US 20240232840A9

(19) **United States**

(12) **Patent Application Publication**

Paradise et al.

(10) **Pub. No.: US 2024/0232840 A9**

(48) **Pub. Date: Jul. 11, 2024**

CORRECTED PUBLICATION

(54) **SYSTEM AND METHOD FOR REMOTELY MODIFYING CLOUD-BASED CLIENT APPLICATIONS**

(71) Applicant: **Skillz Platform Inc.**, Las Vegas, NV (US)

(72) Inventors: **Andrew Paradise**, San Francisco, CA (US); **Meidad Glory**, San Francisco, CA (US); **Vatsal Bhardwaj**, San Francisco, CA (US)

(21) Appl. No.: **18/491,377**

(22) Filed: **Oct. 20, 2023**

Prior Publication Data

(15) Correction of US 2024/0135349 A1 Apr. 25, 2024
See (22) Filed.

(65) US 2024/0135349 A1 Apr. 25, 2024

Related U.S. Application Data

(60) Provisional application No. 63/380,333, filed on Oct. 20, 2022.

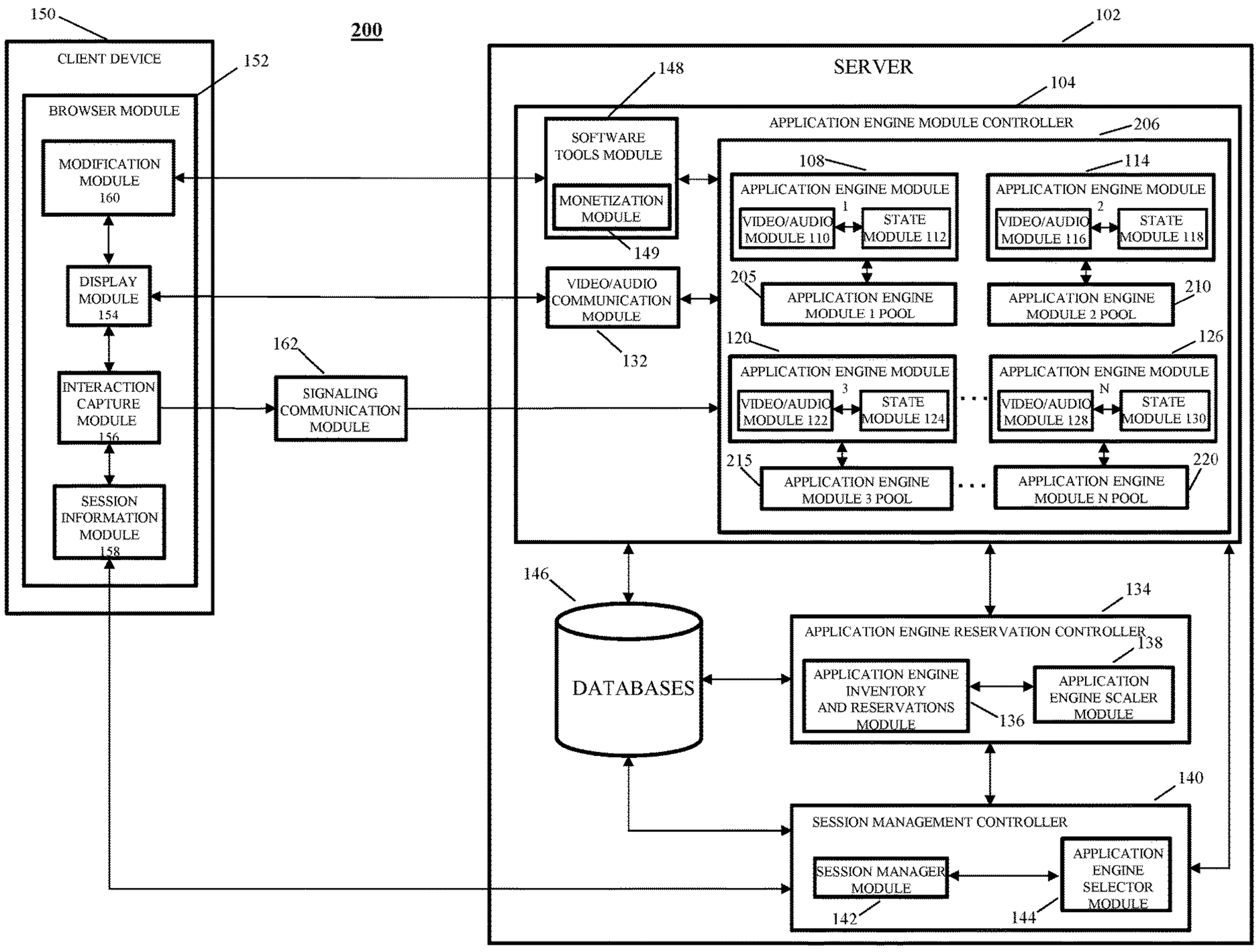
Publication Classification

(51) **Int. Cl.**
G06Q 20/12 (2006.01)

(52) **U.S. Cl.**
CPC ***G06Q 20/12*** (2013.01); *G06Q 2220/165* (2013.01)

(57) **ABSTRACT**

Methods, systems, and apparatus, including computer programs encoded on a computer storage medium, for remotely modifying cloud-based client applications. The method can include: receiving a first request from a first client device of a user to initiate an interactive session with a cloud-based client application; reserving, in response to the first request, an application engine for executing the cloud-based client application remotely from the first client device; receiving a second request from the first client device to modify the cloud-based client application executing in the application engine using one or more software tools, wherein the one or more software tools are configured to execute remotely from the first client device; modifying the cloud-based client application in the application engine using the one or more software tools; and providing media data associated with the modified cloud-based client application to the first user via a display application on the first client device.



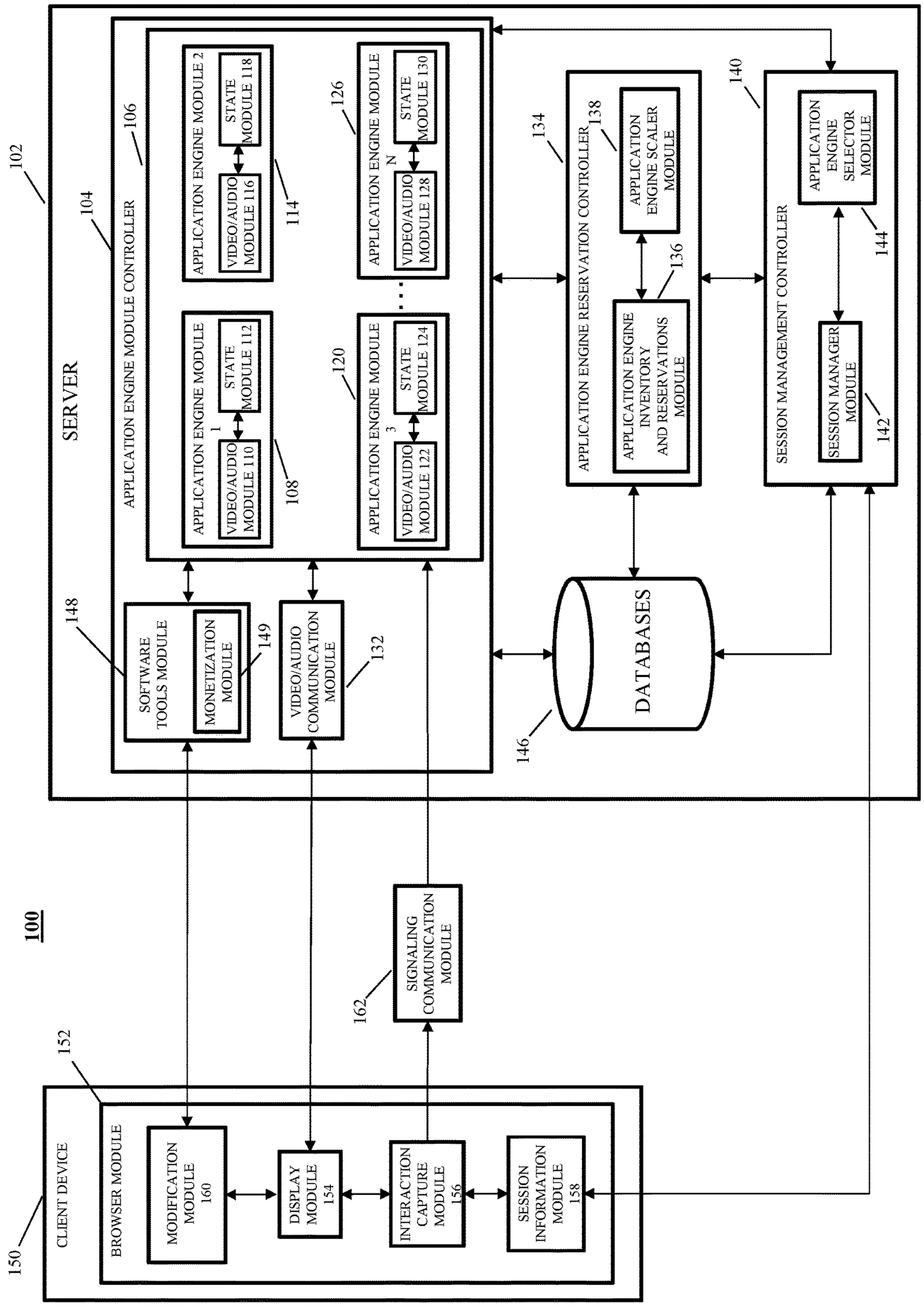

100
SERVER
102
APPLICATION ENGINE MODULE CONTROLLER
104
106
APPLICATION ENGINE MODULE 2
STATE MODULE 118
VIDEO/AUDIO MODULE 116
114
APPLICATION ENGINE MODULE 1
STATE MODULE 112
VIDEO/AUDIO MODULE 110
108
APPLICATION ENGINE MODULE N
126
STATE MODULE 130
VIDEO/AUDIO MODULE 128
APPLICATION ENGINE MODULE 3
120
STATE MODULE 124
VIDEO/AUDIO MODULE 122
SOFTWARE TOOLS MODULE
148
MONETIZATION MODULE
149
VIDEO/AUDIO COMMUNICATION MODULE
132
APPLICATION ENGINE RESERVATION CONTROLLER
134
APPLICATION ENGINE SCALER MODULE
138
APPLICATION ENGINE INVENTORY AND RESERVATIONS MODULE
136
DATABASES
146
SESSION MANAGEMENT CONTROLLER
140
APPLICATION ENGINE SELECTOR MODULE
144
SESSION MANAGER MODULE
142
SIGNALING COMMUNICATION MODULE
162
CLIENT DEVICE
150
BROWSER MODULE
152
MODIFICATION MODULE 160
DISPLAY MODULE 154
INTERACTION CAPTURE MODULE 156
SESSION INFORMATION MODULE 158

FIG. 1

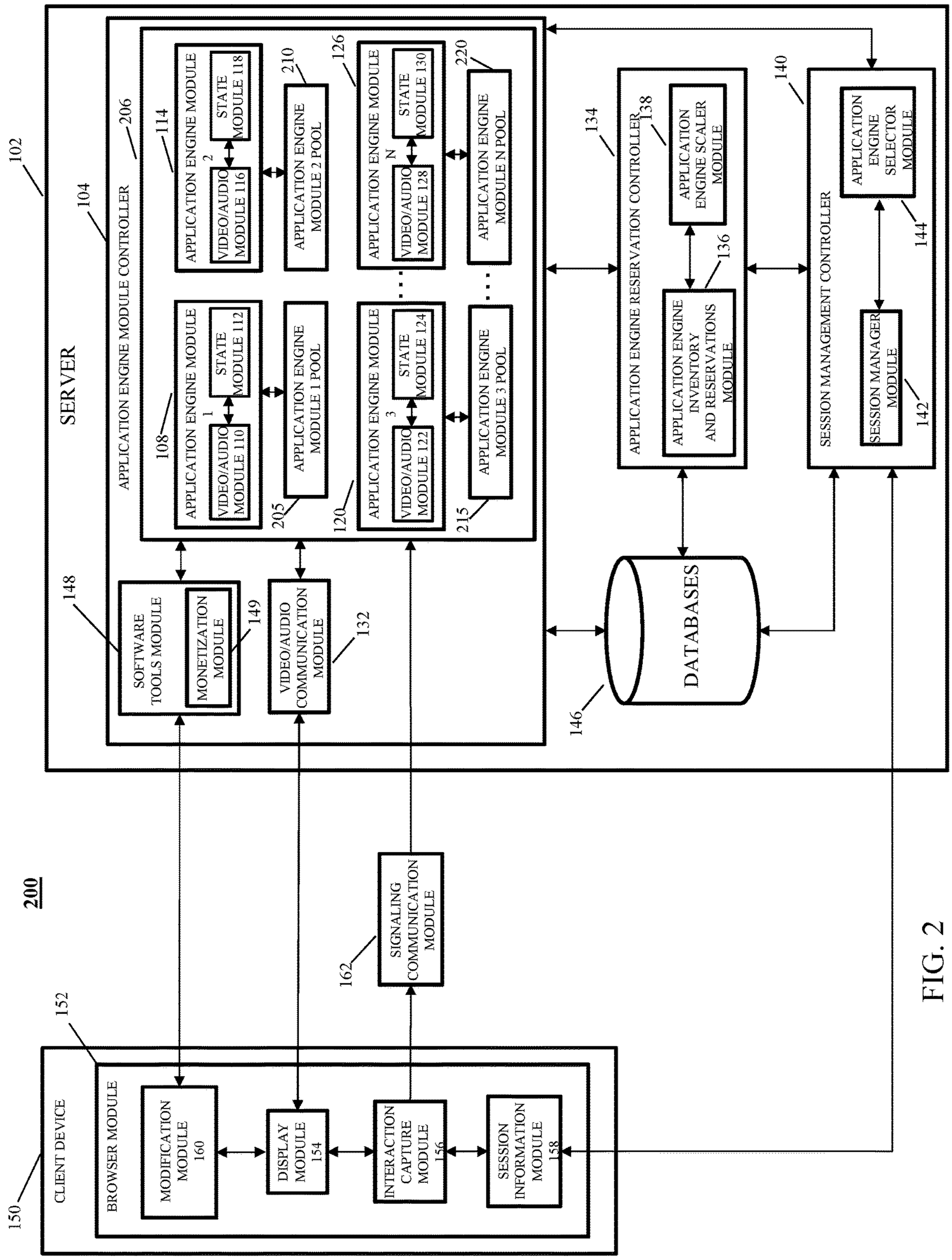

200
CLIENT DEVICE
150
BROWSER MODULE
152
MODIFICATION MODULE 160
DISPLAY MODULE 154
INTERACTION CAPTURE MODULE 156
SESSION INFORMATION MODULE 158
SIGNALING COMMUNICATION MODULE
162
SERVER
102
APPLICATION ENGINE MODULE CONTROLLER
104
APPLICATION ENGINE MODULE 1
108
VIDEO/AUDIO MODULE 110
STATE MODULE 112
APPLICATION ENGINE MODULE 2
114
VIDEO/AUDIO MODULE 116
STATE MODULE 118
APPLICATION ENGINE MODULE N
120
VIDEO/AUDIO MODULE 122
STATE MODULE 124
APPLICATION ENGINE MODULE 126
VIDEO/AUDIO MODULE 128
STATE MODULE 130
APPLICATION ENGINE MODULE 1 POOL
205
APPLICATION ENGINE MODULE 2 POOL
210
APPLICATION ENGINE MODULE 3 POOL
215
APPLICATION ENGINE MODULE N POOL
220
206
SOFTWARE TOOLS MODULE
148
MONETIZATION MODULE
149
VIDEO/AUDIO COMMUNICATION MODULE
132
APPLICATION ENGINE RESERVATION CONTROLLER
134
APPLICATION ENGINE SCALER MODULE
138
APPLICATION ENGINE INVENTORY AND RESERVATIONS MODULE
136
DATABASES
146
SESSION MANAGEMENT CONTROLLER
140
APPLICATION ENGINE SELECTOR MODULE
144
SESSION MANAGER MODULE
142

FIG. 2

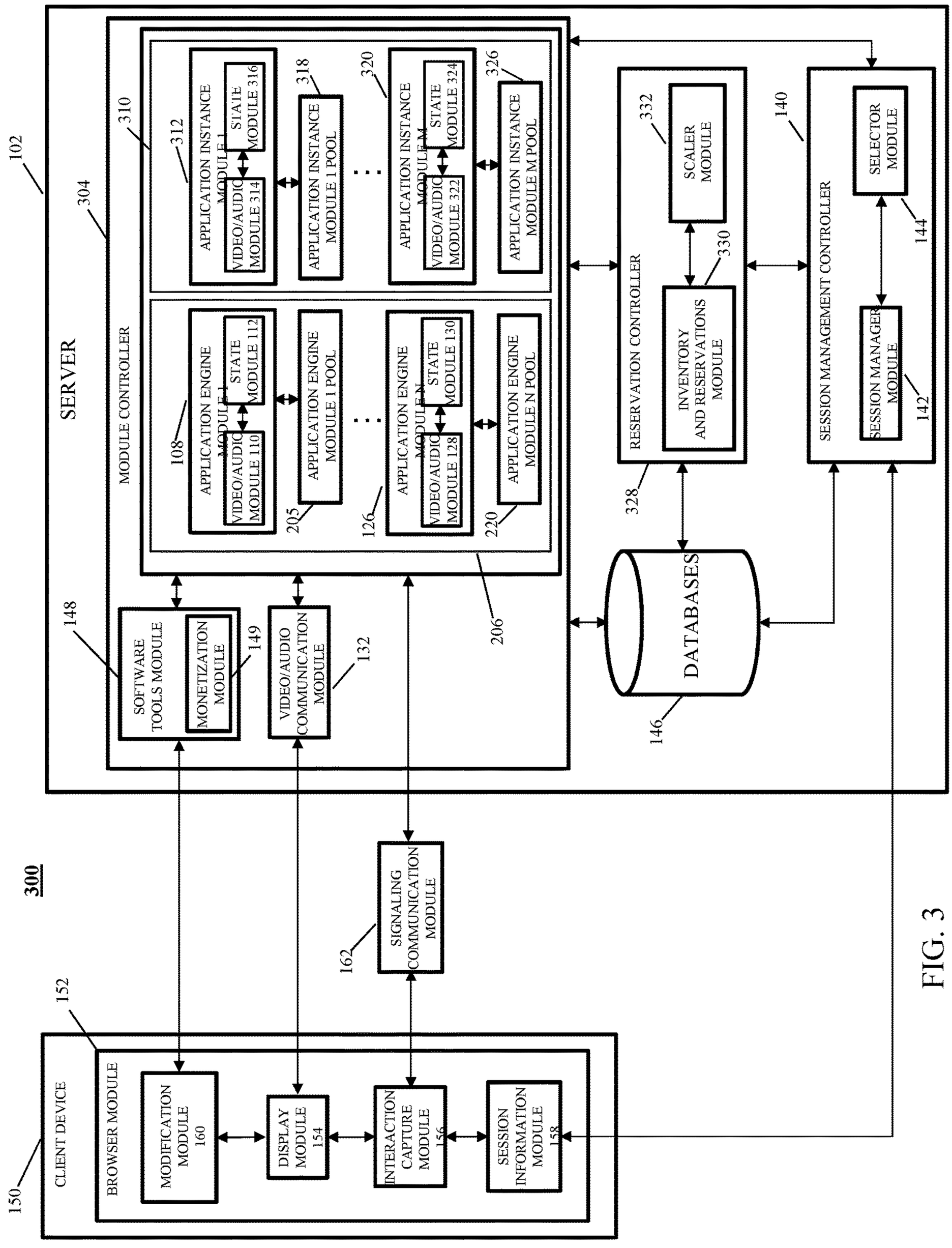

300
SERVER
102
304
MODULE CONTROLLER
310
APPLICATION INSTANCE MODULE 1
312
STATE MODULE 316
VIDEO/AUDIO MODULE 314
APPLICATION INSTANCE MODULE 1 POOL
318
APPLICATION INSTANCE MODULE M
320
STATE MODULE 324
VIDEO/AUDIO MODULE 322
APPLICATION INSTANCE MODULE M POOL
326
APPLICATION ENGINE MODULE 1
108
STATE MODULE 112
VIDEO/AUDIO MODULE 110
APPLICATION ENGINE MODULE 1 POOL
205
APPLICATION ENGINE MODULE N
126
STATE MODULE 130
VIDEO/AUDIO MODULE 128
APPLICATION ENGINE MODULE N POOL
220
206
SOFTWARE TOOLS MODULE
148
MONETIZATION MODULE
149
VIDEO/AUDIO COMMUNICATION MODULE
132
RESERVATION CONTROLLER
328
SCALER MODULE
332
INVENTORY AND RESERVATIONS MODULE
330
DATABASES
146
SESSION MANAGEMENT CONTROLLER
140
SELECTOR MODULE
144
SESSION MANAGER MODULE
142
SIGNALING COMMUNICATION MODULE
162
CLIENT DEVICE
150
BROWSER MODULE
152
MODIFICATION MODULE 160
DISPLAY MODULE 154
INTERACTION CAPTURE MODULE 156
SESSION INFORMATION MODULE 158

