zum Spielen der Spieleanwendung abzuarbeiten. Daher wird in Reaktion auf eine Anforderung zum Spielen eine Spieleanwendung durch einen Endanwender eine virtuelle Maschine instantiiert und die Spieleanwendung wird von der Anwendungslade-Einheit zur Ausführung in der virtuellen Maschine geladen.

[0112] Das System **900** umfasst ferner einen unabhängigen und permanenten Speicher **930**. Insbesondere werden die Aktualisierungen und/oder Informationen, die den Endanwender betreffen, im Speicher **930** gespeichert. Diese Information wird dann verwendet, um eine Schablone einer virtuellen Maschine kundenspezifisch auszulegen, wenn eine Instanz einer virtuellen Maschine, die kundenspezifisch für einen speziellen Endanwender ausgelegt ist, erzeugt wird. Beispielsweise kann die Information eine Information über das Anwenderprofile und Spielsicherungsdateien oder Anwendungssicherungsdateien enthalten, wobei die Anwendungssicherungsdateien Information bereitstellen, die mit den Interaktionen des Anwenders mit der speziellen Anwendung in Beziehung stehen (beispielsweise Spielstatus, etc.). Daher müssen durch die Speicherung der Aktualisierungen und/oder der Anwender betreffenden Information eine Instanz oder ein Abbild einer virtuellen Maschine und eine Anwendung, die speziell für den Anwender zugeschnitten sind, nicht dauerhaft sein, und die Instanz der virtuellen Maschine kann gelöscht werden. Dies bietet zusätzlichen Vorteil, da jede neue Instanz einer virtuellen Maschine ausgehend von einer ursprünglichen Schablone einer virtuellen Maschine startet, die frei von Fehlern wie Virus ist, und sie muss lediglich mit Information aktualisiert werden, die einen anfordernden Endanwender betrifft (beispielsweise Aktualisierungen und/oder Information über das Anwenderprofile).

[0113] Das System **900** umfasst ferner einen Bereitstellung/Zuweisungsverwalter bzw. eine Verwaltungseinheit **940**. In einer Ausführungsform führt der Verwalter **940** ähnliche Funktionen aus wie der Bereitstellungsverwalter **470** aus **Fig.** 4B. Insbesondere ist der Bereitstellung/Zuweisungsverwalter **940** ausgebildet, einer neu instantiierten virtuellen Maschine Ressourcen zuzuweisen. In einer Ausführungsform umfasst der Bereitstellung/Zuweisungsverwalter **940** eine Ressourcendrosselungseinheit **945**, die ausgebildet ist, die Menge an Ressourcen, die ursprünglich der virtuellen Maschine bei der Initialisierung zugewiesen sind, zu reduzieren und/oder zu drosseln, um negative Auswirkungen auf bestehende und funktionsbereite virtuelle Maschinen zu verringern. Nach der Initialisierung ist das Ressourcen-Ausrichtungsmodul **947** ausgebildet, die Ressourcen in der virtuellen Maschine zu erhöhen, bis die ursprüngliche Zuweisung der Ressourcen erreicht ist.

[0114] **Fig.** 10A ist ein Flussdiagramm **1000A**, das ein Verfahren zur Erzeugung von Ressourcen für eine neue Instanz einer virtuellen Maschine zeigt, die eine wolkenbasierte virtualisierte Grafikverarbeitung für eine Fernanzeige gemäß einer Ausführungsform der vorliegenden Offenbarung bereitstellt. In einer noch weiteren Ausführungsform zeigt das Flussdiagramm **1000A** ein computerimplementiertes Verfahren zur Erzeugung von Ressourcen für eine neue Instanz einer virtuellen Maschine, die eine wolkenbasierten virtualisierte Grafikverarbeitung für eine Fernanzeige bereitstellt. In einer weiteren Ausführungsform ist das Flussdiagramm **1000A** in einem Computersystem implementiert, das einen Prozessor und einen mit dem Prozessor verbundenen Speicher hat, der darin gespeichert Befehle enthält, die, wenn sie von dem Computersystem ausgeführt werden, das System veranlassen, ein Verfahren zur Erzeugung von Ressourcen für eine neue Instanz einer virtuellen Maschine auszuführen, die eine wolkenbasierte virtualisierte Grafikverarbeitung für eine Fernanzeige bereitstellt. In einer noch weiteren Ausführungsform sind Befehle zur Ausführung eines Verfahrens, wie es in dem Flussdiagramm **1000A** dargestellt ist, auf einem nicht-flüchtigen computerlesbaren Speichermedium gespeichert, das von einem Computer ausführbare Befehle enthält, um ein Computersystem zu veranlassen, ein Verfahren zur Ressourcenerzeugung für eine neue Instanz einer virtuellen Maschine auszuführen, die eine wolkenbasierte virtualisierte Grafikverarbeitung für eine Fernanzeige bereitstellt. In Ausführungsformen ist das in dem Flussdiagramm **1000A** gezeigte Verfahren durch eine oder mehrere Komponenten des Computersystems **100** oder des Klienten-Geräts **200** aus den **Fig.** 1 und **Fig.** 2 sowie durch das Ressourcen-Erzeugungssystem **900** aus **Fig.** 9 implementierbar.

[0115] Bei **1010** beinhaltet das Verfahren die Erzeugung einer Schablone einer virtuellen Maschine. Die Schablone wird verwendet, um weitere virtuelle Maschinen zu klonen. Bei **1015** beinhaltet das Verfahren die Erzeu-

gung einer Instantiierung bzw. Instanz einer virtuellen Maschine für einen anfordernden Endanwender durch Klonen der Schablone der virtuellen Maschine. Insbesondere ist der Klonvorgang dahingehend effizient, dass in einer kundenspezifischen Instanz eine virtuelle Maschine, die einen Endanwender betrifft, lediglich Modifizierungen an der Schablone und nicht das gesamte Abbild der virtuellen Maschine gespeichert werden müssen. Die Aktualisierungsinformation wird dann verwendet, um die Schablone zu modifizieren, um eine Instanz der virtuellen Maschine zu erzeugen, die für den Endanwender kundenspezifisch ist. In einer Ausführungsform enthält die Aktualisierungsinformation eine Information über das Anwenderprofil und eine Anwendungssicherungsdatei, wie dies zuvor beschrieben ist. Beispielsweise enthalten in einer Spieleanwendung die Daten der „Speichere Spiel-" Datei Information, die den Fortschritt eines Spielers innerhalb des Spieles, wenn es von der Spieleanwendung ausgeführt wird, betrifft.

[0116] In einer Ausführungsform wird die virtuelle Maschine in einem wolkenbasierten Grafikverarbeitungssystem für den anfordernden Endanwender initialisiert. Beispielsweise kann die virtuelle Maschine innerhalb der Architektur **400B** aus **Fig.** 4B instantiiert werden. Ferner beinhaltet in einer wolkenbasierten Spieleplattform, die eine virtualisierte Grafikverarbeitung für Fernanzeigen bietet, die Schablone einer virtuellen Maschine mehrere im Voraus ausgewählte Spieleanwendungen, die für jeden von mehreren Endanwendern verfügbar sind. D. h., die Schablone enthält bereits vollständige Kopien jeder Spieleanwendung. Auf diese Weise braucht der Anwender lediglich auf die Plattform zuzugreifen und eine unterstützte Spieleanwendung anzufordern, und er kann mit dem Spielen dieser Spieleanwendung über eine entsprechende virtuelle Maschine beginnen.

[0117] Bei **1020** beinhaltet das Verfahren das Laden einer Anwendung, die von der virtuellen Maschine ausgeführt wird. Beispielsweise kann der Endanwender auf die Plattform zugreifen, um eine spezielle Anwendung zu verwenden, etwa eine Video-Erzeugungs-Software. In der Spieleumgebung greift der Endanwender typischerweise auf die wolkenbasierte Plattform zu, um eine Spieleanwendung zu spielen. Nach der Instantiierung der virtuellen Maschine wird daher die spezielle Anwendung auf Anforderung des Endanwenders in die virtuelle Maschine geladen und dann ausgeführt.

[0118] Bei **1025** beinhaltet das Verfahren den Zugriff auf eine erste Information, die mit dem Endanwender verknüpft ist. Diese erste Information wird verwendet, um die virtuelle Maschine kundenspezifisch zuzuschneiden und die Instantiierung der Spieleanwendung wird innerhalb der virtuellen Maschine ausgeführt. In einer Ausführungsform wird die erste Information verwendet, um die Schablone einer virtuellen Maschine zu modifizieren, um damit diese für den Endanwender kundenspezifisch auszulegen. Beispielsweise enthält, wie zuvor beschrieben, die erste Information eine Information für die „Speicherung des Spiels" und/oder eine Information des Anwenderprofils.

[0119] Bei **1030** beinhaltet das Verfahren das Laden der ersten Information in eine Instanz der Anwendung. Beispielsweise wird die Information für die „Speicherung des Spiels", die mit der Anwendung verknüpft ist, abgerufen und verwendet, um den Anwender auf seinen oder ihren letzten aktualisierten Stand einer Instanz der Anwendung zu bringen. In einer Spieleumgebung wird die Anwendung auf die letzte qualifizierte Position für den Endanwender oder den Rollenspieler, der zu dem Endanwender gehört, aktualisiert.

[0120] Die Erzeugung einer virtuellen Maschine aus einer Schablone stellt sicher, dass die neue Instanz der virtuellen Maschine aus einer ursprünglichen bzw. unverfälschten Quelle stammt. Damit sind keine Viren in einer neuen Instanz einer virtuellen Maschine enthalten, solange kein Virus in die Information über das Anwenderprofil und/oder in die Aktualisierungsinformation eingedrungen ist. Auf diese Weise sind virtuelle Maschinen nicht permanente Objekte, wenn die Sitzung des Anwenders endet. Insbesondere beinhaltet das Verfahren den Empfang eines Befehls, um die Instanz der virtuellen Maschine zu vernichten. Zu diesem Zeitpunkt wird Information, die spezifisch ist für den Endanwender und mit diesem verknüpft ist, aktualisiert (beispielsweise wird eine „Spiel-Speicher-Datei" aktualisiert) und wird in einem Speichersystem abgelegt, das unabhängig von der virtuellen Maschine ist. Dies ist notwendig, da die Instanz der virtuellen Maschine beendet bzw. vernichtet wird. Die für den Endanwender spezifische mit ihn verknüpfte Information wird dann das nächste Mal verwendet, wenn von dem Endanwender eine virtuelle Maschine angefordert wird.

[0121] **Fig.** 10B ist ein Flussdiagramm **1000B**, das ein Verfahren zur Drosselung der Zuweisung von Ressourcen zeigt, wenn eine neue virtuelle Maschine erzeugt wird, um den Einfluss auf die Funktion bestehender virtueller Maschinen gemäß einer Ausführungsform der vorliegenden Offenbarung zu verringern. Insbesondere kann die Erzeugung der virtuellen Maschine derart, dass die Zuweisung von Ressourcen beinhaltet ist, eine nennenswerte Belastung auf bestehende virtuelle Maschinen ausüben, die gerade Anwendungen ausführen. Diese Belastung bzw. Last betrifft die Ausführung gerade ablaufender Anwendungen und kann in negativer Weise das gesamte Spieleerleben für jene Endanwender beeinträchtigen. Daher liefert das Flussdiagramm

1000B ein Verfahren zur Verringerung dieses Einflusses auf die Funktionsweise bestehender virtueller Maschinen, wenn eine virtuelle Maschine für einen Endanwender instantiiert wird.

[0122] Bei **1050** beinhaltet das Verfahren die Ermittelung einer ursprünglichen Zuweisung mehrerer Ressourcen für die Instantiierung der virtuellen Maschine. D. h., die virtuelle Maschine umfasst mehrere Ressourcen, wovon jede ursprünglich einer ursprünglichen Zuweisung zugeordnet ist. Beispielsweise können Ressourcen CPU-Prozessorkerne, Speicher, etc. umfassen. Beispielsweise können einer virtuellen Maschine ursprünglich acht GPU-Prozessorkerne zugewiesen sein.

[0123] Bei **1055** beinhaltet das Verfahren die Verringerung der ursprünglichen Zuweisung der mehreren Ressourcen. In einer Ausführungsform ist die Verringerung derart, dass die reduzierte Zuweisung kleiner ist als fünfundzwanzig Prozent der ursprünglichen Zuweisung. In einer weiteren Ausführungsform wird die Verringerung für jede der unterschiedlichen Ressourcen ausgeführt, und in einer Implementierung wird die Reduzierung in gleicher Weise über alle unterschiedlichen Ressourcen ausgeführt. D. h., jede der Ressourcen wird um einen entsprechenden Betrag auf weniger als fünfundzwanzig Prozent der ursprünglichen Zuweisung einer entsprechenden Ressourcen verringert. Bei **1060** beinhaltet das Verfahren die Initialisierung der virtuellen Maschine unter Anwendung der reduzierten Zuweisung der mehreren Ressourcen.

[0124] Nachdem die virtuelle Maschine instantiiert ist, muss eine zusätzliche Zuweisung von Ressourcen ausgeführt werden, um die virtuelle Maschine auf den Stand ihrer beabsichtigten Fähigkeiten zu bringen. Daher beinhaltet bei **1065** das Verfahren die Zuweisung weiterer Ressourcen für jede der mehreren Ressourcen. Die Ressourcen werden so zugewiesen, dass eine zusätzliche Zuweisung von Ressourcen für die mehreren Ressourcen so durchgeführt wird, dass die Zuweisung der Ressourcen die ursprüngliche Zuweisung erreicht.

TABELLE 3

AUFLISTUNG VON ANSPRÜCHEN

1. Ein Verfahren für eine Netzwerk-Wolke-Ressourcenerzeugung mit: Erzeugen einer Schablone einer virtuellen Maschine; Erzeugen einer Instanz einer virtuellen Maschine für einen Endanwender durch Klonen der Schablone; Laden einer Anwendung, die von der virtuellen Maschine ausgeführt wird; Zugreifen auf eine erste Information, die zu dem Endanwender gehört; und Laden der ersten Information in eine Instanz der Anwendung.
2. Das Verfahren nach Anspruch 1, wobei die Schablone der virtuellen Maschine mehrere im Voraus ausgewählte Spieleanwendungen umfasst, die für jeden von mehreren Endanwender verfügbar sind.
3. Das Verfahren nach Anspruch 1, wobei der Zugriff auf eine erste Information umfasst: Zugreifen auf eine gespeicherte Spieldatei, die zu der Anwendung gehört.
4. Das Verfahren nach Anspruch 1, wobei der Zugriff auf eine erste Information umfasst: Zugreifen auf ein Anwenderprofil des Anwenders.
5. Das Verfahren nach Anspruch 1, das ferner umfasst: Empfangen eines Befehls zur Vernichtung der Instanz der virtuellen Maschine; und Speichern einer aktualisierten Version der speziellen Information, die mit dem Anwender verknüpft ist, in einem Speichersystem, das unabhängig von der virtuellen Maschine ist.
6. Das Verfahren nach Anspruch 1, wobei die Erzeugung einer Instanz einer virtuellen Maschine umfasst: Ermitteln einer ursprünglichen Zuweisung mehrerer Ressourcen für die Instanz der virtuellen Maschine; Reduzieren der ursprünglichen Zuweisung der mehreren Ressourcen; und Initialisieren der virtuellen Maschine unter Anwendung der reduzierten Zuweisung der mehreren Ressourcen.
7. Das Verfahren nach Anspruch 6, wobei die Reduzierung der ursprünglichen Zuweisung umfasst: Reduzieren der ursprünglichen Zuweisung für jede der mehreren Ressourcen.
8. Das Verfahren nach Anspruch 7, wobei die Reduzierung der ursprünglichen Zuweisung umfasst: Zuweisen jeder der mehreren Ressourcen mit einem Betrag, der kleiner ist als fünfundzwanzig Prozent einer entsprechenden ursprünglichen Zuweisung.

9. Das Verfahren nach Anspruch 6, das ferner umfasst:
nach der Instantiierung der virtuellen Maschine, Zuweisen weiterer Ressourcen für jede der mehreren Ressourcen, um die ursprüngliche Zuweisung zu erreichen.

10. Das Verfahren nach Anspruch 1, wobei die Initialisierung der virtuellen Maschine ferner umfasst:
Initialisieren der Instanz der virtuellen Maschine in einem wolkenbasierten Grafikverarbeitungssystem für einen Endanwender.

11. Ein nicht-flüchtiges computerlesbares Medium mit von einem Computer ausführbaren Befehlen, um ein Computersystem zu veranlassen, ein Verfahren für eine Netzwerk-Wolke-Ressourcenerzeugung auszuführen, mit:
Erzeugen einer Schablone einer virtuellen Maschine;
Erzeugen einer Instanz einer virtuellen Maschine für einen Endanwender durch Klonen der Schablone;
Laden einer Anwendung, die von der virtuellen Maschine ausgeführt wird;
Zugreifen auf eine spezielle Information, die zu dem Endanwender gehört; und
Laden der speziellen Information in eine Instanz der Anwendung.

12. Das computerlesbare Medien nach Anspruch 11, wobei in dem Verfahren die Schablone der virtuellen Maschine mehrere Voraus ausgewählte Spieleanwendungen umfasst, die für jeden der mehreren Endanwender verfügbar sind.

13. Das computerlesbare Medium nach Anspruch 11, wobei der Zugriff auf eine erste Information in dem Verfahren umfasst:
Zugreifen auf eine gespeicherte Spieldatei, die zu der Anwendung gehört; und
Zugreifen auf ein Anwenderprofil des Anwenders.

14. Das computerlesbare Medium nach Anspruch 11, wobei die Erzeugung einer Instanz einer virtuellen Maschine in dem Verfahren umfasst:
Ermitteln einer ursprünglichen Zuweisung mehrerer Ressourcen für eine Instanz einer virtuellen Maschine;
Reduzieren der ursprünglichen Zuweisung der mehreren Ressourcen; und
Initialisieren der virtuellen Maschine unter Anwendung der reduzierten Zuweisung der mehreren Ressourcen.

15. Das computerlesbare Medium nach Anspruch 11, wobei das Verfahren ferner umfasst:
nach der Instantiierung der virtuellen Maschine, Zuweisen zusätzliche Ressourcen für jede der mehreren Ressourcen, um die ursprüngliche Zuweisung zu erreichen.

16. Das computerlesbare Medium nach Anspruch 14, wobei die mehreren Ressourcen eine Anzahl an CPU-Kernen und eine Größe eines Speichers umfassen.

17. Ein Computersystem mit:
einem Prozessor; und
einem mit dem Prozessor verbundenen Speicher, der Befehle gespeichert hat, die, wenn sie von einem Computersystem ausgeführt werden, das Computersystem veranlassen, ein Verfahren zur Netzwerk-Wolke-Ressourcenerzeugung auszuführen, mit:
Erzeugen einer Schablone einer virtuellen Maschine;
Erzeugen einer Instanz einer virtuellen Maschine für einen Endanwender durch Klonen der Schablone;
Laden einer Anwendung, die von der virtuellen Maschine ausgeführt wird;
Zugreifen auf eine spezifische Informationen, die zu dem Endanwender gehört; und
Laden der spezifischen Information in eine Instanz der Anwendung.

18. Das Computersystem nach Anspruch 17, wobei in dem Verfahren die Schablone einer virtuellen Maschine mehrere im Voraus ausgewählte Spieleanwendungen umfasst, die für jeden von mehreren Endanwender verfügbar sind.

19. Das Computersystem nach Anspruch 17, wobei die Erzeugung einer Instanz einer virtuellen Maschine in dem Verfahren umfasst:
Ermitteln einer ursprünglichen Zuweisung mehrerer Ressourcen für die Instanz der virtuellen Maschine;
Reduzieren der ursprünglichen Zuweisung der mehreren Ressourcen; und
Initialisieren der virtuellen Maschine unter Anwendung der reduzierten Zuweisung der mehreren Ressourcen.

20. Das Computersystem nach Anspruch 17, wobei das Verfahren ferner umfasst:
nach der Instantiierung der virtuellen Maschine, Zuweisung weiterer Ressourcen für jede der mehreren Ressourcen, bis die ursprüngliche Zuweisung erreicht ist.

Fensterverwaltung zur Reduzierung der Sichtbarkeit eines Desktop-Betriebssystems, das auf einem Frontfenster angezeigt wird

[0125] Fig. 11 ist eine Blockansicht eines Systems **1100**, das zum Ausführen einer Fensterverwaltung auf einer Fernanzeige zum Zwecke der Minimierung der Sichtbarkeit eines Desktop-Betriebssystems auf einem Frontfenster der Fernanzeige gemäß einer Ausführungsform der vorliegenden Offenbarung ausgeführt wird. Das System **1100** ist in der Wolke-Architektur **400B** aus **Fig.** 4B implementierbar und wird verwendet, um ein Ausgabe-Video entsprechender virtueller Maschinen zu beobachten, um sicherzustellen, dass eine Anwendung, wie sie von einem entsprechenden Endanwender angefordert ist, auf einem Frontfenster einer Anzeige angezeigt wird, und diese Anzeige und/oder Nachrichten eines Desktops bzw. einer Anzeigefläche eines zu Grunde liegenden Betriebssystems werden auf einer oder mehreren Hintergrundfenstern erzeugt. In einer Ausführungsform ist das System **1100**, das für eine Fensterverwaltung ausgebildet ist, innerhalb eines Spiele-Agenten **456A** einer Instanz einer virtuellen Maschine konfiguriert.

[0126] Wie in **Fig.** 11 gezeigt, umfasst das System **1101** ein wolkenbasiertes Grafikverarbeitungssystem **1110**, das eine virtualisierte Grafikverarbeitung für Fernanzeigen bereitstellt. Insbesondere ist das System **1110** in der Lage, mehrere virtuelle Maschinen für mehrere Endanwender durch entsprechende Endgeräte (beispielsweise schlanke Klienten, etc.) zu konfigurieren. Beispielsweise beinhaltet eine Instanz einer virtuellen Maschine ein Betriebssystem, das verwendet wird, um Befehle einer Anwendung an den Ressourcen, die für die virtuelle Maschine verfügbar und dieser zugeordnet sind, auszuführen. In einer Ausführungsform läuft jede der virtuellen Maschinen auf einem Windows-Betriebssystem, obwohl sie als Spieleplattformen konfiguriert und Aufrecht erhalten werden können.

[0127] Das System **1100** umfasst eine virtuelle Maschine **1120**, die durch das wolkenbasierte Grafikverarbeitungssystem instantiiert oder implementiert wird. Die virtuelle Maschine führt eine Anwendung aus, die typischerweise von einem Endanwender ausgewählt wird, um mit dieser zu interagieren. D. h., der Endanwender startet eine Spielesitzung mit dem wolkenbasierten Grafikverarbeitungssystem mit der Absicht, eine wolkenbasierte Spieleanwendung über eine entsprechende Instantiierung bzw. Instanz einer virtuellen Maschine zu spielen. Die virtuelle Maschine erzeugt bei der Ausführung der Anwendung einen Video-Strom, der erzeugte Bilder zum Anzeigen umfasst. Das erzeugte Video wird letztlich kodiert und einer Fernanzeige als Datenstrom zugeführt, um von einem oder mehreren Endanwender betrachtet zu werden.

[0128] Das System **1100** umfasst ein Anwendungsverwaltungsmodul **1130**, das ausgebildet ist sicherzustellen, dass Information, die in einem Frontfenster einer Fernanzeige angezeigt wird, innerhalb eines Kontexts der Anwendung liegt, der mit der ausgeführten Anwendung im Zusammenhang steht. Insbesondere ist ein Monitor **1140** ausgebildet, den Frontpufer bzw. Frontspeicher zu überwachen, um zu erkennen, wann Video-Informationen, die in dem Fontspeicher gespeichert ist, außerhalb des Kontexts der Anwendung liegt. Information, die aus dem Frontspeicher abgerufen oder ausgelesen wird, wird für ein unmittelbares Anzeigen ausgegeben.

[0129] Wenn die in dem Frontspeicher enthaltene Video-Information außerhalb des Kontexts der Anwendung liegt, ist eine Abschwächungsfunktion **1150** ausgebildet, eine Aktion zu ergreifen, die eine Wirkung der Video-Information bei der Darstellung abschwächt. Beispielsweise kann die Video-Information, die außerhalb des Kontexts der Anwendung liegt, eine Nachricht des Betriebssystems oder des Desktops sein, die durch einen Endanwender hervorgerufen wird (beispielsweise Eingabe einer Befehlssequenz, etwa ALT-ENTER), oder dieser Information kann direkt aufgrund der Funktion des Betriebssystems erzeugt sein.

[0130] Fig. 12 ist ein Flussdiagramm **1200**, das ein Verfahren zum Ausführen einer Fensterverwaltung auf einer Fernanzeige zum Zwecke der Minimierung der Sichtbarkeit eines Desktop-Betriebssystems auf einem Frontfenster der Fernanzeige gemäß einer Ausführungsform der vorliegenden Offenbarung zeigt. In einer noch weiteren Ausführungsform zeigt das Flussdiagramm **1200** ein computerimplementiertes Verfahren zur Ausführung einer Fensterverwaltung auf einer Fernanzeige zum Zwecke der Minimierung der Exposition bzw. der Sichtbarkeit eines Desktop-Betriebssystems auf einem Frontfenster der Fernanzeige. In einer weiteren Ausführungsform ist das Flussdiagramm **1200** in einem Computersystem implementiert, das einen Prozessor und einen mit dem Prozessor verbundenen Speicher aufweist, in welchem Befehle gespeichert sind, die, wenn sie von dem Computersystem ausgeführt werden, das System veranlassen, ein Verfahren auszuführen, um eine Fensterverwaltung auf einer Fernanzeige zum Zwecke der Minimierung der Exposition bzw. Sichtbarkeit eines Desktop-Betriebssystems auf einem Frontfenster der Fernanzeige auszuführen. In einer noch weiteren Ausführungsform sind Befehle zum Ausführen eines Verfahrens, wie es in Flussdiagramm **1200** gezeigt ist, auf einem nicht-flüchtigen computerlesbaren Speichermedium gespeichert, das von einem Computer ausführbare Befehle enthält, um ein Computersystem zu veranlassen, ein Verfahren auszuführen, um eine Fensterverwal-

tung auf einer Fernanzeige zum Zwecke der Minimierung der Exposition bzw. Sichtbarkeit eines Desktop-Betriebssystems auf einem Frontfenster der Fernanzeige auszuführen. In Ausführungsformen ist das im Flussdiagramm **1200** gezeigte Verfahren durch eine oder mehrere Komponenten des Computersystems **100** oder des Klienten-Geräts **200** der **Fig.** 1 und **Fig.** 2 sowie durch das System **1100**, das für eine Fensterverwaltung geeignet ist, aus **Fig.** 11 implementierbar.

[0131] Bei **1210** beinhaltet das Verfahren die Ausführung einer Anwendung in einer virtuellen Maschine, die durch ein wolkenbasiertes Grafikverarbeitungssystem implementiert ist. Beispielsweise ist das wolkenbasierte System durch die Architektur **400B** aus **Fig.** 4B implementiert, die ausgebildet ist, mehrere virtuelle Maschinen für mehrere Endanwender bereitzustellen, wobei jede der virtuellen Maschinen ein voll funktionsfähiges Rechensystem ist, das unter einem Betriebssystem arbeitet, etwa dem Windows®-Betriebssystem. In einer Implementierung ist das wolkenbasierte Grafikverarbeitungssystem eine Spieleplattform, in die Endanwender eingeben, um Spieleanwendungen, die in der Spieleplattform gespeichert sind und instantiiert werden, über eine entsprechende virtuelle Maschine zu spielen.

[0132] Während der Ausführung der Anwendung durch die virtuelle Maschine wird ein Video-Strom erzeugt, der erzeugte Bilder für die Anzeige enthält. Bei **1220** beinhaltet das Verfahren die Zuleitung des Video-Stroms der Anwendung zu einem Frontspeicher der entsprechenden virtuellen Maschine, wobei Information aus dem Frontspeicher abgeholt und zur Darstellung auf einer Fernanzeige ausgegeben wird. Durch die Zuleitung des Video-Stroms zu dem Frontspeicher wird sichergestellt, dass der Video-Strom auf einem Frontfenster der Fernanzeige dargestellt wird. Dies ist wichtig für das wolkenbasierte Grafikverarbeitungssystem, das beispielsweise als eine Spieleplattform dient. Durch die Zuleitung der Spieleanwendung zu dem Frontfenster oder dem obersten Fenster wird das Spielerlebnis für den Endanwender verbessert, da der Anwender vollständig in die Spieleumgebung – d. h. Vordergrund und Mitte – eintaucht. Ferner wird der Endanwender nicht mit der Sichtbarkeit des darunter liegenden Betriebssystems konfrontiert (beispielsweise Desktop, Fehlermeldungen, etc.). In einer weiteren Ausführungsform wird das Frontfenster maximiert, so dass der Video-Strom auf dem vollen Bildschirm der Fernanzeige angezeigt wird. In einer Implementierung wird die Zuleitung des Video-Stroms zu dem Frontspeicher und die Maximierung des Frontfensters durch Ausführung von Aufrufen einer Anwendungsprogrammierschnittstelle (API) bewerkstelligt.

[0133] Bei **1230** beinhaltet das Verfahren die Überwachung des Frontspeichers, um zu erkennen, wann eine in dem Frontspeicher enthaltene Video-Information außerhalb des Kontexts der Anwendung liegt. Das heißt, anstatt Information, die den Video-Strom der Anwendung, die gerade von der virtuellen Maschine abgearbeitet wird, betrifft, wird der Frontspeicher mit anderer Video-Information geladen, die außerhalb des Kontexts der Anwendung liegt. Beispielsweise kann die Video-Information spezifisch für das Betriebssystem sein und Antworten zu Befehlssequenzen des Betriebssystems enthalten, die von einem Anwender hervorgerufen werden. Beispielsweise enthalten einige Darstellungen von Befehlssequenzen ALT-TAB, wodurch der Desktop des Betriebssystems auf dem Frontfenster erscheint; oder ALT-ENTER, wodurch das Frontfenster minimiert wird; oder CTL-ALT-DELETE, wodurch das System neu hoch läuft. Jeder dieser Befehle zeigt ein Fenster mit Informationen, die nicht innerhalb des Kontexts der Anwendung liegen, und kann Dialoge oder Nachrichten oder ein zweites Video (beispielsweise Desktop) beinhalten.

[0134] In einer weiteren Ausführungsform beinhaltet die Video-Information einen eingefrorenen Bildschirm. D. h., die Anwendung kann abgestürzt sein und sendet den Video-Strom nicht zu der Fernanzeige. Das Verfahren beinhaltet das Erkennen, wann die Spieleanwendung abgestürzt ist. Dies wird bewerkstelligt, indem erkannt wird, wann der Video-Strom geendet hat. In diesem Falle kann die Fernanzeige auf ein einzelnes Bild zum Anzeigen beschränkt sein. Dies ist wiederum keine gewünschte Spielerfahrung, und Ausführungsformen der vorliegenden Erfindung können diese negative Spielerfahrung erkennen und Aktionen einleiten, um dies abzuschwächen.

[0135] Bei **1240** beinhaltet das Verfahren das Ergreifen einer Maßnahme, um eine Wirkung der Video-Information, die gerade angezeigt wird, abzuschwächen, wenn diese außerhalb des Kontexts der Anwendung liegt. In einer Ausführungsform wird die abschwächende Aktion ausgewählt durch ihre Fähigkeit, die Sichtbarkeit eines Desktops des Betriebssystems, eine Nachricht oder eine andere Kommunikationsform in dem Frontfenster zu minimieren.

[0136] In einer Ausführungsform ist die Video-Information eine Meldung des Betriebssystems oder ein Video, das auf dem Frontfenster angezeigt werden muss. Eine abschwächende Aktion beinhaltet die Ausführung eines API-Aufrufs, um eine Erzeugung des Video-Stroms zurück in den Frontspeicher zu erzwingen, und daher werden erzeugte Bilder aus dem Video-Strom wieder in dem Frontspeicher abgelegt, die für die Anzeige bereit

sind. In einer Ausführungsform beinhaltet die abschwächende Aktion das Ignorieren der Video-Information, die außerhalb des Kontexts der Anwendung liegt, und die Unterdrückung der Zuleitung der Video-Information zu der Anzeige.

[0137] In einer weiteren Ausführungsform wird als letztes Mittel die Sitzung der virtuellen Maschine beendet. Beispielsweise ist die Anwendung abgestürzt und kann innerhalb der Instanz der virtuellen Maschine nicht mehr neu gestartet werden. Daher beinhaltet die abschwächende Aktion die Beendigung einer Anwendersitzung, die die virtuelle Maschine implementiert, und beinhaltet auch die erneute Initialisierung einer weiteren virtuellen Maschine, um eine weitere Instanz der Spieleanwendung auszuführen.

TABELLE 4

AUFLISTUNG VON ANSPRÜCHEN

1. Ein nicht-flüchtiges computerlesbares Medium mit von einem Computer ausführbaren Befehlen, um ein Computersystem zu veranlassen, ein Verfahren für eine Fensterverwaltung auszuführen, wobei das Verfahren umfasst: Ausführen einer Anwendung in einer virtuellen Maschine, die durch ein wolkenbasiertes Grafikverarbeitungssystem implementiert ist, wobei die Anwendung einen Video-Strom mit erzeugten Bilder zum Anzeigen erzeugt; Lenken des Video-Stroms der Anwendung zu einem Frontspeicher, so dass ein Kontext der Anwendung, der den Video-Strom beinhaltet, auf einem Frontfenster einer entsprechenden Anzeige angezeigt wird; Überwachen des Frontspeichers, um zu erkennen, wann Video-Information, die in dem Frontspeicher enthalten ist, außerhalb des Kontexts der Anwendung liegt; und Durchführen einer Aktion, um eine Wirkung der Video-Information, die gerade angezeigt wird und außerhalb des Kontexts der Anwendung liegt, abzuschwächen.
2. Das computerlesbare Medium nach Anspruch 1, wobei das Verfahren ferner umfasst: Ausführen eines API-Aufrufs, um das Frontfenster der entsprechenden Anzeige zu maximieren.
3. Das computerlesbare Medium nach Anspruch 1, wobei das Durchführen einer Aktion in dem Verfahren umfasst: Ausführen eines API-Aufrufs, um eine Erzeugung des Video-Stroms zurück in den Frontspeicher zu erzwingen.
4. Das computerlesbare Medium nach Anspruch 1, wobei die Überwachung des Frontspeichers in dem Verfahren umfasst: Erkennen, wann die Spieleanwendung abgestürzt ist, indem ermittelt wird, dass der Video-Strom geendet hat; Beenden einer Anwendersitzung, die die virtuelle Maschine implementiert; und erneutes Initialisieren einer weiteren virtuellen Maschine, um eine weitere Instanz der Spieleanwendung auszuführen.
5. Das computerlesbare Medium nach Anspruch 1, wobei das Durchführen einer Aktion in dem Verfahren umfasst: Minimieren der Sichtbarkeit eines Windows-Desktops in dem Frontfenster.
6. Das computerlesbare Medium nach Anspruch 1, wobei das Durchführen einer Aktion in dem Verfahren umfasst: Ignorieren der Video-Information, die außerhalb des Kontexts der Anwendung liegt; und Unterdrücken einer Zuleitung der Video-Information zu der Anzeige.
7. Das computerlesbare Medium nach Anspruch 1, wobei in dem Verfahren die Video-Information von einer von einem Anwender hervorgerufenen Befehlssequenz eines Windows®-Betriebssystems erzeugt wird.
8. Das computerlesbare Medium nach Anspruch 1, wobei in dem Verfahren die Anwendung eine Spieleanwendung umfasst.

9. Ein Computersystem mit:
einem Prozessor; und
einem mit dem Prozessor verbundenen Speicher, der Befehle gespeichert hat, die, wenn sie von einem Computersystem ausgeführt werden, das Computersystem veranlassen, ein Verfahren zur Fensterverwaltung auszuführen, wobei das Verfahren umfasst:
Ausführen einer Anwendung in einer virtuellen Maschine, die durch ein wolkenbasiertes Grafikverarbeitungssystem implementiert ist, wobei die Anwendung einen Video-Strom erzeugt, der erzeugte Bilder zum Anzeigen enthält;
Zuleiten des Video-Stroms der Anwendung zu einem Frontspeicher, so dass ein Kontext der Anwendung, der den Video-Strom enthält, auf einen Frontfenster einer entsprechenden Anzeige angezeigt wird;
Überwachen des Frontspeichers, um zu erkennen, wann Video-Information, die in dem Frontspeicher enthalten ist, außerhalb des Kontexts der Anwendung liegt; und
Durchführen einer Aktion, um eine Wirkung der Video-Information, die außerhalb des Kontexts der Anwendung liegt, abzuschwächen.

10. Das Computersystem nach Anspruch 9, wobei das Verfahren ferner umfasst:
Ausführen eines API-Aufrufs, um das Frontfenster der entsprechenden Anzeige zu maximieren.

11. Das Computersystem nach Anspruch 9, wobei das Durchführen einer Aktion in dem Verfahren umfasst:
Ausführen eines API-Aufrufs, um eine Erzeugung des Video-Stroms zurück in den Frontspeicher zu erzwingen.

12. Das Computersystem nach Anspruch 9, wobei die Überwachung des Frontspeichers in dem Verfahren umfasst:
Erkennen, wann die Spieleanwendung abgestürzt ist, indem bestimmt wird, dass der Video-Strom geendet hat;
Beenden einer Anwendersitzung, die die virtuelle Maschine implementiert; und
erneutes Initialisieren einer weiteren virtuellen Maschine, um eine weitere Instanz der Spieleanwendung auszuführen.

13. Das Computersystem nach Anspruch 9, wobei das Durchführen einer Aktion in dem Verfahren umfasst:
Minimieren der Sichtbarkeit eines Fenster-Desktops in dem Frontfenster.

14. Das Computersystem nach Anspruch 9, wobei das Durchführen einer Aktion in dem Verfahren umfasst:
Ignorieren der Video-Information, die außerhalb des Kontexts der Anwendung liegt; und
Unterdrücken der Zuleitung der Video-Information zu der Anzeige.

15. Das Computersystem nach Anspruch 9, wobei in dem Verfahren die Video-Information von einer von einem Anwender hervorgerufenen Befehlssequenz eines Windows®-Betriebssystems erzeugt wird.

16. Das Computersystem nach Anspruch 9, wobei in dem Verfahren die Anwendung eine Spieleanwendung umfasst.

17. Ein System für eine Fensterverwaltung mit:
einem wolkenbasierten Grafikverarbeitungssystem;
einer virtuellen Maschine, die durch das wolkenbasierte Grafikverarbeitungssystem eingerichtet ist, wobei die virtuelle Maschine eine Anwendung ausführt und einen Video-Strom, der erzeugte Bilder zum Anzeigen enthält, erzeugt;
einem Anwendungsverwaltungsmodul, das ausgebildet ist, den Video-Strom der Anwendung zu einem Frontspeicher zuzuführen, so dass ein Kontext der Anwendung, der den Video-Strom enthält, auf einem Frontfenster einer entsprechenden Anzeige angezeigt wird, und wobei das Anwendungsverwaltungsmodul einen API-Aufruf ausführt, um das Frontfenster zu maximieren;
einem Monitor, der zu Überwachung des Frontspeichers ausgebildet ist, um zu erkennen, wann Video-Information, die in dem Fontspeicher enthalten ist, außerhalb des Kontexts der Anwendung liegt; und
einem Abschwächungsmodul, das ausgebildet ist, eine Aktion auszuführen, um eine Wirkung der Video-Information, die außerhalb des Kontexts der Anwendung liegt, abzuschwächen.

18. Das System nach Anspruch 17, wobei das Abschwächungsmodul ausgebildet ist, einen API-Aufruf auszuführen, um eine Erzeugung des Video-Stroms zurück in den Frontspeicher zu erzwingen.

19. Das System nach Anspruch 17, wobei der Monitor ausgebildet ist zu erfassen, wann die Spieleanwendung abgestürzt ist, indem ermittelt wird, dass der Video-Strom geendet hat; Beenden einer Anwendersitzung, die die virtuelle Maschine implementiert; und erneutes Initialisieren einer weiteren virtuellen Maschine, um eine weitere Instanz der Spieleanwendung auszuführen.
20. Das System nach Anspruch 17, wobei das Abschwächungsmodul ausgebildet ist, eine Sichtbarkeit eines Windows-Desktops in dem Frontfenster zu minimieren.

