Exhibit 753-05
Invalidity of the ’753 Patent by Jana

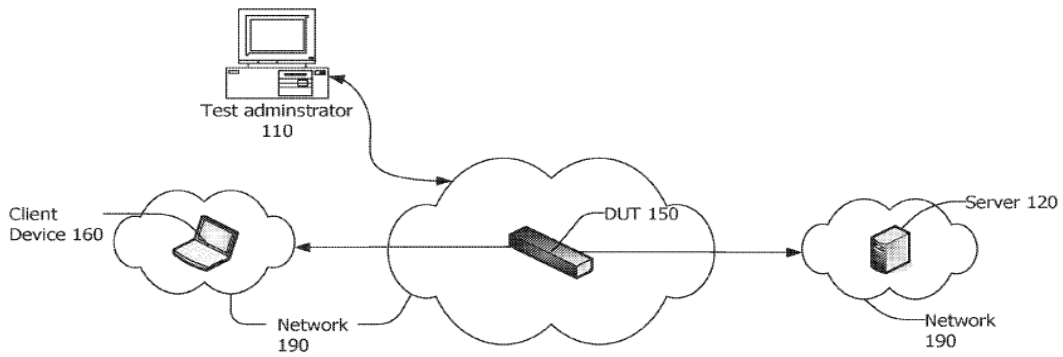
The asserted claims of U.S. Patent No. 10,873,753 are anticipated and/or rendered obvious in view of U.S. Patent No. 9,306,816 (“Jana”) (avail. at. PA035828-PA035838) alone or in combination with the knowledge of one of skill in the art or other prior art.

This claim chart is based on Defendants’ present understanding of the asserted claims, Plaintiffs’ vague and overbroad infringement contentions (and corresponding, implicit claim constructions), and Defendants’ investigation to date. Defendants are not adopting Plaintiffs’ apparent constructions, nor are Defendants admitting to the correctness of any particular construction. Defendants reserve all rights to amend this invalidity claim chart in light of any claim construction developments or any amendments to Plaintiffs’ infringement contentions. Further, as discovery is ongoing and Defendants continue to seek discovery from third parties regarding the references identified in Defendants’ invalidity contentions as well as other potential prior art, Defendants reserve the right to amend or supplement this claim chart in connection with Defendants’ continuing investigation. At a minimum, Defendants intend to revise its invalidity contentions as appropriate based on discovery from third parties as discovery progresses.

The claim chart below identifies where each limitation of each asserted claim of the ’753 patent is anticipated and/or rendered obvious in view of Jana. The citations provided below are exemplary, rather than exhaustive, and Defendants reserve the right to rely upon any other portion of the cited references.

Claim Language	Exemplary Evidence of Anticipation and/or Obviousness
[1.pre] A platform for data flow processing comprising:	To the extent the preamble is limiting, Jana discloses or renders obvious “[a] platform for data flow processing comprising,” as shown in at least the following exemplary citations: *See e.g.,* Jana, Abstract: There is disclosed a system and method for replaying a network capture comprising loading a network capture file that displays at least one traffic flow comprising a sequence of packets, selecting at least one packet to display on a viewer and transmitting the network traffic to a device under test wherein the selected packet is displayed on the viewer when the packet is transmitted to the device under test. *See e.g.,* Jana, 4:27-39: FIG. 1 shows a block diagram of a network environment 100 for replaying network captures in a testing environment on a datacenter device. Replayed network captures comprise packets and associated responses from traffic flows that were transmitted previously on a network and are being replayed on a

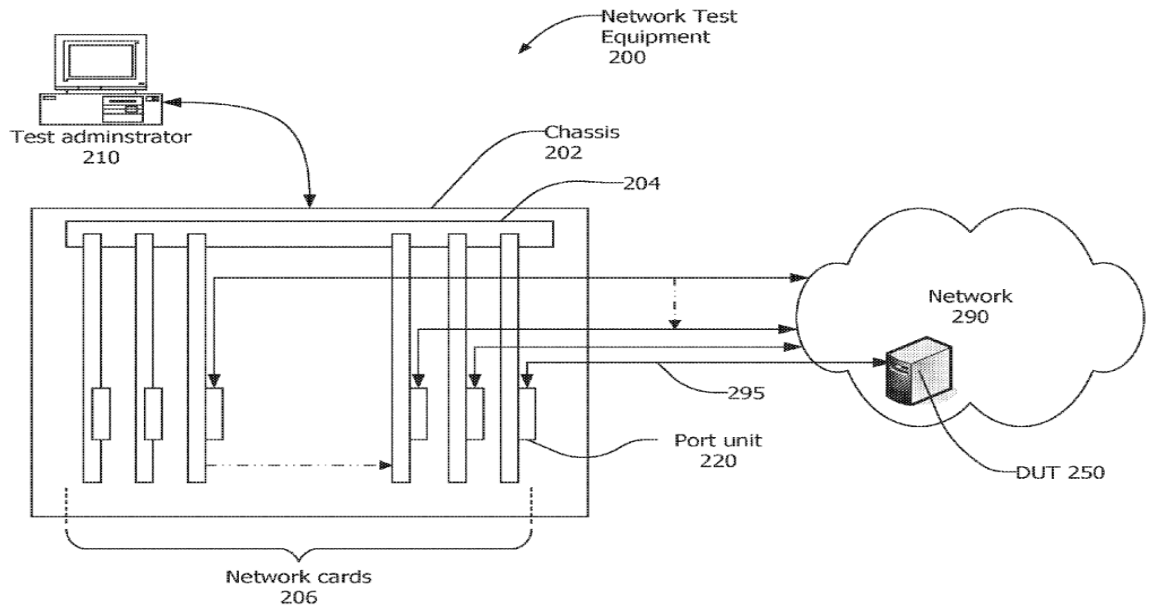
IPR2025-00478
Concurrent Ventures LLC, XtreamEdge Inc. EX2045

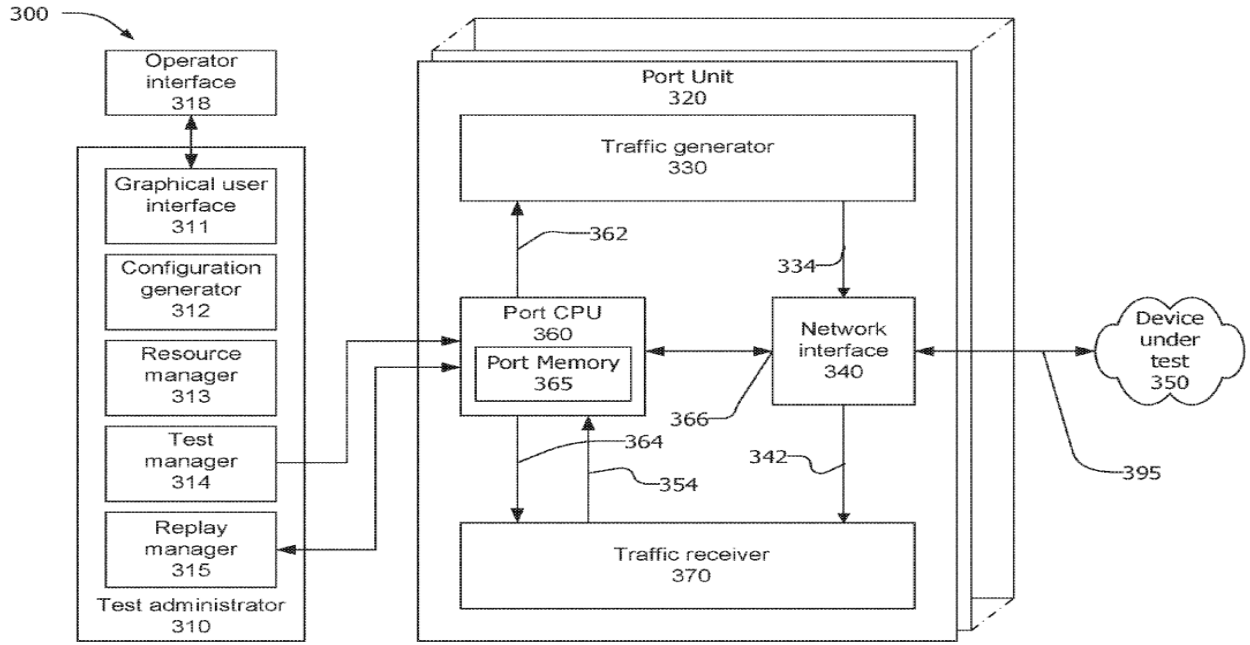
Claim Language	Exemplary Evidence of Anticipation and/or Obviousness
	network after being captured and stored. The environment may include a test administrator 110, server 120, a device under test (DUT) 150, a client device 160 and a network 190. For ease of description, the DUT 150 of FIG. 1 is coupled to a single client device and a single server. The DUT in an actual testing environment may be connected to a large number of emulated client devices and a large number of emulated servers. *See e.g.,* Jana, FIG.1:  FIG. 1
[1.a] one or more swappable pods or cards in one or more chassis, coupled	Jana discloses or renders obvious “one or more swappable pods or cards in one or more chassis, coupled through a messaging interface network,” as shown in at least the following exemplary citations: *See e.g.,* Jana, 4:51-5:45:

Claim Language	Exemplary Evidence of Anticipation and/or Obviousness
through a messaging interface network;	FIG. 2 shows a block diagram of a network environment. The environment may include network test equipment 200 and a network 290 which includes one or more devices under test (DUT) 250. The DUT 250 may be any device capable of communicating over the network 290. The network test equipment 200 may be a network testing device, performance analyzer, conformance validation system, network analyzer, or network management system. The network test equipment 200 may include one or more network cards 206 and a backplane 204 contained or enclosed within a chassis 202. The chassis 202 may be a fixed or portable chassis, cabinet, or enclosure suitable to contain the network test equipment. The network test equipment 200 may be an integrated unit, as shown in FIG. 2. Alternatively, the network test equipment 200 may comprise a number of separate units cooperative to provide traffic generation and/or analysis. The network test equipment 200 and the network cards 206 may support one or more well known standards or protocols such as the various Ethernet and Fibre Channel standards, and may support proprietary protocols as well. The network test equipment 200 and the device under test 250 may communicate simultaneously with one another, and there may be plural logical communications paths between the network test equipment 200 and a device under test 250. The network itself may be comprised of numerous nodes providing numerous physical and logical paths for data to travel. The network cards 206 may include one or more field programmable gate arrays (FPGAs), application specific integrated circuits (ASICs), programmable logic devices (PLDs), programmable logic arrays (PLAs), processors, and other kinds of devices. In addition, the network cards 206 may include software and/or firmware. The term network card encompasses line cards, test cards, analysis cards, network line cards, load modules, interface cards, network interface cards, data interface cards, packet engine cards, service cards, smart cards, switch cards, relay access cards, and the like. The term network card also encompasses modules, units, and assemblies that may include multiple printed circuit boards. Each network card 206 may support a single communications protocol, may support a number of related protocols, or may support a number of unrelated protocols. One or more network cards 206 or their equivalent may be permanently installed in a self-contained test unit or tests appliance. Each network card 206 may contain one or more port unit 220. Each port unit 220 may include circuits and software to generate test traffic and/or to receive and analyze test traffic. Each port unit may be coupled to the test administrator 210. Each port unit 220 may connect to the network 290 through one or

