Claims

1. A method for providing a service to a client over a network, comprising:

receiving a connection request from the client for controlling a virtualized computer service available on a specific virtual machine, the connection request including a plurality of connection parameters that define connection requirements of the client, the connection request received at a virtual machine that executes an image processor algorithm;

interrogating the connection parameters using the image processor algorithm to identify the specific virtual machine providing the virtualized computer service that satisfies the request from the client;

~~accessing~~reading framebuffer data for the specific virtual machine directly from a predetermined location in virtual memory through a hypervisor, the reading occurring without processing by a graphics processor of the hypervisor or underlying physical hardware, to generate a graphical output that represents an image, the framebuffer data ~~being read from the predetermined location in virtual memory, the framebuffer data~~ itself defining the graphical output ~~representing~~that represents the an image associated with a virtual machine display of the specific virtual machine;

processing the framebuffer data into a plurality of image data packets using the image processor algorithm based on the plurality of connection parameters; and

transmitting the image data packets to the client for presenting on a display device associated with the client, the image data packets being specific for the client and represents the image of the virtual machine display.

2. The method of claim 1 , wherein the image data packets includes packaging information related to image processor algorithm and packaging information related to networking.

3. The method of claim 1 , wherein processing the framebuffer data further includes,

analyzing the framebuffer data of the specific virtual machine to determine context;

associating appropriate contextual information to the framebuffer data of the specific virtual machine based on the analysis; and

storing the framebuffer data of the specific virtual machine along with the contextual information.

4. The method of claim 3, further includes,

scanning the framebuffer data of the specific virtual machine periodically;

determining areas of change by comparing scanned framebuffer data against the stored framebuffer data for the specific virtual machine; and

storing the areas of change in a time-based storage array associated with the specific virtual machine for analysis and transmission, wherein storing of the areas of change includes storing of the contextual information associated with the areas of change.

5. The method of claim 4, wherein scanning the framebuffer data further includes,

interrogating a sync-signal associated with the framebuffer data of the specific virtual machine, the sync-signal identifying status of a refresh of the framebuffer data of the specific virtual machine; and

performing the scanning of the framebuffer data of the specific virtual machine based on the interrogation of the sync-signal.

6. The method of claim 5, further includes,

setting the sync-signal prior to a refresh of the framebuffer data for the specific virtual machine; and

re-setting the sync-signal upon completion of the refresh of the framebuffer data for the specific virtual machine.

7. The method of claim 5, wherein the periodic scanning of the framebuffer data at the virtual machine is conducted at one of a refresh frequency of the framebuffer data at the virtual machine or a frequency driven by the connection parameters associated with the client.

8. The method of claim 5, wherein the scanning of the framebuffer data at the virtual machine further includes,

receiving a notification informing of a change in the framebuffer data of the specific virtual machine, the notification received from a graphics driver associated with the specific virtual machine ; and

scanning the framebuffer data of the specific virtual machine based on the notification.

9. The method of claim 4, wherein processing the read framebuffer data further includes,

analyzing the time-based storage array to determine areas of high activity within the framebuffer data;

determining contiguous areas of activity within the areas of high activity of the framebuffer data;

grouping transmission technique friendly traits within the contiguous areas of activity into groups of framebuffer data within the time-based storage array; and

transmitting the groups of contiguous areas of activity from the time-based storage array in a time-based framebuffer data stream to the client based on the connection parameters associated with the client, the transmission of the groups of contiguous areas of activity includes derived contextual information associated with the framebuffer data.

10. The method of claim 9, wherein the transmitting of the image data packets further includes,

determining a format that is compatible with the display device associated with the client based on the connection parameters;

identifying one or more conversion techniques based on the context associated with the framebuffer data, the one or more conversion techniques used in converting the framebuffer data to the format compatible with the display device associated with the client;

converting the framebuffer data into image data packets that comply with the identified format using the one or more conversion techniques, wherein the image data packets include the contextual information associated with the framebuffer data; and

transmitting the formatted image data packets to the client for rendering at the display device of the client.

11. The method of claim 9, further includes,

examining one or more characteristics associated with an area of change of the framebuffer data;

extracting a portion of the framebuffer data related with the area of change based on the one or more characteristics using the identified conversion technique; and

transmitting the extracted portion of the framebuffer data to the client as an out-of-band stream using the identified conversion technique, wherein the out-of-band stream including the contextual information of the framebuffer data,

wherein the extracted portion of the framebuffer data is a video clip.