Verfahren und Systeme zur dynamischen Zuweisung von Ressourcen für virtuelle Maschinen in einer Spieleplattform, die eine wolkenbasierte virtualisierte Grafikverarbeitung für Fernanzeigen bereitstellt

[0138] Fig. 13A–O zeigen die Implementierung einer Spieleplattform, die eine wolkenbasierte virtualisierte Grafikverarbeitung für Fernanzeigen gemäß Ausführungsformen der vorliegenden Offenbarung bereitstellt. Die Spieleplattform bietet ein vollständiges Spielerlebnis für angeschlossene Endanwender. Beispielsweise stellt die Spieleplattform die notwendigen CPU-und GPU-Verarbeitungsleistungen über eine entsprechende virtuelle Maschine bereit und liefert eine angeforderte Spieleanwendung, die der Endanwender spielen will. Es gilt somit: die Endanwender verbinden sich mit der Spieleplattform über Anwendersitzungen, um die unterstützten Spieleanwendungen zu spielen. Die in den **Fig.** 13A–O bereitgestellten Ansichten liefern weitere Details und Eigenschaften, die zuerst in der Architektur **400B** aus **Fig.** 4B dargestellt sind. Obwohl sich die **Fig.** 13A–O auf Spieleanwendungen beziehen, ist zu beachten, dass die wolkenbasierte Grafikverarbeitungs-Plattform ausgebildet ist, eine beliebige Art von Anwendung zu ermöglichen.

[0139] Fig. 13A ist eine Blockansicht eines Systems **1300A**, das eine wolkenbasierte Grafikverarbeitung für Endanwender bereitstellt, wobei ein Bereitstellungverwalter bzw. eine Bereitstellungsverwaltungseinheit Spieleplätze oder virtuelle Maschinen für Endanwender in Spielesitzungen gemäß einer Ausführungsform der vorliegenden Erfindung instantiiert. Der Bereitstellungsverwalter **1303** koordiniert das Klienten-Gerät **1302**, um einen Spieleplatz **1301** in Bezug zu einer angeforderten Anwendung zu nstantiieren. Der Bereitstellungsverwalter **1303** kommuniziert mit der OpsDienst-Schicht **1347**, um zu ermitteln, welche Leistungsstufe eines Spiele-Agenten für die angeforderte Anwendung erforderlich ist. Sobald die virtuelle Maschine mit der optimalen Menge an Ressourcen zum Spielen der Spieleanwendung instantiiert ist, ist der Bereitstellungsverwalter **1303** ausgebildet, mit dem instantiierten Spieleplatz oder virtuellen Maschine **1301** über den Spiele-Agenten **1305** zu kommunizieren. Der Spiele-Agent **1305** ist ausgebildet, eine Instanz der angeforderten Anwendung zu erzeugen. Der Datenstrom-Dienstleister **1306** kommuniziert mit der Strom-Komponente **1307** auf der Klienten-Seite **1302**, um Video-Daten, die durch die Ausführung der Spieleanwendung in dem Spieleplatz **1301** erzeugt werden, zu dem Klienten-Gerät **1302** zu übertragen.

[0140] Fig. 13B ist ein Netzwerkdiagramm **1300B**, das die Netzwerkverbindungen zwischen den Komponenten des wolkenbasierten Grafikverarbeitungssystems **400B** gemäß einer Ausführungsform der vorliegenden Offenbarung zeigt.

[0141] Fig. 13C ist eine Ansicht **1300C**, die den Ablauf für eine Initialisierung einer standardmäßigen Spielesitzung zwischen einem End-Klienten **1302** und einem Spieleplatz (VM) **1301** über einen Bereitstellungsverwalter **1303** gemäß einer Ausführungsform der vorliegenden Offenbarung darstellt. Wie gezeigt, wird im Schritt 1 auf eine Liste von Spielen zugegriffen. Der Bereitstellungsverwalter **1303** (beispielsweise ein Bifröst-Klient) erhält die Liste von einem Positionsverwalter, der wiederum die Liste von einem Spiele-Datenlager bzw. Datenstore erhält. Im Schritt 2 wird von dem Klienten-Gerät **1302** eine Anforderung für das Spielen eines Spiels abgesetzt. Im Schritt 3 wird die Anforderung dem Bereitstellungsverwalter **1303** zugeleitet, der die Erzeugung einer Instanz eines Spieleplatzes **1301** in Reaktion auf die Anforderung koordiniert. Insbesondere kommuniziert im Schritt 4 der Bereitstellungsverwalter **1303** mit der Dienst-Registrierung **1304**, um eine verfügbare virtuelle Maschine zu erhalten. Der Bereitstellungsverwalter passt die Ressourcenkapazität der virtuellen Maschinen, die auf die empfohlene Ressourcen ausgerichtet sind, die für die angeforderte Spieleanwendung erforderlich sind, an, um für den Endanwender das bestmögliche Spielerlebnis bereitzustellen. Die Dienst-Registrierung **1304** verwaltet die mehreren virtuellen Maschinen und unterhält den Status für jede der virtuellen Maschinen. Damit hat die Dienst-Registrierung **1304** Kenntnis davon, welche virtuellen Maschinen aktiv verwendet oder instantiiert sind, und welche virtuellen Maschinen untätig und bereit sind, zugeordnet zu werden.

[0142] Im Schritt 5 kommuniziert der Bereitstellungsverwalter **1303** mit dem zugeordneten Spieleplatz **1301** über einen Spiele-Agenten **1305**. Im Schritt 6 wird eine Verhandlung ausgeführt, um den Spieleplatz **1301** zu instantiieren. Im Schritt 7 installiert der Spieleplatz **1301** die Spieleanwendung. Im Schritt 8 ist die Anwendersitzung aktiv. Im Schritt 9 öffnet der Spieleplatz die Anschlüsse, die erforderlich sind, um ein Video von der

Spieleanwendung zu dem End-Klienten **1302** über den Bereitstellungsverwalter **1303** zu übertragen. Im Schritt 11 wird eine Kommunikationssitzung zwischen dem Mjölnir-Dienstleister **1306** an dem Spieleplatz **1301** und dem Mjölnir-Klienten **1307** am End-Klienten **1302** eingerichtet. Die Mjölnir-Dienstleister stellen die Kodierung und die Paketbildung von Informationen bereit. Beispielsweise ist der Mjölnir-Dienstleister **1306** ausgebildet, grafische Video-Daten zu kodieren und Pakete zu bilden, die von dem Spieleplatz (VM) **1301** durch die Ausführung einer Spieleanwendung erzeugt werden. Im Schritt 12 beginnt der Spieleplatz **1301** mit der Ausführung der Spieleanwendung. Im Schritt 13 beginnt der Mjölnir-Dienstleister **1306** mit der Übertragung des Videos von der Spieleanwendung zu dem Mjölnir-Klienten **1307** in dem Klienten-Gerät **1303**.

[0143] Fig. 13D ist ein Diagramm **1300D**, das den Ablauf zur Zuweisung von Ressourcen (beispielsweise CPU-Kerne, virtuelle GPU, GPU, Speicher, etc.) zu einem Spieleplatz **1301** gemäß einer Ausführungsform der vorliegenden Offenbarung zeigt. Der in **Fig.** 13D dargestellte Prozess beginnt mit der Erzeugung einer GUID-Sitzungs-ID bzw. Kennung und einer Anforderung für die Spieleanwendung mit der ID bzw. Kennung. Das Klienten-Gerät **1303** kommuniziert mit dem Bereitstellungsverwalter **1303**. Im Schritt 2 prüft der Sitzungs-Handhabungs-Strang bzw. Thread **1310** in dem Bereitstellungsverwalter **1303**, so dass sichergestellt ist, dass die Sitzungs-ID im aktiven Sitzungsvorrat (wiederherstellen, wenn sie besteht) ist; validiert die Spieleanwendung, prüft die Maschinenart, fügt eine Anforderung in REDIS-Sitzungsvorrat und der FIFO-Warteschlange **1313** auf der Grundlage des Maschinentyps des Spieleplatzes **1301** hinzu. Es gibt eine FIFO-Warteschlange für jede Maschinenart an virtueller Maschine. **1312** im Schritt 3 wird einer der Zuweisungsstränge in dem Spielesitzung-Bereitstellungsverwalter (GSPM) **1311** aufgeweckt. Im Schritt 4 gibt der Sitzungs-Handhabungsstrang **1310** das Sitzungsobjekt an das Klienten-Gerät **1303** zurück. Im Schritt 5 wird der GSPM-Strang aktiv; erhält eine GLOBALE Verriegelung für die FIFO-Warteschlange (wird im Hinblick auf Verriegelung für Abmachungen geprüft und zurückgesetzt, wenn diese gefunden wird); und liest das erste Element aus jeder Warteschlange aus. Im Schritt 6 weist der GSPM **1311** die früheste angeforderte Sitzung zu, indem der erste relevante freie/heiße Platz aus der Dienst-Registrierung **1304** (durch SIM) entfernt wird. Im Schritt 7 wird die zugewiesene Sitzung von der FIFO-Warteschlange **1313** zu der „zugewiesen"-Liste verschoben und die globale Verriegelung wird freigegeben.

[0144] In **Fig.** 13D bestätigt im Schritt 8 der GSPM **1311** für den VM Host-Verwalter, dass der Platz in Ordnung ist mittels der VM-Host-API: SeatSelectionMessage bzw. PlatzAuswahlNachricht. Insbesondere gilt: bei a) fordert der GSPM den VM Host-Verwalter **1314** an; bei b) bestätigt der VM Host-Verwalter **1314** die Verfügbarkeit des Platzes intern (es wird sichergestellt, dass keine Anforderung zur Abgrenzung anhängig ist); bei c) bestätigt der VM Host-Verwalter **1314** die Unversehrtheit des Platzes; bei d) legt der VM Host-Verwalter **1314** den Spieleplatz **1301** in den zugewiesenen Behälter in der Dienst-Registrierung **1304** ab (über eine eigene SIM); und bei e) antwortet der VM Host-Verwalter **1314** der SPM, dass alles in Ordnung ist.

[0145] In **Fig.** 13D weist im Schritt 9 der GSPM **1311** den Anschluss mit gesichertem Reverse Proxy bzw. Umkehr-Proxy für diesen Spieleplatz **1301** zu. Im Schritt 10 verhandelt der GSPM **1311** mit dem Spiele-Agenten **1305** über den zugewiesenen Spieleplatz **1301** und übergibt Information über das Spielprofil, Anschlüsse, etc.. Im Schritt 11 berichtet der Spiele-Agent **1305** an den Bereitstellungsverwalter **1302** über den Spiele-Agenten-Zuhörer(GA-Zuhörer)-Strang, dass der Spieleplatz für eine Verbindung mit dem Klienten bereit ist.

[0146] Fig. 13E ist ein Diagramm **1300E**, das den Ablauf für die Freigabe eines Spieleplatzes **1301** gemäß einer Ausführungsform der vorliegenden Offenbarung zeigt. Im Schritt 1 liefert das Klienten-Gerät **1303** oder Spiele-Agent **1305** eine Benachrichtigung, dass die Spieleanwendung beendet ist. Im Schritt 2 tätigt der Sitzungs-Handhabungsstrang **1310** oder der GA-Zuhörer-Strang in dem Bereitstellungsverwalter **1303** einen API-Aufruf, um den Spielesitz freizugeben (beispielsweise durch Aufruf „GameSeatProvisionMgr API: release", wodurch der VM Host-Verwalter **1314** über einen weiteren API-Aufruf (beispielsweise VM Host API: PlatzFreigabeNachricht bzw. SeatReleaseMessage) aufruft. Im Schritt 3 stellt der VM Host-Verwalter **1314** die virtuelle Maschine wieder her. Im Schritt 4 legt, sobald die virtuelle Maschine zurückkommt, der VM Host-Verwalter **1314** die virtuelle Maschine zurück in den heißen/freien Behälter in der Dienst-Registrierung **1304** ab.

[0147] Fig. 13F ist ein Diagramm **1300F**, das den Ablauf zur Handhabung eines nicht behebbaren Fehlers während einer Spielesitzung gemäß einer Ausführungsform der vorliegenden Offenbarung zeigt. Im Schritt 1 liefert der Spiele-Agent **1305** eine Benachrichtigung über einen nicht behebbaren Fehler, oder dass BSOD für einen Spieleplatz **1301** aufgetreten ist, oder dass ein anderer schwerwiegender Fehler aufgetreten ist (beispielsweise eine Zeitüberschreitung des Bereitstellungsverwalters bei der Kommunikation mit dem Spiele-Agenten, etc.). Im Schritt 2 ruft der Bereitstellungsverwalter **1303** der SPM oder den Spieleplatz-Bereitstellungsverwalter **1311** über einen API-Aufruf auf (beispielsweise „GameSeatProvisionMgr API: release + error". Im Schritt 3 ruft der SPM **1311** den VM Host-Verwalter **1314** über einen API-Aufruf auf (beispielsweise „VM-

Host API: fence + release”). Im Schritt 4 markiert der VM Host-Verwalter **1314** den Spieleplatz **1301** als abgegrenzt bzw. umzäunt und führt eine Zurücksetzung oder andere geeignete Aktionen durch. Im Schritt 5 legt der VM Host-Verwalter **1314** den Spieleplatz **1301** zurück in den geeigneten Behälter (beispielsweise Fehler/abgegrenzt) in der Dienst-Registrierung **1304**.

[0148] Fig. 13G ist ein Diagramm **1300G**, das den Ablauf für die Zuweisung eines Spieleplatzes **1301** gemäß einer Ausführungsform der vorliegenden Offenbarung zeigt. Im Schritt 1 fordert das Klienten-Gerät **1303** eine Spieleanwendung aus der Spieleplattform oder dem wolkenbasierten Grafikverarbeitungssystems an. Im Schritt 2 ruft der Bereitstellungsverwalter **1303** den SPM **1311** über einen API-Aufruf (beispielsweise „GameSeatProvisionMgr API: Allocate bzw. Zuweisen”) auf. Im Schritt 2 ruft der SPM **1311** die Dienst-Registrierung **1304** über das Dienst-Registrierungs-Schnittstellenmodul und die SIM API auf, wobei das Dienst-Registrierungs-Schema verwendet wird. Im Schritt 4 gibt die Dienst-Registrierung **1304** ein „KEIN” heißer Spieleplatz zurück, was angibt, dass kein Spieleplatz verfügbar ist. Im Schritt 5 bestätigt der SPM **1311** für den VM Host-Verwalter **1314**, dass der Platz in Ordnung ist, wobei dies über einen API-Aufruf erfolgt (beispielsweise „VM-Host API: SeatAllocationMessage bzw. PlatzZuweisungsNachricht”): wobei bei a) der VM Host-Verwalter **1314** die Verfügbarkeit des Platzes bestätigt; bei b) der VM Host-Verwalter **1314** den Platz in den zugewiesenen Behälter in der Dienst-Registrierung **1304** ablegt (über seine eigene SIM); und bei c) der VM Host-Verwalter **1314** dem SPM **1311** antwortet, dass alles in Ordnung ist. Im Schritt 6 verhandelt der Bereitstellungsverwalter **1303** mit dem Spiele-Agenten **1305** über den zugewiesenen Spieleplatz **1301**.

[0149] Fig. 13H ist ein Diagramm **1300H**, das den Ablauf für die Kommunikation zwischen dem Klienten-Gerät **1303** und dem Bereitstellungsverwalter **1303** gemäß einer Ausführungsform der vorliegenden Offenbarung zeigt. Im Schritt 1 errichtet das Klienten-Gerät **1303** eine anfängliche Verbindung mit dem Bereitstellungsverwalter **1303** unter Anwendung des Hypertext-Transfer-Protokolls (HTTP). Der Bereitstellungsverwalter **1303** antwortet und führt Zuweisungen, etc. aus. Im Schritt 2 sendet das Klienten-Gerät **1303** eine Nachricht nach dem Anwender-Datagramm-Protokoll (UDP) mit dem Inhalt „Was ist mein Status?”. Dies wird regelmäßig ausgeführt. Im Schritt 3 sendet der Bereitstellungsverwalter **1303** eine UDP-Antwort mit dem Sitzungsstatus. Im Schritt 4 führt, wenn es keine UDP-Antwort von dem Bereitstellungsverwalter **1303** innerhalb einer gewissen Anzahl von Sekunden (beispielsweise Y) gibt, das Klienten-Gerät **1303** einen HTTP-Aufruf durch, um einen Sitzungsstatus zu erhalten.

[0150] Fig. 13I ist ein Diagramm **1300I**, das den Ablauf für eine Wiederverbindung einer Sitzung gemäß einer Ausführungsform der vorliegenden Offenbarung darstellt. Wie gezeigt, wird bei einer erfolgreichen Verbindung zwischen dem Klienten-Gerät und einem bestehenden Spieleplatz der Mjölnir-Klient in dem Klienten-Gerät erneut gestartet.

[0151] Fig. 13J ist ein Diagramm **1300J**, das die Eigenschaften eines Spiele-Datenvorrats bzw. Datenlagers bzw. Stores **1320** darstellt. Das Spiele-Datenlager **1320** liefert Zugriff auf viele Spieleanwendungen für die virtuellen Maschinen in einem wolkenbasierten Grafikverarbeitungssystem (zuvor beschrieben). Insbesondere interagiert das Datenlager **1320** mit den Klienten-Geräten und mit Spiele-Agenten der virtuellen Maschinen. Das Datenlager **1320** sendet eine Liste von Spiele unterstützter Spieleanwendungen zu jedem der Klienten-Geräte. Jedes Gebiet/DC ist ferner in der Lage, eine eigene Liste an Spielen (einschließlich einer Spielabgrenzung) zu unterhalten. Jeder Operator/MSO ist in der Lage, seine eigene Liste an Spielen (einschließlich einer Spielabgrenzung) zu unterhalten. Und Klienten-Geräte sind in der Lage, die Spieleliste aus der Wolke-Spieleplattform zu laden, die die Spielesitzungen bereitstellt.

[0152] Wie gezeigt, stellt das Spiele-Datenlager **1320** Spielprofile für jede der Spieleanwendungen bereit. Beispielsweise enthält das Spielprofil Titel-Daten **1322**; Titel-Bestände **1321**; Filterdaten **1323**; Startdaten **1324**; und Zaundaten bzw. Abgrenzungsdaten. Die Spieltiteldaten **1322** (in **Fig.** 13K gezeigt) enthalten die Spiele-ID (eindeutig); die Aufbau-ID; Veröffentlichungsdatum (in der Wolke); Bildschirmaufnahmen (beispielsweise Spieler 1) und Videos (URL-Verbindungen) (beispielsweise aufgenommen als Dateinamen); Einzelspieler/Mehr-Spieler-Fähigkeiten; unterstützte Sprachen; Spielname, Name des veröffentliches, schade, Beschreibung des Spiels; Datum der Verteilung des Spiels, QRL des Herausgebers; eine URL für Unterstützung oder zusätzliche Eigenschaften (beispielsweise die URL GeForce.com); Titelbild oder Verbindung zum Titelbild (beispielsweise als Dateinamen angegeben); Bewertung/Bedarfsanforderungen; Kaufparameter (beispielsweise Verbindung zu einem Verkaufspfad); und Warngrenzen für nicht erfolgte Anwendereingabe (beispielsweise in Sekunden).

[0153] Ferner beinhalten die Titel-Bestände **1321** binäre/heruntergeladene Bestände, auf die durch die URLs der Titeldaten verwiesen wird. Die Filterdaten **1323** enthalten erlaubte Gebiete, eine Liste unterstützter Hardware-Konfigurationen und VM-Abbilder, um das Betriebssystem/Treiber mit einzuschließen, und kompatible

Eingabegeräte. Die Startdaten **1324** enthalten ein Installationsskript, ein Startskript, Art des Schlüsselbildblocks und des Anfangsbildblocks, Startzeit, Spielprozessname und Hilfsprozesse, POPS pro Hardware, ein Skript/eine Exe-Datei für die Spieleinstellungen, ein Spieldisketten-I/O-Profil, maximale Grenze für nicht erfolgte Anwendereingabe (in Sekunden). Die Zaundaten bzw. Abgrenzungsdaten enthalten Spiel ein/Spiel aus (zonenweit oder Anwender/Kontostufe) und die Zeitgrenze für das Spiel (beispielsweise global).

[0154] Fig. 13K ist ein Diagramm **1300K**, das den Datenablauf eines Spielprofils zeigt. Im Schritt 1 fordert das Klienten-Gerät (über eine Anwenderschnittstelle zu einem Endanwender) eine Spielliste an. Im Schritt 2 erhält der Wolke-Klient SDK die Spielliste von dem einen oder den mehreren Bereitstellungsverwaltern **1302**. Im Schritt 3 erhält der Bereitstellungsverwalter **1303** eine Spielliste für nicht abgegrenzte Spiele durch Prüfung gespeicherter Titel und Filter-Daten (beispielsweise als Ordner implementiert), wie zuvor beschrieben ist. Die Struktur der Spielprofildaten beinhaltet das folgende: „Spiele/Freigabe" zeigt an, dass Spieleanwendungen nutzbar sind; „Spiele/abgegrenzt" zeigt an, dass Spieleanwendungen nicht verfügbar sind; und „Spiele/Rückspeicherung" zeigt an, welche Spieleanwendungen aktuell zurück gespeichert werden. Im Schritt 4 gibt der Bereitstellungsverwalter gefilterte Titel-Daten an das Klienten-Gerät **1303** zurück. Im Schritt 5 lädt die UI in dem Klienten-Gerät **1303** die Titel-Bestände von den HTTP-Datei-Dienstleistern herunter. Im Schritt 6 wird die Spieleanforderung an den Bereitstellungsverwalter **1303** von dem Klienten-Gerät **1303** gesendet. Im Schritt 7 prüft der Bereitstellungsverwalter **1303** die Filter-Daten im Hinblick auf geeignete Spieleplätze und weist aus der Spieleplatz-Registrierung **1304** zu. Im Schritt 8 sendet der Bereitstellungsverwalter **1303** die Startdaten an den Spiele-Agenten **1305** im Hinblick auf den erhaltenen Spieleplatz **1301**. Im Schritt 9 werden ein normaler Start des Spiels und eine Initialisierung der Sitzung durchgeführt.

[0155] Fig. 13L ist ein Diagramm **1300**, das eine dynamische Auflösung für Spieleanwendungen und Anwendungen beim Start der Sitzung gemäß einer Ausführungsform der vorliegenden Offenbarung darstellt.

[0156] FIG. M ist eine Blockansicht **1300M**, die die Komponenten jeder Schicht unter Anwendung eines wolkenbasierten virtualisierten Grafikverarbeitungssystems gemäß einer Ausführungsform der vorliegenden Offenbarung darstellt. Beispielsweise umfasst die Wolke-Infrastruktur **1330** eine VM-Steuerung, physikalische Knotenpunkte, eine Virtualisierung, NAS, Netzwerk, Überwachung und Werkzeuge, Qualität der Dienstleistung (QoS), Unterstützung, etc.. Der Software-Schicht Stapel **1331** implementiert den Dienst für die wolkenbasierte Grafikverarbeitung (beispielsweise Spieleplattform) und umfasst einen Spieleverwalter bzw. eine Spielverwaltungseinheit **1332**, der die Datenübertragung, Kodierung, QoS-Verwaltung, Speicherung und Abrufung, DRM/serielle Schlüsselverwaltung) ausführt. Die Software-Stapelschicht **1331** beinhaltet den Bereitstellungsverwalter **1302**, der eine Ausgabeeinheit, eine Anwender-Warteschlangenbildung, eine Anwenderverifizierung und eine Spieleregistrierung enthält. Die Spiel-Klienten-Schicht **1335** umfasst diverse Funktionskomponenten, enthält einen Proto-Klienten, ein GFE, eine oder mehrere Verbindungen zu Netz-Inhalten (beispielsweise GeForce.com), einen nativen Tablett-Klienten und eine Anwenderverwaltung. Auch ist eine Datenbank **1334** ausgebildet, langfristige Daten zu speichern, die zum Erzeugen virtueller Maschinen verwendet werden, die speziell für den Anwender sind. Beispielsweise speichert die Datenbank **1334** Information über Anwenderprofile und Information über die Spielespeicherung, wie dies zuvor beschrieben ist.

[0157] FIG. N ist eine Blockansicht, die einen Spieleverwalter **1340** und einen Bereitstellungsverwalter **1303** gemäß einer Ausführungsform der vorliegenden Offenbarung darstellt. Wie gezeigt, ist eine Instanz eines Spieleverwalters **1340** in einer virtuellen Maschine oder einem Spieleplatz **1301** vorgesehen. Der Spieleverwalter **1340** verwaltet die Spieleanwendung, die in dem Spieleplatz **1301** gestartet wird, und enthält einen Spiele-Agenten **1305**. Der Spiele-Agent **1305** umfasst das folgende: eine Starteinheit, die das Starten von Anwendungen, die Beendigung von Anwendungen, das Aufräumen und die Bewachung von Anwendungen durchführt; einen Spiele-Zustandsverwalter, der Speicher-und Lade-Operationen ausführt; und einen seriellen Schlüsselverwalter. Der Spieleverwalter **1340** umfasst ferner einen Datenstrom-Dienstleister **1341**, der die abgehenden Datenströme verwaltet. Der Datenstrom-Verwalter **1341** enthält ferner einen QoS-Verwalter, eine Video-Übertragungseinheit und eine Audio-Übertragungseinheit und eine Übertragungseinheit für Anwenderaktionen.

[0158] Es gibt eine Instanz eines Bereitstellungsverwalters in einem Netzwerk aus wolkenbasierten virtuellen Maschinen, die von mehreren Dienstleistern versorgt werden, die eine Dienstleister-Wolke **461A**–N bereitstellen. Der Bereitstellungsverwalter beinhaltet ein Anwender-Verifizierungsmodul, ein Registrierungsmodul, eine Starteinheit für eine physikalische Instanz, eine Ansammlung bzw. Pool von Instanzen, eine Spiele-Agenten-Schnittstelle und eine Warteschlangenbildung von Anwendern.

[0159] Fig. 13Q ist eine Blockansicht diverser Implementierungen einer GPU-Virtualisierung für die Bereitstellung einer wolkenbasierten Grafikverarbeitung für Fernanzeigen. Im Typ 1 gibt es eine physikalische Virtuali-

sierung von GPU. D. h., jede virtuelle Maschine (beispielsweise VM1 und VM2) ist einer physikalischen GPU, etwa einem Chipsatz, über eine Virtualisierungsschnittstelle (beispielsweise XEN virtuelle Maschinenschnittstelle) zugeordnet. Im Typ 2 gibt es eine Virtualisierung von GPU. D. h., jede virtuelle Maschine (beispielsweise VM1 und VM2) ist einer virtuellen GPU über eine Virtualisierungsschnittstelle zugeordnet. Im Typ 3 gibt es wiederum eine Virtualisierung von GPU, jedoch über eine pseudo-virtuelle Maschine. D. h., eine virtuelle Maschine wird selbst in eine oder mehrere virtuelle Sub-Maschinen (beispielsweise VBox) aufgeteilt. Jede der virtuellen Sub-Maschinen ist der virtuellen GPU zugeordnet.

[0160] **Fig.** 14A–H sind Diagramme, die ein System und ein Verfahren zum dynamischen Zuweisen und Zuordnen von Spieleplätzen in einer wolkenbasierten Spiele/Anwendungen-Umgebung oder Spieleplattform in Ausführungsformen der vorliegenden Erfindung zeigen. Wiederum stellt die Spieleplattform ein vollständiges Spielerlebnis für angeschlossene Endanwender bereit. Beispielsweise stellt die Spieleplattform die erforderlichen CPU-und GPU-Verarbeitungsleistungen über eine entsprechende virtuelle Maschinen bereit und liefert eine angeforderte Spieleanwendung, die der Endanwender zu spielen wünscht. Wichtig ist, dass sich Endanwender mit der Spieleplattform über Anwendersitzungen verbinden, um unterstützte Spieleanwendungen zu spielen. Die Darstellungen, die in den **Fig.** 14A–H angegeben sind, stellen weitere Details und Merkmale bereit, die erstmals in der Architektur **400B** aus **Fig.** 4B gezeigt und weiter in den **Fig.** 13A–O dargestellt sind. Obwohl die **Fig.** 14A–H sich auf Spieleanwendungen beziehen, ist zu beachten, dass die wolkenbasierte grafische Verarbeitungsplattform ausgebildet ist, eine beliebige Art von Anwendung zu ermöglichen.

[0161] Bei der Zuweisung von Spieleplätzen müssen die virtuellen Maschinen verwaltet werden. Beispielsweise müssen die virtuellen Maschinen gestartet und/oder instantiiert werden. Eine aktive virtuelle Maschine muss verwaltet und über das Netzwerk im Hinblick auf seine Verfügbarkeit und Fähigkeiten bekannt gegeben werden. Die virtuelle Maschine muss nach dem Ende der Anwendung oder wenn die virtuelle Maschine „schlecht" wird, oder aus einem anderen Grund entsorgt werden. Die Verwaltung der virtuellen Maschine betrifft die Gegebenheiten der virtuellen Maschine und nicht notwendigerweise die Anwendung, die in der virtuellen Maschine abgearbeitet wird, was mit der Verwaltung von Anwendungen in Beziehung steht.

[0162] Andererseits beinhaltet die Verwaltung von Anwendungen die Verwaltung von Anforderungen zum Spielen einer Spieleanwendung, wie dies zuvor beschrieben ist. Die Verwaltung von Anwendungen umfasst das Akzeptieren der Anforderung, das Anpassen der Anforderung an geeignet mit Ressourcen ausgestattete Spieleplätze durch Kenntnis über die Anforderungen der Anwendung für die Aufgabenausführung, die Bereitstellung einer Rückkopplung über die Initialisierungsprozedur für das Spiel, die Einrichtung einer Spielesitzung und die Beendigung einer Spielesitzung.

[0163] Insbesondere wird ein Anwender Sitzungsverwalter auf Portalen oder den Maschinen ausgeführt, die Anwenderanforderungen akzeptieren, wozu der Bereitstellungsverwalter **1302** gehört. Ferner kennen der VM-Verwalter oder der VM Host-Verwalter **462** die Art, wie virtuelle Maschinen zu handhaben sind, kennen aber die spezifischen Eigenheiten der Anwendung nicht. Die Platz-Registrierung **1304** liefert einen Ort, an welchem Spieleplätze bekannt gegeben werden und einen Ort, an welchem die Zuweisung und Zuweisungsstatus von Spieleplätzen verwaltet werden (für Zuweisung entnommen und für spätere Zuweisung zurückgegeben).

[0164] **Fig.** 14A ist ein Diagramm **1400A**, das die Verarbeitung einer eintreffenden Anforderung gemäß einer Ausführungsform der vorliegenden Offenbarung darstellt. Wie gezeigt, empfängt die Lastausgleichseinheit **466** die eintreffenden Anforderung von einem Endanwender oder einem Klienten-Gerät **1302**. Es sind viele Endanwender und mögliche Anhäufungen Anforderungen vorhanden, und es sind nur eine begrenzte Anzahl an Anwender-Sitzungsverwalter vorhanden, die in dem einen oder den mehreren Portalen **1402** angeordnet sind. Um Fortschrittsaktualisierungen für das Klienten-Gerät **1302** bereitzustellen, legt nach dem Akzeptieren einer Anforderung der Anwender-Sitzungsverwalter **1401** eine Anforderung in eine mit Priorität versehene FIFO-Warteschlange, in der sie asynchronen verarbeitet wird. Wie in **Fig.** 14A gezeigt ist, werden Anforderungen in FIFO-Warteschlangen angeordnet, die auf der Grundlage der Leistungsklasse konfiguriert sind. D. h., jede Warteschlange ist mit der Leistungsklasse eines speziellen Spieleplatzes (hohe, mittlere, geringe Ressourceneigenschaften) verknüpft. Beispielsweise kann die Warteschlange **1403** mit einer hohen Leistungsklasse verknüpft sein, und die Warteschlange **1404** kann mit der mittleren Leistungsklasse verknüpft sein. Insbesondere können abhängig von den Anforderungen der Anwendung eine oder mehrere Leistungsklassen virtueller Maschinen diese Anwendung bedienen. Beispielsweise wird die Anwendung in der virtuellen Maschine mit der höchsten Leistung abgearbeitet und ergibt das insgesamt beste Spielerlebnis, sie kann aber auch auf einer virtuellen Maschine mit mittlerer Leistung ausgeführt werden und ergibt dennoch ein zufriedenstellendes Spielerlebnis.

[0165] Die Zuordnung eines Spieleplatzes einer speziellen Leistungsklasse hängt von diversen Faktoren ab. Beispielsweise werden die Anforderungen des Anwenders betrachtet, etwa, welche Anwendung der Anwender auszuführen wünscht. Ferner werden Ressourcenerfordernisse für die Anforderung berücksichtigt, etwa welche Erfordernisse notwendig sind, um die Anwendung auszuführen (beispielsweise CPU, GPU, Speicher, Bandbreitenanforderungen. Ferner wird der Prozess, der zum Initialisieren einer Spielesitzung verwendet wird, berücksichtigt, und es wird auch berücksichtigt, wann die Beendigung der Spielesitzung erwartet wird.

[0166] Fig. 14B ist ein Diagramm **1400B**, das die weitere Verarbeitung eintreffender Anforderungen darstellt, um eine Anwendung gemäß einer Ausführungsform der vorliegenden Offenbarung auszuführen. In einem Portal **1402** erhält der Anwender-Sitzungsverwalter Kenntnis über die angeforderten Eigenheiten der Anwendung und fragt nach einem Dienst oder einem Spieleplatz, der die anwendungsspezifischen Erfordernisse erfüllen kann. Insbesondere spricht dem Anwender-Sitzungsverwalter **1401** mit dem Platz-Bereitstellungsverwalter **1406** und liefert Parameter für die gerade angeforderten Dienstleistungen. Der Platz-Bereitstellungsverwalter **1406** erhält Kenntnis, wie die Anforderung zu interpretieren ist und übersetzt sie in eine SIM-Anforderung, die von dem SIM-Modul **1407** verarbeitet wird. Das SIM-Modul **1407** spricht mit der Platz-Registrierung **1304** unter Anwendung eines Schlüssels, der von dem Platz-Bereitstellungsverwalter **1406** bereitgestellt wird, ruft den Wert ab, und gibt diesen an den Platz-Bereitstellungsverwalter **1406** zurück. Der Platz-Bereitstellungsverwalter **1406** interpretiert dann den Wert und gibt ihn an den Anwender-Sitzungsverwalter **1401** zurück. Insbesondere sind Spieleplätze als Schlüssel-Wert-Paare dargestellt.

[0167] Aus Sicht der Platz-Registrierung **1304** sind alle Einträge Schlüssel-Wert-Paare. Ferner sind mehrere Werte für den gleichen Schlüssel zulässig, in welchem Falle der Wert als eine Liste gespeichert ist. Aus Sicht des Platz-Bereitstellungsverwalters **1406** werden Schlüssel verwendet, um Endpunkt-virtuelle Maschinen abzurufen, die in der Lage sind, gewisse Aufgaben auszuführen. In einem Schlüssel-Wert-Paar nimmt der Schlüssel die Form an „SeatFree/performance_class bzw. /PlatzFrei/Leistungsklasse/". Ein beispielhafter Schlüssel umfasst" Seat/Free/Medium bzw. Platz/Frei/Medium/", was angibt, dass eine virtuelle Maschine der Lage ist, einen „mittleren" Grad an Dienstleistungen auszuführen. In einem Schlüssel-Paar kann der Wert folgende Form als Beispiel annehmen: 10.0.172.1.192.168.1.1 bei 10.0.0.1. Dieser Wert gibt an, dass die virtuelle Maschine unter dem Verwaltung-Host mit der IP-Adresse von 10.0.0.1 existiert.

[0168] Fig. 14C ist ein Diagramm **1400C**, das Platz-Registereinträge für die Verarbeitung von Anforderungen darstellt. Es gibt viele unterschiedliche Behälter bzw. „Eimer", in denen eine virtuelle Maschine oder ein Spieleplatz registriert wird. Ein heißer Behälter **1410** enthält eine Liste von Plätzen, die verfügbar oder frei sind, und zugeordneter Spieleplätze. Ein Start-Behälter **1411** enthält eine Liste von Spieleplätzen, die aktuell initialisiert werden. Ein Abgrenzungsbehälter **1412** enthält eine Liste von Spieleplätzen, die nicht verfügbar und damit abgegrenzt sind. Ein Fehlerbehälter **1413** enthält eine Liste von Spieleplätzen, die fehlerhaft waren.

[0169] Fig. 14D ist ein Diagramm **1400D**, das den Vorgang der Platzzuweisung darstellt. Insbesondere gibt der Platz-Bereitstellungsverwalter **1406** den ersten freien Dienst oder Spieleplatz zurück, der eine Anforderung gemäß einer Ausführungsform der vorliegenden Offenbarung abarbeiten kann. In einer Ausführungsform werden Plätze mit der höchsten Leistungsklasse, die erforderlich ist, zuerst bereitgestellt, dann wird die nächst höchste bereitgestellt, etc.. Der Eintrag wird von dem Platz-Bereitstellungsverwalter **1406** abgerufen, um die Stelle des Platzes tatsächlich zu finden. Der Anwender-Sitzungsverwalter **1401** besitzt nun die IP-Adresse, die den Spieleplatz repräsentiert, der in der Lage ist, die erforderliche Aufgabe auszuführen oder der gerade die Anwendung ausführt. Der Anwender-Sitzungsverwalter kontaktiert den VM Host-Verwalter **462**, der diesen Spieleplatz steuert, und fordert auf, die Unversehrtheit dieses Spieleplatzes sicherzustellen. Insbesondere fordert der VM Host-Verwalter den Spiele-Agenten **1305** auf, Auskunft über die Unversehrtheit des Spieleplatzes **1301** zu geben. Der VM Host-Verwalter entspricht dem und markiert markiert den Spieleplatz **1301** als zugewiesen. Der Anwender-Sitzungsverwalter **1401** kontaktiert dann den Spiele-Agenten **1305**, um die Spieleanwendung zu initiieren.

[0170] Bei der Platzzuweisung ist ein Spieleplatz in der freien Liste die meiste Zeit verfügbar (beispielsweise über 95%). Daher wird der Eintrag in der Platz-Registrierung in die HEISS/zugewiesen-Liste verschoben. Wenn ein Spieleplatz in der freien Liste nicht verfügbar ist, wird ein Fehler-Code zurückgegeben. In diesem Falle ist die gesamte physikalische Hardware in Benutzung und es wird ein Warteprozess ausgeführt, bis eine Freigabe der Dienstleistung auftritt, bevor eine Anforderung abgearbeitet werden kann. In einem weiteren Falle können, obwohl es nicht mehr HEISSE virtuelle Maschinen gibt, mir virtuelle Maschinen aktiv gemacht werden, um den steigenden Bedarf zu einem speziellen Zeitpunkt gerecht zu werden. Beim Warten ist der Platz-Bereitstellungsverwalter **1406** ferner ausgebildet, eine Funktion zu ermöglichen mit „Ruf mich, wenn diese Art von

Platz verfügbar ist". D. h., der Platz-Bereitstellungsverwalter **1406** prüft im Hinblick auf eine Benachrichtigung, dass eine gewisse Art von Spieleplatz aus der Platz-Registrierung **1304** verfügbar geworden ist.

[0171] Fig. 14E ist ein Diagramm **1400E**, das die Kommunikation zwischen dem Spiele-Agenten **1305** und dem Anwender-Sitzungsverwalter **1401A** oder **1401B** gemäß einer Ausführungsform der vorliegenden Offenbarung zeigt. Der Spiele-Agent **1305** sendet Nachrichten zurück zu den Anwender-Sitzungsverwaltern **1401A** und **1401B** in Bereitstellungsverwaltern, um Aktualisierungen über den Fortschritt der Spieleinitialisierung, Übertragung von Aktualisierungen, Abstürzen, etc. anzugeben. Diese Kommunikation wird unter Verwendung der IP-Adresse des Portals **1402A** oder **1402B** oder des Bereitstellungsverwalters ausgeführt, der die Spieleinitialisierung initialisiert hat. Wenn diese IP-Adresse nicht reagiert/nicht erreichbar ist, dann ist der Spiele-Agent in der Lage, an alle Bereitstellungsverwalter (beispielsweise **1402A**–N) eine Anforderung als Rundruf zu senden, wobei gefragt wird „gibt es einen PM da draußen", der meine Anforderung handhaben kann. Alle Bereitstellungsverwalter hören zu und werden antworten. Ferner sendet der Spiele-Agent **1305** Aktualisierungen an den ersten Bereitstellungsverwalter, der antwortet. Es gibt zwei wichtige Vorteile, wenn ein Rundruf gemacht wird, wenn auf die Registrierung für den Bereitstellungsverwalter Bezug genommen wird und ein Spieleplatz gesucht wird. Erstens, wenn ein Bereitstellungsverwalter nicht erfolgreich ist, dann würde der Verkehr zu der Registrierung in einer Zunahme des 1000 fachen resultieren (beispielsweise ein Bereitstellungsverwalter pro 1000 Spieleplätze). Ferner kann ein nicht erfolgreicher Bereitstellungsverwalter weiterhin in der Registrierung sein.

