US 20120172135A1

(19) United States
(12) Patent Application Publication
Kniberg et al.
(10) Pub. No.: US 2012/0172135 A1
(43) Pub. Date: Jul. 5, 2012

(54) GAMING SYSTEM

(76) Inventors: **Henrik Kniberg**, Ska (SE); **Ulf Abrink**, Balsta (SE); **Jens Nilsson**, Saltsjo-Boo (SE); **Peter Vincent**, Hagersten (SE); **Thomas Rizos**, Nacka (SE); **Hakan Andersson**, Sundbyberg (SE); **Jan Otterling**, Vallingby (SE); **Ulf Larsson**, Taby (SE)

(21) Appl. No.: **13/413,180**

(22) Filed: **Mar. 6, 2012**

Related U.S. Application Data

(63) Continuation of application No. 12/433,615, filed on Apr. 30, 2009, now abandoned, which is a continuation of application No. 11/719,059, now abandoned, filed as application No. PCT/SE05/01717 on Nov. 14, 2005.

(60) Provisional application No. 60/522,843, filed on Nov. 12, 2004.

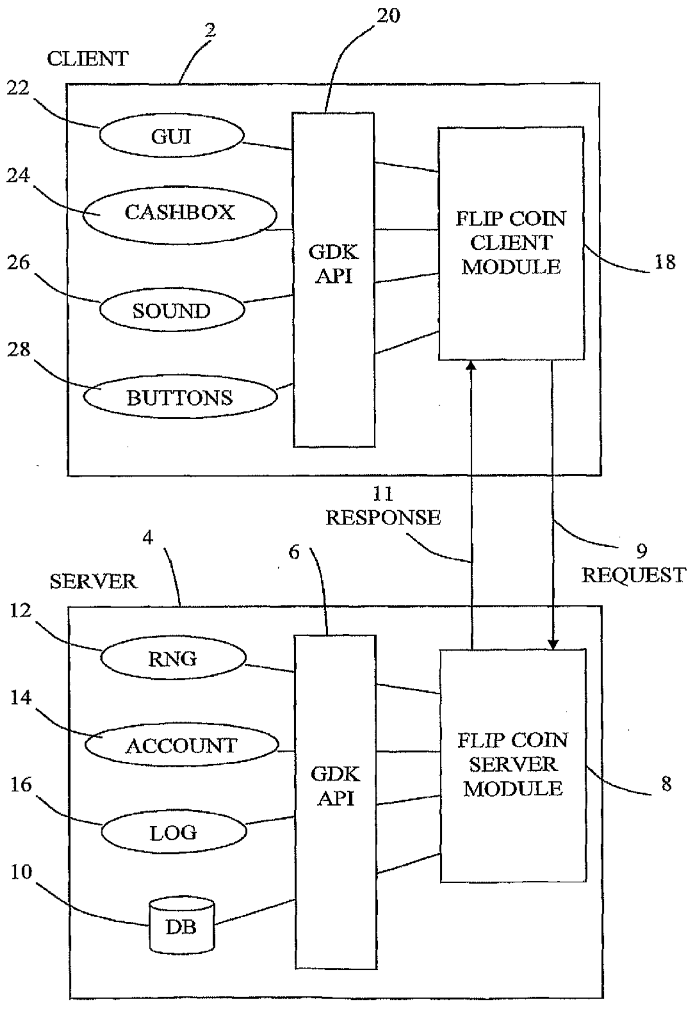
Publication Classification

(51) **Int. Cl.**
A63F 9/24 (2006.01)

(52) **U.S. Cl.** **463/42**

(57) ABSTRACT

A method and a system for operating a gaming application in a computer based gaming system having a client gaming machine connected to a gaming server, wherein: a client module of a game application program is executed in the client gaming machine; game parameters are determined in response to player input signals and dependent on logical rules for the game defined in said client module; a server module of said game application program is executed in the gaming server in response to said game parameters being communicated to said server module; outcome parameters are determined depended on conditions for the game defined in said server module, dependent on said communicated game parameters and dependent on a random number generated in the gaming server.



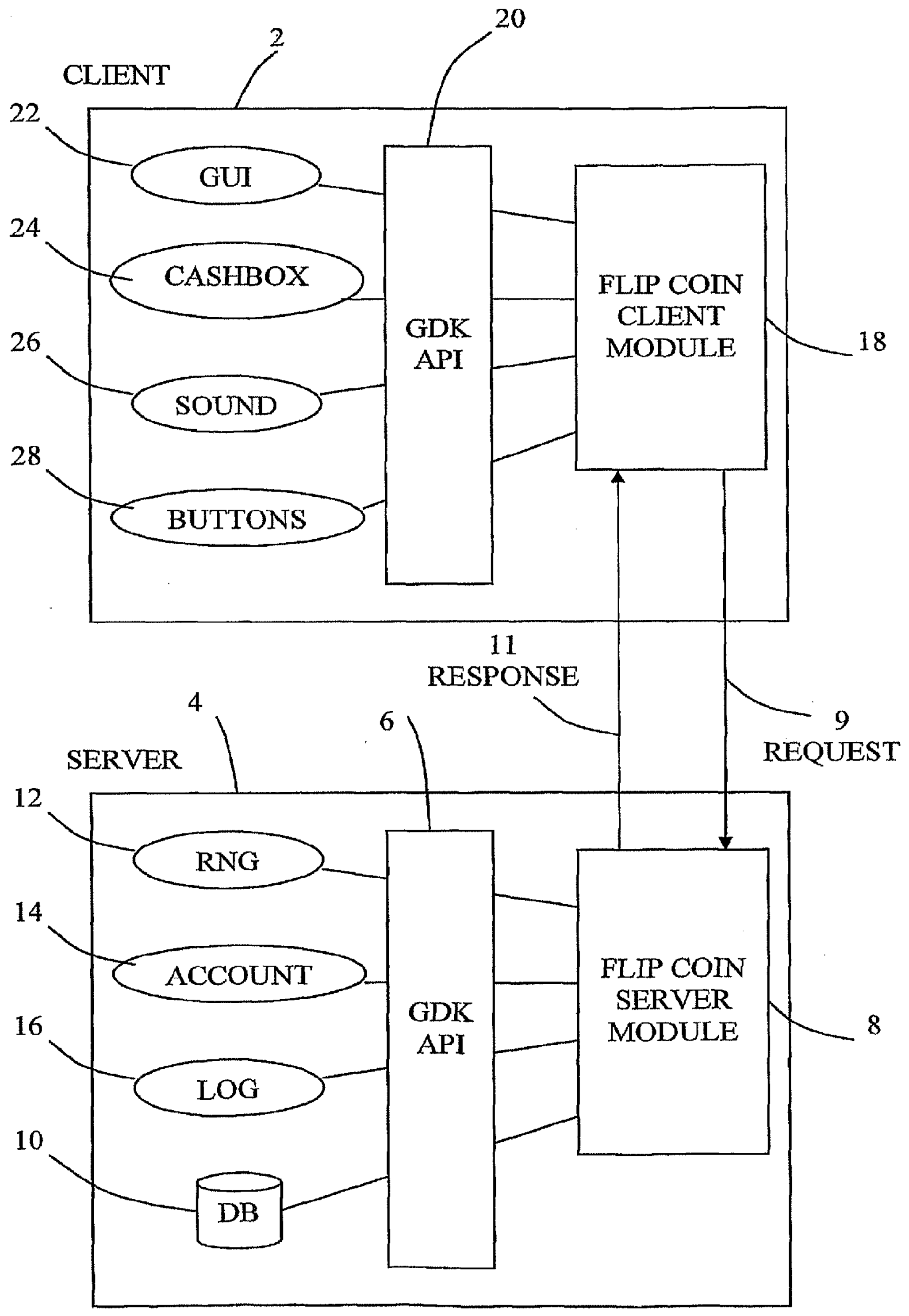

2
CLIENT
20
22
GUI
24
CASHBOX
26
SOUND
28
BUTTONS
GDK API
FLIP COIN CLIENT MODULE
18
11
RESPONSE
9
REQUEST
4
SERVER
6
12
RNG
14
ACCOUNT
16
LOG
10
DB
GDK API
FLIP COIN SERVER MODULE
8

