

Systems and Methods for Data Synchronization in a Network Application

RELATED APPLICATIONS

[0001] This application claims priority to Chinese Patent Application No. 201210297918.8, "Systems and Methods for Data Synchronization in a Network Application," filed on August 21, 2012, which is hereby incorporated by reference in its entirety.

FIELD OF THE INVENTION

[0002] The present invention generally relates to Internet technology, in particular to a network application method and apparatus for data synchronization.

BACKGROUND OF THE INVENTION

[0003] With the development of the Internet, network applications are also booming on the Internet, users could achieve data exchange through a variety of network applications running on different networks. Specifically, among a plurality of users running the same network application, each user can use the network interface provided by the network application to trigger the various operation instructions. The network application server will handle the received operation instructions and generate the appropriate data. Then based on the generated operation data, the network application server return the user-triggered operations as feedback on different user interfaces of the application. For example, if the network application is a game application, two users play the game application and exchange information with each other using the Internet. When one user triggers the movement of a character in the game application through any application interface, the game character will move accordingly. The game application server will then forward the character movement command to the other network application used by another user, thereby enabling the corresponding character movement on the user interface. However, since different network applications have different network environments, if a network application's network environment is poor or if there is great volatility of the network environment, they may prevent the appropriate data synchronization of different network applications.

SUMMARY

[0004] The above deficiencies and other problems associated with network environment fluctuations are reduced or eliminated by the invention disclosed below. In some embodiments, the invention is implemented in a computer system that has one or more processors, memory and one or more modules, programs or sets of instructions stored in the memory for performing multiple functions. Instructions for performing these functions may be included in a computer program product configured for execution by one or more processors.

[0005] One aspect of the invention involves a computer-implemented method performed by a computer system. The computer system receives user input events associated with a network application. Each user input event corresponds to a respective image frame of the network application and the user input events include at least one remote user input event from a remote source. The computer generates and updates response data from the respective image frames associated with the user input events to a current image frame of the network application. In some embodiments, the user input events are all from a local user of the computer. The computer also generates and updates synchronization data from the respective image frame associated with the at least one remote user input event to the current image frame of the network application. Finally, the computer replaces the response data with the synchronization data.

[0006] Another aspect of the invention involves a computer system. The computer system includes one or more processors, memory, and one or more program modules stored in the memory and configured for execution by the one or more processors. The one or more program modules include: a user input events receive module configured to receive user input events associated with a network application, wherein each user input event corresponds to a respective image frame of the network application and the user input events include at least one remote user input event from a remote source; a local response data update module configured to generate and update response data from the respective image frames associated with the user input events to a current image frame of the network application; a remote data synchronization module configured to generate and update synchronization data from the respective image frame associated with the at least one remote user input event to the current

image frame of the network application; and a data replacement module configured to replace the response data with the synchronization data.

[0007] Another aspect of the invention involves a non-transitory computer readable storage medium having stored therein one or more instructions, which when executed by a computer system cause the computer system to: receive user input events associated with a network application, wherein each user input event corresponds to a respective image frame of the network application and the user input events include at least one remote user input event from a remote source; generate and update response data from the respective image frames associated with the user input events to a current image frame of the network application; generate and update synchronization data from the respective image frame associated with the at least one remote user input event to the current image frame of the network application; and replace the response data with the synchronization data.

[0008] Some embodiments may be implemented on either the client side or the server side of a client-server network environment.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] The aforementioned features and advantages of the invention as well as additional features and advantages thereof will be more clearly understood hereinafter as a result of a detailed description of preferred embodiments when taken in conjunction with the drawings.

[0010] Figure 1 is a flowchart illustrative of how a network application handles different types of user input events in accordance with some embodiments.

[0011] Figure 2 is a flowchart illustrative of how the network application processes remote user input events from a remote computer in accordance with some embodiments.

[0012] Figure 3 is a flowchart illustrative of how the network application updates the synchronization data based on the response data in accordance with some embodiments.