FIG. 3

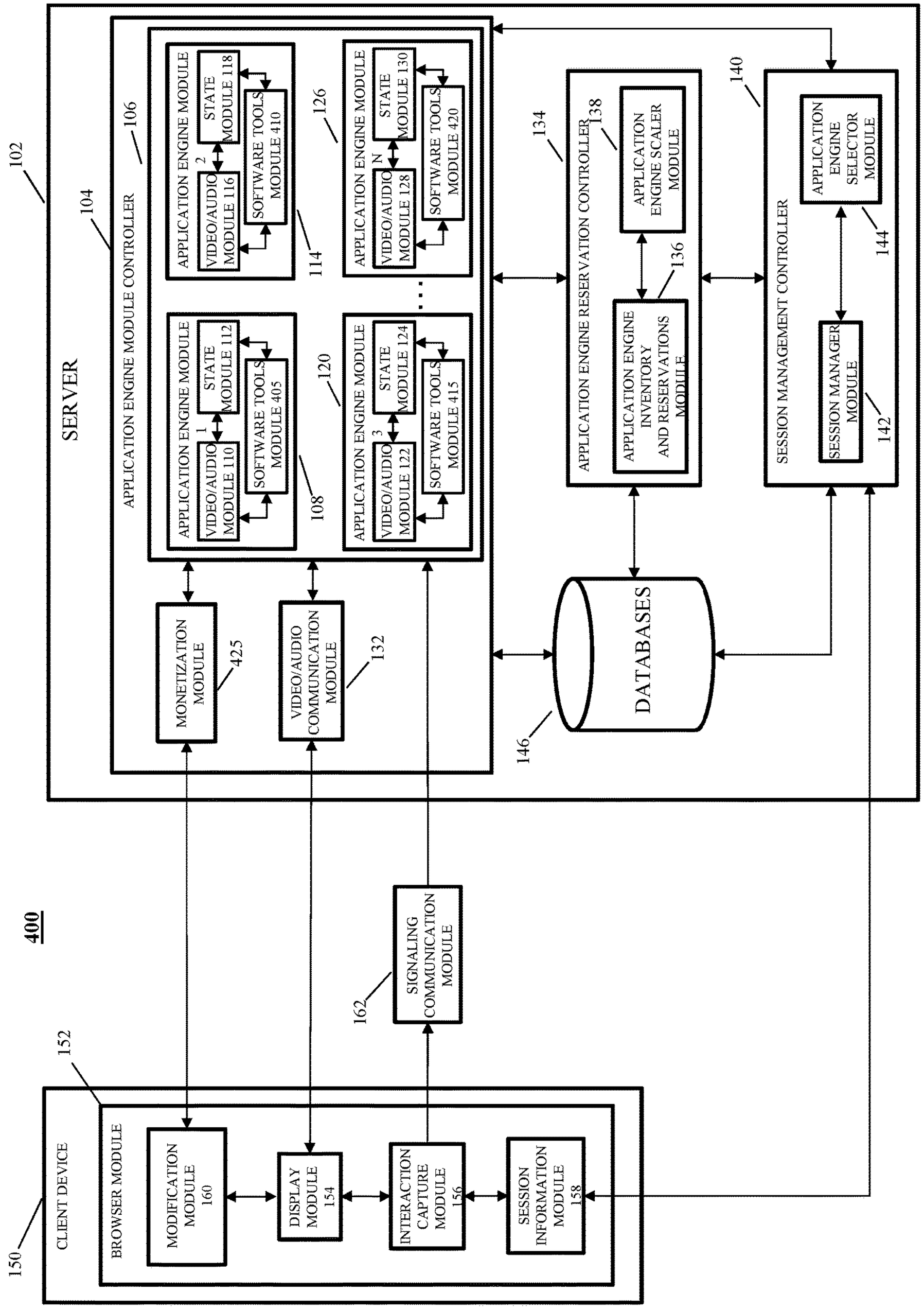

400
102
SERVER
104
APPLICATION ENGINE MODULE CONTROLLER
106
APPLICATION ENGINE MODULE 2
STATE MODULE 118
VIDEO/AUDIO MODULE 116
SOFTWARE TOOLS MODULE 410
114
APPLICATION ENGINE MODULE 1
STATE MODULE 112
VIDEO/AUDIO MODULE 110
SOFTWARE TOOLS MODULE 405
108
APPLICATION ENGINE MODULE N
STATE MODULE 130
VIDEO/AUDIO MODULE 128
SOFTWARE TOOLS MODULE 420
126
APPLICATION ENGINE MODULE 3
STATE MODULE 124
VIDEO/AUDIO MODULE 122
SOFTWARE TOOLS MODULE 415
120
MONETIZATION MODULE
425
VIDEO/AUDIO COMMUNICATION MODULE
132
134
APPLICATION ENGINE RESERVATION CONTROLLER
138
APPLICATION ENGINE SCALER MODULE
136
APPLICATION ENGINE INVENTORY AND RESERVATIONS MODULE
140
SESSION MANAGEMENT CONTROLLER
APPLICATION ENGINE SELECTOR MODULE
144
SESSION MANAGER MODULE
142
DATABASES
146
SIGNALING COMMUNICATION MODULE
162
150
CLIENT DEVICE
152
BROWSER MODULE
MODIFICATION MODULE 160
DISPLAY MODULE 154
INTERACTION CAPTURE MODULE 156
SESSION INFORMATION MODULE 158

FIG. 4

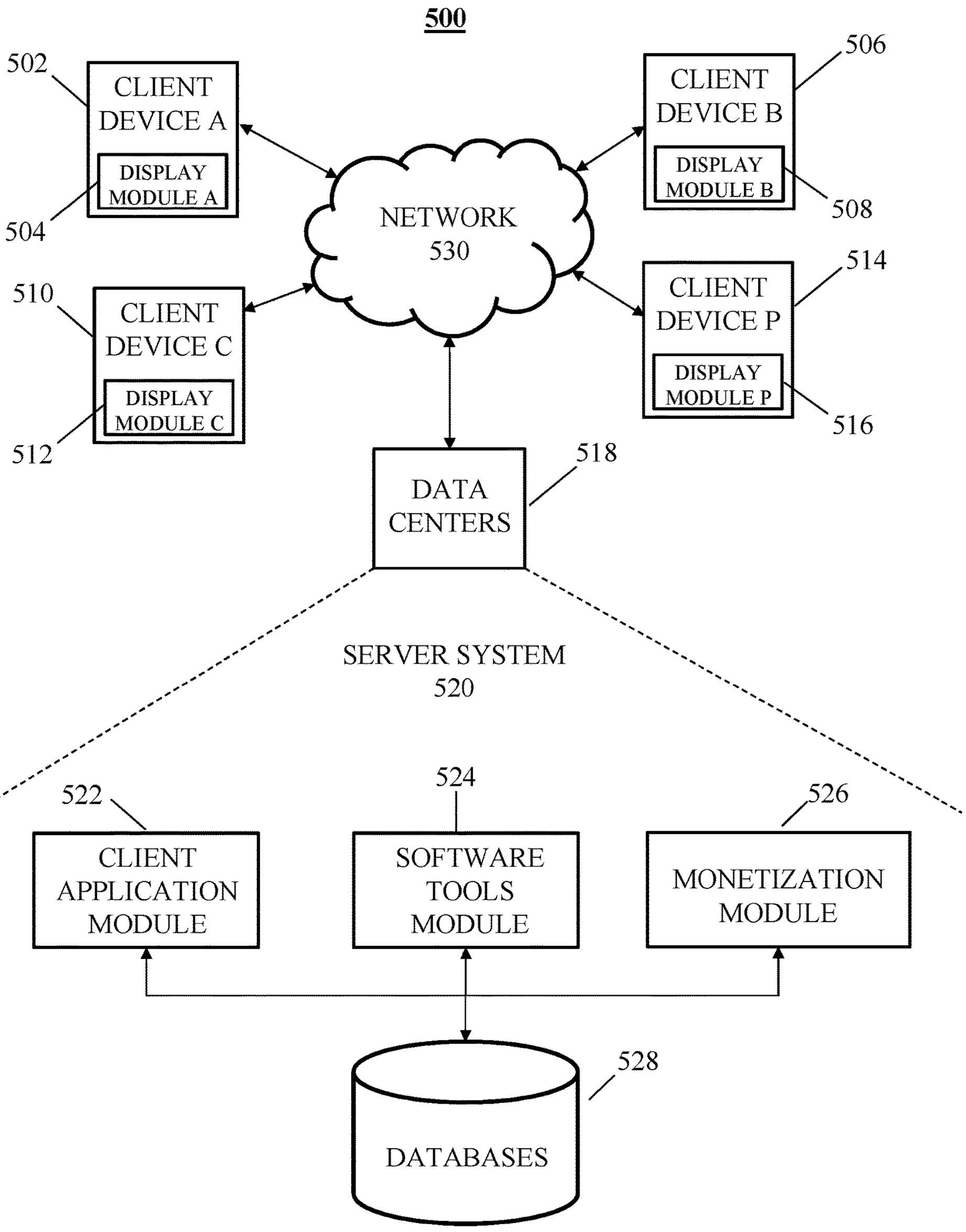

500
502
CLIENT DEVICE A
DISPLAY MODULE A
504
506
CLIENT DEVICE B
DISPLAY MODULE B
508
NETWORK 530
510
CLIENT DEVICE C
DISPLAY MODULE C
512
514
CLIENT DEVICE P
DISPLAY MODULE P
516
518
DATA CENTERS
SERVER SYSTEM 520
522
CLIENT APPLICATION MODULE
524
SOFTWARE TOOLS MODULE
526
MONETIZATION MODULE
528
DATABASES

FIG. 5

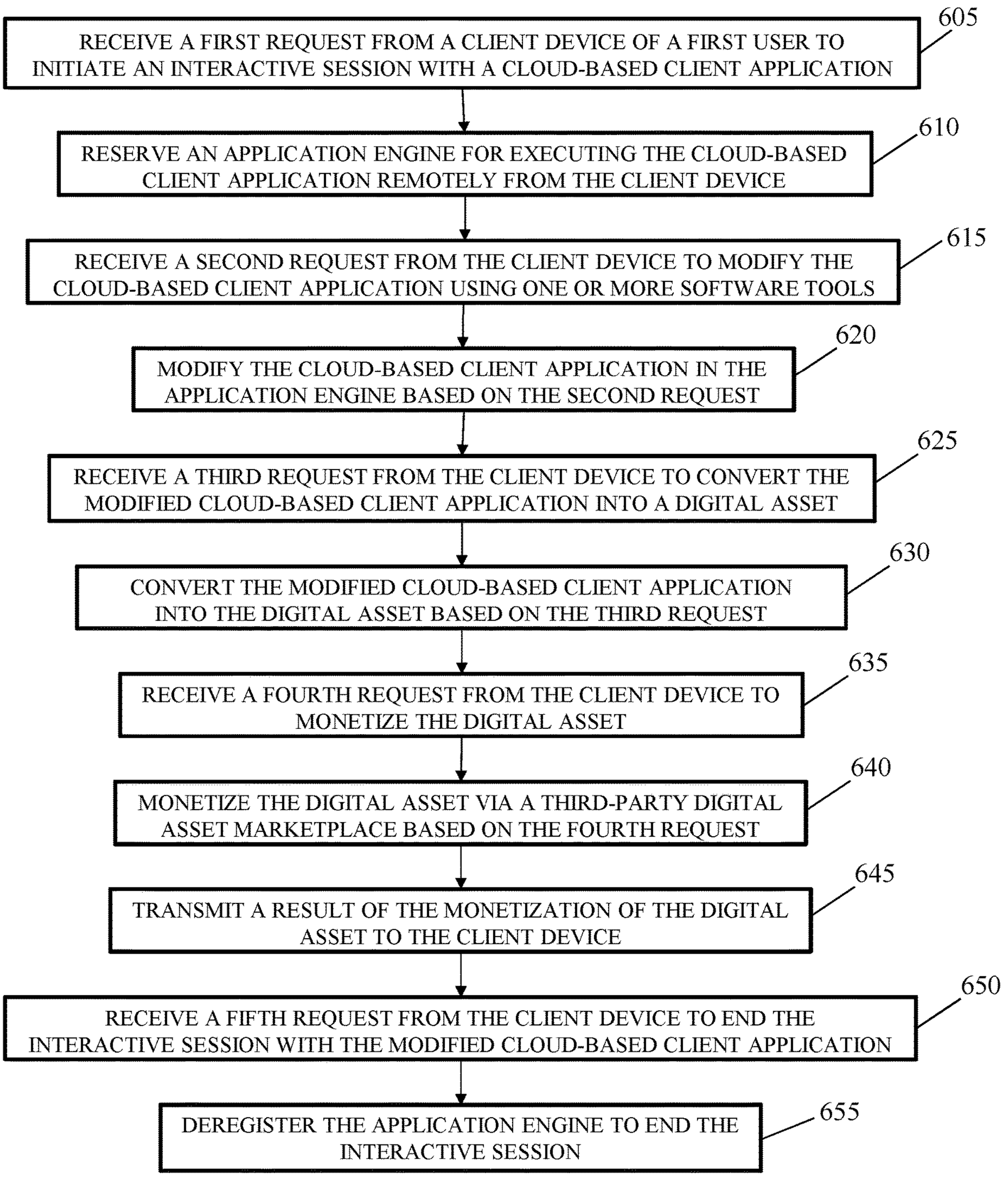

600
RECEIVE A FIRST REQUEST FROM A CLIENT DEVICE OF A FIRST USER TO INITIATE AN INTERACTIVE SESSION WITH A CLOUD-BASED CLIENT APPLICATION
605
RESERVE AN APPLICATION ENGINE FOR EXECUTING THE CLOUD-BASED CLIENT APPLICATION REMOTELY FROM THE CLIENT DEVICE
610
RECEIVE A SECOND REQUEST FROM THE CLIENT DEVICE TO MODIFY THE CLOUD-BASED CLIENT APPLICATION USING ONE OR MORE SOFTWARE TOOLS
615
MODIFY THE CLOUD-BASED CLIENT APPLICATION IN THE APPLICATION ENGINE BASED ON THE SECOND REQUEST
620
RECEIVE A THIRD REQUEST FROM THE CLIENT DEVICE TO CONVERT THE MODIFIED CLOUD-BASED CLIENT APPLICATION INTO A DIGITAL ASSET
625
CONVERT THE MODIFIED CLOUD-BASED CLIENT APPLICATION INTO THE DIGITAL ASSET BASED ON THE THIRD REQUEST
630
RECEIVE A FOURTH REQUEST FROM THE CLIENT DEVICE TO MONETIZE THE DIGITAL ASSET
635
MONETIZE THE DIGITAL ASSET VIA A THIRD-PARTY DIGITAL ASSET MARKETPLACE BASED ON THE FOURTH REQUEST
640
TRANSMIT A RESULT OF THE MONETIZATION OF THE DIGITAL ASSET TO THE CLIENT DEVICE
645
RECEIVE A FIFTH REQUEST FROM THE CLIENT DEVICE TO END THE INTERACTIVE SESSION WITH THE MODIFIED CLOUD-BASED CLIENT APPLICATION
650
DEREGISTER THE APPLICATION ENGINE TO END THE INTERACTIVE SESSION
655

FIG. 6

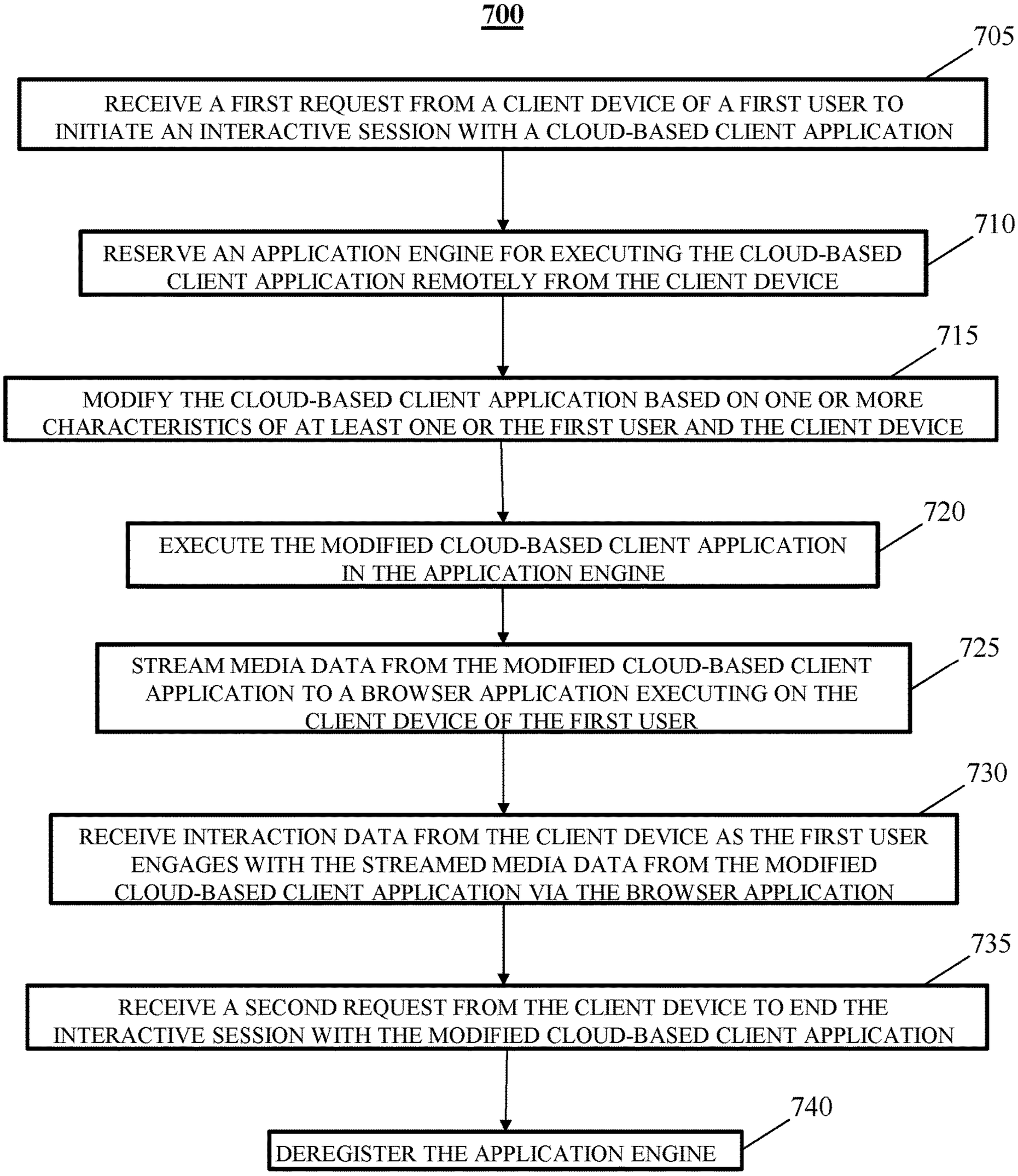

700
705
RECEIVE A FIRST REQUEST FROM A CLIENT DEVICE OF A FIRST USER TO INITIATE AN INTERACTIVE SESSION WITH A CLOUD-BASED CLIENT APPLICATION
710
RESERVE AN APPLICATION ENGINE FOR EXECUTING THE CLOUD-BASED CLIENT APPLICATION REMOTELY FROM THE CLIENT DEVICE
715
MODIFY THE CLOUD-BASED CLIENT APPLICATION BASED ON ONE OR MORE CHARACTERISTICS OF AT LEAST ONE OR THE FIRST USER AND THE CLIENT DEVICE
720
EXECUTE THE MODIFIED CLOUD-BASED CLIENT APPLICATION IN THE APPLICATION ENGINE
725
STREAM MEDIA DATA FROM THE MODIFIED CLOUD-BASED CLIENT APPLICATION TO A BROWSER APPLICATION EXECUTING ON THE CLIENT DEVICE OF THE FIRST USER
730
RECEIVE INTERACTION DATA FROM THE CLIENT DEVICE AS THE FIRST USER ENGAGES WITH THE STREAMED MEDIA DATA FROM THE MODIFIED CLOUD-BASED CLIENT APPLICATION VIA THE BROWSER APPLICATION
735
RECEIVE A SECOND REQUEST FROM THE CLIENT DEVICE TO END THE INTERACTIVE SESSION WITH THE MODIFIED CLOUD-BASED CLIENT APPLICATION
740
DEREGISTER THE APPLICATION ENGINE

