EXHIBIT D-13

Invalidity Contentions for U.S. Patent No. 7,702,719
Based on U.S. Patent No. 5,838,909 ("Roy")

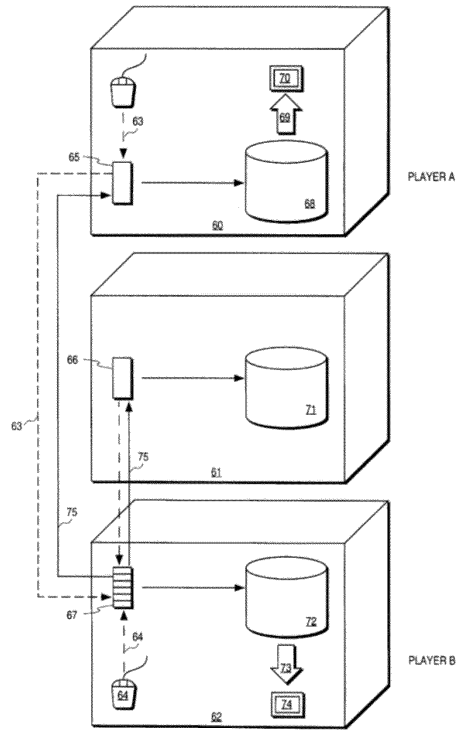
Based upon IBM's counterclaims, infringement contentions, and apparent claim constructions and application of the claims to Carvana's accused products, as best as they can be deciphered, the reference charted below anticipates or at least renders obvious the asserted claims. These invalidity contentions are not an admission by Carvana that the accused products are covered by or infringe the asserted claims, particularly when these claims are properly construed and applied. These invalidity contentions are not an admission that Carvana concede or acquiesce to any claim construction implied or suggested by IBM's counterclaims or infringement contentions. Nor is Carvana asserting any claim construction positions through these charts, including whether the preamble is a limitation. The portions of the prior art reference cited below are not exhaustive but are exemplary in nature.

Asserted Claims 1-4, 8, 10-12, and 26 of U.S. Patent No. 7,702,719 ("the '719 patent") are anticipated and/or obvious in view of U.S. Patent No. 5,838,909 ("Roy"), either alone or in combination with the knowledge of a person of ordinary skill in the art ("POSITA"), or in combination with one or more other references disclosed in Carvana's Invalidity Contentions, including the other charted references, and the Applicant's admitted prior art ("AAPA") of the '719 patent.

Roy was filed on May 23, 1996 and issued on November 17, 1998. Roy is prior art under at least pre-AIA 35 U.S.C. §§ 102(a), (b), and (e).

In addition, Carvana incorporates by reference the prior art disclosures identified in the invalidity claim charts served in prior matters, including the claims charts provided in and with IBM-CARVANA00020414 (Zynga 3/17/2023); IBM-CARVANA00022219 (Zynga 10/27/2023); and IBM-CARVANA00022845 (Zynga 12/12/2023).

Claim Language	Prior Art References
1(pre). A method for use in a client/server system of reducing interactions between a client and server in association with an application being accessed by the client at	To the extent the preamble is limiting, Roy discloses a method for use in a client/server system (*e.g.*, the client/server systems shown in Figures 1, 4, 5, or 6) of reducing interactions between a client (*e.g.*, computer system 60) and server (*i.e.*, central server 61 or whichever computer system has the "master event server" and "master database"-in the example of Figure 6, the "master event server" and "master database" are migrated from central server 61 to computer system 62 such that computer system 62 becomes the "server", and the client computers, such as computer system 60, may connect directly to computer system 62 as depicted by the dotted line 63) in association with

Claim Language	Prior Art References
the server, the method comprising the steps of:	an application (*i.e.*, an online multiplayer game) being accessed by the client at the server. For example, *see* Figure 6:  **FIG. 6** Roy is directed to providing "a method of reducing network latency in accessing a database in a network." 1:66-67. And Roy discloses at least two different ways regarding how portions of the application are executed on the client with reduced server interaction. First, "local events" are processed solely by the client without interaction with the server. *See* Fig. 8A. Roy explains that an "event" occurs each time a user issues a command (*e.g.*, player A on computer system 60 "clicks" something in the game). If the event is determined to be a "local

Claim Language	Prior Art References
	event," then it is processed locally by the client and the results displayed without requiring any server interaction. *See* Fig. 8A. Specifically, "[e]vents 64 from computer system 62 are received by slave event server 67 and then passed along to the master event server 66, unless the events 64 are purely local events for updating only the slave copy 72 of the database." 5:37-40. "Purely local events might be, for example, events which set display options or other functions that are only relevant to [that] computer system." 5:40-42. "Similarly, events 63 originating from computer system 60 are received first by slave event server 65 and then passed along to master event server 66, unless those events are purely local to computer system 60." 5:42-46; *see also* 5:29-33 ("The display 74 of computer system 62 is updated by rendering commands 73 from the slave copy 72 of master database 71, rather than from the master database itself, thereby reducing bandwidth consumption on the network."). Second, by copying the master event server and master database from central server 61 to one of the computer systems (*e.g.*, computer system 62), all events generated by the respective player on that computer system (*e.g.*, player Bon computer system 62) are processed by that computer system without interaction with central server 61, regardless of whether the events are deemed local events or master events. *See* 6:5-14 ("[M]aster event server 66 is migrated to computer system 62. More specifically, the status of event server 66 is changed from master event server to slave event server, and the status of event server 67 is changed from slave event server to master event server. Similarly, the master database 71 is migrated to computer system 62. . . . Events 75 output by (newly designated) master event server 67 are then considered to be master events. . . ."). For example, *see* the following passages and/or figures, as well as all related disclosures: • "A method and apparatus for reducing network latency during execution of a multiple-player game across a computer network are provided. A master database represents a world model for the game, and a master event server sequences user inputs, or events, for updating the master database. The master event server and the master database are initially located on a particular node in the network. Every other node that is used by a player in the game has a slave event server and a slave database. Slave event servers are responsible for updating their local slave database, sending events from their local node to the master

Claim Language	Prior Art References
	event server, and forwarding events to and from other slave servers. In the method, a determination is made of which player in the game has a role in the game requiring the least latency of any role and which node in the network is being used by that player. The master event server and the master database are then mi rated to that node by changing the status of the slave server of that node to that of master event server. Once the master event server has been migrated, other slave servers can establish a direct connection with the new master event server." Abstract. • "One such problem is that even computer games which are designed for multiple players using different computers tend to be sensitive to latency in communicating data between computers. Consider, for example, a computer game in which two or more players at remote locations interact with each other within a fictitious "game world." The game world can be modeled by a database, which might be located on one of the player's computers or on a central server. The players interact with each other by entering inputs through a joystick, mouse, keyboard, or other input/output device. In order for the game to be enjoyable, the game world must accurately reflect the current history of user inputs, or "events," received from all players. To resolve ambiguity of event ordering, events are typically collected at a central server that establishes the event sequence. However, differences in delays between the various computers involved in the game can hinder the proper sequencing of events when updating the game world. In addition, the speed with which events are communicated to and from the game world (*i.e.*, the database) can be critical, depending upon the type of game being played. For example, in naturally fast-paced games, such as those involving close combat between characters and many simulations, high latency can cause the game to be unrealistic, if not completely unplayable. High latency can be quite problematic even in games in which only particular players have latency-sensitive roles." 1:23-48. • "FIG. SA illustrates a technique according to the present invention for implementing a multi-user game over the Internet. In FIG. SA, the game is played by two players, players A and B, operating computer systems 60 and 62, respectively. It is noted, however, that the present invention is equally applicable to a game played simultaneously by more than two players. Initially, the master database 71 and the master event server 66 are located in a

Claim Language	Prior Art References
	central server computer system 61. Located in computer system 60 (used by player A) are a slave event queue 65 and a slave copy 68 of master database 71. Similarly, located in computer system 62 (used by player B) is a slave event server 67 and a slave copy 72 of master database 71. The display 74 of computer system 62 is updated by rendering commands 73 from the slave copy 72 of master database 71, rather than from the master database 71 itself, thereby reducing bandwidth consumption on the network. Similarly, the display 70 associated with computer system 60 is updated by rendering commands 69 from slave copy 68 of master database 71, rather than directly from master database 71. Events 64 from computer system 62 are received by slave event server 67 and then passed along to the master event server 66, unless the events 64 are purely local events for updating only the slave copy 72 of the database. Purely local events might be, for example, events which set display options or other functions that are only relevant to computer system 62. Similarly, events 63 originating from computer system 60 are received first by slave event server 65 and then passed along to master event server 66, unless those events are purely local to computer system 60. Events received by the master event server 66 are used to update the master database 71 and are passed along to all slave event servers as master events 75." 5:17-49. • "FIG. 6 illustrates a multi-player game running in a network in which a slave event server has established a direct connection to the master event server." 2:38-41. • "The present invention is particularly suited to reducing latency when implementing a multi-player game on the Internet. In particular, the present invention is suited to implementing a game in which a fictitious game world is modeled by a database, and in which one particular player of the game has a role with a latency requirement that is significantly more restrictive than the other players' roles. An example of such a game is a simulated (American) football game in which one player plays offense and a second player plays defense at any given point in time. In such a game, the player on offense might be responsible for movement of the players, including the speed and direction of various players, while the player on defense might be responsible only for selecting a predefined defensive pattern. Hence, in such a game, the player on offense would have substantial interaction with the game world, whereas the player on defense would have little or no

Claim Language	Prior Art References
	interaction with the game world. Accordingly, the role of the player on offense would have a substantially more restrictive latency requirement than the role of the defensive player. Another example of a game in which one player has substantially more restrictive latency requirement than the other players is a role-playing fantasy game. In such a game, one player acting as a "superhero" may participate on a very active level, while various parameters of the game might be determined at a high level by one or a small number of players who participate only in a peripheral manner (*e.g.*, by setting game conditions, selecting obstacles for the superhero to overcome, etc.). Hence, the player having the superhero role would have a much more restrictive latency requirement than the other players, because he interacts with the database on a much more active level." 3:16-4 7. • "Referring now to FIG. 1, a network configuration in which the present invention may be implemented is illustrated. A central server computer 1 is coupled via the Internet 3 to a number of client computer systems 2. The server computer system 1 and each of the client computer systems 2 includes stored game software, including software in which the present invention is embodied. The game software includes various application program interfaces (APIs) providing an interface to operating system software. FIG. 2 illustrates an example of an architecture which may be used for any of the client computer systems 2 or the server computer system 1. The architecture includes a central processing unit (CPU) 10, memory 11, a mass storage device 12, a modem 13, and various input/output (I/O) devices 14. Memory 11 represents both random access memory (RAM) and read-only memory (ROM). Mass storage device 12 includes an optical, magnetic, or other similar storage medium suitable for use with a personal computer system. I/O devices 14 may include any combination of devices such as a mouse, a joystick, a trackball, a keyboard, a monitor, or other I/O devices. The software in which the present invention is embodied may be stored in mass storage device 12, memory 11, or both. In addition, the software may be distributed between two or more of the computer systems illustrated in FIG. 1. FIG. 4A shows a representation of a computer system 40 running game software according to one prior art embodiment. A database 43 provides a world model for the game (*i.e.*, a "game world"). User inputs 41 are received through a game-compatible I/O device 46 (such as a mouse or joystick) and provided to software 42. Software 42 receives and sequences user inputs, or "events," that are used to update the database 43. Rendering commands 45 are

Claim Language	Prior Art References
	sent from the database to a display 44 of the computer system 40 to generate various scenes representing the game world. Techniques for implementing a game as illustrated in FIG. 4A are well-known. FIG. 4B shows a well-known configuration for implementing a multi-user game on a network. In the configuration of FIG. 4B, the game world is provided by a database 53 located on a central server 51. Client computer systems 50 and 52 are used by Players A and B of the game, respectively. Software 54 located in the central server 51 receives events 58 from computers 50 and 52 and provides these events to the database 53 in order to update the game world. Rendering commands 56 are then sent from the updated database 53 in the central server 51 across the network to update the displays 55 and 57 of client computers 50 and 52, respectively. As mentioned above, implementing as shown in FIG. 4B on the Internet can be problematic if either of Players A and B are subject to a significant latency restriction. In addition, the transmission of rendering commands 56 from server 51 to clients 50 and 52 can be hindered by bandwidth limitations of the network. Consequently, the present invention provides a technique for overcoming such difficulties." 3:48-4:36. One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Invalidity Contentions Cover Pleading, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart. Specific prior art combinations (in conjunction with the knowledge of a POSITA) are as follows: <u>Roy+ Rover</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of Rover. *See* '719 Patent Invalidity Chart, Ex. D-8 (Rover) at