[0013] Figure 4 is a block diagram illustrative of modules in the network application for handling different types of user input events in accordance with some embodiments.

[0014] Figure 5 is a block diagram illustrative of a remote data synchronization module in the network application in accordance with some embodiments.

[0015] Figure 6 is a block diagram illustrative of a data processing module in the network application in accordance with some embodiments.

[0016] Figure 7 is a block diagram illustrative of how computers associated with two users of the network application synchronize user input events with each other in accordance with some embodiments.

[0017] Figure 8 is a block diagram of a network environment in accordance with some embodiments.

[0018] Figure 9 is a block diagram of an exemplary client computer in accordance with some embodiments.

[0019] Figure 10 is a block diagram of a server in accordance with some embodiments.

[0020] Like reference numerals refer to corresponding parts throughout the several views of the drawings.

DESCRIPTION OF EMBODIMENTS

[0021] Reference will now be made in detail to embodiments, examples of which are illustrated in the accompanying drawings. In the following detailed description, numerous specific details are set forth in order to provide a thorough understanding of the subject matter presented herein. But it will be apparent to one skilled in the art that the subject matter may be practiced without these specific details. In other instances, well-known methods, procedures, components, and circuits have not been described in detail so as not to unnecessarily obscure aspects of the embodiments.

[0022] Figure 8 is a block diagram of a network environment in accordance with some embodiments. The network environment 800 includes one or more client computers (830-1, 830-3) and one or more servers 810. The client computers are connected to the servers via a network 820 (e.g., the Internet). Meanwhile different client computer may also

be able to communicate with each other via the network 820. For example, as described below, the user input events generated at the client computer 830-1 may be transmitted to the server 810 and/or the client computer 830-2. Through the network 810, different entities in the network environment 800 can exchange information with each other. In some embodiments, the network environment 800 supports an on-line game application. Different users participate in the on-line game application from different client computers. The server 810 includes a user information database 812 for storing information of the users including, but not limited to, a user identifier, a password, a user account balance, a current user status (e.g., the last input by the user), etc.

[0023] Figure 1 is a flowchart illustrative of how a network application handles different types of user input events in accordance with some embodiments. For illustrative purpose, the network application is an on-line game application running at one of the client computers shown in Figure 8. At step S10, the computer receives user input events associated with the network application. The user input events include both local user input events from the computer as well as remote user input events from remote computers. The local user input events are generated by a user of the computer or triggered by an action of the user. For example, when the user moves the mouse or presses a keystroke, a local user input event may be generated accordingly. Similarly, the movement of a mouse by a second user at a second client computer that is currently running the same network application first generates a local user input event at the second client computer and the local user input event is then transmitted to the client computer through the network 810, which may be deemed to be a remote user input event. Such user input events may cause the movement of an object on the graphical user interface of the network application. Ideally, the movement of the same object on different graphical user interfaces at different computers should happen nearly simultaneously. But in reality, it may take different amount of time for the same user input event to arrive at different client computers due to the potential network congestions, which may result in a downgrade in user experience. Therefore, one function of the present invention is to improve the synchronization between the movements of objects controlled by different users on different graphical user interfaces. In some embodiments, each user input event received by the computer corresponds to a respective image frame of the network application, which indicates when the event was captured by the network application, and the user input events include at least one remote user input event from a remote source.

[0024] At step S30, the computer generates and updates response data from image frames associated with the user input events to a current image frame of the network application. In some embodiments, upon receipt of a local user input event (or a user input event from a nearby computer that has little delay), the computer determines an initial image frame associated with the user input event and then generates the response data associated with the initial image frame. For example, the response data may include information about how a particular object moves on the graphical user interface across different image frames. If the initial image frame is different from the current image frame, the computer then updates the response data accordingly. As described below, the response data is used for rendering the current image frame on the graphical user interface. In some embodiments, the computer updates the response data corresponding to the respective image frames associated with the user input events at a first frame rate, e.g., 60 frames per second, which is usually the default frame rate of the network application. Because the graphical user interface is rendered frame by frame, the update to the response data is also performed on a frame-by-frame basis. Each time there is an update to the response data, the corresponding image frame is also re-rendered using the updated response data such that the computer can respond to the user inputs almost instantaneously. This process is performed continuously and repeatedly such that whenever there is a new user input event (e.g., a local user input event), the computer will update the response data such that the network application is always in synch with the user's instructions.