FIG. 7

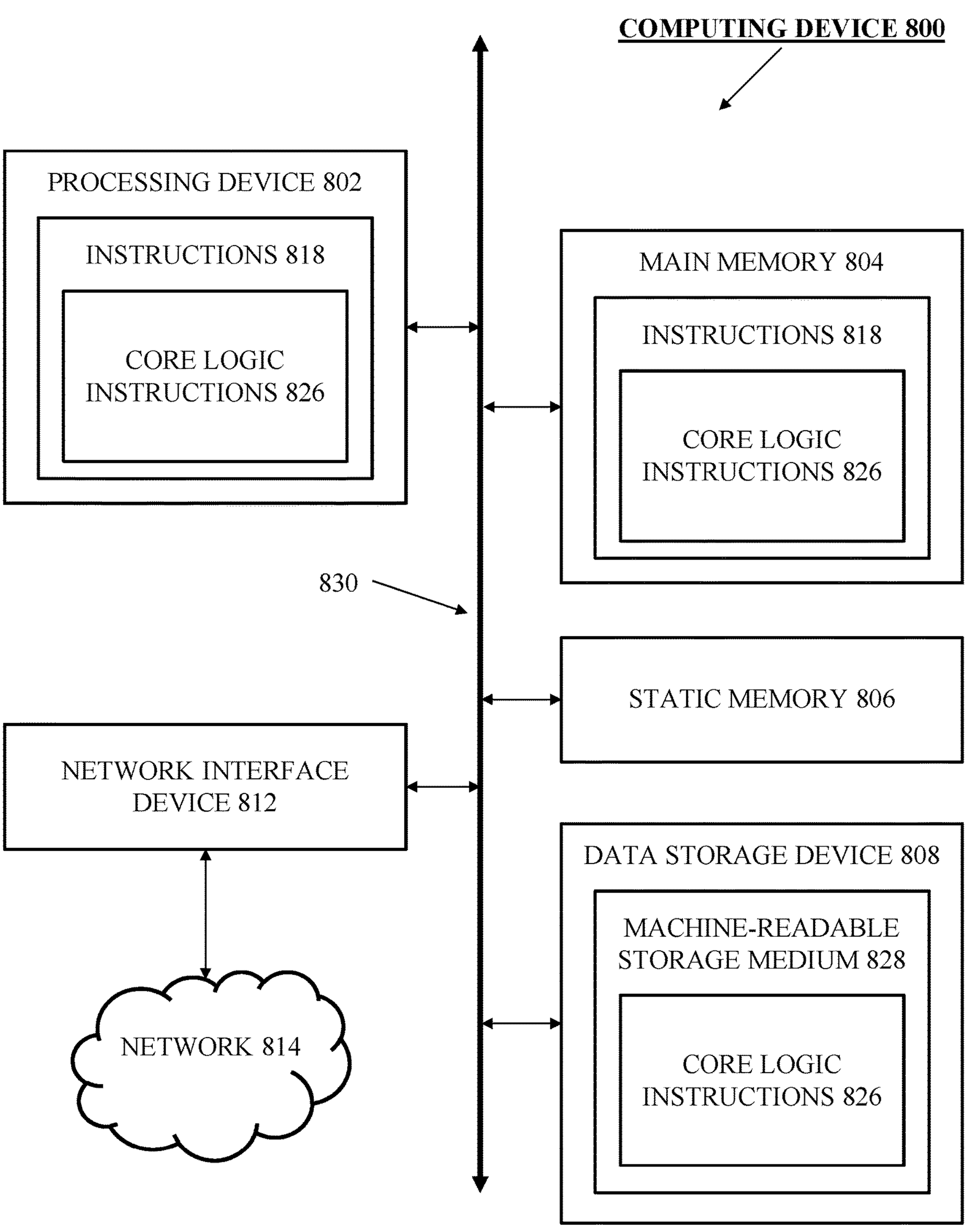

COMPUTING DEVICE 800
PROCESSING DEVICE 802
INSTRUCTIONS 818
CORE LOGIC INSTRUCTIONS 826
MAIN MEMORY 804
INSTRUCTIONS 818
CORE LOGIC INSTRUCTIONS 826
830
STATIC MEMORY 806
NETWORK INTERFACE DEVICE 812
DATA STORAGE DEVICE 808
MACHINE-READABLE STORAGE MEDIUM 828
CORE LOGIC INSTRUCTIONS 826
NETWORK 814

FIG. 8

SYSTEM AND METHOD FOR REMOTELY MODIFYING CLOUD-BASED CLIENT APPLICATIONS

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application claims the benefit of U.S. Provisional Application No. 63/380,333, filed Oct. 20, 2022, the entire contents of which are hereby incorporated by reference herein.

BACKGROUND

[0002] The development and distribution of client applications designed to run on mobile devices (referred to as mobile applications) have increased significantly along with the proliferation of mobile devices. Mobile applications can provide users with a wide variety of functions and features and access to a plethora of information to satisfy virtually any need or interest. Mobile video game modification ("modding" or "mods") is a popular method used by gamers and fans to evolve the gameplay, augment content creation, and add new functionality that is not present in a mobile video game as originally published by the video game developer. Additionally, monetization has been complicated for mobile video game modding. For example, app store-based distribution for mobile applications has made it difficult for game developers and creators to create, distribute, and monetize modded games. In some cases, game developers have purchased popular mods or let the community completely evolve on its own. The lack of a clear monetization model has been an impediment in the growth of modding communities and has made it more difficult for creators to participate.

SUMMARY

[0003] The present invention is directed to a system and method for remotely modifying cloud-based client applications. According to the present invention, a client application can be emulated remotely on a cloud-based client application platform. The video and audio from the cloud-based client application can be streamed in real-time to a browser running on the client device of a user. The user can interact with the cloud-based client application through the local browser as if the cloud-based client application was running natively on the client device. Inputs from the user can be transmitted back to the client application engine to allow the user to control and interact with the cloud-based client application in real-time. The cloud-based client application platform can provide cloud-based software tools to allow users to remotely create, augment, add new or change existing functionality, or otherwise modify cloud-based client applications for distribution and use by other users. The cloud-based client application platform can also provide tools for monetizing modified cloud-based client applications, for example, by allowing users to mint, buy, and sell non-fungible tokens (NFTs) to monetize modified cloud-based client applications in an NFT ecosystem. The present invention can support interaction with the cloud-based client application on any client device running any operating system and independent of the underlying hardware and device characteristics of the client device.

[0004] In an aspect, a method comprises receiving, by at least one data processor, a first request from a first client device of a user to initiate an interactive session with a cloud-based client application, wherein the cloud-based client application is configured to execute on a second client device, and wherein the second client device comprises a computing platform different from the first client device. For example, modifying the cloud-based application executing in the application engine comprises storing data associated with the modified cloud-based application in a database.

[0005] For example, the modified cloud-based application is configured to be retrieved and used in the application engine in a subsequent active session.

[0006] For example, using the one or more software tools is performed via a browser installed on the client device.

[0007] For example, modifying the cloud-based application comprises augmenting content creation, adding new functionality, removing functionality, or changing existing functionality.

[0008] For example, the method can further comprise: receiving, by the at least one data processor, a third request from the first client device to convert the modified cloud-based client application into a digital asset; and converting, by the at least one data processor, the modified cloud-based client application into the digital asset based on the third request.

[0009] For example, the method can further comprise: receiving, by the at least one processor, a fourth request from the first client device to transform the digital asset; monetizing, by the at least one processor, the digital asset via a third-party platform based on the fourth request; and transmitting, by the at least one processor, a result of the transformation of the digital asset to the first client device.

[0010] For example, transforming the digital asset comprises converting the digital asset into a non-fungible token (NFT).

[0011] In an aspect, a system comprises at least one data processor; and memory storing instructions that, when executed by the at least one data processor, cause the at least one data processor to perform operations. The operations comprise receiving a first request from a first client device of a user to initiate an interactive session with a cloud-based client application, wherein the cloud-based client application is configured to execute on a second client device, and wherein the second client device comprises a computing platform different from the first client device. The operations also comprise reserving, in response to the first request, an application engine from a pre-instantiated application engine pool of a plurality of pre-instantiated application engine pools for executing the cloud-based client application remotely from the first client device. The operations also comprise receiving a second request from the first client device to modify the cloud-based client application executing in the application engine using one or more software tools, wherein the one or more software tools are configured to execute on the computing platform different from the first client device. The operations also comprise modifying the cloud-based client application executing in the application engine using the one or more software tools. The operations also comprise providing media data associated with the modified cloud-based client application to the first user via a display application executing on the first client device.

[0012] For example, modifying the cloud-based application executing in the application engine comprises storing data associated with the modified cloud-based application in a database.

[0013] For example, the modified cloud-based application is configured to be retrieved and used in the application engine in a subsequent active session.

[0014] For example, using the one or more software tools is performed via a browser installed on the client device.

[0015] For example, modifying the cloud-based application comprises augmenting content creation, adding new functionality, removing functionality, or changing existing functionality.

[0016] For example, the operations further comprise: receiving a third request from the first client device to convert the modified cloud-based client application into a digital asset; and converting the modified cloud-based client application into the digital asset based on the third request.

[0017] For example, the operations further comprise: receiving a fourth request from the first client device to monetize the digital asset; monetizing the digital asset via a third-party digital asset marketplace based on the fourth request; and transmitting a result of the monetization of the digital asset to the first client device.

[0018] For example, transforming the digital asset comprises converting the digital asset into a non-fungible token (NFT).

[0019] In an aspect, a computer program product comprises a non-transitory machine readable medium storing instructions that, when executed by at least one programmable processor forming part of at least one computing system, cause the at least one programmable processor to perform operations. The operations comprise receiving a first request from a first client device of a user to initiate an interactive session with a cloud-based client application, wherein the cloud-based client application is configured to execute on a second client device, and wherein the second client device comprises a computing platform different from the first client device. The operations also comprise reserving, in response to the first request, an application engine from a pre-instantiated application engine pool of a plurality of pre-instantiated application engine pools for executing the cloud-based client application remotely from the first client device. The operations also comprise receiving a second request from the first client device to modify the cloud-based client application executing in the application engine using one or more software tools, wherein the one or more software tools are configured to execute on the computing platform different from the first client device. The operations also comprise modifying the cloud-based client application executing in the application engine using the one or more software tools; and providing media data associated with the modified cloud-based client application to the first user via a display application executing on the first client device.

BRIEF DESCRIPTION OF THE DRAWINGS

[0020] The embodiments described above will be more fully understood from the following detailed description taken in conjunction with the accompanying drawings. The drawings are not intended to be drawn to scale. For purposes of clarity, not every component may be labeled in every drawing. In the drawings:

[0021] FIG. **1** is a block diagram illustrating an example system for remotely modifying cloud-based client applications;

[0022] FIG. **2** is a block diagram illustrating an example cloud-based client application platform with a plurality of application engine module pools for remotely modifying cloud-based client applications;

[0023] FIG. **3** is a block diagram illustrating an example cloud-based client application platform with a plurality of application engine module pools and a plurality of application instance module pools for remotely modifying cloud-based client applications;

[0024] FIG. **4** is a block diagram illustrating an example cloud-based client application platform for remotely modifying cloud-based client applications;

[0025] FIG. **5** is a block diagram illustrating an example system for remotely modifying and monetizing cloud-based client applications;

[0026] FIG. **6** is a flowchart illustrating an example method for remotely modifying and monetizing cloud-based client applications;

[0027] FIG. **7** is a flowchart illustrating an example method for remotely modifying cloud-based client applications; and

[0028] FIG. **8** is a block diagram of an example computing device that may perform one or more of the operations described herein, in accordance with the present embodiments.

DETAILED DESCRIPTION OF THE INVENTION

[0029] Certain exemplary embodiments will now be described to provide an overall understanding of the principles of the structure, function, manufacture, and use of the devices and methods disclosed herein. One or more examples of these embodiments are illustrated in the accompanying drawings. Those skilled in the art will understand that the devices and methods specifically described herein and illustrated in the accompanying drawings are non-limiting exemplary embodiments and that the scope of the present invention is defined solely by the claims. The features illustrated or described in connection with one exemplary embodiment may be combined with the features of other embodiments. Such modifications and variations are intended to be included within the scope of the present invention. Further, in the present disclosure, like-named components of the embodiments generally have similar features, and thus within a particular embodiment each feature of each like-named component is not necessarily fully elaborated upon.

[0030] The present invention is directed to a system and method for remotely modifying cloud-based client applications. According to the present invention, a client application can be rendered remotely on an emulated client device on a cloud-based client application platform. The video and audio from the cloud-based client application can be streamed in real-time to a browser or other application running on the client device of a user. The user can interact with the cloud-based client application through the local browser or other application as if the cloud-based client application was running natively on the client device. Inputs from the user can be transmitted back to the remote emulated device to allow the user to control and interact with the cloud-based client application in real-time. In some implementations of the present invention, the cloud-based client application platform can provide cloud-based software tools to allow users to remotely create, augment, add new or change existing functionality, or otherwise modify cloud-based client applications for distribution and use by other

users. The cloud-based client application platform can also provide tools for monetizing modified cloud-based client applications, for example, by allowing users to mint, buy, and sell non-fungible tokens (NFTs) to monetize modified cloud-based client applications in an NFT ecosystem. The present invention can support interaction with the cloud-based client application on any client device running any operating system and independent of the underlying hardware and device characteristics of the client device. Cloud-based client applications can include, for example, mobile games or any other suitable type of client application that can be remotely operated from a client device. The present invention can simplify the process of client application modification, distribution, and monetization, since users would not need to install each client application, can try many different client applications easily, and can switch and move between client applications with ease on any client device. Merely for purposes of discussion and not limitation, the present disclosure can refer to cloud-based mobile games as an exemplary cloud-based client application to illustrate various aspects of the present invention. However, the present invention can be used in and with any suitable type of client application that is capable of being modified, can be rendered remotely, and can be operated remotely on a client device.

[0031] FIG. **1** is a block diagram illustrating an example cloud-based client application platform **100** for remotely modifying cloud-based client applications, in accordance with embodiments of the disclosure. The cloud-based client application platform **100** can include a server system **102**. The server can be cloud-based (e.g., Amazon Web Services, Google Cloud, Microsoft Azure, or other suitable cloud-based infrastructure provider) or be comprised of dedicated server hardware. The server system **102** can include an application engine module controller **104**. The application engine module controller **104** can include a plurality of application engine modules **106** to support any appropriate type of cloud-based client application. In embodiments, each application engine module of the plurality of application engine modules **106** can be configured to emulate a particular type and configuration of client device for executing a cloud-based client application. For example, in some implementations of the present invention, each or any of the application engine modules can comprise a suitable client device emulator. Each cloud-based client application can execute within a respective application engine module in the plurality of application engine modules **106** as if the client application was executing natively on the given client device. The application engine module controller **104** can allow registration/deregistration of application engine modules, atomic reservation of application engine modules, querying of application engine modules, and the like. The plurality of application engine modules **106** can include a first application engine module **108**, a second application engine module **114**, a third application engine module **120**, . . . , and a Nth application engine module **126**, where N can be any suitable natural number and can vary over time, as discussed below. Each application engine module **108**, **114**, **120**, . . . **126** can include a video/audio module for managing video and/or audio media data generated by the cloud-based client application executing in the respective application engine module and transmitting such video and/or audio media data to a client device for display to a user. Each application engine module **108**, **114**, **120**, . . . **126** can also include a state module for receiving and processing input (e.g., mapping input to the video/audio stream, etc.) from the client device generated while the user (remotely) interacts with the cloud-based client application. Each state module can also maintain the state of the cloud-based client application executing in the respective application engine module **108**, **114**, **120**, . . . **126**. For example, the first application engine module **108** can include a first video/audio module **110** in communication with a first state module **112**. The second application engine module **114** can include a second video/audio module **116** in communication with a second state module **118**. The third application engine module **120** can include a third video/audio module **122** in communication with a third state module **124**. The Nth application engine module **126** can include a Nth video/audio module **128** in communication with a Nth state module **130**. The application engine module controller **104** can maintain any suitable number of application engine modules and register (i.e., instantiate, allocate, or otherwise create) or deregister (i.e., delete or deallocate) application engine modules as necessary. In an embodiment, the server system **102** can include a plurality of application engine module controllers **104**, with each application engine module controller **104** managing respective pluralities of application engine modules **106** to form one or more application engine module clusters.

[0032] In some implementations of the present invention, each application engine module **108**, **114**, **120**, . . . **126** can run as an essential container in a ReplicaSet in Kubernetes or the like with, for example, a webRTC bridge (a media channel for video/audio media data), OpenGL, or other suitable technology to support real-time media communication capabilities with browsers or other appropriate applications running on the client devices. A ReplicaSet is a process that can run multiple instances of an application engine module and keep the specified number of application engine modules constant. The ReplicaSet can maintain the specified number of application engine module instances running in or with the application engine module controller **104** at any given time to prevent users from losing access to their cloud-based client application when an application engine module fails or is inaccessible. In an embodiment, each application engine module **108**, **114**, **120**, . . . **126** can also include an envoy proxy (as either or both an egress proxy or an ingress proxy) that can handle health checks and the like and, for example, CORS (Cross-Origin Resource Sharing) requests or other like requests. Additionally or alternatively, each application engine module **108**, **114**, **120**, . . . **126** can include a debug bridge for receiving and processing debug and other application engine commands (e.g., from the application engine reservation controller **134** discussed below).

[0033] In an embodiment, the application engine module controller **104** can include a video/audio communication module **132** to support media communication between an application engine module of the plurality of application engine modules **106** and the client device. In an alternative embodiment, the video/audio communication module **132** can reside within the server system **102** but outside of and in communication with the application engine module controller **104**. Additionally or alternatively, each or any application engine module of the plurality of application engine modules **106** can include or be in communication with a respective video/audio communication module **132**. In other

words, each or any of a plurality of video/audio communication modules **132** can reside in or be associated with each or any of the application engine modules of the plurality of application engine modules **106**. In some implementations of the present invention, the video/audio communication module **132** can comprise a TURN (Traversal Using Relay NAT) server or the like (e.g., a STUN (Session Traversal Utilities for NAT) server, etc.) to support communication of video/audio media data between an application engine module and the client device. Additionally or alternatively, the video/audio communication module **132** can comprise a suitable encoder to encode video to, for example, h.264 or VP8 for WebRTC or the like. However, other suitable server communication protocols, encoding techniques, and/or transmission protocols for video/audio media data are possible.

[0034] The server system **102** can include an application engine reservation controller **134** in communication with the application engine module controller **104**. In embodiments, the application engine reservation controller **134** can control various aspects of the plurality of application engine modules **106**, including, for example, orchestrating application engine scaling, controlling application engine access, managing application engine module initialization for users, and the like. The application engine reservation controller **134** can include an application engine inventory and reservations module **136**. The application engine inventory and reservations module **136** can track and manage the inventory of all application engine modules in the plurality of application engine modules **106**. The application engine inventory and reservations module **136** can also grant application engine reservations for use in active sessions. In some implementations of the present invention, when a user requests initiation of an active session with a cloud-based client application, the application engine inventory and reservations module **136** can reserve an appropriate application engine module, load or cause to load the reserved application engine module with the requested cloud-based client application (if not already loaded or pre-loaded), populate or cause to populate the loaded cloud-based client application with the appropriate user and/or application data (e.g., user profile information, user account information, application state information, etc. to support user interaction and engagement with the cloud-based client application), and bind or otherwise link the reserved application engine module to that active session. If no application engine module is available for the active session, the application engine inventory and reservations module **136** can instruct the application engine module controller **104** (e.g., via the application engine scaler module **138**, discussed below) to add and register a new application engine module.

[0035] In some implementations of the present invention, the registration process can be performed asynchronously. For example, the application engine inventory and reservations module **136** can wait for the new application engine module to be added and reserved by polling the application engine module controller **104** periodically or receiving an appropriate message from the application engine module controller **104** once the task has been completed. Once the application engine module is registered, it can be reserved for and bound to the active session. In some implementations of the present invention, the application engine module that has been reserved for the active session can be accessible only to the user that requested it, and only for the lifespan of the active session. In other words, an application engine module that has been reserved for a particular active session may not be used or accessed by another user during that active session. In some implementations of the present invention, the data stored in or otherwise used by the reserved application engine module during the active session can be ephemeral. For example, the application engine module can be deregistered at the end of an active session. Accordingly, any and all data stored or otherwise used by the application engine module during that active session can be deleted, for example, no later than the deregistration of the application engine module. In an alternative embodiment, the data stored in or otherwise used by the reserved application engine module during the active session can be stored (e.g., in the databases **146**), retrieved, and used in and with a subsequent active session by the user. Once the application engine module has been deregistered, it can be reused for another active session or its computing resources can be reallocated to a different application engine module.

[0036] In embodiments, the application engine reservation controller **134** can include an application engine scaler module **138** in communication with the application engine inventory and reservations module **136**. The application engine scaler module **138** can respond to inventory and reservation changes by adding and instantiating new application engine modules to the plurality of application engine modules **106**. The application engine scaler module **138** can proactively scale the number of application engine modules in the plurality of application engine modules **106** based on, for example, usage trends or other suitable metrics to maintain a sufficiently-sized pool of reservable application engine modules. For example, the application engine scaler module **138** can store a predetermined threshold (e.g., provided or generated based on usage trends or other appropriate metrics) for the size of the pool of the plurality of application engine modules **106**. If the number of reservable application engine modules decreases below the predetermined threshold, the application engine scaler module **138** can add new application engine modules to the plurality of application engine modules **106** to increase the number of reservable application engine modules above the predetermined threshold. If the increase in the number of reservable application engine modules becomes too great (e.g., above a second predetermined threshold), the application engine scaler module **138** can either do nothing or can delete or otherwise deallocate unneeded extra application engine modules from the plurality of application engine modules **106**. Additionally or alternatively, suitable machine learning/artificial intelligence techniques can be used by the application engine scaler module **138** to appropriately and dynamically scale the number of reservable application engine modules in the plurality of application engine modules **106**. For example, a machine learning model can be trained based on usage data from users (e.g., usage trends, peak usage, off-hour usage, etc.). The machine learning model can then be used to dynamically add and delete/deallocate application engine modules in the plurality of application engine modules **106**. The machine learning model can be updated or otherwise adapted as the usage data evolves over time.