Claim Language	Prior Art References
	claim preamble 1(pre) (incorporated herein by reference). A POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Rover's teachings about moving application data and logic between a client and server using relocatable dynamic objects ("RDOs"). For example, *see* claim limitations 1(a) and 1(b), and claim 12. <u>Roy + Bennett</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of Bennett. *See* '719 Patent Invalidity Chart, Ex. D-15 (Bennett) at claim preamble 1(pre) (incorporated herein by reference). A POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Bennett's teachings about moving application data and logic between a client and server using objects. For example, *see* claim limitations 1(a) and 1(b). <u>Roy+ Hurly</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of The Videotex and Teletex Handbook by Paul Hurly ("Hurly"). Hurly was published in 1985 and qualifies as prior art to the '849 patent under at least 35 U.S.C. §§ 102(a) and/or (b). A POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Hurly's teachings about downloading data and/or logic on demand from a server to a client computer. For example, *see* claim 12. For example, Roy and Hurly are in the same field of endeavor (interactive electronic information delivery systems over a network). *See* Roy, Abstract ("A method and apparatus for reducing network latency during execution of a multiple-player game across a computer network are provided."); Hurly at 18 ("Viewdata is the generic name for fully interactive videotex. This means that the user's requests for information, or service, are actually sent to, received by, and acted on by the centralized computer."). As another example, Hurly describes how games such as backgammon can be played online. Hurly at 168. <u>Roy+ Lowery</u>

Claim Language	Prior Art References
	To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of Lowery. *See* '719 Patent Invalidity Chart, Ex. D-1 (Lowery) at claim preamble 1(pre) (incorporated herein by reference). A POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Lowery's teachings about downloading a subset of application logic over a network on demand. For example, *see* claim 12.
1(a). configuring the server to store a model associated with the application and to execute view-generating and controller logic associated with the application; and	Roy discloses configuring the server (*i.e.*, central server 61 or whichever computer system has the "master event server" and "master database") to store a model (*i.e.* a "database represent[ing] a world model for the game") associated with the application (*i.e.*, an online multiplayer game) and to execute view-generating and controller logic associated with the application. Roy configures the server to store a model associated with the online multiplayer game. In particular, the "master database" "provides a world model for the game (*i.e.*, a 'game world')." 4:8-10, 40; *see also* 1:28-30 ("The game world can be modeled by a database, which might be located on one of the player's computers or on a central server."). "In the preferred embodiment, the master database is located on the same machine as the master event server." 4:45-46. Roy further configures the server to execute controller logic ("CL"), and the server does, in fact, execute CL. In particular, Figure 2 "illustrates an example of an architecture which may be used for any of the client computer systems 2 or the server computer system 1." 3:56-59. "I/O devices 14 may include any combination of devices such as a mouse, a joystick, a trackball, a keyboard, a monitor, or other I/O devices." 3:66-4:1. "User inputs 41 are received through a game-compatible I/O device 46 (such as a mouse or joystick) and provided to software 42." 4:9-11. "Software 42 receives and sequences user inputs, or 'events,' that are used to update the database 43." 4:11-13. For example, in the configuration depicted in Figure 4B, "the game world is provided by a database 53 located on central server 51." 4:19-21. "Software 54 located in the central server 51 receives events 58 from computers 50 and 52 and provides these events to the database 53 in order to update the game world [*i.e.*, the server executes CL and updates the model of the game world stored in database 53]." 4:23-26. As another example, as shown by Figure 6 and described above, computer system 62 becomes the "server." In this case, events "passed along to the master event server [67] are used to update the

Claim Language	Prior Art References
	master database [72]." 5:46-47. Player B may generate events that cause CL to be executed on the server (*i.e.*, computer system 62) and the model in the master database 72 to be updated. Additionally, other players may generate events (such as player A on computer system 60) that cause CL to be executed on the server (*i.e.*, computer system 62) and the model in the master database 72 to be updated. *See* 5:37-49. Roy further configures the server to execute view-generating logic ("VGL"), and the server does, in fact, execute VGL. For example, returning to Figure 4B, "[s]oftware 54 located in the central server 51 receives events 58 from computers 50 and 52 and provides these events to the database 53 in order to update the game world." 4:23-26. "Rendering commands 56 [*i.e.*, a view] are then sent from the updated database 53 in the central server 51 across the network to update the displays 55 and 57 of client computers 50 and 52, respectively." 4:26-29. As another example, returning to Figure 6 wherein computer 62 becomes the "server," computer 62 receives events from player B on computer 62 or from other players (*e.g.*, player A on computer 60), updates master database 72, and generates a view to display on screen 7 4 of computer 62. *See* 7: 1-8 ("[A]n event is input to the master event server, either from a user or from an open connection (step 822). If the event is determined to be a local event (step 823), then the event is sent only to the master database."); 4:50-55 ("By providing slave copies of the database in every other machine involved in the game, the need to send rendering commands across the network is reduced or eliminated, which reduces bandwidth consumption."). For example, *see* the following passages and/or figures, as well as all related disclosures: • "A method and apparatus for reducing network latency during execution of a multiple-player game across a computer network are provided. A master database represents a world model for the game, and a master event server sequences user inputs, or events, for updating the master database. The master event server and the master database are initially located on a particular node in the network. Every other node that is used by a player in the game has a slave event server and a slave database. Slave event servers are responsible for updating their local slave database, sending events from their local node to the master event server, and forwarding events to and from other slave servers. In the method, a

Claim Language	Prior Art References
	determination is made of which player in the game has a role in the game requiring the least latency of any role and which node in the network is being used by that player. The master event server and the master database are then migrated to that node by changing the status of the slave server of that node to that of master event server. Once the master event server has been migrated, other slave servers can establish a direct connection with the new master event server." Abstract. • "One such problem is that even computer games which are designed for multiple players using different computers tend to be sensitive to latency in communicating data between computers. Consider, for example, a computer game in which two or more players at remote locations interact with each other within a fictitious "game world." The game world can be modeled by a database, which might be located on one of the player's computers or on a central server. The players interact with each other by entering inputs through a joystick, mouse, keyboard, or other input/output device. In order for the game to be enjoyable, the game world must accurately reflect the current history of user inputs, or "events," received from all players. To resolve ambiguity of event ordering, events are typically collected at a central server that establishes the event sequence. However, differences in delays between the various computers involved in the game can hinder the proper sequencing of events when updating the game world. In addition, the speed with which events are communicated to and from the game world (*i.e.*, the database) can be critical, depending upon the type of game being played. For example, in naturally fast-paced games, such as those involving close combat between characters and many simulations, high latency can cause the game to be unrealistic, if not completely unplayable. High latency can be quite problematic even in games in which only particular players have latency-sensitive roles." 1:23-48. • "Referring now to FIG. 1, a network configuration in which the present invention may be implemented is illustrated. A central server computer 1 is coupled via the Internet 3 to a number of client computer systems 2. The server computer system 1 and each of the client computer systems 2 includes stored game software, including software in which the present invention is embodied. The game software includes various application program interfaces (APIs) providing an interface to operating system software. FIG. 2 illustrates an example

Claim Language	Prior Art References
	of an architecture which may be used for any of the client computer systems 2 or the server computer system 1. The architecture includes a central processing unit (CPU) 10, memory 11, a mass storage device 12, a modem 13, and various input/output (I/O) devices 14. Memory 11 represents both random access memory (RAM) and read-only memory (ROM). Mass storage device 12 includes an optical, magnetic, or other similar storage medium suitable for use with a personal computer system. I/O devices 14 may include any combination of devices such as a mouse, a joystick, a trackball, a keyboard, a monitor, or other I/O devices. The software in which the present invention is embodied may be stored in mass storage device 12, memory 11, or both. In addition, the software may be distributed between two or more of the computer systems illustrated in FIG. 1. FIG. 4A shows a representation of a computer system 40 running game software according to one prior art embodiment. A database 43 provides a world model for the game (*i.e.*, a "game world"). User inputs 41 are received through a game-compatible I/O device 46 (such as a mouse or joystick) and provided to software 42. Software 42 receives and sequences user inputs, or "events," that are used to update the database 43. Rendering commands 45 are sent from the database to a display 44 of the computer system 40 to generate various scenes representing the game world. Techniques for implementing a game as illustrated in FIG. 4A are well-known. FIG. 4B shows a well-known configuration for implementing a multi-user game on a network. In the configuration of FIG. 4B, the game world is provided by a database 53 located on a central server 51. Client computer systems 50 and 52 are used by Players A and B of the game, respectively. Software 54 located in the central server 51 receives events 58 from computers 50 and 52 and provides these events to the database 53 in order to update the game world. Rendering commands 56 are then sent from the updated database 53 in the central server 51 across the network to update the displays 55 and 57 of client computers 50 and 52, respectively. As mentioned above, implementing as shown in FIG. 4B on the Internet can be problematic if either of Players A and B are subject to a significant latency restriction. In addition, the transmission of rendering commands 56 from server 51 to clients 50 and 52 can be hindered b bandwidth limitations of the network. Consequently, the present invention provides a technique for overcoming such difficulties." 3:48-4:36.

Claim Language	Prior Art References
	• "In the technique of the present invention, every machine that participates in the game runs software referred to herein as an "event server." One of the event servers is known as the "master event server" and is responsible for sequencing all changes to a master database representing the game world. All event servers other than the master event server are considered to be "slave event servers". In the preferred embodiment, the master database is located on the same machine as the master event server. Slave copies of the master database (or portions thereof) are stored on each machine other than the machine in which the master event server is located. By providing one master copy of the database, events originating from different machines can be more easily synchronized. By providing slave copies of the database in every other machine involved in the game, the need to send rendering commands across the network is reduced or eliminated, which reduces bandwidth consumption. In one embodiment of the present invention, the master database is divided into multiple sections, or "objects", each of which may reside on a different machine. Each section functions as a master database relating to a particular aspect or subset of the game, and a separate master event server is associated with each section. An event server generally performs the following primary functions: (1) connecting to other event servers; (2) routing events to the local database and to other event servers; (3) migrating the master event server; and (4) reconnecting to a new event server such that events are not lost or duplicated. Each event server maintains a list of open connections to other event servers, one of which is always tagged as the route to the master (R TM) event server. As will be described in greater detail below, the master event server is migrated to a machine used by the player having the role with the most restrictive latency requirement. In the embodiment described above in which there are multiple master event servers corresponding to multiple sections of the master database, each master event server can be independently migrated to another machine, as appropriate. Further, all machines participating in the game which do not have the master event server are then able to reconnect to the machine which has the master event server or to any other machine in order to improve network latency." 4:37-5:16. • "FIG. SA illustrates a technique according to the present invention for implementing a multi-user game over the Internet. In FIG. SA, the game is played by two players, players

Claim Language	Prior Art References
	A and B, operating computer systems 60 and 62, respectively. It is noted, however, that the present invention is equally applicable to a game played simultaneously by more than two players. Initially, the master database 71 and the master event server 66 are located in a central server computer system 61. Located in computer system 60 (used by player A) are a slave event queue 65 and a slave copy 68 of master database 71. Similarly, located in computer system 62 (used by player B) is a slave event server 67 and a slave copy 72 of master database 71. The display 74 of computer system 62 is updated by rendering commands 73 from the slave copy 72 of master database 71, rather than from the master database 71 itself, thereby reducing bandwidth consumption on the network. Similarly, the display 70 associated with computer system 60 is updated by rendering commands 69 from slave copy 68 of master database 71, rather than directly from master database 71. Events 64 from computer system 62 are received by slave event server 67 and then passed along to the master event server 66, unless the events 64 are purely local events for updating only the slave copy 72 of the database. Purely local events might be, for example, events which set display options or other functions that are only relevant to computer system 62. Similarly, events 63 originating from computer system 60 are received first by slave event server 65 and then passed along to master event server 66, unless those events are purely local to computer system 60. Events received by the master event server 66 are used to update the master database 71 and are passed along to all slave event servers as master events 75." 5:17-49. • "According to the present invention, a determination is made of which computer system participating in the game is used by the player having the role which requires the least latency. Once this is determined, the master event server 66 is migrated to the machine used by that player. Referring now to FIG. 5B, assume that it has been determined that player B (using computer system 62) has the role in the game which requires the least latency. Consequently, master event server 66 is migrated to computer system 62. More specifically, the status of event server 66 is changed from master event server to slave event server, and the status of event server 67 is changed from slave event server to master event server. Similarly, the master database 71 is migrated to computer system 62. The status of slave event server 65 associated with Player A remains slave event server. Events 7 5 output by (newly designated) master event server 67 are then considered to be master