[0172] Fig. 14F ist ein Diagramm **1400F**, das gemäß einer Ausführungsform der vorliegenden Offenbarung den Prozess zeigt, der eingesetzt wird, wenn ein Spieleplatz freigegeben wird. Sobald eine Aufgabe (beispielsweise Spielesitzung) abgeschlossen ist, ruft ein Anwender-Sitzungsverwalter **1401** den Platz-Bereitstellungsverwalter **1406** auf, um einen Spieleplatz freizugeben. Der Platz-Bereitstellungsverwalter **1406** beinhaltet, dass der VM Host-Verwalter **462**, der in der physikalischen Maschine/Dienstleister ausgeführt wird, den Spieleplatz/Dienst (beispielsweise die virtuelle Maschine zurücksetzt) im Schritt 1 zurücksetzt. Im Schritt 2 weist der VM Host-Verwalter die Dienst-Registrierung **1304** an, den Spieleplatz zu entfernen, und somit wird dieser aus dem zugeordneten Block entfernt. Im Schritt 3 setzt der VM Host-Verwalter **462** den Dienst zurück und sendet die Registrierung an die Dienst-Registrierung **1304**, so dass der Spieleplatz nunmehr frei und verfügbar ist. D. h., der Status des Spieleplatzes wird von dem zugeordneten Block zu dem freien Block verschoben.

[0173] Wenn Platzregistrierungen unterhalten werden, ist der VM Host-Verwalter dafür verantwortlich, die Platzregistrierungen zu unterhalten. Der VM Host-Verwalter verifiziert, dass die Spieleplätze unter seiner Steuerung in der Dienst-Registrierung **1304** registriert werden. Wenn ferner der VM Host-Verwalter entdeckt, dass ein Spieleplatz in der Dienst-Registrierung **1304** aus irgendeinem Grund fehlt, dann fügt der VM Host-Verwalter den Spieleplatz der Dienst-Registrierung **1304** hinzu. Die Prüfung im Hinblick auf den Status der Spieleplätze erfolgt regelmäßig. Beispielsweise weiß der VM Host-Verwalter, wo die Registrierung sein sollte (beispielsweise in welchem Behälter, so dass die Prüfung entsprechend der korrekten Behälterliste erfolgt). Ferner ist die Hinzufügung eines Spieleplatzes zu der Dienst-Registrierung **1304** eine synchronisierte Operation für den VM Host-Verwalter. Der VM Host-Verwalter verarbeitet auch alle Bedingungen, die in dem Prozess der Platzverwaltung auftreten können. Die Bedingungen werden so gehandhabt, dass eine erneute Platzregistrierungen nicht zu schnell vorgenommen wird, und es wird eine erneute Verifizierung vor der erneuten Registrierung ausgeführt.

[0174] In der mit Priorität versehenen FIFO-Verarbeitung wird, wenn eine Anforderung zum Spielen eines Spieles eintrifft, diese Anforderung mit einer laufenden globalen Sequenznummer versehen und in der Warteschlange angeordnet, die den angeforderten Fähigkeiten entspricht. Dies impliziert, dass es so viele Warteschlangen für eintreffende Anforderungen gibt wie Platzfähigkeiten vorhanden sind. Der Strang bzw. Thread, der Anforderungen verarbeitet, erzeugt eine kurzlebige globale Verriegelung, um die wahre FIFO-Natur der Warteschlange sicherzustellen. D. h., nur ein einziger Strang, der Anforderungen verarbeitet, kann die globale Verriegelung zu einem Zeitpunkt in Besitz haben. Der Strang wird versuchen, die Anforderung mit der kleinsten Sequenznummer mit einem Element (das mit einem Spieleplatz verknüpft ist) zu erfüllen. Wenn keine Plätze verfügbar sind, wird er versuchen, das nächst niedrigere Anforderungselement (die nächste Sequenznummer in der Reihenfolge) mit einem Spieleplatz zu erfüllen, der geringere Fähigkeiten besitzt, sofern ein derartiger Platz existiert. Dies geschieht, um eine maximale Platzbesetzung zu gewährleisten. D. h., eine Anforderung für einen Platz mit weniger Fähigkeiten liegt nicht hinter eine Anforderung für einen Platz mit höheren Fähigkeiten fest und die FIFO-Integrität wird dennoch beibehalten.

[0175] Fig. 14G ist ein Diagramm **1400G**, das eine Aufrufantwort eines Platz-Bereitstellungsverwalters gemäß einer Ausführungsform der vorliegenden Offenbarung zeigt. Der Platz-Bereitstellungsverwalter **1406** un-

terstützt eine Eigenschaft „Ruf mich, wenn dieser Dienst verfügbar ist". Insbesondere führt der Platz-Bereitstellungsverwalter **1406** keine Abfrage der Platz-Registrierung durch, lauscht aber im Hinblick auf Rundrufen der VM Host-Verwalter. D. h., der VM Host-Verwalter registriert den Platz in der Platz-Registrierung **1304**, und der VM Host-Verwalter teilt die Platzverfügbarkeit in einem Rundruf mit, sobald die virtuelle Maschine HEISS ist.

[0176] Wenn der Platz-Bereitstellungsverwalter **1406** einen Rundruf empfängt, wird eine Aufrufantwort-Methode „Platz könnte verfügbar sein" aufgerufen. Die Implementierung der Aufrufantwort-Methode des Anwenders-Sitzungsverwalters benachrichtigt den oder die Strang oder Stränge, die die Anforderung aus der Anwenderwarteschlange verarbeiten, und dieser Strang erhält den ersten Eintrag aus der mit Priorität versehenen FIFO-Warteschlange und fährt mit der normalen Ausführung fort. Insbesondere können mehrere Stränge im Hinblick auf einen einzelnen Spieleplatz konkurrieren, der verfügbar wird, aber die Verwendung der globalen Verriegelung stellt sicher, dass die Integrität der FIFO-Warteschlange beibehalten wird. Ferner ruft der die Warteschlange verarbeitende Strang einen Endpunkt (beispielsweise Spiele-Agenten) auf, um den Start der Aufgabe auszuführen.

[0177] **Fig.** 14H ist ein Diagramm **1400H**, das die Netzwerkverbindung zwischen Komponenten einer wolkenbasierten grafischen Verarbeitungsplattform, die eine Platzzuweisung ausführt, gemäß einer Ausführungsform der vorliegenden Offenbarung zeigt. In einer Implementierung kombiniert das Diagramm **1400H** Teile der Diagramme **1400A** und **1400F**, wobei die Diagramme **1400A** und **1400H** den Prozess der Handhabung einer eintreffenden Anforderung durch die Anwender-Sitzungsverwalter **1401** zeigen, die in den Portalen **1402** der Bereitstellungsverwalter **1330** angeordnet sind. Die Anforderungen werden der mit Priorität versehenen FIFO (beispielsweise **1403**) zugeführt. Der Anwender-Sitzungsverwalter **1401** hat Kenntnis über die Gegebenheiten der angeforderten Anwendung und fordert einen Dienst oder einen Spieleplatz an, der die speziellen Erfordernisse der Anwendung erfüllen kann, indem in der Dienst-Registrierung **1304** nachgefragt wird. Ferner zeigen die Diagramme **1400F** und **1400H** den Prozess der Zuweisung und der Freigabe eines Spieleplatzes unter Anwendung der VM Host-Verwalter **462**, der Platz-Registrierung **1304** und den Verwaltungs-Hosts.

[0178] **Fig.** 15 ist ein Flussdiagramm, das ein Verfahren zum Zuweisen einer virtuellen Maschine zu einem End-Klienten in einem wolkenbasierten grafischen Verarbeitungssystem gemäß einer Ausführungsform der vorliegenden Offenbarung darstellt. In einer noch weiteren Ausführungsform zeigt das Flussdiagramm **1500** ein computerimplementiertes Verfahren zur Zuweisung einer virtuellen Maschine zu einem Eend-Klienten in einem wolkenbasierten grafischen Verarbeitungssystem. In einer weiteren Ausführungsform ist das Flussdiagramm **1500** in einem Computersystem implementiert, das einen Prozessor und einen mit dem Prozessor verbundenen Speicher aufweist, der darin gespeichert Befehle hat, die, wenn sie von dem Computersystem ausgeführt werden, das System veranlassen, ein Verfahren zur Zuweisung einer virtuellen Maschine zu einem End-Klienten in einem wolkenbasierten grafischen Verarbeitungssystem auszuführen. In einer noch weiteren Ausführungsform sind Befehle zur Ausführung eines Verfahrens, das im Flussdiagramm **1500** gezeigt ist, auf einem nicht-flüchtigen computerlesbaren Speichermedium gespeichert, das von einem Computer ausführbare Befehle enthält, um ein Computersystem zu veranlassen, ein Verfahren zur Zuweisung einer virtuellen Maschine zu einem End-Klienten in einem wolkenbasierten grafischen Verarbeitungssystem auszuführen. In Ausführungsformen ist das in dem Flussdiagramm **1500** dargestellte Verfahren durch eine oder mehrere Komponenten des Computersystems **100** oder des Klienten-Geräts **200** der **Fig.** 1 und **Fig.** 2 implementierbar.

[0179] Das Flussdiagramm **1500** zeigt ein Verfahren, um Spieleplätze in einer Wolke bzw. Cloud zuzuweisen bzw. zu reservieren und zuzuordnen, wobei jeder Spieleplatz ausgebildet ist, eine oder mehrere Instanzen einer Spieleausführung oder Anwendungsumgebung zu unterhalten. Ein Software-Modul-Bereitstellungsverwalter **1303** verarbeitet Anforderungen für Spieleplätze in der Weise einer Warteschlange. Er übersetzt die Anforderung in einen Spieleplatz einer speziellen Leistungsklasse, um in optimaler Weise die beste Spielerfahrung für den Endanwender bereitzustellen. Jede Leistungsklasse für die Spieleplätze ist mit einer Warteschlange verknüpft, in die Anforderung eingefügt wird.

[0180] Insbesondere beinhaltet bei **1510** das Verfahren den Empfang einer Anforderung zur Ausführung eine Anwendung von einem Klienten-Gerät, das zu einem Endanwender gehört. Bei **1520** beinhaltet das Verfahren die Ermittelung einer ersten Leistungsklasse für die Anwendung, die angibt, welche Art von Ressourcen erforderlich sind, um die Anwendung korrekt auszuführen, um damit den Endanwender ein gutes Spielerlebnis zu ermöglichen. Es können mehr als eine Leistungsklasse der Anwendung zugeordnet sein. Bei **1530** beinhaltet das Verfahren die Ermittelung einer ersten virtuellen Maschine der ersten Leistungsklasse, die verfügbar ist. D. h., virtuellen Maschinen werden ebenfalls Leistungsklassen zugewiesen, die denen entsprechen, die der Anwendung zugewiesen sind. Durch die Übereinstimmung von Leistungsklassen zwischen der Anwendung und der virtuellen Maschine wird für den Endanwender eine optimale Spielerfahrung ermöglicht, wenn er bei-

spielsweise die Spieleanwendung ausführt. Bei **1540** beinhaltet das Verfahren die Zuordnung der ersten virtuellen Maschine zum Zwecke des Ausführens der Anwendung in Verbindung mit dem Klienten-Gerät. D. h., die erste virtuelle Maschine wird dem Endanwender zugeordnet.

[0181] Wenn ermittelt wird, dass keine virtuelle Maschine der ersten Leistungsklasse verfügbar ist, dann beinhaltet das Verfahren die Ermittelung einer zweiten Leistungsklasse für die Anwendung und Bestimmung, dass eine zweite virtuelle Maschine verfügbar ist, die ebenfalls der zweiten Leistungsklasse entspricht. Die zweite virtuelle Maschine wird dann zum Zwecke des Ausführens der Anwendung dem Endanwender zugeordnet.

[0182] Anforderungen nach Spieleplätzen werden in der Weise zuerst-hinein-zuerst-heraus (FIFO) behandelt. Eine Anzahl von Arbeiter-Strängen bearbeiten die diversen Warteschlangen, indem das erste Element in den Warteschlangen angeschaut wird, auf die Verfügbarkeit in einem Schlüssel-Wert-Vorrat bzw. Store hin geprüft wird, und wenn verfügbar, wird die Anforderung in Form eines FIFO entfernt. Eine globale Verriegelung wird verwendet, um sicherzustellen, dass Anforderungen in einer FIFO-Manier abgearbeitet werden. Die Funktionsweise der globalen Verriegelung ist in Verbindung mit **Fig.** 16 beschrieben. Der Bereitstellungsverwalter **1302** kontaktiert den Ressourcenbesitzer des Spieleplatzes im Hinblick auf eine Bestätigung und eine aktualisierte Registrierung des Spieleplatzes. Die Zuweisung wird abgeschlossen, indem der Spiele-Agent **1305**, der in dem Spieleplatz **1301** ausgeführt wird, kontaktiert wird und indem die Details der Anforderung weitergereicht werden.

[0183] Wenn die FIFO-Warteschlange eingerichtet wird, wird einer Anforderung eine eindeutige globale Sequenznummer zugeordnet. Die Anforderung wird in einer oder mehreren FIFO-Warteschlangen abhängig von den zugeordneten Leistungsklassen der Anwendung angeordnet. Die Verarbeitung der Anforderungen wird bewerkstelligt, indem das erste Element aus jeder der mehreren Warteschlangen abgerufen wird, wobei jedes Element in den mehreren Warteschlangen mit einer Anforderung verknüpft ist, die eine entsprechende Sequenznummer hat. Es wird ermittelt, welches der Elemente die kleinste Sequenznummer hat. Es wird eine verfügbare virtuelle Maschine der Anforderung zugeordnet, die mit diesem Element verknüpft ist. Wenn keine virtuelle Maschine für dieses Element verfügbar ist, dann wird das nächste Element ermittelt, dass die nächste kleinste Sequenznummer hat. Es wird eine verfügbare Maschine diesem nächsten Element zugeordnet. Allgemeiner gesagt, wenn kein Platz für das Element mit der kleinste Sequenznummer ermittelt wird, dann ermittelt das Verfahren wiederholt ein nächstes Element mit der kleinsten Sequenznummer, wobei das nächste Element im Hinblick auf die Platzzuordnung und Zuweisungen bzw. Reservierungen noch nicht berücksichtigt worden ist. Danach wird eine verfügbare virtuelle Maschine diesem Element zugeordnet.

[0184] In einem physikalischen Dienstleister, der mehrere virtuelle Maschinen unterhält, wird jede virtuelle Maschine als ein Spieleplatz (beispielsweise **1301**) betrachtet. Es sollte beachtet werden, dass jede virtuelle Maschine mehrere Plätze ermöglichen kann, oder dass ein einzelner Platz von mehreren virtuellen Maschinen unterhalten werden kann. Unterschiedliche Anwendungen können unterschiedliche Mengen an Ressourcen und/oder Verarbeitungsleistung erfordern. Beispielsweise erfordert eine 3D-(dreidimensional)Spieleanwendung unter Umständen mehr Ressourcen und Verarbeitungsleistung als eine 2D-(zweidimensionale) Spieleanwendung. Auf der Grundlage der Spieleanwendung, den verfügbaren Plätzen und ihren entsprechenden Leistungsklassen, nicht verfügbaren Plätze und der Leistungsklasse der Spieleanwendung sowie den Leistungsklassen von Spieleplätzen, die diese Anwendung in der Vergangenheit unterstützten, wird einem gegebenen Platz eine gewisse Stufe an Leistungsklasse zugewiesen. Daher kann ermittelt werden, ob ein Platz geeignet ist, die Ausführung einer Spieleanwendung zu beginnen, die mehr oder weniger Verarbeitungsleistung benötigt. Daher können Spieleplätze unterschiedlichen Anwendungen auf der Grundlage von Leistungsklassen, den verfügbaren Spieleplätzen und der auszuführenden Anwendung zugewiesen oder zugeordnet werden.

[0185] **Fig.** 16 ist ein Flussdiagramm **1600**, das ein computerimplementiertes Verfahren zur Implementierung einer globalen Verriegelung zeigt, um die Handhabung von Anforderungen gemäß einer Ausführungsform der vorliegenden Offenbarung zu ordnen. In einer noch weiteren Ausführungsform zeigt das Flussdiagramm **1600** ein computerimplementiertes Verfahren zur Implementierung einer globalen Verriegelung zur Ordnung der Handhabung von Anforderungen. In einer weiteren Ausführungsform ist das Flussdiagramm **1600** in einem Computersystem implementiert, das einen Prozessor und einen mit dem Prozessor verbundenen Speicher aufweist, der gespeicherte Befehle enthält, die, wenn sie von dem Computersystem ausgeführt werden, das System veranlassen, ein Verfahren zur Implementierung einer globalen Verriegelung zur Ordnung der Handhabung von Anforderungen auszuführen. In einer noch weiteren Ausführungsform sind Befehle zur Ausführung eines Verfahrens, wie es in dem Flussdiagramm **1600** gezeigt ist, auf einem nicht-flüchtigen computerlesbaren Speichermedium gespeichert, das von einem Computer ausführbare Befehle aufweist, um ein Com-

putersystem zu veranlassen, ein Verfahren zur Implementierung einer globalen Verriegelung zur Ordnung der Handhabung von Anforderungen auszuführen. In Ausführungsformen ist das im Flussdiagramm **1600** gezeigte Verfahren durch eine oder mehrere Komponenten des Computersystems **100** oder des Klienten-Geräts **200** der **Fig.** 1 und **Fig.** 2 implementierbar.

[0186] Bei **1601** versucht das Verfahren, die globale Verriegelung zu erhalten. Bei **1602** wird versucht, die globale Verriegelung zu erhalten. Bei **1603** wird ein Versuch unternommen, die Verriegelung zu erhalten, indem der Name der Verriegelung geprüft wird und indem geprüft wird, ob der aktuelle Strang oder ein weiterer Strang den Namen der Verriegelung festlegen. Bei **1603** wird, wenn dies nicht der Fall ist, ermittelt, dass der Name nicht der gleiche ist, und dass ein weiterer Strang den Namen festgelegt hat, und der Prozess geht weiter zu **1605**. Wenn wahr, dann wird bei **1603** die Verriegelung erhalten, und der Prozess geht weiter zu **1604**, wo die Anforderung zugewiesen werden kann.

[0187] Wenn andererseits die Verriegelung nicht erhalten wird, dann geht der Prozess weiter zu **1605** und prüft die Unversehrtheit der Verriegelung, um zu erkennen, ob die Verriegelung verworfen oder ersetzt werden sollte. Bei **1620** verifiziert der Prozess, dass die Verriegelung unversehrt ist. Bei **1621** wird der Wert der Verriegelung aus der REDIS-Datenbank erhalten. Bei **1622** wird ein Vergleich durchgeführt, um zu erkennen, ob ein Zeitüberlauf aufgetreten ist, d. h. die längste Lebensdauer einer Verriegelung. Bei **1623** wird versucht, die Verriegelung freizugeben. D. h., der Wert der Verriegelung wird ein zweites Mal erhalten, und es wird ein neuer Wert dort mit einem Namen platziert, der mit dem aktuellen Strang verknüpft ist. Bei **1624** wird ein Vergleich durchgeführt, um zu bestimmen, ob die das zweite Mal erhaltene Verriegelung den gleichen Namen hat wie die das erste Mal erhaltene Verriegelung. Wenn sie identisch sind, dann kann der Strang die Verriegelung freigeben und die Verriegelung (seinen Namen zu der Verriegelung in REDIS hinterlegen) bei **1625** erhalten. Wenn sie nicht gleich sind, dann hat ein weiterer Strang diesen Strang gezwungen, die Verriegelung freizugeben. Damit wird bei **1626** der das zweite Mal erhaltene Wert wieder zurück in die Speicherstelle in der REDIS-Datenbank geschrieben.

[0188] Wenn die Verriegelung bei **1630** freigegeben wird, wird die Freigabe bei **1631** aggressiv ausgeführt und wird ungefähr zehnmal versucht. Wenn dies erfolgreich ist, dann endet der Prozess, und die Verriegelung ist freigegeben. Wenn jedoch diese nicht freigegeben wird, dann wird bei **1632** ein Löschversuch für den Namen der Verriegelung versucht. Wenn dies bei **1633** erfolgreich ist, dann wird die Verriegelung freigegeben, und der Prozess geht weiter zu **1637**. Wenn andererseits dies nicht bei **1633** erfolgreich ist, dann wird 6 Millisekunden lang bei **1634** gewartet, und es wird zu **1631** zurückgesprungen. Wenn es nicht erfolgreich war, oder wenn der wiederholte Versuch die maximale Anzahl an Versuchen übersteigt, dann wird in dem Prozess bei **1635** eine Ausnahme bei **1636** ausgegeben.

[0189] Wieder mit Bezug zu **1606** gilt, wenn die Verriegelung nicht erhalten wurde, dann wird bei **1607** ein Verriegelungspuls-Mechanismus ausgeführt. Dieser Mechanismus liefert eine Benachrichtigung, dass eine Verriegelung freigegeben worden ist. Die Benachrichtigung wird für den nächsten Strang in der Reihe zur Erhaltung der Verriegelung bereitgestellt, wobei dieser Strang bei **1601** beginnt.

[0190] **Fig.** 17A–F sind Darstellungen diverser Verfahren, die für die Platzzuweisung in einem wolkenbasierten grafischen Verarbeitungssystem gemäß Ausführungsformen der vorliegenden Offenbarung implementiert sind. Beispielsweise ist **Fig.** 17A ein Flussdiagramm **1700A**, das ein Verfahren zur Platzzuweisung durch einen Strang zeigt, der eine Anforderung aus einer FIFO-Warteschlange gemäß einer Ausführungsform der vorliegenden Offenbarung verarbeitet. **Fig.** 17B–C sind Flussdiagramme **1700B**–C, die ein Verfahren zur Verarbeitung einer Anforderung zeigen, sobald eine globale Verriegelung erhalten wird, gemäß einer Ausführungsform der vorliegenden Offenbarung. **Fig.** 17D ist ein Flussdiagramm **1700D**, das ein Verfahren zur Initiierung eines Spiels gemäß einer Ausführungsform der vorliegenden Offenbarung zeigt. **Fig.** 17E ist ein Flussdiagramm **1700E**, das ein Verfahren zur Wiederherstellung eines Strangs für die „Zuordnungsliste" gemäß einer Ausführungsform der vorliegenden Offenbarung zeigt, wobei dies bei einem Fehler nach einer Platzzuweisung geschieht, wobei ein separater Strang die „Zuordnungsliste" überwacht. **Fig.** 17F ist ein Flussdiagramm **1700S**, das ein Verfahren zum Lauschen nach Fernkommunikation gemäß einer Ausführungsform der vorliegenden Offenbarung zeigt.

TABELLE 5

AUFLISTUNG VON ANSPRÜCHEN

1. Ein Verfahren zur Zuweisung mit: Empfangen einer Anforderung zur Ausführung einer Anwendung von einem Klienten-Gerät, das mit einem Endanwender verknüpft ist; Bestimmen einer ersten Leistungsklasse für die Anwendung; Bestimmen einer ersten virtuellen Maschine der ersten Leistungsklasse, die verfügbar ist; und Zuordnen der ersten virtuellen Maschine zum Zwecke der Ausführung der Anwendung in Verbindung mit dem Klienten-Gerät.
2. Das Verfahren nach Anspruch 1, das ferner umfasst: Bestimmen einer zweiten Leistungsklasse für die Anwendung, wenn keine virtuelle Maschine der ersten Leistungsklasse verfügbar ist; Bestimmen einer zweiten virtuellen Maschine der zweiten Leistungsklasse, die verfügbar ist; und Zuordnen der zweiten virtuellen Maschine zur Verwendung durch den Endanwender, um die Anwendung auszuführen.
3. Das Verfahren nach Anspruch 1, das ferner umfasst: Zuordnen einer eindeutigen globalen Sequenznummer zu der Anforderung; Platzieren der Anforderung in einer oder mehreren FIFO-Warteschlangen abhängig von zugeordneten Leistungsklassen der Anwendung;
4. Das Verfahren nach Anspruch 3, das ferner umfasst: Abrufen eines ersten Elements aus jeder der mehreren Warteschlangen, wobei jedes Element in den mehreren Warteschlangen mit einer Anforderung verknüpft ist, die eine entsprechende Sequenznummer besitzt; Ermitteln, welches der abgerufenen Elemente die kleinste Sequenznummer hat; und Zuordnen einer verfügbaren virtuellen Maschine zu einer Anforderung, die mit dem Element mit der kleinsten Sequenznummer verknüpft ist.
5. Das Verfahren nach Anspruch 4, das ferner umfasst: Bestimmen, dass keine virtuelle Maschine verfügbar ist für das Element mit der kleinsten Sequenznummer; Ermitteln eines nächsten Elements mit einer nächsten kleinsten Sequenznummer; und Zuordnen einer verfügbaren virtuellen Maschine zu dem Element mit der nächsten kleinsten Sequenznummer.
6. Das Verfahren nach Anspruch 4, das ferner umfasst: Bestimmen, dass keine virtuelle Maschine für das Element mit der kleinsten Sequenznummer verfügbar ist; wiederholte Ermittlung eines nächsten Elements mit einer nächsten kleinsten Sequenznummer, das noch nicht berücksichtigt worden ist; und Zuordnen einer verfügbaren virtuellen Maschine zu dem Element mit der nächsten kleinsten Sequenznummer, das noch nicht berücksichtigt worden ist.
7. Das Verfahren nach Anspruch 1, wobei die virtuelle Maschine in einem wolkenbasierten Verarbeitungssystem instantiiert wird, das eine Grafikverarbeitung für eine Fernanzeige, die mit dem Klienten-Gerät in Verbindung steht, bereitstellt.
8. Ein nicht-flüchtiges computerlesbares Medium mit von einem Computer ausführbaren Befehlen, die ein Computersystem veranlassen, ein Verfahren zur Zuweisung auszuführen, mit: Empfangen einer Anforderung zur Ausführung einer Anwendung von einen Klienten-Gerät, das zu einem Endanwender gehört; Bestimmen einer ersten Leistungsklasse für die Anwendung; Bestimmen einer ersten virtuellen Maschine der ersten Leistungsklasse, die verfügbar ist; und Zuordnen der ersten virtuellen Maschine zum Zwecke des Ausführens der Anwendung in Verbindung mit dem Klienten-Gerät.

9. Das computerlesbare Medium nach Anspruch 8, wobei das Verfahren ferner umfasst:
Bestimmen einer zweiten Leistungsklasse für die Anwendung, wenn keine virtuelle Maschine der ersten Leistungsklasse verfügbar ist;
Bestimmen einer zweiten virtuellen Maschine der zweiten Leistungsklasse, die verfügbar ist; und
Zuordnen der zweiten virtuellen Maschine zur Verwendung durch den Endanwender zum Ausführen der Anwendung.

10. Das computerlesbare Medium nach Anspruch 8, wobei das Verfahren ferner umfasst:
Zuordnen einer einzigartigen bzw. eindeutigen globalen Sequenznummer zu der Anforderung;
Anordnen der Anforderung in einer oder mehreren FIFO-Warteschlangen abhängig von zugeordneten Leistungsklassen der Anwendung;

11. Das computerlesbare Medium nach Anspruch 10, wobei das Verfahren ferner umfasst:
Abrufen des ersten Elements aus jeder der mehreren Warteschlangen, wobei jedes Element in den mehreren Warteschlangen mit einer Anforderung verknüpft ist, die eine entsprechende Sequenznummer aufweist;
Bestimmen, welches der abgerufenen Elemente die kleinste Sequenznummer hat; und
Zuordnen einer verfügbaren virtuellen Maschine zu einer Anforderung, die mit dem Element mit der kleinsten Sequenznummer verknüpft ist.

12. Das computerlesbare Medium nach Anspruch 11, wobei das Verfahren ferner umfasst:
Bestimmen, dass keine virtuelle Maschine für das Element mit der kleinsten Sequenznummer verfügbar ist;
Bestimmen eines nächsten Elements mit einer nächsten kleinsten Sequenznummer; und
Zuordnen einer verfügbaren virtuellen Maschine zu dem Element mit der nächsten kleinsten Sequenznummer.

13. Das computerlesbare Medium nach Anspruch 11, wobei das Verfahren ferner umfasst:
Bestimmen, dass keine virtuelle Maschine für das Element ins der kleinsten Sequenznummer verfügbar ist;
wiederholtes Bestimmen eines nächsten Elements mit einer nächsten kleinsten Sequenznummer, die noch nicht berücksichtigt worden ist; und
Zuordnen einer verfügbaren virtuellen Maschine zu dem Element mit der nächsten kleinsten Sequenznummer, die noch nicht berücksichtigt worden ist.

14. Das computerlesbare Medium nach Anspruch 8, wobei in dem Verfahren die virtuelle Maschine in einem wolkenbasierten Verarbeitungssystem instantiiert wird, das eine Grafikverarbeitung für eine Fernanzeige, die zu dem Klienten-Gerät gehört, bereitstellt.

15. Ein Computersystem mit:
einem Prozessor; und
einem mit dem Prozessor verbundenen Speicher, der gespeicherte Befehle aufweist, die, wenn sie von einem Computersystem ausgeführt werden, das Computersystem veranlassen, ein Verfahren zur Zuweisung auszuführen, mit:
Empfangen einer Anforderung zur Ausführung einer Anwendung von einen Klienten-Gerät, das mit einem Endanwender verknüpft ist;
Bestimmen einer ersten Leistungsklasse für die Anwendung;
Bestimmen einer ersten virtuellen Maschine der ersten Leistungsklasse, die verfügbar ist; und
Zuordnen der ersten virtuellen Maschine zum Zwecke der Ausführung der Anwendung in Verbindung mit dem Klienten-Gerät.

16. Das Computersystem nach Anspruch 1, wobei das Verfahren ferner umfasst:
Bestimmen einer zweiten Leistungsklasse für die Anwendung, wenn keine virtuelle Maschine der ersten Leistungsklasse verfügbar ist;
Bestimmen einer zweiten virtuellen Maschine der zweiten Leistungsklasse, die verfügbar ist; und
Zuordnen der zweiten virtuellen Maschine zur Verwendung durch den Endanwender, um die Anwendung auszuführen.

17. Das Computersystem nach Anspruch 15, wobei das Verfahren ferner umfasst:
Zuordnen einer eindeutigen globalen Sequenznummer zu der Anforderung;
Anordnen der Anforderung in einer oder mehreren FIFO-Warteschlangen abhängig von zugeordneten Leistungsklassen der Anwendung;

18. Das Computersystem nach Anspruch 17, wobei das Verfahren ferner umfasst: Abrufen eines ersten Elements aus jeder der mehreren Warteschlangen, wobei jedes Element in den mehreren Warteschlangen mit einer Anforderung mit einer entsprechenden Sequenznummer verknüpft ist; Bestimmen, welches der abgerufenen Elemente die kleinste Sequenznummer hat; und Zuordnen einer verfügbaren virtuellen Maschine zu einer Anforderung, die mit dem Element mit der kleinsten Sequenznummer verknüpft ist.
19. Das Computersystem nach Anspruch 18, wobei das Verfahren ferner umfasst: Bestimmen, dass keine virtuelle Maschine für das Element mit der kleinsten Sequenznummer verfügbar ist; Bestimmen eines nächsten Elements mit einer nächsten kleinsten Sequenznummer; und Zuordnen einer verfügbaren virtuellen Maschine zu dem Element mit der nächsten kleinsten Sequenznummer.
20. Das Computersystem nach Anspruch 18, wobei das Verfahren ferner umfasst: Bestimmen, dass keine virtuelle Maschine verfügbar ist für das Element mit der kleinsten Sequenznummer; wiederholtes Bestimmen eines nächsten Elements mit einer nächsten kleinsten Sequenznummer, das noch nicht berücksichtigt worden ist; und Zuordnen einer verfügbaren virtuellen Maschine zu dem Element mit der nächsten kleinsten Sequenznummer, das noch nicht berücksichtigt worden ist.

[0191] Somit sind gemäß Ausführungsformen der vorliegenden Offenbarung Systeme und Verfahren beschrieben, die eine wolkenbasierte virtualisierte Grafikverarbeitung für entfernte Anzeigen bzw. Fernanzeigen implementieren, wie sie durch visuelle Computerapparate eingerichtet werden.

[0192] Obwohl die vorhergehende Offenbarung diverse Ausführungsformen unter Anwendung spezieller Blockansichten, Flussdiagramme und Beispiele angibt, kann jede Blockansichtskomponente, jeder Schritt eines Flussdiagramms, jede Operation und/oder jede Komponente, die hierin beschrieben und/oder dargestellt ist, individuell und/oder zusammen unter Anwendung einer breiten Fülle von Hardware-, Software- oder Firmware-(oder einer beliebigen Kombination davon)Konfiguration implementiert werden. Ferner sollte die Offenbarung von Komponenten, die in anderen Komponenten enthalten sind, als Beispiel betrachtet werden, dahingehend, dass viele bauliche Varianten implementiert werden können, um die gleiche Funktion zu erreichen.

[0193] Die Prozessparameter und der Ablauf von Schritten, wie sie hierin beschrieben und/oder dargestellt sind, sind lediglich beispielhaft und können nach Bedarf variiert werden. Beispielsweise sind die dargestellten und/oder beschriebenen Schritte in einer speziellen Reihenfolge gezeigt oder erläutert, wobei diese Schritte nicht notwendigerweise in der dargestellten oder angegebenen Reihenfolge ausgeführt werden müssen. Die diversen beispielhaften Verfahren, die hierin beschrieben und/oder dargestellt sind, können ausgeführt werden, indem einer oder mehrere der hierin beschriebenen dargestellten Schritte weggelassen werden, oder indem weitere Schritte zusätzlich zu den beschriebenen Schritten eingeführt werden.

1. Obwohl diverse Ausführungsformen beschrieben und/oder dargestellt sind im Zusammenhang mit vollständig funktionellen Rechensystemen, können eine oder mehrere dieser anschaulichen Ausführungsformen als ein Programmprodukt in einer Vielzahl von Formen verteilt werden, unabhängig von der speziellen Art des computerlesbaren Mediums, das zum tatsächlichen Ausführen der Verteilung verwendet wird. Die hierin offenbarten Ausführungsformen können auch unter Verwendung von Software-Modulen, die gewisse Aufgaben ausführen, eingerichtet werden. Diese Software-Module können Skript-, Batch- oder andere ausführbare Dateien beinhalten, die auf einem computerlesbaren Speichermedium oder in einem Rechensystem gespeichert werden können. Diese Software-Module können ein Rechensystem konfigurieren, um eine oder mehrere der anschaulichen hierin offenbarten Ausführungsformen auszuführen. Ein oder mehrere der Software-Module, die hierin offenbart sind, können in einer Cloud-Rechenumgebung bzw. Wolke-Rechenumgebung implementiert werden. Wolke-Rechenumgebungen bieten diverse Dienste und Anwendungen über das Internet an. Diese wolkenbasierte Dienste (beispielsweise Software als Dienst, Plattform als Dienstleistung, Infrastruktur als eine Dienstleistung, etc.) können über eine Netz-Browser-Anwendung oder über eine Fern-Schnittstelle abgerufen werden. Diverse hierin beschriebene Funktionen können über eine entfernte Desktop-Umgebung oder eine andere wolkenbasierte Rechnerumgebung bereitgestellt werden.

[0194] Die vorhergehende Beschreibung ist zum Zwecke der Erläuterung mit Bezug zu speziellen Ausführungsformen angegeben. Jedoch sollen die anschaulichen Erläuterungen oben nicht erschöpfend sein oder sollen die Erfindung nicht auf die genaue offenbarte Form einschränken. Es sind viele Modifizierungen und Variationen im Lichte der obigen Lehren möglich. Die Ausführungsformen wurden ausgewählt und beschrie-

ben, um die Prinzipien der Erfindung und ihrer praktischen Anwendungen am besten zu erklären, um damit den Fachmann in die Lage zu versetzen, die Erfindung am besten zu nutzen, und diverse Ausführungsformen mit diversen Modifizierungen können auf die spezielle betrachtete Verwendung hin zugeschnitten werden.

[0195] Es sind somit Ausführungsformen gemäß der vorliegenden Offenbarung beschrieben. Obwohl die vorliegende Offenbarung in speziellen Ausführungsformen beschrieben ist, sollte beachtet werden, dass die Offenbarung nicht durch derartige Ausführungsformen als eingeschränkt erachtet werden sollte, sondern sie sollte im Lichte den folgenden Ansprüchen bewertet werden.

Patentansprüche

1. Eine Vorrichtung zur Bereitstellung grafischer Verarbeitung, mit:
einer dualen CPU-Sockel-Architektur mit einem ersten CPU-Sockel und einem zweiten CPU-Sockel;
mehreren GPU-Leiterplatten, die mehrere GPU-Prozessoren bereitstellen, die mit dem ersten CPU-Sockel und dem zweiten CPU-Sockel verbunden sind, wobei jede GPU-Leiterplatte zwei oder mehr der mehreren GPU-Prozessoren aufweist; und
einer Kommunikationsbusschnittstelle, die den ersten CPU-Sockel mit einer ersten Teilgruppe aus einer oder mehreren GPU-Leiterplatten und den zweiten CPU-Sockel mit einer zweiten Teilgruppe aus einer oder mehreren GPU-Leiterplatten verbindet.

2. Die Vorrichtung nach Anspruch 1, die ferner umfasst:
einen Mehrkern-Prozessor, der mit dem ersten CPU-Sockel verbunden ist.

3. Die Vorrichtung nach Anspruch 2, wobei der Mehrkern-Prozessor einen XEON E 2670-Prozessor umfasst, der mit dem ersten CPU-Sockel verbunden ist.

4. Die Vorrichtung nach Anspruch 1, wobei die Kommunikationsbusschnittstelle umfasst:
mehrere Kommunikationsbrücken, wovon jede eine entsprechende GPU-Leiterplatte mit dem ersten CPU-Sockel und dem zweiten CPU-Sockel verbindet.

5. Die Vorrichtung nach Anspruch 4, wobei mindestens eine der Kommunikationsbrücken eine PCIe-Brücke umfasst.

6. Die Vorrichtung nach Anspruch 5, wobei jede der mehreren GPU-Leiterplatten umfasst:
einen Brückenteiler, der mit einer entsprechenden PCIe-Brücke verbunden ist; und
zwei oder mehr der mehreren GPU-Prozessoren.

7. Die Vorrichtung nach Anspruch 5, wobei die mehreren GPU-Leiterplatten der dualen CPU-Sockel-Architektur symmetrisch verteilt sind.

8. Die Vorrichtung nach Anspruch 6, die ferner umfasst:
eine erste GPU-Leiterplatte und eine zweite GPU-Leiterplatte, die jeweils mit dem ersten CPU-Sockel verbunden sind; und
eine dritte GPU-Leiterplatte und eine vierte GPU-Leiterplatte, die jeweils mit dem zweiten CPU-Sockel verbunden sind;
wobei jeweils die erste, die zweite, die dritte und die vierte GPU-Leiterplatte eine Zwei-GPU-Prozessor-Konfiguration aufweist.

9. Die Vorrichtung nach Anspruch 5, wobei die mehreren GPU-Leiterplatten in der dualen CPU-Sockel-Architektur unsymmetrisch verteilt sind.

10. Die Vorrichtung nach Anspruch 1, die ferner umfasst:
Ein Frontseiten-Busnetzwerk, das den ersten CPU-Sockel mit dem zweiten CPU-Sockel verbindet;

11. Die Vorrichtung nach Anspruch 1, wobei das Frontseiten-Busnetzwerk umfasst:
eine QPI-Verbindung, die den ersten CPU-Sockel kommunizierend mit dem zweiten CPU-Sockel verbindet; und
ein einzelnes Betriebssystem zur Verwaltung von CPU-Prozessoren, die mit dem ersten CPU-Sockel und dem zweiten CPU-Sockel entsprechend der Unterstützung durch die QPI-Verbindung gekoppelt sind.

12. Die Vorrichtung nach Anspruch 1, wobei mindestens einer der mehreren Grafikprozessoren einen Nvidia GK107-Grafikprozessorchip umfasst.

13. Die Vorrichtung nach Anspruch 1, wobei ein GPU-Prozessor eine virtuelle Maschine in einer Eins-zu-Eins-Zuordnung unterstützt.

14. Die Vorrichtung nach Anspruch 1, die ferner umfasst:
mehrere virtualisierte GPU-Prozessoren, die von den mehreren GPU-Prozessoren unterstützt sind, wobei eine virtualisierte GPU eine virtuelle Maschine unterstützt.

15. Die Vorrichtung nach Anspruch 1, wobei die duale CPU-Sockel-Architektur in einer Sandy-Brückenkonfiguration konfiguriert ist.