[0025] As noted above, the local user input event at one computer may become a remote user input event at another computer. Similarly, the user input events arriving at the computer may also include remote user input events, which may be out-of-synch with the local current status of the network application at the computer. Therefore, at step S50, the computer also generates and updates synchronization data from the image frame associated with at least one remote user input event to the current image frame of the network application. Like the update to the response data, the update to the synchronization data is also performed on a frame-by-frame basis until the number of updates to the response data is the same as the number of updates to the synchronization data. In some embodiments, the synchronization data cannot be used for rendering the current image frame. Instead, at step S70, the computer first replaces the response data, which has already been used for rendering the current graphical user interface, with the synchronization data by, e.g., storing the

synchronization data in the data structure for storing the response data. In some embodiments, the computer then uses the synchronization data to render the graphical user interface of the network application. By applying the synchronization data to the network application, the instance of the network application running at the computer is kept in synch with other instances of the same network application running at other computers such that the users at different client computers can play the same game with each other in a synchronized fashion.

[0026] In some embodiments, the response data is deemed to be immune from the network environment (e.g., network congestions). Regardless of whether a user input event arriving at a computer is local or remote, the computer will act quickly by updating the response data and use the updated response data to render the current image frame. But for a remote user input event that comes from a client computer where there is a significant delay, the computer cannot update the response data based on the remote user input event because the source client computer may have already updated its own response data before the arrival of the remote user input event. Instead, the computer uses the remote user input event to update its synchronization data, which effectively measures a gap between an image frame associated with the remote user input event and the current image frame of the network application. The synchronization data is then updated repeatedly (e.g., frame by frame) until it corresponds to the same image frame as the response data. Finally, the computer uses the updated synchronization data to render the image frame such that different users can exchange data with each other using the same network application synchronously.

[0027] Figure 2 is a flowchart illustrative of how the network application processes remote user input events from a remote computer in accordance with some embodiments. At step S500, the computer receives a remote user input event from a remote computer. Note that the remote user input event is assumed to be out-of-synch with computer and therefore cannot be used for updating the response data directly. In response, at step S510, the computer generates initial synchronization data for the respective image frame associated with the remote user input event. For example, if this is the first time that the computer receives a remote user input event from that remote computer, the computer may need to generate the initial synchronization data corresponding to the image frame associated with the remote user input event. Next, at step S530, the computer performs a frame-by-frame update

to the initial synchronization data until a number of updates to the synchronization data is the same as a number of updates to the response data and then replaces the response data with the synchronization data.

[0028] For example, it is assumed that the computer receives a remote user input event and the remote user input event corresponds to the Nth image frame of the network application and the response data corresponds to the (N+M)th image frame ($M \geq 0$). In other words, the two computers are currently out of synchronization. Therefore, the computer needs to quickly update the synchronization data to the (N+M)th image frame so as to bring the two computer back in synch again.

[0029] Figure 3 is a flowchart illustrative of how the network application updates the synchronization data to be in synch with the response data in accordance with some embodiments. At step S531, the computer updates the synchronization data corresponding to the image frame associated with the remote user input event at a second frame rate (e.g., 90 or 120 frames per second), which is usually greater than the first frame rate used by the computer for updating the response data. If not, the synchronization data will never be able to back in synch with the response data. At step S533, the computer determines the number of updates to the synchronization data and the number of updates to the response data. At step S535, the computer determines whether the number of updates to the synchronization data is the same as the number of updates to the response data. If so (Yes), at step S537, the computer then replaces the response data with the updated synchronization data. If not (No), the computer then returns to update the synchronization data at step S531.