Claim Language	Prior Art References
	events, rather than slave events. As mentioned above, the present invention is equally applicable in a situation in which many players using many computers are involved in the game using the Internet as a communication medium. The logical configuration of computers involved in the game in that case would likely have the form of a tree hierarchy, as illustrated in FIG. 3. FIG. 3 illustrates the logical configuration of several computer systems 20 through 26. Computer system 20 initially contains the master event server and the master database and is therefore shown as the root node of the tree hierarchy. In a game situation involving two or more players, a given computer may not be directly connected to the computer having the master event server; such is the case for computers 23 through 26 in FIG. 3. Consequently, certain computers, such as computers 21, 22 and 25, will be 30 required to forward events received from other computers. The present invention provides a technique for handling such forwarding, which is discussed below in connection with FIGS. 8A and 8B. The present invention includes steps by which the slave event server sends slave events to the master event server over the network, and by which the master event server sends (multi-casts) master events over the network to every slave event server in the game. The master event server has the responsibility of ensuring that slave events originating from any given machine are processed in the order in which they were sent. This is a concern, since it is possible for two slave events to arrive out of order when the master event server migrates or when a slave event server reconnects to a different event server (*e.g.*, after migration of the master event server)." 5:65-6:45. • "FIG. 8B illustrates a routine used by a master event server for opening connections and routing events to slave event servers. Initially, the master event server accepts any new connections and adds the connections to its connection list (step 821). Next, an event is input to the master event server, either from a user or from an open connection (step 822). If the event is determined to be a local event (step 823), then the event is sent only to the master database. However, if the event is not a local event, then the event is considered to be, and is marked as, a master event (step 825). Hence, when a slave event is received and processed by the master event server, it effectively becomes a master event once it is retransmitted (multi-cast) by the master event server. Once the event is marked as a master event, the master event server checks the event number associated with that event to verify that no earlier events from the sending event server are missing (step 826). It is possible for

Claim Language	Prior Art References
	events to be missing in situations in which a sending slave event server has recently reconnected to a newly assigned master event server. In that situation, two events sent by the slave event server may take different routes to their destination and may therefore arrive out of sequence. Consequently, once all earlier events have been processed (step 827), the event is sent to the local database and to every event server listed in the connection list of the master event server (step 828)." 7:1-25. • "Referring again to FIG. 5B, once the master event server has been migrated to computer system 62, it may not be desirable from a latency standpoint (or for other reasons) to require events originating from computer system 60 to pass through the central server computer system 61. For example, once the master event server has been migrated to a different machine, it may be appropriate for one slave event server to reconnect to another slave event server to reduce its latency. Referring to both FIG. 5B and FIG. 6, note that slave event server 65 on computer 60 was initially connected to event server 66 on computer 61 (FIG. 5B). Event server 66 was initially the master event server. However, once event server 67 in computer 62 becomes the master event server, it becomes more efficient for slave event server 65 to connect directly to event server 67 (FIG. 6). Consequently, the present invention allows any given slave event server in the game to alter its connections in order to improve network latency. For example, slave event server can establish a new connection directly to the newly designated master event server 67 and drop its existing connection to the former master event server 66, as illustrated in FIG. 6." 8:52-9:5. One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in

Claim Language	Prior Art References
	Carvana's Invalidity Contentions Cover Pleading, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart. Specific prior art combinations (in conjunction with the knowledge of a POSITA) are as follows: <u>Roy+ Rover</u> Carvana incorporates by reference the '719 invalidity claim chart for Rover: A Toolkit for Mobile Information Access ("Rover"), which includes exemplary disclosures relevant to claim element [1(a)]. To the extent IBM argues that Roy does not disclose "configuring the server to store a model associated with the application and to execute view-generating and controller logic associated with the application" and/or that the server does not actually execute that VGL and CL, a POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Rover's teachings about moving application data and logic between a client and server using relocatable dynamic objects ("RDOs"). A POSITA would have been motivated to modify the online gaming system described in Roy based on Rover's teachings. For example, Roy and Rover are in the same field of endeavor (interactive electronic information delivery systems over a network). *See* Roy, Abstract ("A method and apparatus for reducing network latency during execution of a multiple-player game across a computer network are provided."); Rover at 156 ("The Rover toolkit offers applications a uniform distributed object system based on a client/server architecture .... [The toolkit] combines relocatable dynamic objects and queued remote procedure calls to provide unique services for 'roving' mobile applications."). A POSITA examining Roy and the online multiplayer game system described therein would be motivated to modify the client/server architecture of that game system depending on the specific requirements for a given game. For example, a POSITA would understand that a fast-paced first person shooter would (or could) require a different architecture than, for example, a lighter-weight card game. As such, a POSITA would *see*k to distribute the program logic effectively and efficiently between the server and client devices. Rover discloses a client/server architecture that utilizes RDOs to move logic between the server and client as desired by the application designer.

Claim Language	Prior Art References
	For example, Rover explains that "[t]he Rover toolkit combines relocatable dynamic objects and queued remote procedure calls to provide unique services for 'roving' mobile applications." Rover at 156. "A relocatable dynamic object is an object with a well-defined interface that can be dynamically loaded into a client computer from a server computer (or vice versa) to reduce client-server communication requirements." *Id.* "The Rover toolkit also offers advantages for workstation applications by providing a uniform distributed object architecture for code shipping, object caching, and asynchronous object invocation." *Id.* "Depending on the power of the mobile host and the available bandwidth, Rover dynamically adapts and moves functionality between the client and server." *Id.* at 158. "By migrating a GUI to the client, input events can be processed locally, even if the client becomes disconnected." *Id.* In light of at least the foregoing, a POSITA would understand that data and application logic for an online multiplayer game described by Roy could be allocated to either the server or the client, and would be motivated to incorporate Rover's method for distributing that data and logic (including model data, CL, VGL, and any other logic) because, for example, "[d]epending on the power of the [client] and the available bandwidth, Rover dynamically adapts and moves functionality between the client and server." *Id.* at 158. <u>Roy + Bennett</u> Carvana incorporates by reference the '719 invalidity claim chart for Experience With Distributed Smalltalk by John K. Bennett ("Bennett "), which includes exemplary disclosures relevant to claim element [1(a)]. To the extent IBM argues that Roy does not disclose "configuring the server to store a model associated with the application and to execute view-generating and controller logic associated with the application " and/or that the server does not actually execute that VGL and CL, a POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Bennett's teachings about building distributed applications using objects. A POSITA would have been motivated to modify the online gaming system described in Roy based on Bennett's teachings. For example, Roy and Bennett are in the same field of endeavor (*e.g.*, distributed applications). *See* Roy, Abstract ("A method and apparatus for reducing network latency during execution of a multiple-player game across a computer network are provided."); Bennett at 157 ("Distributed Smalltalk (DS) is an implementation of Smalltalk that supports the interaction of many Smalltalk users on many machines. It provides communication and interaction

Claim Language	Prior Art References
	among remote Smalltalk users, direct access to remote objects, and a degree of object sharing among users. . . . This paper concentrates on . . . how DS is used to build distributed applications. Three example applications are described: remote file access, animated distributed objects, and offloading a compute-intensive task to a non-Smalltalk host."). A POSITA examining Roy and the online multiplayer game system described therein would be motivated to modify the client/server architecture of that game system depending on the specific requirements for a given game. For example, a POSITA would understand that a fast-paced first person shooter would (or could) require a different architecture than, for example, a lighter-weight card game. As such, a POSITA would *see*k to distribute the program logic effectively among the server and client devices. Bennett discloses a method that enables an application designer to distribute objects containing data and/or logic between servers and client devices, and a POSITA would be motivated to use Bennett's methods. For example, Distributed Smalltalk "provides communication and interaction among geographically remote Smalltalk users, direct access to remote objects, the ability to construct distributed applications in the Smalltalk environment, and a degree of object sharing among users." Bennett at 157. "Potential applications of Distributed Smalltalk include mail systems, remote computation servers, remote file servers, and collaborative software development." *Id.* "The distributed aspects of the system are largely user transparent and preserve the reactive quality of Smalltalk objects." *Id.* As another example, as described by the '719 Patent, applications can be designed "in terms of the MVC (Model-View-Controller) paradigm" by using the programming language Smalltalk. '719 Patent, Background. "Smalltalk is an object-oriented programming language and highly interactive uniformly object-structed programming environment." Bennett at 157. The '719 Patent incorporates by reference A Cookbook for Using the Model-View-Controller User Interface Paradigm in the Smalltalk-BO System by Glenn E. Krasner and Stephen T. Pope ("Pope"). Pope teaches about the "the Model-View-Controller (MVC) programming paradigm and methodology used in Smalltalk-80 programming system." Pope at 2; *see also* Bennett at 161 ("The [object-oriented programming] of these classes are part of the Smalltalk-80 ... a registered trademark of ParcPlace systems."). Pope describes the benefits of using Smalltalk: "The user

Claim Language	Prior Art References
	interface of the Smalltalk-80 programming environment[] was developed using a particular strategy of representing information, display, and control." Pope at 3. "This strategy was chosen to satisfy two goals: (1) to create the special set of system components needed to support a highly interactive software development process, and (2) to provide a general set of system components that make it possible for programmers to create portable interactive graphical applications easily." *Id.* Pope further describes the benefits of MVC programming. For example: **MVC and the Issues of Reusability and Pluggability** When building interactive applications, as with other programs, modularity of components has enormous benefits. Isolating functional units from each other as much as possible makes it easier for the application designer to understand and modify each particular unit, without having to know everything about the other units. Our experiences with the Smalltalk-76 programming system showed that one particular form of modularity--a three-way separation of application components--has payoff beyond merely making the designer's life easier. This three-way division of an application entails separating (1) the parts that represent the model of the underlying application domain from (2) the way the model is presented to the user and from (3) the way the user interacts with it. Model-View-Controller (MVC) programming is the application of this three-way factoring, whereby objects of different classes take over the operations related to the application domain (the model), the display of the application's state (the view), and the user interaction with the model and the view (the controller). In earlier Smalltalk system user interfaces, the tools that were put into the interface tended to consist of arrangements of four basic viewing idioms: paragraphs of text, lists of text (menus), choice "buttons," and graphical forms (bit- or pixel-maps). These tools also tended to use three basic user interaction paradigms: browsing, inspecting and editing. A goal of the current Smalltalk-80 system was to be able to define user interface components for handling these idioms and paradigms once, and share them among all the programming environment tools and user-written applications using the methodology of MVC programming. Pope at 3. Bennett's Distributed Smalltalk "is an implementation of Smalltalk." Bennett at 157. Thus, because a POSITA would understand that Smalltalk could be used to design applications using an MVC paradigm, a POSITA would similarly understand that Distributed Smalltalk could be used to design applications using an MVC paradigm. At least in light of the foregoing, a POSITA would be motivated to create and/or modify the online gaming system described by Roy by using Distributed Smalltalk. For example, a POSITA may *see*k to "offload[] a compute-intensive task" to the server. Bennett at 157. Similarly, a POSITA could distribute DS objects

Claim Language	Prior Art References
	containing data and logic (including model data, VGL, CL, or any other logic) between the client and server depending on the requirements for a specific application, and would understand that the application could be implemented per the MVC paradigm in view of Bennett's use of Smalltalk. <u>Roy + Hurly</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of The Videotex and Teletex Handbook by Paul Hurly ("Hurly"). Hurly was published in 1985 and qualifies as prior art to the '849 patent under at least 35 U.S.C. §§ 102(a) and/or (b). A POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Hurly's teachings about downloading data and/or logic on demand from a server to a client computer. For example, *see* claim 12. For example, Roy and Hurly are in the same field of endeavor (interactive electronic information delivery systems over a network). *See* Roy, Abstract ("A method and apparatus for reducing network latency during execution of a multiple-player game across a computer network are provided."); Hurly at 18 ("Viewdata is the generic name for fully interactive videotex. This means that the user's requests for information, or service, are actually sent to, received by, and acted on by the centralized computer."). As another example, Hurly describes how games such as backgammon can be played online. Hurly at 168. <u>Roy + Lowery</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of Lowery. *See* '719 Patent Invalidity Chart, Ex. D-1 (Lowery) at claim limitation 1(a) (incorporated herein by reference). A POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Lowery's teachings about downloading a subset of application logic over a network on demand. For example, *see* claim 12. <u>Roy + (Rover or Bennett) + Wiltshire</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of either Rover or Bennett and further in view of Wiltshire. *See*

Claim Language	Prior Art References
	'719 Patent Invalidity Chart, Ex. D-11 (Wiltshire) at claim limitation 1(a) (incorporated herein by reference). A POSITA would have been motivated to modify the online gaming system described in Roy based on Wiltshire's teachings. For example, Roy and Wiltshire are in the same field of endeavor (online gaming systems). *See* Roy, Abstract ("A method and apparatus for reducing network latency during execution of a multiple-player game across a computer network are provided."); Wiltshire, Abstract ("The computer gaming system of the present invention allows for transparent modifications and upgrades to the gaming programs by executing gaming programs on a server/host computer connected to a plurality of client/terminal computers via communication pathways."). A POSITA would understand from Wiltshire that online gaming systems could execute CL and/or VGL on a server. For example, Wiltshire teaches a client/server architecture for online games wherein most of the VGL and CL is performed on a server. "[T]he gaming programs are executed entirely on the server/host computer, with only wagering input and display output operations executed on the client/terminal computers." Wiltshire, Abstract. In this configuration, "the cost of hardware and software required for each client/terminal computer is greatly reduced." *Id.* And a "patron of a client/terminal computer can access any of the gaming programs executed on the server/host computer." *Id.* In light of at least the foregoing, a POSITA would understand that executing CL and/or VGL on the server could be beneficial depending on the application (*e.g.*, type of game) and would be motivated to distribute CL and/or VGL to the server for an online game as described by Roy depending on the specific requirements of that game. For example, a POSITA would understand that a game could have high processing requirements for VGL and would thus distribute more VGL to the server computer with more processing power. In any case, at minimum, a POSITA would be motivated to at least experiment with distributing CL and/or VGL to the server in light of Wiltshire. For example, a POSITA could distribute the CL and/or VGL as objects as described by Rover and/or Bennett.