Claim Language	Exemplary Evidence of Anticipation and/or Obviousness
	more ports. Each port unit 220 may be connected to the network 290 through a communication medium 295, which may be a wire, an optical fiber, a wireless link, or other communication medium. Each port unit 220 refers to a module within the network test equipment that connects to the DUT 250 at a port along either the N-S communication axis or the E-W communication axis. Each port unit 220 connected to the DUT may be both a source of test traffic and a destination for test traffic. Each port unit 220 may emulate a plurality of logical source and/or destination addresses. Each port unit 220 may emulate a plurality of network users, clients, peers, servers, or other network devices. The backplane 204 may serve as a bus or communications medium for the network cards 206. The backplane 204 may also provide power to the network cards 206. *See e.g.,* Jana, 6:8-33: The network traffic on network 290 and produced and received by various nodes, including the ports on the network cards 206 includes both control plane traffic and data plane traffic. The control plane and data plane are virtualizations that are useful in conceptualizing the roles of two kinds of network traffic. Control plane traffic includes packets that implement the various Supported communications protocols, including administrative communications among nodes to maintain communications among the nodes. An example of control plane traffic are routing notices or packets informing a second node that a first node is inaccessible. Data plane traffic includes packets carrying data between nodes. The data plane carries network traffic that communicates data over a network. For example, those packets that carry data for an audio stream are considered to be communicated over the data plane and are considered data plane traffic. Referring now to FIG. 3, a test system 300, which may be the network test equipment 300 may include a test administrator 310 coupled to a plurality of port units including an exemplary port unit 320. The port unit 320 may include a port central processor unit 360 (CPU), a traffic generator unit 330, a traffic receiver unit 370, and a network interface unit 340 which couples the port unit 320 to a device under test 350. The port unit 320 may be all or part of a network card such as the network cards 206. *See e.g.,* Jana, 10:66-11:14:

Claim Language	Exemplary Evidence of Anticipation and/or Obviousness
	The methods described herein may be implemented on one or more FPGAs and/or other hardware devices, such as, for example, digital logic devices. The methods described herein may be implemented as software, namely network testing software, running on a network testing system and executed by a processor, Such as a processor on a network card or a processor in a blade or other card with a processor in a network testing system. The network testing software may be stored on a volatile or nonvolatile memory device or storage medium included in or on and/or coupled with a computing device, a network testing system, a network card, or other card. The methods may be implemented on one or more network cards in a single network testing system or may be implemented on one or more network cards on each of two or more network testing systems. *See e.g.,* Jana, FIG. 2:

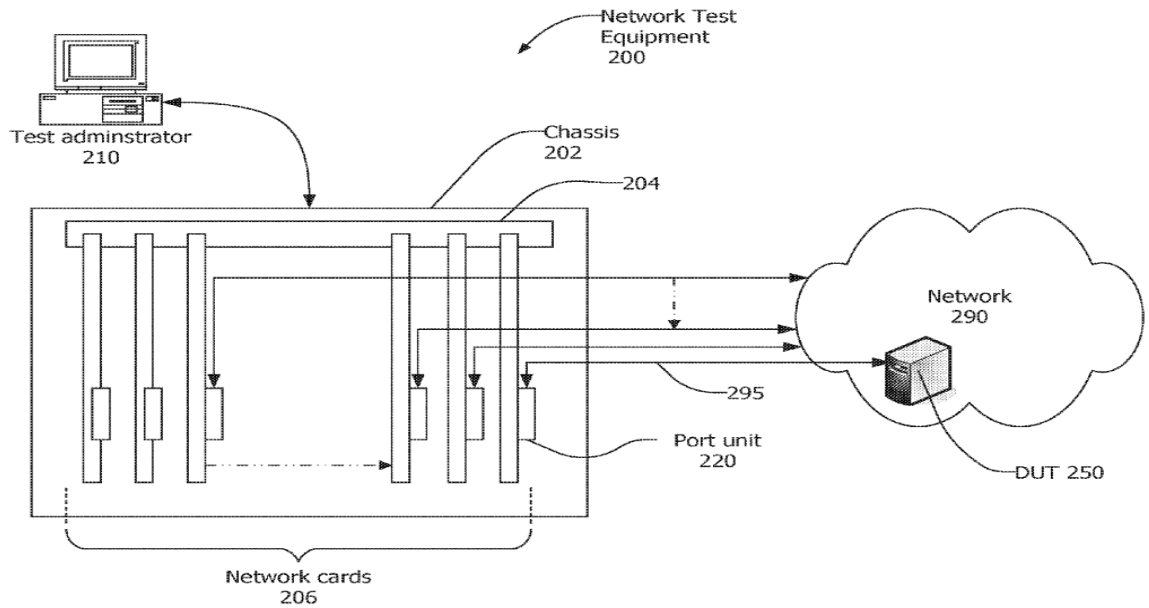
Claim Language	Exemplary Evidence of Anticipation and/or Obviousness
	 FIG. 2 *See e.g.,* Jana, FIG. 3:

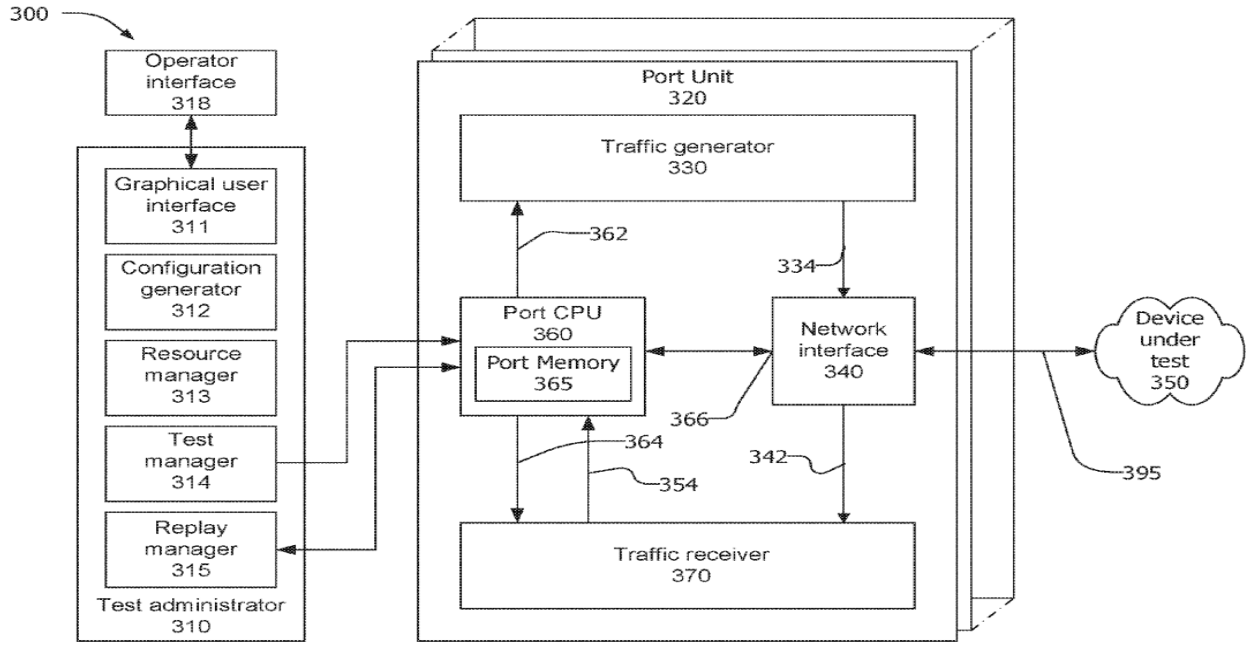
Claim Language	Exemplary Evidence of Anticipation and/or Obviousness
	 FIG. 3
[1.b] each of the one or more swappable pods or cards having one or more hardware modules or one or more software modules;	Jana discloses or renders obvious “each of the one or more swappable pods or cards having one or more hardware modules or one or more software modules,” as shown in at least the following exemplary citations: *See e.g.,* Jana, 5:10-26: The network cards 206 may include one or more field programmable gate arrays (FPGAs), application specific integrated circuits (ASICs), programmable logic devices (PLDS), programmable logic arrays (PLAS), processors, and other kinds of devices. In addition, the network cards 206 may

Claim Language	Exemplary Evidence of Anticipation and/or Obviousness
	include software and/or firmware. The term network card encompasses line cards, test cards, analysis cards, network line cards, load modules, interface cards, network interface cards, data interface cards, packet engine cards, service cards, smartcards, Switch cards, relay access cards, and the like. The term network card also encompasses modules, units, and assemblies that may include multiple printed circuit boards. Each network card 206 may support a single communications protocol, may support a number of related protocols, or may support a number of unrelated protocols. One or more network cards 206 or their equivalent may be permanently installed in a self-contained test unit or tests appliance. *See e.g.,* Jana, 5:27-43: Each network card 206 may contain one or more port unit 220. Each port unit 220 may include circuits and software to generate test traffic and/or to receive and analyze test traffic. Each port unit may be coupled to the test administrator 210. Each port unit 220 may connect to the network 290 through one or more ports. Each port unit 220 may be connected to the network 290 through a communication medium 295, which may be a wire, an optical fiber, a wireless link, or other communication medium. Each port unit 220 refers to a module within the network test equipment that connects to the DUT 250 at a port along either the N-S communication axis or the E-W communication axis. Each port unit 220 connected to the DUT may be both a source of test traffic and a destination for test traffic. Each port unit 220 may emulate a plurality of logical source and/or destination addresses. Each port unit 220 may emulate a plurality of network users, clients, peers, servers, or other network devices. *See e.g.,* Jana, 6:33-59: The port CPU 360 may include a processor (not shown), a memory 365 coupled to the processor, and various specialized units, circuits, software and interfaces for providing the functionality and features described here. The processes, functionality and features may be embodied in whole or in part in software which operates on the processor and may be in the form of firmware, an application program, an applet (e.g., a Java applet), a browser plug-in, a COM object, a dynamic linked library (DLL), a script, one or more subroutines, or an operating system component or service. The hardware and software and their functions may be distributed such that some functions are performed by the processor and others by other devices. The memory 365 may include random access memory, or a combination of random access memory and other memory such as read-only memory and flash

Claim Language	Exemplary Evidence of Anticipation and/or Obviousness
	memory. The memory 365 may store both data and program instructions for execution by the processor. The memory 3 65 may include a received packet buffer. The received packet buffer may have capacity to store at least the longest anticipated received packet. The received packet buffer may have capacity to store multiple packets. A traffic receiver 370 may write one or more received packets into the received packet buffer, for example using direct memory access. After each received packet is stored, the traffic receiver 380 may send a signal (such as an interrupt, a flag, or another signal) to the processor 360 indicating that a received packet is waiting in the received packet buffer. *See e.g.,* Jana, 6:60-7:26: The memory 365 may include a transmit packet buffer. The transmit packet buffer may have capacity to store at least the longest anticipated packet to be transmitted. The transmit packet buffer may have capacity to store multiple packets. The processor 360 may write one or more packets to be transmitted into the transmit packet buffer. After each packet to be transmitted is stored, the processor 360 may send a -signal (such as an interrupt, a flag, or another signal) to a traffic generator 330 indicating that a packet to be transmitted is waiting in the transmit packet buffer. The traffic generator 360 may then extract the packet to be transmitted from the transmit packet buffer, for example by direct memory access. Programs stored in the memory 365 may include a TCP layer program that causes the processor to open, manage, and close TCP connections with device under test 350. The TCP layer program may be a layer within a TCP/IP protocol stack operating on the port CPU. The port CPU may be configured to maintain a large plurality of concurrent TCP connections with the device under test 350, and may open a million or more TCP connections during a network test session. For each TCP connection, the processor may send packets via a buffer in the memory 365 and the traffic generator 330, and may and receive packets via the traffic receiver 370 and a receive buffer in the memory 365. For each TCP connection, the processor may store TCP connection data in a TCP connection memory within the memory 365. The stored TCP connection data may include, for example, the state of the connection, the highest sequence number transmitted, and the highest sequence number acknowledged.