FIG 1

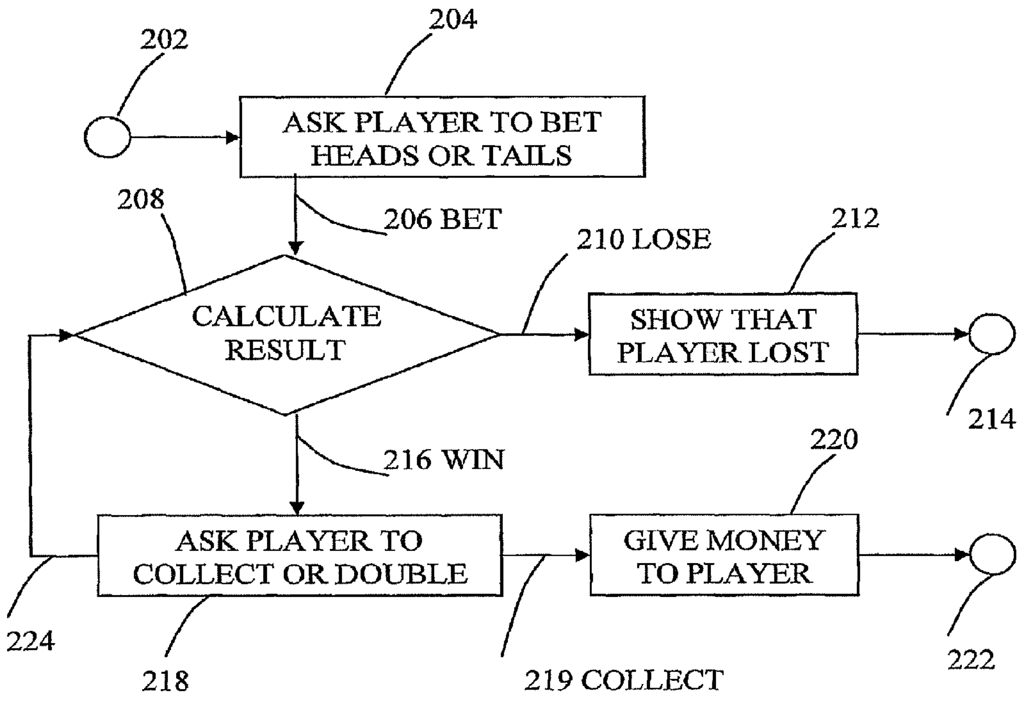

202
204
ASK PLAYER TO BET HEADS OR TAILS
208
206 BET
CALCULATE RESULT
210 LOSE
212
SHOW THAT PLAYER LOST
214
216 WIN
220
ASK PLAYER TO COLLECT OR DOUBLE
GIVE MONEY TO PLAYER
224
218
219 COLLECT
222

FIG 2

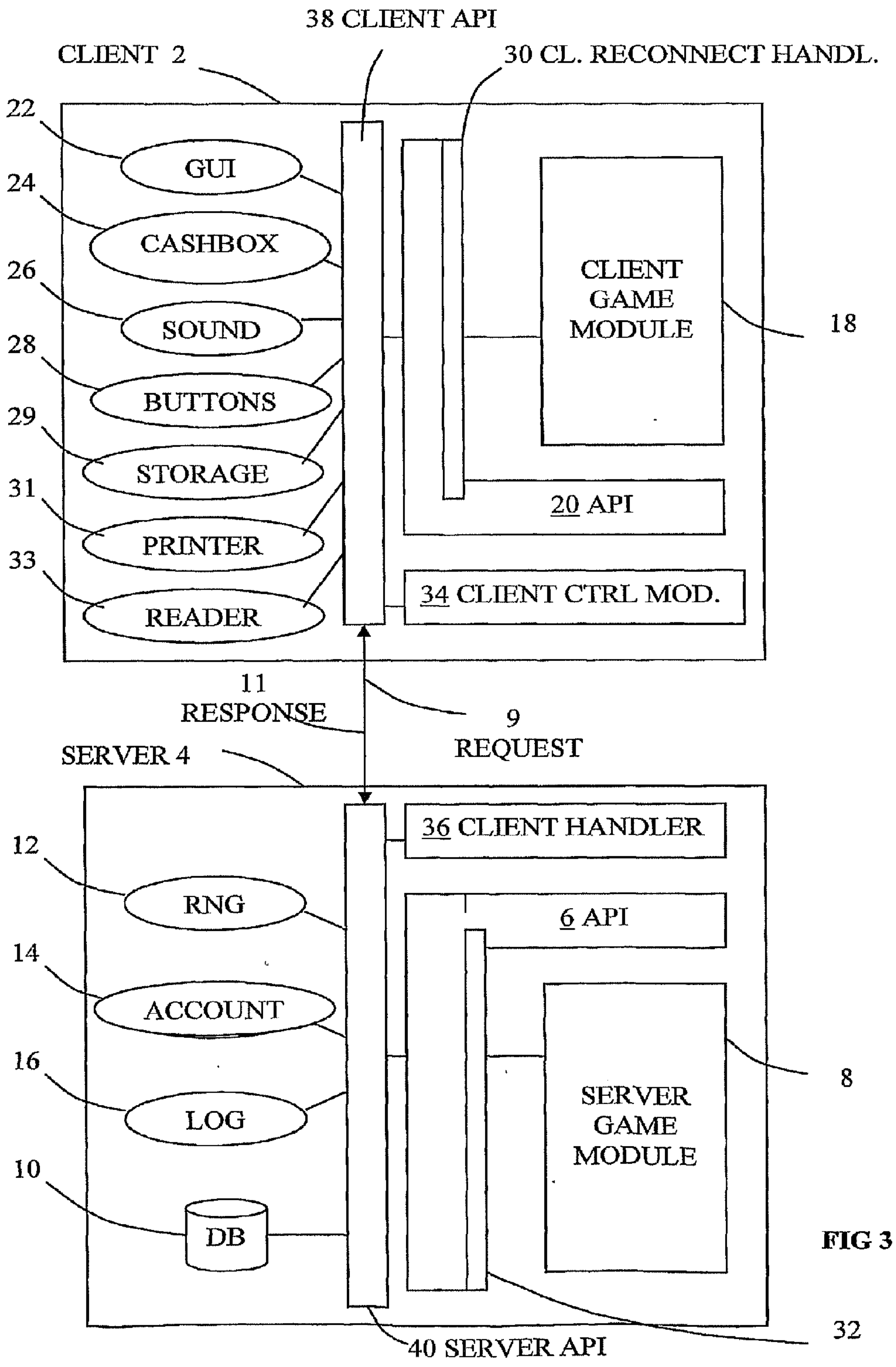

38 CLIENT API
CLIENT 2
30 CL. RECONNECT HANDL.
22
GUI
24
CASHBOX
26
SOUND
28
BUTTONS
29
STORAGE
31
PRINTER
33
READER
CLIENT GAME MODULE
18
20 API
34 CLIENT CTRL MOD.
11
RESPONSE
9
REQUEST
SERVER 4
36 CLIENT HANDLER
12
RNG
14
ACCOUNT
16
LOG
10
DB
6 API
SERVER GAME MODULE
8
32
40 SERVER API