Claim Language	Prior Art References
	Roy + History of PHP To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of History of PHP. "PHP as it's known today is actually the successor to a product named PHP/FI. Created in 1994 by Rasmus Lerdorf, the very first incarnation of PHP was a simple set of Common Gateway Interface (CGI) binaries written in the C programming language. Originally used for tracking visits to his online resume, he named the suite of scripts "Personal Home Page Tools," more frequently referenced as "PHP Tools." Over time, more functionality was desired, and Rasmus rewrote PHP Tools, producing a much larger and richer implementation. This new model was capable of database interaction and more, providing a framework upon which users could develop simple dynamic web applications such as guestbooks. In June of 1995, Rasmus » released the source code for PHP Tools to the public, which allowed developers to use it as they saw fit. This also permitted - and encouraged - users to provide fixes for bugs in the code, and to generally improve upon it." A POSITA would have been motivated to modify the teachings in Roy based on History of PHP's teachings about server side scripting for dynamic page creation. One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Supplemental Invalidity Contentions, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart.
1(b). configuring the client to store at least a subset of the model associated with the application and to execute at	Roy configures the client (*e.g.*, computer system 60) to store at least a subset of the model associated with the application (*i.e.*, the online multiplayer game) and to execute at least a subset of the VGL and CL associated with the application.

Claim Language	Prior Art References
least a subset of the view-generating and controller logic associated with the application;	Roy configures the client (*e.g.*, computer system 60) to store at least a subset of the model associated with the online game. As noted, the full model is contained in the "master database" which is preferably "located on the same machine as the master event server." 4:44-45. Copies of that database, or subsets thereof, are stored on the clients. In particular, "[s]lave copies of the master database (or portions thereof) are stored on each machine other than the machine in which the master event server is located." 4:47-48. Roy further configures the client (*e.g.*, computer system 60) to execute at least a subset of the CL, and the client does, in fact, execute the subset of CL. For example, computer system 60 executes controller logic to process events and determine whether they should be passed along to the master event server or whether they are "local events" that are used to update the local model stored in database 70. *See* 5: 17-49. This routing is done by an "event server 65" on computer system 60. Every computer system connected to the game has an "event server" and each is tasked with "routing events to the local database and to other event servers." 4:62- 66. And the master event server can be moved between the central server 61 and any computer device (*e.g.*, computer systems 60 and 62). Thus, the same CL can be copied and executed on any computer device in the system. Specifically, "a determination is made of which computer system participating in the game is used by the player having the role which requires the least latency. Once this is determined, the master event server 66 is migrated to the machine used by that player. ... Similarly, the master database 71 is migrated to computer system 62." 6: 1-11. Further, it would be understood by a POSITA, or at the least be obvious to a POSITA, that the CL on each computer system would be the same for efficiency and consistency. For example, it would be obvious to a POSITA that the same CL would be used on each computer to process the "purely local events" such as those to "set display options or other functions that are purely local to computer system 60." 5:40-43. Furthermore, because any computer system may easily become the "master event server", a POSITA would understand that the CL on each computer would be the same. Roy further configures the client (*e.g.*, computer system 60) to execute at least a subset of the VGL, and the client does, in fact, execute the subset of VGL. As explained, "[s]lave copies of the master database (or portions thereof) are stored on each machine other than the machine in which the master event server is located." 4:47-48. Roy further explains that "[b]y providing slave copies of the database in every other machine involved in the game, the need to send rendering

Claim Language	Prior Art References
	commands across the network is reduced or eliminated, which reduces bandwidth consumption." 4:51-54. Thus, every computer system that is part of the game (including the one configured to be the server) executes VGL to create a view for the respective player. It would be understood to a POSITA, or at least rendered obvious, that the VGL is the same on each computer system for the sake of efficiency and consistency. *See* 5:17-44 ("The display 74 of computer system 62 is updated by rendering commands 73 from the slave copy 72 of master database 71, rather than from the master database 71 itself, thereby reducing bandwidth consumption on the network. Similarly, the display 70 associated with computer system 60 is updated by rendering commands 69 from slave copy 68 of master database 71, rather than directly from master database 71."). This is particularly true where the disclosed application of Roy's invention is multiplayer games where each computer shows the same world model to individual players of the game. *See* 3: 16-4 7. A POSITA would understand that, for a given game, the VGL used to render views of the same model would be the same at each user's computer (*e.g.*, the client and the computer designated as the master). For example, *see* the following passages and/or figures, as well as all related disclosures: • "A method and apparatus for reducing network latency during execution of a multiple-player game across a computer network are provided. A master database represents a world model for the game, and a master event server sequences user inputs, or events, for updating the master database. The master event server and the master database are initially located on a particular node in the network. Every other node that is used by a player in the game has a slave event server and a slave database. Slave event servers are responsible for updating their local slave database, sending events from their local node to the master event server, and forwarding events to and from other slave servers. In the method, a determination is made of which player in the game has a role in the game requiring the least latency of any role and which node in the network is being used by that player. The master event server and the master database are then migrated to that node by changing the status of the slave server of that node to that of master event server. Once the master event server has been migrated, other slave servers can establish a direct connection with the new master event server." Abstract.

Claim Language	Prior Art References
	• "Referring now to FIG. 1, a network configuration in which the present invention may be implemented is illustrated. A central server computer 1 is coupled via the Internet 3 to a number of client computer systems 2. The server computer system 1 and each of the client computer systems 2 includes stored game software, including software in which the present invention is embodied. The game software includes various application program interfaces (APIs) providing an interface to operating system software. FIG. 2 illustrates an example of an architecture which may be used for any of the client computer systems 2 or the server computer system 1. The architecture includes a central processing unit (CPU) 10, memory 11, a mass storage device 12, a modem 13, and various input/output (I/O) devices 14. Memory 11 represents both random access memory (RAM) and read-only memory (ROM). Mass storage device 12 includes an optical, magnetic, or other similar storage medium suitable for use with a personal computer system. I/O devices 14 may include any combination of devices such as a mouse, a joystick, a trackball, a keyboard, a monitor, or other I/O devices. The software in which the present invention is embodied may be stored in mass storage device 12, memory 11, or both. In addition, the software may be distributed between two or more of the computer systems illustrated in FIG. 1. FIG. 4A shows a representation of a computer system 40 running game software according to one prior art embodiment. A database 43 provides a world model for the game (*i.e.*, a "game world"). User inputs 41 are received through a game-compatible I/O device 46 (such as a mouse or joystick) and provided to software 42. Software 42 receives and sequences user inputs, or "events," that are used to update the database 43. Rendering commands 45 are sent from the database to a display 44 of the computer system 40 to generate various scenes representing the game world. Techniques for implementing a game as illustrated in FIG. 4A are well-known. FIG. 4B shows a well-known configuration for implementing a multi-user game on a network. In the configuration of FIG. 4B, the game world is provided by a database 53 located on a central server 51. Client computer systems 50 and 52 are used by Players A and B of the game, respectively. Software 54 located in the central server 51 receives events 58 from computers 50 and 52 and provides these events to the database 53 in order to update the game world. Rendering commands 56 are then sent from the updated database 53 in the central server 51 across the network to update the displays 55 and 57 of client computers 50 and 52, respectively. As mentioned above, implementing

Claim Language	Prior Art References
	as shown in FIG. 4B on the Internet can be problematic if either of Players A and B are subject to a significant latency restriction. In addition, the transmission of rendering commands 56 from server 51 to clients 50 and 52 can be hindered b bandwidth limitations of the network. Consequently, the present invention provides a technique for overcoming such difficulties." 3:48- 4:36. • "In the technique of the present invention, every machine that participates in the game runs software referred to herein as an "event server." One of the event servers is known as the "master event server" and is responsible for sequencing all changes to a master database representing the game world. All event servers other than the master event server are considered to be "slave event servers". In the preferred embodiment, the master database is located on the same machine as the master event server. Slave copies of the master database (or portions thereof) are stored on each machine other than the machine in which the master event server is located. By providing one master copy of the database, events originating from different machines can be more easily synchronized. By providing slave copies of the database in every other machine involved in the game, the need to send rendering commands across the network is reduced or eliminated, which reduces bandwidth consumption. In one embodiment of the present invention, the master database is divided into multiple sections, or "objects", each of which may reside on a different machine. Each section functions as a master database relating to a particular aspect or subset of the game, and a separate master event server is associated with each section. An event server generally performs the following primary functions: (1) connecting to other event servers; (2) routing events to the local database and to other event servers; (3) migrating the master event server; and (4) reconnecting to a new event server such that events are not lost or duplicated. Each event server maintains a list of open connections to other event servers, one of which is always tagged as the route to the master (R TM) event server. As will be described in greater detail below, the master event server is migrated to a machine used by the player having the role with the most restrictive latency requirement. In the embodiment described above in which there are multiple master event servers corresponding to multiple sections of the master database, each master event server can be independently migrated to another machine, as appropriate. Further, all machines participating in the game which do not have the master event server are then able to

Claim Language	Prior Art References
	reconnect to the machine which has the master event server or to any other machine in order to improve network latency." 4:37-5:16. • "FIG. SA illustrates a technique according to the present invention for implementing a multi-user game over the Internet. In FIG. SA, the game is played by two players, players A and B, operating computer systems 60 and 62, respectively. It is noted, however, that the present invention is equally applicable to a game played simultaneously by more than two players. Initially, the master database 71 and the master event server 66 are located in a central server computer system 61. Located in computer system 60 (used by player A) are a slave event queue 65 and a slave copy 68 of master database 71. Similarly, located in computer system 62 (used by player B) is a slave event server 67 and a slave copy 72 of master database 71. The display 74 of computer system 62 is updated by rendering commands 73 from the slave copy 72 of master database 71, rather than from the master database 71 itself, thereby reducing bandwidth consumption on the network. Similarly, the display 70 associated with computer system 60 is updated by rendering commands 69 from slave copy 68 of master database 71, rather than directly from master database 71. Events 64 from computer system 62 are received by slave event server 67 and then passed along to the master event server 66, unless the events 64 are purely local events for updating only the slave copy 72 of the database. Purely local events might be, for example, events which set display options or other functions that are only relevant to computer system 62. Similarly, events 63 originating from computer system 60 are received first by slave event server 65 and then passed along to master event server 66, unless those events are purely local to computer system 60. Events received by the master event server 66 are used to update the master database 71 and are passed along to all slave event servers as master events 75." 5:17-49. • "According to the present invention, a determination is made of which computer system participating in the game is used by the player having the role which requires the least latency. Once this is determined, the master event server 66 is migrated to the machine used by that player. Referring now to FIG. 5B, assume that it has been determined that player B (using computer system 62) has the role in the game which requires the least latency. Consequently, master event server 66 is migrated to computer system 62. More

Claim Language	Prior Art References
	specifically, the status of event server 66 is changed from master event server to slave event server, and the status of event server 67 is changed from slave event server to master event server. Similarly, the master database 71 is migrated to computer system 62. The status of slave event server 65 associated with Player A remains slave event server. Events 7 5 output by (newly designated) master event server 67 are then considered to be master events, rather than slave events. As mentioned above, the present invention is equally applicable in a situation in which many players using many computers are involved in the game using the Internet as a communication medium. The logical configuration of computers involved in the game in that case would likely have the form of a tree hierarchy, as illustrated in FIG. 3. FIG. 3 illustrates the logical configuration of several computer systems 20 through 26. Computer system 20 initially contains the master event server and the master database and is therefore shown as the root node of the tree hierarchy. In a game situation involving two or more players, a given computer may not be directly connected to the computer having the master event server; such is the case for computers 23 through 26 in FIG. 3. Consequently, certain computers, such as computers 21, 22 and 25, will be 30 required to forward events received from other computers. The present invention provides a technique for handling such forwarding, which is discussed below in connection with FIGS. 8A and 8B. The present invention includes steps b which the slave event server sends slave events to the master event server over the network, and by which the master event server sends (multi-casts) master events over the network to every slave event server in the game. The master event server has the responsibility of ensuring that slave events originating from any given machine are processed in the order in which they were sent. This is a concern, since it is possible for two slave events to arrive out of order when the master event server migrates or when a slave event server reconnects to a different event server (*e.g.*, after migration of the master event server)." 5:65-6:45. • "Referring again to FIG. 5B, once the master event server has been migrated to computer system 62, it may not be desirable from a latency standpoint (or for other reasons) to require events originating from computer system 60 to pass through the central server computer system 61. For example, once the master event server has been migrated to a different machine, it may be appropriate for one slave event server to reconnect to another slave event server to reduce its latency. Referring to both FIG. 5B and FIG. 6, note that