Claim Language	Exemplary Evidence of Anticipation and/or Obviousness
	Programs stored in the memory 365 may also include a capture routine that causes the processor 360 to capture selected packets from each session. The captured packets may be extracted and stored in a capture memory 340 within the memory 365. *See e.g.,* Jana, 10:66-11:13: The methods described herein may be implemented on one or more FPGAs and/or other hardware devices, such as, for example, digital logic devices. The methods described herein may be implemented as software, namely network testing software, running on a network testing system and executed by a processor, Such as a processor on a network card or a processor in a blade or other card with a processor in a network testing system. The network testing software may be stored on a volatile or nonvolatile memory device or storage medium included in or on and/or coupled with a computing device, a network testing system, a network card, or other card. The methods may be implemented on one or more network cards in a single network testing system or may be implemented on one or more network cards on each of two or more network testing systems. *See e.g.,* Jana, FIG. 2:

Claim Language	Exemplary Evidence of Anticipation and/or Obviousness
	 FIG. 2 *See e.g.*, Jana, FIG. 3:

Claim Language	Exemplary Evidence of Anticipation and/or Obviousness
	 FIG. 3

[1.c] one or more of the plurality of swappable pods or cards having a portion for user-definable hardware modules or user-definable software modules;	Jana discloses or renders obvious “one or more of the plurality of swappable pods or cards having a portion for user-definable hardware modules or user-definable software modules,” as shown in at least the following exemplary citations: *See e.g.,* Jana, 4:51-5:45: FIG. 2 shows a block diagram of a network environment. The environment may include network test equipment 200 and a network 290 which includes one or more devices under test (DUT) 250. The DUT 250 may be any device capable of communicating over the network 290.

	The network test equipment 200 may be a network testing device, performance analyzer, conformance validation system, network analyzer, or network management system. The network test equipment 200 may include one or more network cards 206 and a backplane 204 contained or enclosed within a chassis 202. The chassis 202 may be a fixed or portable chassis, cabinet, or enclosure suitable to contain the network test equipment. The network test equipment 200 may be an integrated unit, as shown in FIG. 2. Alternatively, the network test equipment 200 may comprise a number of separate units cooperative to provide traffic generation and/or analysis. The network test equipment 200 and the network cards 206 may support one or more well known standards or protocols such as the various Ethernet and Fibre Channel standards, and may support proprietary protocols as well. The network test equipment 200 and the device under test 250 may communicate simultaneously with one another, and there may be plural logical communications paths between the network test equipment 200 and a device under test 250. The network itself may be comprised of numerous nodes providing numerous physical and logical paths for data to travel. The network cards 206 may include one or more field programmable gate arrays (FPGAs), application specific integrated circuits (ASICs), programmable logic devices (PLDs), programmable logic arrays (PLAs), processors, and other kinds of devices. In addition, the network cards 206 may include software and/or firmware. The term network card encompasses line cards, test cards, analysis cards, network line cards, load modules, interface cards, network interface cards, data interface cards, packet engine cards, service cards, smart cards, switch cards, relay access cards, and the like. The term network card also encompasses modules, units, and assemblies that may include multiple printed circuit boards. Each network card 206 may support a single communications protocol, may support a number of related protocols, or may support a number of unrelated protocols. One or more network cards 206 or their equivalent may be permanently installed in a self-contained test unit or tests appliance. Each network card 206 may contain one or more port unit 220. Each port unit 220 may include circuits and software to generate test traffic and/or to receive and analyze test traffic. Each port unit may be coupled to the test administrator 210. Each port unit 220 may connect to the network 290 through one or more ports. Each port unit 220 may be connected to the network 290 through a communication medium 295, which may be a wire, an optical fiber, a wireless link, or other communication medium. Each port unit 220 refers to a module within the network test equipment that connects to the DUT 250 at a port along either the N-S communication axis or the E-W communication axis. Each port unit 220 connected to the DUT may be both a source of test traffic and a destination for test traffic. Each port unit 220 may

IPR2025-00478
Concurrent Ventures LLC, XtreamEdge Inc. EX2045

	emulate a plurality of logical source and/or destination addresses. Each port unit 220 may emulate a plurality of network users, clients, peers, servers, or other network devices. The backplane 204 may serve as a bus or communications medium for the network cards 206. The backplane 204 may also provide power to the network cards 206. *See e.g.,* Jana, 6:8-33: The network traffic on network 290 and produced and received by various nodes, including the ports on the network cards 206 includes both control plane traffic and data plane traffic. The control plane and data plane are virtualizations that are useful in conceptualizing the roles of two kinds of network traffic. Control plane traffic includes packets that implement the various Supported communications protocols, including administrative communications among nodes to maintain communications among the nodes. An example of control plane traffic are routing notices or packets informing a second node that a first node is inaccessible. Data plane traffic includes packets carrying data between nodes. The data plane carries network traffic that communicates data over a network. For example, those packets that carry data for an audio stream are considered to be communicated over the data plane and are considered data plane traffic. Referring now to FIG. 3, a test system 300, which may be the network test equipment 300 may include a test administrator 310 coupled to a plurality of port units including an exemplary port unit 320. The port unit 320 may include a port central processor unit 360 (CPU), a traffic generator unit 330, a traffic receiver unit 370, and a network interface unit 340 which couples the port unit 320 to a device under test 350. The port unit 320 may be all or part of a network card such as the network cards 206. *See e.g.,* Jana, 10:66-11:14: The methods described herein may be implemented on one or more FPGAs and/or other hardware devices, such as, for example, digital logic devices. The methods described herein may be implemented as software, namely network testing software, running on a network testing system and executed by a processor, Such as a processor on a network card or a

	processor in a blade or other card with a processor in a network testing system. The network testing software may be stored on a volatile or nonvolatile memory device or storage medium included in or on and/or coupled with a computing device, a network testing system, a network card, or other card. The methods may be implemented on one or more network cards in a single network testing system or may be implemented on one or more network cards on each of two or more network testing systems. *See e.g.,* Jana, 5:10-26: The network cards 206 may include one or more field programmable gate arrays (FPGAs), application specific integrated circuits (ASICs), programmable logic devices (PLDS), programmable logic arrays (PLAS), processors, and other kinds of devices. In addition, the network cards 206 may include software and/or firmware. The term network card encompasses line cards, test cards, analysis cards, network line cards, load modules, interface cards, network interface cards, data interface cards, packet engine cards, service cards, smartcards, Switch cards, relay access cards, and the like. The term network card also encompasses modules, units, and assemblies that may include multiple printed circuit boards. Each network card 206 may support a single communications protocol, may support a number of related protocols, or may support a number of unrelated protocols. One or more network cards 206 or their equivalent may be permanently installed in a self-contained test unit or tests appliance. *See e.g.,* Jana, 5:27-43: Each network card 206 may contain one or more port unit 220. Each port unit 220 may include circuits and software to generate test traffic and/or to receive and analyze test traffic. Each port unit may be coupled to the test administrator 210. Each port unit 220 may connect to the network 290 through one or more ports. Each port unit 220 may be connected to the network 290 through a communication medium 295, which may be a wire, an optical fiber, a wireless link, or other communication medium. Each port unit 220 refers to a module within the network test equipment that connects to the DUT 250 at a port along either the N-S communication axis or the E-W communication axis. Each port unit 220 connected to the DUT may be both a source of test traffic and a destination for test traffic. Each port unit 220 may emulate a plurality of logical source and/or destination addresses. Each port unit 220 may emulate a plurality of network users, clients, peers, servers, or other network devices.

	See e.g., Jana, 6:33-59: The port CPU 360 may include a processor (not shown), a memory 365 coupled to the processor, and various specialized units, circuits, software and interfaces for providing the functionality and features described here. The processes, functionality and features may be embodied in whole or in part in software which operates on the processor and may be in the form of firmware, an application program, an applet (e.g., a Java applet), a browser plug-in, a COM object, a dynamic linked library (DLL), a script, one or more subroutines, or an operating system component or service. The hardware and software and their functions may be distributed such that some functions are performed by the processor and others by other devices. The memory 365 may include random access memory, or a combination of random access memory and other memory such as read-only memory and flash memory. The memory 365 may store both data and program instructions for execution by the processor. The memory 3 65 may include a received packet buffer. The received packet buffer may have capacity to store at least the longest anticipated received packet. The received packet buffer may have capacity to store multiple packets. A traffic receiver 370 may write one or more received packets into the received packet buffer, for example using direct memory access. After each received packet is stored, the traffic receiver 380 may send a signal (such as an interrupt, a flag, or another signal) to the processor 360 indicating that a received packet is waiting in the received packet buffer. *See e.g.,* Jana, 6:60-7:26: The memory 365 may include a transmit packet buffer. The transmit packet buffer may have capacity to store at least the longest anticipated packet to be transmitted. The transmit packet buffer may have capacity to store multiple packets. The processor 360 may write one or more packets to be transmitted into the transmit packet buffer. After each packet to be transmitted is stored, the processor 360 may send a -signal (such as an interrupt, a flag, or another signal) to a traffic generator 330 indicating that a packet to be transmitted is waiting in the transmit packet buffer. The traffic generator 360 may then extract the packet to be transmitted from the transmit packet buffer, for example by direct memory access. Programs stored in the memory 365 may include a TCP layer program that causes the processor to open, manage, and close TCP connections with device under test 350. The TCP layer program may be a layer within a TCP/IP protocol stack operating on the port CPU. The port CPU may be configured to maintain

IPR2025-00478
Concurrent Ventures LLC, XtreamEdge Inc. EX2045

	a large plurality of concurrent TCP connections with the device under test 350, and may open a million or more TCP connections during a network test session. For each TCP connection, the processor may send packets via a buffer in the memory 365 and the traffic generator 330, and may and receive packets via the traffic receiver 370 and a receive buffer in the memory 365. For each TCP connection, the processor may store TCP connection data in a TCP connection memory within the memory 365. The stored TCP connection data may include, for example, the state of the connection, the highest sequence number transmitted, and the highest sequence number acknowledged. Programs stored in the memory 365 may also include a capture routine that causes the processor 360 to capture selected packets from each session. The captured packets may be extracted and stored in a capture memory 340 within the memory 365. *See e.g.,* Jana, 10:66-11:13: The methods described herein may be implemented on one or more FPGAs and/or other hardware devices, such as, for example, digital logic devices. The methods described herein may be implemented as software, namely network testing software, running on a network testing system and executed by a processor, Such as a processor on a network card or a processor in a blade or other card with a processor in a network testing system. The network testing software may be stored on a volatile or nonvolatile memory device or storage medium included in or on and/or coupled with a computing device, a network testing system, a network card, or other card. The methods may be implemented on one or more network cards in a single network testing system or may be implemented on one or more network cards on each of two or more network testing systems. *See e.g.,* Jana, FIG. 2:

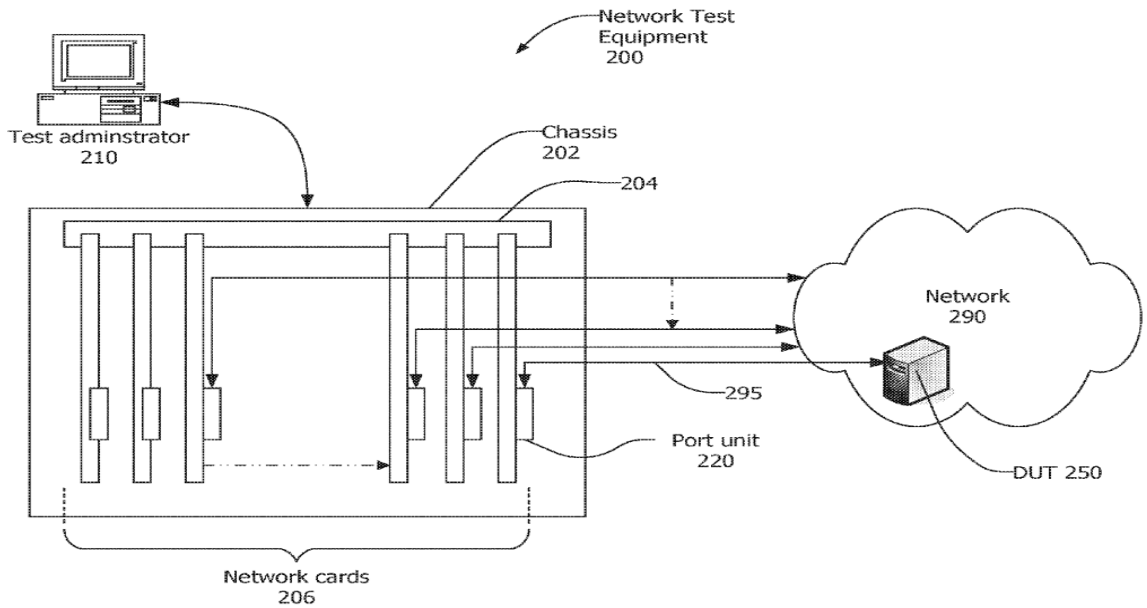
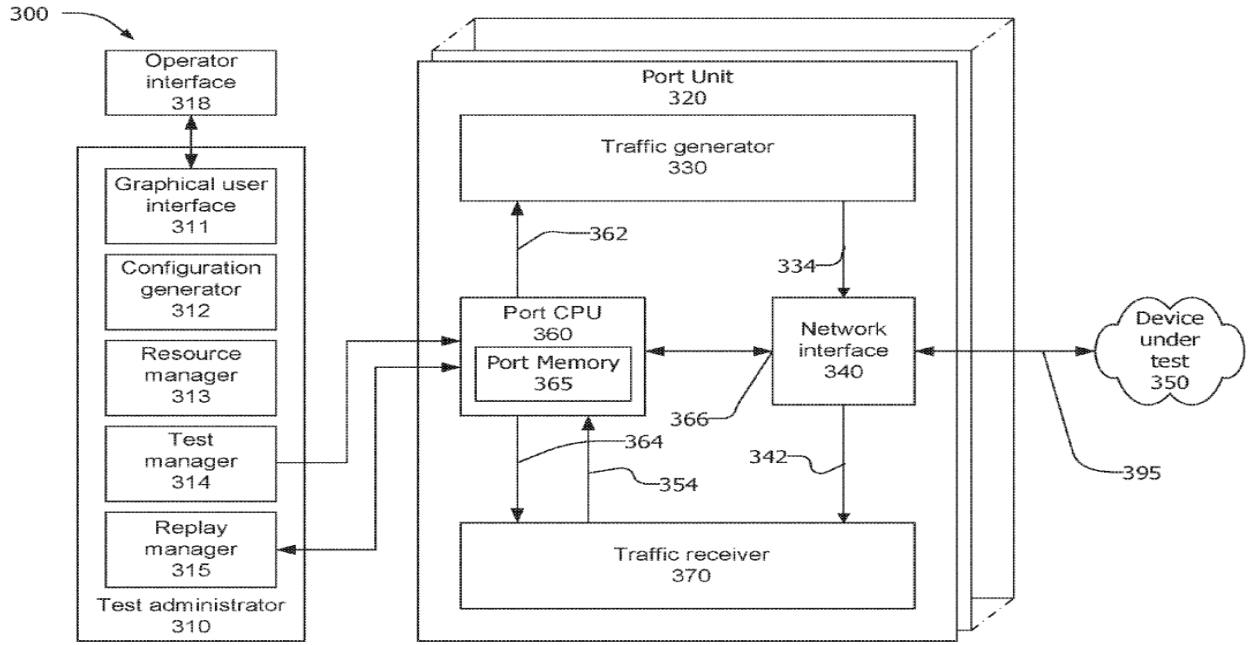
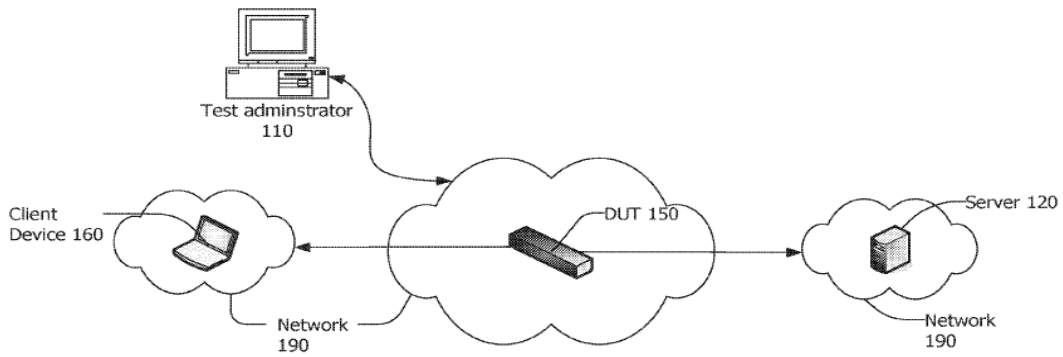


FIG. 2

See e.g., Jana, FIG. 3:

	 FIG. 3

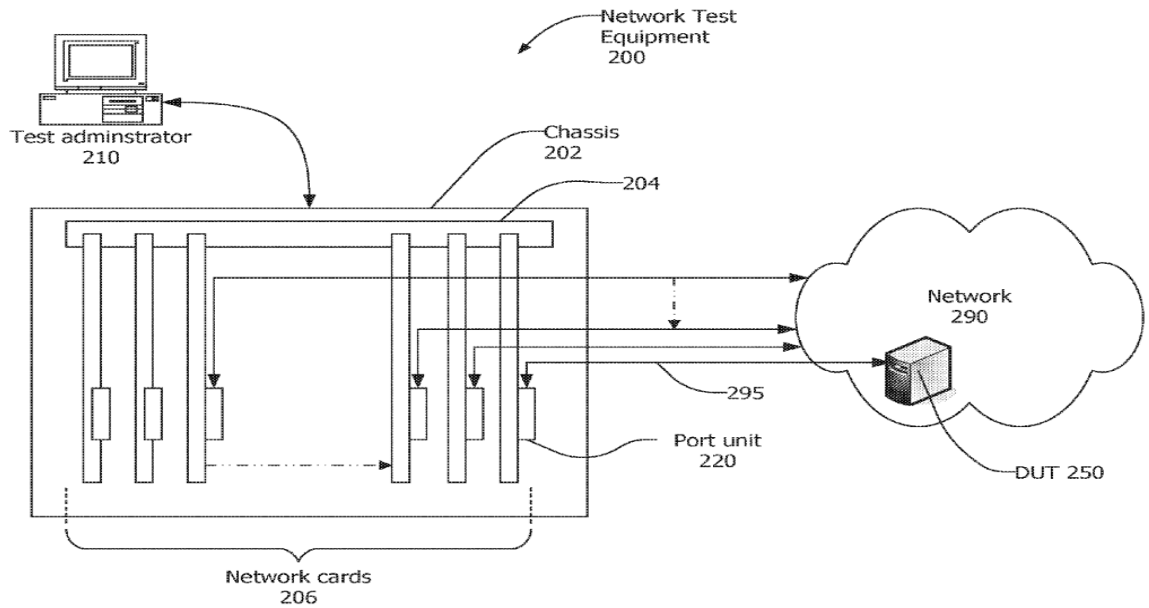
[1.d] the plurality of swappable pods or cards being user-configurable to implement data flow processing architectures; and	Jana discloses or renders obvious “the plurality of swappable pods or cards being user-configurable to implement data flow processing architectures,” as shown in at least the following exemplary citations: *See e.g.,* Jana, Abstract: There is disclosed a system and method for replaying a network capture comprising loading a network capture file that displays at least one traffic flow comprising a sequence of packets, selecting at least one packet to display on a viewer and transmitting the network traffic to a device under test wherein the selected packet is displayed on the viewer when the packet is transmitted to the device under test. *See e.g.,* Jana, 4:27-39:

	FIG. 1 shows a block diagram of a network environment 100 for replaying network captures in a testing environment on a datacenter device. Replayed network captures comprise packets and associated responses from traffic flows that were transmitted previously on a network and are being replayed on a network after being captured and stored. The environment may include a test administrator 110, server 120, a device under test (DUT) 150, a client device 160 and a network 190. For ease of description, the DUT 150 of FIG. 1 is coupled to a single client device and a single server. The DUT in an actual testing environment may be connected to a large number of emulated client devices and a large number of emulated servers. *See e.g.,* Jana, FIG.1:  FIG. 1
[1.e] a network coupled to the one or more swappable	Jana discloses or renders obvious “a network coupled to the one or more swappable pods or cards and supporting messaging-based communication using packets each having a header with a chassis identifier, a board identifier, a module identifier, an instance identifier, and a type identifier, so that each type of module,