[0037] In some implementations of the present invention, the plurality of application engine modules **106** can be associated with or otherwise correspond to and be in communication with one or more application engine module

pools, which can be managed by, for example, the application engine inventory and reservations module **136** and/or the application engine scaler module **138**. For example, the application engine modules for each or any emulated client device supported by the cloud-based client application platform **100** can be maintained in a single, large application engine module pool. Application engine modules for active sessions requested by users can be allocated, reserved, and instantiated from such a single application engine module pool by, for example, the application engine inventory and reservations module **136** and/or the application engine scaler module **138**. However, accessing and instantiating each application engine module from such a single, large application engine module pool can create potential bottlenecks. For example, it can take a substantial amount of time and computing resources to spin-up each application engine module separately, particularly when there could be tens of thousands, hundreds of thousands, or even millions of users for which corresponding application engine modules are being instantiated for the cloud-based client application platform **100** at any given time. The resulting allocation delays could adversely affect performance of and user interaction with the cloud-based client application platform **100**, which could result in user dissatisfaction and even user churn.

[0038] FIG. **2** is a block diagram illustrating an example cloud-based client application platform **200** with a plurality of application engine module pools for remotely modifying cloud-based client applications, in accordance with embodiments of the disclosure. In some implementations of the present invention, each or any of a plurality of application engine modules **206** can be associated and in communication with a separate application engine module pool that can be maintained by, for example, the application engine inventory and reservations module **136** and/or the application engine scaler module **138**, with each application engine module pool supporting an individual type of application engine module or a set of related types of application engine modules. The plurality of application engine modules **206** can operate in a manner similar to that discussed and described with respect to the plurality of application engine modules **106** in FIG. **1**. In the embodiment illustrated in FIG. **2**, each cloud-based client application can have or be associated with its own individual application engine module pool. For example, each pool of the plurality of application engine module pools can be configured to support a single cloud-based client application, such that each pool of the plurality of application engine module pools can be allocated on a per cloud-based client application basis. For purposes of illustration and not limitation, a klondike solitaire mobile game designed to run on an Android smartphone can be supported by the first application engine module **108**, which can be associated with a first application engine module pool **205** of first application engine modules. A klondike solitaire PC game designed to run on a PC laptop can be supported by the second application engine module **114**, which can be associated with a second application engine module pool **210** of second application engine modules. A klondike solitaire mobile game designed to run on an iOS smartphone can be supported by the third application engine module **120**, which can be associated with a third application engine module pool **215** of third application engine modules. A klondike solitaire game designed to run on a game console can be supported by the Nth application engine module **126**, which can be associated with an Nth application engine module pool **220** of Nth application engine modules, and so forth. Alternatively, each pool of the plurality of application engine module pools can be configured to support a set of related application engine modules (e.g., application engine modules that emulate Android smartphones, such as with a particular operating system version), such that each pool of the plurality of application engine module pools can be allocated on a per application engine module-type basis. Any suitable number of application engine module pools can be maintained and supported for any appropriate number of application engine modules by, for example, the application engine inventory and reservations module **136** and/or the application engine scaler module **138**, depending on, for instance, the number of cloud-based client applications (or, alternatively, application engine module types) supported by the system or any appropriate subset thereof.

[0039] In some implementations of the present invention, each or any of the application engine modules in each pool of the plurality of application engine module pools can be "pre-warmed." For example, the application engine modules in each pool of the plurality of application engine module pools can be pre-allocated and pre-instantiated within the application engine module pool and then put into standby or sleep mode by, for instance, the application engine inventory and reservations module **136** and/or the application engine scaler module **138**. By spinning up the application engine modules in each application engine module pool in the background and maintaining those spun-up application engine modules in a standby mode, an application engine module can be immediately (or nearly immediately) reserved and assigned to a user upon request of an active session with little or no concomitant delay, since awakening an application engine module that is in standby mode can take substantially less time than instantiating the application engine module from scratch. Once an application engine module in an application engine module pool is reserved and assigned, the reserved and assigned application engine module can be immediately (or nearly immediately) replaced by another "pre-warmed" application engine module from the corresponding application engine module pool so that it can be ready for immediate (or nearly immediate) reservation by and assignment to the next user requesting an active session. Such a "pre-warmed" replacement process can continue as each application engine module in an application engine module pool is reserved and assigned. In such a manner, application engine modules from each pool of the plurality of application engine module pools can be allocated to users in real-time. In the present disclosure, "real-time" can refer to processing that can occur instantaneously or within a short period of time (e.g., a few seconds) so that there are minimal delays between when the request is received and when the request is processed and the response provided. The computing resources allocated to a reserved application engine module for an active session can be deleted or otherwise deallocated immediately (or nearly immediately) once the active session ends and the reserved application engine module is no longer needed for that active session. In some implementations of the present invention, such freed up computing resources can be re-allocated by, for example, the application engine inventory and reservations module **136** and/or the application engine scaler module **138** to add and pre-warm another application engine module in the

associated application engine module pool or a different application engine module pool, rather than maintaining computing resources for a reserved application engine module that is no longer needed for an inactive session.

[0040] The size of each or any of the plurality of application engine module pools can be fixed or dynamic. For example, the size of an application engine module pool can be fixed such that a predetermined application engine module pool size number of pre-warmed application engine modules can be maintained in the pool. As pre-warmed application engine modules in an application engine module pool are allocated, the application engine inventory and reservations modules **136** and/or the application engine scaler module **138** can replace the allocated application engine modules with additional pre-warmed application engine modules to maintain the size of the application engine module pool at or around the predetermined application engine module pool size number. The predetermined application engine module pool size number can be based on, for example, usage trends, historical usage information, or other appropriate metrics and the like for the particular application engine module. Accordingly, highly used application engine modules can have a greater predetermined application engine module pool size number while less used application engine modules can have a smaller predetermined application engine module pool size number.

[0041] Additionally or alternatively, the size of an application engine module pool can be dynamic. In some implementations of the present invention, the application engine inventory and reservations module **136** and/or the application engine scaler module **138** can monitor in real-time the usage trends of each or any of the plurality of application engine modules **206**. For example, as usage trends increase for a particular application engine module, the application engine scaler module **138** can increase the size of the associated application engine module pool (and, therefore, the number of pre-warmed application engine modules maintained in the application engine module pool) to accommodate the greater usage so that application engine module allocation can continue in real-time without added delay. Conversely, as usage trends decrease for a particular application engine module, the application engine scaler module **138** can decrease the size of the associated application engine module pool (and, therefore, the number of pre-warmed application engine modules maintained in the application engine module pool) to accommodate the lower usage so that computing resources are not unnecessarily overused and can be allocated elsewhere. In such an embodiment, the size of each or any of the application engine module pools can be dynamically altered to accommodate periods of increased use (e.g., spikes in usage) as well as lulls in usage to maintain real-time allocation of application engine modules.

[0042] In some implementations of the present invention, the application engine inventory and reservations module **136** and/or the application engine scaler module **138** can include a suitable predictive model or use appropriate machine learning/artificial intelligence techniques to determine the size of an application engine module pool of the plurality of application engine module pools. For example, a machine learning model can be trained based on usage trend data for each supported application engine module to modify the application engine module pool size for an application engine module to minimize the amount of time to allocate the application engine module and maintain a reduced allocation time for the real-time application engine module allocation. The machine learning model can then be used to dynamically and appropriately modify the size of the application engine module pool associated with the application engine module based on the usage exhibited at a particular time for that application engine module. The machine learning model can be updated or otherwise adapted as the usage trends for the application engine module change and evolve over time, as monitored by, for example, the application engine inventory and reservations module **136** and/or the application engine scaler module **138**. In some implementations of the present invention, the size of each application engine module pool in a subset of the plurality of application engine module pools can be fixed while the size of other application engine module pools in the plurality of application engine module pools can be dynamic. For example, application engine modules that are highly used can be associated with application engine module pools whose sizes are dynamic, while application engine modules that are less used can be associated with application engine module pools whose sizes are fixed. Additionally or alternatively, the size of a particular application engine module pool can be fixed at certain times and dynamic at other times. For example, for an application engine module that generally experiences lower usage, the size of the corresponding application engine module pool can be fixed. However, during (e.g., intermittent) periods of higher usage (e.g., usage spikes), the size of the corresponding application engine module pool can be dynamic. The application engine inventory and reservations module **136** and/or the application engine scaler module **138** can switch the size of the application engine module pool back to fixed once the period of higher usage ends. In this way, the application engine inventory and reservations module **136** and/or the application engine scaler module **138** can adapt to changing usage conditions as necessary to maintain a reduced allocation time for the real-time allocation of application engine modules.

[0043] In some implementations of the present invention, the one or more application engine module pools can be maintained in and with the plurality of application engine modules **206**, as illustrated in FIG. **2**. However, in some implementations of the present invention, each or any of the one or more application engine module pools can be maintained in or with the application engine module controller **104** outside of the plurality of application engine modules **206** but within the server system **102**. Additionally or alternatively, each or any of the one or more application engine module pools can be maintained in or with the server system **102**, but outside of the application engine module controller **104**, such as, for example, in databases **146**. Additionally or alternatively, each or any of the one or more application engine module pools can be maintained in or with a suitable cloud or server storage system that is remote from and in communication with the server system **102**.

[0044] As illustrated in FIGS. **1** and **2**, the server system **102** can include a session management controller **140** for managing active sessions. The session management controller **140** can be in communication with the application engine reservation controller **134**. In some implementations of the present invention, the session management controller **140** can also be coupled to or otherwise in communication with the application engine module controller **104**. The session

management controller **140** can include a session manager module **142** for managing the allocation of application engine modules, including registration, deregistration, and scaling (e.g., in communication and coordination with the application engine inventory and reservations module **136** and the application engine scaler module **138**). The session manager module **142** can also act as coordinator for the server system **102** to support, for example, load balancing and latency. The session manager module **142** can receive a request from the client device to initiate an active session. Additionally, the session manager module **142** can receive session information from the client device (e.g., in addition to or as part of the request) that can be used to select an appropriate application engine module. In an embodiment, the session information received from the client device can include, for example, one or more characteristics of the cloud-based client application for which an active session is requested, one or more characteristics of the client device (e.g., client device type, operating system, etc.), and other like characteristics or data.

[0045] The session management controller **140** can include an application engine selector module **144** in communication with the session manager module **142**. The session manager module **142** can pass the received session information to the application engine selector module **144**, which can use such information to select an appropriate application engine module for the requested active session. For purposes of illustration and not limitation, a user with a non-Android client device, such as, for example, a PC laptop, can request an active session with a (cloud-based) klondike solitaire mobile game designed to run on an Android smartphone. Based on the session information passed to the application engine selector module **144**, the application engine selector module **144** can select an Android application engine for the (cloud-based) klondike solitaire mobile game for the active session. The application engine selector module **144** can be in communication with the application engine inventory and reservations module **136**. Consequently, the application engine selector module **144** can instruct the application engine inventory and reservations module **136** to reserve an Android application engine module for the (cloud-based) klondike solitaire mobile game from the plurality of application engine modules **106** or **206**. Once reserved, the (cloud-based) Android klondike solitaire mobile game application engine module can be bound to the requested active session so that no other user can use that reserved application engine module during the active session. The application engine selector module **144** can issue a command (e.g., passed to the reserved Android mobile game application engine module via the application engine inventory and reservations module **136**) to start the (cloud-based) klondike solitaire mobile game in the Android application engine module. The user can then begin remotely interacting with the (cloud-based) klondike solitaire mobile game via their PC laptop as if the klondike solitaire mobile game was executing natively on the PC laptop itself. If an Android klondike solitaire cloud-based mobile game application engine module is not available, the application engine inventory and reservations module **136** can instruct the application engine scaler module **138** to add and reserve an Android klondike solitaire cloud-based mobile game application engine module to the plurality of application engine modules **106** or **206** in the manner described above.

[0046] In some implementations of the present invention, the session management controller **140** (e.g., the session manager module **142** and the application engine selector module **144**) can coordinate with and control the application engine reservation controller **134** (e.g., the application engine inventory and reservations module **136** and application engine scaler module **138**) to manage the one or more application engine module pools as illustrated and discussed with respect to FIG. **2**. Additionally or alternatively, the session management controller **140** can directly coordinate with and control the application engine module controller **104** to manage the one or more application engine module pools.

[0047] In an embodiment, since the application engine reservation controller **134** can identify and maintain, for example, usage trends for application engine modules (e.g., per user, for all users, or any subset thereof), some implementations of the present invention can also recommend cloud-based client applications to each or any user. In an embodiment, the session manager module **142** can include a recommendations engine or the like. The recommendations engine can receive usage trend information from the application engine reservation controller **134** and use such information to recommend different cloud-based client applications to users. For example, the recommendations engine can recommend the most popular cloud-based client application (e.g., overall, in a particular genre, for users using a particular type of client device, etc.), a list of most popular cloud-based client applications at a particular time or over a range of time (e.g., the past hour, the past day, the past week, etc.), and the like. The recommendations engine can also recommend cloud-based client applications that are similar to or complement one or more cloud-based client applications with which a user has interacted previously. For example, if the user has played a cloud-based klondike solitaire mobile game, the recommendations engine can recommend other cloud-based games in the same or similar genre (e.g., spider solitaire, tri peaks solitaire, etc.), regardless of the device platform on which the game is based (since the device platform will be emulated by one of the plurality of application engine modules **106** or **206**). Other recommendations are possible. Additionally or alternatively, suitable machine learning/artificial intelligence techniques can be used by the recommendation engine to dynamically provide cloud-based client application recommendations to users. For example, a machine learning model can be trained based on usage data from users (e.g., usage trends for all users or any subset thereof, peak usage, off-hour usage, etc.). The machine learning model can then be used to dynamically recommend cloud-based client applications to users based on the usage trends. The machine learning model can be updated or otherwise adapted as the usage data evolves over time. Such recommendations can be communicated to the client device (e.g., as a URL or other link or hyperlink in a modal window or other appropriate notification or message). If a user accepts a recommendation (e.g., by selecting a URL or other link or hyperlink to the cloud-based client application), the session manager module **142** can have the indicated application engine module selected and reserved in the manner described herein to initiate an active session for the user.

[0048] The server system **102** can include one or more databases **146** that can be in communication with one or more of the application engine module controller **104**, the

application engine reservation controller **134**, and the session management controller **140**. The databases **146** can be cloud-based or reside in one or more physical storage systems. The databases **146** can include, for example, any suitable information related to the application engine modules, management of the application engine modules, active session information, historical session information, usage trend or other metric data, cloud-based client application state data, copies of each or any cloud-based client application (or modified cloud-based client application, as discussed below), input from the client devices, session logs, and other like data.

[0049] According to an embodiment, the server system **102** can provide cloud-based software tools and other suitable functionality to allow users to remotely create, augment or otherwise modify cloud-based client applications for distribution via the cloud-based client application platform **100** or **200**. For example, such cloud-based software tools can be used to modify an existing cloud-based client application that executes during an active session in one of the plurality of application engine modules **106** or **206** to create an augmented or modified cloud-based client application. In some implementations of the present invention, the cloud-based software tools can allow users to modify non-mobile applications into cloud-based mobile applications for distribution via the cloud-based client application platform **100** or **200**. In an embodiment, the application engine module controller **104** can include a software tools module **148** in communication with the plurality of application engine modules **106** or **206**. The software tools module **148** can allow "modders" (e.g., client application developers, client application users, etc.) to augment content, add new or change existing functionality, create, or otherwise modify the cloud-based client applications. For example, the software tools module **148** can include or otherwise provide appropriate software tools for creating and editing digital content (e.g., creating and/or editing text, images, video, audio, or any combination thereof) within the cloud-based client applications. Additionally or alternatively, the software tools modules **148** can include or otherwise provide suitable software development tools for generating, editing, and compiling computer code for the cloud-based client applications. Other software tools are possible. The software tools module **148** can include or provide any suitable cloud-based software tools that can be used to build, create, augment, modify, generate, and/or distribute a cloud-based client application via the cloud-based client application platform **100** or **200**. Using such cloud-based software tools, a wide variety of users can be provided with the features and functionality to "mod" cloud-based client applications to augment content creation, add new or change existing functionality, create, or otherwise evolve the cloud-based client applications as desired by the users.

[0050] In an embodiment, the software tools module **148** can be used with each or any of the plurality of application engine modules **106** or **206** to modify the cloud-based client applications supported by the respective application engine modules **106** or **206**. For example, after the application engine reservation controller **134** has reserved, registered and bound an application engine module to an active session (e.g., via the application engine inventory and reservations module **136** in communication with the emulation module controller **104**), the software tools module **148** can provide appropriate tools to modify or otherwise augment the cloud-based client application running in the reserved application engine module. As additional application engine modules are instantiated for other active sessions, the software tools module **148** can provide additional and/or alternative tools to modify or otherwise augment the cloud-based client applications executing in those additional application engine modules. In an embodiment, the software tools module **148** can provide a full suite of diverse software tools to support a wide variety of cloud-based client applications. Additionally or alternatively, the software tools module **148** can provide a subset of software tools for use in remotely modifying or augmenting the cloud-based client applications executing in the application engine modules. In an embodiment, the software tools module **148** can use suitable machine learning/artificial intelligence techniques to automatically and dynamically select and provide to users the appropriate tools based upon, for example, the characteristics of the cloud-based client application (e.g., the type of client application, the type of operating system in which the client application will be run, etc.), the type of modifications being made to the cloud-based client application (e.g., image editing software tools if image modifications are being made, text editing software tools if text editing is being performed, etc.), or the like. For example, a machine learning model can be trained based on tool usage data from users (e.g., which software tools were used and how often, which cloud-based client applications were modified with which tools, etc.). The machine learning model can then be used to dynamically add and select suitable software tools for the user to modify a given cloud-based client application. The machine learning model can be updated or otherwise adapted as the tool usage data evolves over time. For example, the software tools module **148** can maintain a set of core software tools, while additional or alternative software tools can be stored in and automatically retrieved from, for example, the databases **146** as needed and as necessary as the user is remotely modifying a cloud-based client application.

[0051] In an embodiment, once the modifications to the cloud-based client application have been completed, the data stored in or otherwise used by the application engine module (e.g., any or all data associated with the modified cloud-based client application, including the modified cloud-based client application itself) during the active session can be stored in the databases **146** so that the modified cloud-based client application can be retrieved and used in a subsequent active session. Consequently, the modified cloud-based client application can be distributed to and accessed by other users of the cloud-based client application platform **100** or **200**. In an additional embodiment, the software tools module **148** can provide tools and functionality for remotely modifying non-mobile applications (e.g., PC-based applications, game console-based applications, etc.) into cloud-based client applications for distribution via the cloud-based client application platform **100** or **200** in the manner described previously. For example, the software tools module **148** can provide appropriate tools to augment application content to accommodate different sized screens of mobile devices, compilers and other software development tools for creating and/or editing software code and re-compiling the applications to run on a (different) target client device platform and operating system (e.g., iOS, Android, etc.), and the like. The software tools module **148** can also provide suitable tools for uploading the modified or otherwise converted cloud-based

client application to the cloud-based client application platform **100** or **200**. For purposes of illustration and not limitation, the application engine reservation controller **134** can reserve, register and bind a first application engine module to an active session (e.g., via the application engine inventory and reservations module **136** in communication with the emulation module controller **104**) to support the non-mobile application for a first operating system environment (e.g., a PC-based application). The software tools module **148** can then provide the user with suitable tools to convert the non-mobile application to a cloud-based client application. The application engine reservation controller **134** can reserve, register and bind a second application engine module to an active session (e.g., via the application engine inventory and reservations module **136** in communication with the emulation module controller **104**) to support the new cloud-based client application for a second operating system environment (e.g., an iOS application). For example, the binary executable of the cloud-based client application that is output by a compiler (or cross-compiler) provided by the software tools module **148** for the first application engine module can be stored in the second application engine module. The second application engine module can become a new application engine module available in the plurality of application engine modules **106** or **206** that can be instantiated and used by other users, allowing distribution of the new cloud-based client application to a wide variety of users of the cloud-based client application platform **100** or **200**. In an alternative embodiment, the binary executable of the cloud-based client application that is output by a compiler (or cross-compiler) provided by the software tools module **148** for the first application engine module can be distributed directly to and execute on the desired platform (e.g., an application in a mobile app store) in addition to or instead of being saved to another (second) application engine module in the cloud-based client application platform **100** or **200**.