Claim Language	Prior Art References
	slave event server 65 on computer 60 was initially connected to event server 66 on computer 61 (FIG. 5B). Event server 66 was initially the master event server. However, once event server 67 in computer 62 becomes the master event server, it becomes more efficient for slave event server 65 to connect directly to event server 67 (FIG. 6). Consequently, the present invention allows any given slave event server in the game to alter its connections in order to improve network latency. For example, slave event server can establish a new connection directly to the newly designated master event server 67 and drop its existing connection to the former master event server 66, as illustrated in FIG. 6." 8:52-9:5. One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Invalidity Contentions Cover Pleading, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart. Specific prior art combinations (in conjunction with the knowledge of a POSITA) are as follows: <u>Roy+ Rover</u> Carvana incorporates by reference the '719 invalidity claim chart for Rover: A Toolkit for Mobile Information Access ("Rover"), which includes exemplary disclosures relevant to claim element 1(b). To the extent IBM argues that Roy does not disclose "configuring the client to store at least a subset of the model associated with the application and to execute at least a subset of the view-generating and controller logic associated with the application " and/or that the client does not actually execute that VGL and CL, a POSITA would have been motivated to modify the online

Claim Language	Prior Art References
	multiplayer gaming system in Roy based on Rover's teachings about moving application logic between a client and server using relocatable dynamic objects ("RDOs"). A POSITA would have been motivated to modify the online gaming system described in Roy based on Rover's teachings. For example, Roy and Rover are in the same field of endeavor (interactive electronic information delivery systems over a network). *See* Roy, Abstract ("A method and apparatus for reducing network latency during execution of a multiple-player game across a computer network are provided."); Rover at 156 ("The Rover toolkit offers applications a uniform distributed object system based on a client/server architecture. . . . [The toolkit] combines relocatable dynamic objects and queued remote procedure calls to provide unique services for 'roving' mobile applications."). A POSITA examining Roy and the online multiplayer game system described therein would be motivated to modify the client/server architecture of that game system depending on the specific requirements for a given game. For example, a POSITA would understand that a fast-paced first person shooter would (or could) require a different architecture than, for example, a lighter-weight card game. As such, a POSITA would *see*k to distribute the program logic effectively and efficiently between the server and client devices. Rover discloses a client/server architecture that utilizes RDOs to move logic between the server and client as desired by the application designer. For example, Rover explains that "[t]he Rover toolkit combines relocatable dynamic objects and queued remote procedure calls to provide unique services for 'roving' mobile applications." Rover at 156. "A relocatable dynamic object is an object with a well-defined interface that can be dynamically loaded into a client computer from a server computer (or vice versa) to reduce client-server communication requirements." *Id.* "The Rover toolkit also offers advantages for workstation applications by providing a uniform distributed object architecture for code shipping, object caching, and asynchronous object invocation." *Id.* "Depending on the power of the mobile host and the available bandwidth, Rover dynamically adapts and moves functionality between the client and server." *Id.* at 158. "By migrating a GUI to the client, input events can be processed locally, even if the client becomes disconnected." *Id.* In light of at least the foregoing, a POSITA would understand that data and application logic for an online multiplayer game described by Roy could be allocated to either the server or the client, and would be motivated to incorporate Rover's method for distributing that data and logic (including model data, CL, VGL, and any other logic)

Claim Language	Prior Art References
	because, for example, "[d]epending on the power of the [client] and the available bandwidth, Rover dynamically adapts and moves functionality between the client and server." *Id.* at 158. <u>Roy + Bennett</u> Carvana incorporates by reference the '719 invalidity claim chart for Experience With Distributed Smalltalk by John K. Bennett ("Bennett "), which includes exemplary disclosures relevant to claim element [1(a)]. To the extent IBM argues that Roy does not disclose "configuring the client to store at least a subset of the model associated with the application and to execute at least a subset of the view-generating and controller logic associated with the application " and/or that the client does not actually execute that VGL and CL, a POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Bennett's teachings about building distributed applications using objects. A POSITA would have been motivated to modify the online gaming system described in Roy based on Bennett's teachings. For example, Roy and Bennett are in the same field of endeavor (*e.g.*, distributed applications). *See* Roy, Abstract ("A method and apparatus for reducing network latency during execution of a multiple-player game across a computer network are provided."); Bennett at 157 ("Distributed Smalltalk (DS) is an implementation of Smalltalk that supports the interaction of many Smalltalk users on many machines. It provides communication and interaction among remote Smalltalk users, direct access to remote objects, and a degree of object sharing among users. . . . This paper concentrates on . . . how DS is used to build distributed applications. Three example applications are described: remote file access, animated distributed objects, and offloading a compute-intensive task to a non-Smalltalk host."). A POSITA examining Roy and the online multiplayer game system described therein would be motivated to modify the client/server architecture of that game system depending on the specific requirements for a given game. For example, a POSITA would understand that a fast-paced first person shooter would (or could) require a different architecture than, for example, a lighter-weight card game. As such, a POSITA would *see*k to distribute the program logic effectively among the server and client devices.

Claim Language	Prior Art References
	Bennett discloses a method that enables an application designer to distribute objects containing data and/or logic between servers and client devices, and a POSITA would be motivated to use Bennett's methods. For example, Distributed Smalltalk "provides communication and interaction among geographically remote Smalltalk users, direct access to remote objects, the ability to construct distributed applications in the Smalltalk environment, and a degree of object sharing among users." Bennett at 157. "Potential applications of Distributed Smalltalk include mail systems, remote computation servers, remote file servers, and collaborative software development." *Id.* "The distributed aspects of the system are largely user transparent and preserve the reactive quality of Smalltalk objects." *Id.* As another example, as described by the '719 Patent, applications can be designed "in terms of the MVC (Model-View-Controller) paradigm" by using the programming language Smalltalk. '719 Patent, Background. "Smalltalk is an object-oriented programming language and highly interactive uniformly object-structed programming environment." Bennett at 157. The '719 Patent incorporates by reference A Description of the Model-View-Controller User Interface Paradigm in the Smalltalk-BO System by Glenn E. Krasner and Stephen T. Pope ("Pope"). Pope teaches about the "the Model-View-Controller (MVC) programming paradigm and methodology used in Smalltalk-80 programming system." Pope at 2; *see also* Bennett at 161 ("The [object-oriented programming] of these classes are part of the Smalltalk-80 ... a registered trademark of ParcPlace systems."). Pope describes the benefits of using Smalltalk: "The user interface of the Smalltalk-80 programming environment[] was developed using a particular strategy of representing information, display, and control." Pope at 3. "This strategy was chosen to satisfy two goals: (1) to create the special set of system components needed to support a highly interactive software development process, and (2) to provide a general set of system components that make it possible for programmers to create portable interactive graphical applications easily." *Id.* Pope further describes the benefits of MVC programming. For example:

Claim Language	Prior Art References
	MVC and the Issues of Reusability and Pluggability When building interactive applications, as with other programs, modularity of components has enormous benefits. Isolating functional units from each other as much as possible makes it easier for the application designer to understand and modify each particular unit, without having to know everything about the other units. Our experiences with the Smalltalk-76 programming system showed that one particular form of modularity--a three-way separation of application components--has payoff beyond merely making the designer's life easier. This three-way division of an application entails separating (1) the parts that represent the model of the underlying application domain from (2) the way the model is presented to the user and from (3) the way the user interacts with it. Model-View-Controller (MVC) programming is the application of this three-way factoring, whereby objects of different classes take over the operations related to the application domain (the model), the display of the application's state (the view), and the user interaction with the model and the view (the controller). In earlier Smalltalk system user interfaces, the tools that were put into the interface tended to consist of arrangements of four basic viewing idioms: paragraphs of text, lists of text (menus), choice "buttons," and graphical forms (bit- or pixel-maps). These tools also tended to use three basic user interaction paradigms: browsing, inspecting and editing. A goal of the current Smalltalk-80 system was to be able to define user interface components for handling these idioms and paradigms once, and share them among all the programming environment tools and user-written applications using the methodology of MVC programming. Pope at 3. Bennett's Distributed Smalltalk "is an implementation of Smalltalk." Bennett at 157. Thus, because a POSITA would understand that Smalltalk could be used to design applications using an MVC paradigm, a POSITA would similarly understand that Distributed Smalltalk could be used to design applications using an MVC paradigm. At least in light of the foregoing, a POSITA would be motivated to create and/or modify the online gaming system described by Roy by using Distributed Smalltalk. For example, a POSITA may *see*k to "offload[] a compute-intensive task" to the server. Bennett at 157. Similarly, a POSITA could distribute DS objects containing data and logic (including model data, VGL, CL, or any other logic) between the client and server depending on the requirements for a specific application, and would understand that the application could be implemented per the MVC paradigm in view of Bennett's use of Smalltalk. <u>Roy+ Hurly</u>

Claim Language	Prior Art References
	To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of The Videotex and Teletex Handbook by Paul Hurly ("Hurly"). Hurly was published in 1985 and qualifies as prior art to the '849 patent under at least 35 U.S.C. §§ 102(a) and/or (b). A POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Hurly's teachings about downloading data and/or logic on demand from a server to a client computer. For example, *see* claim 12. For example, Roy and Hurly are in the same field of endeavor (interactive electronic information delivery systems over a network). *See* Roy, Abstract ("A method and apparatus for reducing network latency during execution of a multiple-player game across a computer network are provided."); Hurly at 18 ("Viewdata is the generic name for fully interactive videotex. This means that the user's requests for information, or service, are actually sent to, received by, and acted on by the centralized computer."). As another example, Hurly describes how games such as backgammon can be played online. Hurly at 168. <u>Roy+ Lowery</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of Lowery. *See* '719 Patent Invalidity Chart, Ex. D-1 (Lowery) at claim limitation 1(b) (incorporated herein by reference). A POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Lowery's teachings about downloading a subset of application logic over a network on demand. For example, *see* claim 12. <u>Roy + History of PHP</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of History of PHP. "PHP as it's known today is actually the successor to a product named PHP/FI. Created in 1994 by Rasmus Lerdorf, the very first incarnation of PHP was a simple set of Common Gateway Interface (CGI) binaries written in the C programming language. Originally used for tracking visits to his online resume, he named the suite of scripts "Personal Home Page Tools," more frequently referenced as "PHP Tools." Over time, more functionality was desired, and Rasmus rewrote PHP Tools, producing a much larger and richer implementation. This new model was capable of database interaction and more, providing a framework upon which users could develop simple dynamic web applications such as

Claim Language	Prior Art References
	guestbooks. In June of 1995, Rasmus » released the source code for PHP Tools to the public, which allowed developers to use it as they saw fit. This also permitted - and encouraged - users to provide fixes for bugs in the code, and to generally improve upon it." A POSITA would have been motivated to modify the teachings in Roy based on History of PHP's teachings about server side scripting for dynamic page creation. One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Supplemental Invalidity Contentions, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart.
1(c). wherein one or more portions of the application are performed at the client without the client having to interact with the server, and	Roy discloses that one or more portions of the application are performed at the client without the client having to interact with the server as described above regarding preamble 1(pre) (*e.g.*, "local events" are processed at the client without having to interact with the server). For example, *see* the following passages and/or figures, as well as all related disclosures: *See above* regarding claim preamble 1(pre) which is incorporated by reference herein. One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in

Claim Language	Prior Art References
	Carvana's Invalidity Contentions Cover Pleading, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart. Specific prior art combinations (in conjunction with the knowledge of a POSITA) are as follows: <u>Roy+ Rover</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of Rover. *See* '719 Patent Invalidity Chart, Ex. D-8 (Rover) at claim limitation 1(c) (incorporated herein by reference). A POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Rover's teachings about moving application data and logic between a client and server using relocatable dynamic objects ("RDOs"). For example, *see* claim limitations 1(a) and 1(b), and claim 12. <u>Roy + Bennett</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of Bennett. *See* '719 Patent Invalidity Chart, Ex. D-15 (Bennett) at claim limitation 1(c) (incorporated herein by reference). A POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Bennett's teachings about moving application data and logic between a client and server using objects. For example, *see* claim limitations 1(a) and 1(b). <u>Roy+ Hurly</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of The Videotex and Teletex Handbook by Paul Hurly ("Hurly"). Hurly was published in 1985 and qualifies as prior art to the '849 patent under at least 35 U.S.C. §§ 102(a) and/or (b). A POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Hurly's teachings about downloading data and/or logic on demand from a server to a client computer. For example, *see* claim 12. For example, Roy and Hurly are in the same field of endeavor (interactive electronic information delivery systems over a network).