[0030] Note that in the approach described above, the generation and update of the response data and the generation and update of the synchronization data are independent from each other because they are generated by different users at different computers using the same network application. However, when the update to the synchronization data satisfies the predefined condition, the synchronization data may be brought back in synchronization with the response data and therefore improve the user experience of the network application.

[0031] Figure 4 is a block diagram illustrative of modules in the network application for handling different types of user input events in accordance with some embodiments. In particular, the user input events receive module 10 is configured to receive user input events

associated with the network application. The user input events include both local user input events from the computer as well as remote user input events from remote computers. The local user input events are generated by a user of the computer or triggered by an action of the user. For example, when the user moves the mouse or presses a keystroke, a local user input event may be generated accordingly. Similarly, the movement of a mouse by a second user at a second client computer that is currently running the same network application first generates a local user input event at the second client computer and the local user input event is then transmitted to the client computer through the network 810, which may be deemed to be a remote user input event. Such user input events may cause the movement of an object on the graphical user interface of the network application. Ideally, the movement of the same object on different graphical user interfaces at different computers should happen nearly simultaneously. But in reality, it may take different amount of time for the same user input event to arrive at different client computers due to the potential network congestions, which may result in a downgrade in user experience. Therefore, one function of the present invention is to improve the synchronization between the movements of objects controlled by different users on different graphical user interfaces. In some embodiments, each user input event received by the user input events receive module 10 corresponds to a respective image frame of the network application, which indicates when the event was captured by the network application, and the user input events include at least one remote user input event from a remote source.

[0032] The local response data update module 30 is configured to generate and update response data from image frames associated with the user input events to a current image frame of the network application. In some embodiments, upon receipt of a local user input event (or a user input event from a nearby computer that has little delay), the local response data update module 30 determines an initial image frame associated with the user input event and then generates the response data associated with the initial image frame. For example, the response data may include information about how a particular object moves on the graphical user interface across different image frames. If the initial image frame is different from the current image frame, the local response data update module 30 then updates the response data accordingly. As described below, the response data is used for rendering the current image frame on the graphical user interface. In some embodiments, the local response data update module 30 updates the response data corresponding to the respective

image frames associated with the user input events at a first frame rate, e.g., 60 frames per second, which is usually the default frame rate of the network application. Because the graphical user interface is rendered frame by frame, the update to the response data is also performed on a frame-by-frame basis. Each time there is an update to the response data, the corresponding image frame is also re-rendered using the updated response data such that the computer can respond to the user inputs almost instantaneously. This process is performed continuously and repeatedly such that whenever there is a new user input event (e.g., a local user input event), the computer will update the response data such that the network application is always in synch with the user's instructions.

[0033] The remote data synchronization module 50 is configured to generate and update synchronization data from the image frame associated with at least one remote user input event to the current image frame of the network application. Like the update to the response data, the update to the synchronization data is also performed on a frame-by-frame basis by the remote data synchronization module 50 until the number of updates to the response data is the same as the number of updates to the synchronization data. In some embodiments, the synchronization data cannot be used for rendering the current image frame. Instead, the remote data synchronization module 50 provides the update synchronization data to a data replacement module 70, which is configured to first replace the response data with the synchronization data by, e.g., storing the synchronization data in the data structure for storing the response data. In some embodiments, an image rendition module (not shown in Figure 4) uses the synchronization data to render the graphical user interface of the network application. By applying the synchronization data to the network application, the instance of the network application running at the computer is kept in synch with other instances of the same network application running at other computers such that the users at different client computers can play the same game with each other in a synchronized fashion.