12. The method of claim 9, further includes,

dynamically detecting changes to connection parameters associated with the client, the changes to connection parameters identifying a change in the display device associated with the client;

dynamically re-sampling and re-sizing the framebuffer data of the specific virtual machine into appropriate image data packets based on the changes to the connection parameters detected at the client, the dynamic re-sampling and re-sizing of the framebuffer data enabling rendering of the framebuffer data at the display device of the client.

13. The method of claim 9, further includes,

detecting a plurality of display devices associated with the client, wherein each of the plurality of display devices is of a distinct type;

determining a format compatible with each type of display device of the client;

generating multiple sets of image data packets using the one or more conversion techniques, each set of image data packets generated by converting the framebuffer data using the format that is compatible with a specific type of display device associated with the client, wherein each set of image data packets includes contextual information associated with the framebuffer data; and

transmitting each set of the image data packets to the client for rendering at the respective display devices of the client.

14. The method of claim 1 , wherein each image data packet includes data that is assigned to define and render a specific portion of the image of the virtual machine display at the client and wherein the specific portion of the image is defined by at least one pixel.

15. The method of claim 1, wherein the framebuffer data is created by an operating system of the virtual machine and wherein access to the framebuffer data is provided by a hypervisor, the hypervisor providing an interface between the client and the virtual machine.

16. A method for providing a service to a client over a network, comprising:

receiving a connection request from the client for controlling a virtualized computer service, the connection request including a plurality of connection parameters that defines the connection requirements of the client, the connection request received at a virtual machine executing an image processor algorithm;

interrogating the connection parameters using the image processor algorithm to identify a specific service layer providing a type of service requested and a specific virtual machine within the specific service layer for providing the virtualized computer service that satisfies the connection request from the client, wherein each service layer hosts a distinct type of service;

connecting to the specific service layer that provides the specific virtualized

computer service;

~~accessing~~reading framebuffer data for the specific virtual machine directly from a predetermined location in virtual memory through a hypervisor in the specific service layer, the reading occurring without processing by a graphics processor of the hypervisor or underlying physical hardware to generate a graphical output that represents an image ~~framebuffer data being read from the predetermined location in virtual memory~~, the framebuffer data itself defining ~~a~~the graphical output that represents the ~~representing an~~ image associated with a virtual machine display of the specific virtual machine within the identified service layer;

processing the framebuffer data into a plurality of image data packets using the image processor algorithm based on the plurality of connection parameters; and

transmitting the image data packets to the client for presenting on a display device associated with the client, the image data packets being specific for the client and represents the image of the virtual machine display.

17. The method of claim 16, wherein each image data packet includes data that is assigned to define and render a specific portion of the image of the virtual machine display at the client and wherein the specific portion of the image is defined by at least one pixel.

18. The method of claim 16, further includes,

detecting a change in connection parameters at the client, the change in connection parameters defining a new virtualized computer service requested by the client;

disconnecting the connection to the identified virtual machine;

interrogating the changed connection parameters using the image processor algorithm to identify a specific new virtual machine providing the new virtualized computer service requested by the client;

establishing connection to the specific new virtual machine based on the interrogation of the changed connection parameters, the connection to the specific new virtual machine enabling access to a framebuffer data for the specific new virtual machine at a predetermined location in virtual memory associated with the specific new virtual machine;

accessing the framebuffer data for the specific new virtual machine from the predetermined location in virtual memory associated with the specific new virtual machine, the accessed framebuffer data read from the predetermined location in virtual memory for the specific virtual machine and processed into a plurality of image data packets for transmission using the image processor algorithm and presentation on a display device of the client, the image data packets being specific for the client and represents the image of

the specific new virtual display associated with the new virtual machine on which the new virtualized computer service is executing.

19. The method of claim 18, wherein disconnecting the connection further includes disconnecting from the identified service layer.

20. A method for providing a service to a client over a network, comprising:

providing an image processor algorithm on at least one virtual machine, the virtual machine with the image processor algorithm executing on a hypervisor on which a plurality of virtual machines are executing;

processing a connection request at the image processor algorithm of the at least one virtual machine, the connection request received from the client, wherein the processing of the connection request identifying a plurality of connection parameters defining connection requirements associated with the client and a specific virtual machine having a virtualized computer service that can satisfy the connection request;

~~accessing~~reading framebuffer data for the specific virtual machine directly from a predetermined location in virtual memory through the hypervisor, ~~the framebuffer data being read from the predetermined location in the virtual memory~~the reading occurring without processing by a graphics processor of the hypervisor or underlying physical hardware to generate a~~, the framebuffer data defining~~ graphical output ~~representing an~~that represents the image associated with a virtual machine display of the specific virtual machine;

processing the framebuffer data into a plurality of image data packets using the image processor algorithm based on the plurality of connection parameters, wherein the plurality of image data packets includes contextual information associated with the read framebuffer data; and

transmitting the image data packets with the contextual information to the client for presenting on a display device associated with the client, the image data packets being specific for the client and represents the image of the VM display.