Claim Language	Prior Art References
	See Roy, Abstract ("A method and apparatus for reducing network latency during execution of a multiple-player game across a computer network are provided."); Hurly at 18 ("View data is the generic name for fully interactive videotex. This means that the user's requests for information, or service, are actually sent to, received by, and acted on by the centralized computer."). As another example, Hurly describes how games such as backgammon can be played online. Hurly at 168. <u>Roy+ Lowery</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of Lowery. *See* '719 Patent Invalidity Chart, Ex. D-1 (Lowery) at claim limitation 1(c) (incorporated herein by reference). A POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Lowery's teachings about downloading a subset of application logic over a network on demand. For example, *see* claim 12.
1(d). further wherein the client and the server both locally maintain at least a portion of the model and execute the view-generating and controller logic associated therewith.	Roy discloses that the client and the server both locally maintain at least a portion of the model and execute the view-generating and controller logic associated therewith as described above regarding claim elements 1(a) and 1B. For example, *see* the following passages and/or figures, as well as all related disclosures: *See above* regarding claim elements 1(a) and 1(b) which are incorporated by reference herein. One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Invalidity Contentions Cover Pleading, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart.

Claim Language	Prior Art References
	Specific prior art combinations (in conjunction with the knowledge of a POSITA) are as follows: <u>Roy+ Rover</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of Rover. *See* '719 Patent Invalidity Chart, Ex. D-8 (Rover) at claim limitation 1(d) (incorporated herein by reference). A POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Rover's teachings about moving application data and logic between a client and server using relocatable dynamic objects ("RDOs"). For example, *see* claim limitations 1(a) and 1(b), and claim 12. <u>Roy + Bennett</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of Bennett. *See* '719 Patent Invalidity Chart, Ex. D-15 (Bennett) at claim limitation 1(d) (incorporated herein by reference). A POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Bennett's teachings about moving application data and logic between a client and server using objects. For example, *see* claim limitations 1(a) and lB. <u>Roy+ Hurly</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of The Videotex and Teletex Handbook by Paul Hurly ("Hurly"). Hurly was published in 1985 and qualifies as prior art to the '849 patent under at least 35 U.S.C. §§ 102(a) and/or (b). A POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Hurly's teachings about downloading data and/or logic on demand from a server to a client computer. For example, *see* claim 12. For example, Roy and Hurly are in the same field of endeavor (interactive electronic information delivery systems over a network). *See* Roy, Abstract ("A method and apparatus for reducing network latency during execution of a multiple-player game across a computer network are provided."); Hurly at 18 ("Viewdata is the generic name for fully interactive videotex. This means that the user's requests for information, or

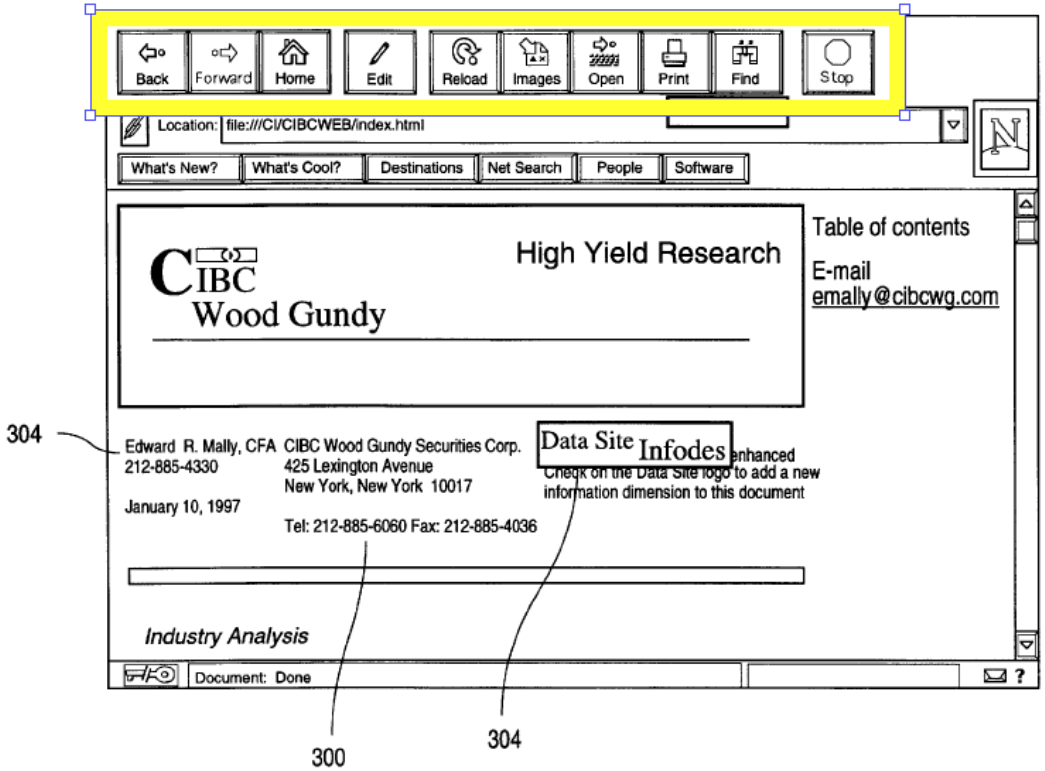
Claim Language	Prior Art References
	service, are actually sent to, received by, and acted on by the centralized computer."). As another example, Hurly describes how games such as backgammon can be played online. Hurly at 168. <u>Roy+ Lowery</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of Lowery. *See* '719 Patent Invalidity Chart, Ex. D-1 (Lowery) at claim limitation 1(d) (incorporated herein by reference). A POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Lowery's teachings about downloading a subset of application logic over a network on demand. For example, *see* claim 12. <u>Roy + History of PHP</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of History of PHP. "PHP as it's known today is actually the successor to a product named PHP/FI. Created in 1994 by Rasmus Lerdorf, the very first incarnation of PHP was a simple set of Common Gateway Interface (CGI) binaries written in the C programming language. Originally used for tracking visits to his online resume, he named the suite of scripts "Personal Home Page Tools," more frequently referenced as "PHP Tools." Over time, more functionality was desired, and Rasmus rewrote PHP Tools, producing a much larger and richer implementation. This new model was capable of database interaction and more, providing a framework upon which users could develop simple dynamic web applications such as guestbooks. In June of 1995, Rasmus » released the source code for PHP Tools to the public, which allowed developers to use it as they saw fit. This also permitted - and encouraged - users to provide fixes for bugs in the code, and to generally improve upon it." A POSITA would have been motivated to modify the teachings in Roy based on History of PHP's teachings about server side scripting for dynamic page creation. One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not

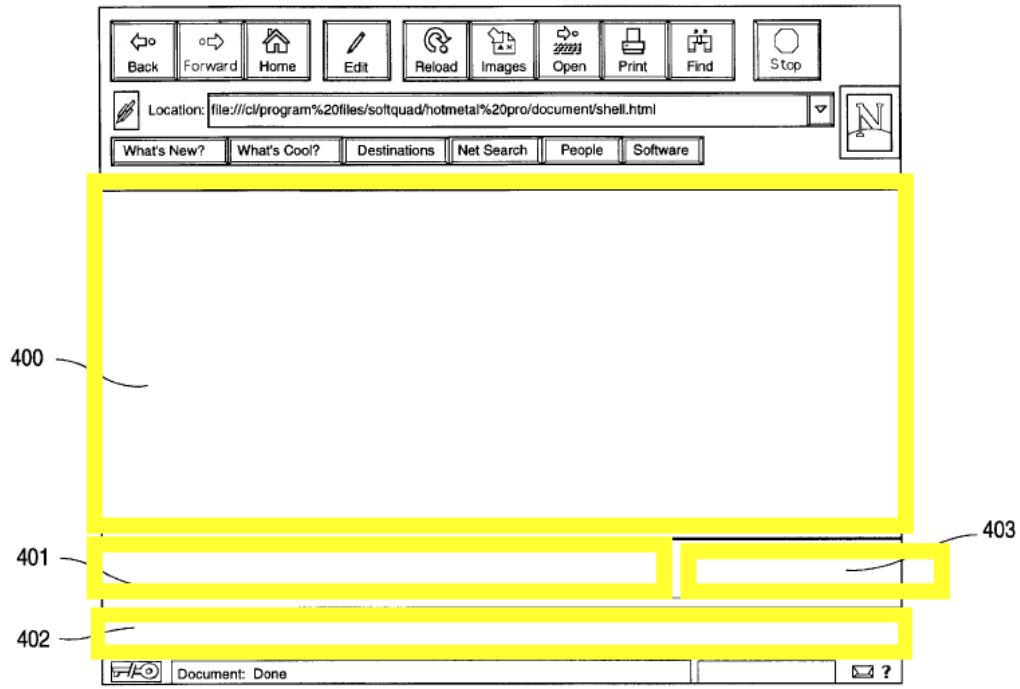
Claim Language	Prior Art References
	found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Supplemental Invalidity Contentions, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart.
2. The method of claim 1, wherein the client and server communicate over a HyperText Transport Protocol network.	Roy discloses (or at minimum it would be obvious to a POSITA) that the client and server may communicate over a HyperText Transport Protocol (HTTP) network. In particular, Roy discloses a method for solving latency issues for multiplayer games that are played on the Internet. *See* 1: 19-22 ("[T]here are unique problems associated with implementing a multi-player computer game across a wide area network (WAN), particular one as expansive as the Internet."). It would be obvious to a POSITA that HTTP would or could be used to communicate over the Internet. For example, *see* the following passages and/or figures, as well as all related disclosures: *See above* regarding claim preamble 1(pre) which is incorporated by reference herein. One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Invalidity Contentions Cover Pleading, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart. Specific prior art combinations (in conjunction with the knowledge of a POSITA) are as follows: <u>Roy+ Rover</u>

Claim Language	Prior Art References
	To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of Rover. *See* '719 Patent Invalidity Chart, Ex. D-8 (Rover) at claim 2 (incorporated herein by reference). A POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Rover's teachings about moving application data and logic between a client and server using relocatable dynamic objects ("RDOs"). For example, *see* claim limitations 1(a) and 1(b), and claim 12. Rover also teaches how a client and server communicate over an HTTP network. For example, the Rover toolkit "permits applications full access to the objects of the World-Wide Web (WWW)." Rover, Introduction. "The Rover toolkit supports several transport protocols (*e.g.*, HTTP and [SMTP] over various communication media (*e.g.*, Ethernet, WaveLAN, and phone lines). *Id.* In light of at least the foregoing, a POSITA would be motivated to try HTTP as a transport protocol for the online gaming system described by Roy. <u>Roy+ Lowery</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of Lowery. *See* '719 Patent Invalidity Chart, Ex. D-1 (Lowery) at claim 2 (incorporated herein by reference). A POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Lowery's teachings about downloading a subset of application logic over a network on demand. For example, *see* claim 12. Lowery also teaches how a client and server communicate over an HTTP network. For example, Lowery explains that "the web server application 20 is an application running on the server 18 for communicating with multiple clients 12 using a communications protocol such as the HyperText Transfer Protocol (HTTP)." Lowery at 9:1-15. As another example, the "web server application 20 may also communicate with the client 12 using any suitable communications protocol." *Id.* at 9:30-32. "For example, HTTP, the File Transfer Protocol (FTP), or the Telnet protocol may be used." *Id.* (emphasis added). In light of at least the foregoing, a POSITA would be motivated to try HTTP as a transport protocol for the online gaming system described by Roy. One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to