16. Die Vorrichtung nach Anspruch 1, wobei die mehreren GPU-Leiterplatten identisch sind.

17. Die Vorrichtung nach Anspruch 1, wobei die duale CPU-Sockel-Architektur und die mehreren GPU-Leiterplatten mehrere virtuelle Maschinen unterstützen, wobei jede Teile eines oder mehrerer CPU-Kerne und einer oder mehrerer GPU-Prozessoren aufweist.

18. Die Vorrichtung nach Anspruch 1, wobei die duale CPU-Sockel-Architektur und die mehreren GPU-Leiterplatten in einem System von pseudo-virtuellen Maschinen eingerichtet sind, das unter einem einzelnen Betriebssystem arbeitet und mehrere Anwendungen für einen oder mehrere Endanwender ausführt.

19. Eine an einem Netzwerk angebundene GPU-Einrichtung mit:
mehreren Verarbeitungsleiterplatten, die mehrere virtuelle CPU-und GPU-Prozessoren bereitstellen, wobei jede der Verarbeitungsleiterplatten umfasst:
eine duale CPU-Sockel-Architektur mit einem ersten CPU-Sockel und einem zweiten CPU-Sockel;
mehrere GPU-Leiterplatten, die mehrere GPU-Prozessoren bereitstellen, die mit dem ersten CPU-Sockel und dem zweiten CPU-Sockel verbunden sind, wobei jede GPU-Leiterplatte zwei oder mehr der mehreren GPU-Prozessoren aufweist;
mehrere erste Kommunikationsbrücken, die jeweils eine entsprechende GPU-Leiterplatte mit dem ersten CPU-Sockel und dem zweiten CPU-Sockel verbinden; und
eine Kommunikationsschnittstelle, die den ersten CPU-Sockel mit einer ersten Teilgruppe aus einer oder mehreren GPU-Leiterplatten und den zweiten CPU-Sockel mit einer zweiten Teilgruppe aus einer oder mehreren GPU-Leiterplatten verbindet.

20. Die Vorrichtung nach Anspruch 18, wobei jede der mehreren GPU-Leiterplatten umfasst:
einen Brückenteiler, der mit einer entsprechenden Kommunikationsbrücke verbunden ist; und
zwei oder mehr der mehreren GPU-Prozessoren.

21. Die Vorrichtung nach Anspruch 18, wobei die mehreren der GPU-Leiterplatten in der dualen CPU-Sockel-Architektur symmetrisch verteilt sind, so dass eine erste GPU-Leiterplatte und eine zweite GPU-Leiterplatte jeweils mit dem ersten CPU-Sockel verbunden sind, und eine dritte GPU-Leiterplatte und eine vierte GPU-Leiterplatte jeweils mit dem zweiten CPU-Sockel verbunden sind, wobei jeweils die erste, die zweite, die dritte und die vierte GPU-Leiterplatte eine Zwei-GPU-Prozessor-Konfiguration aufweisen.

22. Die an einem Netzwerk angebundene GPU-Einrichtung nach Anspruch 18, die ferner umfasst:
mehrere Emulatoren für eine Recheneinrichtung, die von den mehreren Verarbeitungsleiterplatten unterstützt werden.

Es folgen 47 Seiten Zeichnungen

Anhängende Zeichnungen

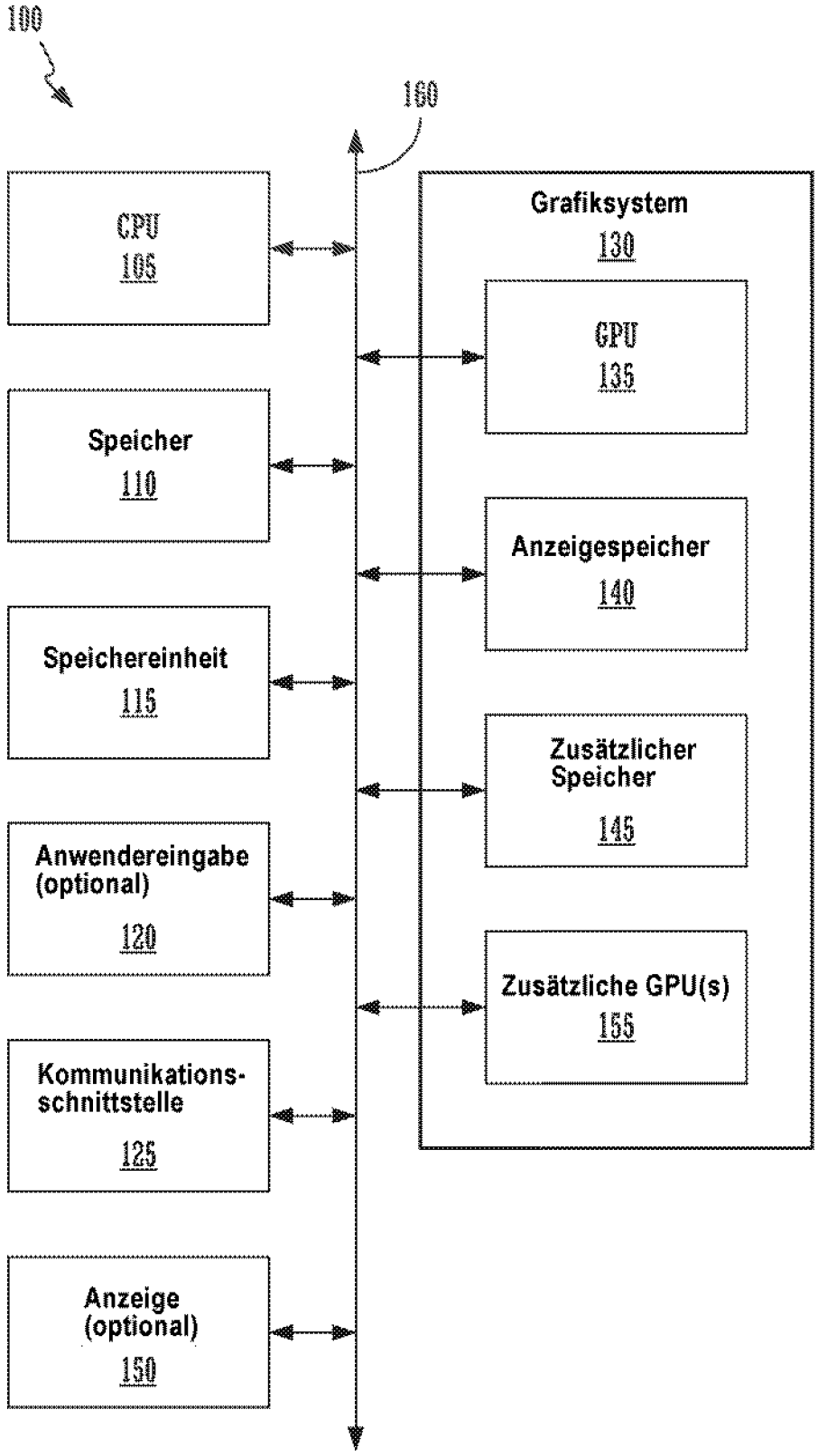


FIG. 1

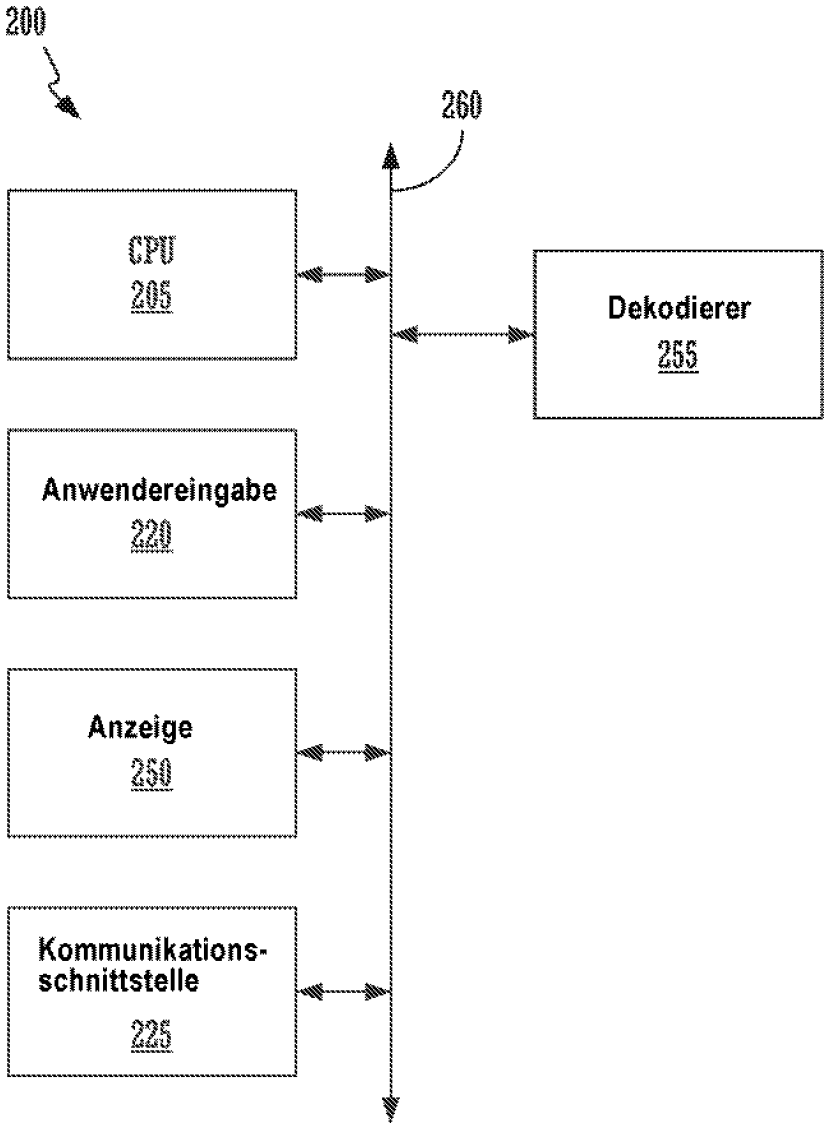

200
260
CPU
205
Dekodierer
255
Anwendereingabe
220
Anzeige
250
Kommunikations-
schnittstelle
225

FIG. 2

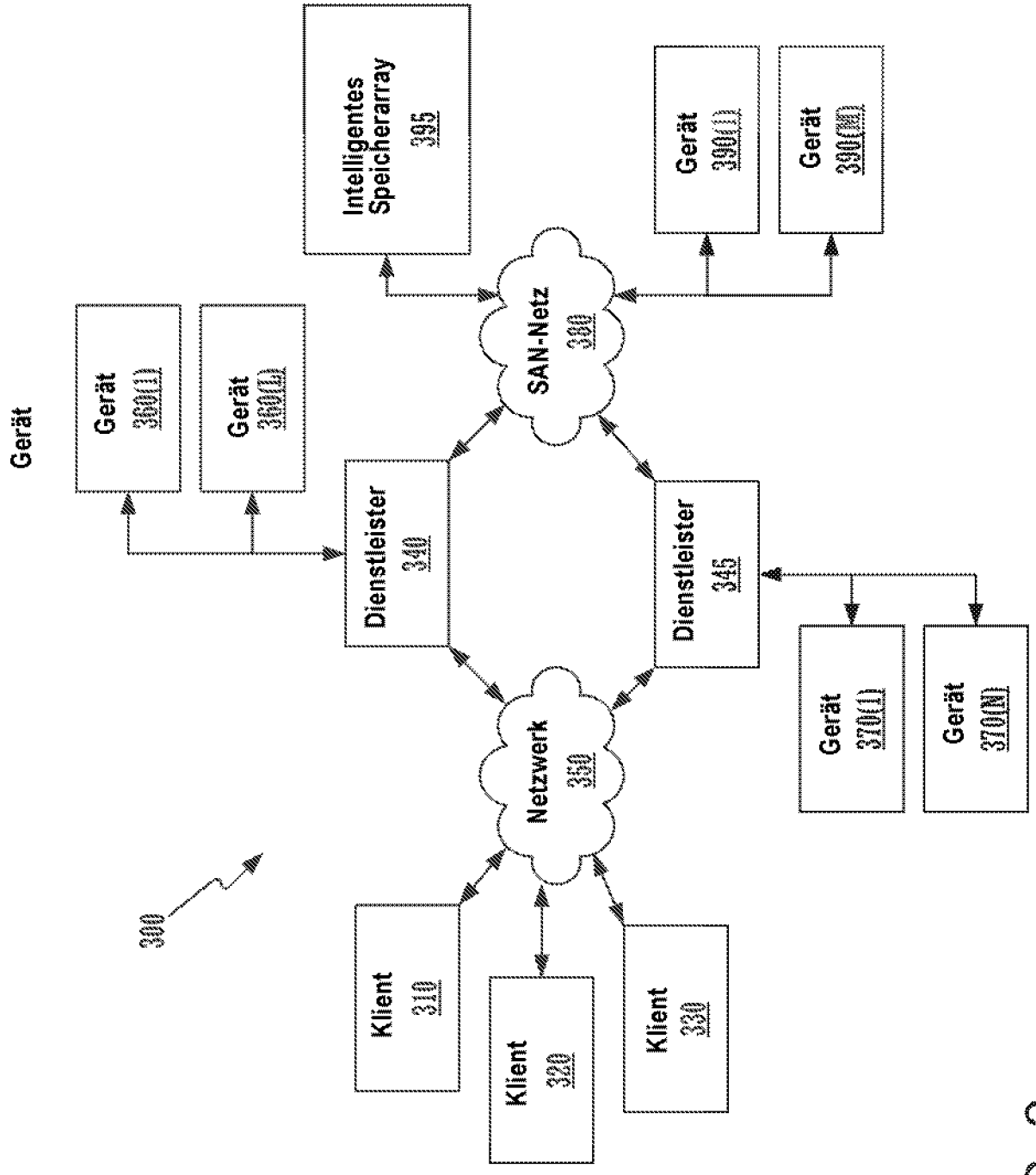

Gerät
Gerät 360(1)
Gerät 360(L)
Intelligentes Speicherarray 395
Gerät 390(1)
Gerät 390(M)
SAN-Netz 380
Dienstleister 340
Dienstleister 345
Gerät 370(1)
Gerät 370(N)
Netzwerk 350
300
Klient 310
Klient 320
Klient 330

FIG. 3

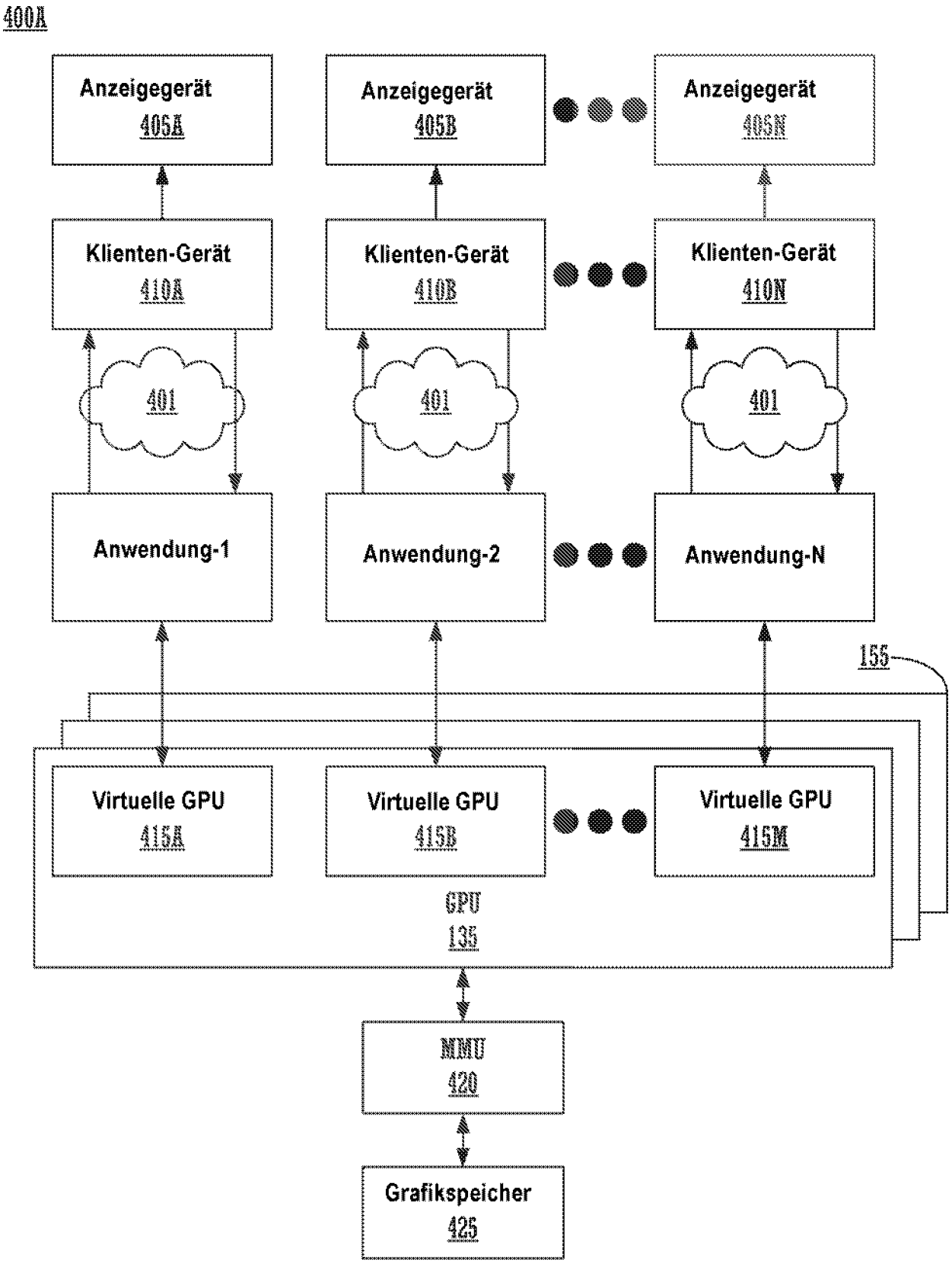

400A
Anzeigegerät
405A
Anzeigegerät
405B
Anzeigegerät
405N
Klienten-Gerät
410A
Klienten-Gerät
410B
Klienten-Gerät
410N
401
401
401
Anwendung-1
Anwendung-2
Anwendung-N
155
Virtuelle GPU
415A
Virtuelle GPU
415B
Virtuelle GPU
415M
GPU
135
MMU
420
Grafikspeicher
425

FIG. 4A

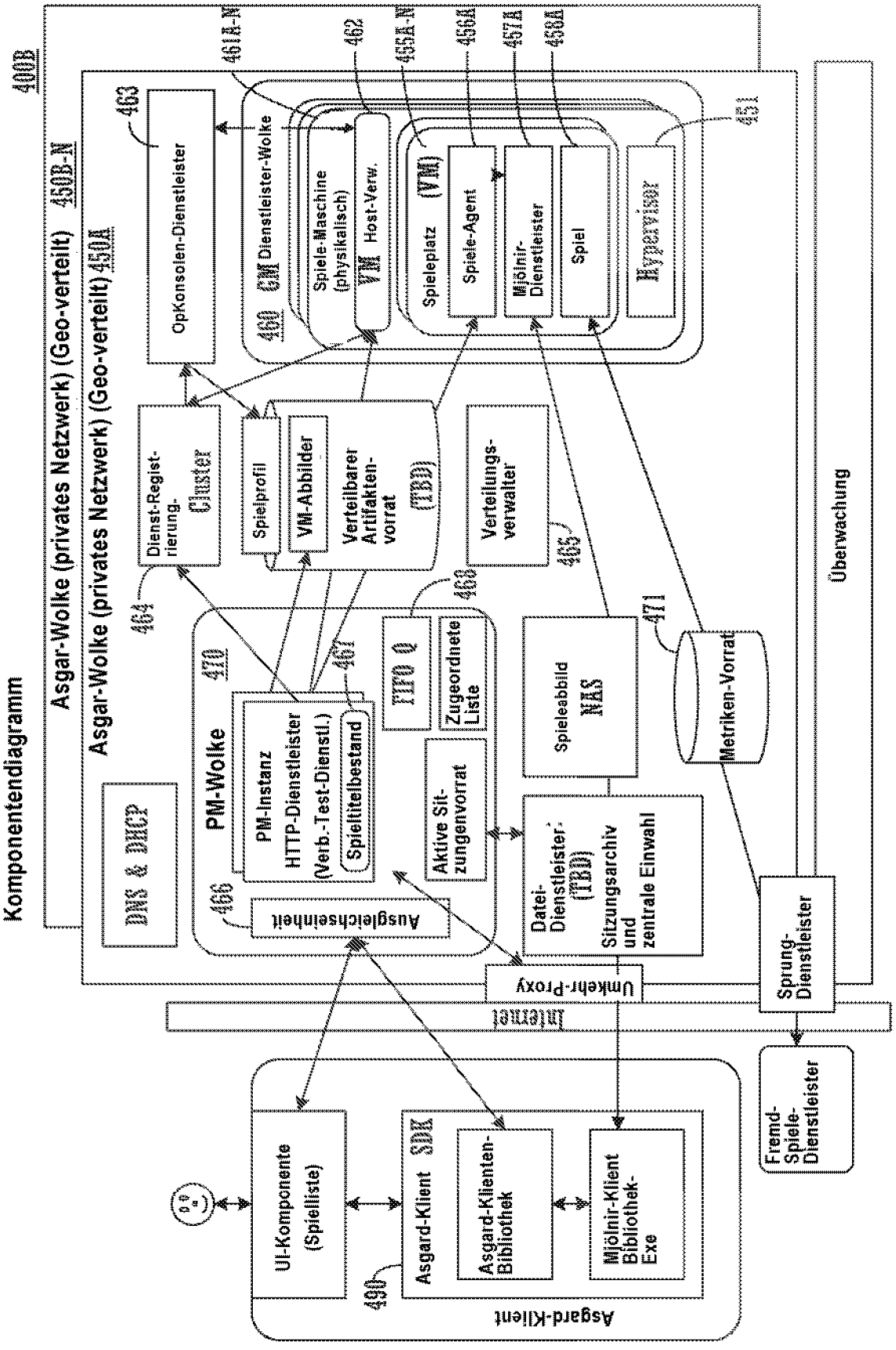
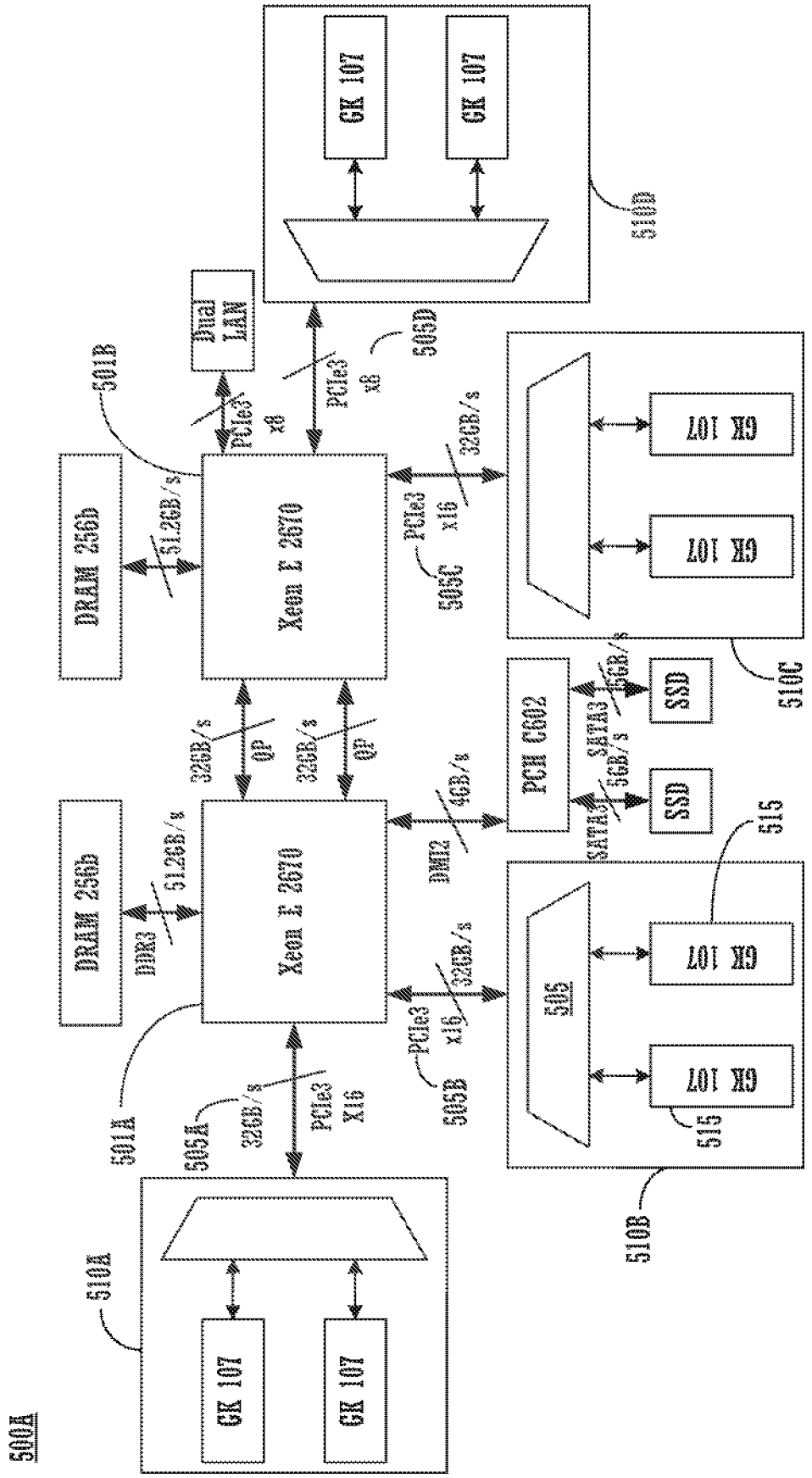
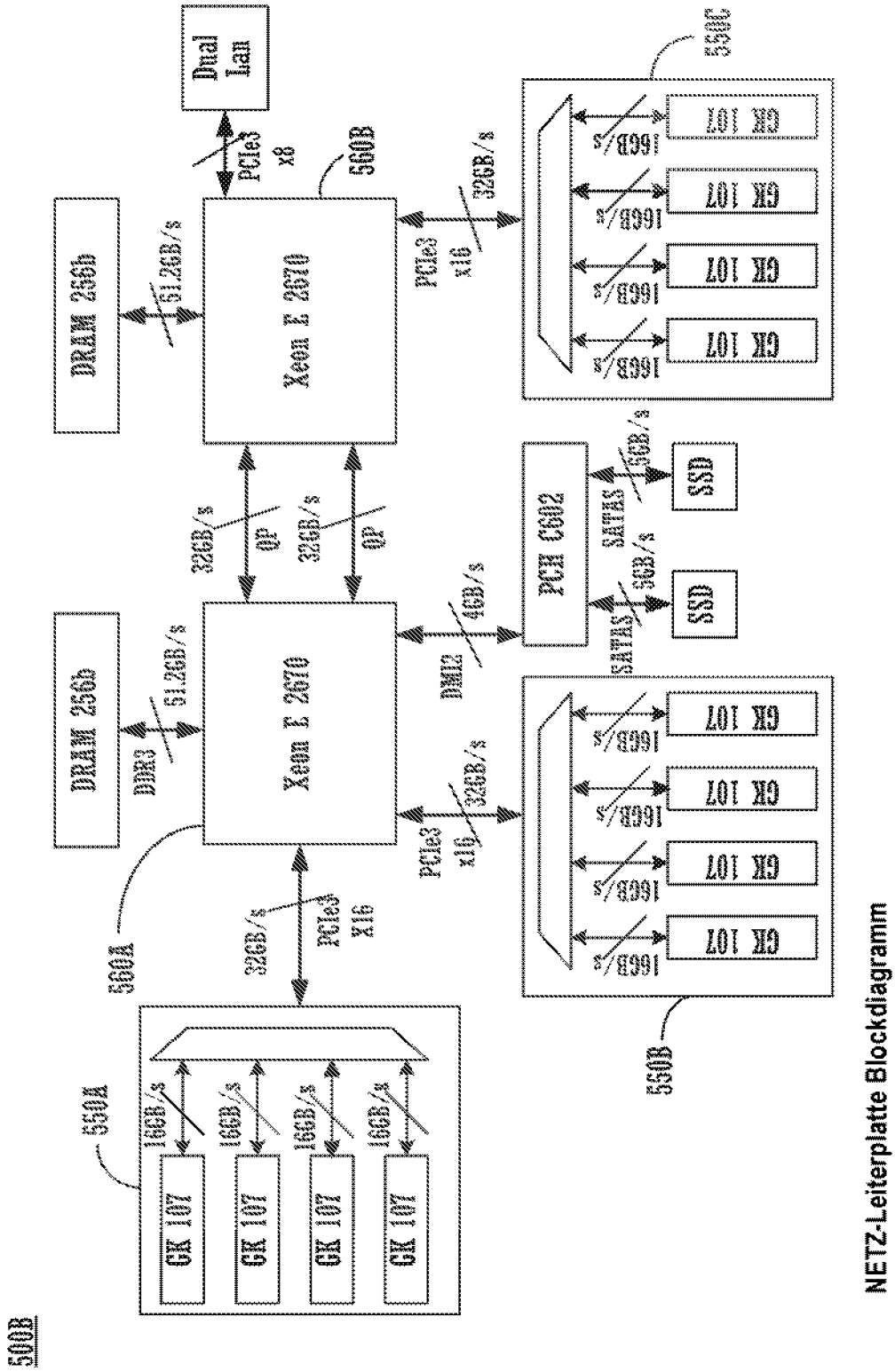

Komponentendiagramm
400B
Asgar-Wolke (privates Netzwerk) (Geo-verteilt) 450B-N
Asgar-Wolke (privates Netzwerk) (Geo-verteilt) 450A
461A-N
462
455A-N
456A
457A
458A
463
OpKonsolen-Dienstleister
460 CM Dienstleister-Wolke
Spiele-Maschine (physikalisch)
VM Host-Verw.
Spielepatz (VM)
Spiele-Agent
Mjölnir-Dienstleister
Spiel
Hypervisor
451
Dienst-Regist-rierung-Dienster
Spielprofil
VM-Abbilder
Verteilbarer Artifakten-vorrat (TBD)
Verteilungs-verwalter
464
465
468
PM-Wolke 410
PM-Instanz HTTP-Dienstleister (Verb.-Test-Dienstl.)
Spieltitelbestand
467
FIFO Q
Zugeordnete Liste
Aktive Sit-zungenvorrat
Spieleabbild NAS
471
Metriken-Vorrat
DNS & DHCP
466
Ausgleichseinheit
Datei-Dienstleister (TBD) Sitzungsarchiv und zentrale Einwahl
Überwachung
Umkehr-Proxy
Sprung-Dienstleister
Internet
Fremd-Spiele-Dienstleister
Asgard-Klient
UI-Komponente (Spielliste)
Asgard-Klient SDK
Asgard-Klienten-Bibliothek
Mjölnir-Klient Bibliothek-Exe
490

FIG. 4B



NETZ-Leiterplatte Blockdiagramm

FIG. 5A



NETZ-Leiterplatte Blockdiagramm

FIG. 5B

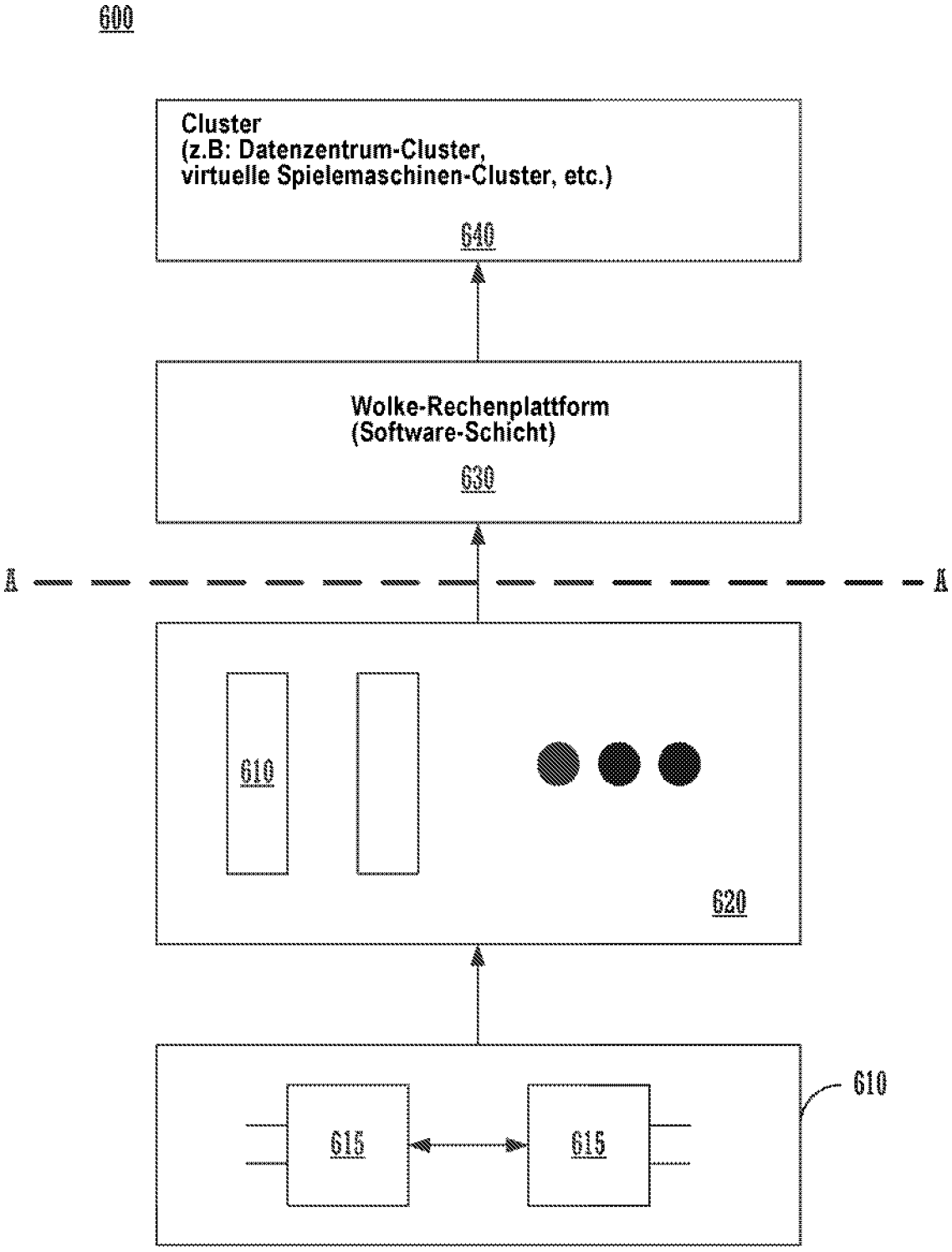

600
Cluster
(z.B: Datenzentrum-Cluster,
virtuelle Spielemaschinen-Cluster, etc.)
640
Wolke-Rechenplattform
(Software-Schicht)
630
A
A
610
620
610
615
615

FIG. 6

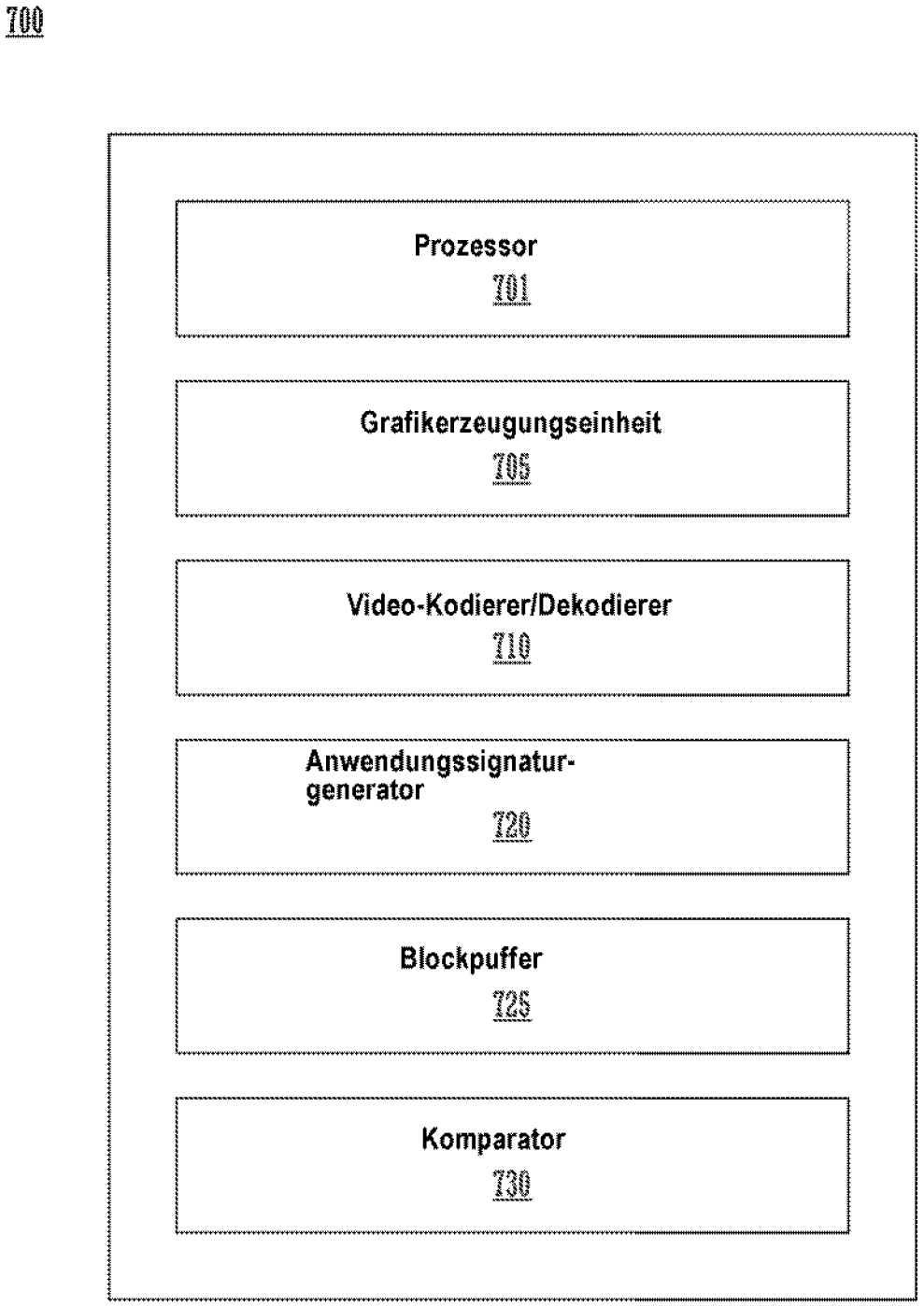

700
Prozessor
701
Grafikerzeugungseinheit
705
Video-Kodierer/Dekodierer
710
Anwendungssignatur-
generator
720
Blockpuffer
725
Komparator
730