21. The method of claim 20, further includes providing an image processor algorithm independently in a plurality of virtual machines, each virtual machine with the image processor algorithm configured to access and read corresponding framebuffer data of other virtual machines.

22. The method of claim 20, further includes providing an image processor algorithm independently in each of the plurality of virtual machines executing on the hypervisor, each

of the plurality of virtual machines with the image processor algorithm configured to access and read corresponding framebuffer data of other virtual machines.

23. The method of claim 20, wherein processing the read framebuffer data further includes,

analyzing the framebuffer data of the specific virtual machine providing the virtualized computer service;

determining context for the framebuffer data based on the analysis;

associating appropriate contextual information for the framebuffer data; and

generating a plurality of image data packets for the framebuffer data, each of the image data packets including at least a portion of the framebuffer data and the associated contextual information for the portion of framebuffer data.

24. A method for providing a user access to a computing profile, the computing profile defining a desktop for the user, the desktop providing access to applications and data of the user, the method comprising:

a) requesting connection to the computing profile through an Internet connection from a client;

b) identifying a virtual machine for the computing profile in response to the requested connection;

c) identifying framebuffer data for the identified virtual machine;

d) reading the framebuffer data directly from a predetermined location in virtual memory through a hypervisor, wherein the reading occurs without processing by a graphics processor of the hypervisor or underlying physical hardware to generate a graphical output that represents an image, the framebuffer data itself defining the graphical output that represents the image associated with a virtual machine display of the identified virtual machine;

e) processing the framebuffer data based on characteristics of the client from which the connection request is made, the processing producing image data packets;

f) forwarding the image data packets to the client, the image data packets being processed at the client and presented on a display of the client;

g) repeating (d) - (f) during a session, while exchanging backchannel communication between the client and the virtual machine.

25. The method of claim 24, wherein each image data packet includes data that is assigned to define and render a specific portion of the image of the virtual machine display at the client and wherein the specific portion of the image is defined by at least one pixel.

26. The method of claim 24, wherein processing the framebuffer data further includes,

analyzing the framebuffer data to determine context;

associating contextual information to the framebuffer data based on the analysis;

generating image data packets for the framebuffer data using one or more conversion techniques, the image data packets being compatible with the display device associated with the client.

27. The method of claim 24, wherein the backchannel communication between the client and the virtual machine includes data to one or more applications executing on the virtual machine.

28. The method of claim 24, wherein processing of image data packets at the client further includes removing packaging information and mapping the data to a specific portion of the display of the client.

29. A method for providing access to a session of a virtual machine (VM), comprising:

(a) obtaining access to the session of the VM;

(b) identifying framebuffer data for the VM;

(c) reading the framebuffer data directly from a predetermined location in virtual memory through a hypervisor, wherein the reading occurs without processing by a graphics processor of a hypervisor or underlying physical hardware to generate a graphical output that represents an image, the framebuffer data itself defining the graphical output that represents the image associated with a VM display of the VM;

(d) processing the framebuffer data based on characteristics of the client to produce image data packets;

(e) forwarding the image data packets, produced for the characteristics of the client, to the client;

(f) repeating (c) - (e) during a time when accessing the session of the VM while exchanging backchannel communication between the client and the VM.

30. The method of claim 29, wherein each image data packet includes data that is assigned to define and render a specific portion of the image of the virtual machine display at the client and wherein the specific portion of the image is defined by at least one pixel.

31. The method of claim 30, where in the characteristics of the client are defined by one or more connection parameters that include any one or more of location of the client, location of virtual service to connect, type of client, type of display, quality of connection, communication connection bandwidth, type of data requested, configuration of display associated with the client including resolution, bit depth, refresh rate.

32. The method of claim 30, wherein obtaining access to the session is through a service broker and an image processor algorithm executing on a VM running on a hypervisor, the service broker providing information associated with mapping of clients to the VM and the image processor algorithm providing API to access the framebuffer data associated with the VM.