[0052] In some implementations of the present invention, the software tools module **148** can include a monetization module **149**. In an alternative embodiment, the monetization module **149** can be separate from the software tools module **148** within or external to the application engine module controller **104**. The monetization module **149** can provide tools and support for monetizing modified cloud-based client applications. In an embodiment, the monetization module **149** can allow users to mint, buy, and sell non-fungible tokens (NFTs) to monetize modified cloud-based client applications in an NFT ecosystem. Embodiments of the present invention can allow for minting of NFTs on a suitable blockchain, converting and saving users' cloud-based client applications as NFTs, and creating a way for cloud-based client application developers and content creators to earn on cloud-based client applications or instances of the cloud-based client applications (e.g., a particular "mod" of a cloud-based client application, a particular competitive and/or popular match of a cloud-based mobile game application or the like). An NFT is a digital asset that is similar to a cryptocurrency. In contrast to cryptocurrencies, NFTs are not interchangeable. Although it is possible to create and sell several identical NFTs, it is the rarity of the NFT that generally gives it value. NFTs are part of a blockchain, a system in which a record of transactions is maintained across several computers that are linked in a peer-to-peer network. Each NFT contains, for example, owner identification, metadata, safe file links, and other identifying information. It also can include a smart contract requiring, for instance, an additional payment to the original owner for each transfer of ownership or licensed use. Such information makes each NFT unique, and the blockchain keeps a decentralized, secure, and permanent record of who owns the NFT, the provenance of the NFT's ownership, and the rights granted by the NFT, among other information. NFTs can have only one owner at a time. Many NFTs are part of the Ethereum blockchain, which supports both NFTs and the Ethereum cryptocurrency. Other blockchains that support NFTs include, for example, Solana, Zilliqa, Cardano, Flow, and Tezos. Other blockchains supporting NFTs are possible. NFTs can be used to publish, offer, auction, or sell ownership of, or access to, digital assets, such as, digital art, digital texts, digital video games, digital music, and digital goods or locations in virtual worlds, among other digital items. NFTs can have intrinsic value much like cryptocurrency and they can be used to remove intermediaries, simplify transactions, and create new markets, among other things.

[0053] Once a user has modified a cloud-based client application via an instantiated application engine module, the monetization module **149** can provide appropriate tools and functionality to mint an NFT based on the modified cloud-based client application. For example, the monetization module **149** can provide the user with a selection of blockchain technologies that can be used for the NFT (e.g., Ethereum, Solana, etc.). The monetization module **149** can also provide the user with appropriate tools to create a digital wallet and a selection of marketplaces for the NFT. In some implementations of the present invention, the server system **102** (e.g., via the monetization module **149**) can provide a suitable infrastructure for supporting an NFT marketplace ecosystem. In an alternative embodiment, the monetization module **149** can communicate with one or more appropriate third-party NFT marketplaces (e.g., OpenSea, Nifty Gateway, Mintable, etc.). In such an alternative embodiment, if the user selects a third-party NFT marketplace, the monetization module **149** can upload the digital file of the modified cloud-based client application to the third-party NFT marketplace. Once uploaded, the monetization module **149** can provide selections to the user (e.g., via the modification module **160** of browser module **152** of client device **150**, as discussed in more detail below) for choosing how to monetize the NFT. For example, the NFT can be sold at a fixed price, through a timed auction, or an unlimited auction, although other choices are possible (e.g., the user could decide to give away the NFT or to sell it for a nominal price). The user can also specify royalties if the NFT is resold on the secondary market. Once the user has selected the various characteristics of the NFT, the monetization module **149** can instruct the third-party marketplace to create the NFT, which can then be sold on the third-party NFT marketplace. Alternatively, the monetization module **149** can perform one or more functions of the third-party NFT marketplace to generate the NFT prior to uploading it to the third-party NFT marketplace.

[0054] In an embodiment, the monetization module **149** can also provide tools and functionality to support the purchase of NFTs of modified cloud-based client applications by users of the cloud-based client application platform **100** or **200**. For example, the monetization module **149** can provide tools for opening an NFT marketplace account

(either with a NFT marketplace managed by the server system **102** or a third-party NFT marketplace). The monetization module **149** can also provide tools for creating a blockchain wallet and funding the wallet with the correct cryptocurrency or token required to participate in the NFT marketplace's activity, which can allow the user to purchase NFTs of modified cloud-based client applications via the cloud-based client application platform **100** or **200**. Thus, some implementations of the present invention can combine cloud-based "mods" of client applications with NFTs to address problems of ownership, authenticity, provenance, and attribution. Additionally, the present invention can ensure that programmatic smart contracts govern the distribution and monetization of cloud-based client application NFTs (e.g., payments split between developer, publishers/platform, and creators). The "modded" cloud-based client applications can be distributed independently by creators, monetized as NFTs as the creators see fit, and be purchased/sold in a suitable NFT marketplace.

[0055] In some implementations of the present invention, the cloud-based client application platform **100** or **200** can include one or more client devices **150** in communication with the server system **102** (e.g., via the Internet or other suitable network). According to embodiments, the client device **150** can be any suitable type of device that is capable of running an appropriate browser or other suitable application through which a user can interact (e.g., via touchscreen and/or pointing device, such as a mouse, stylus, or the like), such as, for example, a smartphone, a tablet, a laptop, a desktop computer, a game console, a smart television, a smart watch, or any other appropriate device that is capable of supporting such user input and interaction through the browser or other suitable application. In embodiments, the present invention can support interaction with any appropriate type of cloud-based client application on any suitable number of client devices **150** each running any appropriate type of operating system and independent of the underlying hardware and device characteristics of the client device **150**.

[0056] The cloud-based client application can be any appropriate client application that is configured to execute natively on any suitable type of client device **148**, but can be remotely executed and rendered in the cloud in an application engine module of the plurality of application engine modules **106** or **206** of the server system **102** in accordance with embodiments of the present invention. The cloud-based client application, such as, for example, a cloud-based mobile game or other web-based or suitable cloud-based client application, can be remotely provided as an end-user client application with which the users can interact via the server system **102**. The cloud-based client application can relate to and/or provide a wide variety of functions and information, including, for example, entertainment (e.g., a game, music, videos, etc.), business (e.g., word processing, accounting, spreadsheets, etc.), news, weather, finance, sports, etc. In certain embodiments, the cloud-based client application can provide a digital game, such as a mobile digital game, a PC game, or console game. The digital game can be or include, for example, a sports game, an adventure game, a virtual playing card game, a virtual board game, a puzzle game, a racing game, or any other appropriate type of digital game. In an embodiment, the digital game can be an asynchronous competitive skill-based game, in which players can compete each against other in the digital game, but do not have to play the digital game at the same time. In an alternative embodiment, the digital game can be a synchronous competitive skill-based game, in which players can play the digital game at the same time and can compete against each other in the digital game in real-time. Other suitable cloud-based digital games are possible, whether mobile-based, PC-based, console-based, or otherwise. For cloud-based client applications in which two or more users can compete against or otherwise engage with each other, each user can interact with the cloud-based client application in a respective reserved and assigned application engine module and compete, play against, or otherwise engage each other using the respective application engine modules and the backend architecture supported by the cloud-based client application (e.g., game servers and the like in the context of digital games). Additionally or alternatively, one or more users interacting with the cloud-based client application in a respective reserved and assigned application engine module can compete, play against, or otherwise engage or interact with users who are interacting with the client application installed directly and executing natively on their client device (i.e., without the use of application engine modules). Consequently, embodiments of the present invention can allow users across a wide variety of different computing platforms to engage with each other independent of the underlying client device and operating platform used by each or any user for their client device **150**.

[0057] The client device **150** can include a browser module **152**. The browser module **152** can support any suitable type of browser application, such as, for example, Google Chrome, Mozilla Firefox, Microsoft Edge, Apple Safari, DuckDuckGo, Opera, Brave, Vivaldi, Firefox Focus, a proprietary browser, or the like. Additionally or alternatively, the browser module **152** can be a suitable application capable of supporting the local rendering of the video/audio information streamed from the cloud-based client applications to the client device **150** (e.g., via OpenGL or the like). In an embodiment, the browser module **152** can include a display module **154**, an interaction capture module **156**, a session information module **158**, and a modification module **160** in communication with each other. The display module **154** can support the display of video and/or audio information streamed from the cloud-based client application running in the corresponding application engine module executing on the server system **102**. In an embodiment, the video/audio output of the cloud-based client application running in an application engine module can be streamed in real-time via the Internet (or other suitable network) to the browser module **152** on the client device **150** and displayed to the user via the display module **154**. The display module **154** can receive the streamed video/audio output from the application engine module via the video/audio communication module **132** and render it appropriately on the display screen of the client device **150**. The user can interact with the video/audio stream of the cloud-based client application in real-time. In some implementations of the present invention, the user's inputs can be collected by the interaction capture module **156** and communicated (over the Internet or other suitable network) to the application engine module in the plurality of application engine modules **106** or **206** that has been assigned to the user's active session via signaling communication module **162** (e.g., a suitable data and signaling channel, such as a gRPC gateway or the like). The user's inputs can then be mapped to the video/audio stream by the state module of the assigned application engine

module to allow the user to (remotely) control and interact with the cloud-based client application in real-time as if the cloud-based client application was running natively on the client device **150**.

[0058] In an embodiment, the display module **154** can also provide a display of any or all cloud-based client applications that are available for use by the user (e.g., as URLs or other links or hyperlinks in the browser application, as linked icons on a desktop in the user interface of the client device **150**, etc.). The interaction capture module **156** can capture the user's selection of a desired cloud-based client application (e.g., by the user clicking on or otherwise selecting one of the displayed URLs, links, hyperlinks, or icons). Once the user selects a desired cloud-based client application, the session information module **158** can initiate an active session (e.g., via a REST call or the like) by communicating the request to the session management controller **140**. The session management controller **140** can then begin the process of reserving an appropriate application engine module for the active session, as described above. The session information module **158** can poll the session management controller **140** periodically (e.g., via REST or the like) until an application engine module has been reserved and assigned for an active session. Alternatively, the session management controller **140** can transmit a suitable message or notification to the session information module **158** when an application engine module has been reserved and assigned for the active session. Once the application engine module has been reserved and assigned to the user's active session, the display module **154** can connect to the reserved application engine module through the media (video/audio) channel (e.g., via video/audio communication module **132**) and begin receiving the streamed video/audio media data from the cloud-based client application. As the user interacts with the video/audio media data being streamed from the cloud-based client application executing within the reserved application engine module, the interaction capture module **156** can provide the captured user input to the reserved application engine module via the signaling communication module **162** (e.g., webRTC signaling, via OpenGL, or other suitable technology) to allow the user to remotely engage and interact with the cloud-based client application.

[0059] In an embodiment, the modification module **160** can be in communication with the software tools module **148** on the server system **102**. The modification module **160** can provide the user with a suitable interface on the client device **150** through which to access the software tools provided by the software tools module **148**. Using the modification module **160**, the user can "mod" cloud-based client applications to augment content creation, add new or change existing functionality, create, or otherwise evolve the cloud-based client applications as desired by the user. For example, the modification module **160** can provide a suitable interface via the display module **154** to display a list of the software tools available from the software tools module **148** for remotely modifying the cloud-based client application via the client device **150**, as well as access to the cloud-based client application in the application engine module. The user can select one or more tools as necessary to modify the cloud-based client application in any desired manner, change tools, and continue to modify the cloud-based client application as desired. Any and all modifications can be passed from the modification module **160** to the software tools module **148** so that the cloud-based client application can be correspondingly modified in the given application engine module. Once the modifications to the cloud-based client application have been completed, the user can save the modified cloud-based client application via the modification module **160**, which can communicate such information to the software tools module **148**. The software tools modules **148** can cause the data stored in or otherwise used by the application engine module (e.g., any or all data associated with the modified cloud-based client application, including the modified cloud-based client application itself) during the active session to be stored in the databases **146** so that the modified cloud-based client application can be retrieved and used in a subsequent active session (either as an update to the existing application engine module or as a separate and new application engine module). Such "modded" cloud-based client applications can then be used by other users, allowing distribution of the "modded" cloud-based client applications to a wide variety of users of the cloud-based client application platform **100** or **200**. Thus, embodiments of the present invention can allow any user to create and/or modify cloud-based client applications by remotely accessing software tools through a suitable browser on the client device **150** without requiring the user to separately install each tool on the client device **150** itself. Since users do not install the software tools or the client applications (original or modified) locally, embodiments of the present invention can reduce or eliminate storage requirements for such tools and client applications on the client devices **150**, reduce network bandwidth requirements (since each tool package or client application does not need to be downloaded/uploaded in its entirety to/from each client device **150**), and the like.

[0060] In some implementations of the present invention, the modification module **160** can also provide the user with a suitable interface on the client device **150** through which to access and use the monetization module **149** via the software tools module **148**. Using the modification module **160**, the user can mint, buy, and sell NFTs to monetize modified cloud-based client applications in an NFT ecosystem via the client device **150** in conjunction with the monetization module **149**. For example, the modification module **160** can provide a suitable interface via the display module **154** to display appropriate tools and functionality available via the monetization module **149** to mint an NFT based on the modified cloud-based client application. For example, the modification module **160** can display to the user a selection of blockchain technologies available from and accessible by the monetization module **149** that can be used for the NFT. The modification module **160** can also display to the user appropriate tools from the monetization module **149** to create a digital wallet and a selection of marketplaces for the NFT. For purposes of discussion and not limitation, if the user selects a third-party NFT marketplace, the modification module **160** can display functionality available via the monetization module **149** to allow the user to upload the digital file of the modified cloud-based client application to the third-party NFT marketplace. Once uploaded, the modification module **160** can display selections to the user from the monetization module **149** for choosing how to monetize the NFT. Once the user has selected the various characteristics of the NFT, the modification module **160** can cause the monetization module **149** to instruct the third-party marketplace to create the NFT,

which can then be sold on the third-party NFT marketplace. The modification module **160** can also display tools and provide functionality from the monetization module **149** to support the purchase of NFTs of modified cloud-based client applications by users of the cloud-based client application platform **100** or **200** via the client devices **150**. For example, the modification module **160** can display suitable tools and functionality to the user available from the monetization module **149** for opening an NFT marketplace account. The modification module **160** can also display appropriate tools and functionality to the user available from the monetization module **149** for creating a blockchain wallet and funding the wallet with the correct cryptocurrency or token required to participate in the activity of the NFT marketplace, which can allow the user to purchase NFTs of modified cloud-based client applications via the cloud-based client application platform **100** or **200**.

[0061] In some implementations of the present invention, the cloud-based client application can be paused during an active session. For example, the user can pause the cloud-based client application at any particular or desired position, point or time during the active session with the cloud-based client application (e.g., by pressing a suitable "pause" button displayed to the user in or with the browser application). Additionally or alternatively, the cloud-based client application can be paused if the connection between the browser module **152** and the server system **102** is interrupted or lost during the active session. Additionally or alternatively, the cloud-based client application can be paused if the user leaves the active session (e.g., by pressing a browser "back" button or closing the browser application). In some implementations of the present invention, one or more suitable cookies can be stored on the client device **150** (e.g., in or associated with the browser module **152**), which the browser module **152** can check to determine if there are any active sessions for the user. Such information can be communicated to the session management controller **140** via the session information module **158**. The user's interaction with the cloud-based client application can resume at the point paused once the connection has been reestablished or the user returns to the active session (e.g., by pressing a suitable "play" button displayed to the user in or with the browser application, pressing a "forward" button or re-opening the browser application). In an embodiment, the active session can be held open or otherwise maintained for a predetermined period of time (e.g., 1 hour, 2 hours, 1 day, etc.) while the active session is paused or if the user exits or disconnects from the active session. After the predetermined period of time has expired, the active session can be ended and the application engine module can be deregistered. If the user desires to continue the active session before expiration of the predetermined period of time, the user can, for example, press a "play" button, refresh the webpage in the browser application or select the same URL, link, or hyperlink used to initiate the original active session to rejoin the previously assigned and reserved application engine module. In some implementations of the present invention, the active session can be held open or otherwise maintained indefinitely until the user exits or otherwise ends the active session.

[0062] By maintaining the reserved application engine module and the state of the cloud-based client application during such a pause, some implementations of the present invention can allow users to begin an active session on one client device **150**, pause the active session at a particular or desired position, point or time, and then continue the active session on one or more other (different) client devices **150** from the paused position, point or time. For purposes of illustration and not limitation, a user can request to initiate an active session on a first client device **150** (e.g., a smartphone). The user can then pause the active session (e.g., by closing the browser application on the first client device **150**) at a particular or desired position, point or time. The state of the cloud-based client application can be saved in the state module of the application engine module assigned and registered for that active session. The user can then switch to a second client device **150** (e.g., a desktop computer) and select the same URL, link, or hyperlink (through the display module **154** in the browser module **152** running on the second client device **150**) that was used to initiate the original active session on the first client device **150**. Such action can cause the active session to resume in the browser module **150** of the second client device **148**. Thus, the user can be presented with the same digital page of the active session of the cloud-based client application paused at the same position, point or time from the first client device **148**. The user can seamlessly continue to interact with the same cloud-based client application via the second client device **150** (e.g., by pressing an appropriate "play" button in the browser application) from the paused position, point or time in the same or similar manner as if the user had continued interacting with the cloud-based client application on the first client device **150**. To support such platform switching, the session management controller **140** can maintain a user ID or other appropriate identifier that uniquely identifies the user in addition to the module ID. Using both the user ID and the module ID in combination with the cloud-based client application selection from the user (received from the second client device **150**) and the saved state of the cloud-based client application, the session management controller **140** can identify the correct (paused) application engine module to re-assign to the second client device **150** to allow the user to continue the active session on the second client device **150** from the same position, point or time paused on the first client device **150**. In an alternative embodiment, if the user selects the same cloud-based client application from the same URL, link, or hyperlink on the second client device **150**, the user can be presented with a choice of either continuing the same active session on the cloud-based client application from the same paused position, point or time or starting a new active session with the cloud-based client application on the second client device **150**.

[0063] In the event of an outage or crash of an application engine module during an active session, the session management controller **140** can reserve and allocate a new application engine module to the user. The cloud-based client application running in the application engine module can be populated with information from the previous (pre-outage/pre-crash) active session (e.g., using the cloud-based client application state and other session information retrieved from the databases **146**) to mirror the state of the cloud-based client application at the time just before the outage or crash so that it appears to the user that the active session was simply paused. The user can then return to and reengage with the active session in the manner previously described.

[0064] In some implementations of the present invention, the session management controller **140** (e.g., via the session

manager module **142**) can operate in conjunction with the browser module **152** (e.g., via one or both of the display module **154** or the interaction capture module **156**) to measure and monitor network latency information or other suitable characteristics of network communications between the client device **150** and server system **102** during an active session and to communicate such latency information to the session management controller **140** (e.g., via the session information module **158**) The network latency information can be used to temporarily pause the active session if network latency is above a threshold that is predetermined or dynamic (e.g., based on network latency trends, usage trends, etc.) and then resume the active session once the network latency falls below the threshold. Additionally or alternatively, the network latency information can be used by the server system **102** (e.g., via the video/audio communication module **132**) for network connection quality management. For example, the video/audio communication module **132** can use the network latency information to determine the network connection quality to the client device **150**. If the network connection quality is below a threshold that is predetermined or dynamic (e.g., based on network latency trends, usage trends, etc.), the video/audio communication module **132** can modify the quality and/or characteristics of the video/audio media data that is being streamed to the client device **150**. For example, if the network connection quality is below the threshold, the video/audio communication module **132** can lower the frame rate, lower the resolution, and the like of the streamed video/audio media data so as to maintain smooth display and interaction for the cloud-based client application on the client device **150** during the active session (e.g., to maintain smooth gameplay on the client device **150** in the context of cloud-based mobile games), so that the active session does not need to be paused or otherwise interrupted. The video/audio communication module **132** can update the quality and/or characteristics of the stream of video/audio media data periodically or continuously throughout the active session (e.g., lowering and/or raising the frame rate, resolution, etc. of the video/audio media data stream as the network connection quality fluctuates over the course of the active session). Consequently, the server system **102** can use the network latency information to determine, vary, and control the video/audio streaming quality to the client devices **150** to maintain a smooth display and interaction for the cloud-based client application for the user.

[0065] Additionally or alternatively, the network latency information can be used by the session management controller **140** to determine and select appropriate application engine modules for users. For example, in some implementations of the present invention, the server system **102** can include a plurality of application engine module controllers **104**, with each application engine module controller **104** managing respective pluralities of application engine modules **106** or **206** to form one or more application engine module clusters, as illustrated and discussed with respect to FIGS. **1** and **2**. The application engine module clusters can be geographically distributed such that each of the respective pluralities of application engine modules **106** or **206** can be located in different geographical locations. When a user requests an active session with a cloud-based client application, the network latency information can be used by the session management controller **140** to cause an application engine module to be reserved and assigned that is closest geographically or otherwise has the smallest network latency to the client device **150** of the user. By measuring and monitoring the network latency information for all client devices **150**, the server system **102** can create a heat map or the like of the network latencies throughout the cloud-based client application platform **100** or **200** that can be used to reserve and assign an optimal application engine module (e.g., optimal from the standpoint of network latency or network connection quality) to each user so as to maintain smooth display and interaction for the cloud-based client application on the client devices **150** during the respective active sessions.