FIG 3

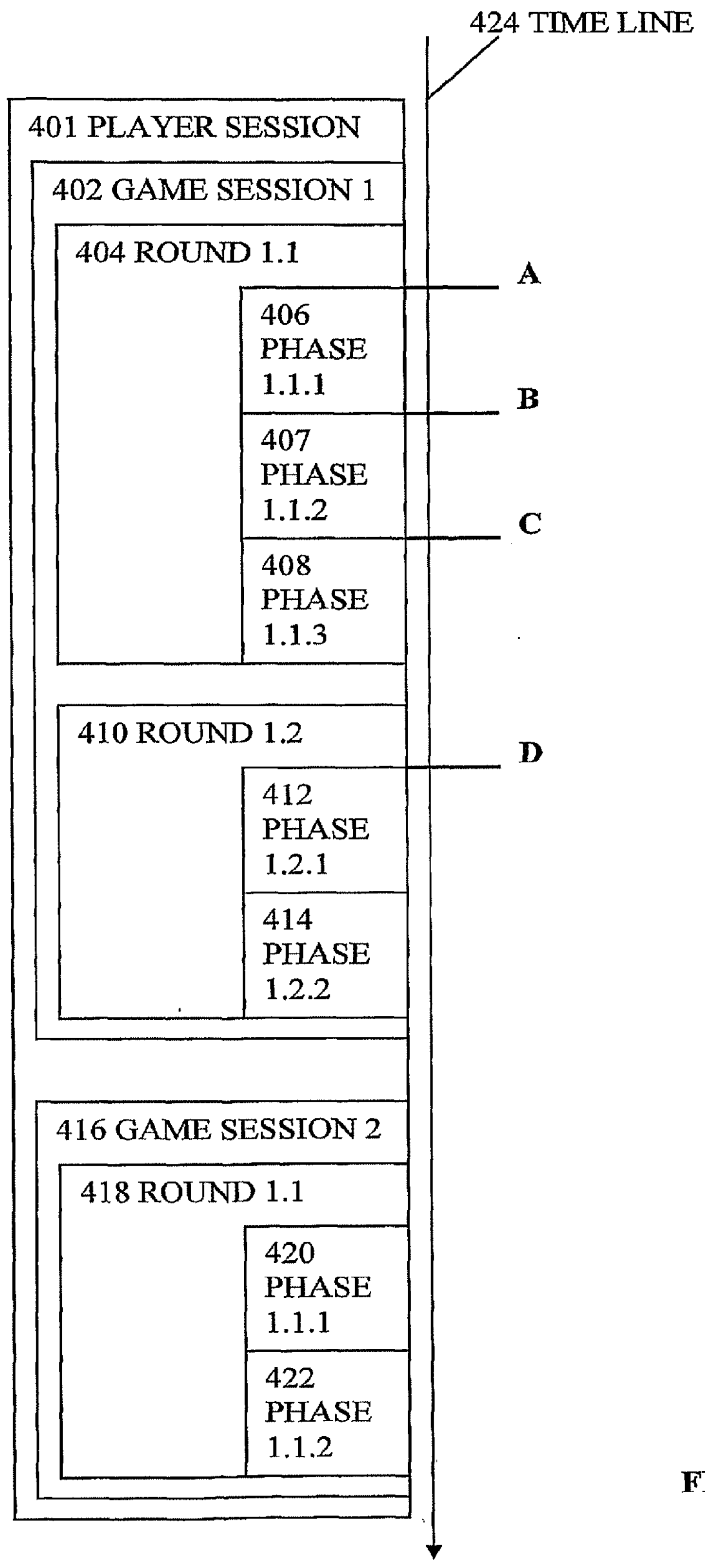

424 TIME LINE
401 PLAYER SESSION
402 GAME SESSION 1
404 ROUND 1.1
406 PHASE 1.1.1
407 PHASE 1.1.2
408 PHASE 1.1.3
A
B
C
410 ROUND 1.2
D
412 PHASE 1.2.1
414 PHASE 1.2.2
416 GAME SESSION 2
418 ROUND 1.1
420 PHASE 1.1.1
422 PHASE 1.1.2