FIG. 7

800A

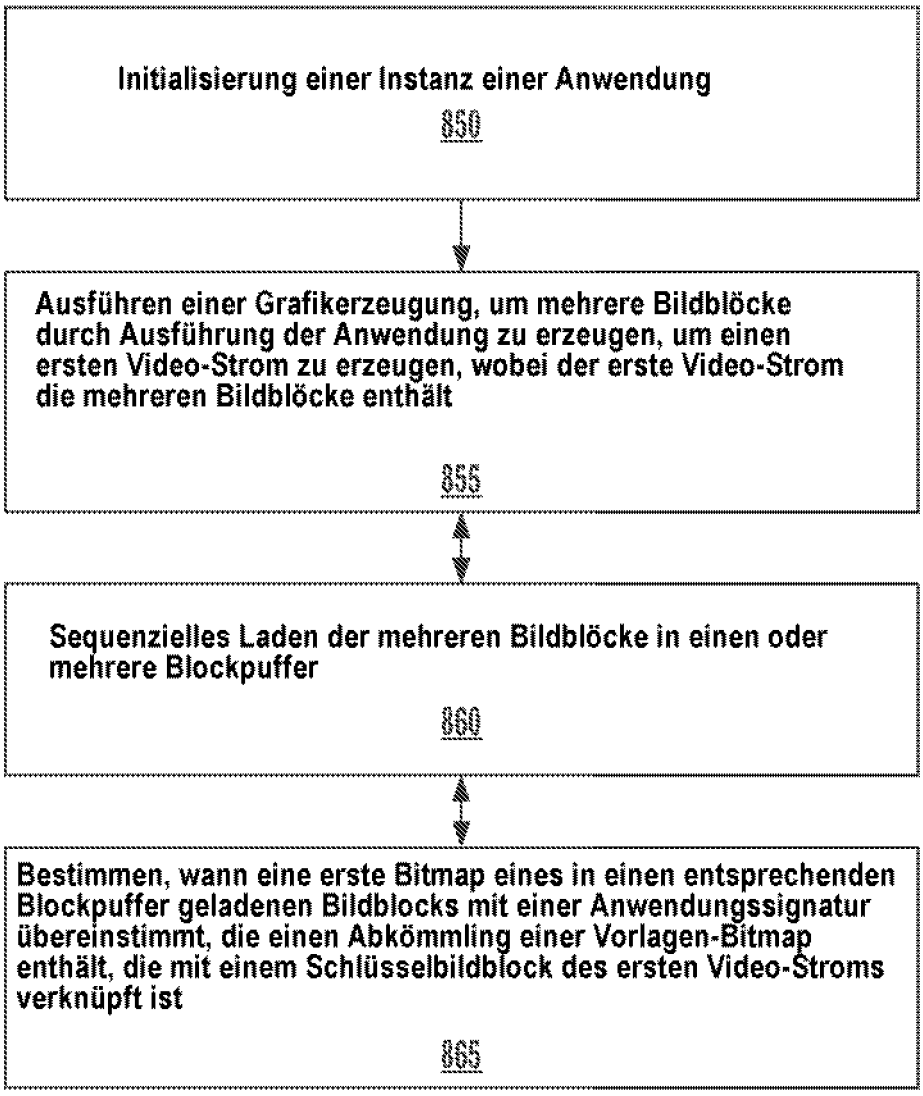

Initialisierung einer Instanz einer Anwendung
850
Ausführen einer Grafikerzeugung, um mehrere Bildblöcke durch Ausführung der Anwendung zu erzeugen, um einen ersten Video-Strom zu erzeugen, wobei der erste Video-Strom die mehreren Bildblöcke enthält
855
Sequenzielles Laden der mehreren Bildblöcke in einen oder mehrere Blockpuffer
860
Bestimmen, wann eine erste Bitmap eines in einen entsprechenden Blockpuffer geladenen Bildblocks mit einer Anwendungssignatur übereinstimmt, die einen Abkömmling einer Vorlagen-Bitmap enthält, die mit einem Schlüsselbildblock des ersten Video-Stroms verknüpft ist
865

FIG. 8A

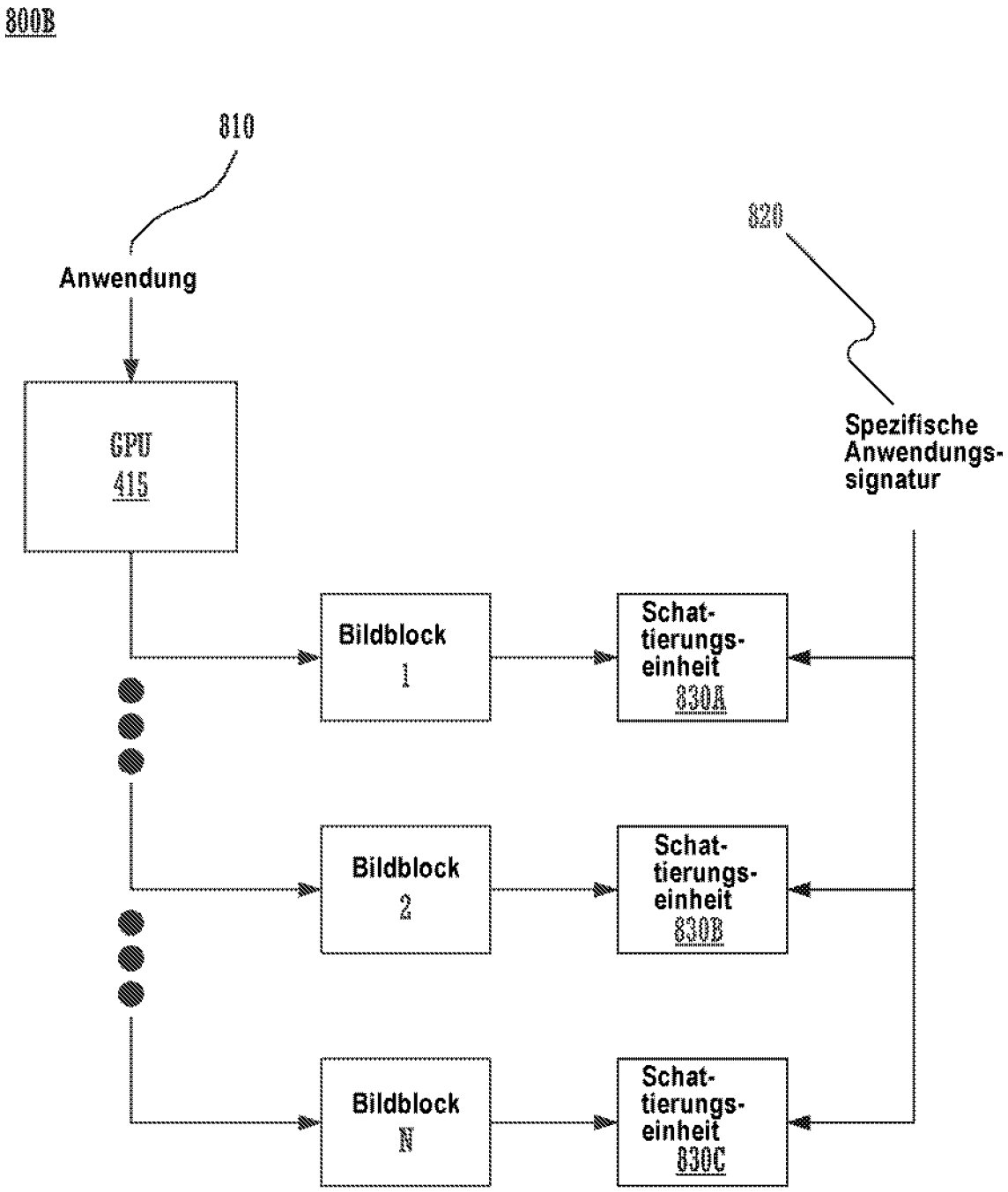

800B
810
Anwendung
820
Spezifische Anwendungs-signatur
GPU 415
Bildblock 1
Schat-tierungs-einheit 830A
Bildblock 2
Schat-tierungs-einheit 830B
Bildblock N
Schat-tierungs-einheit 830C

FIG. 8B

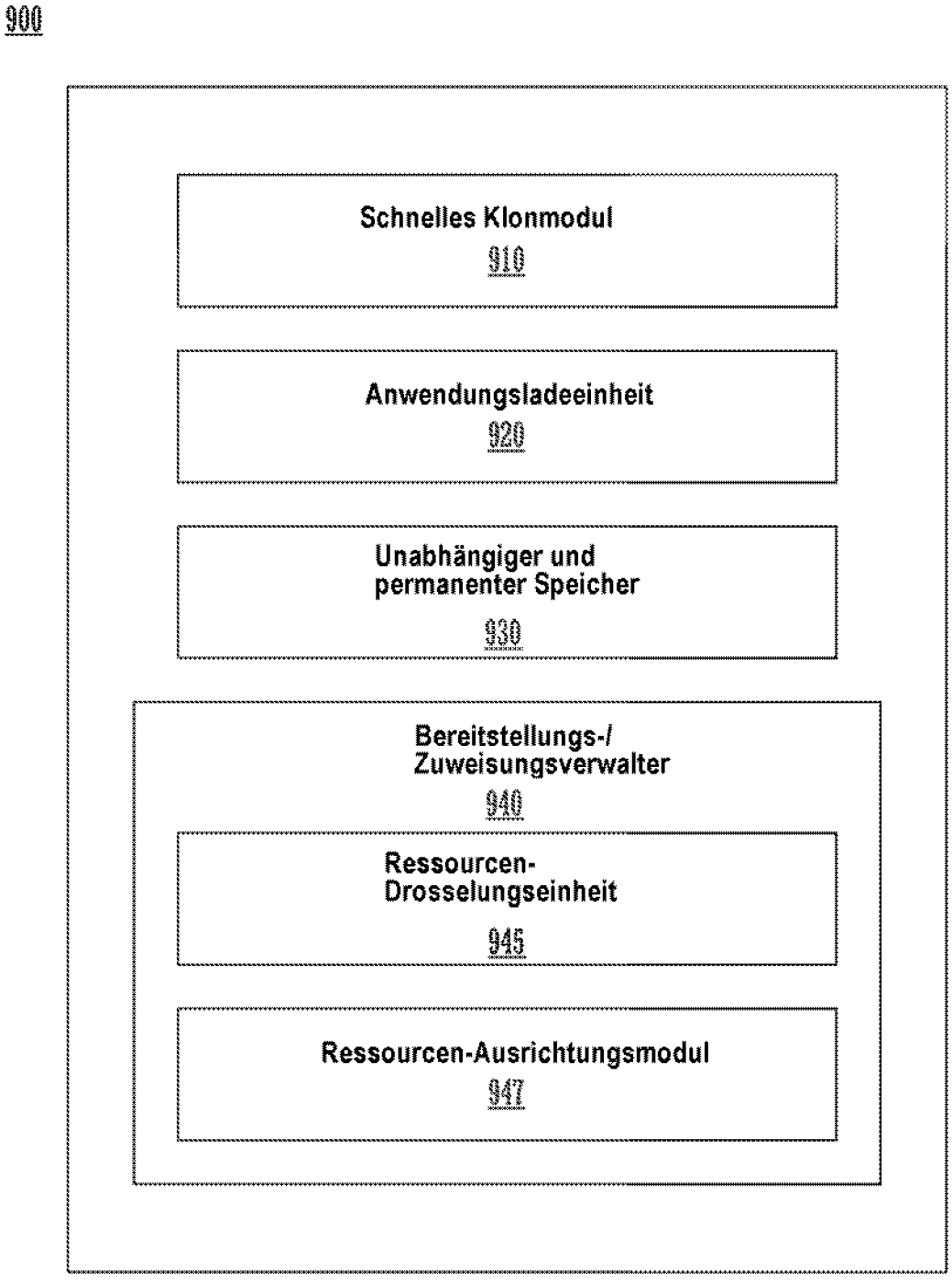

900
Schnelles Klonmodul
910
Anwendungsladeeinheit
920
Unabhängiger und permanenter Speicher
930
Bereitstellungs-/ Zuweisungsverwalter
940
Ressourcen-Drosselungseinheit
945
Ressourcen-Ausrichtungsmodul
947

FIG. 9

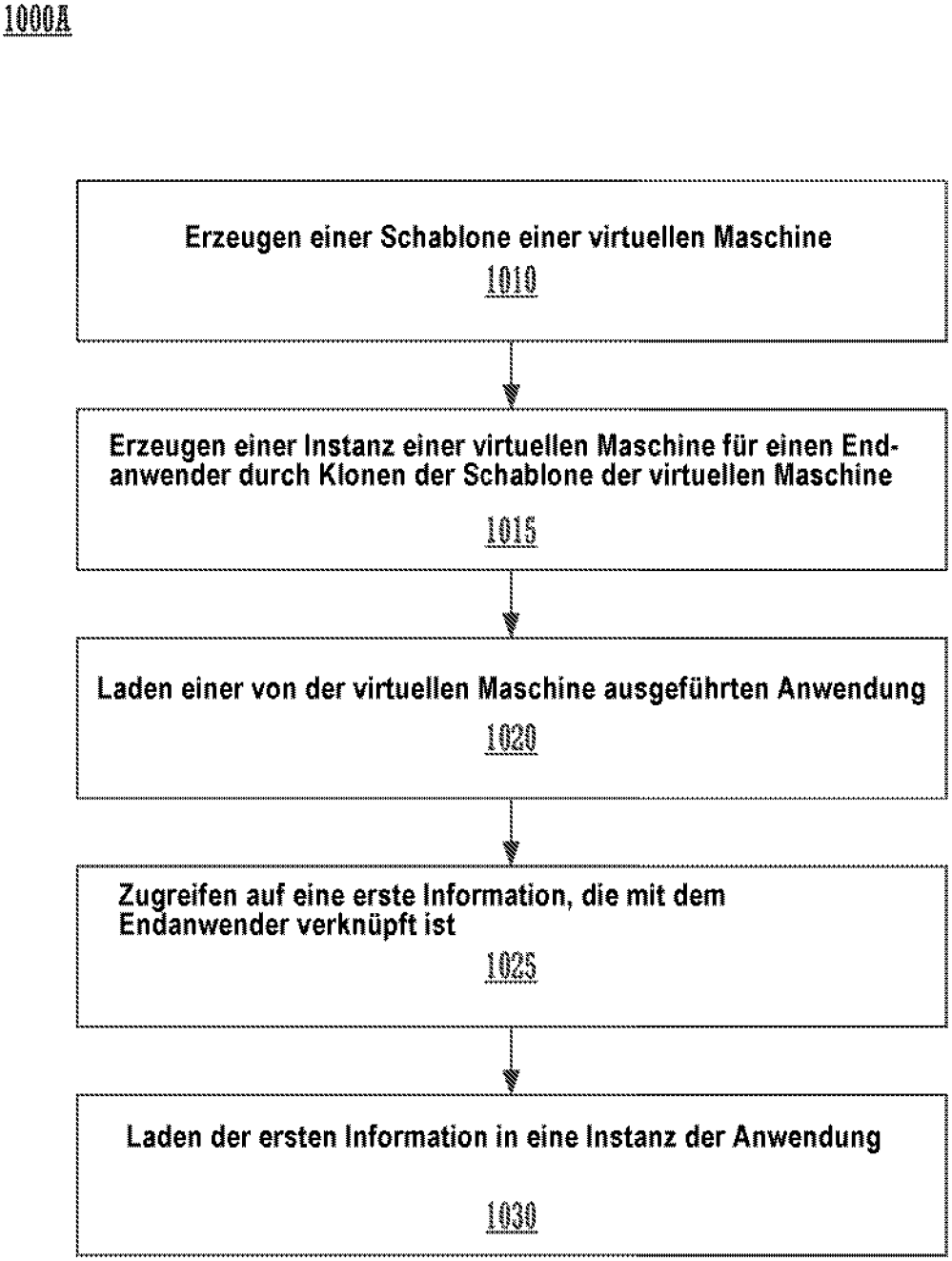

1000A
Erzeugen einer Schablone einer virtuellen Maschine
1010
Erzeugen einer Instanz einer virtuellen Maschine für einen Endanwender durch Klonen der Schablone der virtuellen Maschine
1015
Laden einer von der virtuellen Maschine ausgeführten Anwendung
1020
Zugreifen auf eine erste Information, die mit dem Endanwender verknüpft ist
1025
Laden der ersten Information in eine Instanz der Anwendung
1030

FIG. 10A

1000B

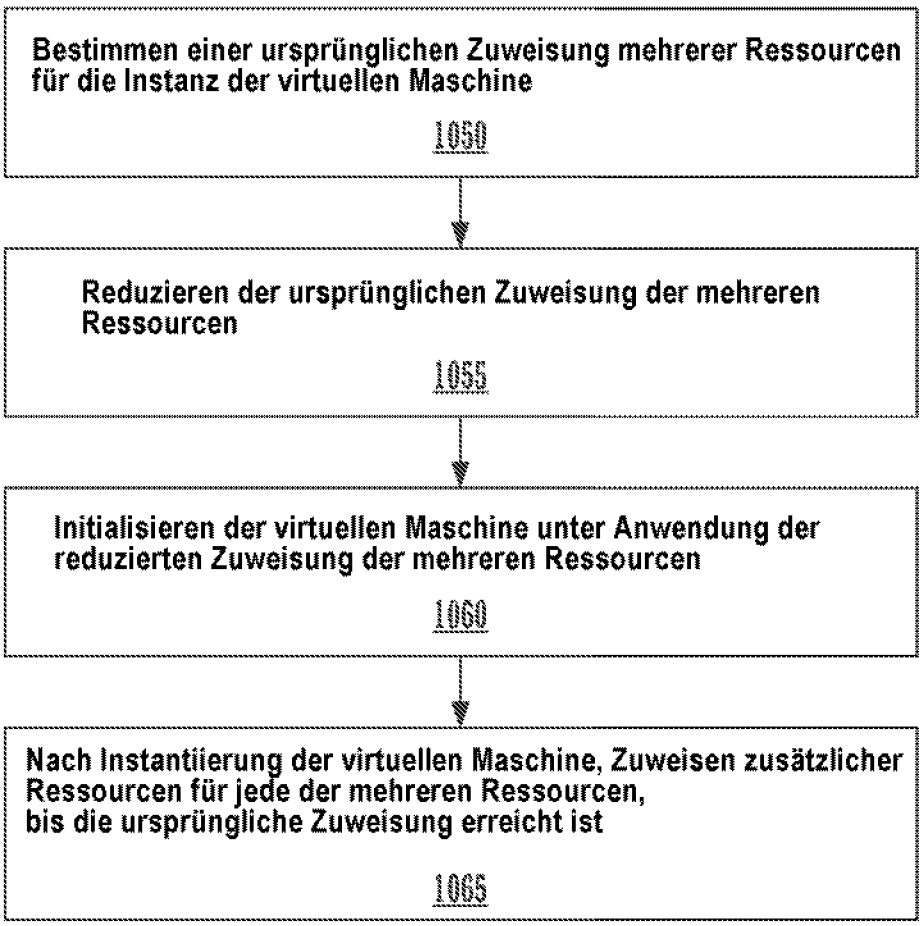


FIG. 10B

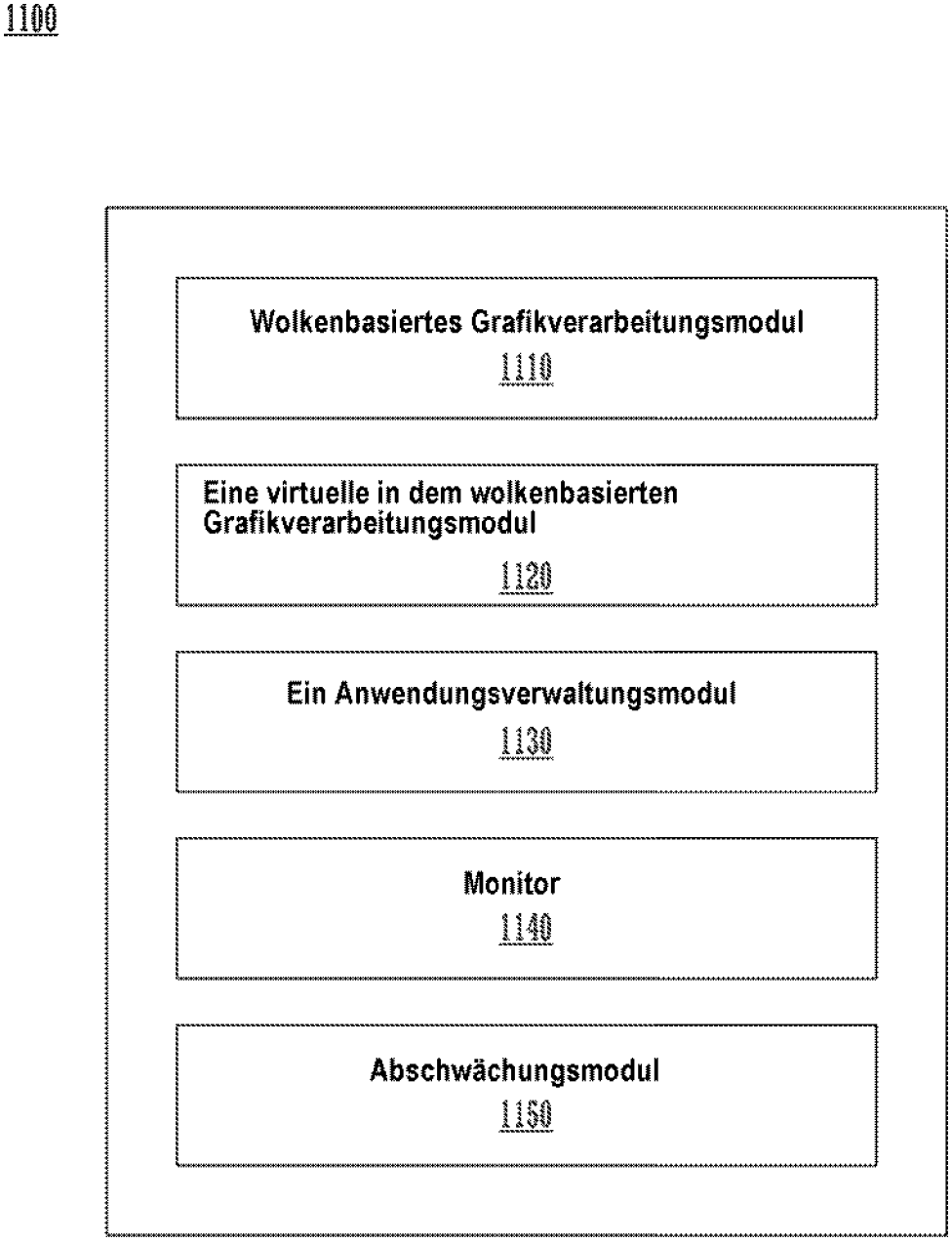

1100
Wolkenbasiertes Grafikverarbeitungsmodul
1110
Eine virtuelle in dem wolkenbasierten Grafikverarbeitungsmodul
1120
Ein Anwendungsverwaltungsmodul
1130
Monitor
1140
Abschwächungsmodul
1150

FIG. 11

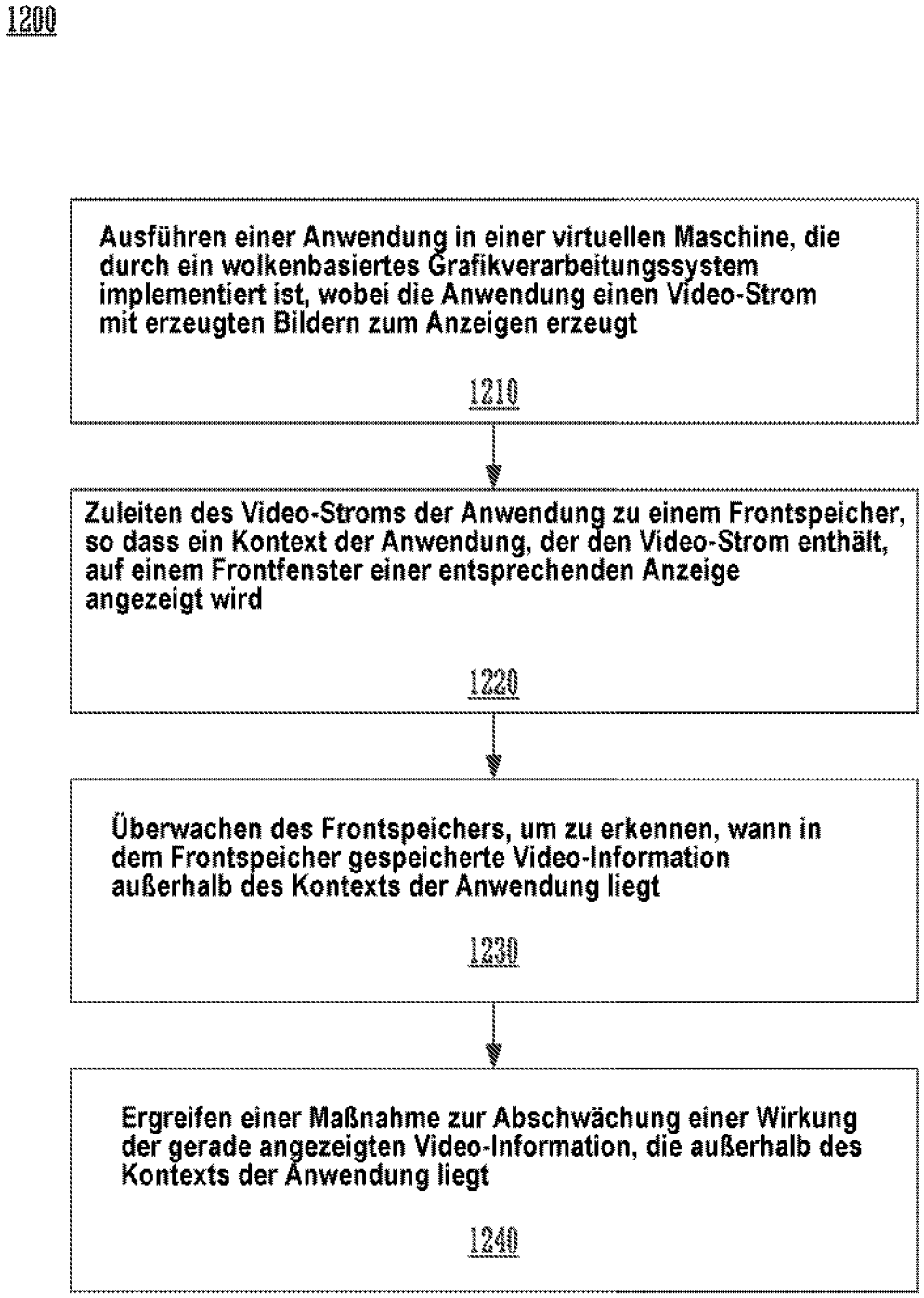

1200
Ausführen einer Anwendung in einer virtuellen Maschine, die durch ein wolkenbasiertes Grafikverarbeitungssystem implementiert ist, wobei die Anwendung einen Video-Strom mit erzeugten Bildern zum Anzeigen erzeugt
1210
Zuleiten des Video-Stroms der Anwendung zu einem Frontspeicher, so dass ein Kontext der Anwendung, der den Video-Strom enthält, auf einem Frontfenster einer entsprechenden Anzeige angezeigt wird
1220
Überwachen des Frontspeichers, um zu erkennen, wann in dem Frontspeicher gespeicherte Video-Information außerhalb des Kontexts der Anwendung liegt
1230
Ergreifen einer Maßnahme zur Abschwächung einer Wirkung der gerade angezeigten Video-Information, die außerhalb des Kontexts der Anwendung liegt
1240

FIG. 12

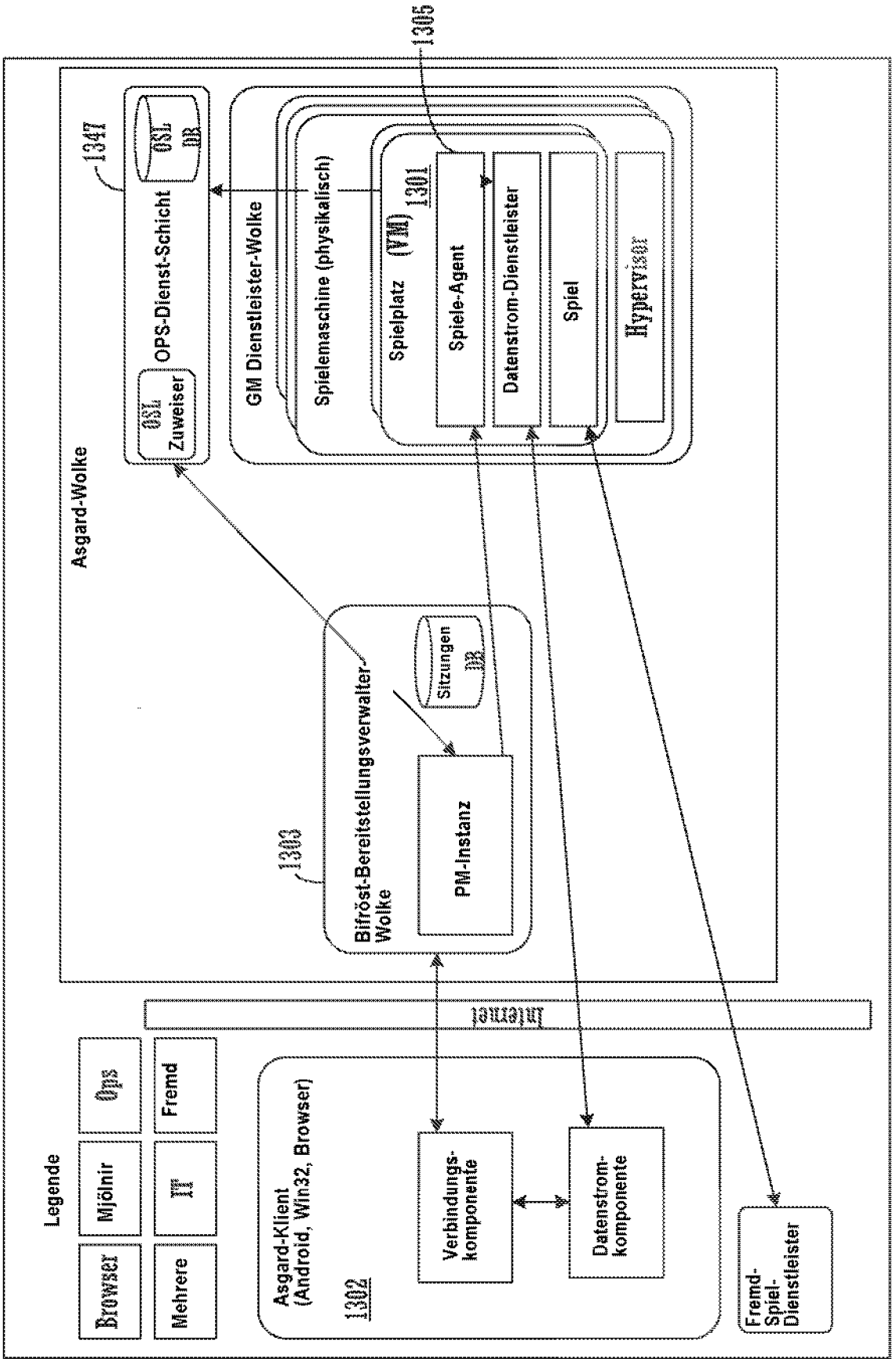

1300A
Asgard-Wolke
1347
OPS-Dienst-Schicht
OSL DB
OSL Zuweiser
GM Dienstleister-Wolke
Spielemaschine (physikalisch)
Spielplatz (VM) 1301
Spiele-Agent
Datenstrom-Dienstleister
Spiel
Hypervisor
1305
Bifröst-Bereitstellungsverwalter-Wolke
1303
Sitzungen DB
PM-Instanz
Internet
Asgard-Klient (Android, Win32, Browser)
1302
Verbindungs-komponente
Datenstrom-komponente
Fremd-Spiel-Dienstleister
Legende
Browser
Mjölnir
Ops
Mehrere
IT
Fremd

FIG. 13A

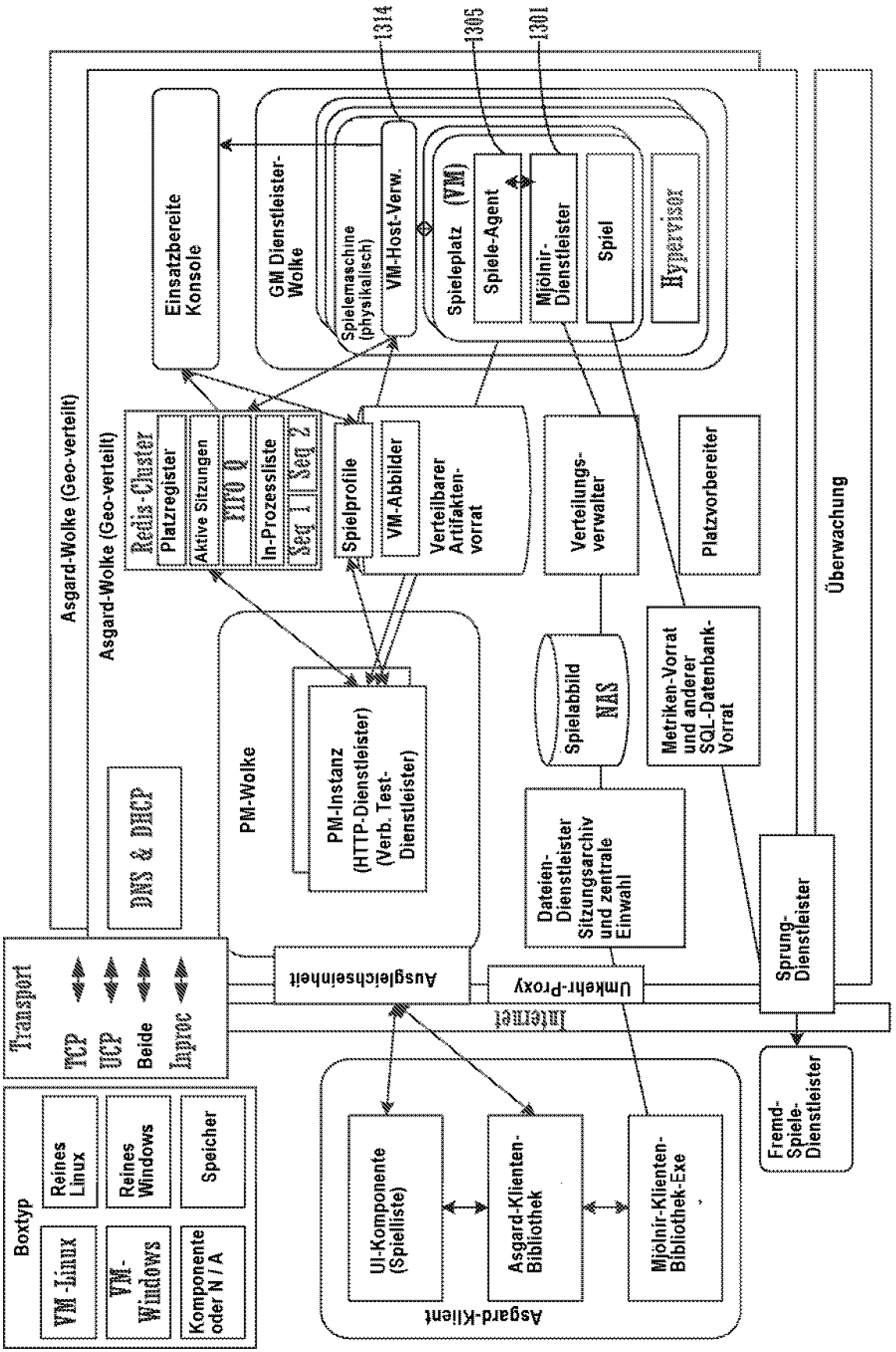

Asgard-Wolke (Geo-verteilt)
Asgard-Wolke (Geo-verteilt)
Einsatzbereite Konsole
GM Dienstleister-Wolke
Spielemaschine (physikalisch)
VM-Host-Verw.
Spieleplatz (VM)
Spiele-Agent
Mjölnir-Dienstleister
Spiel
Hypervisor
1314
1305
1301
Redis-Cluster
Platzregister
Aktive Sitzungen
In-Prozessliste
Seq 1
Seq 2
Spielprofile
VM-Abbilder
Verteilbarer Artifakten-vorrat
Verteilungs-verwalter
Platzvorbereiter
DNS & DHCP
PM-Wolke
PM-Instanz (HTTP-Dienstleister) (Verb. Test-Dienstleister)
Spielabbild NAS
Metriken-Vorrat und anderer SQL-Datenbank-Vorrat
Dateien-Dienstleister Sitzungsarchiv und zentrale Einwahl
Überwachung
Transport
TCP
UCP
Beide
Lnproc
Ausgleichseinheit
Umkehr-Proxy
Internet
Sprung-Dienstleister
Boxtyp
VM-Linux
VM-Windows
Komponente oder N/A
Reines Linux
Reines Windows
Speicher
Asgard-Klient
UI-Komponente (Spielliste)
Asgard-Klienten-Bibliothek
Mjölnir-Klienten-Bibliothek-Exe
Fremd-Spiele-Dienstleister

FIG. 13B

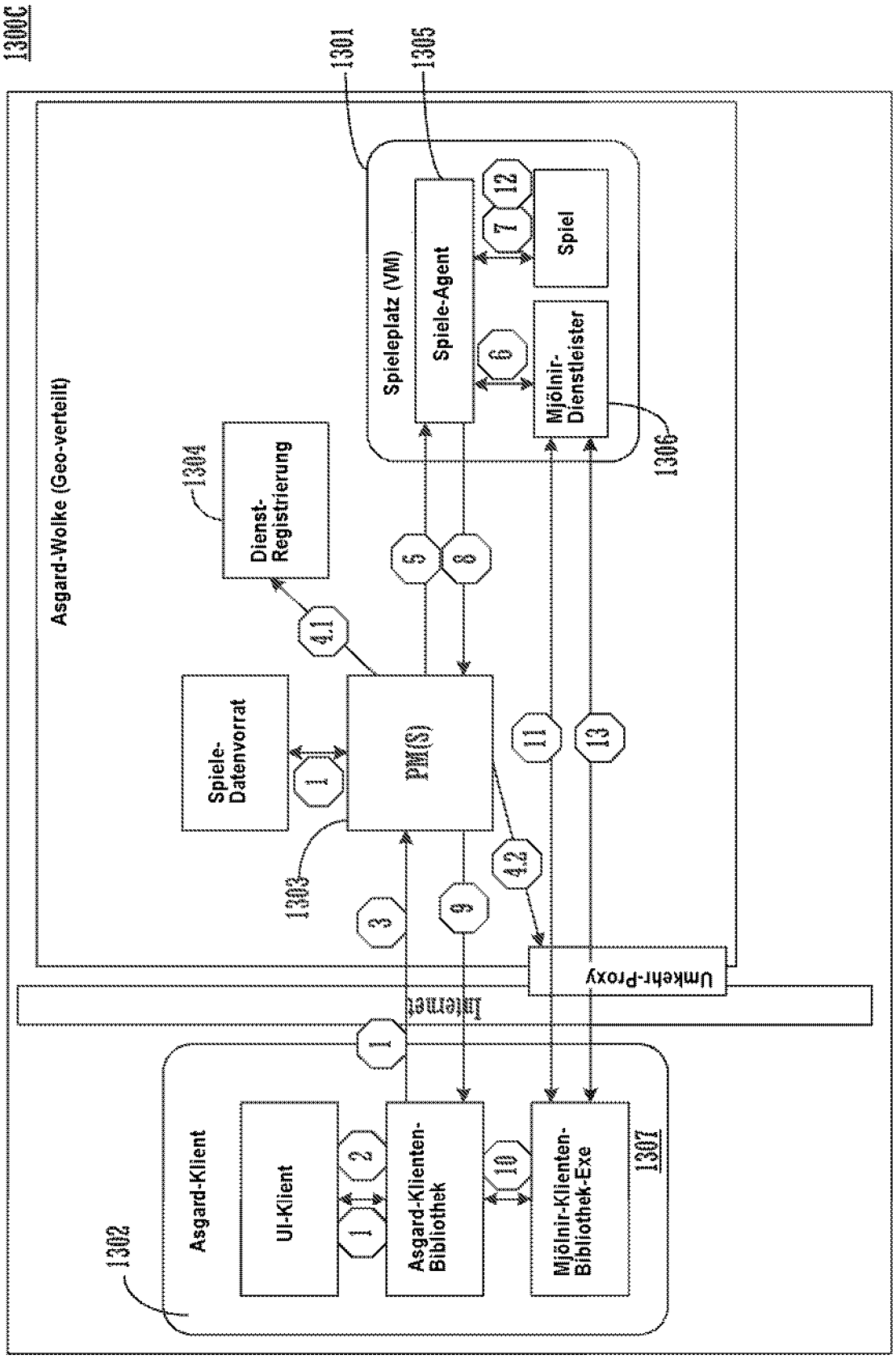

1300C
1301
1305
Asgard-Wolke (Geo-verteilt)
Spieleplatz (VM)
Spiele-Agent
Spiel
Mjölnir-Dienstleister
1306
1304
Dienst-Registrierung
Spiele-Datenvorrat
PM(S)
1303
Umkehr-Proxy
Asgard-Klient
UI-Klient
Asgard-Klienten-Bibliothek
Mjölnir-Klienten-Bibliothek-Exe
1307
1302