pods or cards and supporting messaging-based communication using packets each having a header with a chassis identifier, a board identifier, a module identifier, an instance identifier, and a type identifier, so that each type of module, each instance of a type of module, and each module on each board in each chassis can be addressed through the header.	each instance of a type of module, and each module on each board in each chassis can be addressed through the header.” Plaintiffs have cited the use of Internet Protocol (IP) packets as allegedly satisfying limitation [1.e]. *See* Compl. at 55; ’753 Infringement Contentions at 44-45. Jana discloses or renders obvious claim limitation [1.e] under Plaintiffs’ apparent interpretation of the claim. *See e.g.,* Jana, 4:51-5:45: FIG. 2 shows a block diagram of a network environment. The environment may include network test equipment 200 and a network 290 which includes one or more devices under test (DUT) 250. The DUT 250 may be any device capable of communicating over the network 290. The network test equipment 200 may be a network testing device, performance analyzer, conformance validation system, network analyzer, or network management system. The network test equipment 200 may include one or more network cards 206 and a backplane 204 contained or enclosed within a chassis 202. The chassis 202 may be a fixed or portable chassis, cabinet, or enclosure suitable to contain the network test equipment. The network test equipment 200 may be an integrated unit, as shown in FIG. 2. Alternatively, the network test equipment 200 may comprise a number of separate units cooperative to provide traffic generation and/or analysis. The network test equipment 200 and the network cards 206 may support one or more well known standards or protocols such as the various Ethernet and Fibre Channel standards, and may support proprietary protocols as well. The network test equipment 200 and the device under test 250 may communicate simultaneously with one another, and there may be plural logical communications paths between the network test equipment 200 and a device under test 250. The network itself may be comprised of numerous nodes providing numerous physical and logical paths for data to travel. The network cards 206 may include one or more field programmable gate arrays (FPGAs), application specific integrated circuits (ASICs), programmable logic devices (PLDs), programmable logic arrays (PLAs), processors, and other kinds of devices. In addition, the network cards 206 may include software and/or firmware. The term network card encompasses line cards, test cards, analysis cards, network line cards, load modules, interface cards, network interface cards, data interface cards, packet engine cards, service cards, smart cards, switch cards, relay access cards, and the like. The term network card also encompasses modules, units, and assemblies that may include multiple printed circuit boards. Each network card 206 may support a single communications protocol, may support a number of related

	protocols, or may support a number of unrelated protocols. One or more network cards 206 or their equivalent may be permanently installed in a self-contained test unit or tests appliance. Each network card 206 may contain one or more port unit 220. Each port unit 220 may include circuits and software to generate test traffic and/or to receive and analyze test traffic. Each port unit may be coupled to the test administrator 210. Each port unit 220 may connect to the network 290 through one or more ports. Each port unit 220 may be connected to the network 290 through a communication medium 295, which may be a wire, an optical fiber, a wireless link, or other communication medium. Each port unit 220 refers to a module within the network test equipment that connects to the DUT 250 at a port along either the N-S communication axis or the E-W communication axis. Each port unit 220 connected to the DUT may be both a source of test traffic and a destination for test traffic. Each port unit 220 may emulate a plurality of logical source and/or destination addresses. Each port unit 220 may emulate a plurality of network users, clients, peers, servers, or other network devices. The backplane 204 may serve as a bus or communications medium for the network cards 206. The backplane 204 may also provide power to the network cards 206. *See e.g.,* Jana, 6:8-33: The network traffic on network 290 and produced and received by various nodes, including the ports on the network cards 206 includes both control plane traffic and data plane traffic. The control plane and data plane are virtualizations that are useful in conceptualizing the roles of two kinds of network traffic. Control plane traffic includes packets that implement the various Supported communications protocols, including administrative communications among nodes to maintain communications among the nodes. An example of control plane traffic are routing notices or packets informing a second node that a first node is inaccessible. Data plane traffic includes packets carrying data between nodes. The data plane carries network traffic that communicates data over a network. For example, those packets that carry data for an audio stream are considered to be communicated over the data plane and are considered data plane traffic. Referring now to FIG. 3, a test system 300, which may be the network test equipment 300 may include a test administrator 310 coupled to a plurality of port units including an

	exemplary port unit 320. The port unit 320 may include a port central processor unit 360 (CPU), a traffic generator unit 330, a traffic receiver unit 370, and a network interface unit 340 which couples the port unit 320 to a device under test 350. The port unit 320 may be all or part of a network card such as the network cards 206. *See e.g.,* Jana, 10:66-11:14: The methods described herein may be implemented on one or more FPGAs and/or other hardware devices, such as, for example, digital logic devices. The methods described herein may be implemented as software, namely network testing software, running on a network testing system and executed by a processor, Such as a processor on a network card or a processor in a blade or other card with a processor in a network testing system. The network testing software may be stored on a volatile or nonvolatile memory device or storage medium included in or on and/or coupled with a computing device, a network testing system, a network card, or other card. The methods may be implemented on one or more network cards in a single network testing system or may be implemented on one or more network cards on each of two or more network testing systems. *See e.g.,* Jana, FIG. 2:

	 FIG. 2 *See e.g.,* Jana, FIG. 3:

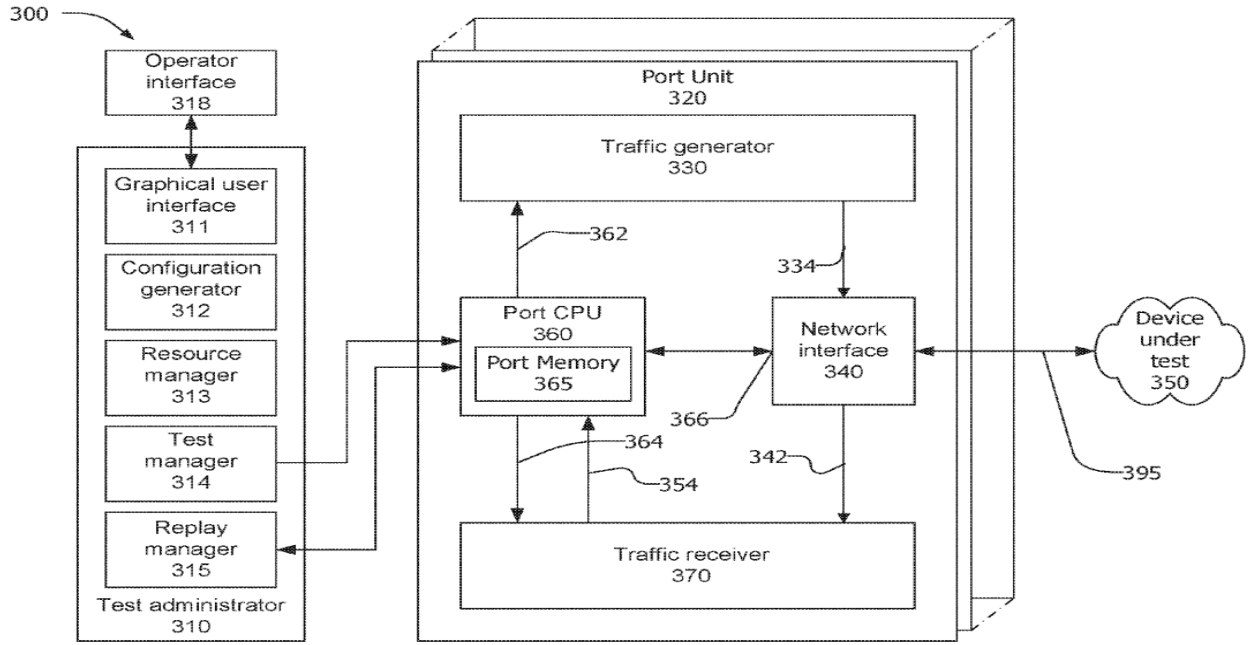


FIG. 3

See e.g., Jana, 1:21-34:

> Datacenter devices generate and test heavy traffic loads that are typical of realistic networks. At times when devices are being tested, it is desirable to replay (or re-perform) a set of data traffic that was used to test a network. However, there 25 exists a difficulty in communications networks in replaying various types of traffic. Specifically, there exists a difficulty in replaying TCP captures having multiple HTTP requests and responses in a TCP session. Each HTTP requests/responses in
> a TCP session from client to device under test may be sent to 30 multiple TCP sessions from device under test to server. Challenge is that replay module is stateless and does not have knowledge of any protocol.

	See e.g., Jana, 2:37-55: Layer 4 protocols govern end-to-end message delivery in a network. In particular, the Transmission Control Protocol (TCP) provides for reliable delivery of packets streams using a system of sequential acknowledgement and retransmission when necessary. TCP is a connection-oriented protocol in which two devices exchange messages to open a virtual connection via the network. Once a connection is opened, bidirectional communications may occur between the connected devices. The connection may exist until closed by one of the devices. Opening and closing a connection both require several steps at which specific messages are exchanged between the two devices. A connection may also be closed when an anticipated response is not received by one device for a pre determined period of time, commonly called a “time-out'. A TCP connection is considered to be “stateful since each device must maintain information describing the state of the connection (being opened, established, being closed), what data has been sent, and what sent data has been acknowledged. *See e.g.,* Jana, 2:37-55: A TCP connection is defined by a pair of IP Addresses (fields in the IP header of each packet of the TCP connection) and a pair of TCP Ports (fields in the TCP header of each TCP packet). TCP Ports define what higher level protocol or process is using the services of the TCP connection. TCP Ports should not be confused with hardware ports (connections between hardware devices and a network). The header of each TCP packet contains a sequence number identifying the first data byte in the TCP packet, and an acknowledgement number identifying the next data byte expected to be received. The sequence and acknowledgement numbers are, in simplified terms, a count of the number of data bytes sent and received, respectively, over the TCP connection. Two devices communicating via a TCP connection maintain data indicating the highest sequence number transmitted and the highest acknowledgement number received. This information can be used to retransmit packets that are not acknowledged in a timely manner. *See e.g.,* Jana, 7:5-21: Programs stored in the memory 365 may include a TCP layer program that causes the processor to open, manage, and close TCP connections with device under test 350. The TCP layer program may be a layer

	within a TCP/IP protocol stack operating on the port CPU. The port CPU may be configured to maintain a large plurality of concurrent TCP connections with the device under test 350, and may open a million or more TCP connections during a network test session. For each TCP connection, the processor may send packets via a buffer in the memory 365 and the traffic generator 330, and may and receive packets via the traffic receiver 370 and a receive buffer in the memory 365. For each TCP connection, the processor may store TCP connection data in a TCP connection memory within the memory 365. The stored TCP connection data may include, for example, the state of the connection, the highest sequence number transmitted, and the highest sequence number acknowledged. *See e.g.,* Jana, 8:13-36: To establish and use TCP connections, the port CPU 360 may prepare appropriate TCP packets and provide the TCP packets to the traffic generator 330. The traffic generator 330 may insert the TCP packets into the outgoing test traffic. The traffic receiver 370 may separate received TCP packets from received stateless traffic and send the received TCP packets to the port CPU 220 for processing. Some amount of interactive communications may be required between the port unit 320 and the device under test 350 during a test session. The interactive communications may include, for example, control plane traffic, TCP connections and application-layer transactions. Such traffic may be initiated, managed, and processed by the port CPU 360 and thus may be referred to as “CPU traffic”. As shown in FIG.3, the port CPU 360 may be coupled to the network interface unit 340 such that CPU traffic 366 may be communicated between the port CPU 360 and the device under test 350 via the network interface unit 340. Although not shown in FIG.3, outgoing CPU traffic may be routed from the Port CPU 360 to the network interface unit 340 via the traffic generator unit 330, and incoming CPU traffic may be routed from the network interface unit to the port CPU 360 via the traffic receiver unit 350.