[0034] In some embodiments, the response data is deemed to be immune from the network environment (e.g., network congestions). Regardless of whether a user input event arriving at a computer is local or remote, the computer will act quickly by updating the response data and use the updated response data to render the current image frame. But for a remote user input event that comes from a client computer where there is a significant delay, the computer cannot update the response data based on the remote user input event because

the source client computer may have already updated its own response data before the arrival of the remote user input event. Instead, the computer uses the remote user input event to update its synchronization data, which effectively measures a gap between an image frame associated with the remote user input event and the current image frame of the network application. The synchronization data is then updated repeatedly (e.g., frame by frame) until it corresponds to the same image frame as the response data. Finally, the computer uses the updated synchronization data to render the image frame such that different users can exchange data with each other using the same network application synchronously.

[0035] Figure 5 is a block diagram illustrative of a remote data synchronization module in the network application in accordance with some embodiments. In this example, the remote data synchronization module 50 further includes an input event response module 510 and a data processing module 530. For example, the input event response module 510 is configured to receive a remote user input event from a remote computer and generate initial synchronization data for the respective image frame associated with the remote user input event. For example, the remote user input event is assumed to be out-of-synch with computer and therefore cannot be used for updating the response data directly. If this is the first time that the computer receives a remote user input event from that remote computer, the computer may need to generate the initial synchronization data corresponding to the image frame associated with the remote user input event. The data processing module 530 is configured to perform a frame-by-frame update to the initial synchronization data until a number of updates to the synchronization data is the same as a number of updates to the response data and then replaces the response data with the synchronization data.

[0036] For example, it is assumed that the computer receives a remote user input event and the remote user input event corresponds to the Nth image frame of the network application and the response data corresponds to the (N+M)th image frame ($M \geq 0$). In other words, the two computers are currently out of synchronization. Therefore, the computer needs to quickly update the synchronization data to the (N+M)th image frame so as to bring the two computer back in synch again.

[0037] Figure 6 is a block diagram illustrative of a data processing module in the network application in accordance with some embodiments. In this example, the data

processing module 530 further includes a synchronization data update module 531, a data update frequency determination module 533, and a response data update module 535. For example, the synchronization data update module 531 is configured to update the synchronization data corresponding to the image frame associated with the remote user input event at a second frame rate (e.g., 90 or 120 frames per second), which is usually greater than the first frame rate used by the computer for updating the response data. The data update frequency determination module 533 is configured to determine the number of updates to the synchronization data and the number of updates to the response data. The response data update module 535 is configured to determine whether the number of updates to the synchronization data is the same as the number of updates to the response data. If so, the response data update module 535 then replaces the response data with the updated synchronization data. If not, the response data update module 535 then notifies the synchronization data update module 531 to update the synchronization data.

[0038] Note that in the approach described above, the local response data update module 30 and the remote data synchronization module 50 operate independently from each other because they are generated by different users at different computers using the same network application. However, when the update to the synchronization data satisfies the predefined condition, the synchronization data may be brought back in synchronization with the response data and therefore improve the user experience of the network application. In some embodiments, the response data update module 30 is located at a client computer while the remote data synchronization module 50 is located at the server 810. In some other embodiments, both the response data update module 30 and the remote data synchronization module 50 may be located at the server 810.

[0039] Figure 7 is a block diagram illustrative of how computers associated with two users of the network application synchronize user input events with each other in accordance with some embodiments. In this example, the user 1 (700-1) may use a local computer to exchange information with the user 2 (700-2) that uses another computer. For example, the two users may play an on-line game application. For example, the computer associated with the user 1 may include a user input events receive module 10 for receiving different types of user input events. For the received user input events, the local response data update module 30 is responsible for processing local user input events from the user 1 (e.g., generating and

updating response data associated with the local user input events). Within the remote data synchronization module 50, the input event response module 510 and the data processing module 530 are responsible for processing remote user input events (that may be out of synch with the local user input events) and keeping the different computers running the same network application in synch with each other.