FIG. 13C

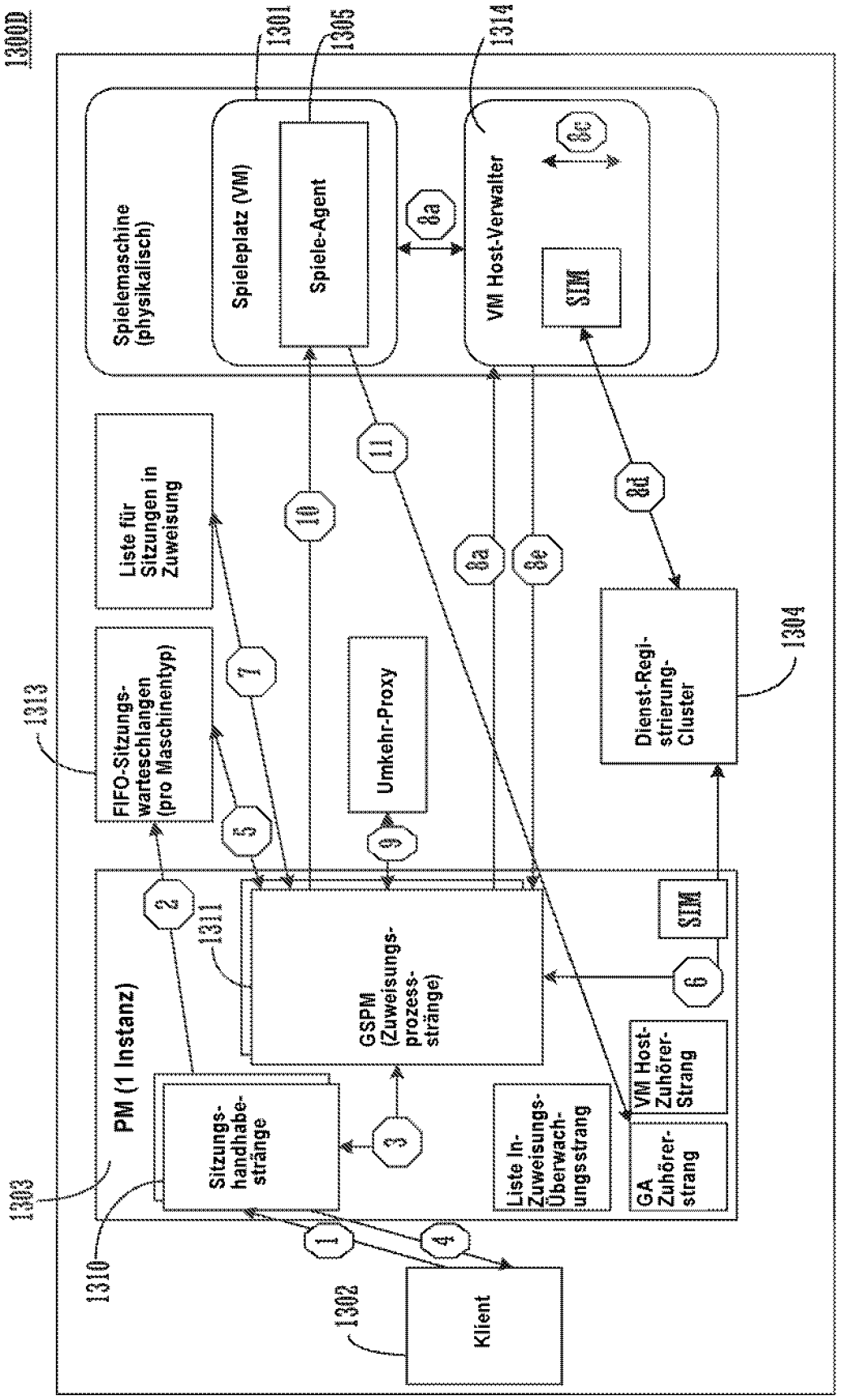

1300D
1301
1305
1314
Spielemaschine (physikalisch)
Spieleplatz (VM)
Spiele-Agent
VM Host-Verwalter
SIM
8a
8c
8d
8e
11
10
7
5
9
2
6
3
4
1
Liste für Sitzungen in Zuweisung
FIFO-Sitzungs-warteschlangen (pro Maschinentyp)
Umkehr-Proxy
Dienst-Regi-strierung-Cluster
1304
1313
1311
GSPM (Zuweisungs-prozess-stränge)
PM (1 Instanz)
Sitzungs-handhabe-stränge
Liste In-Zuweisungs-Überwach-ungsstrang
VM Host-Zuhörer-Strang
GA Zuhörer-strang
1303
1310
1302
Klient

FIG. 13D

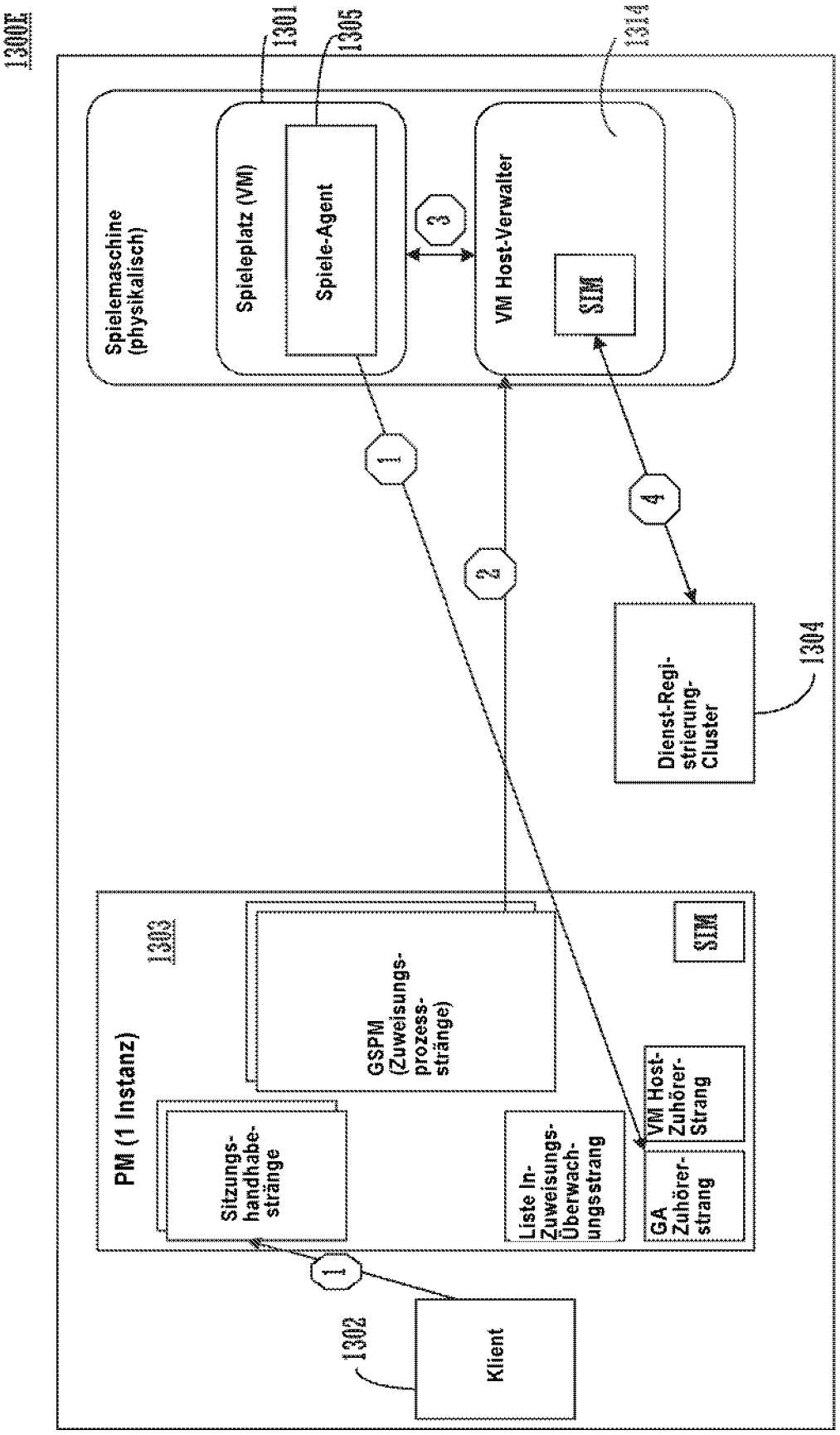

1300E
1301
1305
1314
Spielemaschine (physikalisch)
Spieleplatz (VM)
Spiele-Agent
VM Host-Verwalter
SIM
1
2
3
4
Dienst-Regi-strierung-Cluster
1304
PM (1 Instanz)
1303
GSPM (Zuweisungs-prozess-stränge)
SIM
Sitzungs-handhabe-stränge
Liste In-Zuweisungs-Überwach-ungsstrang
VM Host-Zuhörer-Strang
GA Zuhörer-strang
1
1302
Klient

FIG. 13E

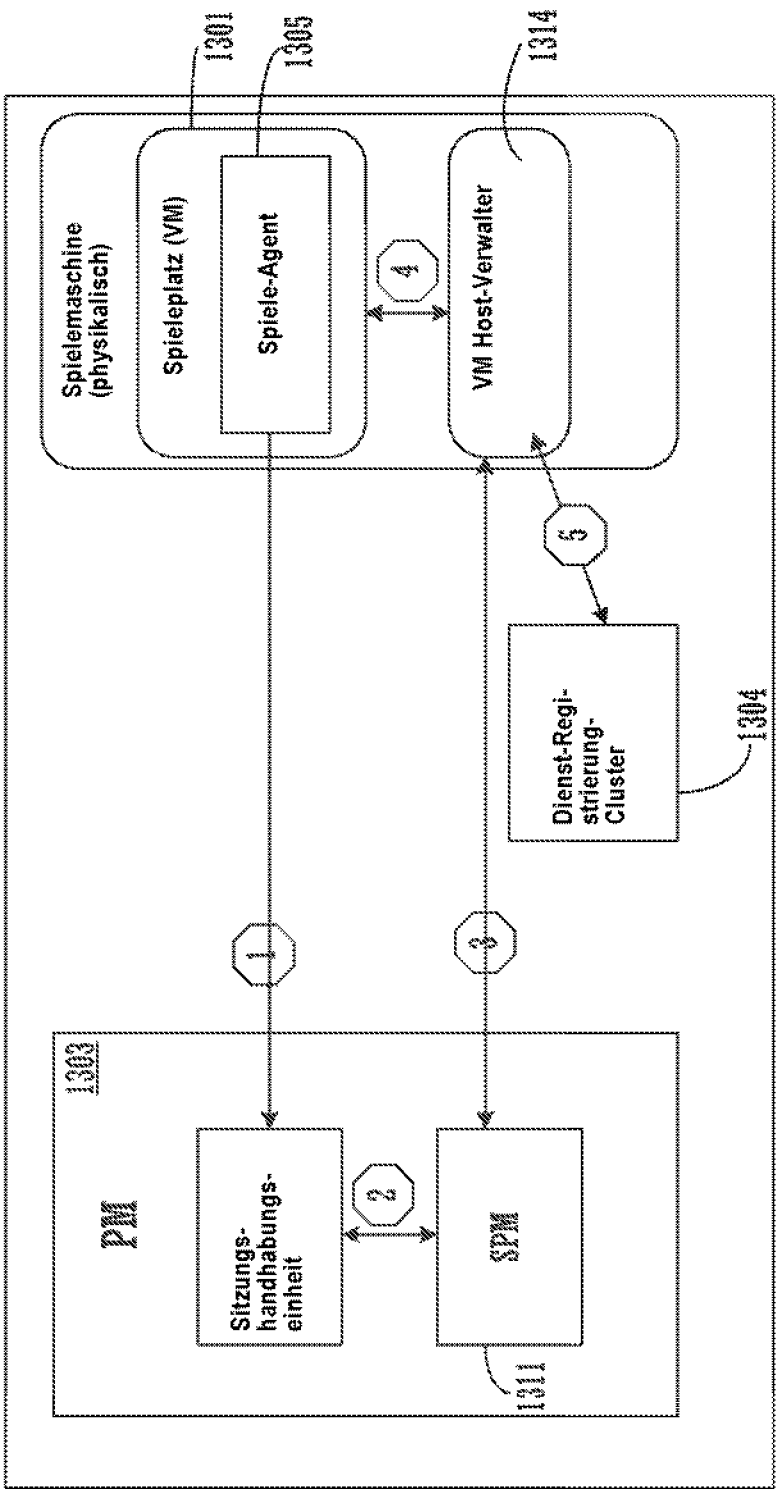

1300F
1301
1305
1314
Spielemaschine (physikalisch)
Spieleplatz (VM)
Spiele-Agent
VM Host-Verwalter
4
5
Dienst-Regi-strierung-Cluster
1304
1
3
1303
PM
Sitzungs-handhabungs-einheit
2
SPM
1311

FIG. 13F

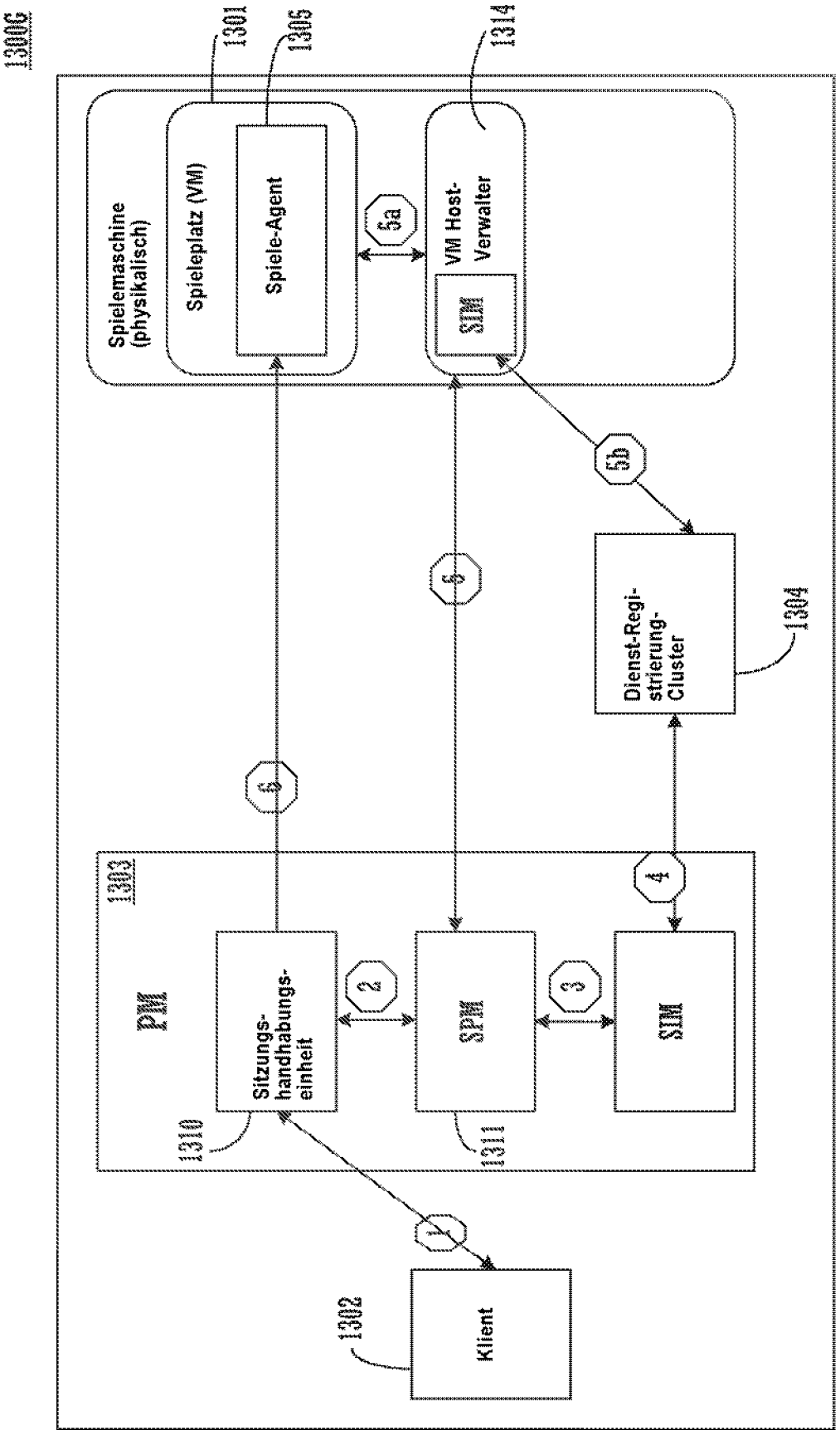

13006
1301
1305
1314
Spielemaschine (physikalisch)
Spieleplatz (VM)
Spiele-Agent
VM Host-Verwalter
SIM
5a
5b
6
Dienst-Regi-strierung-Cluster
1304
1303
PM
Sitzungs-handhabungs-einheit
SPM
SIM
1310
1311
1302
Klient
1
2
3
4

FIG. 13G

1300H

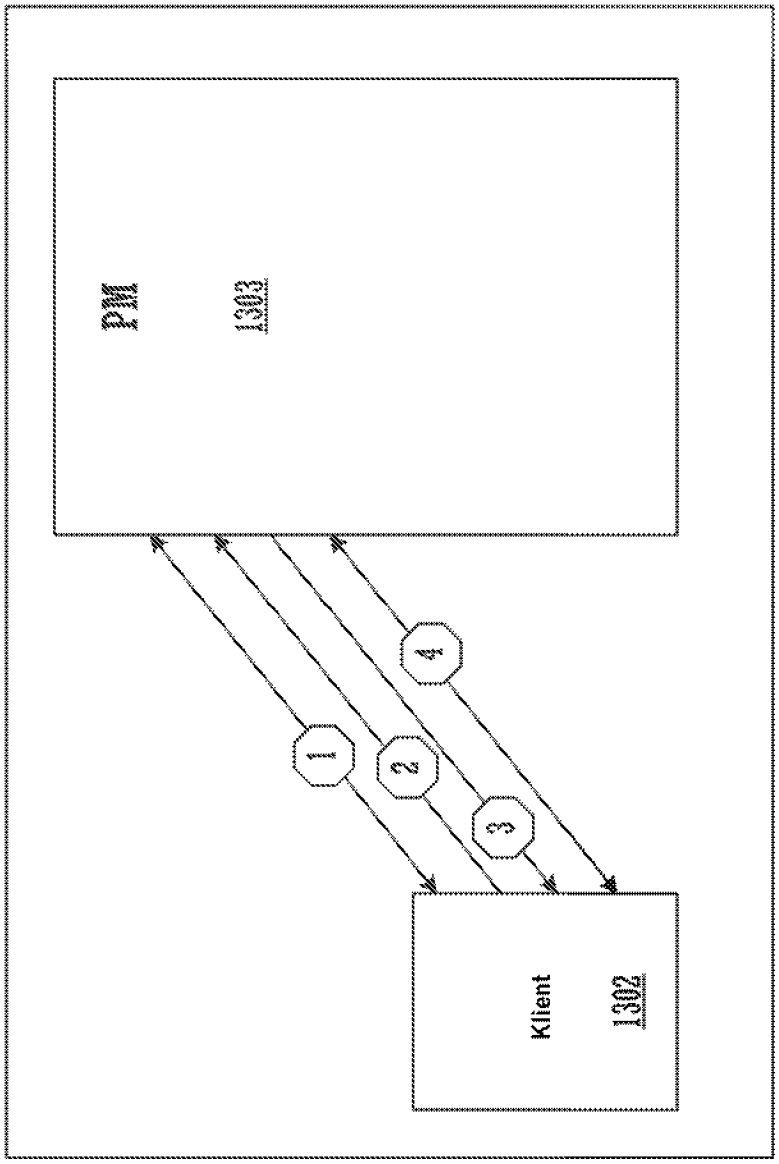


FIG. 13H

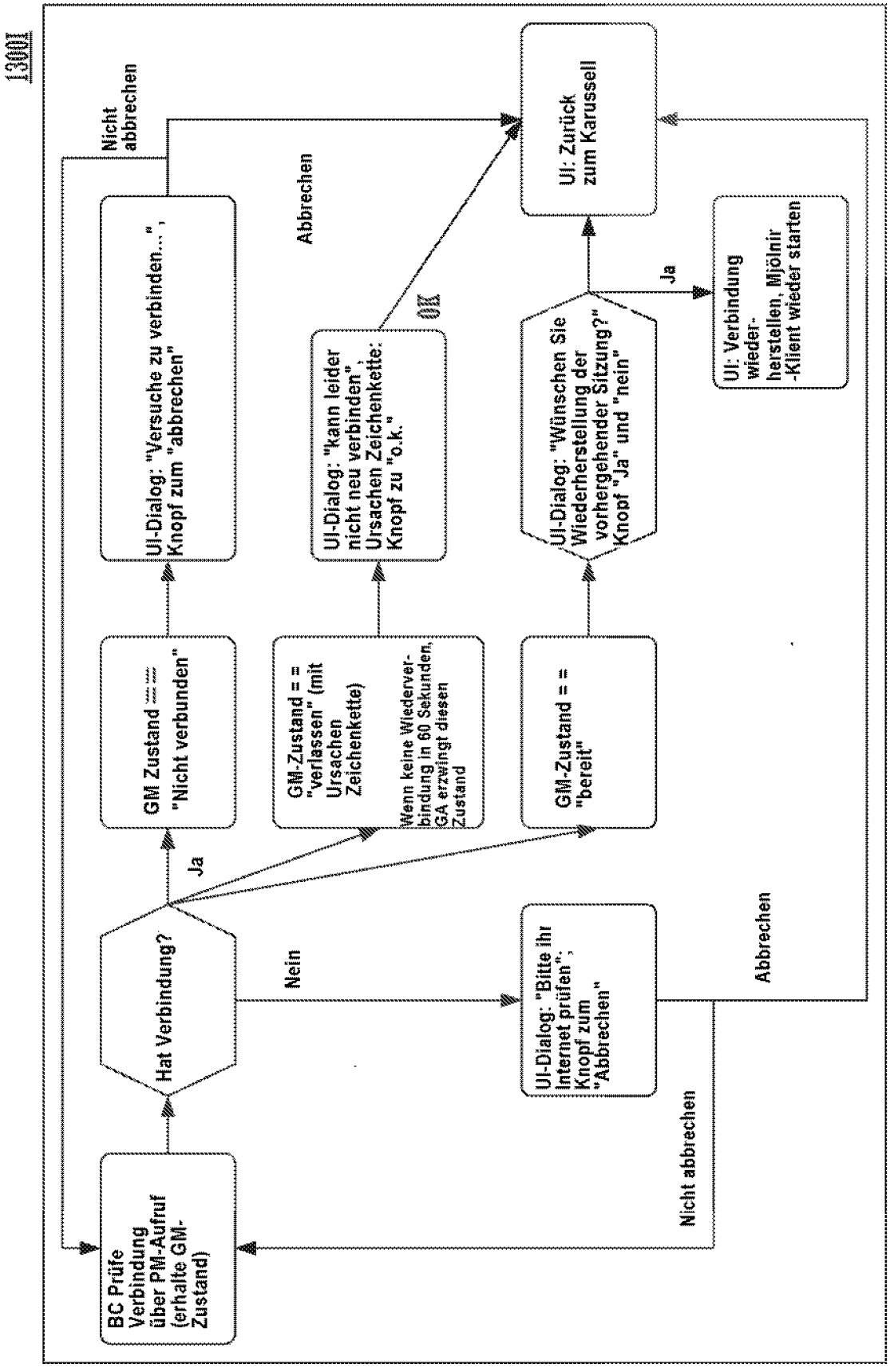

1300I
BC Prüfe Verbindung über PM-Aufruf (erhalte GM-Zustand)
Hat Verbindung?
Ja
Nein
GM Zustand = = "Nicht verbunden"
UI-Dialog: "Versuche zu verbinden...", Knopf zum "abbrechen"
Nicht abbrechen
Abbrechen
GM-Zustand = = "verlassen" (mit Ursachen Zeichenkette)
Wenn keine Wiederverbindung in 60 Sekunden, GA erzwingt diesen Zustand
UI-Dialog: "kann leider nicht neu verbinden", Ursachen Zeichenkette: Knopf zu "o.k."
OK
UI: Zurück zum Karussell
GM-Zustand = = "bereit"
UI-Dialog: "Bitte ihr Internet prüfen", Knopf zum "Abbrechen"
Nicht abbrechen
Abbrechen
UI-Dialog: "Wünschen Sie Wiederherstellung der vorhergehender Sitzung?" Knopf "Ja" und "nein"
Ja
UI: Verbindung wieder-herstellen, Mjolnir -Klient wieder starten

FIG. 13I

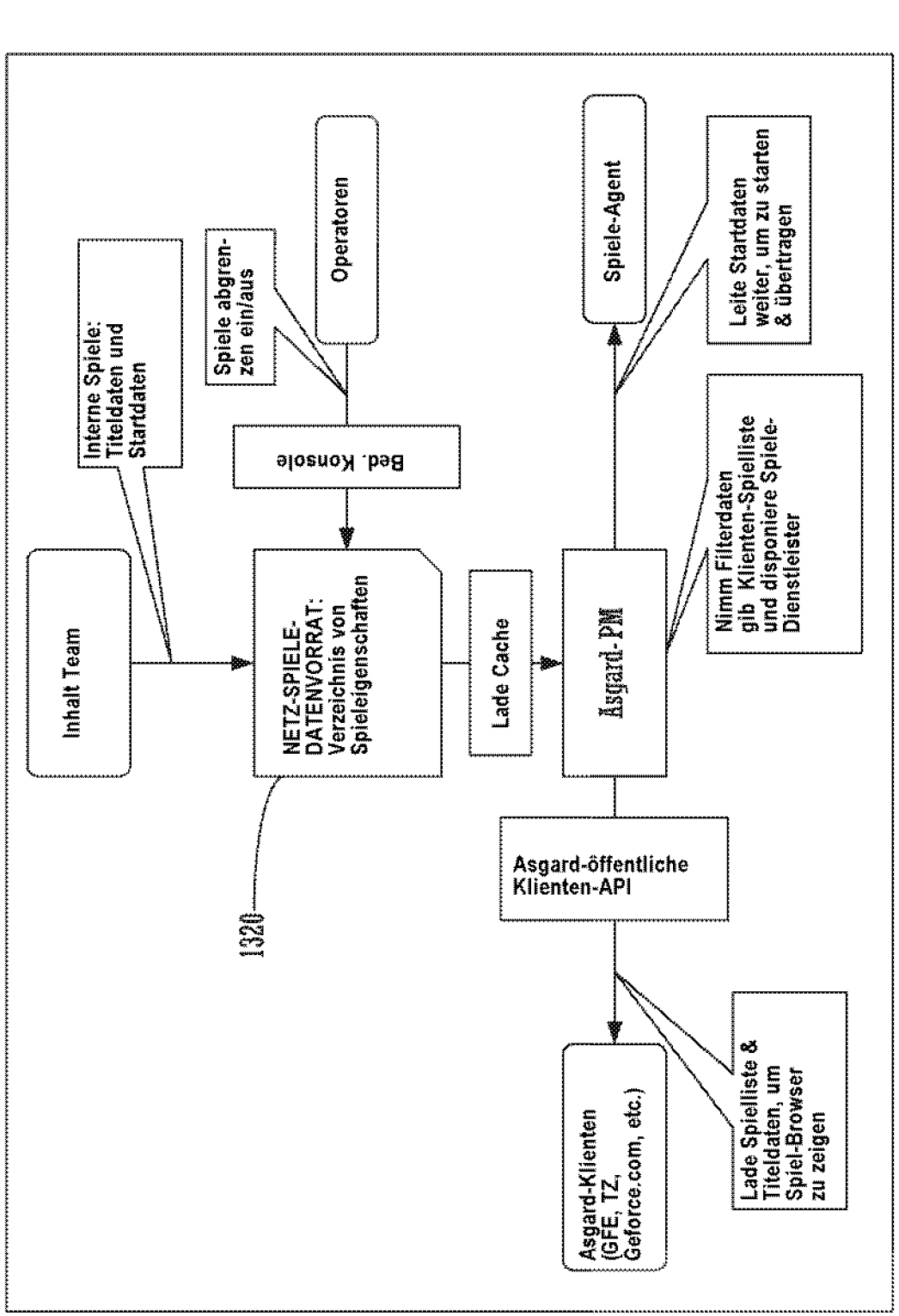

1300
Interne Spiele: Titeldaten und Startdaten
Spiele abgrenzen ein/aus
Operatoren
Spiele-Agent
Leite Startdaten weiter, um zu starten & übertragen
Bed. Konsole
Inhalt Team
NETZ-SPIELE-DATENVORRAT: Verzeichnis von Spieleigenschaften
Lade Cache
Asgard-PM
Nimm Filterdaten gib Klienten-Spielliste und disponiere Spiele-Dienstleister
1320
Asgard-öffentliche Klienten-API
Asgard-Klienten (GFE, TZ, Geforce.com, etc.)
Lade Spielliste & Titeldaten, um Spiel-Browser zu zeigen

FIG. 13J

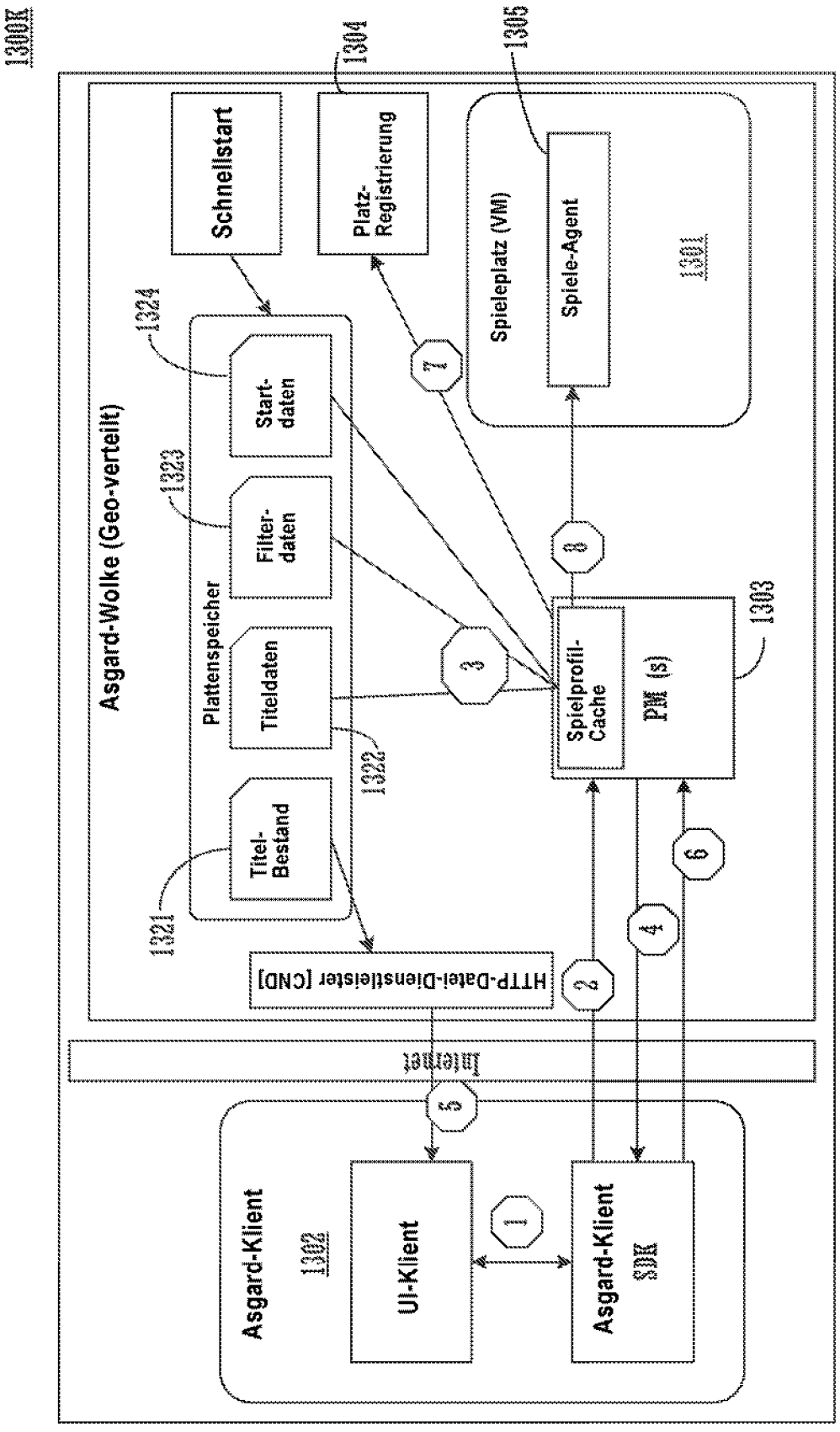

1300K
Asgard-Wolke (Geo-verteilt)
Schnellstart
1304
Platz-Registrierung
1305
Spieleplatz (VM)
Spiele-Agent
1301
1324
Start-daten
1323
Filter-daten
Plattenspeicher
Titeldaten
1322
Titel-Bestand
1321
HTTP-Datei-Dienstleister [CND]
Spielprofil-Cache
PM (s)
1303
Internet
Asgard-Klient
1302
UI-Klient
Asgard-Klient SDK
1
2
3
4
5
6
7
8

FIG. 13K

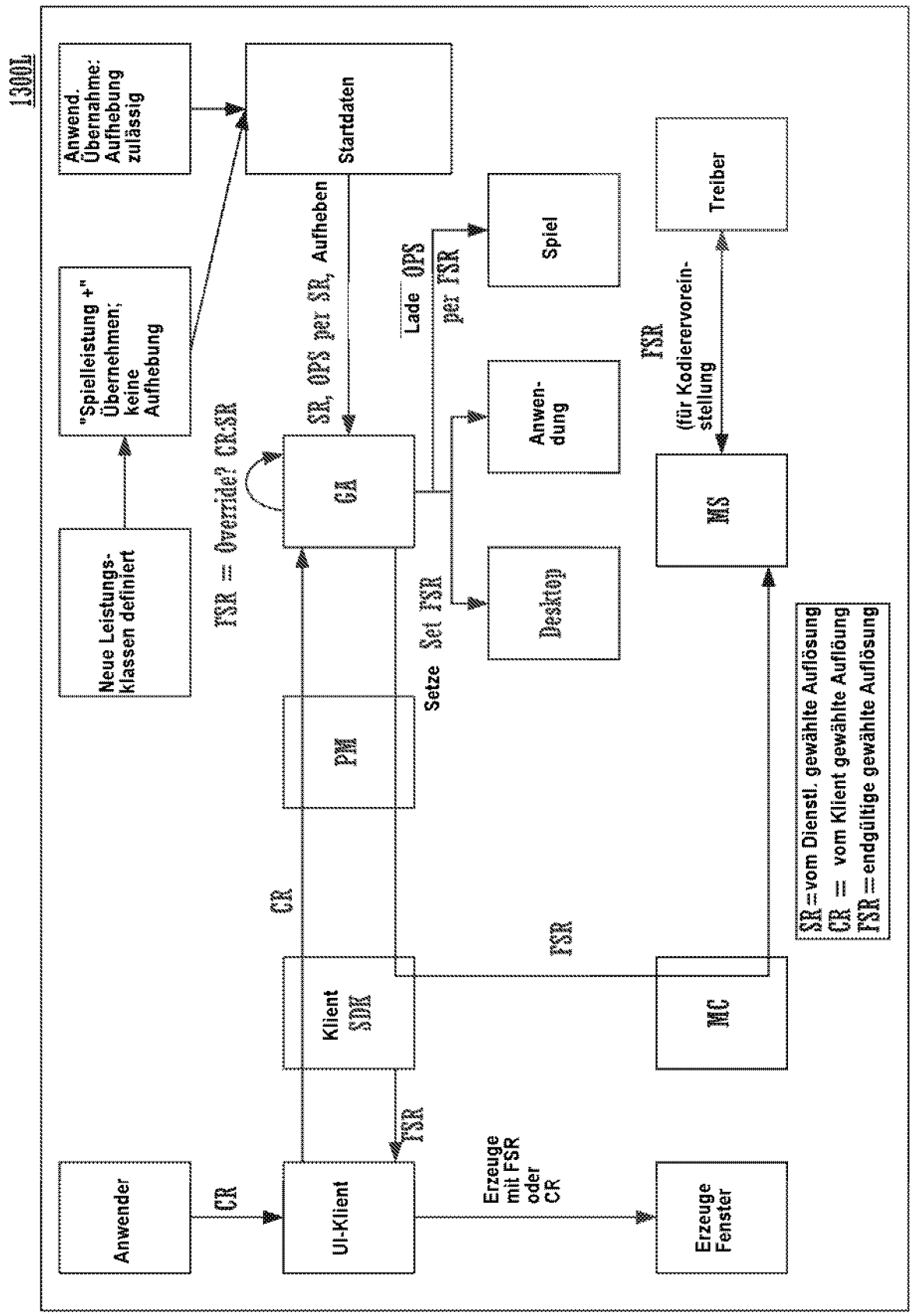

1300L
Anwend. Übernahme: Aufhebung zulässig
Startdaten
"Spielleistung +" Übernehmen; keine Aufhebung
Neue Leistungs-klassen definiert
TSR = Override? CR:SR
SR, OPS per SR, Aufheben
Lade OPS per FSR
Spiel
Anwen-dung
Desktop
Setze Set FSR
GA
PM
CR
FSR
Klient SDK
TSR
Anwender
CR
UI-Klient
Erzeuge mit FSR oder CR
Erzeuge Fenster
MC
FSR
(für Kodierervorein-stellung
MS
Treiber
SR = vom Dienstl. gewählte Auflösung
CR = vom Klient gewählte Auflösung
FSR = endgültige gewählte Auflösung

FIG. 13L

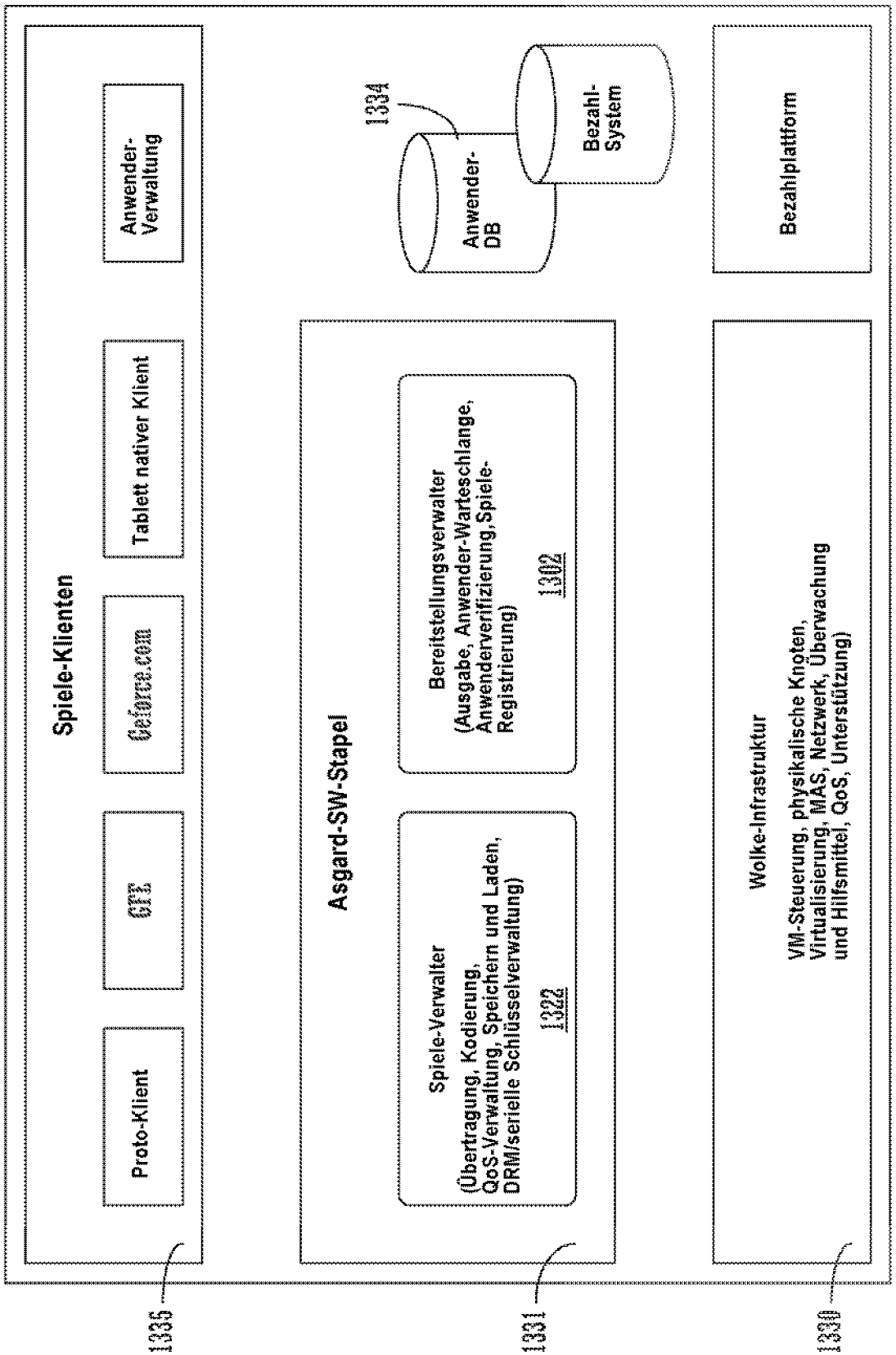

1300M
Spiele-Klienten
Proto-Klient
GTI
Cefore.com
Tablett nativer Klient
Anwender-Verwaltung
1335
Asgard-SW-Stapel
Spiele-Verwalter
(Übertragung, Kodierung, QoS-Verwaltung, Speichern und Laden, DRM/serielle Schlüsselverwaltung)
1322
Bereitstellungsverwalter
(Ausgabe, Anwender-Warteschlange, Anwenderverifizierung, Spiele-Registrierung)
1302
1331
1334
Anwender-DB
Bezahl-System
Wolke-Infrastruktur
VM-Steuerung, physikalische Knoten, Virtualisierung, MAS, Netzwerk, Überwachung und Hilfsmittel, QoS, Unterstützung)
1330
Bezahlplattform