2. The platform of claim 1, further comprising: at least a subset of the one or more	Jana discloses or renders obvious “at least a subset of the one or more swappable pods or cards having an option for direct connection to one or more pods, cards or components, bypassing the module messaging interface network.” Plaintiffs allege infringement of claim 2 based on multiple cards being deployed in a server. ’753 Infringement Contentions at 52 (“multiple AMD Pensando DSCs can be deployed within a single server and

swappable pods or cards having an option for direct connection to one or more pods, cards or components, bypassing the module messaging interface network.	these AMD Pensando DSCs are connected to each other without using the module messaging interface network”). Jana discloses or renders obvious claim 2 under Plaintiffs’ apparent interpretation of the claim. *See e.g.,* Jana, 4:51-5:45: FIG. 2 shows a block diagram of a network environment. The environment may include network test equipment 200 and a network 290 which includes one or more devices under test (DUT) 250. The DUT 250 may be any device capable of communicating over the network 290. The network test equipment 200 may be a network testing device, performance analyzer, conformance validation system, network analyzer, or network management system. The network test equipment 200 may include one or more network cards 206 and a backplane 204 contained or enclosed within a chassis 202. The chassis 202 may be a fixed or portable chassis, cabinet, or enclosure suitable to contain the network test equipment. The network test equipment 200 may be an integrated unit, as shown in FIG. 2. Alternatively, the network test equipment 200 may comprise a number of separate units cooperative to provide traffic generation and/or analysis. The network test equipment 200 and the network cards 206 may support one or more well known standards or protocols such as the various Ethernet and Fibre Channel standards, and may support proprietary protocols as well. The network test equipment 200 and the device under test 250 may communicate simultaneously with one another, and there may be plural logical communications paths between the network test equipment 200 and a device under test 250. The network itself may be comprised of numerous nodes providing numerous physical and logical paths for data to travel. The network cards 206 may include one or more field programmable gate arrays (FPGAs), application specific integrated circuits (ASICs), programmable logic devices (PLDs), programmable logic arrays (PLAs), processors, and other kinds of devices. In addition, the network cards 206 may include software and/or firmware. The term network card encompasses line cards, test cards, analysis cards, network line cards, load modules, interface cards, network interface cards, data interface cards, packet engine cards, service cards, smart cards, switch cards, relay access cards, and the like. The term network card also encompasses modules, units, and assemblies that may include multiple printed circuit boards. Each network card 206 may support a single communications protocol, may support a number of related protocols, or may support a number of unrelated protocols. One or more network cards 206 or their equivalent may be permanently installed in a self-contained test unit or tests appliance.

	Each network card 206 may contain one or more port unit 220. Each port unit 220 may include circuits and software to generate test traffic and/or to receive and analyze test traffic. Each port unit may be coupled to the test administrator 210. Each port unit 220 may connect to the network 290 through one or more ports. Each port unit 220 may be connected to the network 290 through a communication medium 295, which may be a wire, an optical fiber, a wireless link, or other communication medium. Each port unit 220 refers to a module within the network test equipment that connects to the DUT 250 at a port along either the N-S communication axis or the E-W communication axis. Each port unit 220 connected to the DUT may be both a source of test traffic and a destination for test traffic. Each port unit 220 may emulate a plurality of logical source and/or destination addresses. Each port unit 220 may emulate a plurality of network users, clients, peers, servers, or other network devices. The backplane 204 may serve as a bus or communications medium for the network cards 206. The backplane 204 may also provide power to the network cards 206. *See e.g.,* Jana, 6:8-33: The network traffic on network 290 and produced and received by various nodes, including the ports on the network cards 206 includes both control plane traffic and data plane traffic. The control plane and data plane are virtualizations that are useful in conceptualizing the roles of two kinds of network traffic. Control plane traffic includes packets that implement the various Supported communications protocols, including administrative communications among nodes to maintain communications among the nodes. An example of control plane traffic are routing notices or packets informing a second node that a first node is inaccessible. Data plane traffic includes packets carrying data between nodes. The data plane carries network traffic that communicates data over a network. For example, those packets that carry data for an audio stream are considered to be communicated over the data plane and are considered data plane traffic. Referring now to FIG. 3, a test system 300, which may be the network test equipment 300 may include a test administrator 310 coupled to a plurality of port units including an exemplary port unit 320. The port unit 320 may include a port central processor unit 360 (CPU), a traffic generator unit 330, a traffic receiver unit 370, and a network interface unit 340 which couples the port

	unit 320 to a device under test 350. The port unit 320 may be all or part of a network card such as the network cards 206. *See e.g.,* Jana, 10:66-11:14: The methods described herein may be implemented on one or more FPGAs and/or other hardware devices, such as, for example, digital logic devices. The methods described herein may be implemented as software, namely network testing software, running on a network testing system and executed by a processor, Such as a processor on a network card or a processor in a blade or other card with a processor in a network testing system. The network testing software may be stored on a volatile or nonvolatile memory device or storage medium included in or on and/or coupled with a computing device, a network testing system, a network card, or other card. The methods may be implemented on one or more network cards in a single network testing system or may be implemented on one or more network cards on each of two or more network testing systems. *See e.g.,* Jana, FIG. 2:

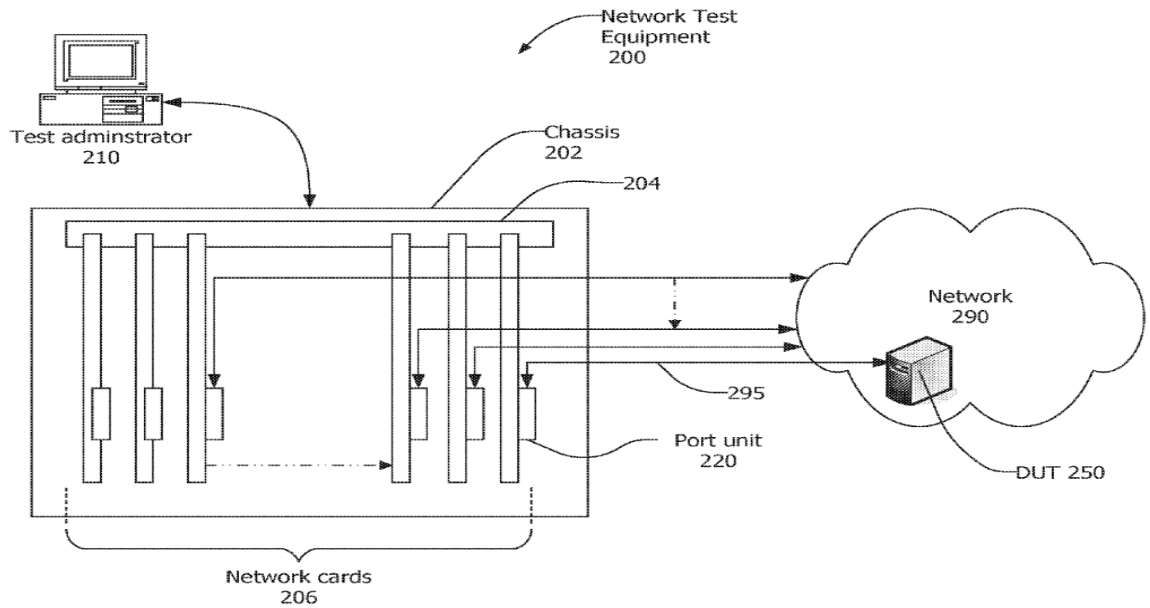
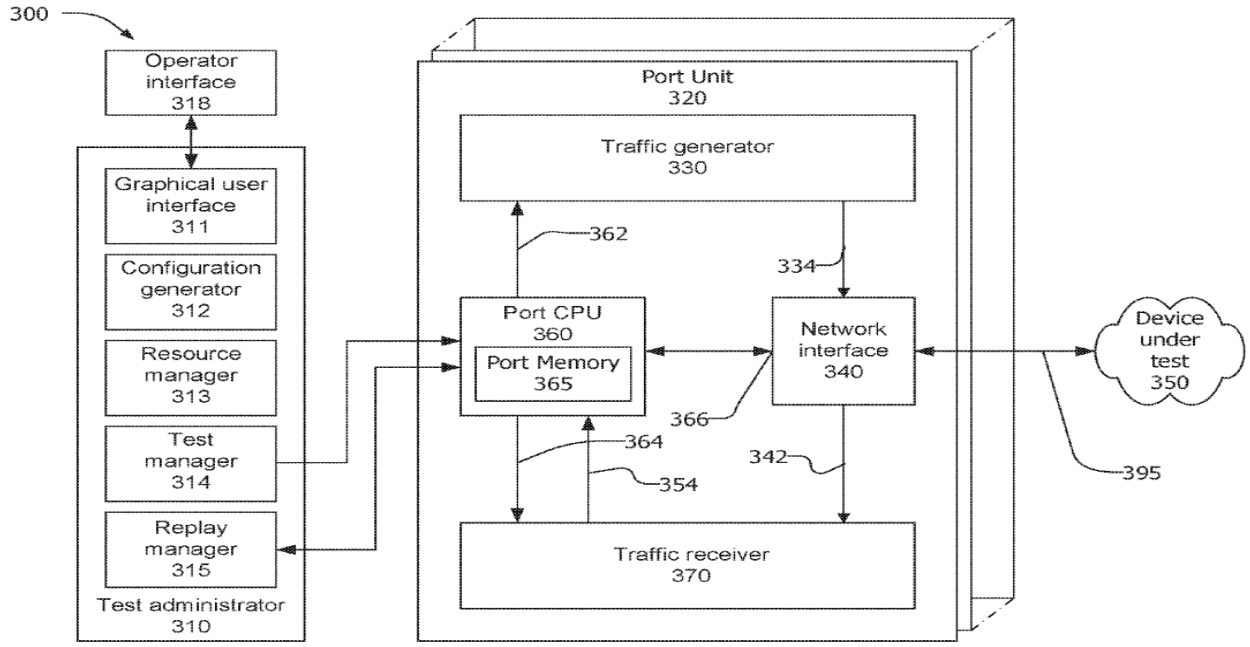


FIG. 2

See e.g., Jana, FIG. 3:

	 FIG. 3
3. The platform of claim 1, wherein the one or more swappable pods or cards comprises at least one processor card having a at least one processor allocatable for offloading hardware tasks to software, orchestrating data	Jana discloses or renders obvious “the one or more swappable pods or cards comprises at least one processor card having a at least one processor allocatable for offloading hardware tasks to software, orchestrating data flows, orchestrating configurations of hardware modules, and executing in cooperation with a sandbox of one of the plurality of swappable pods or cards.” Plaintiffs cited the presence of processors that allegedly “work in cooperation with a sandbox” as allegedly satisfying claim 3. ’753 Infringement Contentions at 53. Jana discloses or renders obvious claim 3 under Plaintiffs’ apparent interpretation of the claim. *See e.g.,* Jana, 4:51-5:45: FIG. 2 shows a block diagram of a network environment. The environment may include network test equipment 200 and a network 290 which includes one or more devices under test (DUT) 250. The DUT 250 may be any device capable of communicating over the network 290.