[0066] According to embodiments, because the cloud-based client applications can be run remotely in server system **102** and can be displayed on any suitable client device **150**, the present invention can also allow cloud-based client applications to be integrated into third-party platforms, including the third party websites and applications, by embedding or otherwise providing a URL or link or hyperlink to one or more cloud-based client applications into the third-party platform. Such integration can allow those third parties to more easily offer different features and functionality via the integrated cloud-based client application that complement and enhance the interaction and engagement with their third-party platforms. For purposes of illustration and not limitation, a retailer can incorporate any suitable cloud-based client application within their consumer touchpoints by embedding or otherwise providing a URL or other link or hyperlink to the cloud-based client application. For example, a retailer can incorporate a cloud-based mobile game into the retailer's website or application in accordance with embodiments of the present invention. A user can interact with the cloud-based mobile game in the retailer's website or application as if the cloud-based mobile game was executing on the user's client device, in the manner described herein. If the user wins the integrated cloud-based mobile game, the user can be awarded prizes in the retailer's own currency (e.g., loyalty rewards, coupons, discounts on products/services, free merchandise, etc.). The prize information can be communicated to the third-party platform from the integrated cloud-based mobile game (e.g., via the server system **102**) and maintained in the retailer's website or application for use by the user. For example, a fast-food restaurant can integrate a cloud-based mobile game into their website or application in accordance with embodiments of the present invention. The user can select and engage with the integrated cloud-based mobile game via the website or application by, for example, selecting an appropriate embedded or otherwise provided URL or other link or hyperlink to the integrated cloud-based mobile game and then (remotely) interacting with the integrated cloud-based mobile game. If the user wins the integrated cloud-based mobile game, the user can be awarded a free side order of a food item (or other suitable prize or offering appropriate to the retailer) that can be redeemed at the fast-food restaurant. If the user loses, the user can be allowed to compete again, offered a lesser prize, offered another chance to play, etc. The present invention can support the integration of any suitable type of cloud-based client application into any appropriate third-party platform. By allowing users to interact with a cloud-based client application integrated into and through the third party's website or application in accordance with the present invention, the third party can increase user engagement metrics with their platform without requir-

ing users to download and install the cloud-based client application to their client devices **150**.

[0067] By integrating one or more cloud-based client applications into a third party's platform, distinct experiences can be tailored for users on the third-party platform that may not available to the general population who access a non-integrated version of the cloud-based client application or access it from a different website or application. For example, the integrated cloud-based client application on the third-party platform can provide features, functionality, interface modifications, and the like that are available to users of the integrated cloud-based client application. For instance, if the integrated cloud-based client application is a cloud-based mobile game, users interacting with the integrated cloud-based mobile game can have access to special or promotional tournaments that are not available to general users of the cloud-based mobile game. Additionally or alternatively, users interacting with the integrated cloud-based client application can be presented with special limited time offers (LTOs), advertisements, or other promotions that are not available to general users of the cloud-based client application. Additionally or alternatively, users interacting with the integrated cloud-based client application can be presented with modified or enhanced user interfaces (e.g., different screens, customizations, third-party branding information, etc.) that are not available to general users of the cloud-based client application. Other modifications to the user experience of an integrated cloud-based client application (including modifications to the graphical user interface inside and/or outside of the integrated cloud-based client application) to support the third-party platform are possible.

[0068] In some implementations of the present invention, the data collected from users interacting with the integrated cloud-based client application can also be used to additionally or alternatively tailor the user experience of the integrated cloud-based client application. In an embodiment, the features, functionality, and/or capabilities of the integrated cloud-based client application on the third-party platform can be targeted to individual users based on the third-party platform user data, data from or associated with the user's client device **150**, and the like. For example, the server system **102** can identify the geographical location of each user, such as via the GPS of the client device **150** (e.g., passed to the session management controller **140** via the session information module **158**), from account profile information of the user (e.g., retrieved from the databases **146**), or the like. A user who accesses the integrated cloud-based client application from one geographical location (e.g., a first state or region) can be presented with a user experience that is different from that of a user who accesses the integrated cloud-based client application from a different geographical location (e.g., a second state or region). The different user experience can include, for example, different user interfaces, different tournaments (e.g., in the context of cloud-based mobile games), different features and functionality, and the like. For instance, if the integrated cloud-based client application is a cloud-based mobile game, the user data can be used to match players for the integrated cloud-based mobile game. For example, for a competitive, skill-based integrated cloud-based mobile game, the user information can be used to match users with other users who are interacting with the integrated cloud-based mobile game rather than with users from the general population of the (non-integrated) cloud-based mobile game. Such player matching can assist in ensuring fairness in the competitions and/or to enforce anti-fraud/anti-cheat measures, as a user of the integrated cloud-based mobile game may have access to features and functionality that are not available to the general population of users of the (non-integrated) cloud-based mobile game.

[0069] Some implementations of the present invention can be used to integrate cloud-based client applications into online or digital advertisements. For purposes of illustration and not limitation, a cloud-based mobile game can be integrated into a digital advertisement to create an interactive ad unit, such as a playable ad. In some implementations of the present invention, the integrated cloud-based mobile game can be accessed from the playable ad by having the user click on a URL, link, or hyperlink embedded or otherwise provided in the playable ad. Alternatively, the cloud-based mobile game can start automatically within the playable ad upon surfacing, presentation, or display of the playable ad to the user. A user can interact with the integrated cloud-based mobile game in and through the playable ad, for example, to win prizes, promotions, and other like rewards as part of the digital ad campaign. In an embodiment, such rewards could be used for the cloud-based mobile game itself (e.g., virtual currency or rewards) to incentivize users to interact with the playable ad. For example, users can use such rewards for making in-app purchases or for other types of redemptions or purchases within the integrated cloud-based mobile game itself while interacting with the playable ad. In an alternative embodiment, such rewards can be used for or with another application or website associated with the digital ad campaign to drive user engagement. However, any suitable cloud-based client application can be integrated into a digital advertisement to create an interactive ad unit. For example, an interactive ad unit or other digital advertisement for a cloud-based client application product can include a trial version (or even a fully-enabled, but time-limited version) of the cloud-based client application product integrated into the interactive ad unit or other digital ad. Consequently, users can engage with and evaluate the cloud-based client application product (e.g., within and through the interactive ad unit or other digital ad) without having to wait to download and install it or otherwise purchase it before evaluation.

[0070] FIG. **3** is a block diagram illustrating an example cloud-based client application platform **300** with a plurality of application engine module pools and a plurality of application instance pools for remotely modifying cloud-based client applications, in accordance with embodiments of the disclosure. As discussed previously, each of the application engine modules in the plurality of application engine modules **206** can be configured to emulate a particular type and configuration of client device for executing a cloud-based client application. Additionally or alternatively, the server system **102** can support cloud-based client application instances (also referred to as "application instances") that can be designed and configured to execute natively in the cloud on the server system **102**. Thus, an application instance can be an instance of a cloud-based client application that can be configured to run natively on the server system **102** without the use of an application engine module from the plurality of application engine modules **206** (or **106**). Accordingly, the server system **102** can include a module controller **304**, which can operate in a manner similar to that discussed and described with respect to the

application engine module controller **104** of FIGS. **1** and **2**. The module controller **304** can include a plurality of application engine modules **206** to support and emulate any appropriate type of client device to execute any suitable type of cloud-based client application (as discussed and described with respect to FIGS. **1** and **2**). The module controller **304** can also include a plurality of application instance modules **310** to support and execute any appropriate type of application instance that can run natively on the server system **102**. The module controller **304** can allow registration/deregistration of application instance modules, atomic reservation of application instance modules, querying of application instance modules, and the like in a manner similar to that described for the plurality of application engine modules **206** (and **106**). The plurality of application instance modules **310** can include a first application instance module **312**, . . . , and a Mth application instance module **320**, where M can be any suitable natural number and can vary over time. Each application instance module **312**, . . . **320** can include a video/audio module for managing video and/or audio media data generated by the application instance and transmitting such video and/or audio media data to a client device for display to a user. Each application instance module **312**, . . . **320** can also include a state module for receiving and processing input (e.g., mapping input to the video/audio stream, etc.) from the client device **150** generated while the user (remotely) interacts with the application instance. Each state module can also maintain the state of the application instance. For example, the first application instance module **312** can include a first video/audio module **314** in communication with a first state module **316**. The Mth application instance module **320** can include a Mth video/audio module **322** in communication with a Mth state module **324**. The module controller **304** can maintain any suitable number of application instance modules and register (i.e., instantiate, allocate, or otherwise create) or deregister (i.e., delete or deallocate) application instance modules as necessary. In an embodiment, the server system **102** can include a plurality of module controllers **304**, with each module controller **304** managing respective pluralities of application engine modules **206** to form application engine module clusters and respective pluralities of application instance modules **310** to form application instance module clusters. The server system **102** and the module controller **304** can support any suitable number and combination of application engine modules and application instance modules.

[0071] As illustrated in FIG. **3**, the server system **102** can include a reservation controller **328** in communication with the module controller **304**, which can operate in a manner similar to that discussed and described with respect to the application engine reservation controller **134** of FIGS. **1** and **2**. In embodiments, the reservation controller **134** can also control various aspects of the plurality of application instance modules **310**, including, for example, orchestrating application instance module scaling, controlling application instance module access, managing application instance module initialization for users, and the like. The reservation controller **328** can include an inventory and reservations module **330**, which can operate in a manner similar to that discussed and described with respect to the application engine inventory and reservations module **136** of FIGS. **1** and **2**. The inventory and reservations module **330** can also track and manage the inventory of all application instance modules in the plurality of application instance modules **310**. The inventory and reservations module **330** can grant application instance module reservations for use in active sessions. In some implementations of the present invention, when a user requests initiation of an active session with an application instance module, the inventory and reservations module **330** can reserve an appropriate application instance module, populate or cause to populate the application instance module with the appropriate user and/or application data (e.g., user profile information, user account information, application instance state data, etc. to support user interaction and engagement with the application instance), and bind the reserved application instance module to that active session. If no application instance module is available for the active session, the inventory and reservations module **330** can instruct the module controller **304** (e.g., via the scaler module **332**, discussed below) to add and register a new application instance module for use in the requested active session.

[0072] In some implementations of the present invention, the registration process can be performed asynchronously. For example, the inventory and reservations module **330** can wait for the new application instance module to be added and reserved by polling the module controller **304** periodically or receiving an appropriate message from the module controller **304** once the task has been completed. Once the application instance module is registered, it can be reserved for and bound to the active session. In some implementations of the present invention, the application instance module that has been reserved for the active session can be accessible only to the user that requested it, and only for the lifespan of the active session. In other words, an application instance module that has been reserved for a particular active session may not be used or accessed by another user during that active session. In some implementations of the present invention, the data stored in or otherwise used by the reserved application instance module during the active session can be ephemeral. For example, the application instance module can be deregistered at the end of an active session. Accordingly, any and all data stored or otherwise used by the application instance module during that active session can be deleted, for example, no later than the deregistration of the application instance module. In an alternative embodiment, the data stored in or otherwise used by the reserved application instance module during the active session can be stored (e.g., in databases **146**), retrieved, and used in a subsequent active session by the user. Once the application instance module has been deregistered, it can be reused for another active session or its computing resources can be reallocated to a different application instance module.

[0073] In embodiments, the reservation controller **328** can include a scaler module **332** in communication with the inventory and reservations module **330**, which can operate in a manner similar to that discussed and described with respect to the application engine scaler module **138** of FIGS. **1** and **2**. The scaler module **332** can also respond to inventory and reservation changes by adding and instantiating new application instance modules to the plurality of application instance modules **310**. The scaler module **332** can proactively scale the number of application instance modules in the plurality of application instance modules **310** based on, for example, usage trends or other suitable metrics to maintain a sufficiently-sized pool of reservable application instance modules. For example, the scaler module **332**

can store a predetermined threshold (e.g., provided or generated based on usage trends or other suitable metrics) for the size of the pool of the plurality of application instance modules **310**. If the number of reservable application instance modules decreases below the predetermined threshold, the scaler module **332** can add new application instance modules to the plurality of application instance modules **310** to increase the number of reservable application instance modules above the predetermined threshold. If the increase in the number of reservable application instance modules becomes too great (e.g., above a second predetermined threshold), the scaler module **332** can either do nothing or can delete or otherwise deallocate unneeded extra application instance modules from the plurality of application instance modules **310**. Additionally or alternatively, suitable machine learning/artificial intelligence techniques can be used by or with the scaler module **332** to appropriately and dynamically scale the number of reservable application instance modules in the plurality of application instance modules **310**. For example, a machine learning model can be trained based on usage data from users (e.g., usage trends, peak usage, off-hour usage, etc.). The machine learning model can then be used to dynamically add and delete/deallocate application instance modules in the plurality of application instance modules **310**. The machine learning model can be updated or otherwise adapted as the usage data evolves over time.

[0074] In some implementations of the present invention, each or any of the plurality of application instance modules **310** can be associated and in communication with a separate application instance module pool that can be maintained by, for example, the inventory and reservations module **330** and/or the scaler module **332**, with each application instance module pool supporting an individual type of application instance module or a set of related types of application instance modules. In such an embodiment, each application instance can have or be associated with its own individual application instance module pool. For example, each pool of the plurality of application instance module pools can be configured to support a single application instance, such that each pool of the plurality of application instance module pools can be allocated on a per application instance basis. For purposes of illustration and not limitation, a klondike solitaire game designed to run natively on the server system **102** can be supported by the first application instance module **312**, which can be associated with a first application instance module pool **318** of first application instance modules. A bingo game designed to run natively on the server system **102** can be supported by the Mth application instance module **320**, which can be associated with an Mth application instance module pool **326** of Mth application instance modules, and so forth. Any suitable number of application instance module pools can be maintained and supported for any appropriate number of application instance modules by, for example, the inventory and reservations module **330** and/or the scaler module **332**, depending on, for instance, the number of application instances supported by the system or any appropriate subset thereof.

[0075] In some implementations of the present invention, each or any of the application instance modules in each pool of the plurality of application instance module pools can be "pre-warmed." For example, the application instance modules in each pool of the plurality of application instance module pools can be pre-allocated and pre-instantiated within the application instance module pool and then put into standby or sleep mode by, for instance, the inventory and reservations module **330** and/or the scaler module **332**. By spinning up the application instance modules in each application instance module pool in the background and maintaining those spun-up application instance modules in a standby mode, an application instance module can be immediately (or nearly immediately) reserved and assigned to a user upon request of an active session with little or no concomitant delay, since awakening an application instance module that is in standby mode can take substantially less time than instantiating the application instance module from scratch. Once an application instance module in an application instance module pool is reserved and assigned, the reserved and assigned application instance module can be immediately (or nearly immediately) replaced by another "pre-warmed" application instance module from the corresponding application instance module pool so that it can be ready for immediate (or nearly immediate) reservation by and assignment to the next user requesting an active session. Such a "pre-warmed" replacement process can continue as each application instance module in an application instance module pool is reserved and assigned. In such a manner, application instance modules from each pool of the plurality of application instance module pools can be allocated to users in real-time. The computing resources allocated to a reserved application instance module for an active session can be deleted or otherwise deallocated immediately (or nearly immediately) once the active session ends and the reserved application instance module is no longer needed for that active session. In some implementations of the present invention, such freed up computing resources can be reallocated by, for example, the inventory and reservations module **330** and/or the scaler module **332** to add and pre-warm another application instance module in the associated application instance module pool or a different application instance module pool, rather than maintaining computing resources for a reserved application instance module that is no longer needed for an inactive session.

[0076] The size of each or any of the plurality of application instance module pools can be fixed or dynamic. For example, the size of an application instance module pool can be fixed such that a predetermined application instance module pool size number of pre-warmed application instance modules can be maintained in the pool. As pre-warmed application instance modules in an application instance module pool are allocated, the inventory and reservations module **330** and/or the scaler module **332** can replace the allocated application instance modules with additional pre-warmed application instance modules to maintain the size of the application instance module pool at or around the predetermined application instance module pool size number. The predetermined application instance module pool size number can be based on, for example, usage trends or other historical usage information and the like for the particular application instance module. Accordingly, highly used application instance modules can have a greater predetermined application instance module pool size number while less used application instance modules can have a smaller predetermined application instance module pool size number.

[0077] Additionally or alternatively, the size of an application instance module pool can be dynamic. In some implementations of the present invention, the inventory and

reservations module **330** and/or the scaler module **332** can monitor in real-time the usage trends of each or any of the plurality of application instance modules **310**. For example, as usage trends increase for a particular application instance module, the scaler module **332** can increase the size of the associated application instance module pool (and, therefore, the number of pre-warmed application instance modules maintained in the application instance module pool) to accommodate the greater usage so that application instance module allocation can continue in real-time without added delay. Conversely, as usage trends decrease for a particular application instance module, the scaler module **332** can decrease the size of the associated application instance module pool (and, therefore, the number of pre-warmed application instance modules maintained in the application instance module pool) to accommodate the lower usage so that computing resources are not unnecessarily overused and can be allocated elsewhere. In such an embodiment, the size of each or any of the application instance module pools can be dynamically altered to accommodate periods of increased use (e.g., spikes in usage) as well as lulls in usage to maintain real-time application instance module allocation.

[0078] In some implementations of the present invention, the inventory and reservations module **330** and/or the scaler module **332** can include a suitable predictive model or use appropriate machine learning/artificial intelligence techniques to determine the size of an application instance module pool of the plurality of application instance module pools. For example, a machine learning model can be trained based on usage trend data for each supported application instance module to modify the application instance module pool size for an application instance module to minimize the amount of time to allocate the application instance module and maintain a reduced allocation time for the real-time application instance module allocation. The machine learning model can then be used to dynamically and appropriately modify the size of the application instance module pool associated with the application instance module based on the usage exhibited at a particular time for that application instance module. The machine learning model can be updated or otherwise adapted as the usage trends for the application instance module change and evolve over time, as monitored by, for example, the inventory and reservations module **330** and/or the scaler module **332**.

[0079] In some implementations of the present invention, the size of each application instance module pool in a subset of the plurality of application instance module pools can be fixed while the size of other application instance module pools in the plurality of application instance module pools can be dynamic. For example, application instance modules that are highly used can be associated with application instance module pools whose sizes are dynamic, while application instance modules that are less used can be associated with application instance module pools whose sizes are fixed. Additionally or alternatively, the size of a particular application instance module pool can be fixed at certain times and dynamic at other times. For example, for an application instance module that generally experiences lower usage, the size of the corresponding application instance module pool can be fixed. However, during (e.g., intermittent) periods of higher usage (e.g., usage spikes), the size of the corresponding application instance module pool can be dynamic. The inventory and reservations module **330** and/or the scaler module **332** can switch the size of the application instance module pool back to fixed once the period of higher usage ends. In this way, the inventory and reservations module **330** and/or the scaler module **332** can adapt to changing usage conditions as necessary to maintain a reduced allocation time for the real-time allocation of application instance modules.

[0080] In some implementations of the present invention, the one or more application instance module pools can be maintained in and with the plurality of application instance modules **310**, as illustrated in FIG. **3**. However, in some implementations of the present invention, each or any of the one or more application instance module pools can be maintained in or with the module controller **304** outside of the plurality of application instance modules **310** but within the server system **102**. Additionally or alternatively, each or any of the one or more application instance module pools can be maintained in or with the server system **102**, but outside of the module controller **304**, such as, for example, in databases **146**. Additionally or alternatively, each or any of the one or more application instance module pools can be maintained in or with a suitable cloud or server storage system that is remote from and in communication with the server system **102**.

[0081] In some implementations of the present invention, the reserved and assigned application engine module and/or the reserved and assigned application instance module can each be associated with a different, unique application engine module identification (ID) number or other suitable identifier (e.g., comprised of numeric or alphanumeric characters or the like) that uniquely identifies the reserved and assigned application engine module and/or the reserved and assigned application instance module during the active session. The module ID can be used by, for example, the session management controller **140** to track and manage the reserved and assigned application engine module and/or the reserved and assigned application instance module during the active session to ensure that communications from the client device **150** during the active session are addressed and forwarded to the correct application engine module or application instance module and vice versa, and the like. In an embodiment, when the active session has completed, the session information module **158** can send an appropriate message to the session management controller **140** so that the session management controller **140** can disconnect the active session and deregister the application engine module or the application instance module used during the active session. In some implementations of the present invention, any and all data stored or otherwise used by the registered application engine module or the registered application instance module during that active session can be deleted upon deregistration of the application engine module or the application instance module. In an alternative embodiment, the data stored in or otherwise used by the reserved application engine module or the reserved application instance module during the active session can be stored (e.g., in the databases **146**), retrieved, and used in a subsequent active session by the user. Once the application engine module or the application instance module has been deregistered, it can be reused for another active session or its computing resources can be reallocated to a different application engine module or application instance module.

[0082] FIG. **4** is a block diagram illustrating an example cloud-based client application platform **400** for remotely modifying cloud-based client applications, in accordance

with an embodiment of the disclosure. The cloud-based client application platform **400** provides features and functionality similar to that illustrated and discussed with respect to the cloud-based client application platform **100** of FIG. **1**. However, in some implementations of the present invention, the cloud-based client application platform **400** can use and include a plurality of application engine module pools and provide features and functionality similar to that illustrated and discussed with respect to the cloud-based client application platform **200** of FIG. **2**. Alternatively, in some implementations of the present invention, the cloud-based client application platform **400** can use and include a plurality of application engine module pools and a plurality of application instance pools and provide features and functionality similar to that illustrated and discussed with respect to the cloud-based client application platform **300** of FIG. **3**.