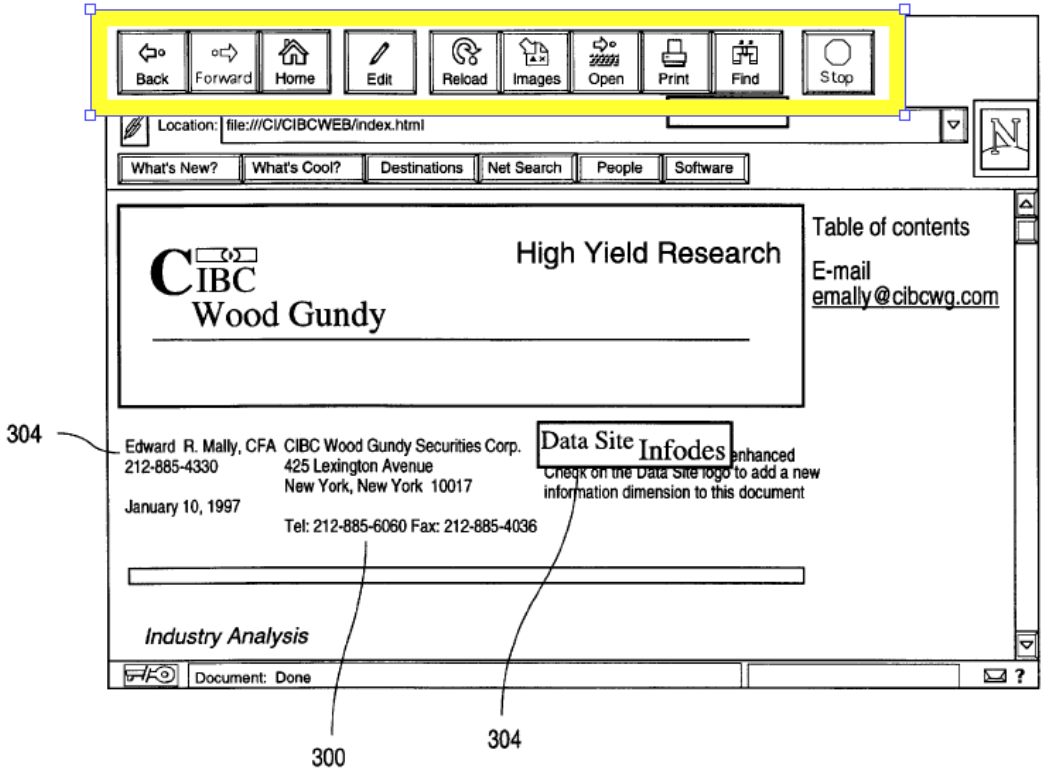
Claim Language	Prior Art References
	teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Supplemental Invalidity Contentions, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart.
3. The method of claim 1, wherein the client performs the one or more portions of the application in accordance with browser software running thereon.	<u>Roy + Hobbs</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of Hobbs. *See* '719 Patent Invalidity Chart, Ex. D-16 (Hobbs) at claim 3 (incorporated herein by reference). "In another embodiment of the present invention, at least one embedded application is sent from the second network resource to the browser. The embedded application performs one or more of the functions previously performed by the application that is executed on the second network resource in the first embodiment of the present invention. The embedded application is preferably an applet." *Hobbs, c. 8; ll.23-29.* A POSITA would have been motivated to modify the teachings in Roy based on Hobbs's teachings about server side rendering. One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Supplemental Invalidity Contentions, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart.
4. The method of claim 3, wherein the configuring step	<u>Roy + Hobbs</u>

Claim Language	Prior Art References
further comprises the step of partitioning a screen area associated with the browser software into frames.	To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of Hobbs. *See* '719 Patent Invalidity Chart, Ex. D-16 (Hobbs) at claim 4 (incorporated herein by reference). "Additionally, the applications on the Proxy Server 207 preferably cause browser 204 to subdivide its display into a plurality of frames, cause a record that satisfies the query argument to appear in the largest of the frames, cause a series of graphic symbols, e.g., buttons, to appear in a second frame for enabling a user to manually select one of a plurality of databases or Data Warehouses; cause an excerpt of text that includes a link in a file of information to appear in a third frame; and cause information associated with the source of the file of information to appear in a fourth frame." *Hobbs, c. 15; ll. 7-17.* A POSITA would have been motivated to modify the teachings in Roy based on Hobbs's teachings about server side rendering. One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Supplemental Invalidity Contentions, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart.
8. The method of claim 4, wherein the configuring step further comprises forming at least one frame with which application-independent view-generating logic and controller logic is associated.	<u>Roy + Hobbs</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of Hobbs. *See* '719 Patent Invalidity Chart, Ex. D-16 (Hobbs) at claim 8 (incorporated herein by reference).

Claim Language	Prior Art References
	The highlighted buttons may be application independent in that they appear to be standard browser navigation buttons and not necessarily associated with the application running on the proxy server (e.g., the expert query application).  FIG. 6A *Hobbs, FIG. 6A.*

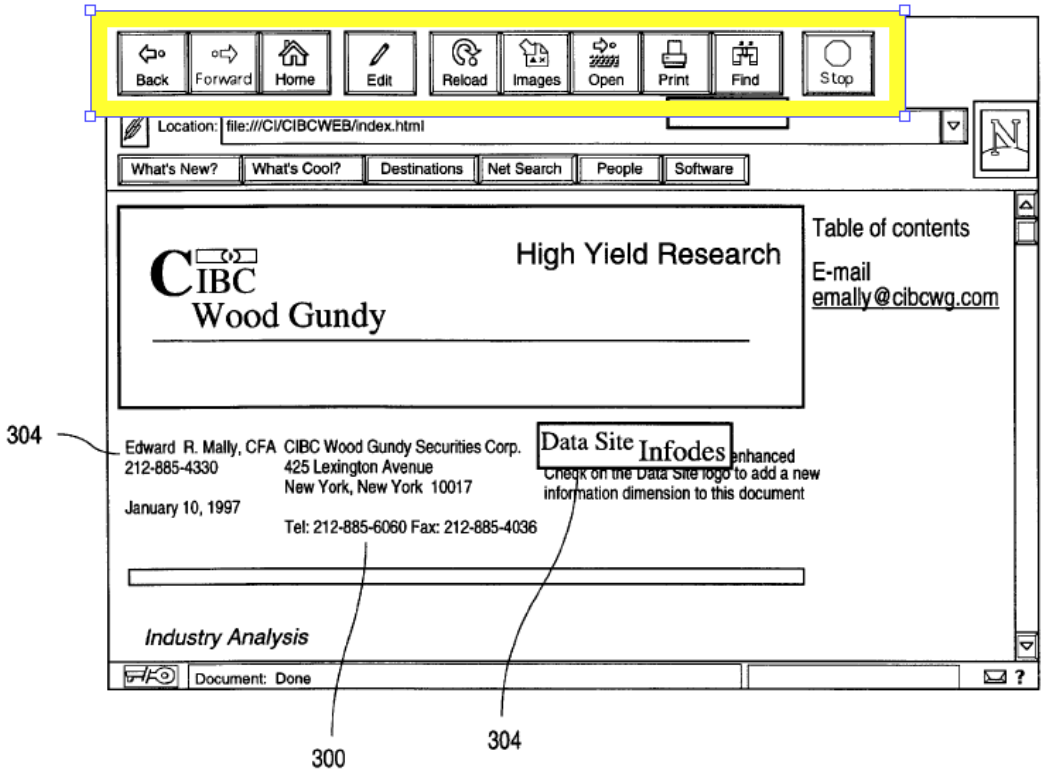
Claim Language	Prior Art References
	In FIG. 7, the highlighted frames are the four frames generated by/controlled by the application on the proxy server. Thus, the above areas of the screen which are not those four frames could be considered "application-independent."  **FIG. 7** *Hobbs, FIG. 7.*

Claim Language	Prior Art References
	One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Supplemental Invalidity Contentions, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart.

Claim Language	Prior Art References
10. The method of claim 8, wherein the at least one application-independent view-generating logic and controller logic frame serves as an application programming interface for developing views to be displayed in accordance with the application.	 FIG. 6A *Hobbs, FIG. 6A.* "It is to be noted that other transport layer application programming interfaces (API's) such as the Component Object Model (COM); Distributed COM (DCOM); IBM's System Object Model (SOM) and Distributed SOM (DSOM) for networks and Java; Microsoft's ActiveX; Common Object Request Broker Architecture (CORBRA) (with enhancements) and Sun Microsystem's

Claim Language	Prior Art References
	Java's Remote Method Invocation (RMI) for the Internet may also be used in the present invention." *Hobbs, c. 12; ll. 55-63.* A POSITA would have been motivated to modify the teachings in Roy based on Hobbs's teachings about server side rendering. <u>Roy + History of PHP</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of History of PHP. "PHP as it's known today is actually the successor to a product named PHP/FI. Created in 1994 by Rasmus Lerdorf, the very first incarnation of PHP was a simple set of Common Gateway Interface (CGI) binaries written in the C programming language. Originally used for tracking visits to his online resume, he named the suite of scripts "Personal Home Page Tools," more frequently referenced as "PHP Tools." Over time, more functionality was desired, and Rasmus rewrote PHP Tools, producing a much larger and richer implementation. This new model was capable of database interaction and more, providing a framework upon which users could develop simple dynamic web applications such as guestbooks. In June of 1995, Rasmus » released the source code for PHP Tools to the public, which allowed developers to use it as they saw fit. This also permitted - and encouraged - users to provide fixes for bugs in the code, and to generally improve upon it." A POSITA would have been motivated to modify the teachings in Roy based on History of PHP's teachings about server side scripting for dynamic page creation. One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Supplemental

Claim Language	Prior Art References
	Invalidity Contentions, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart.
11. The method of claim 10, wherein the views are implemented in accordance with the HyperText Markup Language and the application programming interface is implemented in accordance with the JavaScript language.	Roy + Hobbs To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of Hobbs. *See* '719 Patent Invalidity Chart, Ex. D-16 (Hobbs)

Claim Language	Prior Art References
	at claim 11 (incorporated herein by reference).  FIG. 6A *Hobbs, FIG. 6A.*

Claim Language	Prior Art References
	"The invention comprises an information template which maybe a file of data in Hypertext Markup Language (HTML), or other mark up language embodying text, audio, graphics, and/or video elements…" *Hobbs, c. 10; ll. 19-22.* A POSITA would have been motivated to modify the teachings in Roy based on Hobbs's teachings about server side rendering. "The second FRAME element automatically loads the contents of the first of a series of HTML files into frame 2 501. These HTML files contain the graphic images of button bars, with each button bar linked in its HREF statement toa specific network address of a network resource, Such as a Database Server, a database or a Data Warehouse. These button bars alternatively appear in frame 2501 as "pressed" or 'down", or not pressed or "up," every time a viewer presses a button bar. Such actions can be accomplished in HTML code, or C++, ActiveX, Java, JavaScript, Visual Basic computer programs, or other programming languages 268, 263 (shown in FIG. 5)." *Hobbs, c. 22, ll.53-64.* One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Supplemental Invalidity Contentions, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart.
12. The method of claim 1, wherein the at least a subset of the model, the view-generating and the controller logic associated with the application are downloaded from the	Roy discloses (or at minimum renders obvious) that at least a subset of the model, the view-generating and controller logic associated with the application are downloaded from the server to the client upon demand. For example, Roy explains how an "event server which is not yet involved in the game connects to another event server that is already part of the game." 5:50-64. It would be understood by (or at the very least rendered obvious to) a POSITA that, when a new user/computer joins the game, a "slave cop[y]" of the database would be transferred to the new

Claim Language	Prior Art References
server to the client upon demand.	computer to enable Roy's methods to work. 4:51-55 ("By providing slave copies of the database in every other machine involved in the game, the need to send rendering commands across the network is reduced or eliminated, which reduces bandwidth consumption."). Further, in light of at least the foregoing, it would be obvious to a POSITA that a subset of the VOL and CL could be downloaded at this same time, *i.e.*, when the subset of the model is downloaded. For example, *see* the following passages and/or figures, as well as all related disclosures: *See above* regarding claim elements 1(a), 1(b), and 1(d) which are incorporated by reference herein. *See also*, *e.g.*: • "Referring now to FIG. 7, a routine is illustrated by which by an event server which is not yet involved in the game connects to another event server that is already part of the game. An event server which is not yet part of the game is, by definition, a slave event server. Referring to FIG. 7, the slave event server first connects to any other event server, X, which is known to be a part of the game (step 701). The knowledge of which event servers are already part of the game, and their logical addresses, can be provided to the connecting event server in any of a number of possible ways. The methods for providing such knowledge are not germane to the present invention. Next, the slave event server which is joining the game initializes its connection list to include X (step 702). X is then tagged as the RTM (route to master) in the connection list (step 703)." 5:50-64. One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art.

Claim Language	Prior Art References
	Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Invalidity Contentions Cover Pleading, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart. Specific prior art combinations (in conjunction with the knowledge of a POSITA) are as follows: <u>Roy+ Rover</u> Carvana incorporates by reference the '719 invalidity claim chart for Rover: A Toolkit for Mobile Information Access ("Rover "), which includes exemplary disclosures relevant to claim 12. To the extent IBM argues that Roy does not disclose "wherein the at least a subset of the model, the view-generating and the controller logic associated with the application are downloaded from the server to the client upon demand," a POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on Rover's teachings about downloading a subset of application logic on demand over a network by using "relocatable dynamic objects " or "RDOs." Rover at 156, col. 2. A POSITA would have been motivated to modify one or more of the view data systems in Hurly based on Rover's teachings. For example, Roy and Rover are in the same field of endeavor (interactive electronic information delivery systems over a network). *See* Roy, Abstract ("A method and apparatus for reducing network latency during execution of a multiple-player game across a computer network are provided."); Rover at 156 ("The Rover toolkit offers applications a uniform distributed object system based on a client/server architecture. . . . [The toolkit] combines relocatable dynamic objects and queued remote procedure calls to provide unique services for 'roving' mobile applications."). A POSITA examining Roy and the online multiplayer game system described therein would need to determine how to distribute game logic to each client computer. At the relevant time (early 2000s), a POSITA would understand that the logic could either be sent on a disk of some sort (*e.g.*, CD-ROM) or sent over a network. A POSITA would be motivated to consider both options.