[0040] In this example, the computer associated with the user 1 receives three local input events triggered by the user 1, local user input event 1, local user input event 2, and local user input event 3, which are respectively associated with three different image frames of the network application. As noted above, for each user input event, the local response data update module 30 is triggered to generate and update response data and use the response data to render the respective image frames. When the network application renders the fourth image frame, a remote user input event from the computer associated with the user 2 arrives at the computer. This event triggers the input event response module 510 of the computer to generate the synchronization data based on the local user input event 1 and the remote user input event 1. The data processing module 530 then updates the synchronization data from the first image frame to the second, the third, and the fourth image frame such that the synchronization data is updated the same number of times like the response data. Finally, the synchronization data is stored in the data structure for storing the response data and then used for rendering the fourth image frame accordingly such that the fourth image frame at the computer associated with the user 1 is in synchronization with the corresponding image frame at the computer associated with the user 2. Note that the computer associated with the user 2 processes the three local user input events in substantially the same manner if they arrive at the computer with a substantial delay such that they cannot be used for updating the response data.

[0041] Figure 9 is a block diagram of an exemplary client computer in accordance with some embodiments. The exemplary client computer 830 typically includes one or more processing units (CPU's) 902, one or more network or other communications interfaces 904, memory 910, and one or more communication buses 909 for interconnecting these components. The communication buses 909 may include circuitry (sometimes called a chipset) that interconnects and controls communications between system components. The client computer 830 may include a user input device 905, for instance, a display 906 and a

keyboard 908. Memory 910 may include high speed random access memory and may also include non-volatile memory, such as one or more magnetic disk storage devices. Memory 910 may include mass storage that is remotely located from the CPU's 902. In some embodiments, memory 910 stores the following programs, modules and data structures, or a subset or superset thereof:

- an operating system 912 that includes procedures for handling various basic system services and for performing hardware dependent tasks;
- a network communication module 914 that is used for connecting the client computer 830 to the server 810 or other computers via one or more communication networks (wired or wireless), such as the Internet, other wide area networks, local area networks, metropolitan area networks, and so on;
- a user interface module 916 configured to receive user inputs through the user interface 905; and
- one or more client-side network applications 918, each network application further including a user input events receive module 10, a local response data update module 30, a remote data synchronization module 50, and a data replacement module 70 as described above in connection with Figure 4.

[0042] Figure 10 is a block diagram of a server in accordance with some embodiments. The exemplary server 810 typically includes one or more processing units (CPU's) 1002, one or more network or other communications interfaces 1004, memory 1010, and one or more communication buses 1009 for interconnecting these components. The communication buses 1009 may include circuitry (sometimes called a chipset) that interconnects and controls communications between system components. The server 810 may optionally include a user interface 1005, for instance a display 1006 and a keyboard 1008. Memory 1010 may include high speed random access memory and may also include non-volatile memory, such as one or more magnetic disk storage devices. Memory 1010 may include mass storage that is remotely located from the CPU's 1002. In some embodiments, memory 1010 stores the following programs, modules and data structures, or a subset or superset thereof:

- an operating system 1012 that includes procedures for handling various basic system services and for performing hardware dependent tasks;
- a network communication module 1014 that is used for connecting the server 810 to other servers or client computers 830 via one or more communication networks (wired or wireless), such as the Internet, other wide area networks, local area networks, metropolitan area networks, and so on;
- a system initialization module 1016 that initializes other modules and data structures stored in the memory 1012 required for the appropriate operation of the server 810;
- an optional user interface module 1018 configured to receive user inputs through the user interface 1005; and
- one or more server-side network applications 1020, each network application further including user information data 1022 (e.g., a user identifier, a password, a user account balance, a current user status).

[0043] While particular embodiments are described above, it will be understood it is not intended to limit the invention to these particular embodiments. On the contrary, the invention includes alternatives, modifications and equivalents that are within the spirit and scope of the appended claims. Numerous specific details are set forth in order to provide a thorough understanding of the subject matter presented herein. But it will be apparent to one of ordinary skill in the art that the subject matter may be practiced without these specific details. In other instances, well-known methods, procedures, components, and circuits have not been described in detail so as not to unnecessarily obscure aspects of the embodiments.