[0083] As illustrated in FIG. **4**, the software tools module can be instantiated within each or any of the plurality of application engine modules **106**. For example, each or any of the application engine modules **108**, **114**, **120**, . . . **126** can include a software tools module that can include or provide any suitable cloud-based software tools that can be used to build, create, augment, modify, generate, and/or distribute a cloud-based client application via the cloud-based client application platform **400**. For example, the first application engine module **108** can include a software tools module **405** in communication with the video/audio module **110** and the state module **112**. The second application engine module **114** can include a software tools module **410** in communication with the video/audio module **116** and the state module **118**. The third application engine module **120** can include a software tools module **415** in communication with the video/audio module **122** and the state module **124**. The Nth application engine module **126** can include a software tools module **420** in communication with the video/audio module **128** and the state module **130**. A software tools module can be instantiated within an application engine module when the application engine module is reserved, registered, and bound to an active session by the application engine reservation controller **134** (e.g., via the application engine inventory and reservations module **136** in communication with the application engine module controller **104**). In an embodiment, each or any of the respective software tools modules can provide a full suite of diverse software tools to support a wide variety of cloud-based client applications. Additionally or alternatively, each or any of the respective software tools modules can provide a subset of software tools for use in remotely modifying or augmenting the cloud-based client applications executing in the application engine modules. Such a subset of tools can be based on, for example, the characteristics of the cloud-based client application executing in the application engine module (e.g., the type of cloud-based client application, the type of operating system in which the cloud-based client application will be run, etc.), the type of modifications being made to the cloud-based client application executing in the application engine module (e.g., image editing software tools if image modification are being made, text editing software tools if text editing is being performed, etc.), or the like.

[0084] In an embodiment, the tools and functionality supported by the respective software tools modules can be available to and accessible by any users via the modification module **160** executing on the client device **150**, as previously discussed with respect to the cloud-based client application platform **100** of FIG. **1**. However, since certain users may simply want to interact and engage with a cloud-based client application and not desire to modify a cloud-based client application, accessibility to the tools and functionality supported by the respective software tools modules can be limited to a subset of users by the cloud-based client application platform **400** or by the user. For example, the ability to access a respective software tools module can be provided as, for instance, a button or other suitable graphical element that can be displayed to the user through the modification module **160** via the display module **154** when the user accesses an instantiated application engine module. Selection of such a button or graphical element can result in the modification module **160** displaying or causing to display to the user (e.g., via the display module **154**) the software tools available from the software tools module in the application engine module for the active session. Alternatively, such access can be hidden from or otherwise not displayed to certain users (e.g., novice or beginning users, users not interested in remotely modifying client applications, etc.). For example, online user accounts associated with each user (e.g., as maintained by the cloud-based client application platform **400**) can provide one or more appropriate account settings that can allow a user to specify whether or not the user desires access to such tools. If the user does not desire such access, no tools or ability to access the tools (e.g., no button or other graphical element) would be displayed to the user in the active session. Additionally or alternatively, predetermined rules can be established (e.g., by an administrator of the cloud-based client application platform **400**) to limit access to such tools to certain users (e.g., experienced users, etc.). Other ways of restricting or limiting access to respective software tools modules are possible.

[0085] In an embodiment, the application engine module controller **104** can include a monetization module **425** in communication with the plurality of application engine modules **106**. In such an embodiment, the monetization module **425** can be separate from the respective software tools modules executing in the respective application engine modules. In an alternative embodiment, the monetization module **425** can be included in or with each or any of the respective software tools modules in the respective application engine modules. The monetization module **425** can provide features and functionality similar to that discussed with respect to the monetization module **149** of FIG. **1**. Thus, the monetization module **425** can provide tools and support for monetizing modified cloud-based client applications to allow users to mint, buy, and sell NFTs to monetize modified cloud-based client applications in an NFT ecosystem as discussed previously with respect to monetization module **149** of FIG. **1**.

[0086] FIG. **5** is a block diagram illustrating an example cloud-based client application platform **500** for remotely modifying and monetizing cloud-based client applications, in accordance with an embodiment of the disclosure. A server system **520** can provide functionality for remotely displaying and modifying cloud-based client applications on client devices of users. The server system **520** can include software components and databases that can be deployed at one or more data centers **518** in, for example, one or more geographic locations. The software components of the server system **520** can include a client application module **522**, a software tools module **524**, and a monetization

module **526**. The software components can include subcomponents that can execute on the same or on different individual data processing apparatus. The server system **520** can include one or more databases **528**. The databases **528** can be cloud-based or reside in one or more physical storage systems. The software components and data will be further described below.

[0087] As illustrated in FIG. **5**, the client application module **522**, the software tools module **524**, and the monetization module **526** can communicate with the databases **528**. The databases **528** can include, for example, user information, client applications and associated data, modified cloud-based client applications and associated data, software tools for remotely modifying cloud-based client applications, monetization information (NFTs, data associated with NFTs, cryptocurrency wallet information, user account information, such as cryptocurrency balances and the like, etc.), digital content for remotely modifying cloud-based client applications (e.g., images, videos, sounds, text, etc.), and the like. The cloud-based client applications can relate to and/or provide a wide variety of functions and information, including, for example, entertainment (e.g., a game, music, videos, etc.), business (e.g., word processing, accounting, spreadsheets, etc.), news, weather, finance, sports, etc. In certain embodiments, the cloud-based client application can provide a cloud-based mobile game. The cloud-based mobile game can be or include, for example, a sports game, an adventure game, a virtual playing card game, a virtual board game, a puzzle game, a racing game, or any other appropriate type of mobile game. In an embodiment, the cloud-based mobile game can be an asynchronous competitive skill-based game, in which players can compete each against other in the cloud-based mobile game, but do not have to play the cloud-based mobile game at the same time. In an alternative embodiment, the cloud-based mobile game can be a synchronous competitive skill-based game, in which players can play the cloud-based mobile game at the same time and can compete against each other in the cloud-based mobile game in real-time.

[0088] In an embodiment, the cloud-based client application or components thereof can be accessed through a network **530** (e.g., the Internet or the like) by users of client devices, such as client device A **502**, client device B **506**, client device C **510**, . . . , client device P **514**, where P can be any suitable natural number. Each of the client devices can be any appropriate type of electronic device that is capable of displaying the output of the cloud-based client application that is executing remotely on the server system **520** and communicating with the server system **520** through the network **530**, such as, for example, a smart phone, a tablet computer, a laptop computer, a desktop or personal computer, a game console, or the like. Other client devices are possible. Each of the client devices **502**, **506**, **510**, . . . , **514** can include a display module for displaying the output of a cloud-based client application that is executing remotely on the server system **520** and for allowing a user to remotely interact with the cloud-based client application. For example, the client device A **502** can include a display module A **504**, the client device B **506** can include a display module B **508**, the client device C **510** can include a display module C **512**, and the client device P **514** can include a display module P **516**. In an embodiment, each display module can support any suitable type of browser application, such as, for example, Google Chrome, Mozilla Firefox, Microsoft Edge, Apple Safari, DuckDuckGo, Opera, Brave, Vivaldi, Firefox Focus, a proprietary browser, or the like through which the user can access the cloud-based client applications remotely executing on the server system **520**. Additionally or alternatively, each display module can be or otherwise comprise a suitable application capable of supporting the rendering of cloud-based client applications locally in the browser application (e.g., via OpenGL or the like). In an alternative embodiment, the databases **528** or any portions thereof can be stored on one or more client devices. Additionally or alternatively, software components for the cloud-based client application platform **500** (e.g., the client application module **522**, the software tools module **524**, and the monetization module **526**) or any portions thereof can reside on or be used to perform operations on one or more client devices.

[0089] The client application module **522** can be configured to execute cloud-based client applications on the server system **520** and display the output of the cloud-based client applications remotely on the client devices of users. For example, the client application module **522** can provide one or more application engine modules in which each cloud-based client application can execute in a manner similar to that discussed with respect to the server system **102** of FIGS. **1**, **2**, **3**, and **4**. The output of the cloud-based client applications can be displayed within the display module of the client device of the user, which can also allow the user to interact with the cloud-based client application as if the cloud-based client application is executing natively on the client device.

[0090] The software tools module **524** can provide cloud-based software tools and other suitable functionality to allow users to remotely create, augment or otherwise modify cloud-based client applications via the client devices for distribution via the cloud-based client application platform **500**. For example, such cloud-based software tools can be used to modify an existing cloud-based client application that executes in the client application module **522** to create an augmented or modified cloud-based client application. In some implementations of the present invention, the cloud-based software tools can allow users to modify non-cloud based client applications into cloud-based client applications for distribution via the cloud-based client application platform **500** (e.g., suitable compilers, cross-compilers, software development tools, etc.). The software tools module **524** can allow "modders" (e.g., client application developers, client application users, etc.) to augment content, add new or change existing functionality, create, or otherwise modify cloud-based client applications via the client devices. For example, the software tools module **524** can include or otherwise provide appropriate software tools for creating and editing digital content (e.g., text, images, video, audio, or any combination thereof) within the cloud-based client applications. Additionally or alternatively, the software tools modules **524** can include or otherwise provide suitable software development tools for generating, editing, and compiling (or cross-compiling) computer code for the cloud-based client applications. Other software tools are possible. The software tools module **524** can include or provide any suitable cloud-based software tools that can be used to build, create, augment, modify, generate, and/or distribute a cloud-based client application via the cloud-based client application platform **500** using the client devices of the users. Using such cloud-based software tools, a wide variety of users can

be provided with the features and functionality to "mod" cloud-based client applications to augment content creation, add new or change existing functionality, create, or otherwise evolve the cloud-based client applications as desired by the users.

[0091] In an embodiment, the software tools module **524** can provide a full suite of diverse software tools to support a wide variety of cloud-based client applications. Additionally or alternatively, the software tools module **524** can provide a subset of software tools for use in remotely modifying or augmenting the cloud-based client applications executing in the client application module **522**. In an embodiment, the software tools module **524** can use suitable machine learning/artificial intelligence techniques to automatically and dynamically select and provide to users the appropriate software tools based upon, for example, the characteristics of the cloud-based client application (e.g., the type of cloud-based client application, the type of operating system in which the cloud-based client application will be run, etc.), the type of modifications being made to the cloud-based client application (e.g., image editing software tools if image modification are being made, text editing software tools if text editing is being performed, etc.), or the like. For example, a machine learning model can be trained based on tool usage data from users (e.g., which software tools were used and how often, which cloud-based client applications were modified with which tools, etc.). The machine learning model can then be used to dynamically add and select suitable software tools for the user to modify a given cloud-based client application. The machine learning model can be updated or otherwise adapted as the software tool usage data evolves over time. For example, the software tools module **524** can maintain a set of core software tools, while additional or alternative software tools can be stored in and automatically and dynamically retrieved from, for example, the databases **528** as needed and as necessary as the user is remotely modifying a cloud-based client application via the client device of the user.

[0092] In an embodiment, once the modifications to the cloud-based client application have been completed, the data stored in or otherwise used by the client application module **522** (e.g., any or all data associated with the modified cloud-based client application, including the modified cloud-based client application itself) can be stored in the databases **528** so that the modified cloud-based client application can be retrieved and used again by the client application module **522** for other users. Consequently, the modified cloud-based client application can be distributed to and accessed by other users of the cloud-based client application platform **500**. In an additional embodiment, the software tools module **524** can provide tools and functionality for remotely modifying non-cloud based client applications (e.g., PC-based applications, game console-based applications, etc.) into cloud-based client applications for distribution via the cloud-based client application platform **500** in the manner described previously. For example, to convert non-cloud based client applications into cloud-based mobile applications, the software tools module **524** can provide appropriate tools to augment application content to accommodate different sized screens of mobile devices, compilers (or cross-compilers) and other software development tools for creating/editing and re-compiling (or cross-compiling) the client applications to run on a (different) target client platform and operating system (e.g., iOS, Android, etc.), and the like. The software tools module **524** can also provide suitable tools for uploading the modified or otherwise converted cloud-based client application to the cloud-based client application platform **500**.

[0093] In an embodiment, each display module on each client device can provide the user with a suitable interface through which to access the software tools provided by the software tools module **524**. Through the display module, the user can "mod" cloud-based client applications to augment content creation, add new or change existing functionality, create, or otherwise evolve the cloud-based client applications as desired by the user. For example, the display module (e.g., display module A **504** of client device A **502**) can display a list of the software tools available from the software tools module **524** for remotely modifying the cloud-based client application via the client device, as well as access to the cloud-based client application in the client application module **522**. The user can select one or more tools as necessary to modify the cloud-based client application in any desired manner, change tools, and continue to modify the cloud-based client application as desired. Any and all modifications can be passed or otherwise transmitted or communicated from the display module on the client device to the software tools module **524** so that the cloud-based client application can be correspondingly modified in the client application module **522**. Once the modifications to the cloud-based client application have been completed, the user can save the modified cloud-based client application via the display module on the client device, which can communicate such information to the software tools module **524**. The software tools modules **524** can cause the data stored in or otherwise used by the client application module **522** (e.g., any or all data associated with the modified cloud-based client application, including the modified cloud-based client application itself) to be stored in databases **528** so that the modified cloud-based client application can be retrieved and used in a subsequent active session. Such "modded" cloud-based client applications can be used by other users, which can allow distribution of the "modded" cloud-based client applications to a wide variety of users of the cloud-based client application platform **500**. Thus, some implementations of the present invention can allow any user to create and/or modify cloud-based client applications by remotely accessing software tools through a suitable browser on the client device without requiring the user to separately install each tool on the client device itself. Since users do not install and execute the software tools or the client applications (original or modified) locally, embodiments of the present invention can reduce or eliminate storage requirements for such tools and client applications on the client devices, reduce network bandwidth requirements (since each tool package and/or client application does not need to be downloaded/uploaded in its entirety to/from each client device), and the like.

[0094] The monetization module **526** can provide tools and support for monetizing both modified and unmodified cloud-based client applications. In an embodiment, the monetization module **524** can allow users to mint, buy, and sell NFTs to monetize modified cloud-based client applications in an NFT ecosystem. The present invention can allow for minting of NFTs on a suitable blockchain, saving users' cloud-based client applications as NFTs, and creating a process for cloud-based client application developers and content creators to earn on cloud-based client applications or

instances of the cloud-based client applications (e.g., a particular "mod" of a cloud-based client application, a particular competitive and/or popular match of a cloud-based client game application or the like). Once a user has modified a cloud-based client application via the client application module **522**, the monetization module **526** can provide appropriate tools and functionality to mint an NFT based on the modified cloud-based client application. For example, the monetization module **526** can provide the user (via the display module on the client device) with a selection of blockchain technologies that can be used for the NFT. The monetization module **526** can also provide the user (via the display module on the client device) with appropriate tools to create a digital wallet and a selection of marketplaces for the NFT. In an embodiment, the server system **520** can provide a suitable infrastructure for supporting an NFT marketplace ecosystem. In an alternative embodiment, the monetization module **526** can communicate with one or more appropriate third-party NFT marketplaces. For purposes of discussion and not limitation, if the user selects a third-party NFT marketplace, the monetization module **526** can upload the digital file of the modified cloud-based client application to the third-party NFT marketplace. Once uploaded, the monetization module **526** can provide selections to the user (via the display module on the client device) for choosing how to monetize the NFT. Once the user has selected the various characteristics of the NFT, the monetization module **526** can instruct the third-party marketplace to create the NFT, which can then be sold on the third-party NFT marketplace. Alternatively, the monetization module **526** can perform one or more functions of the third-party NFT marketplace to generate the NFT prior to uploading it to the third-party NFT marketplace.

[0095] In an embodiment, the monetization module **526** can also provide tools and functionality to support the purchase of NFTs of modified cloud-based client applications by users of the cloud-based client application platform **500**. For example, the monetization module **526** can provide tools (accessible to users via the display modules on client devices) for opening an NFT marketplace account (either with a NFT marketplace managed by the server system **520** or a third-party NFT marketplace). The monetization module **526** can also provide tools (accessible to users via the display modules on client devices) for creating a blockchain wallet and funding the wallet with the correct cryptocurrency or token required to participate in the NFT marketplace's activity, which can allow the user to purchase NFTs of modified cloud-based client applications via the cloud-based client application platform **500**. Thus, exemplary embodiments of the present invention can combine cloud-based "mods" of cloud-based client applications with NFTs to address problems of ownership, authenticity, provenance, and attribution. Additionally, the present invention can ensure that programmatic smart contracts govern the distribution and monetization of cloud-based client application NFTs (e.g., payments split between developer, publishers/platform, and creators). The "modded" cloud-based client applications can be distributed independently by creators, monetized as NFTs as the creators see fit, and be purchased/sold in a suitable NFT marketplace.

[0096] In an embodiment, the display module on each client device can provide the user with a suitable interface through which to access and use the monetization module **526**. Using the display module on the client device, the user can mint, buy, and sell NFTs to monetize modified cloud-based client applications in an NFT ecosystem using the monetization module **526**. For example, the display module on the client device can display appropriate tools and functionality (available from the monetization module **526**) to mint an NFT based on the modified cloud-based client application. For example, the display module can display to the user a selection of blockchain technologies (available from the monetization module **526**) that can be used for the NFT. The display module can also display to the user appropriate tools (available from the monetization module **526**) to create a digital wallet and a selection of marketplaces for the NFT. For purposes of illustration and not limitation, if the user selects a third-party NFT marketplace, the display module on the client device can display functionality (available from the monetization module **526**) to allow the user to upload the digital file of the modified cloud-based client application to the third-party NFT marketplace. Once uploaded, the display module can display selections to the user (available from the monetization module **526**) for choosing how to monetize the NFT. Once the user has selected the various characteristics of the NFT, the display module can cause the monetization module **526** to instruct the third-party marketplace to create the NFT, which can then be sold on the third-party NFT marketplace. The display module can also display tools and provide functionality (available from the monetization module **526**) to support the purchase of NFTs of modified cloud-based client applications by users of the cloud-based client application platform **500** via the client devices. For example, the display module can display suitable tools and functionality to the user (available from the monetization module **526**) for opening an NFT marketplace account. The display module can also display appropriate tools and functionality to the user (available from the monetization module **526**) for creating a blockchain wallet and funding the wallet with the correct cryptocurrency or token required to participate in the activity of the NFT marketplace, which can allow the user to purchase NFTs of modified cloud-based client applications via the cloud-based client application platform **500**.

[0097] FIG. **6** is a flowchart illustrating an example method **600** for remotely modifying and monetizing cloud-based client applications using, for example, the server system **520** of FIG. **5** (although the server system **102** from FIG. **1**, **2**, **3** or **4** could alternatively be used), in accordance with embodiments of the disclosure. At block **605**, the server system **520** can receive a first request from a client device (e.g., client device **502**) of a first user to initiate an interactive session with a cloud-based client application. The cloud-based client application can be configured to execute on a computing platform different from the client device. At block **610**, the server system **520** can reserve, in response to the first request, an application engine for executing the cloud-based client application remotely from the client device (e.g., using the client application module **522**). The application engine can be bound or otherwise linked to the first user for the duration of the interactive session. At block **615**, the server system **520** can receive a second request from the client device to modify the cloud-based client application using one or more software tools. The one or more software tools can be configured to execute on the computing platform different from the client device. At block **620**, the server system **520** can modify the cloud-based client application in the application engine based on

the second request (e.g., using the software tools module **524**). At block **625**, the server system **520** can receive a third request from the client device to convert the modified cloud-based client application into a digital asset. At block **630**, the server system **520** can convert the modified cloud-based client application into the digital asset based on the third request (e.g., using the monetization module **526**). At block **635**, the server system **520** can receive a fourth request from the client device to monetize the digital asset. At block **640**, the server system **520** can monetize the digital asset using a third-party digital asset marketplace based on the fourth request (e.g., using the monetization module **526**). At block **645**, the server system **520** can transmit a result of the monetization of the digital asset to the client device. At block **650**, the server system **520** can receive a fifth request from the client device to end the interactive session with the modified cloud-based client application. At block **655**, the server system **520** can deregister, in response to the fifth request, the application engine to end the interactive session (e.g., using the client application module **522**). The application engine can be unbound or otherwise unlinked from the first user and the modified cloud-based client application can be available for use by a second or subsequent user.

[0098] FIG. **7** is a flowchart illustrating an example method **700** for remotely interacting with client applications using, for example, the server system **102** of FIG. **1**, **2**, **3** or **4** (although the server system **520** from FIG. **5** could alternatively be used), in accordance with embodiments of the disclosure. At block **705**, the server system **102** can receive a first request from a client device (e.g., client device **150**) of a first user to initiate an interactive session with a cloud-based client application. The first request can include one or more characteristics of at least one of the first user and the client device. The cloud-based client application can be configured to execute on a computing platform different from the client device. At block **710**, the server system **102** can reserve, in response to the first request, an application engine (e.g., from the plurality of application engine modules **106** or **206**) for executing the cloud-based client application remotely from the client device. The application engine can be bound or otherwise linked to the first user for the duration of the interactive session. In an embodiment, each application engine can be reserved for one user such that each application engine can be used by at most a single (bound) user during an interactive session. At block **715**, the server system **102** can modify the cloud-based client application based on the one or more characteristics of at least one of the first user and the client device. In embodiments, the one or more characteristics of the first user and/or the client device can be used to alter the cloud-based client application that executes in the application engine (e.g., for a logged-in state or the like). At block **720**, the server system **102** can execute the modified cloud-based client application in the application engine. At block **725**, the server system **102** can stream media data from the modified cloud-based client application to a browser application or other suitable application (e.g., browser module **148**) executing on the client device of the first user. At block **730**, the server system **102** can receive interaction data from the client device as the first user engages with the streamed media data from the modified cloud-based client application via the browser application. At block **735**, the server system **102** can receive a second request from the client device to end the interactive session with the modified cloud-based client application. In an embodiment, the second request can be a termination of the stream to which the client device is connected rather than a specific request from the client to end the interactive session. At block **740**, the server system **102** can deregister or deallocate the application engine in response to the second request. In an embodiment, the application engine can be deregistered or deallocated from inventory and the processor(s), and process(es) supporting the application engine can be spun down or otherwise terminated so that the application engine is not reused for multiple users. In an alternative embodiment, the application engine can be unbound or otherwise unlinked from the first user and available for use by a second (or subsequent) user.