FIG 4

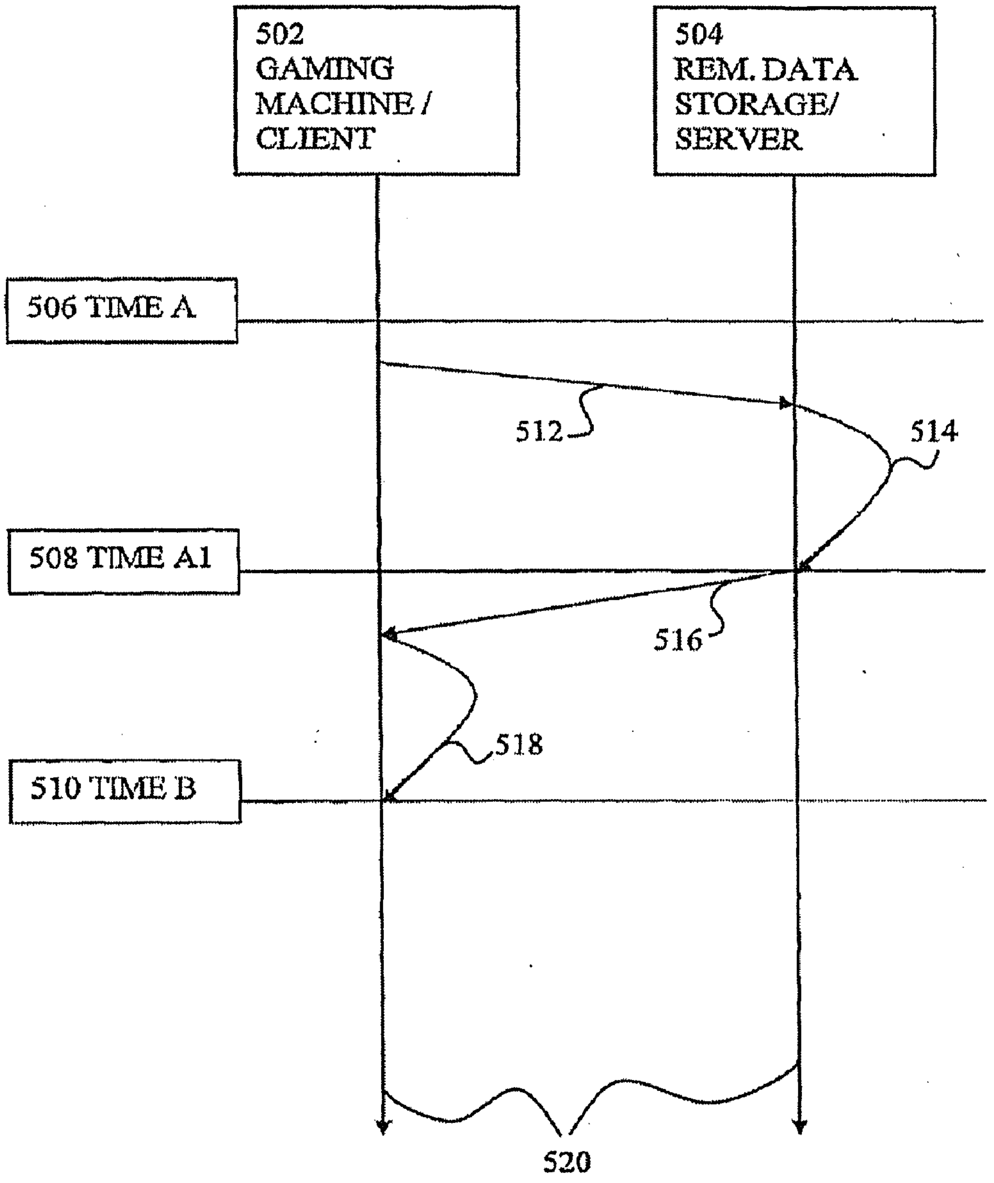

502
GAMING
MACHINE /
CLIENT
504
REM. DATA
STORAGE/
SERVER
506 TIME A
508 TIME A1
510 TIME B
512
514
516
518
520

FIG 5

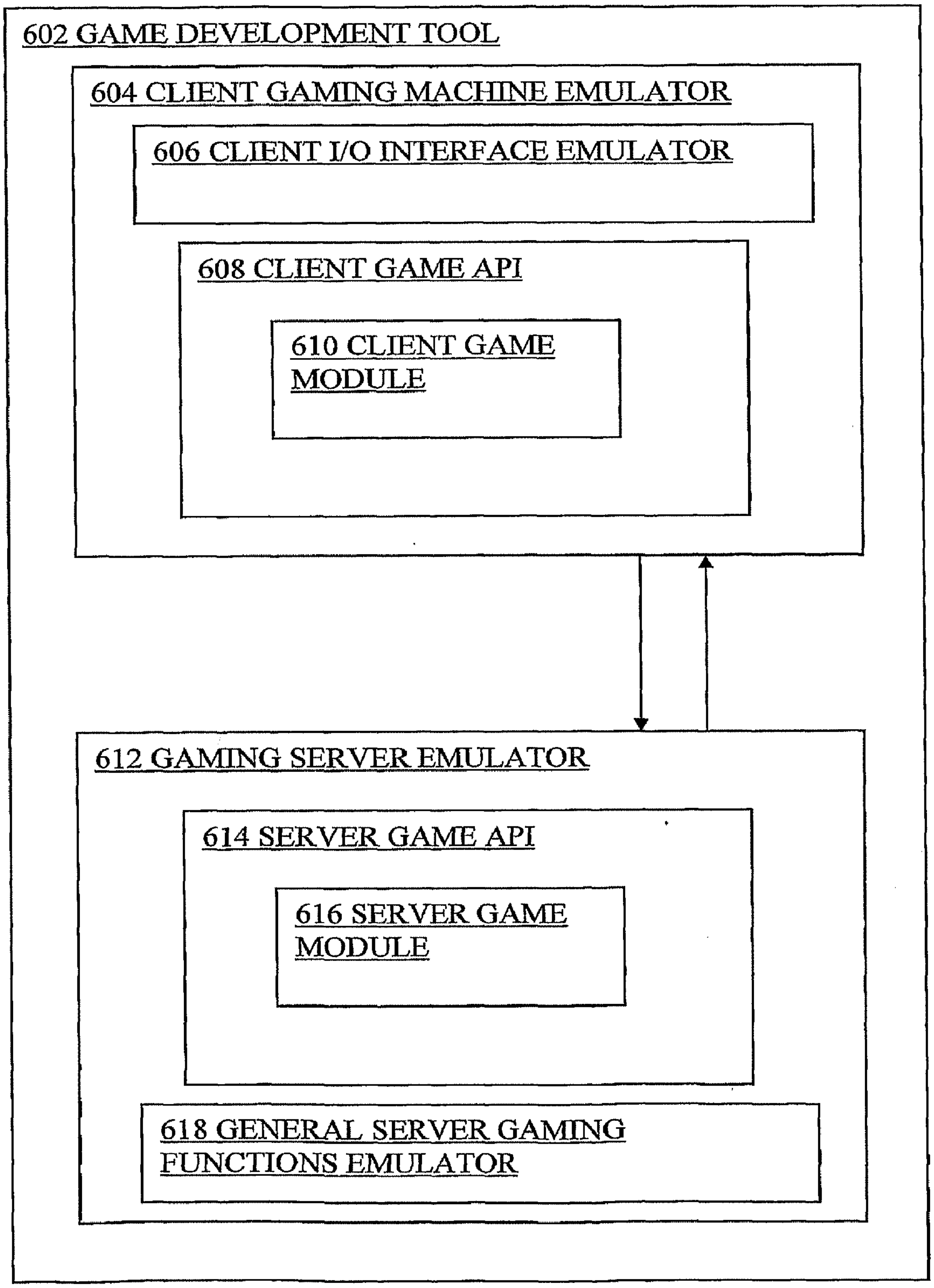

602 GAME DEVELOPMENT TOOL
604 CLIENT GAMING MACHINE EMULATOR
606 CLIENT I/O INTERFACE EMULATOR
608 CLIENT GAME API
610 CLIENT GAME MODULE
612 GAMING SERVER EMULATOR
614 SERVER GAME API
616 SERVER GAME MODULE
618 GENERAL SERVER GAMING FUNCTIONS EMULATOR

FIG 6

GAMING SYSTEM

TECHNICAL FIELD

[0001] The present invention relates in general to a gaming system for operating computerized games of chance, the system comprising gaming machines communicating with a central gaming server.

BACKGROUND

[0002] Traditionally, computerized games of chance have mainly been developed by providers of gaming machines and gaming systems, whereas independent game developers largely have been barred from this market. The reasons are inter alia found in regulatory barriers with regard to betting, paying out winning prizes, security and the like. Another reason is the reluctance of the main providers of gaming machines to expose themselves to increased competition. This has resulted in a rigid tradition that games as well as control and operation information has been implemented in decentralized systems provided by manufacturers of gaming machines, also called interactive video terminals abbreviated IVT:s. However, this manner of manufacturing and operating gaming systems is inefficient, costly, difficult to update and offers a limited control of functionality and security. There is therefore a demand for more flexible gaming systems.

PRIOR ART

[0003] Examples of prior art is found in the following patent publications, which for the sake of enablement of per se known parts of the invention are incorporated by reference in this specification.

[0004] WO03023647 discloses a method for developing gaming programs compatible with a computerized gaming operating system and apparatus.

[0005] WO2004051588 discloses a hosted game development environment.

[0006] US20030148806 discloses a method and apparatus for optimizing game design and development upon multiple game systems.

[0007] WO03028828 discloses a game development architecture that decouples the game logic from the graphic logics.

[0008] US20040180722 discloses a centralized gaming system with modifiable remote display terminals.

[0009] KR010007668 discloses a method and system for network game service.

[0010] CA2358237 discloses a gaming hardware simulator.

[0011] US20020028705 discloses a system and method for generating games and tests.

