UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/150,210	05/09/2016	Perinkulum I. Ganesh	AUS920120344US4	4880

77351 7590 12/19/2019

IBM CORP. (AUS)
C/O THE LAW OFFICE OF JAMES BAUDINO, PLLC
2313 ROOSEVELT DRIVE
SUITE A
ARLINGTON, TX 76016

EXAMINER	
KIRILLOVA, ELLEN A	

ART UNIT	PAPER NUMBER
2464	

MAIL DATE	DELIVERY MODE
12/19/2019	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

Ex parte PERINKULUM I. GANESH, DIMPU K. NATH, and MATTHEW R. OCHS

Appeal 2018-009030
Application 15/150,210
Technology Center 2400

BEFORE ELENI MANTIS MERCADER, DAVID C. MCKONE, and JOHN P. PINKERTON, *Administrative Patent Judges.*

MCKONE, *Administrative Patent Judge.*

DECISION ON APPEAL

STATEMENT OF THE CASE

Pursuant to 35 U.S.C. § 134(a), Appellant[1] appeals from the Examiner's decision to reject claims 1–18. We have jurisdiction under 35 U.S.C. § 6(b).

We REVERSE.

---

[1] We use the word Appellant to refer to "applicant" as defined in 37 C.F.R. § 1.42(a). Appellant identifies the real party in interest as International Business Machines Corp. Appeal Br. 2.

CLAIMED SUBJECT MATTER

The claims are directed to communicating over multiple protocol interfaces in a computing environment. Spec. ¶ 2. Figure 3 reproduced below, illustrates an embodiment.