33. The method of claim 32, wherein the image processor algorithm is executing independently on any one of the same VM running the session or on a VM different than the VM running the session.

34. The method of claim 32, wherein the image data packets at the client are processed to remove packaging information and map the data in the image data packets to specific portions of the display of the client.

35. A system for providing a service to a client over a network, comprising:

a) one or more virtual machines configured to run on a hypervisor, wherein at least one of the virtual machines is executing an image processor algorithm, the image processor algorithm includes logic to,

(i) detect a connection request from the client for controlling a virtualized computer service available at a specific virtual machine;

(ii) identify the specific virtual machine that provides the virtualized computer service by interrogating the connection request;

(iii) access and read framebuffer data of the specific virtual machine from a predetermined location in virtual memoiy through a hypervisor, the framebuffer data defining graphical output representing an image associated with a virtual machine display of the specific virtual machine;

(iv) process the read framebuffer data into image data packets, the image data packets including contextual information associated with the framebuffer data, the logic for processing is configured to,

scan the framebuffer data of the specific virtual machine to discern an image of a virtual machine display of the specific virtual machine including any changes in the framebuffer data of the specific virtual machine and to process the framebuffer data including any changes into a time-based storage array associated with the specific virtual machine;

(v) transmit the image data packets of the specific virtual machine to the client for presenting on a display device associated with the client,

wherein image data packets are specific for the client and represents the image of the virtual machine display.

36. The system of claim 35, further includes,

a) a directory service to provide mapping of clients to the one or more virtual machines and to authenticate information associated with the connection request;

b) a time-based storage array service to store the time-based storage array associated with groups of areas of activity and any changes to the groups of areas of activity at the image of the virtual machine display of the specific virtual machine over time; and

c) a compression technique service to provide compression/transmission techniques for packaging the framebuffer data, wherein the compression/transmission technique is used to provide contextual compression.

37. A system for providing a service to a client over a network, comprising:

a) one or more virtual machines configured to run on a hypervisor, wherein at least one of the virtual machine is executing an image processor algorithm alongside the other virtual machines, the image processor algorithm includes,

(i) a scan engine configured to,

(ia) detect a connection request from the client for controlling a virtualized computer service available on a specific virtual machine, the connection request identifying a plurality of connection parameters defining connection requirements of the client;

(ib) interrogate the connection parameters to identify the specific virtual machine that provides the virtualized computer service;

(ic) interact with a hypervisor to directly access framebuffer data of the specific virtual machine;

(id) scan the framebuffer data of the specific virtual machine to discern an image of a virtual machine display of the specific virtual machine including any changes in the framebuffer data of the specific virtual machine and to process the framebuffer data including any changes into a time-based storage array associated with the specific virtual machine;

(ii) a context engine configured to,

(iia) determine any contiguous areas based on characteristics associated with the framebuffer data and to determine context associated with the framebuffer data; and

(iii) a logic engine configured to

(iiia) analyze the framebuffer data;

(iiib) process the framebuffer data to integrate contextual information associated with the framebuffer data based on the determined context; and

(iiic) transmitting the processed framebuffer data including relevant contextual information of the specific virtual machine to the client for presenting on a display device associated with the client,

wherein the processed framebuffer data being specific for the client and represents the image of the virtual machine display.

38. The system of claim 37, wherein the image processor algorithm further includes a heat engine with a search algorithm that is configured to interact with the time-based storage array to search for any activity within the framebuffer data, identify areas of activity and the level of activity within the framebuffer data based on the search, and to ignore areas of non-activity.

39. The system of claim 37, wherein the scan and the heat engine are further configured to receive user interactions through a backchannel, wherein the user interactions define areas of likely change within the framebuffer data of the specific virtual machine.

40. The system of claim 37~~39~~, further includes a directory service and a policy and authentication service, the directory service being a repository of mappings of a plurality of clients to the one or more virtual machines and the policy and authentication service configured to ascertain policies for authenticating connection request from the client.

41. The system of claim 40, wherein the image processor algorithm further includes a connection engine configured to,

interact with the client to obtain connections parameters and to determine ongoing quality of connection associated with the client;

interact with the policy and authentication service to authenticate connection parameters received from the client and to determine one or more policies for handling the connection request;

interact with the logic engine to provide up-to-date connection parameters associated with ongoing quality of connection;

interact with the directory service to identify the specific virtual machine that can provide the requested virtualized computer service to the client; and

interact with the scan engine to direct the scan engine to the appropriate framebuffer data for the specific virtual machine.