FIG. 13M

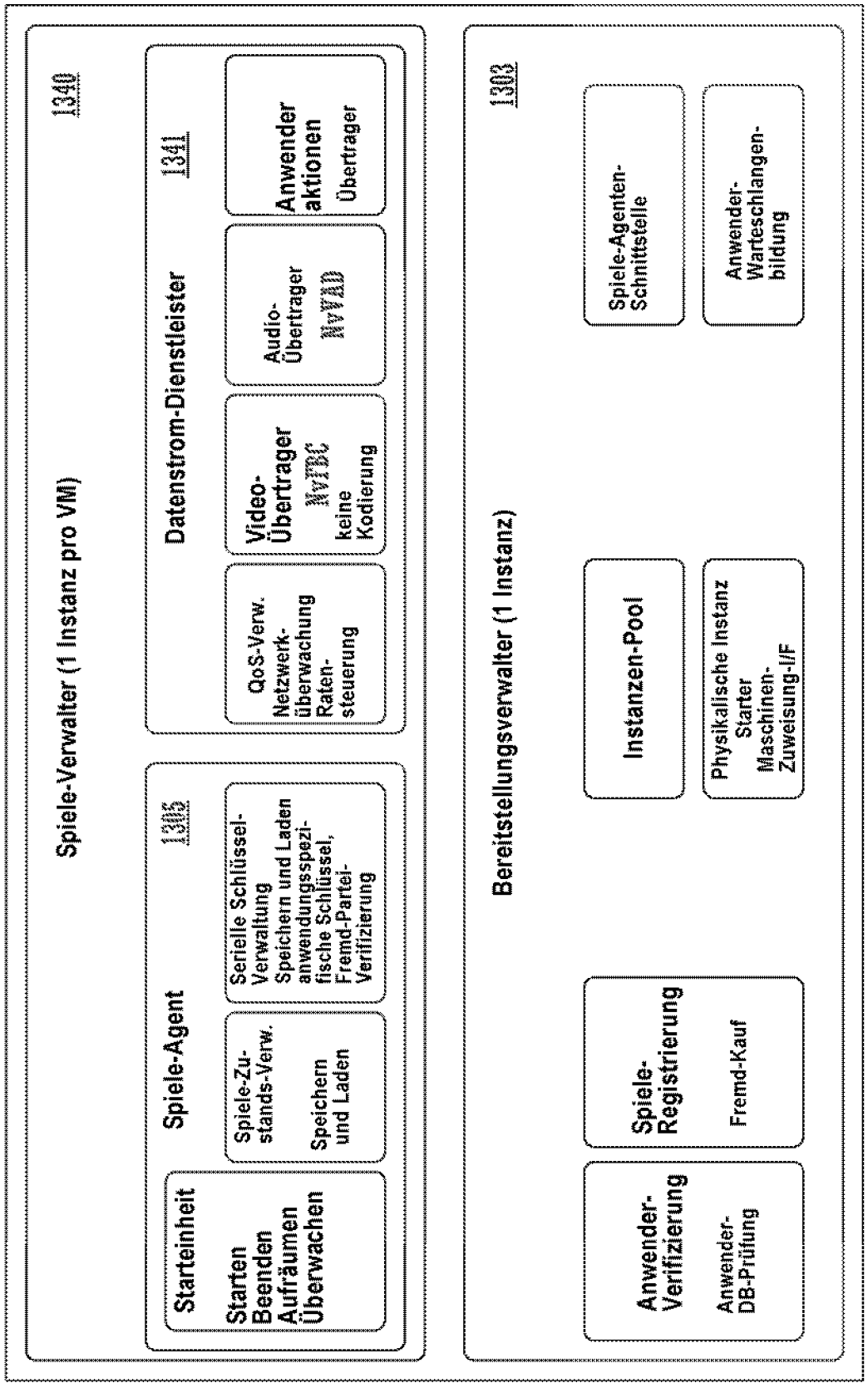

1300N
Spiele-Verwalter (1 Instanz pro VM)
1340
Starteinheit
Starten
Beenden
Aufräumen
Überwachen
Spiele-Agent
1305
Spiele-Zu-stands-Verw.
Speichern und Laden
Serielle Schlüssel-Verwaltung
Speichern und Laden anwendungsspezi-fische Schlüssel, Fremd-Partei-Verifizierung
Datenstrom-Dienstleister
1341
QoS-Verw. Netzwerk-überwachung Raten-steuerung
Video-Übertrager
NVBC
keine Kodierung
Audio-Übertrager
NVAD
Anwender-aktionen
Übertrager
Bereitstellungsverwalter (1 Instanz)
1303
Anwender-Verifizierung
Anwender-DB-Prüfung
Spiele-Registrierung
Fremd-Kauf
Instanzen-Pool
Physikalische Instanz
Starter
Maschinen-Zuweisung-I/F
Spiele-Agenten-Schnittstelle
Anwender-Warteschlangen-bildung

FIG. 13N

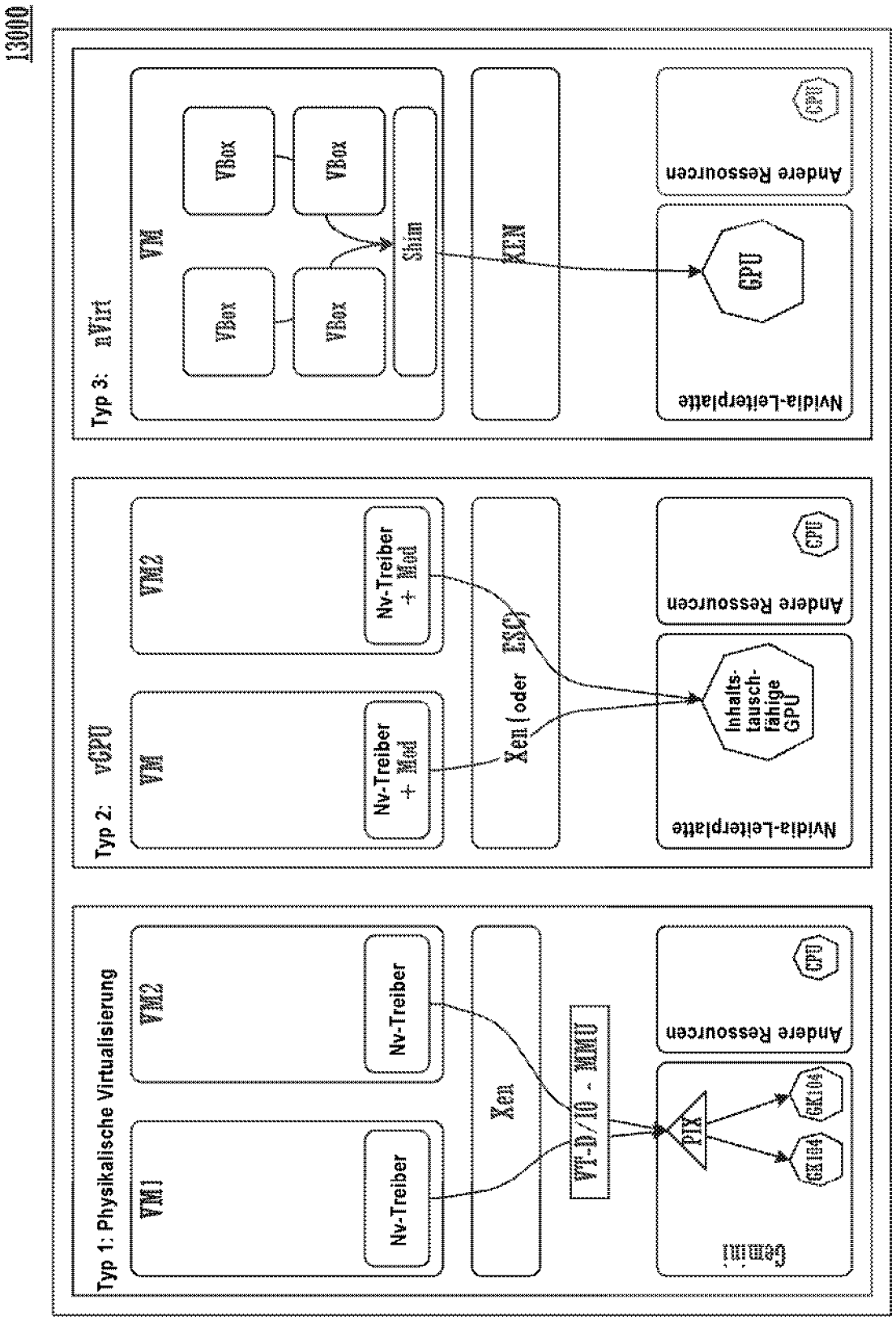

13000
Typ 1: Physikalische Virtualisierung
VM1
VM2
Nv-Treiber
Nv-Treiber
Xen
VT-D / IO - MMU
PCIe
GK104
GK104
Andere Ressourcen
CPU
Typ 2: vGPU
VM
VM2
Nv-Treiber + Mod
Nv-Treiber + Mod
Xen (oder ESXi)
Nvidia-Leiterplatte
Inhalts-tausch-fähige GPU
Andere Ressourcen
CPU
Typ 3: nVirt
VM
VBox
VBox
VBox
VBox
Shim
KTM
Nvidia-Leiterplatte
GPU
Andere Ressourcen
CPU

FIG. 130

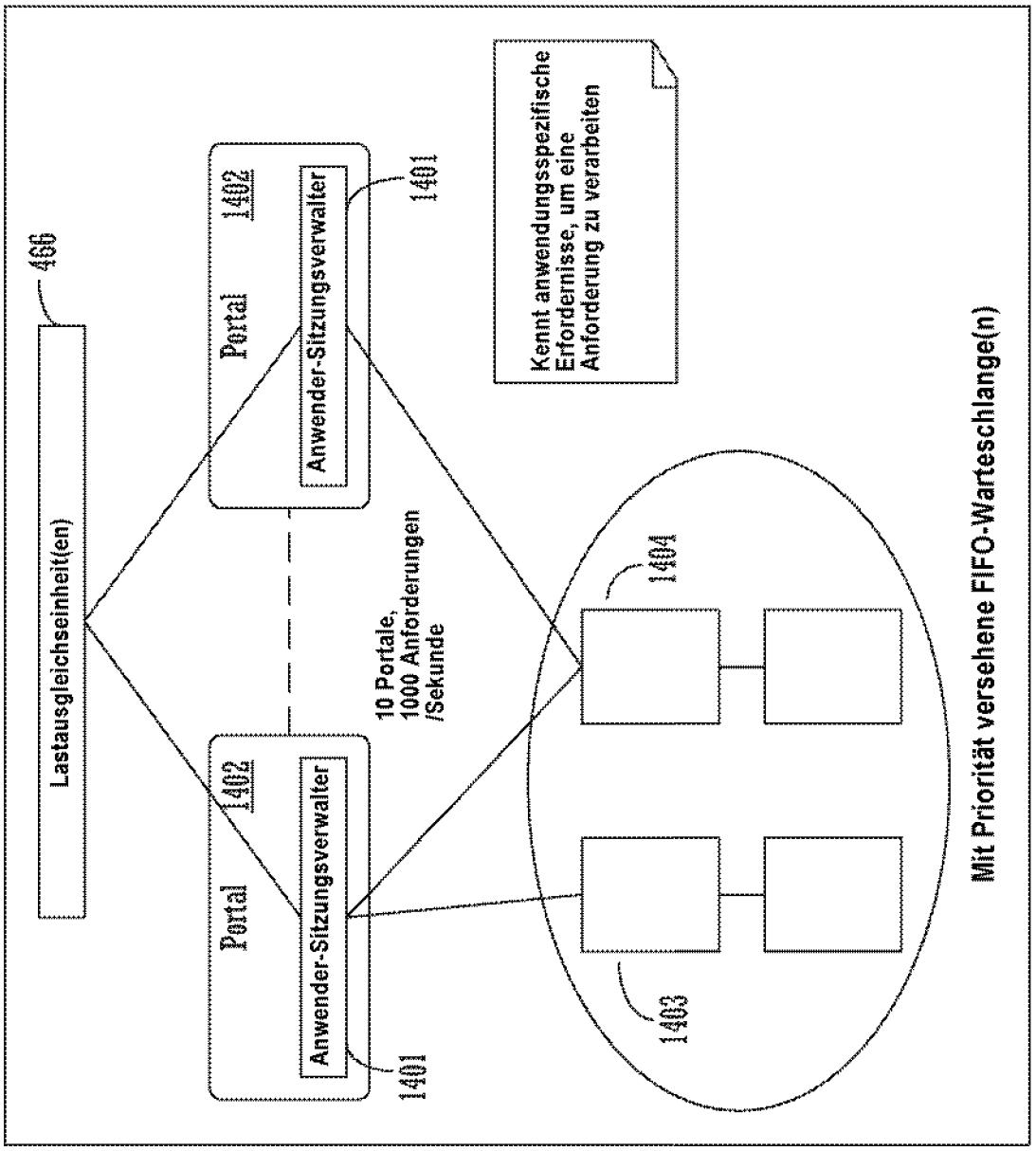

1400A
466
Lastausgleichseinheit(en)
Portal
1402
Anwender-Sitzungsverwalter
1401
10 Portale,
1000 Anforderungen
/Sekunde
Kennt anwendungsspezifische
Erfordernisse, um eine
Anforderung zu verarbeiten
1404
1403
Mit Priorität versehene FIFO-Warteschlange(n)

FIG. 14A

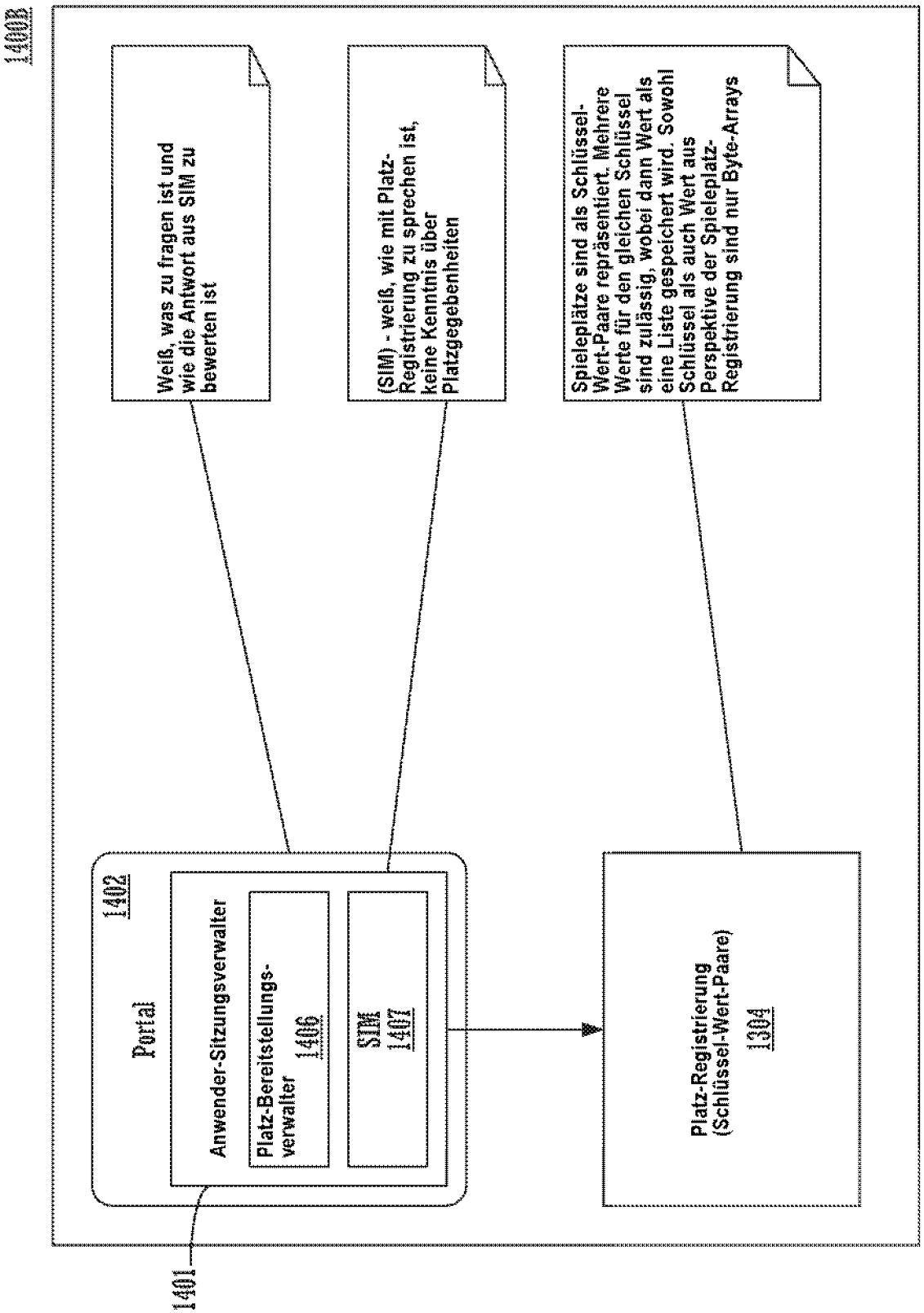

1400B
Weiß, was zu fragen ist und wie die Antwort aus SIM zu bewerten ist
(SIM) - weiß, wie mit Platz-Registrierung zu sprechen ist, keine Kenntnis über Platzgegebenheiten
Spieleplätze sind als Schlüssel-Wert-Paare repräsentiert. Mehrere Werte für den gleichen Schlüssel sind zulässig, wobei dann Wert als eine Liste gespeichert wird. Sowohl Schlüssel als auch Wert aus Perspektive der Spieleplatz-Registrierung sind nur Byte-Arrays
1402
Portal
Anwender-Sitzungsverwalter
Platz-Bereitstellungs-verwalter 1406
SIM 1407
Platz-Registrierung (Schlüssel-Wert-Paare) 1304
1401

FIG. 14B

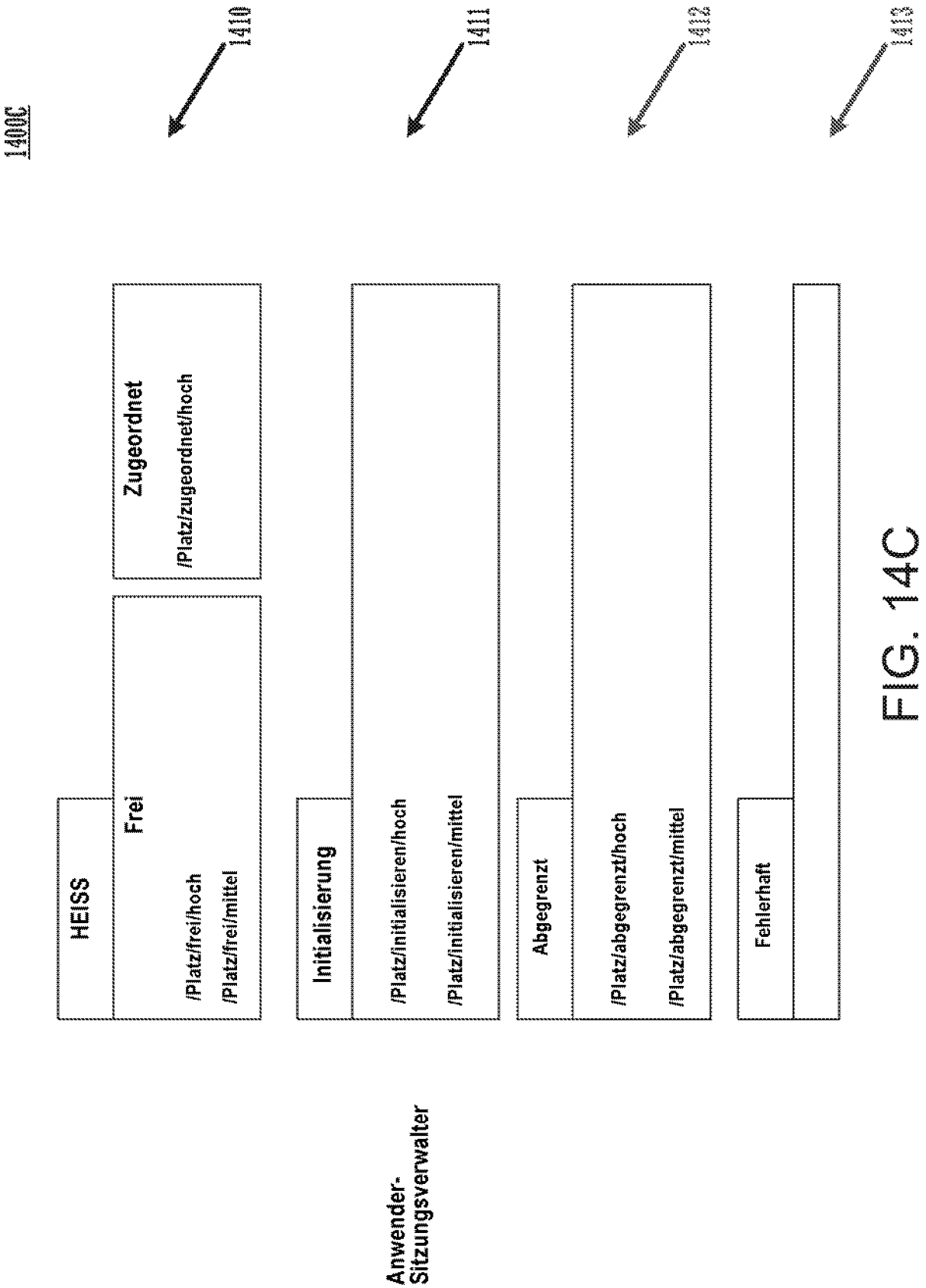

1400C
1410
1411
1412
1413
HEISS
Frei
/Platz/frei/hoch
/Platz/frei/mittel
Zugeordnet
/Platz/zugeordnet/hoch
Initialisierung
/Platz/initialisieren/hoch
/Platz/initialisieren/mittel
Abgegrenzt
/Platz/abgegrenzt/hoch
/Platz/abgegrenzt/mittel
Fehlerhaft
Anwender-Sitzungsverwalter

FIG. 14C

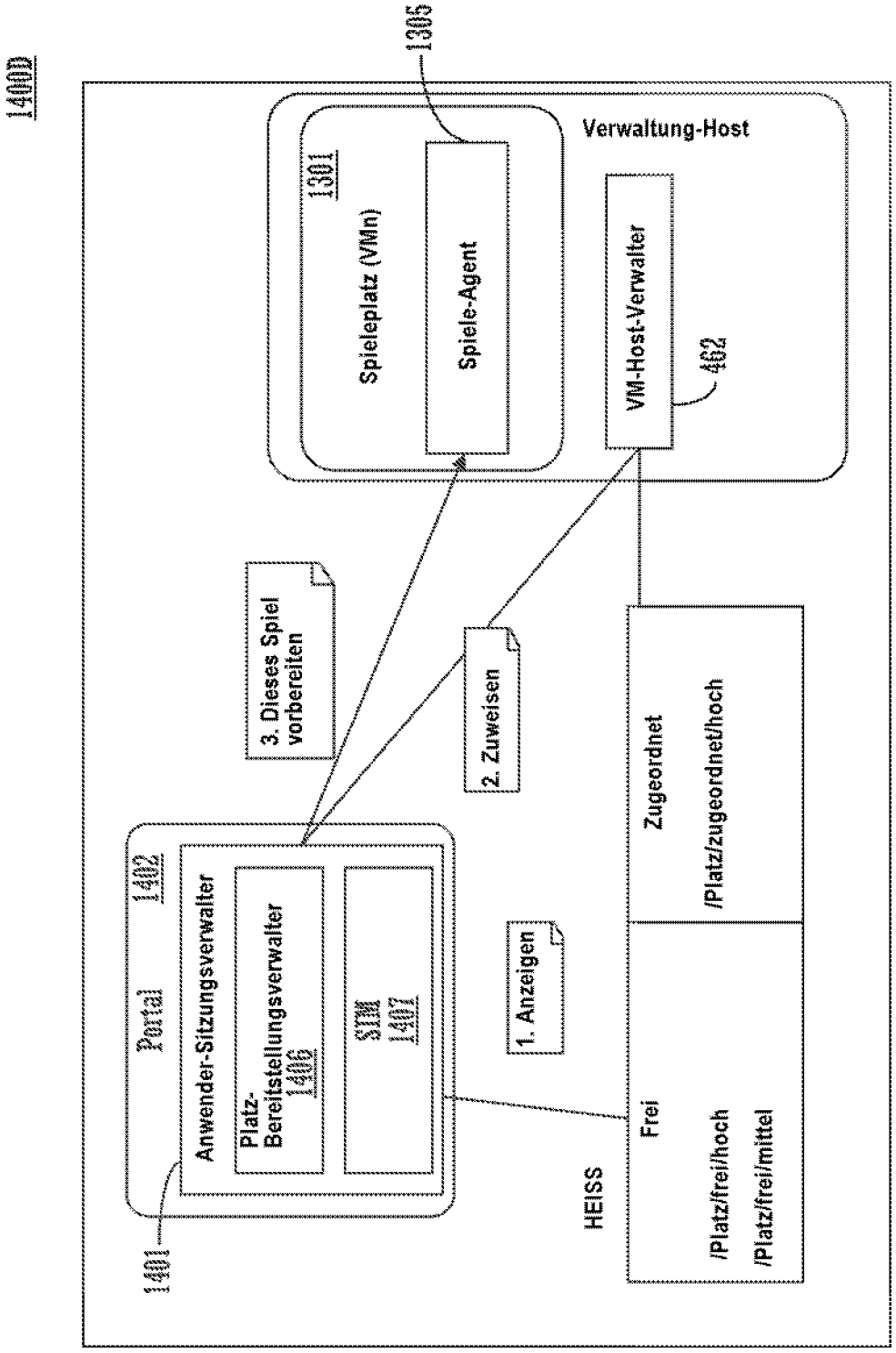

1400D
1305
Verwaltung-Host
1301
Spieleplatz (VMn)
Spiele-Agent
VM-Host-Verwalter
462
3. Dieses Spiel vorbereiten
2. Zuweisen
Zugeordnet
/Platz/zugeordnet/hoch
Portal
1402
Anwender-Sitzungsverwalter
Platz-Bereitstellungsverwalter
1406
SIM
1407
1. Anzeigen
HEISS
Frei
/Platz/frei/hoch
/Platz/frei/mittel
1401

FIG. 14D

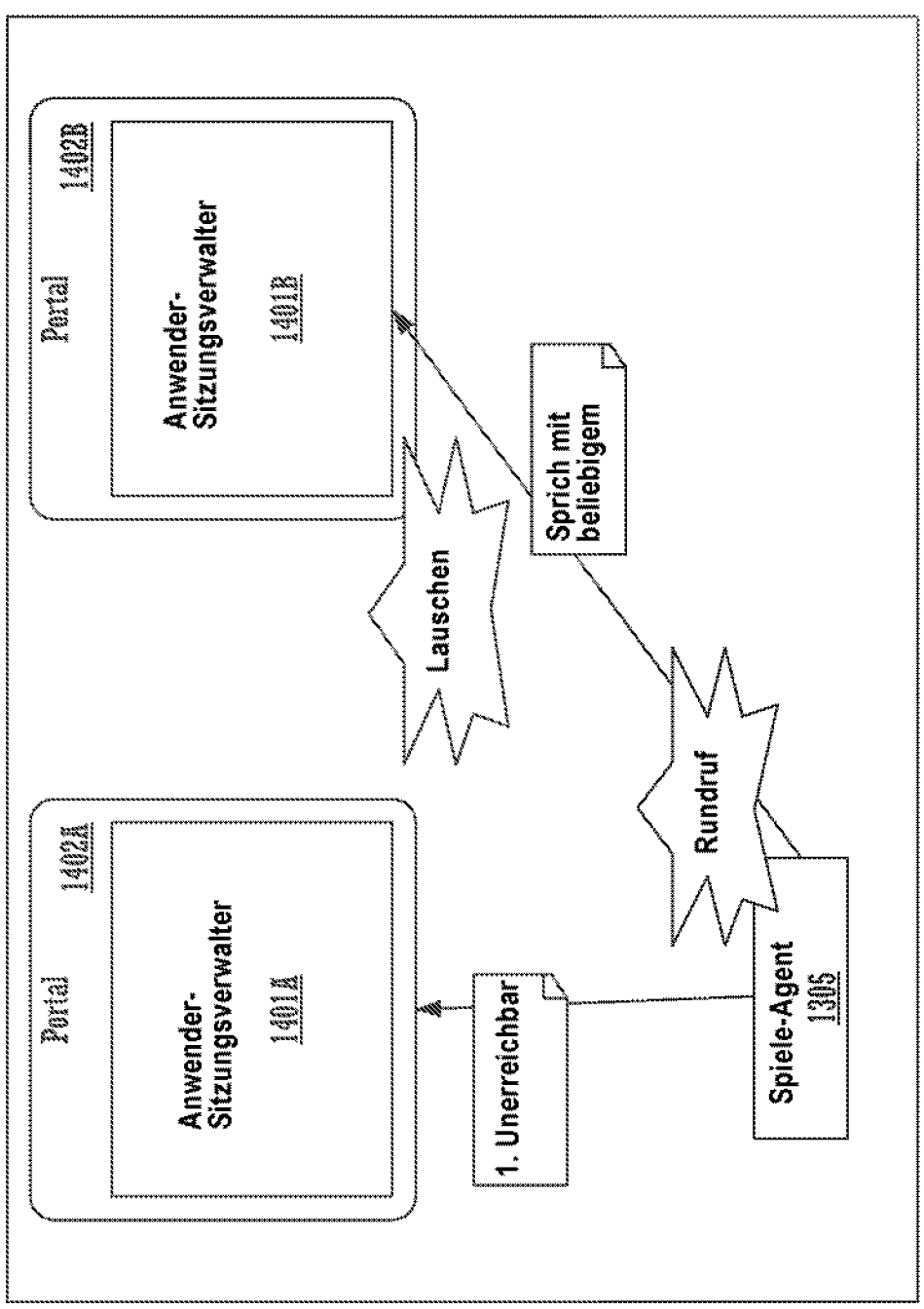

1400F
Portal
1402B
Anwender-Sitzungsverwalter
1401B
Portal
1402A
Anwender-Sitzungsverwalter
1401A
Lauschen
Sprich mit beliebigem
Rundruf
1. Unerreichbar
Spiele-Agent
1305

FIG. 14E

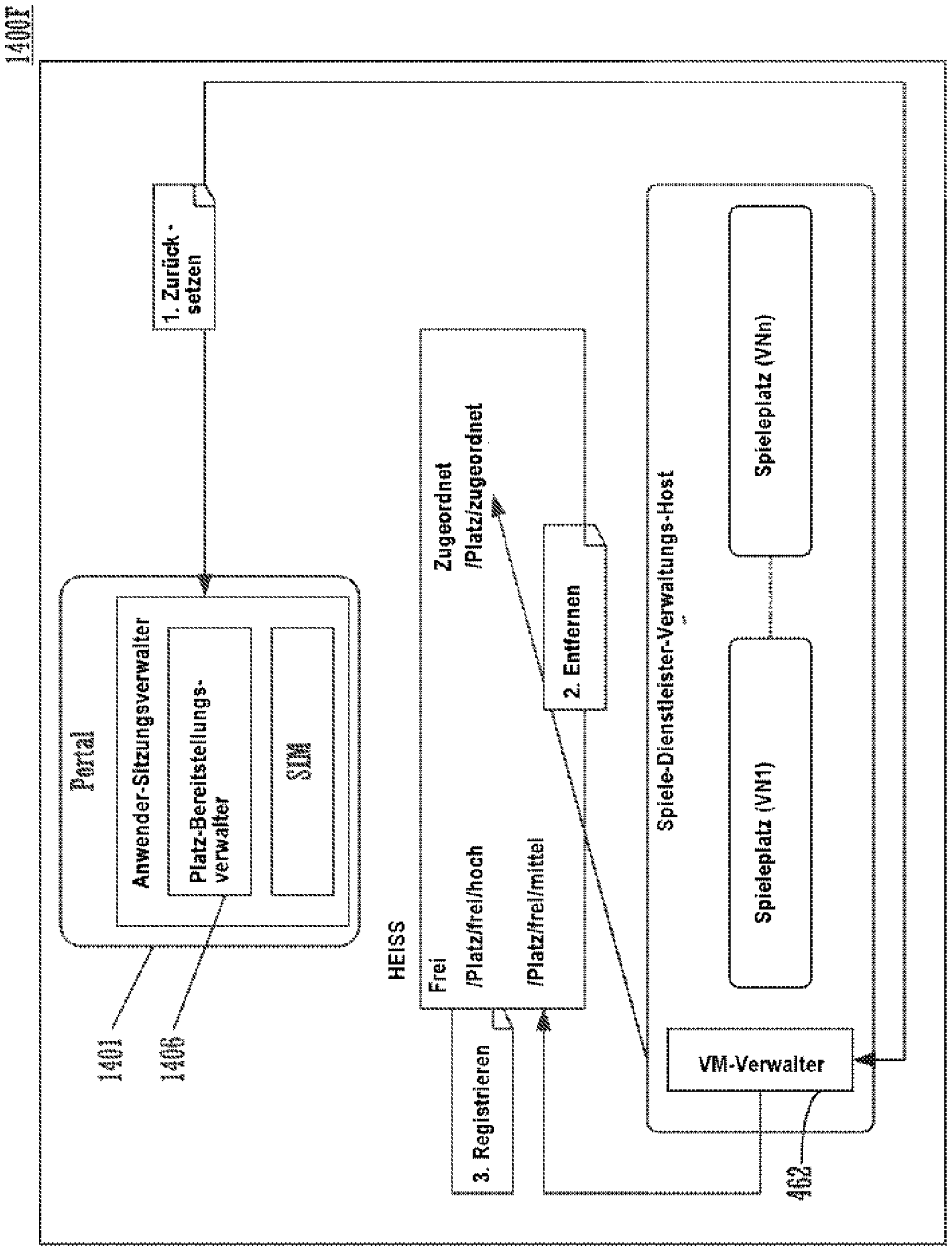

1400F
1. Zurück-setzen
Portal
Anwender-Sitzungsverwalter
Platz-Bereitstellungs-verwalter
SIM
1401
1406
HEISS
Frei
/Platz/frei/hoch
/Platz/frei/mittel
3. Registrieren
Zugeordnet
/Platz/zugeordnet
2. Entfernen
Spiele-Dienstleister-Verwaltungs-Host
Spieleplatz (VNn)
Spieleplatz (VN1)
VM-Verwalter
462

FIG. 14F

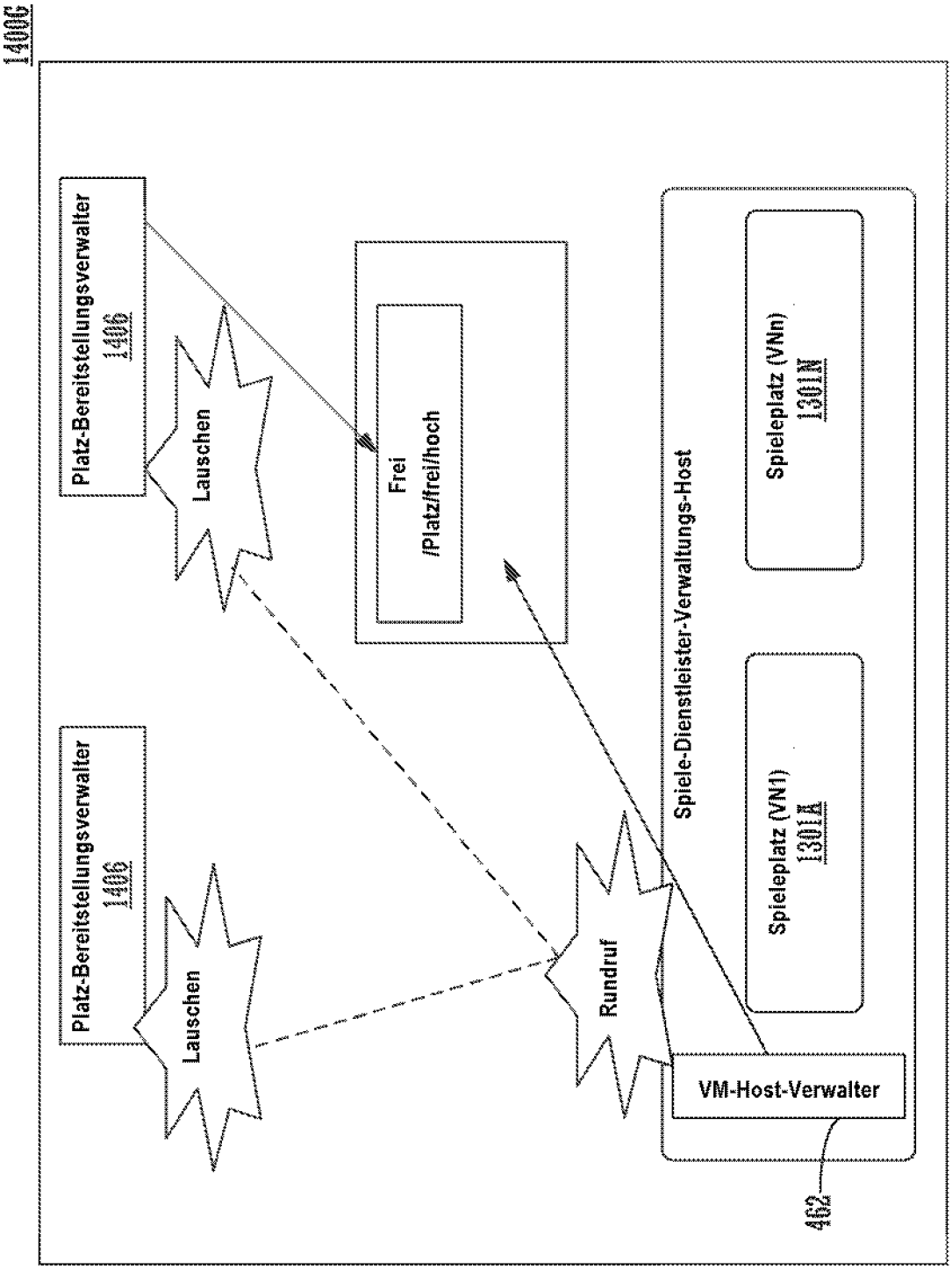

1400G
Platz-Bereitstellungsverwalter
1406
Lauschen
Frei
/Platz/frei/hoch
Spieleplatz (VNn)
1301N
Spiele-Dienstleister-Verwaltungs-Host
Platz-Bereitstellungsverwalter
1406
Lauschen
Rundruf
Spieleplatz (VN1)
1301A
VM-Host-Verwalter
462