[0099] Embodiments of the present invention can improve network efficiency by not requiring users to download and install to their client device numerous (e.g., dozens, hundreds, or more) different client applications with which they would like to interact or the software tools to modify and monetize each of those client applications. Instead, the present invention can improve network traffic and network bandwidth utilization, since it is the video/audio data (from the client applications and software tools) and user input that are communicated between the server system (e.g., server system **102** or server system **520**) and the client device (e.g., client device **150** or client devices **502**, **506**, **510**, . . . , **514**) rather than the entire client application itself or any of the software tools. Additionally, the present invention can reduce computer storage requirements, both on the client side and the server side. For example, users would not be required to download and install any client applications or software tools on and to their client device. In some implementations of the present invention, the server system **102** can store a single version of a cloud-based client application and the software tools which can be accessed from and distributed to any type of client device, instead of having to store numerous different versions of the client application to support a plethora of different types of client devices and software tools to support a myriad of different types of client applications.

[0100] Embodiments of the present invention can also improve the processing efficiency of the client devices **150**, since the client devices **150** can process the video/audio data from the cloud-based client application and software tools more quickly and more efficiently than executing the entire client application and software tools themselves natively on the client device. Embodiments of the present invention can also improve the processing efficiency of the server system **102**. The server system **102** can allocate computing resources when an application engine module and/or application instance module is registered for an active session and deallocate those resources when the active session is completed, rather than maintaining continuous execution of and computing resources to support the cloud-based client application on the server system **102**. In other words, the server system **102** can register and deregister application engine modules and/or application instance modules, respectively, as needed without having to continually maintain and process each cloud-based client application and software tools for each client device. Consequently, the server system **102** can more effectively allocate computer processing resources to those application engine modules and/or application instance modules (and the cloud-based client applications running in those application engine modules or as application instances) that are being used and away from those

application engine modules and/or application instance modules that are not. In some implementations of the present invention, the pre-warming of application engine modules in application engine module pools and/or application instances in application instance pools can also substantially reduce the response time for establishing active sessions. Consequently, embodiments of the present invention can manage users in active sessions at scale to maintain instant (or nearly instant) startup and interaction with the cloud-based client applications (e.g., "instant play" in the context of cloud-based mobile games), particularly when there are, for example, hundreds or thousands of cloud-based client applications, hundreds or thousands of software tools, and millions, tens of millions, or more users interacting with those cloud-based client applications concurrently.

[0101] FIG. **8** is a block diagram of an example computing device **800** that may perform one or more of the operations described herein, in accordance with the present embodiments. The computing device **800** may be connected to other computing devices in a LAN, an intranet, an extranet, and/or the Internet. The computing device **800** may operate in the capacity of a server machine in client-server network environment or in the capacity of a client in a peer-to-peer network environment. The computing device **800** may be provided by a personal computer (PC), a set-top box (STB), a server, a network router, switch or bridge, or any machine capable of executing a set of instructions (sequential or otherwise) that specify actions to be taken by that machine. Further, while only a single computing device **800** is illustrated, the term "computing device" shall also be taken to include any collection of computing devices that individually or jointly execute a set (or multiple sets) of instructions to perform the methods discussed herein.

[0102] The example computing device **800** may include a computer processing device **802** (e.g., a general purpose processor, ASIC, etc.), a main memory **804**, a static memory **806** (e.g., flash memory or the like), and a data storage device **808**, which may communicate with each other via a bus **830**. The computer processing device **802** may be provided by one or more general-purpose processing devices such as a microprocessor, central processing unit, or the like. In an illustrative example, computer processing device **802** may comprise a complex instruction set computing (CISC) microprocessor, reduced instruction set computing (RISC) microprocessor, very long instruction word (VLIW) microprocessor, or a processor implementing other instruction sets or processors implementing a combination of instruction sets. The computer processing device **802** may also comprise one or more special-purpose processing devices, such as an application specific integrated circuit (ASIC), a field programmable gate array (FPGA), a digital signal processor (DSP), network processor, or the like. The computer processing device **802** may be configured to execute the operations described herein, in accordance with one or more aspects of the present disclosure, for performing the operations and steps discussed herein.

[0103] The computing device **800** may further include a network interface device **812**, which may communicate with a network **814**. The data storage device **808** may include a machine-readable storage medium **828** on which may be stored one or more sets of instructions, e.g., instructions for carrying out the operations described herein, in accordance with one or more aspects of the present disclosure. Instructions **818** implementing core logic instructions **826** may also reside, completely or at least partially, within main memory **804** and/or within computer processing device **802** during execution thereof by the computing device **800**, main memory **804** and computer processing device **802** also constituting computer-readable media. The instructions may further be transmitted or received over the network **814** via the network interface device **812**.

[0104] While machine-readable storage medium **828** is shown in an illustrative example to be a single medium, the term "computer-readable storage medium" should be taken to include a single medium or multiple media (e.g., a centralized or distributed database and/or associated caches and servers) that store the one or more sets of instructions. The term "computer-readable storage medium" shall also be taken to include any medium that is capable of storing, encoding or carrying a set of instructions for execution by the machine and that cause the machine to perform the methods described herein. The term "computer-readable storage medium" shall accordingly be taken to include, but not be limited to, solid-state memories, optical media, magnetic media, and the like.

[0105] Embodiments of the subject matter and the operations described in this disclosure can be implemented in digital electronic circuitry, or in computer software, firmware, or hardware, including the structures disclosed in this disclosure and their structural equivalents, or in combinations of one or more of them. Embodiments of the subject matter described in this disclosure can be implemented as one or more computer programs, i.e., one or more modules of computer program instructions, encoded on computer storage medium for execution by, or to control the operation of, data processing apparatus. Alternatively, or in addition, the program instructions can be encoded on an artificially generated propagated signal, e.g., a machine-generated electrical, optical, or electromagnetic signal that is generated to encode information for transmission to suitable receiver apparatus for execution by a data processing apparatus. A computer storage medium can be, or be included in, a computer-readable storage device, a computer-readable storage substrate, a random or serial access memory array or device, or a combination of one or more of them. Moreover, while a computer storage medium is not a propagated signal, a computer storage medium can be a source or destination of computer program instructions encoded in an artificially generated propagated signal. The computer storage medium can also be, or be included in, one or more separate physical components or media (e.g., multiple CDs, disks, or other storage devices).

[0106] The operations described in this disclosure can be implemented as operations performed by a data processing apparatus on data stored on one or more computer-readable storage devices or received from other sources.

[0107] The term "data processing apparatus" encompasses all kinds of apparatus, devices, and machines for processing data, including by way of example a programmable processor, a computer processing device, a computer, a system on a chip, or multiple ones, or combinations, of the foregoing. A computer processing device may include one or more processors which can include special purpose logic circuitry, e.g., an FPGA (field programmable gate array) or an ASIC (application specific integrated circuit), a central processing unit (CPU), a multi-core processor, etc. The apparatus can also include, in addition to hardware, code that creates an execution environment for the computer program in ques-

tion, e.g., code that constitutes processor firmware, a protocol stack, a database management system, an operating system, a cross-platform runtime environment, a virtual machine, or a combination of one or more of them. The apparatus and execution environment can realize various different computing model infrastructures, such as web services, distributed computing and grid computing infrastructures.

[0108] A computer program (also known as a program, software, software application, script, or code) can be written in any form of programming language, including compiled or interpreted languages, declarative, procedural, or functional languages, and it can be deployed in any form, including as a standalone program or as a module, component, subroutine, object, or other unit suitable for use in a computing environment. A computer program may, but need not, correspond to a file in a file system. A program can be stored in a portion of a file that holds other programs or data (e.g., one or more scripts stored in a markup language resource), in a single file dedicated to the program in question, or in multiple coordinated files (e.g., files that store one or more modules, sub programs, or portions of code). A computer program can be deployed to be executed on one computer or on multiple computers that are located at one site or distributed across multiple sites and interconnected by a communication network.

[0109] The processes and logic flows described in this disclosure can be performed by one or more programmable processors executing one or more computer programs to perform actions by operating on input data and generating output. The processes and logic flows can also be performed by, and apparatus can also be implemented as, special purpose logic circuitry, e.g., an FPGA (field programmable gate array) or an ASIC (application specific integrated circuit).

[0110] Processors suitable for the execution of a computer program include, by way of example, both general and special purpose microprocessors, and any one or more processors of any kind of digital computer. Generally, a processor will receive instructions and data from a read only memory or a random access memory or both. The essential elements of a computer are a processor for performing actions in accordance with instructions and one or more memory devices for storing instructions and data. Generally, a computer will also include, or be operatively coupled to receive data from or transfer data to, or both, one or more mass storage devices for storing data, e.g., magnetic disks, magneto optical disks, optical disks, solid state drives, or the like. However, a computer need not have such devices. Moreover, a computer can be embedded in another device, e.g., a smart phone, a mobile audio or media user, a game console, a Global Positioning System (GPS) receiver, or a portable storage device (e.g., a universal serial bus (USB) flash drive), to name just a few. Devices suitable for storing computer program instructions and data include all forms of non-volatile memory, media and memory devices, including, by way of example, semiconductor memory devices, e.g., EPROM, EEPROM, and flash memory devices; magnetic disks, e.g., internal hard disks or removable disks; magneto optical disks; and CD ROM and DVD-ROM disks. The processor and the memory can be supplemented by, or incorporated in, special purpose logic circuitry.

[0111] To provide for interaction with a user, embodiments of the subject matter described in this specification can be implemented on a computer having a display device, e.g., a CRT (cathode ray tube), LCD (liquid crystal display) monitor, a light emitting diode (LED) monitor, or the like, for displaying information to the user and a keyboard and a pointing device, e.g., a mouse, a trackball, a touchpad, a stylus, or the like, by which the user can provide input to the computer. Other kinds of devices can be used to provide for interaction with a user as well; for example, feedback provided to the user can be any form of sensory feedback, e.g., visual feedback, auditory feedback, or tactile feedback; and input from the user can be received in any form, including acoustic, speech, or tactile input. Other possible input devices include touch screens or other touch-sensitive devices such as single or multi-point resistive or capacitive trackpads, voice recognition hardware and software, optical scanners, optical pointers, digital image capture devices and associated interpretation software, and the like. In addition, a computer can interact with a user by sending resources to and receiving resources from a device that is used by the user; for example, by sending web pages to a web browser on a user's client device in response to requests received from the web browser.

[0112] Embodiments of the subject matter described in this disclosure can be implemented in a computing system that includes a back end component, e.g., as a data server, or that includes a middleware component, e.g., an application server, or that includes a front end component, e.g., a client computer having a graphical user interface or a web browser through which a user can interact with an implementation of the subject matter described in this disclosure, or any combination of one or more such back end, middleware, or front end components. The components of the system can be interconnected by any form or medium of digital data communication, e.g., a communication network. Examples of communication networks include a local area network ("LAN") and a wide area network ("WAN"), an inter-network (e.g., the Internet), peer-to-peer networks (e.g., ad hoc peer-to-peer networks), and the like.

[0113] The computing system can include clients and servers. A client and server are generally remote from each other and typically interact through a communication network. The relationship of client and server arises by virtue of computer programs running on the respective computers and having a client-server relationship to each other. In some embodiments, a server transmits data (e.g., an HTML page) to a client device (e.g., for purposes of displaying data to and receiving user input from a user interacting with the client device). Data generated at the client device (e.g., a result of the user interaction) can be received from the client device at the server.

[0114] A system of one or more computers can be configured to perform particular operations or actions by virtue of having software, firmware, hardware, or a combination of them installed on the system that in operation causes or cause the system to perform the actions. One or more computer programs can be configured to perform particular operations or actions by virtue of including instructions that, when executed by data processing apparatus, cause the apparatus to perform the actions.

[0115] Reference throughout this disclosure to "one embodiment" or "an embodiment" means that a particular feature, structure, or characteristic described in connection with the embodiments included in at least one embodiment. Thus, the appearances of the phrase "in one embodiment" or

"in an embodiment" in various places throughout this disclosure are not necessarily all referring to the same embodiment. In addition, the term "or" is intended to mean an inclusive "or" rather than an exclusive "or."

[0116] While this disclosure contains many specific implementation details, these should not be construed as limitations on the scope of any inventions or of what may be claimed, but rather as descriptions of features specific to particular embodiments of particular inventions. Certain features that are described in this disclosure in the context of separate embodiments can also be implemented in combination in a single embodiment. Conversely, various features that are described in the context of a single embodiment can also be implemented in multiple embodiments separately or in any suitable subcombination. Moreover, although features may be described above as acting in certain combinations and even initially claimed as such, one or more features from a claimed combination can in some cases be excised from the combination, and the claimed combination may be directed to a subcombination or variation of a subcombination.

[0117] Similarly, while operations and/or logic flows are depicted in the drawings and/or described herein in a particular order, this should not be understood as requiring that such operations and/or logic flows be performed in the particular order shown or in sequential order, or that all illustrated operations be performed, to achieve desirable results. In certain circumstances, multitasking and parallel processing may be advantageous. Moreover, the separation of various system components in the embodiments described above should not be understood as requiring such separation in all embodiments, and it should be understood that the described program components and systems can generally be integrated together in a single software product or packaged into multiple software products.

[0118] Thus, particular embodiments of the subject matter have been described. Other embodiments are within the scope of the following claims. In some cases, the actions recited in the claims can be performed in a different order and still achieve desirable results. In addition, the processes depicted in the accompanying figures do not necessarily require the particular order shown, or sequential order, to achieve desirable results. In certain implementations, multitasking and parallel processing may be advantageous.

[0119] The words "example" or "exemplary" are used herein to mean serving as an example, instance, or illustration. Any aspect or design described herein as "example" or "exemplary" is not necessarily to be construed as preferred or advantageous over other aspects or designs. Rather, use of the words "example" or "exemplary" is intended to present concepts in a concrete fashion. As used in this application, the term "or" is intended to mean an inclusive "or" rather than an exclusive "or". That is, unless specified otherwise, or clear from context, "X includes A or B" is intended to mean any of the natural inclusive permutations. That is, if X includes A; X includes B; or X includes both A and B, then "X includes A or B" is satisfied under any of the foregoing instances. In addition, the articles "a" and "an" as used in this application and the appended claims should generally be construed to mean "one or more" unless specified otherwise or clear from context to be directed to a singular form. Moreover, use of the term "an embodiment" or "one embodiment" or "an implementation" or "one implementation" throughout is not intended to mean the same embodiment or implementation unless described as such. Furthermore, the terms "first," "second," "third," "fourth," etc. as used herein are meant as labels to distinguish among different elements and may not necessarily have an ordinal meaning according to their numerical designation.

[0120] In the descriptions above and in the claims, phrases such as "at least one of" or "one or more of" may occur followed by a conjunctive list of elements or features. The term "and/or" may also occur in a list of two or more elements or features. Unless otherwise implicitly or explicitly contradicted by the context in which it is used, such a phrase is intended to mean any of the listed elements or features individually or any of the recited elements or features in combination with any of the other recited elements or features. For example, the phrases "at least one of A and B;" "one or more of A and B;" and "A and/or B" are each intended to mean "A alone, B alone, or A and B together." A similar interpretation is also intended for lists including three or more items. For example, the phrases "at least one of A, B, and C;" "one or more of A, B, and C;" and "A, B, and/or C" are each intended to mean "A alone, B alone, C alone, A and B together, A and C together, B and C together, or A and B and C together." In addition, use of the term "based on," above and in the claims is intended to mean, "based at least in part on," such that an unrecited feature or element is also permissible.

[0121] The above description of illustrated implementations of the invention is not intended to be exhaustive or to limit the invention to the precise forms disclosed. While specific implementations of, and examples for, the invention are described herein for illustrative purposes, various equivalent modifications are possible within the scope of the invention, as those skilled in the relevant art will recognize. The subject matter described herein can be embodied in systems, apparatus, methods, and/or articles depending on the desired configuration. The implementations set forth in the foregoing description do not represent all implementations consistent with the subject matter described herein. Instead, they are merely some examples consistent with aspects related to the described subject matter. Although a few variations have been described in detail above, other modifications or additions are possible. In particular, further features and/or variations can be provided in addition to those set forth herein. For example, the implementations described above can be directed to various combinations and subcombinations of the disclosed features and/or combinations and subcombinations of several further features disclosed above. Other implementations may be within the scope of the following claims.

What is claimed is:

1. A method, comprising:

receiving, by at least one data processor, a first request from a first client device of a user to initiate an interactive session with a cloud-based client application, wherein the cloud-based client application is configured to execute on a second client device, and wherein the second client device comprises a computing platform different from the first client device;

reserving, by the at least one data processor and in response to the first request, an application engine from a pre-instantiated application engine pool of a plurality of pre-instantiated application engine pools for executing the cloud-based client application remotely from the first client device;

receiving, by the at least one data processor, a second request from the first client device to modify the cloud-based client application executing in the application engine using one or more software tools, wherein the one or more software tools are configured to execute on the computing platform different from the first client device;

modifying, by the at least one data processor, the cloud-based client application executing in the application engine using the one or more software tools; and

providing, by the at least one data processor, media data associated with the modified cloud-based client application to the first user via a display application executing on the first client device.

2. The method of claim **1**, wherein modifying the cloud-based application executing in the application engine comprises storing data associated with the modified cloud-based application in a database.

3. The method of claim **2**, wherein the modified cloud-based application is configured to be retrieved and used in the application engine in a subsequent active session.

4. The method of claim **1**, wherein using the one or more software tools is performed via a browser installed on the client device.

5. The method of claim **1**, wherein modifying the cloud-based application comprises augmenting content creation, adding new functionality, removing functionality, or changing existing functionality.

6. The method of claim **1**, further comprising:

receiving, by the at least one data processor, a third request from the first client device to convert the modified cloud-based client application into a digital asset; and

converting, by the at least one data processor, the modified cloud-based client application into the digital asset based on the third request.

7. The method of claim **6**, further comprising:

receiving, by the at least one processor, a fourth request from the first client device to transform the digital asset;

monetizing, by the at least one processor, the digital asset via a third-party platform based on the fourth request; and

transmitting, by the at least one processor, a result of the transformation of the digital asset to the first client device.

8. The method of claim **7**, wherein transforming the digital asset comprises converting the digital asset into a non-fungible token (NFT).

9. A system, comprising:

at least one data processor; and

memory storing instructions that, when executed by the at least one data processor, cause the at least one data processor to perform operations comprising:

receiving a first request from a first client device of a user to initiate an interactive session with a cloud-based client application, wherein the cloud-based client application is configured to execute on a second client device, and wherein the second client device comprises a computing platform different from the first client device;

reserving, in response to the first request, an application engine from a pre-instantiated application engine pool of a plurality of pre-instantiated application engine pools for executing the cloud-based client application remotely from the first client device;

receiving a second request from the first client device to modify the cloud-based client application executing in the application engine using one or more software tools, wherein the one or more software tools are configured to execute on the computing platform different from the first client device;

modifying the cloud-based client application executing in the application engine using the one or more software tools; and

providing media data associated with the modified cloud-based client application to the first user via a display application executing on the first client device.

10. The system of claim **9**, wherein modifying the cloud-based application executing in the application engine comprises storing data associated with the modified cloud-based application in a database.

11. The system of claim **10**, wherein the modified cloud-based application is configured to be retrieved and used in the application engine in a subsequent active session.

12. The system of claim **10**, wherein using the one or more software tools is performed via a browser installed on the client device.

13. The system of claim **9**, wherein modifying the cloud-based application comprises augmenting content creation, adding new functionality, removing functionality, or changing existing functionality.

14. The system of claim **9**, wherein the operations further comprise:

receiving a third request from the first client device to convert the modified cloud-based client application into a digital asset; and

converting the modified cloud-based client application into the digital asset based on the third request.

15. The system of claim **14**, wherein the operations further comprise:

receiving a fourth request from the first client device to monetize the digital asset;

monetizing the digital asset via a third-party digital asset marketplace based on the fourth request; and

transmitting a result of the monetization of the digital asset to the first client device.

16. The system of claim **15**, wherein transforming the digital asset comprises converting the digital asset into a non-fungible token (NFT).

17. A computer program product comprising a non-transitory machine readable medium storing instructions that, when executed by at least one programmable processor forming part of at least one computing system, cause the at least one programmable processor to perform operations comprising:

receiving a first request from a first client device of a user to initiate an interactive session with a cloud-based client application, wherein the cloud-based client application is configured to execute on a second client device, and wherein the second client device comprises a computing platform different from the first client device;

reserving, in response to the first request, an application engine from a pre-instantiated application engine pool of a plurality of pre-instantiated application engine

pools for executing the cloud-based client application remotely from the first client device;

receiving a second request from the first client device to modify the cloud-based client application executing in the application engine using one or more software tools, wherein the one or more software tools are configured to execute on the computing platform different from the first client device;

modifying the cloud-based client application executing in the application engine using the one or more software tools; and

providing media data associated with the modified cloud-based client application to the first user via a display application executing on the first client device.

* * * * *