OBJECT OF THE INVENTION

[0012] It is an object of the present invention to provide a system for operating games in a computer based gaming system, which enables game application production for an open gaming system architecture.

SUMMARY OF THE INVENTION

[0013] The object is achieved by providing a gaming system concept in a client-server based architecture devised to operate modular game application programs for executing a game of chance.

[0014] A game application program comprises a game specific client module that is executed in a client gaming machine provided with a game application program interface and a game specific server module executed in a gaming server also provided with a game application program interface. The gaming machine as well as the gaming server comprises general gaming functions that are available as services for different games executed by the respective gaming specific client and server modules. In the client gaming machine there are general client gaming functions such as a graphical user interface, a cashbox, sound presentation means, buttons and data communications means typically devised for providing an interface between the player and the specific game as well an interface between the specific game and the server. In the gaming server there are general server gaming functions in particular a random number generator, a database and a gaming application program interface via which the gaming specific server module calls the general gaming functions upon a request from the client module.

[0015] During execution of a game the client game module of the game application program interacts with a player via input/output interfaces, typically a set of buttons and a display screen, and determines game parameters dependent on a set of logical rules for the game defined in the client game module and dependent on player input signals. Game parameters are for example symbols, positions, card values or card colours dependent on the nature of the game as well as a value for a current bet. At game stages that involve the determining of a random outcome, selected game parameters are communicated together with a request to the server game module. The server game module is executed and thereby determines a random outcome dependent on a set of logical rules that define conditions for outcomes, dependent on said selected game parameters and dependent on a random number that the server game module receives on request from the random number generator provided in the server. The determined outcome, typically comprising a win or lose outcome and possibly a prize, is communicated back to the client game module which after receipt determines and outputs a game presentation dependent on the outcome. The bet values and the prize values are preferably further dependent on configuration parameters that are centrally pre-settable and made available to the game application program. In this manner the game logics is separated from security sensitive, reliability sensitive and configuration related functions and information that are provided in the central server.

[0016] In accordance with an aspect of the invention the game application program modules are embedded behind the application program interfaces that provide a well defined interface for communication with functions external from game application program. The gaming machine is for this purpose provided with a client game application program interface devised for communication with the game application program module. General client gaming functions, e.g. the mentioned input/output interface, are provided in the client gaming machine and are available for the client game module via the client game application program interface. Similarly, the gaming server comprises general server gaming functions, e.g. the mentioned random number generator and a database, and a server game application program interface via which the server game module communicates with the general server gaming functions.

[0017] In accordance with another aspect of the invention, the game application program interfaces comprise means for managing the respective game modules. These game manager functions comprise configuration handling and provisions of data structures that carries game parameters, game requests and game responses.

[0018] In accordance with yet another aspect of the invention, the system is configured to record in the server database, inputs to the game software during execution. More specifically, at least user inputs and all random numbers that occur during the execution of a game are stored in the server database with an associated game session identity code. In this context, the stored user input should be understood to mean a selection of the actual user input, the received user input signal, a representation of the user input or the response by the game application program on a user input. This enables the reconstruction of a game session as well as verification of correctness as well as statistics and monitoring functions.

BRIEF DESCRIPTION OF ACCOMPANYING DRAWINGS

[0019] The inventive concept is further explained by means of examples and in conjunction with the accompanying drawings, in which:

[0020] FIG. **1** shows a schematic outline of a client-server based gaming system according to an embodiment of the invention.

[0021] FIG. **2** shows a flow chart of a simple example of a gaming application.

[0022] FIG. **3** shows a variety of the gaming system in accordance with the invention.

[0023] FIG. **4** shows a schematic view of different phases of a game.

[0024] FIG. **5** shows a schematic sequence diagram.

[0025] FIG. **6** shows a schematic illustration of an embodiment of a game development tool.

DETAILED DESCRIPTION OF EMBODIMENTS OF THE INVENTION

[0026] The Figures illustrate the configuration of a gaming system as well as a method for operating a gaming application, a method for developing a gaming application and a computer program product in accordance with the invention.

[0027] FIG. **1** shows schematically a client and server based computerised gaming system with a gaming machine **2**, herein also called a video lottery terminal, set up as a client gaming machine **2** and a gaming server **4** that are communicatively coupled. The gaming machine **2** and the gaming server **4** are provided with data processors, memory means, data communications interfaces, control programs, user input/output interfaces etc. in a per se well known manner. Different functions and features that are specific for the present invention are preferably realised by means of software computer program code executed on data processing means comprised in the server and in the client respectively, or by means of specifically designed electronic components, or by means of combinations of software and electronic components. In the example of FIG. **1** there is only a single client gaming machine but of course a number of client gaming machines can be and is normally connected to a server. In this context a server generally means hardware and software units in a central system that comprises and provides server functions, database functions and other centralised functions to connectable client gaming machines.

[0028] The server **4** is provided with a game application program interface, in short called server game API **6**, enabling communication between a server module of a specific game application program **8** and general server gaming functions **10**, **12**, **14**, **16** installed on the server. The general server gaming functions are provided to be available for any specific game application program independently of the specific game content. These general server gaming functions are typically critical functions such as a database **10**, a random number generator **12**, an account service function **14**, a log service function **16**, or other functions that beneficially are shared and used by different specific game application programs.

[0029] The client gaming machine **2** is also provided with a game application program interface, in short called client game API **20**, enabling communication between a client game module **18** of the specific game application program and general client gaming functions **22**, **24**, **26**, **28** installed on the client gaming machine **2** and used by different client game modules. The general client gaming functions are designed for assisting in implementing and executing a specific game on the client gaming machine **2** and are available for the client game module **18**. These general client gaming functions are in different embodiments a selection of a graphical user interface GUI **22**, a cashbox function **24**, a sound function **26**, user input interface function, for example buttons, **28**, data storage **29**, a printer **3**, a bar code reader **33** and other functions that are related to the performance of a game. The client game module **18** is communicatively coupled to the corresponding server game module **8** for communicating requests **9** and responses **11** in order to utilize the general gaming functions provided in the server. For each game a message protocol for communication between the client module and the server module is generated, the protocol is for example based on XML and is shared by the client and the server.

[0030] A specific game application program in accordance with the invention thus comprises a server game module **8** and a client game module **18** that communicate either directly or via an application program interface on the client side and the server side respectively as shown in FIG. **1** and FIG. **3**. The client game module **18** uses a selection of general client gaming functions that are available in the client gaming machine, whereas the server module **8** uses a selection of general server gaming functions **10**, **12**, **14**, **16** that are commonly used by different game applications and that are provided and available centrally in the server **4**.

[0031] In accordance with an embodiment of the invention, interruption or abnormal termination of a player session during playing a game is handled by means of a reconnect function. The reconnect function is managed by storing the previous game configuration in the database at the beginning of each game round. All user interactions and inputs that occur during the game round as well as the random numbers that were generated are stored in the server database. When a player session is terminated abnormally a reconnect voucher is created by the client gaming machine and output to the user. The voucher comprises account balance information and a voucher identity code used to identify the interrupted game. The player can thereafter request a reconnect by inputting the voucher to the same or another gaming machine. The game is initialised with the game session properties that were stored at the beginning of the round and retrieved from the database by means of the voucher identity code. Thus the reconnect function enables the player to continue the game at the same stage as when interrupted.