Claim Language	Prior Art References
	Rover explains how to download a subset of application logic over the network on demand. For example, Rover explains that "[t]he Rover toolkit combines relocatable dynamic objects and queued remote procedure calls to provide unique services for 'roving' mobile applications." Rover at 156. "A relocatable dynamic object is an object with a well-defined interface that can be dynamically loaded into a client computer from a server computer (or vice versa) to reduce client-server communication requirements." *Id.* "The Rover toolkit also offers advantages for workstation applications by providing a uniform distributed object architecture for code shipping, object caching, and asynchronous object invocation." *Id.* "Depending on the power of the mobile host and the available bandwidth, Rover dynamically adapts and moves functionality between the client and server." *Id.* at 158. For example, "[b]y migrating a GUI to the client, input events can be processed locally, even if the client becomes disconnected." *Id.* As another example, Rover describes an e-mail system "which is contained in RDOs retrieved from an HTTP server when the user first loads the browser"-*i.e.*, on demand. *Id.* at 165. Rover further explains how to handle a situation wherein the client connection to a server is lost, which a POSITA would be motivated to incorporate into the online gaming system described by Roy to handle situations wherein one of the clients are disconnected. For example: "Queued remote procedure call is a communication system that permits applications to continue to make non-blocking remote procedure call requests even when a host is disconnected, with requests and responses being exchanged upon network reconnection." *Id.* at 156. A POSITA would be motivated to incorporate Rover's methods for downloading a subset of logic over a network into the online gaming system described by Roy to download data and program logic from the server to each client device rather than, for example, require use of a CD-ROM. A POSITA would also be motivated to utilize Rover's teachings concerning how to handle a client that becomes disconnected, as described above. <u>Roy + Bennett</u> To the extent IBM argues that this limitation is not disclosed or rendered obvious by Roy, it is rendered obvious by Roy in view of Bennett. *See* '719 Patent Invalidity Chart, Ex. D-15 (Bennett) at claim 12 (incorporated herein by reference). A POSITA would have been motivated to modify

Claim Language	Prior Art References
	the online multiplayer gaming system in Roy based on Bennett's teachings about moving application data and logic between a client and server using objects. For example, *see* claim limitations 1(a) and 1B. <u>Roy+ Hurly</u> To the extent IBM argues that Roy does not disclose "wherein the at least a subset of the model, the view-generating and the controller logic associated with the application are downloaded from the server to the client upon demand," a POSITA would have been motivated to modify the online multiplayer gaming system in Roy based on The Videotex and Teletex Handbook by Paul Hurly ("Hurly"). Hurly was published in 1985 and qualifies as prior art to the '849 patent under at least 35 U.S.C. §§ 102(a) and/or (b). Hurly teaches how application logic and data may be downloaded on demand from a server to a client computer. A POSITA would have been motivated to modify one or more of the view data systems in Roy based on Hurly's teachings. For example, Roy and Hurly are in the same field of endeavor (interactive electronic information delivery systems over a network). *See* Roy, Abstract ("A method and apparatus for reducing network latency during execution of a multiple-player game across a computer network are provided."); Hurly at 18 ("Viewdata is the generic name for fully interactive videotex. This means that the user's requests for information, or service, are actually sent to, received by, and acted on by the centralized computer."). And both concern online gaming applications. Roy discloses how to "reduc[e] network latency during execution of a multiple-player game across a computer network." Abstract. Hurly describes how games can be played online such as, for example, Backgammon on the Grassroots viewdata system:

Claim Language	Prior Art References
	  Hurly at 168. Hurly explains that online computer gaming with videotex systems provides "both on-line competition either with the service computers or with a network of other players across the country, or the opportunity to receive program instructions directly into the user's terminal." *Id.* at 341. "This new concept of telegaming extends the domain of computer gaming and offers the new dimension of networking coupled with the opportunity to access the latest games." *Id. see also*, *e.g.*, *id.* at 353 ("In the 1980s, networks of computer users, some using . . . interactive features like telegaming, are being established across nations and continents. These new communities have been made possible by a new technology."); *id.* at 46 ("Numerous adventure and action games can be played on existing viewdata networks .... Players can choose to player either against the computer or as part of a group."). A POSITA examining Roy and the online multiplayer game system described therein would need to determine how to distribute game logic to the client computers. At the relevant time (early 2000s), a POSITA would understand that the logic could either be sent on a disk of some sort (*e.g.*, CD-ROM) or sent over a network. A POSITA would be motivated to consider both options. Hurly teaches how to download program logic and data over a network. In particular, Hurly discloses "downline loading"-a "data communication process that permits one computer to transfer programs and data to another computer." Hurly at 49; *see also id.* at 389 ("Download[:] The

Claim Language	Prior Art References
	process of loading a computer program and/or data from a remote location. The term downline loading is also used."). "In the case of viewdata systems, downline loading also eliminates the cost of continuous terminal-to-host computer communication." *Id.* Hurly teaches that "[t]his could be especially important for consumer applications." *Id.* at 169; *see also id.* at 115 ("The technology exists to permit any user to downline load data to a local terminal."). "Since most computer functions are at least theoretically suitable for downline loading to an appropriate computer terminal, the business potential, particularly for viewdata systems, is great." *Id.* at 155. "In fact, the swing to distributed processing and word processing installation are strong indications of this potential." *Id.* Hurly further explains how application logic can be downloaded on demand. For example, "Several U.S. firms in the video game business have developed a service called Playcable." *Id.* at 49. "Using cable-distributed broadband teletext (cabletext), a subscriber selects a video game to play and, instead of inserting a cartridge, simply instructs the teletext terminal to pluck the appropriate video game telesoftware from the transmission cycle and load it into the memory of the subscriber's computer." *Id.* In light of at least the foregoing, a POSITA would understand that the "downline loading" or "downloading" could include a subset of the model, VGL, and CL associated with the application. For example, a POSITA would understand that "pluck[ing] the appropriate video game telesoftware" could include downloading at least a subset of the model, VGL, and CL associated with the application. *See id.* And a POSITA would understand that the download would (or could) be on demand (*e.g.*, the "subscriber selects" and the logic is "load[ed] into the memory of the subscriber's computer"). *See id.* A POSITA would be motivated to incorporate downline loading from Hurly into the online gaming system described by Roy to download data and program logic to each client device over the network rather than, for example, require use of a CD-ROM. As another example, Hurly's methods provide users "opportunity to access the latest games." Hurly at 341. <u>Roy+ Lowery</u>

Claim Language	Prior Art References
	Carvana incorporates by reference the '719 invalidity claim chart for U.S. Patent No. 6,44, 111 to Keith A. Lowery ("Lowery"), which includes exemplary disclosures relevant to claim 12. To the extent IBM argues that Roy does not disclose "wherein the at least a subset of the model, the view-generating and the controller logic associated with the application are downloaded from the server to the client upon demand," a POSITA would have been motivated to modify the online multiplayer gaming system described by Roy based on Lowery's teachings regarding downloading a subset of application logic over a network on demand. A POSITA would have been motivated to modify one or more of the view data systems in Hurly based on Lowery's teachings. For example, Roy and Lowery are in the same field of endeavor (interactive electronic information delivery systems over a network). *See* Roy, Abstract ("A method and apparatus for reducing network latency during execution of a multiple-player game across a computer network are provided."); Lowery, Abstract ("A request is sent from a client (12) to a server (18) over a communications link (16). A web server (20) on the server responds to the request with a dynamically generated, selected characteristic enabled, transient applet (26) including a plurality of data items (28) therein."). A POSITA examining Roy and the online multiplayer game system described therein would need to determine how to distribute game logic to each client computer. At the relevant time (early 2000s), a POSITA would understand that the logic could either be sent on a disk of some sort (*e.g.*, CD-ROM) or sent over a network. A POSITA would be motivated to consider both options. Lowery explains how to download a subset of application logic over a network. For example, Lowery explains that "[a] request is sent from a client (12) to a server (18) over a communications link (16)." Abstract. "A web server (20) on the server responds to the request with a dynamically generated, selected characteristic enabled, transient applet (26) including a plurality of data items (28) therein." *Id.* A POSITA would understand that this dynamically generated applet would (or could) comprise a subset of the model, VGL, and CL, particularly in light of Lowery's disclosure that the applet includes all the logic necessary needed to "review[], manipulate, and utilize[e] the data items." *See* 15:17-45 ("When the applet 26 is generated, the applet 26 also comprises substantially all of the data manipulation capabilities for reviewing, manipulating, and utilizing the data items 28 represented by pre-loaded elements 36 in the applet 26. Since the applet 26

Claim Language	Prior Art References
	comprises substantially all of the data items 28 appropriate to respond to the request, repeated transfers of information over the communications link 16 may be avoided. Stated another way, the pre-loaded elements 36 may leech off of the loading of the data manipulation capabilities to provide data and functionality to the applet 26."). A POSITA would be motivated to incorporate Lowery's methods for downloading a subset of logic over a network into the online gaming system described by Roy to download data and program logic from the server to each client device rather than, for example, require use of a CD-ROM.
26(pre). A method for use in a client/server system of reducing interactions between a client and server in association with an application being accessed by the client at the server, the method comprising the steps of:	To the extent the preamble is limiting, Roy discloses this limitation as described above regarding claim preamble 1(pre). For example, *see* the following passages and/or figures, as well as all related disclosures: *See above* regarding claim preamble 1(pre) which is incorporated by reference herein. One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Invalidity Contentions Cover Pleading, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart.
26(a). configuring the server to:	Roy discloses this limitation as described above regarding claim element 1(a). For example, *see* the following passages and/or figures, as well as all related disclosures: *See above* regarding claim element 1(a) which is incorporated by reference herein.

Claim Language	Prior Art References
(i) store a model associated with the application; (ii) execute view-generating logic associated with the application; and (iii) execute controller logic associated with the application; and	One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Invalidity Contentions Cover Pleading, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart.
26(b). configuring the client to: (i) store at least a subset of the model associated with the application; (ii) execute at least a subset of the view-generating logic associated with the application; and (iii) execute at least a subset of the controller logic associated with the application;	Roy discloses this limitation as described above regarding claim element 1(b). For example, *see* the following passages and/or figures, as well as all related disclosures: *See above* regarding claim element 1(b) which is incorporated by reference herein. One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Invalidity Contentions Cover Pleading, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart.
26(c). wherein one or more portions of the application are performed at the client without	Roy discloses this limitation as described above regarding claim element 1(c). For example, *see* the following passages and/or figures, as well as all related disclosures: *See above* regarding claim element 1(c) which is incorporated by reference herein.

Claim Language	Prior Art References
the client having to interact with the server, and	One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Invalidity Contentions Cover Pleading, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart.
26(d). further wherein the client and the server both locally maintain at least a portion of the model and execute the view-generating and controller logic associated therewith;	Roy discloses this limitation as described above regarding claim element 1(d). For example, *see* the following passages and/or figures, as well as all related disclosures: *See above* regarding claim element 1(d) which is incorporated by reference herein. One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Invalidity Contentions Cover Pleading, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart.
26(e). further wherein, in accordance with such a dual model-view-controller arrangement, a model comprises application data, rules, and algorithms affecting	Roy discloses this limitation as described above regarding claim elements 1(a) and 1B. For example, *see* the following passages and/or figures, as well as all related disclosures: *See above* regarding claim elements 1(a) and 1B which are incorporated by reference herein.

Claim Language	Prior Art References
the data, a view comprises a screen or window representation of a subset of the model that the application chooses to display, and a controller comprises the logic that processes user requests, and causes the model to be changed and the view to be refreshed.	One of ordinary skill would find this limitation disclosed either expressly or inherently in the teachings of this reference and its incorporated disclosures taken as a whole, or in combination with the state of the art at the time of the alleged invention. To the extent this reference is not found to teach this element explicitly, implicitly, or inherently, the element would have been obvious to one of ordinary skill in the art based on this reference, common sense, ordinary creativity of one of ordinary skill in the art, and the state of the art. Additionally, it would have been obvious to combine this reference with one or more other prior art references identified in Carvana's Invalidity Contentions Cover Pleading, particularly, § IV, A. Rather than repeat those disclosures here, they are incorporated by reference into this chart.