[0044] The terminology used in the description of the invention herein is for the purpose of describing particular embodiments only and is not intended to be limiting of the invention. As used in the description of the invention and the appended claims, the singular forms "a," "an," and "the" are intended to include the plural forms as well, unless the context clearly indicates otherwise. It will also be understood that the term "and/or" as used herein refers to and encompasses any and all possible combinations of one or more of the associated listed items. It will be further understood that the terms "includes," "including," "comprises," and/or "comprising," when used in this specification, specify the presence of stated features,

operations, elements, and/or components, but do not preclude the presence or addition of one or more other features, operations, elements, components, and/or groups thereof.

[0045] As used herein, the term "if" may be construed to mean "when" or "upon" or "in response to determining" or "in accordance with a determination" or "in response to detecting," that a stated condition precedent is true, depending on the context. Similarly, the phrase "if it is determined [that a stated condition precedent is true]" or "if [a stated condition precedent is true]" or "when [a stated condition precedent is true]" may be construed to mean "upon determining" or "in response to determining" or "in accordance with a determination" or "upon detecting" or "in response to detecting" that the stated condition precedent is true, depending on the context.

[0046] Although some of the various drawings illustrate a number of logical stages in a particular order, stages that are not order dependent may be reordered and other stages may be combined or broken out. While some reordering or other groupings are specifically mentioned, others will be obvious to those of ordinary skill in the art and so do not present an exhaustive list of alternatives. Moreover, it should be recognized that the stages could be implemented in hardware, firmware, software or any combination thereof.

[0047] The foregoing description, for purpose of explanation, has been described with reference to specific embodiments. However, the illustrative discussions above are not intended to be exhaustive or to limit the invention to the precise forms disclosed. Many modifications and variations are possible in view of the above teachings. The embodiments were chosen and described in order to best explain the principles of the invention and its practical applications, to thereby enable others skilled in the art to best utilize the invention and various embodiments with various modifications as are suited to the particular use contemplated.

What is claimed is:

1. A computer-implemented method for data synchronization, comprising:
 - at a computer having one or more processors and memory storing programs executed by the one or more processors,
 - receiving user input events associated with a network application, wherein each user input event corresponds to a respective image frame of the network application and the user input events include at least one remote user input event from a remote source;
 - generating and updating response data from the respective image frames associated with the user input events to a current image frame of the network application;
 - generating and updating synchronization data from the respective image frame associated with the at least one remote user input event to the current image frame of the network application; and
 - replacing the response data with the synchronization data.
2. The method of claim 1, wherein generating and updating response data from the respective image frames associated with the user input events to the current image frame of the network application further includes:
 - updating the response data corresponding to the respective image frames associated with the user input events at a first frame rate; and
 - rendering the current image frame of the network application using the response data.
3. The method of claim 2, wherein generating and updating synchronization data from the respective image frame associated with the at least one remote user input event to the current image frame of the network application further includes:
 - generating initial synchronization data for the respective image frame associated with the at least one remote user input event; and
 - performing a frame-by-frame update to the initial synchronization data until a number of updates to the synchronization data is the same as a number of updates to the response data.
4. The method of claim 3, wherein performing the frame-by-frame update to the initial synchronization data until a number of updates to the synchronization data is the same as a number of updates to the response data further includes:

updating the synchronization data corresponding to the respective image frame associated with the at least one remote user input event at a second frame rate;

determining the number of updates to the synchronization data and the number of updates to the response data, respectively;

if the number of updates to the synchronization data is the same as the number of updates to the response data, replacing the response data with the synchronization data; and
otherwise, repeating the updating of the synchronization data.