[0032] FIG. **3** shows a more detailed view of the configuration of a client and a server in a gaming system in accordance with an embodiment of the invention and similar to that of FIG. **1**. In the gaming server **4** the server game module **8** is embedded behind an application program interface called server game API **6** through which all communication of the server game module **8** takes place. The gaming server **4** further comprises a server application program interface in short called server API **40** through which all communication with the general server gaming functions **10**, **12**, **14**, **16** from the part of the server game API **6** as well as from the part of other server functions and external communication takes place. The server **4** is further provided with a server reconnect handler **32** that in a preferred embodiment is integrated with the server game API **6**. The server reconnect handler **32** comprises functional means devised to manage game session information relating to a reconnect identity code and devised to communicate inputs to and outputs from the server game module **8** for the purpose of re-executing the game session up to the point of interruption. This functionality realises a replay function and substitutes the client functionality during reconstruction of a game session.

[0033] The gaming server **4** is also provided with further server function modules, in the exemplifying embodiment more specifically comprising a client handler **36** that is communicatively coupled to the server API **40**. The client handler **36** manages, inter alia handles and serves, communications and functions of the client **2** other than the specific game applications. As illustrated in the drawing with a double arrow, communications with the client gaming machine **2** takes place via the server API **40** and a similar client API **38** provided in the client gaming machine **2**.

[0034] In a manner similar to that described above, the communication with the general client gaming functions is carried out via the client API **38**. The client gaming machine **2** comprises a client control module **34** that controls communications and general functions of the client gaming machine other than the specific game applications and communicates via the client API **38**. In the gaming client **2** the client game module **18** is, similar to the configuration of the server, embedded behind an application program interface called client game API **20** through which all communication of the client game module **18** takes place. The client **2** is further provided with a client reconnect handler **30** that in a preferred embodiment is integrated with the client game API **6**. The client reconnect handler **30** comprises functional means devised to manage game session information relating to a reconnect identity code and devised to communicate inputs to and outputs from the client game module **8** for the purpose of re-executing the game session up to the point of interruption. This functionality realises a replay function and substitutes the client functionality during reconstruction of a game session. In one embodiment, the replay is executed in the server after having retrieved stored game session information from the database to the server reconnect handler. In another embodiment, the replay is executed in the client after having retrieved the stored game session information from the database and communicated said information to the client reconnect handler. In yet another embodiment, a part of the replay is executed in the server and a part is executed in the client. The latter variety allows for replaying overlapping or entire parts of the replay in the server and in the client, and to verify that the respective replay execution is correct.

[0035] FIG. **2** shows schematically a simple example of a portion of a gaming application in accordance with the invention, more particularly a flip coin game of chance. The game is run by executing the client game module **18** and the general client gaming functions of the flip coin gaming application in a client gaming machine in step **202**. In step **204** the player is presented a message asking the player to bet on heads or tails. The player places a bet **206** and a result is calculated in **208**. Step **208** involves the client game module **18** sending a request to the server game module **8** to generate an outcome of the game. The server game module in its turn calls the random number generator **12** and receives a random number in return. The server game module calculates an outcome according to predetermined rules for the game and dependent on the returned random number. Thereafter, a response with the outcome Win or Lose is communicated back to the client game module. If the outcome is Lose **210** the player is presented a message showing that player lost **212**, and the game is ended in **214**. If, on the other hand the outcome is Win **216** the player is presented a message asking player to collect the prize or double a bet again **218**. If the player inputs a request to Double **224**, a new result is calculated in **208** in the above manner. If on the other hand the player inputs a request to Collect **219**, the prize, usually in the form of cash or credit money, is paid to the player and the game ends in **222**. The payout of a prize again preferably involves requesting services from the server game module and for example utilizing the general server gaming functions account function **16** and database function **10**.

[0036] In accordance with the inventive concept, the client game module comprises computer program code portions devised to control the data processing system in the client gaming machine to determine game parameters in response to player input signals and dependent on logical rules for the game defined in said client module. The server module comprises computer program code portions devised to control the data processing system to determine an outcome dependent on conditions for the game defined in said server module, dependent on said game parameters being communicated from the client module and dependent on a random number received from the random number generator **12** that is provided in the server and available for the server game module.

[0037] The client module and the server module of the game application are thus isolated from functions other than game specific features by means of game API:s. For the purpose of further isolating and de-coupling the game application program from sensitive functions a previous game configuration is stored in the server database at the beginning of each game round. All user interactions and inputs that occur during the game round as well as the random numbers that were generated are stored in the server database. This enables reconstruction and verification of interrupted game sessions.

[0038] FIG. **4** illustrates schematically the lifecycle of a game in accordance with an embodiment of the invention drawn in relation to a timeline **424**. A player session **401** is initiated by a player by inputting start commands via a user interface of the gaming machine. The player initiates a game and inputs a bet in terms of a monetary value by means of some kind of payment method such as coins or an account transaction and thereby starts a first game session **402**. The game session progresses in discrete steps herein called game rounds and exemplified with a first round **404** and a second round **410**. Each round in turn progresses in discrete steps called game phases. So for example, round **404** comprises three game phases **406**, **407** and **408**. The transition between game phases is driven by game round events which in different embodiments may have different content and different triggering mechanisms.

[0039] A game round event is triggered by an input that starts the generation of a set of associated elements of critical game session data that defines a game result preferably comprising the current bet value, a current generated random number and a current win value. The game round event would usually be triggered by a player making an input through an I/O interface such as a push button that conveys game commands like "Deal cards!" in a poker game.

[0040] In an embodiment of the invention, a game round event is triggered usually by an input from the player and game parameters that define a game state are determined by means of logical rules realised in the client game module. The client game module contacts the server game module with a request in response to the triggering of a game round event. The server game module executes the request and creates game result defining game session data for the current game phase. This data is stored in the server database and a response comprising the game result defining data is transmitted to the client whereupon a presentation of the game result is output to the player and the game phase is ended. The presentation of the game result to the player typically comprises updating a screen display of the gaming machine.

[0041] At the beginning of each game phase, in FIG. **4** illustrated with time indicators A, B, C, D, the gaming machine, e.g. the client game module, is set in a waiting mode waiting for input from the player. When a game round event is triggered by the player, the gaming machine, e.g. the client game module and the server game module executes the game rules, moves the game process to the beginning of the next phase, stops and again goes into the waiting mode to wait for player input. From a gaming system macro perspective the execution of a game progresses in discrete steps where the game phases are the smallest units of execution.

[0042] FIG. **5** shows a schematic sequence diagram that illustrates communications between a client game module **502** of a gaming machine and a remote data storage and server game module **504**, as well as executed steps occurring during a game phase depicted in relation to time lines **520**. The current game phase starts at **506** Time A and a waiting mode is entered and lasts until there is a game round event **512** comprising transmitting a request **512** from the gaming machine/client game module **502**. The request is received at the remote data storage/server game module **504** whereupon in step **514** the request is executed and resulting game session data is stored in a database residing in the data storage. In the simpler embodiment not involving a client-server configuration, a procedure call to generate the game result defining data would correspond to the request. The execute and storage step **514** is completed at **508** Time A1 and thereafter in step **516** a response comprising resulting game session data is transmitted from the remote data storage/server game module **504** to the gaming machine/client game module **502**. After receipt of the response the gaming machine/client game module **502** presents an output, e.g. an animation of a game outcome via the display screen dependent on the result of the execution in the game round event, which is terminated at **510** Time B with a transition into the next game phase. The execution and storage step **514** is treated to be atomic in the sense that either it is completed or it is entirely failed. A game phase is considered to be complete when the execution and storage step **514** has been completed, i.e. at **508** Time A1, whether or not the following steps **516**, **518** up to 510 Time B have been completed when an interruption occurs.