FIG. 14G

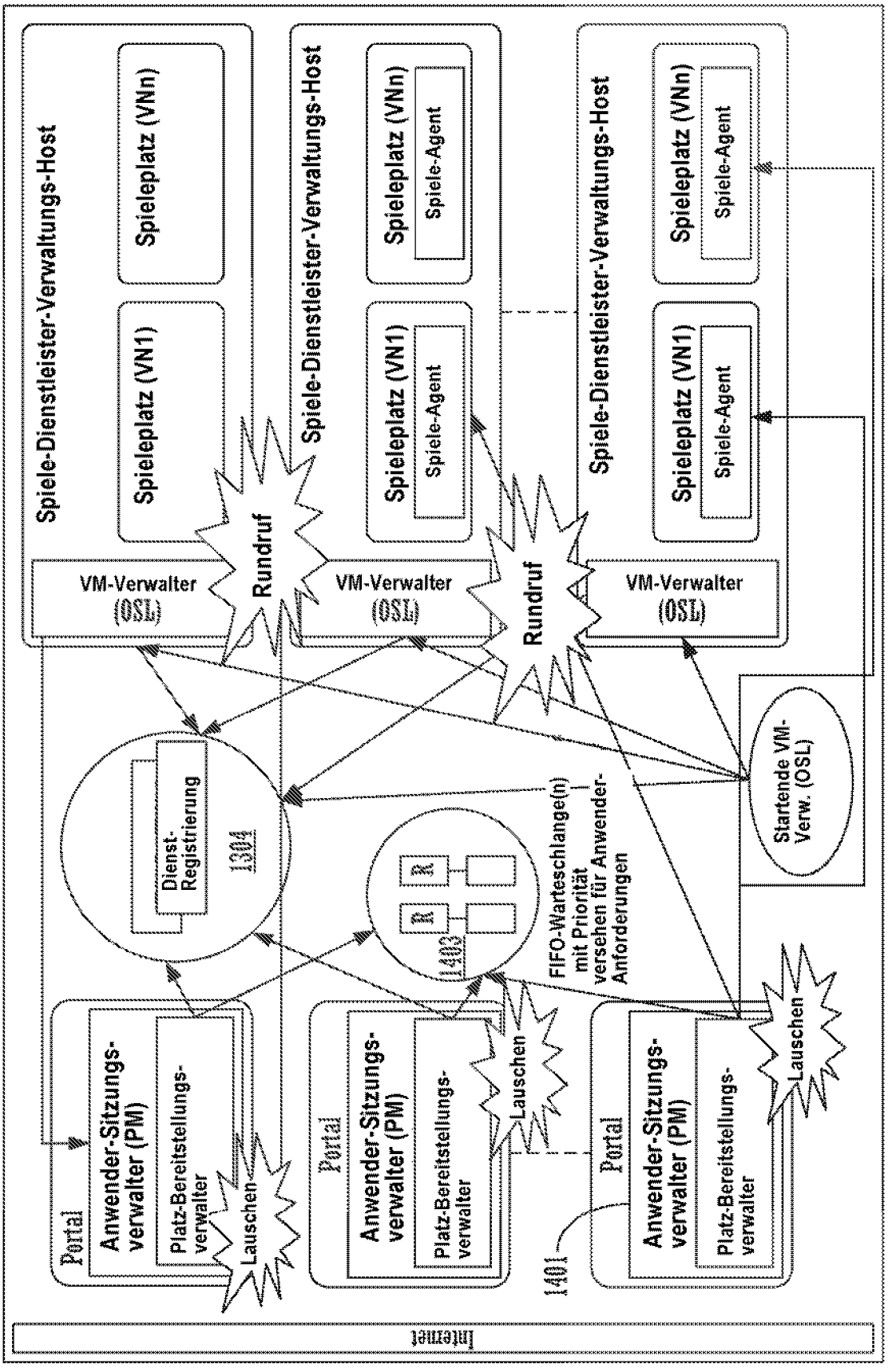

1400H
Spiele-Dienstleister-Verwaltungs-Host
Spieleplatz (VNn)
Spieleplatz (VN1)
Spiele-Agent
VM-Verwalter (OSL)
Rundruf
Spiele-Dienstleister-Verwaltungs-Host
Spieleplatz (VNn)
Spieleplatz (VN1)
Spiele-Agent
VM-Verwalter (OSL)
Rundruf
Spiele-Dienstleister-Verwaltungs-Host
Spieleplatz (VNn)
Spieleplatz (VN1)
Spiele-Agent
VM-Verwalter (OSL)
Dienst-Registrierung
1304
1403
FIFO-Warteschlange(n) mit Priorität versehen für Anwender-Anforderungen
Startende VM-Verw. (OSL)
Portal
Anwender-Sitzungs-verwalter (PM)
Platz-Bereitstellungs-verwalter
Lauschen
Portal
Anwender-Sitzungs-verwalter (PM)
Platz-Bereitstellungs-verwalter
Lauschen
Portal
Anwender-Sitzungs-verwalter (PM)
Platz-Bereitstellungs-verwalter
Lauschen
1401
Internet

FIG. 14H

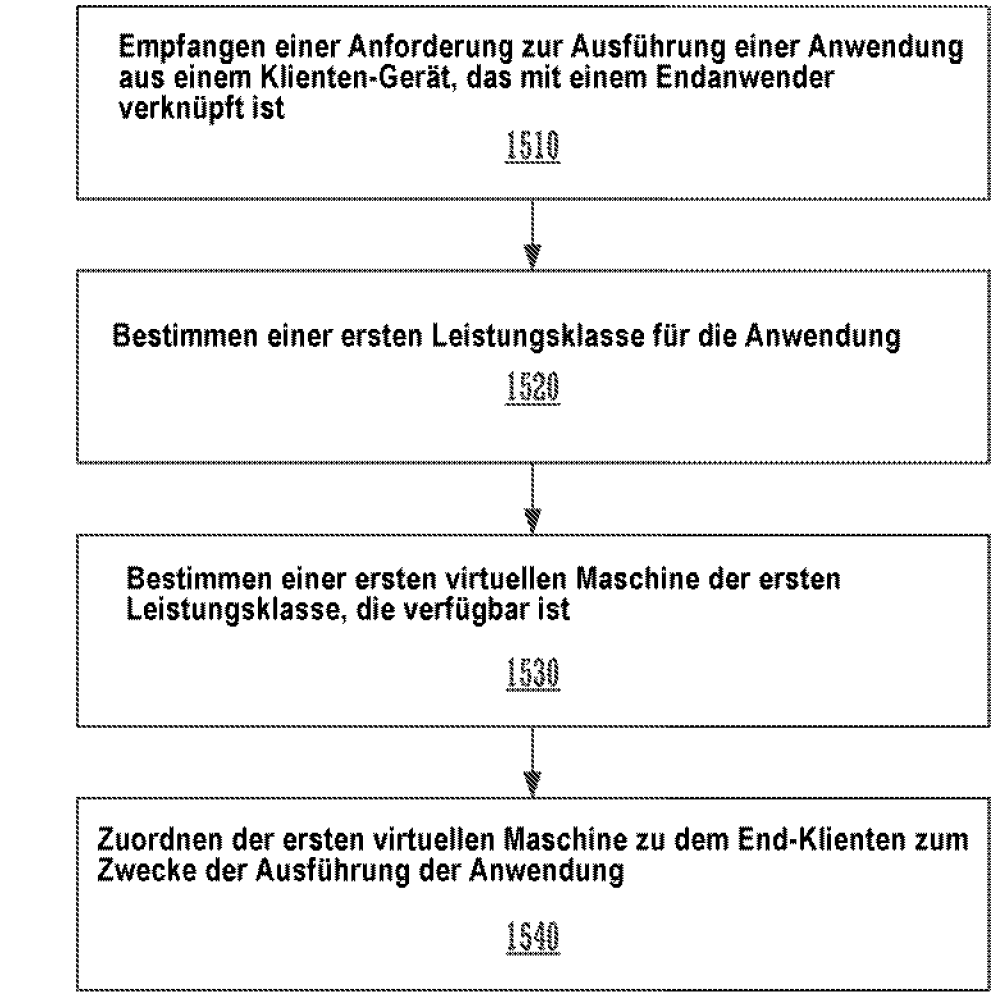

1500
Empfangen einer Anforderung zur Ausführung einer Anwendung aus einem Klienten-Gerät, das mit einem Endanwender verknüpft ist
1510
Bestimmen einer ersten Leistungsklasse für die Anwendung
1520
Bestimmen einer ersten virtuellen Maschine der ersten Leistungsklasse, die verfügbar ist
1530
Zuordnen der ersten virtuellen Maschine zu dem End-Klienten zum Zwecke der Ausführung der Anwendung
1540

FIG. 15

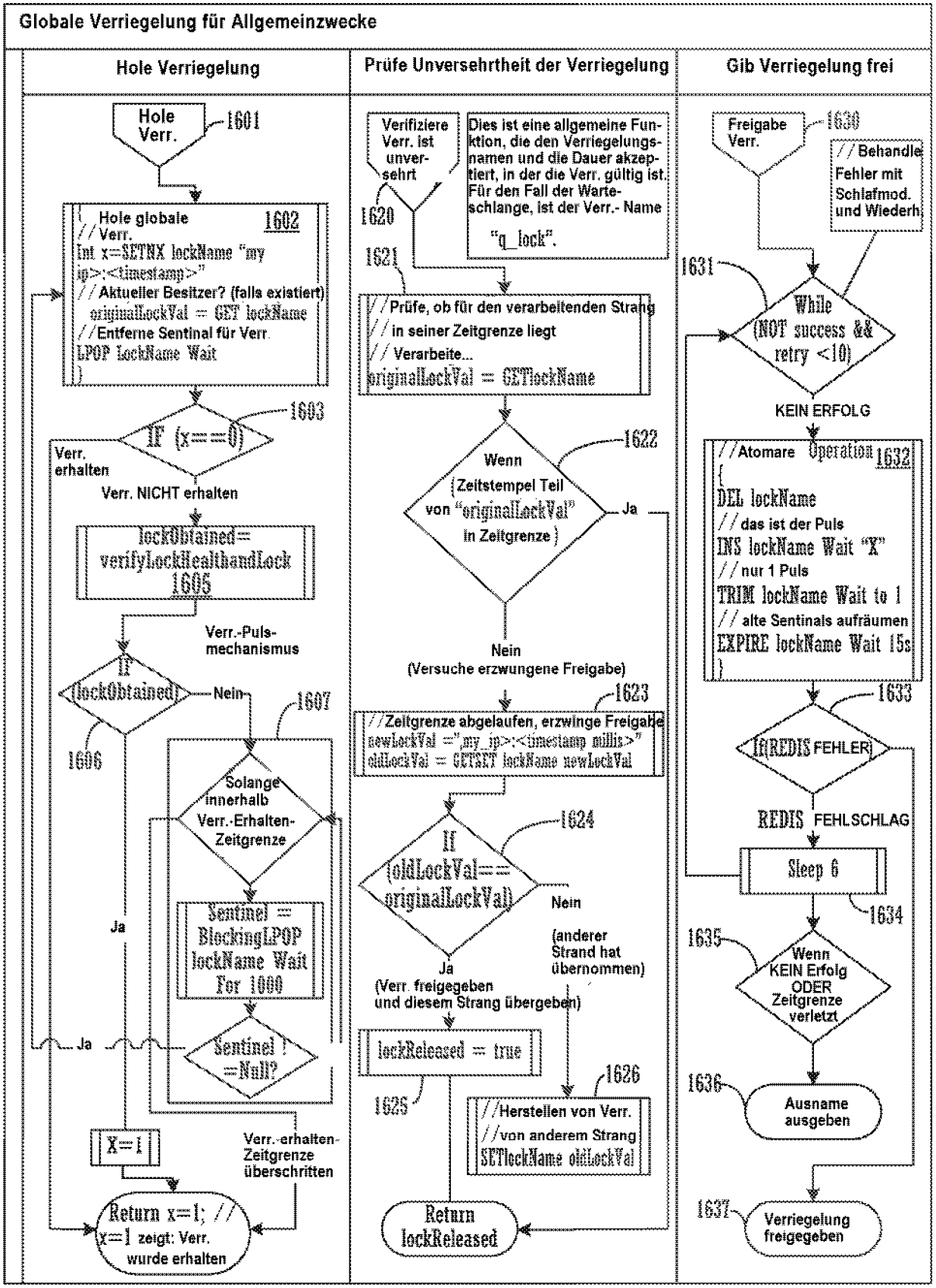

Globale Verriegelung für Allgemeinzwecke
Hole Verriegelung
Prüfe Unversehrtheit der Verriegelung
Gib Verriegelung frei
Hole Verr.
1601
Hole globale Verr.
1602
int x=SETNX lockName "my ip>:<timestamp>"
//Aktueller Besitzer? (falls existiert)
originalLockVal = GET lockName
//Entferne Sentinal für Verr.
LPOP lockName Wait
}
IF (x==0)
1603
Verr. erhalten
Verr. NICHT erhalten
lockObtained= verifyLockHealthandLock 1605
Verr.-Puls-mechanismus
IF (lockObtained)
Nein
1606
1607
Solange innerhalb Verr.-Erhalten-Zeitgrenze
Ja
Sentinel = BlockingLPOP lockName Wait For 1000
Sentinel != Null?
Ja
X=1
Verr.-erhalten-Zeitgrenze überschritten
Return x=1; // x=1 zeigt: Verr. wurde erhalten
Verifiziere Verr. ist unversehrt
1620
Dies ist eine allgemeine Funktion, die den Verriegelungsnamen und die Dauer akzeptiert, in der die Verr. gültig ist. Für den Fall der Warteschlange, ist der Verr.- Name "q_lock".
1621
//Prüfe, ob für den verarbeitenden Strang
//in seiner Zeitgrenze liegt
//Verarbeite...
originalLockVal = GETlockName
Wenn (Zeitstempel Teil von "originalLockVal" in Zeitgrenze)
1622
Ja
Nein
(Versuche erzwungene Freigabe)
1623
//Zeitgrenze abgelaufen, erzwinge Freigabe
newLockVal ="my_ip>:<timestamp millis>"
oldLockVal = GETSET lockName newLockVal
IF (oldLockVal== originalLockVal)
1624
Nein
(anderer Strand hat übernommen)
Ja
(Verr. freigegeben und diesem Strang übergeben)
lockReleased = true
1625
1626
//Herstellen von Verr.
//von anderem Strang
SETlockName oldLockVal
Return lockReleased
Freigabe Verr.
1630
//Behandle Fehler mit Schlafmod. und Wiederh
1631
While (NOT success && retry <10)
KEIN ERFOLG
//Atomare Operation
1632
{
DEL lockName
//das ist der Puls
INS lockName Wait "X"
//nur 1 Puls
TRIM lockName Wait to 1
//alte Sentinals aufräumen
EXPIRE lockName Wait 15s
}
1633
IF(REDIS FEHLER)
REDIS FEHLSCHLAG
Sleep 6
1634
1635
Wenn KEIN Erfolg ODER Zeitgrenze verletzt
1636
Ausname ausgeben
1637
Verriegelung freigegeben

FIG. 16

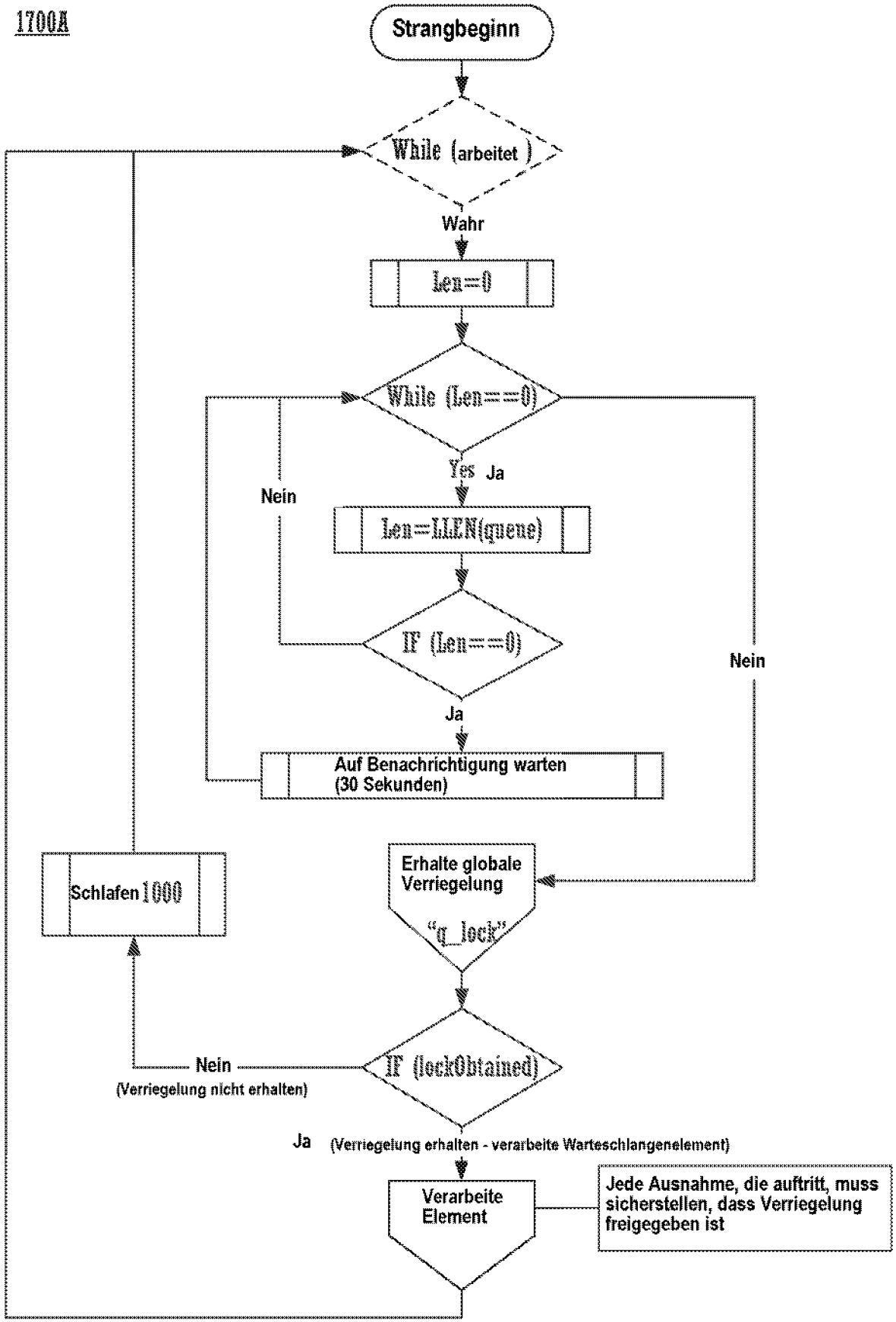

1700A
Strangbeginn
While (arbeitet)
Wahr
Len=0
While (Len==0)
Yes Ja
Nein
Len=LLEN(quene)
IF (Len==0)
Nein
Ja
Auf Benachrichtigung warten
(30 Sekunden)
Schlafen 1000
Erhalte globale
Verriegelung
"q_lock"
Nein
(Verriegelung nicht erhalten)
IF (lockObtained)
Ja
(Verriegelung erhalten - verarbeite Warteschlangenelement)
Verarbeite
Element
Jede Ausnahme, die auftritt, muss
sicherstellen, dass Verriegelung
freigegeben ist

FIG. 17A

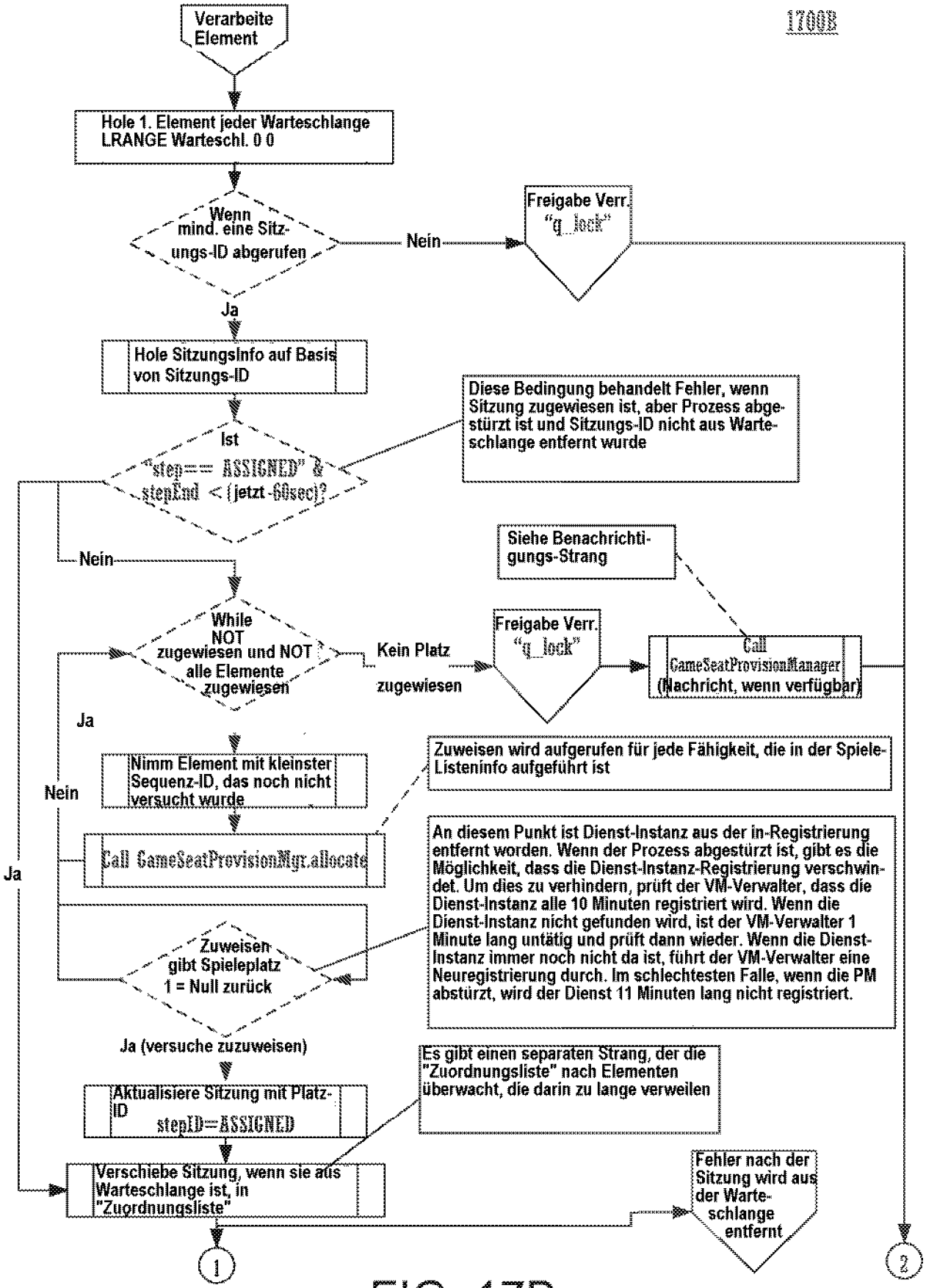

1700B
Verarbeite Element
Hole 1. Element jeder Warteschlange LRANGE Warteschl. 0 0
Wenn mind. eine Sitzungs-ID abgerufen
Nein
Freigabe Verr. "q_lock"
Ja
Hole SitzungsInfo auf Basis von Sitzungs-ID
Diese Bedingung behandelt Fehler, wenn Sitzung zugewiesen ist, aber Prozess abgestürzt ist und Sitzungs-ID nicht aus Warteschlange entfernt wurde
Ist "step== ASSIGNED" & stepEnd < (jetzt -60sec)?
Nein
Siehe Benachrichtigungs-Strang
While NOT zugewiesen und NOT alle Elemente zugewiesen
Kein Platz zugewiesen
Freigabe Verr. "q_lock"
Call GameSeatProvisionManager (Nachricht, wenn verfügbar)
Ja
Nimm Element mit kleinster Sequenz-ID, das noch nicht versucht wurde
Zuweisen wird aufgerufen für jede Fähigkeit, die in der Spiele-Listeninfo aufgeführt ist
Nein
Call GameSeatProvisionMgr.allocate
An diesem Punkt ist Dienst-Instanz aus der in-Registrierung entfernt worden. Wenn der Prozess abgestürzt ist, gibt es die Möglichkeit, dass die Dienst-Instanz-Registrierung verschwindet. Um dies zu verhindern, prüft der VM-Verwalter, dass die Dienst-Instanz alle 10 Minuten registriert wird. Wenn die Dienst-Instanz nicht gefunden wird, ist der VM-Verwalter 1 Minute lang untätig und prüft dann wieder. Wenn die Dienst-Instanz immer noch nicht da ist, führt der VM-Verwalter eine Neuregistrierung durch. Im schlechtesten Falle, wenn die PM abstürzt, wird der Dienst 11 Minuten lang nicht registriert.
Zuweisen gibt Spieleplatz 1 = Null zurück
Ja
Ja (versuche zuzuweisen)
Es gibt einen separaten Strang, der die "Zuordnungsliste" nach Elementen überwacht, die darin zu lange verweilen
Aktualisiere Sitzung mit Platz-ID stepID=ASSIGNED
Verschiebe Sitzung, wenn sie aus Warteschlange ist, in "Zuordnungsliste"
Fehler nach der Sitzung wird aus der Warteschlange entfernt
1
2

FIG. 17B

1700C

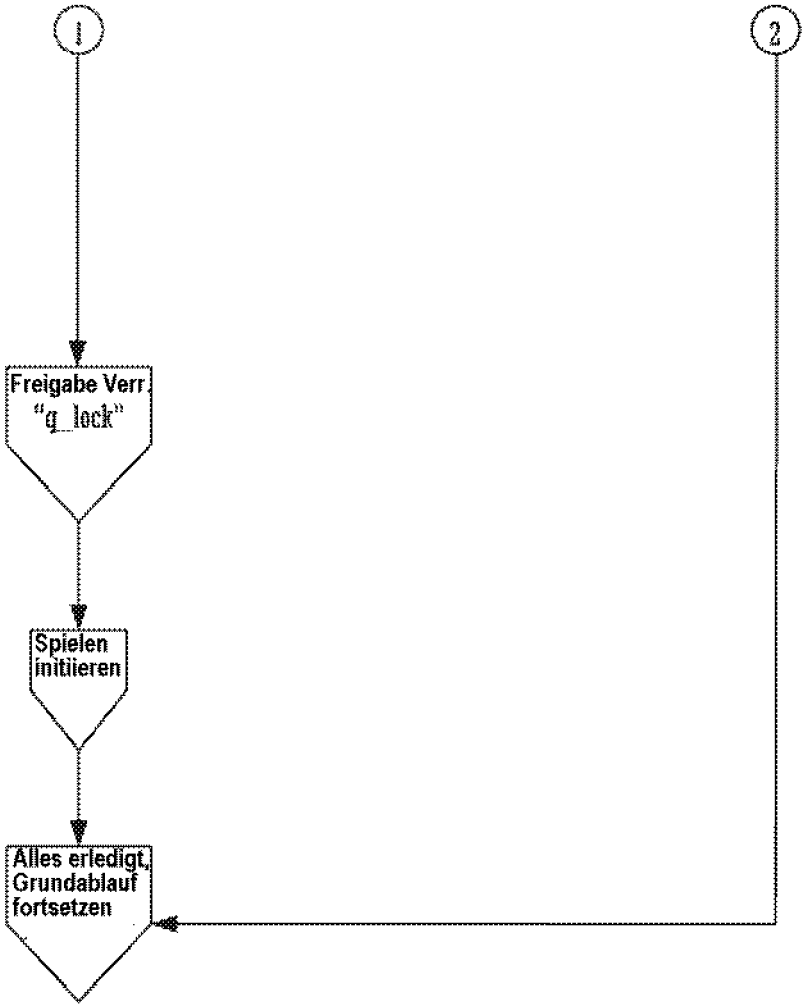


FIG. 17C

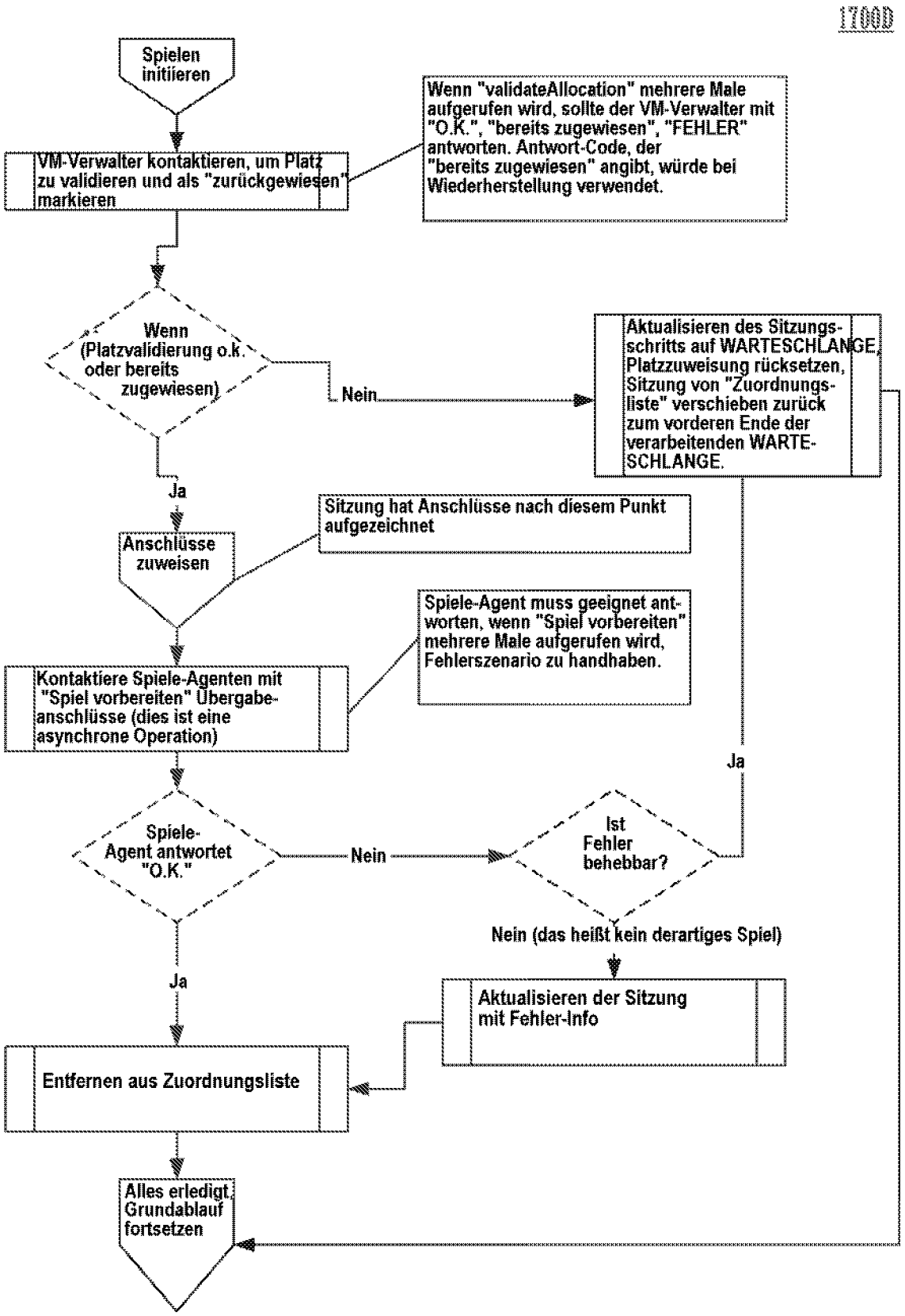

1700D
Spielen initiieren
VM-Verwalter kontaktieren, um Platz zu validieren und als "zurückgewiesen" markieren
Wenn "validateAllocation" mehrere Male aufgerufen wird, sollte der VM-Verwalter mit "O.K.", "bereits zugewiesen", "FEHLER" antworten. Antwort-Code, der "bereits zugewiesen" angibt, würde bei Wiederherstellung verwendet.
Wenn (Platzvalidierung o.k. oder bereits zugewiesen)
Nein
Aktualisieren des Sitzungsschritts auf WARTESCHLANGE, Platzzuweisung rücksetzen, Sitzung von "Zuordnungsliste" verschieben zurück zum vorderen Ende der verarbeitenden WARTESCHLANGE.
Ja
Anschlüsse zuweisen
Sitzung hat Anschlüsse nach diesem Punkt aufgezeichnet
Kontaktiere Spiele-Agenten mit "Spiel vorbereiten" Übergabeanschlüsse (dies ist eine asynchrone Operation)
Spiele-Agent muss geeignet antworten, wenn "Spiel vorbereiten" mehrere Male aufgerufen wird, Fehlerszenario zu handhaben.
Spiele-Agent antwortet "O.K."
Nein
Ist Fehler behebbar?
Ja
Nein (das heißt kein derartiges Spiel)
Ja
Aktualisieren der Sitzung mit Fehler-Info
Entfernen aus Zuordnungsliste
Alles erledigt, Grundablauf fortsetzen

FIG. 17D

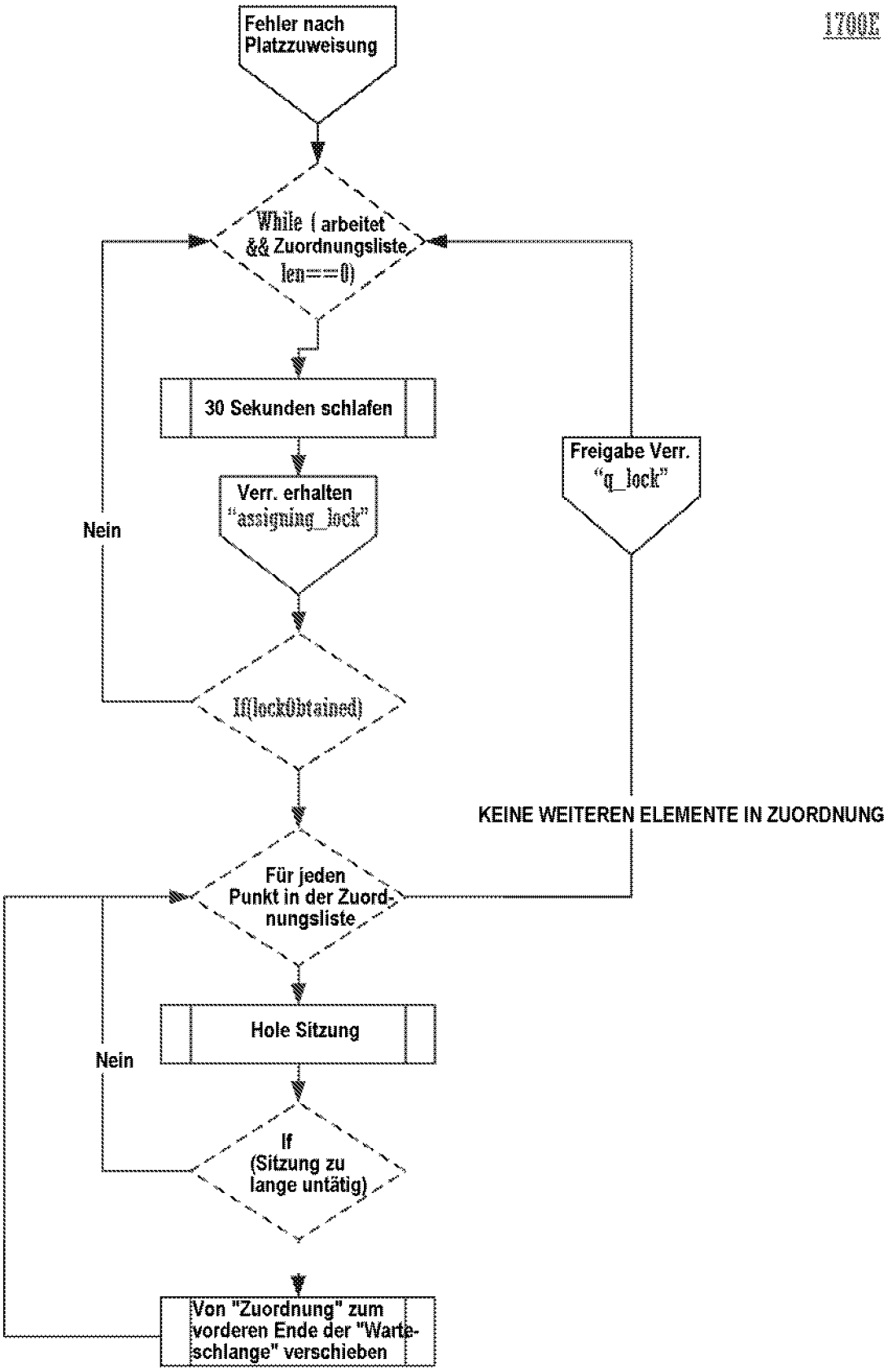

1700E
Fehler nach Platzzuweisung
While (arbeitet && Zuordnungsliste.len==0)
30 Sekunden schlafen
Verr. erhalten "assigning_lock"
If(lockObtained)
Nein
Freigabe Verr. "q_lock"
KEINE WEITEREN ELEMENTE IN ZUORDNUNG
Für jeden Punkt in der Zuordnungsliste
Hole Sitzung
Nein
If (Sitzung zu lange untätig)
Von "Zuordnung" zum vorderen Ende der "Warteschlange" verschieben

FIG. 17E

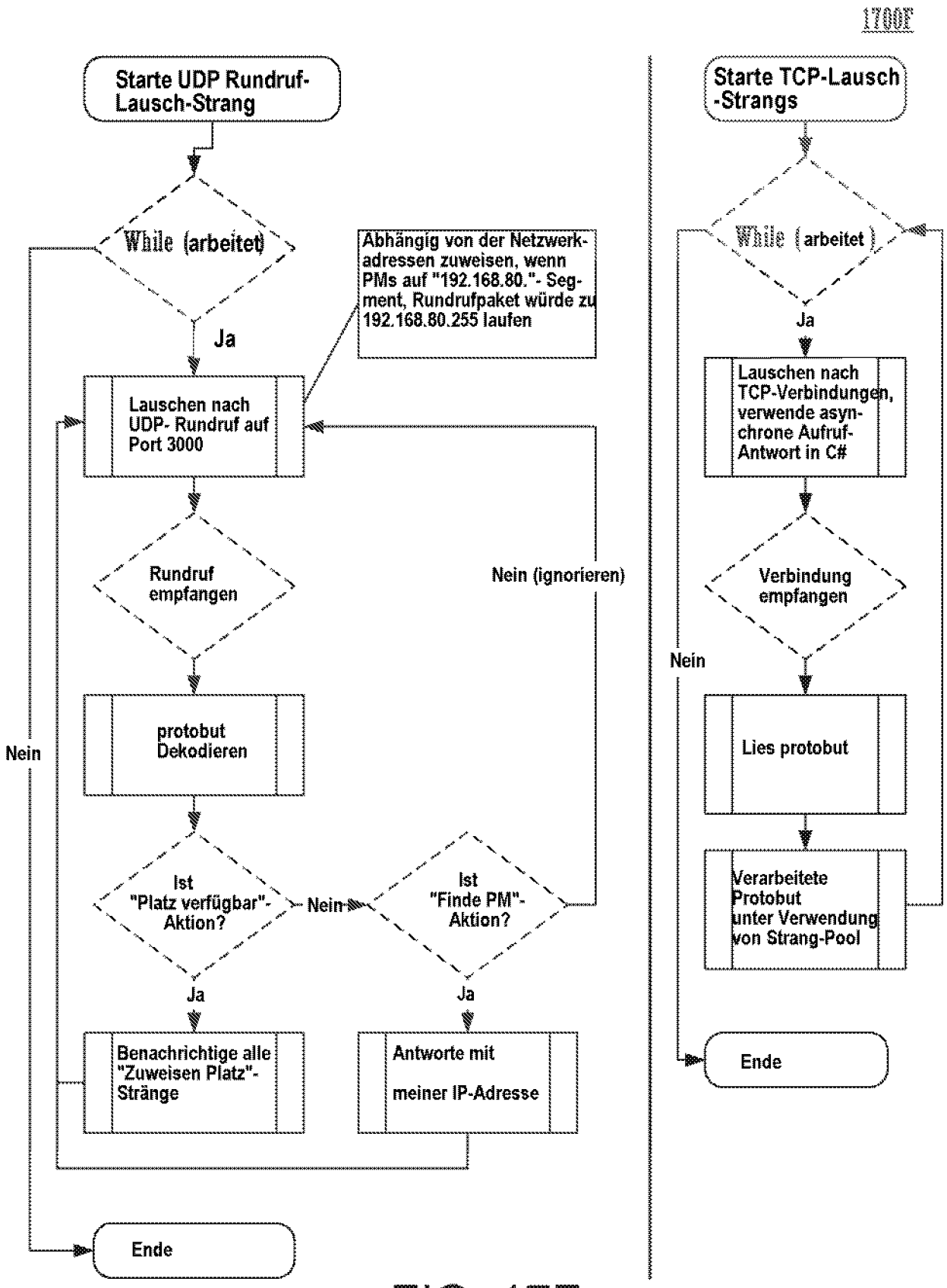

1700F
Starte UDP Rundruf-Lausch-Strang
While (arbeitet)
Ja
Abhängig von der Netzwerkadressen zuweisen, wenn PMs auf "192.168.80."- Segment, Rundrufpaket würde zu 192.168.80.255 laufen
Lauschen nach UDP- Rundruf auf Port 3000
Rundruf empfangen
Nein (ignorieren)
protobut Dekodieren
Nein
Ist "Platz verfügbar"-Aktion?
Nein
Ist "Finde PM"-Aktion?
Ja
Ja
Benachrichtige alle "Zuweisen Platz"-Stränge
Antworte mit meiner IP-Adresse
Ende
Starte TCP-Lausch-Strangs
While (arbeitet)
Ja
Lauschen nach TCP-Verbindungen, verwende asynchrone Aufruf-Antwort in C#
Verbindung empfangen
Nein
Lies protobut
Verarbeitete Protobut unter Verwendung von Strang-Pool
Ende

FIG. 17F