42. The system of claim 41, wherein the image processor algorithm further includes an output engine to transmit the processed framebuffer data as image data packets that are sized optimally for efficient transmission to the client for presenting in response to the request, wherein the size of the image data packets are based on the connection parameters.

43. The system of claim 37, wherein the image processor algorithm is executable independently on one or more of a plurality of virtual machines running on the hypervisor, where the image processor algorithm is used for accessing and reading corresponding framebuffer data of other virtual machines running on the hypervisor.

44. The system of claim 37, wherein the image processor algorithm is executable independently on each one of the virtual machines running on the hypervisor, where the image processor algorithm of each of the virtual machines is used for accessing and reading corresponding framebuffer data of other virtual machines running on the hypervisor.

45. The system of claim 37, further includes a plurality of service layers, each of the plurality of service layers provide a specific type of service requested by the client, each of the plurality of service layers includes one or more virtual machines, the image processor algorithm running on the virtual machine configured to access the framebuffer data of the virtual machines in each of the plurality of service layers.

46. The system of claim 45, wherein at least one of the virtual machines in each of the plurality of service layers is independently executing the image processor algorithm.

47. The system of claim 37, wherein the client includes logic configured to process the image data packets received from the virtual machine executing the image processor algorithm, where the processing of the image data packets includes,

removing packaging information from the image data packets; and

mapping the data to a specific portion of the display of the client.

48. An image processor algorithm embodied on a non-transitory computer readable medium, the image processor algorithm which when executed on a processor of a virtual machine within a virtual center is configured to provide a service to a client over a network, the image processor algorithm comprising:

programming logic for receiving a connection request from the client at the virtual machine executing the image processor algorithm requesting a service, the connection request including a plurality of connection parameters defining connection requirements of the client;

programming logic for interrogating the connection parameters to identify a specific virtual machine providing the service that satisfies the request from the client;

programming logic for accessing framebuffer data for the specific virtual machine from a predetermined location in virtual memory through a hypervisor associated with the specific virtual machine, the framebuffer data being read from the predetermined location in virtual memory, the framebuffer data defining a graphical output representing an image associated with a virtual machine display of the specific virtual machine;

programming ~~instructions~~ logic for processing the read framebuffer data into a plurality of image data packets using the image processor algorithm based on the plurality of connection parameters, the programming logic for processing includes programming logic to,

scan the framebuffer data of the specific virtual machine to discern an image of a virtual machine display of the specific virtual machine including any changes in the framebuffer data of the specific virtual machine and to process the framebuffer data including any changes into a time-based storage array associated with the specific virtual machine; and

programming logic for transmitting the image data packets to the client for presenting on a display device associated with the client, the image data packets being specific for the client and represents the image of the virtual machine display.

49. The image processor algorithm of claim 48, wherein programming logic for processing the accessed framebuffer data further includes,

programming logic for analyzing the framebuffer data of the identified specific virtual machine to determine context; and

programming logic for associating appropriate contextual information to the framebuffer data of the specific virtual machine based on the analysis.

50. The image processor algorithm of claim ~~49~~48, programming logic for scanning changes further includes,

programming logic for scanning the framebuffer data of the specific virtual machine periodically;

programming logic for determining areas of change by comparing scanned framebuffer data against framebuffer data stored for the specific virtual machine; and

programming logic for storing the areas of change in ~~a~~the time-based storage array associated with the virtual machine for analysis and transmission, wherein storing of the areas of change includes storing of contextual information associated with the areas of change.

51. The image processor algorithm of claim 50, wherein the programming logic for scanning the framebuffer data further includes,

programming logic for interrogating a ~~sync-signal~~sync signal associated with the framebuffer data of the specific virtual machine, the sync signal identifying status of a refresh of the framebuffer data of the specific virtual machine; and

programming logic for performing the scanning of the framebuffer data of the specific virtual machine based on the interrogation of the sync-signal,

wherein the scanning is performed periodically at a refresh frequency of the framebuffer data at the virtual machine or at a frequency driven by one or more of the connection parameters associated with the client.

52. The image processor algorithm of claim 51, further includes,

programming logic for setting the sync-signal prior to a refresh of the framebuffer data for the specific virtual machine; and

programming logic for re-setting the sync-signal upon completion of the refresh of the framebuffer data of the specific virtual machine.