[0043] The communication between the gaming machine and the server as well as other steps that are performed for realising the invention are described in the following exemplifying embodiment based on the client-server configuration. Reference is made to FIG. **1-4**. The numbered list below is merely for reference purpose and does not necessarily mean that the steps are performed in a sequence corresponding to the indicated numerical order.

[0044] 1. A player initiates a player session **401** on the client gaming machine **2** by inputting a start command to the client game module **18** via an I/O-interface (**22**, **24**, **28**) which may be a traditional button or a button field on a touch screen. The initiation of a player session would preferably also comprise a monetary transaction for bets in the game, for example by the player adding a coin to a cash box **24** or by means of an account transaction.

[0045] 2. The client reconnect handler **30** comprised in the client gaming machine **2** transmits a request for the reservation of a reconnect identity code together with a client identification code for identification of the specific gaming machine to the server. This request is received by the server reconnect handler **32** similarly comprised in the gaming server **4**.

[0046] 3. The request is executed by the server reconnect handler **32** whereby a reconnect identity code is generated and stored associated with the client identification code in a database **10**.

[0047] 4. A player session identity code is generated and associated with the reconnect identity code in the database **10**, and a player session **402** is established.

[0048] 5. The reconnect identity code is transmitted to the client reconnect handler **30** and is stored in local data storage **29** in the client gaming machine **2** for the purpose of enabling communication of the reconnect identity code to the player.

[0049] 6. A selected game is started by the player inputting a game start command to the game client module **18** via the game application program interface **20** of the client gaming machine **2**, and a request to start a game session is transmitted to the gaming server **4**.

[0050] 7. A game session identity code is generated and stored associated with the player session identity code, a game session **402** is established and a game phase **406** of a game round **404** is entered.

[0051] 8. The player triggers a game round event by giving a game related input to the client game module **18** whereupon a request for a service is transmitted to the server game module **8**. An input signal from the player is received by the client game module and a function determines game parameters dependent on logical rules based on the nature of the game and generates a request to send to the server game module. The client game API provides an instance of a data structure, preferably in the shape of a data object, for the client game module to access and store game context parameters comprising said determined game parameters. The determined game context parameters and the generated request are then communicated to the game server module.

[0052] 9. The server game API receives the game context parameters and the request, and provides an instance of a data structure, preferably in the shape of a data object, for the server game module to access and store. The data structure comprises allocations for game context parameters, the game request and the game response. The game response is the response of the server game module of to the game request generated by the client game module. The server game module thus accesses and receives or reads the game parameters and the request from the data structure. The request is executed by the server game module **8** with the aid of the service functions of the server. The execution of this request would typically comprise the generation of a random number RNG and the determination of an outcome dependent on the RNG. Outcome parameters are thus determined and a response is generated and stored in the game response allocation of the data structure. The game response and possibly also the game context parameters are then communicated back to the client game module. Possibly also game configuration parameters are communicated to the client game module with the response.

[0053] 10. An embodiment of the invention comprises a cashing mechanism for intermediary storage of a game state. The game state information would comprise game context parameters, configuration parameters, previous requests, and previous responses. This feature is useful in multistate games.

[0054] 11. A consistency or error check may occur in the server before the response is sent to the client. This would for example include calculating bet values and win values and compare with a resulting monetary balance of the player.

[0055] 12. Execution steps that are performed by the server game module **8** for each request as well as results and outcomes of the execution make up game session data, i.e. information that applies to the currently ongoing game session. A subset of the game session data is the result of a game round event and applies to the current game phase. A selection of these game session data are compiled and cashed, i.e. temporarily stored in data storage of the server **8**. Game session data generally comprises game parameters generated by the client game module s well as parameters generated by the server game module. The selection may vary in different execution cases and embodiments, and would preferably comprise: the bet value, the random number and the win value that are valid for the current game phase. The selection of game session data may also comprise optional pieces of information regarding the sequence of events called event history, each request and response, a pot at stake, the request, the response to the client gaming machine, game configuration information and a status indicator devised to indicate whether the game session has been completed or interrupted e.g. indicating last event=true/false.

[0056] 13. The selection of game session data is stored in the database **10** and is transmitted with a response to the client gaming machine **2**. The received selection of game session data is cashed, i.e. temporarily stored in data storage of the client gaming machine **2**.

[0057] 14. The response, including information about the outcome, is received by the client game module. The outcome of the game round event is presented to the player for example via image output on a presentation screen of the client gaming machine **2** and possibly a win value being added to a monetary balance of the player.

[0058] 15. The steps 8-12 are normally repeated until a game round is ended for example by the fact that a bet monetary balance has been consumed, or the player selects a new game to play.

[0059] 16. If an interruption occurs, for example triggered when the client looses contact with the server, then the reconnect identity code for the current game execution instance is in the server database associated with a status indicator indicating that the game execution has been interrupted.

[0060] 17. The reconstruction of an interrupted game can be implemented in various manners. One embodiment comprises of the following steps. With the reconnect identity code as a key, the associated game session data is retrieved from the server database under the control of the server reconnect handler **30**. The server reconnect handler **30** uses the retrieved game session data as input to the server game module **8** and generates the last response from the server game module that should have been transmitted to the client gaming machine unless the interruption had occurred. In different embodiments Reconnect information comprising game session data is compiled and transmitted to the client reconnect handler **30** of the client gaming machine. In one embodiment this reconnect information comprises the game identifications for the server game module and the client game module, game session data including an initial monetary balance, all requests and all responses of the event history. In the client gaming machine, the client game module is initiated and the game is executed by the client reconnect handler **30** using the game session data as input up to the last completed game phase before the point of interruption, called the reconnect target point. In contrast with the normal execution of a game, the requests that are generated in the reconnect execution are discarded and after each request the game is presented with the corresponding response from the game session data. Since the client reconnect handler **30** has access to all the requests as well as the responses to the requests it is enabled that a check of a proper reconstruction of the game session can be performed. Preferably, the game is executed up to the reconnect target point without presenting the intermediate results to the player in order to speed up the execution and avoid confusing the player.

[0061] 18. After the last event and thereby the last completed game phase has been executed, the corresponding result and state of the game is presented to the player via the graphical user interface and the game enters a waiting mode waiting for the next input from the player.

[0062] The invention has the effect that the game application program and its developers do not have to and is not allowed to deal with other functions than the logical rules for the game. These other functions comprise input/output interfaces, functions relating to statistics parameters and configuration parameters, as well as problems and functions relating to the interruption, reconnect and reconstruction of a game. Instead the framework around the game application program controls its execution and the input for reconstruction of a game. The invention has been described by way of exemplifying embodiments, but naturally there are various manners of realising the invention within the scope of the claims.

Game Development Tool

[0063] An embodiment of the invention comprises a method and a computer program product for realising a game development tool. A method of developing a computer based gaming application in accordance with the inventive concept comprises providing software adapted to simulate a client gaming machine and a server adapted to simulate a gaming server. The client and the server are adapted to be executed in a common data processing system and are preferably implemented in a computer program product as computer executable code stored on a data carrier or a memory. This computer program product embodies a game developer's kit enabling game developer to develop games for a gaming system in accordance with the inventive concept and provides a simulated game system configuration as described above but executable on a local stand alone computer.