5. The method of claim 4, wherein the second frame rate is greater than the first frame rate.

6. The method of claim 1, further comprising:
rendering the current image frame of the network application using the synchronization data.

7. A computer system, comprising:
one or more processors;
memory; and
one or more program modules stored in the memory and configured for execution by the one or more processors, the one or more program modules including:

a user input events receive module configured to receive user input events associated with a network application, wherein each user input event corresponds to a respective image frame of the network application and the user input events include at least one remote user input event from a remote source;

a local response data update module configured to generate and update response data from the respective image frames associated with the user input events to a current image frame of the network application;

a remote data synchronization module configured to generate and update synchronization data from the respective image frame associated with the at least one remote user input event to the current image frame of the network application; and

a data replacement module configured to replace the response data with the synchronization data.

8. The computer system of claim 7, wherein the local response data update module is further configured to:

update the response data corresponding to the respective image frames associated with the user input events at a first frame rate; and

render the current image frame of the network application using the response data.

9. The computer system of claim 8, wherein the remote data synchronization module is further configured to:

generate initial synchronization data for the respective image frame associated with the at least one remote user input event; and

perform a frame-by-frame update to the initial synchronization data until a number of updates to the synchronization data is the same as a number of updates to the response data.

10. The computer system of claim 9, wherein the instruction for performing the frame-by-frame update to the initial synchronization data until a number of updates to the synchronization data is the same as a number of updates to the response data further includes instructions for:

updating the synchronization data corresponding to the respective image frame associated with the at least one remote user input event at a second frame rate;

determining the number of updates to the synchronization data and the number of updates to the response data, respectively;

if the number of updates to the synchronization data is the same as the number of updates to the response data, replacing the response data with the synchronization data; and
otherwise, repeating the updating of the synchronization data.

11. The computer system of claim 10, wherein the second frame rate is greater than the first frame rate.

12. The computer system of claim 7, wherein the data replacement module is further configured to:

render the current image frame of the network application using the synchronization data.

13. A non-transitory computer readable storage medium having stored therein one or more instructions, which, when executed by a computer system, cause the computer system to:

receive user input events associated with a network application, wherein each user input event corresponds to a respective image frame of the network application and the user input events include at least one remote user input event from a remote source;

generate and update response data from the respective image frames associated with the user input events to a current image frame of the network application;

generate and update synchronization data from the respective image frame associated with the at least one remote user input event to the current image frame of the network application; and

replace the response data with the synchronization data.

14. The non-transitory computer readable storage medium of claim 13, wherein the instruction for generating and updating response data from the respective image frames associated with the user input events to the current image frame of the network application further includes instructions for:

updating the response data corresponding to the respective image frames associated with the user input events at a first frame rate; and

rendering the current image frame of the network application using the response data.

15. The non-transitory computer readable storage medium of claim 14, wherein the instruction for generating and updating synchronization data from the respective image frame associated with the at least one remote user input event to the current image frame of the network application further includes instructions for:

generating initial synchronization data for the respective image frame associated with the at least one remote user input event; and

performing a frame-by-frame update to the initial synchronization data until a number of updates to the synchronization data is the same as a number of updates to the response data.

16. The non-transitory computer readable storage medium of claim 15, wherein the instruction for performing the frame-by-frame update to the initial synchronization data until

a number of updates to the synchronization data is the same as a number of updates to the response data further includes instructions for:

updating the synchronization data corresponding to the respective image frame associated with the at least one remote user input event at a second frame rate;

determining the number of updates to the synchronization data and the number of updates to the response data, respectively;

if the number of updates to the synchronization data is the same as the number of updates to the response data, replacing the response data with the synchronization data; and

otherwise, repeating the updating of the synchronization data.

17. The non-transitory computer readable storage medium of claim 16, wherein the second frame rate is greater than the first frame rate.

18. The non-transitory computer readable storage medium of claim 13, wherein one or more instructions further include instructions for rendering the current image frame of the network application using the synchronization data.