flows, orchestrating configurations of hardware modules, and executing in cooperation with a sandbox of one of the plurality of swappable pods or cards.	The network test equipment 200 may be a network testing device, performance analyzer, conformance validation system, network analyzer, or network management system. The network test equipment 200 may include one or more network cards 206 and a backplane 204 contained or enclosed within a chassis 202. The chassis 202 may be a fixed or portable chassis, cabinet, or enclosure suitable to contain the network test equipment. The network test equipment 200 may be an integrated unit, as shown in FIG. 2. Alternatively, the network test equipment 200 may comprise a number of separate units cooperative to provide traffic generation and/or analysis. The network test equipment 200 and the network cards 206 may support one or more well known standards or protocols such as the various Ethernet and Fibre Channel standards, and may support proprietary protocols as well. The network test equipment 200 and the device under test 250 may communicate simultaneously with one another, and there may be plural logical communications paths between the network test equipment 200 and a device under test 250. The network itself may be comprised of numerous nodes providing numerous physical and logical paths for data to travel. The network cards 206 may include one or more field programmable gate arrays (FPGAs), application specific integrated circuits (ASICs), programmable logic devices (PLDs), programmable logic arrays (PLAs), processors, and other kinds of devices. In addition, the network cards 206 may include software and/or firmware. The term network card encompasses line cards, test cards, analysis cards, network line cards, load modules, interface cards, network interface cards, data interface cards, packet engine cards, service cards, smart cards, switch cards, relay access cards, and the like. The term network card also encompasses modules, units, and assemblies that may include multiple printed circuit boards. Each network card 206 may support a single communications protocol, may support a number of related protocols, or may support a number of unrelated protocols. One or more network cards 206 or their equivalent may be permanently installed in a self-contained test unit or tests appliance. Each network card 206 may contain one or more port unit 220. Each port unit 220 may include circuits and software to generate test traffic and/or to receive and analyze test traffic. Each port unit may be coupled to the test administrator 210. Each port unit 220 may connect to the network 290 through one or more ports. Each port unit 220 may be connected to the network 290 through a communication medium 295, which may be a wire, an optical fiber, a wireless link, or other communication medium. Each port unit 220 refers to a module within the network test equipment that connects to the DUT 250 at a port along either the N-S communication axis or the E-W communication axis. Each port unit 220 connected

	to the DUT may be both a source of test traffic and a destination for test traffic. Each port unit 220 may emulate a plurality of logical source and/or destination addresses. Each port unit 220 may emulate a plurality of network users, clients, peers, servers, or other network devices. The backplane 204 may serve as a bus or communications medium for the network cards 206. The backplane 204 may also provide power to the network cards 206. *See e.g.,* Jana, 6:8-33: The network traffic on network 290 and produced and received by various nodes, including the ports on the network cards 206 includes both control plane traffic and data plane traffic. The control plane and data plane are virtualizations that are useful in conceptualizing the roles of two kinds of network traffic. Control plane traffic includes packets that implement the various Supported communications protocols, including administrative communications among nodes to maintain communications among the nodes. An example of control plane traffic are routing notices or packets informing a second node that a first node is inaccessible. Data plane traffic includes packets carrying data between nodes. The data plane carries network traffic that communicates data over a network. For example, those packets that carry data for an audio stream are considered to be communicated over the data plane and are considered data plane traffic. Referring now to FIG. 3, a test system 300, which may be the network test equipment 300 may include a test administrator 310 coupled to a plurality of port units including an exemplary port unit 320. The port unit 320 may include a port central processor unit 360 (CPU), a traffic generator unit 330, a traffic receiver unit 370, and a network interface unit 340 which couples the port unit 320 to a device under test 350. The port unit 320 may be all or part of a network card such as the network cards 206. *See e.g.,* Jana, 10:66-11:14: The methods described herein may be implemented on one or more FPGAs and/or other hardware devices, such as, for example, digital logic devices. The methods described herein may be implemented as software, namely network testing software, running on a network testing system and executed by a processor, Such as a processor on a network card or a

	processor in a blade or other card with a processor in a network testing system. The network testing software may be stored on a volatile or nonvolatile memory device or storage medium included in or on and/or coupled with a computing device, a network testing system, a network card, or other card. The methods may be implemented on one or more network cards in a single network testing system or may be implemented on one or more network cards on each of two or more network testing systems. *See e.g.,* Jana, 5:10-26: The network cards 206 may include one or more field programmable gate arrays (FPGAs), application specific integrated circuits (ASICs), programmable logic devices (PLDS), programmable logic arrays (PLAS), processors, and other kinds of devices. In addition, the network cards 206 may include software and/or firmware. The term network card encompasses line cards, test cards, analysis cards, network line cards, load modules, interface cards, network interface cards, data interface cards, packet engine cards, service cards, smartcards, Switch cards, relay access cards, and the like. The term network card also encompasses modules, units, and assemblies that may include multiple printed circuit boards. Each network card 206 may support a single communications protocol, may support a number of related protocols, or may support a number of unrelated protocols. One or more network cards 206 or their equivalent may be permanently installed in a self-contained test unit or tests appliance. *See e.g.,* Jana, 5:27-43: Each network card 206 may contain one or more port unit 220. Each port unit 220 may include circuits and software to generate test traffic and/or to receive and analyze test traffic. Each port unit may be coupled to the test administrator 210. Each port unit 220 may connect to the network 290 through one or more ports. Each port unit 220 may be connected to the network 290 through a communication medium 295, which may be a wire, an optical fiber, a wireless link, or other communication medium. Each port unit 220 refers to a module within the network test equipment that connects to the DUT 250 at a port along either the N-S communication axis or the E-W communication axis. Each port unit 220 connected to the DUT may be both a source of test traffic and a destination for test traffic. Each port unit 220 may emulate a plurality of logical source and/or destination addresses. Each port unit 220 may emulate a plurality of network users, clients, peers, servers, or other network devices.

	See e.g., Jana, 6:33-59: The port CPU 360 may include a processor (not shown), a memory 365 coupled to the processor, and various specialized units, circuits, software and interfaces for providing the functionality and features described here. The processes, functionality and features may be embodied in whole or in part in software which operates on the processor and may be in the form of firmware, an application program, an applet (e.g., a Java applet), a browser plug-in, a COM object, a dynamic linked library (DLL), a script, one or more subroutines, or an operating system component or service. The hardware and software and their functions may be distributed such that some functions are performed by the processor and others by other devices. The memory 365 may include random access memory, or a combination of random access memory and other memory such as read-only memory and flash memory. The memory 365 may store both data and program instructions for execution by the processor. The memory 3 65 may include a received packet buffer. The received packet buffer may have capacity to store at least the longest anticipated received packet. The received packet buffer may have capacity to store multiple packets. A traffic receiver 370 may write one or more received packets into the received packet buffer, for example using direct memory access. After each received packet is stored, the traffic receiver 380 may send a signal (such as an interrupt, a flag, or another signal) to the processor 360 indicating that a received packet is waiting in the received packet buffer. *See e.g.,* Jana, 6:60-7:26: The memory 365 may include a transmit packet buffer. The transmit packet buffer may have capacity to store at least the longest anticipated packet to be transmitted. The transmit packet buffer may have capacity to store multiple packets. The processor 360 may write one or more packets to be transmitted into the transmit packet buffer. After each packet to be transmitted is stored, the processor 360 may send a -signal (such as an interrupt, a flag, or another signal) to a traffic generator 330 indicating that a packet to be transmitted is waiting in the transmit packet buffer. The traffic generator 360 may then extract the packet to be transmitted from the transmit packet buffer, for example by direct memory access. Programs stored in the memory 365 may include a TCP layer program that causes the processor to open, manage, and close TCP connections with device under test 350. The TCP layer program may be a layer within a TCP/IP protocol stack operating on the port CPU. The port CPU may be configured to maintain

	a large plurality of concurrent TCP connections with the device under test 350, and may open a million or more TCP connections during a network test session. For each TCP connection, the processor may send packets via a buffer in the memory 365 and the traffic generator 330, and may and receive packets via the traffic receiver 370 and a receive buffer in the memory 365. For each TCP connection, the processor may store TCP connection data in a TCP connection memory within the memory 365. The stored TCP connection data may include, for example, the state of the connection, the highest sequence number transmitted, and the highest sequence number acknowledged. Programs stored in the memory 365 may also include a capture routine that causes the processor 360 to capture selected packets from each session. The captured packets may be extracted and stored in a capture memory 340 within the memory 365. *See e.g.,* Jana, 10:66-11:13: The methods described herein may be implemented on one or more FPGAs and/or other hardware devices, such as, for example, digital logic devices. The methods described herein may be implemented as software, namely network testing software, running on a network testing system and executed by a processor, Such as a processor on a network card or a processor in a blade or other card with a processor in a network testing system. The network testing software may be stored on a volatile or nonvolatile memory device or storage medium included in or on and/or coupled with a computing device, a network testing system, a network card, or other card. The methods may be implemented on one or more network cards in a single network testing system or may be implemented on one or more network cards on each of two or more network testing systems. *See e.g.,* Jana, FIG. 2:

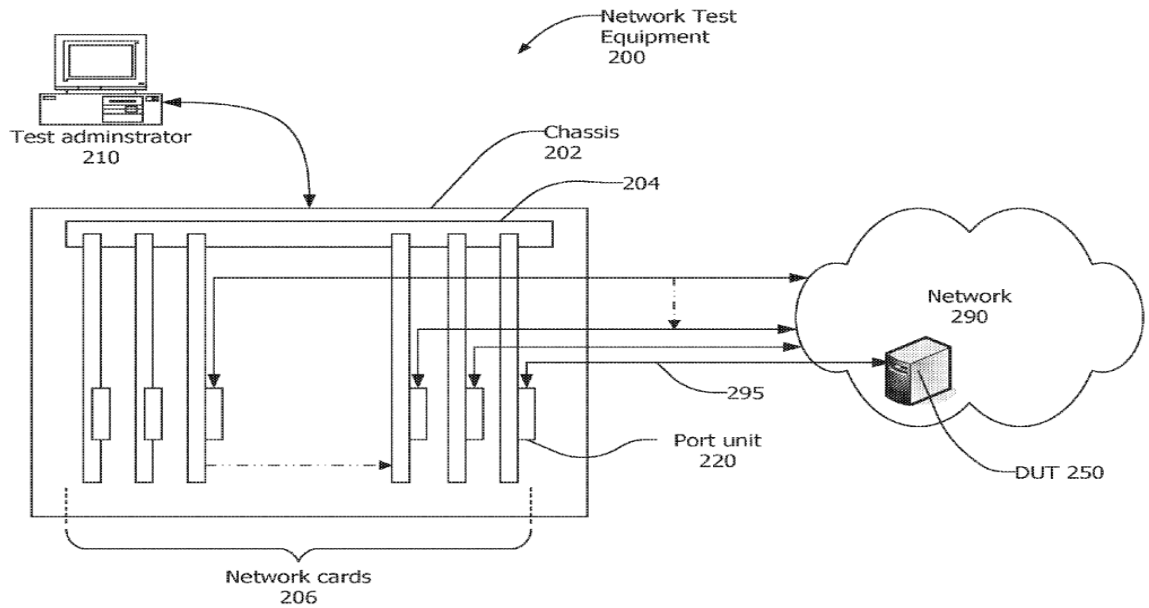
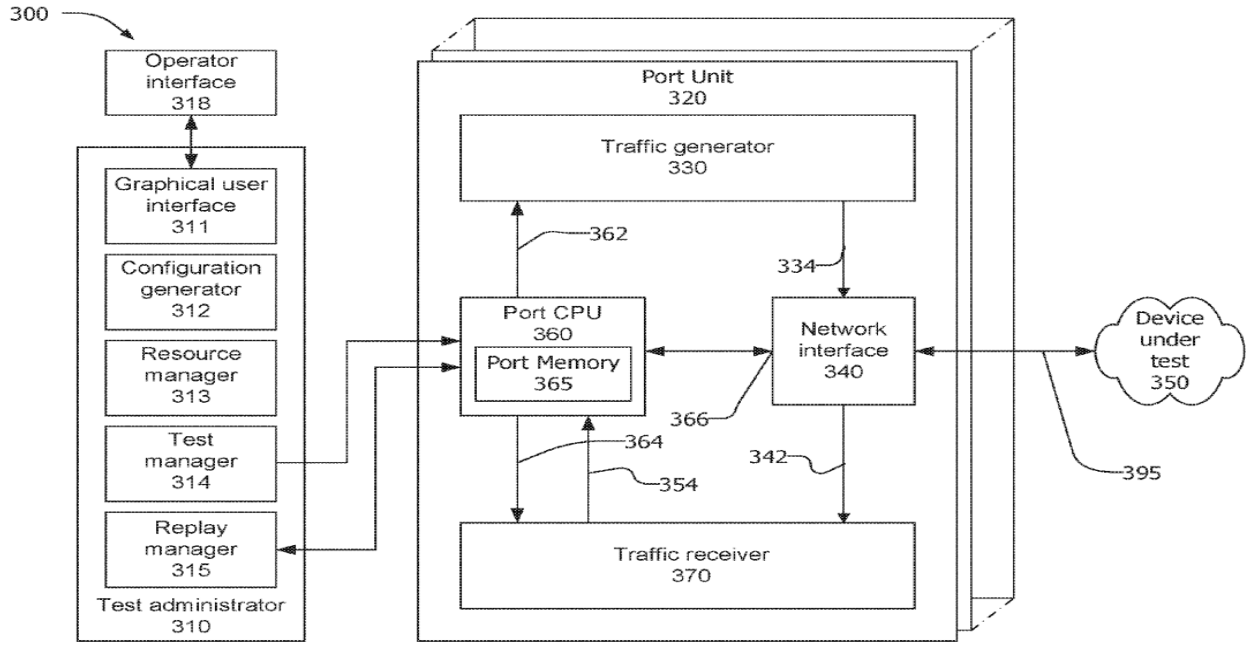
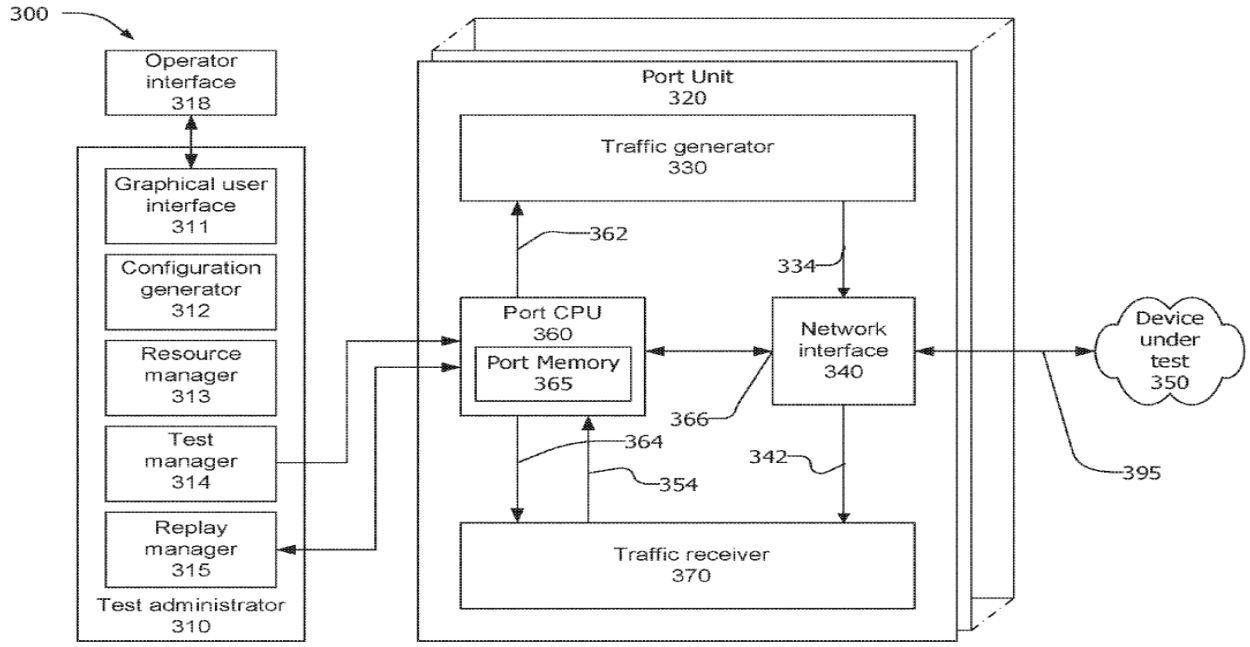


FIG. 2

See e.g., Jana, FIG. 3:

	 FIG. 3
4. The platform of claim 1, further comprising: at least one of the one or more swappable pods or cards having a plurality of types of memory and memory superposition with selectable	Jana discloses or renders obvious “at least one of the one or more swappable pods or cards having a plurality of types of memory and memory superposition with selectable arrangement of the plurality of types of memory as multilevel cache, interleaved memory, swapped memory over a specified address range, and memory allocated to specified address ranges according to parameterized task needs.” Plaintiffs allege infringement of claim 4 by citing support for “various types of memory to optimize performance.” ’753 Infringement Contentions at 53. Jana discloses or renders obvious claim 4 under Plaintiffs’ apparent interpretation of the claim. *See e.g.,* Jana, 6:33-59: The port CPU 360 may include a processor (not shown), a memory 365 coupled to the processor, and various specialized units, circuits, software and interfaces for providing the functionality and features described here. The processes, functionality and features may be embodied in whole or in part in

arrangement of the plurality of types of memory as multilevel cache, interleaved memory, swapped memory over a specified address range, and memory allocated to specified address ranges according to parameterized task needs.	software which operates on the processor and may be in the form of firmware, an application program, an applet (e.g., a Java applet), a browser plug-in, a COM object, a dynamic linked library (DLL), a script, one or more subroutines, or an operating system component or service. The hardware and software and their functions may be distributed such that some functions are performed by the processor and others by other devices. The memory 365 may include random access memory, or a combination of random access memory and other memory such as read-only memory and flash memory. The memory 365 may store both data and program instructions for execution by the processor. The memory 365 may include a received packet buffer. The received packet buffer may have capacity to store at least the longest anticipated received packet. The received packet buffer may have capacity to store multiple packets. A traffic receiver 370 may write one or more received packets into the received packet buffer, for example using direct memory access. After each received packet is stored, the traffic receiver 380 may send a signal (such as an interrupt, a flag, or another signal) to the processor 360 indicating that a received packet is waiting in the received packet buffer. *See e.g.,* Jana, 7:5-26: Programs stored in the memory 365 may include a TCP layer program that causes the processor to open, manage, and close TCP connections with device under test 350. The TCP layer program may be a layer within a TCP/IP protocol stack operating on the port CPU. The port CPU may be configured to maintain a large plurality of concurrent TCP connections with the device under test 350, and may open a million or more TCP connections during a network test session. For each TCP connection, the processor may send packets via a buffer in the memory 365 and the traffic generator 330, and may and receive packets via the traffic receiver 370 and a receive buffer in the memory 365. For each TCP connection, the processor may store TCP connection data in a TCP connection memory within the memory 365. The stored TCP connection data may include, for example, the state of the connection, the highest sequence number transmitted, and the highest sequence number acknowledged. *See e.g.,* Jana, FIG. 3:

	 FIG. 3

[7.pre] A method for data flow processing in a platform, comprising:	*See* claim element 1.pre.
[7.a]. configuring a plurality of swappable pods or cards in one or more chassis, coupled through a messaging	*See* claim elements 1.pre and 1.a.

interface network, to implement a data flow processing architecture;	
[7.b] configuring user-definable hardware modules or user-definable software modules in each of one or more of the plurality of swappable pods or cards;	*See* claim element 1.c.
[7.c] communicating a data flow in accordance with the implemented data flow processing architecture among one or more hardware modules or one or more software modules in each of the plurality of swappable pods or cards, including the user-definable hardware modules or user-definable software modules; and	*See* claim elements 1.c and 1.d.

[7.d] selecting a memory superposition in at least one of the plurality of swappable pods or cards having a plurality of types of memory with selectable arrangement of the plurality of types of memory as multilevel cache, interleaved memory, swapped memory over a specified address range, and memory allocated to specified address ranges according to parameterized task needs.	*See* claim 4.
8. The method of claim 7, wherein the communicating the data flow comprises: communicating the data flow through the messaging interface network, comprising a	*See* claim element 1.e.

network coupled to the plurality of swappable pods or cards and supporting messaging-based communication using packets each having a header with a chassis identifier, a board identifier, a module identifier, an instance identifier, and a type identifier, so that each type of module, each instance of a type of module, and each module on each board in each chassis can be addressed through the header.	
9. The method of claim 7, wherein the communicating the data flow comprises: bypassing the messaging interface network and communicating through a direct connection among two or more pods,	*See* claim 2.

cards or components.	
10. The method of claim 7, further comprising: allocating at least one processor of a processor card, from among the plurality of swappable pods or cards, to offloading hardware tasks to software, orchestrating data flows, orchestrating configurations of hardware modules, or executing in cooperation with a sandbox of one of the plurality of swappable pods or cards.	*See* claim 3.
[13.pre] A tangible, non-transitory, computer-readable media having instructions thereupon which, when executed by a processor, cause the	*See* claim elements 1.pre and 1.a-1.e.

processor to perform a method comprising:	
[13.a] configuring a plurality of swappable pods or cards in one or more chassis, coupled through a messaging interface network, to implement a data flow processing architecture;	*See* claim elements 1.a-1.c.
[13.b] configuring user-definable hardware modules or user-definable software modules in each of one or more of the plurality of swappable pods or cards;	*See* claim element 1.c.
[13.c] communicating a data flow in accordance with the implemented data flow processing architecture among one or more hardware modules or	*See* claim elements 1.c and 1.d.

one or more software modules in each of the plurality of swappable pods or cards, including the user-definable hardware modules or user-definable software modules; and	
[13.d] communicating the data flow through the messaging interface network, comprising a network coupled to the plurality of swappable pods or cards and supporting messaging-based communication using packets each having a header with a chassis identifier, a board identifier, a module identifier, an instance identifier, and a type identifier, so that each type of module, each instance of a type of	*See* claim element 1.e.

module, and each module on each board in each chassis can be addressed through the header.	
14. The computer-readable media of claim 13, wherein the method further comprises: allocating at least one processor of a processor card having a plurality of processors, from among the plurality of swappable pods or cards, to offloading hardware tasks to software, orchestrating data flows, orchestrating configurations of hardware modules, or executing in cooperation with a sandbox of one of the plurality of swappable pods or cards.	*See* claim 3.

15. The computer-readable media of claim 13, wherein the method further comprises: selecting a memory superposition in at least one of the plurality of swappable pods or cards having a plurality of types of memory with selectable arrangement of the plurality of types of memory as multilevel cache, interleaved memory, swapped memory over a specified address range, and memory allocated to specified address ranges according to parameterized task needs.	*See* claim 4.