[0064] This aspect of the invention thus comprises providing in the server a general gaming function that is executable in response to a call; providing in the server a gaming application program interface enabling communication with said general gaming function; and providing in the client a gaming application program interface enabling communication with said client. By means of the gaming application program interfaces the game developer generates and installs in the server a server module of a specific gaming application program; generates and installs in the client a client module of the specific gaming application program; generates and installs in the client gaming specific functions that are executable in response to a call from the client module via the gaming application program interface of the client.

[0065] The client that thus simulates a gaming machine that operates as a container for the games, the graphics and the audio subsystems, respectively, and is responsible for loading, start-up, shut-down and communication with server and the player. Utility libraries, for example in XML or string format are provided as tools for generating reusable functions such as buttons, dice, movements of graphical items and the like.

[0066] In the development process the developer uses general gaming functions provided in the simulated server, for example a simulated database and a random number generator. The database is simulated by reading and writing XML files. Dynamic data that is normally, i.e. in real system operation, stored in the database, such as player sessions and player account balance, is simulated in the server. The random number generator preferably comprises hardware, e.g. in the shape of a hardware based random number generator to be connected to a serial port of the developer's computer and software components. In the possible absence of a hardware random number generator, a random number can be emulated.

[0067] After having developed the program functionality the developer compiles the generated server module, the generated client module and the generated gaming specific functions of the specific gaming application program into a data format that is executable on a gaming system in accordance with the inventive concept.

[0068] The game development tool is devised to emulate all the client and server functions that are described above in the description of client and server functionality. FIG. **6** schematically illustrates an overview of functional units and mechanisms that are emulated in the game development tool **602**. Thus, an embodiment of a computer program product for developing a computer based gaming application for a gaming system comprises computer program code portions devised to a control a data processing system to realise number of emulators. A client gaming machine emulator **604** is devised to emulate a client gaming machine having means for executing a client module **610** of a game application program in the client gaming machine, and means for determining game parameters in response to player input signals and dependent on logical rules for the game defined in said client game module **610**. The game development tool **602** further comprises a gaming server emulator **612** devised to emulate a gaming server having means for executing a server game module **616** of said game application program in the gaming server in response to said game parameters being communicated to said server module, and means for determining an outcome dependent on conditions for the game defined in said server game module **616**, dependent on said communicated game parameters and dependent on a random number. The random number is preferably generated in the gaming server emulator **612** by means of an emulated random number generator comprised in a general server gaming functions emulator **618**. Similarly, a database emulator is comprised to emulate a database of the emulated server **612**.

[0069] The computer program product further comprises a client input/output interface emulator **606** devised to emulate an input/output interface of a client gaming machine for communicating with a player as well as other general client gaining functions (not shown) as previously described. Since the structure coincides with the client and server as described above numeral reference is also made to FIG. **1** and FIG. **3**. Means are further provided for emulating the following functions:

[0070] a client control module (**34**) devised to control the execution of functions and communications of the emulated client gaming machine **604**, said client control module (**34**) being devised to control the execution of said client game module (**18**, **610**) of a game application program provided for the emulated client gaming machine (**604**);

[0071] general client functions (**22**, **24**, **26**, **28**) providing input and output functions for communication between a user and functions of said gaming system;

[0072] a client game application program interface (client game API, 20) enabling communication with a client game module of a game application program;

[0073] a client application program interface (client API, 38) enabling communication between functions within the emulated client gaming machine **604** and between the emulated client gaming machine **604** and the emulated gaming server **612**;

[0074] general server functions (**618**) providing a random number generator (**12**) and a database (**10**) in the emulated server (**4**, **612**);

[0075] a server game application program interface (server game API, **6**, **614**) provided in the emulated server (**4**, **612**) and enabling communication with the server module (**8**, **616**) of said game application program;

[0076] a client handler (**36**) devised to manage communications and functions of the emulated client gaming machine (**2**, **604**);

[0077] a server application program interface (server API, **40**) provided in the emulated server (**4**) and enabling internal communication between functional units (**6**, **8**, **10**, **12**, **32**, **36**) of the emulated server (**4**, **612**) and external communication with functional units of the emulated client gaming machine (**2**, **604**).

[0078] The effect of the game development tool is that a game developer can work, develop and test his game application program in an environment that reproduces a real runtime environment for the game.

1. (canceled)

2. A method for operating a gaming application in a computer based gaming system having a client gaming machine connected to a gaming server, the method comprising:

executing a client module of a game application program in the client gaming machine;

determining game parameters in response to player input signals and dependent on logical rules for the game defined in said client module;

executing a server module of said game application program in the gaming server in response to said game parameters being communicated to said server module;

determining outcome parameters dependent on conditions for the game defined in said server module, dependent on said communicated game parameters and dependent on a random number generated in the gaming server.

3. The method of claim **2**, wherein said outcome parameters are communicated to the client game module and output signals dependent on the outcome parameters are generated and output to realize a presentation of the outcome to the player.

4. The method of claim **2**, wherein all communication with the client game module is carried out via a client game application program interface.

5. The method of claim **2**, wherein game context parameters are provided to the client game module by means of said client game application program interface.

6. The method of claim **2**, wherein game configuration parameters are provided to the client game module by means of said client game application program interface.

7. The method of claim **2**, wherein all communication with the server game module is carried out via a server game application program interface.

8. The method of claim **2**, wherein game context information is provided to the server game module by means of said server game application program interface.

9. The method of claim **2**, wherein game context information comprises said determined game parameters.

10. The method of claim **2**, wherein said game parameters comprises a bet value.

11. The method of claim **2**, wherein said outcome parameters comprises a win value.

12. The method of claim **2**, wherein game configuration parameters are provided to the server game module by means of said server game application program interface.

13. A gaming system for operating a gaming application, the system having

a computer based client gaming machine connected to a gaming server, comprising computer based data processing means devised to:

execute a client module of a game application program in the client gaming machine;

determine game parameters in response to player input signals and dependent on logical rules for the game defined in said client module;

execute a server module of said game application program in the gaming server in response to said game parameters being communicated to said server module;

determine an outcome dependent on conditions for the game defined in said server module, dependent on said communicated game parameters and dependent on a random number generated in the gaming server.

14. The gaming system of claim **13**, and further comprising:

general server functions providing a random number generator and a database in the server;

a server game application program interface provided in the server and enabling communication with the server module of said game application program;

a client handler devised to manage communications and functions of the client gaming machine;

a server application program interface provided in the server and enabling internal communication between functional units of the server and external communication with functional units of the client gaming machine;

general client functions providing input and output functions for communication between the user and functions of the gaming system;

a client game application program interface provided in the client gaming machine and enabling communication with the client game module of said specific game application program;

a client control module devised to control communications and functions of the client gaming machine.

15. The gaming system of claim **13**, and further comprising:

a server game module of a game application program provided in the server and devised to execute functions of a game;

a client game module of said game application program provided in the client gaming machine and devised to execute functions of same game.

16. The gaming system of claim **13**, and further comprising means devised to provide said server game module and/or said client game module with a data structure for storing and accessing game context parameters.

17. The gaming system of claim **13**, and wherein said game context data structure is an instance of a data object provided by said game application program interfaces respectively.

18. The gaming system of claim **13**, and further comprising means devised to provide said server game module and/or said client game module with a data structure for storing and accessing game configuration parameters.

* * * * *