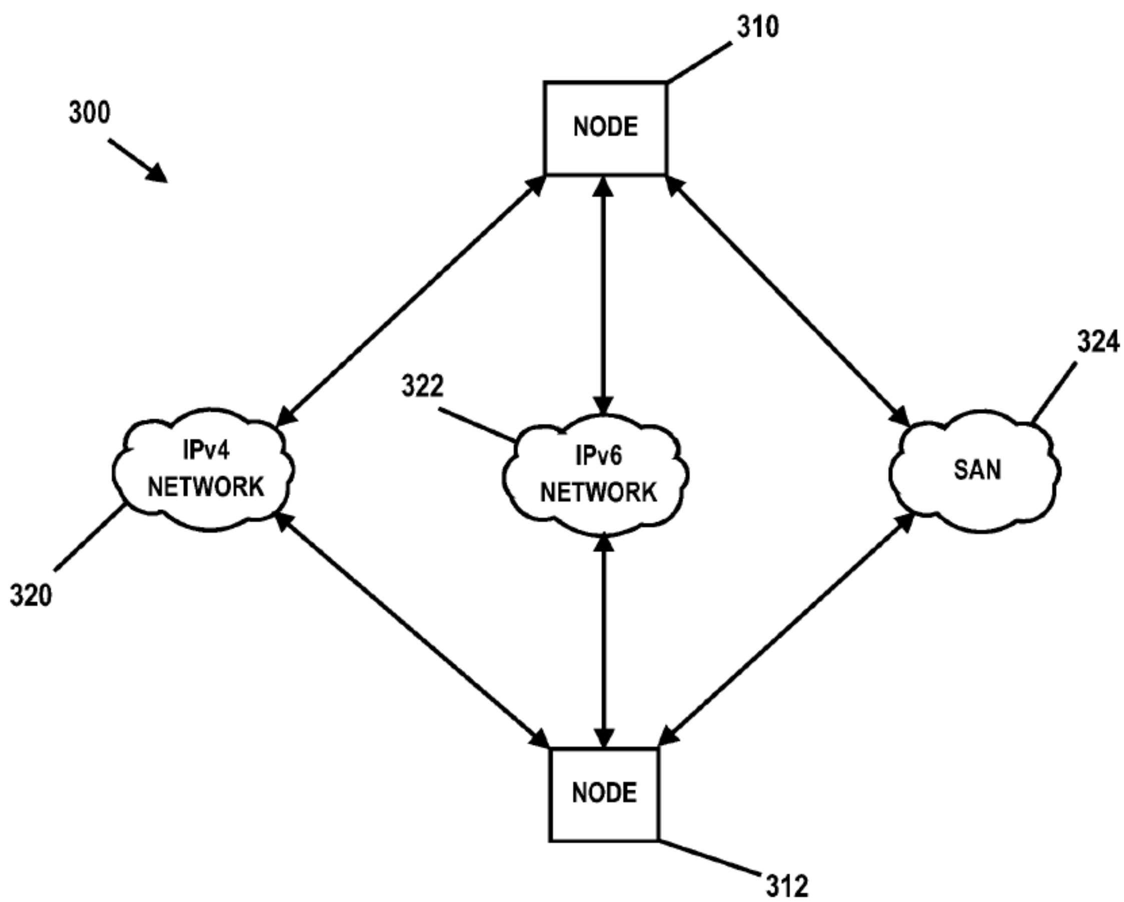

FIG. 3

Figure 3 is a diagram illustrating clustered network environment 300. *Id.* ¶ 34. Nodes 310 and 312 interface with each other via Internet Protocol version 4 (IPv4) network 320, Internet Protocol version 6 (IPv6) network 322, and storage area network (SAN) 324. *Id.*

A large data packet may be fragmented into suitable size packets or fragments to be communicated over a range of communication interfaces between nodes 310 and 312 and reassembled at the destination. *Id.* ¶ 35. According to the Specification, this redundancy ensures that packets can be transmitted even if one interface goes down. *Id.* Ordinarily, an IPv6 interface does not support the fragmentation techniques for IPv4 packets. *Id.* ¶ 37. In one example, information (e.g., IPv4 identifier and fragment offset fields) from IPv4 headers are placed into fragmentation fields of application layer headers of IPv6 packets. This enables the IPv4 packets to be transmitted over an IPv6 interface and then be re-assembled at the destination based on the IPv4 protocol. *Id.* ¶ 38. For re-assembly, interface logic is used for ingress communications that extracts the IPv4 identifier and fragment offset fields from the application layer headers of the IPv6 packets and re-assembles logical data packets according to the IPv4 fragmentation and re-assembly techniques. *Id.* ¶ 42.

Claim 1, reproduced below, is illustrative of the claimed subject matter:

> 1. A method, comprising:
>
> identify a first communications protocol of a network interface for ingress data packet communications received by the network interface;
>
> responsive to the identified first communications protocol of the network interface differing from a second communications protocol used for fragmenting the received ingress data packet communications, derive fragmentation data according to a data packet fragmentation protocol corresponding to the second communications protocol from an application layer header of the data packet communications; and

re-assemble a logical packet from the data packet communications using the data packet fragmentation protocol and the fragmentation data.

REFERENCES

The prior art relied upon by the Examiner is:

Name	Reference	Date
Niinomi	US 2005/0281287 A1	Dec. 22, 2005
Lyon	US 2013/0103853 A1	Apr. 25, 2013
Ganesh	9,356,863 B2	May 31, 2016

REJECTIONS

The Examiner rejects claims 1–18[2] under 35 U.S.C. § 103(a) as being unpatentable over Lyon and Niinomi. Final Act. 3–15.

The Examiner rejects claims 1–4, 6–9, and 11–14 for nonstatutory double patenting in light of Ganesh and Niinomi. *Id.* at 15–21.

The Examiner rejects claims 5, 10, and 15 for nonstatutory double patenting in light of Ganesh, Niinomi, and Lyon. *Id.* at 21–24.

OPINION

Obviousness Rejection of Claims 1–18

Overview of Lyon

Lyon is directed to detecting clients based on communication formats, e.g., by receiving a request from a client capable of communicating via a plurality of supported communication formats, selecting a format expected to give optimal performance, and directing the client to a server configured

[2] The Examiner states that claims 1–17 are rejected. Final Act. 4. Nevertheless, the Examiner finds claim 18 obvious in the body of the rejection. *Id.* at 15.

to communicate in the selected format. Lyon, Abstract. Figure 5, reproduced below, illustrates an example:

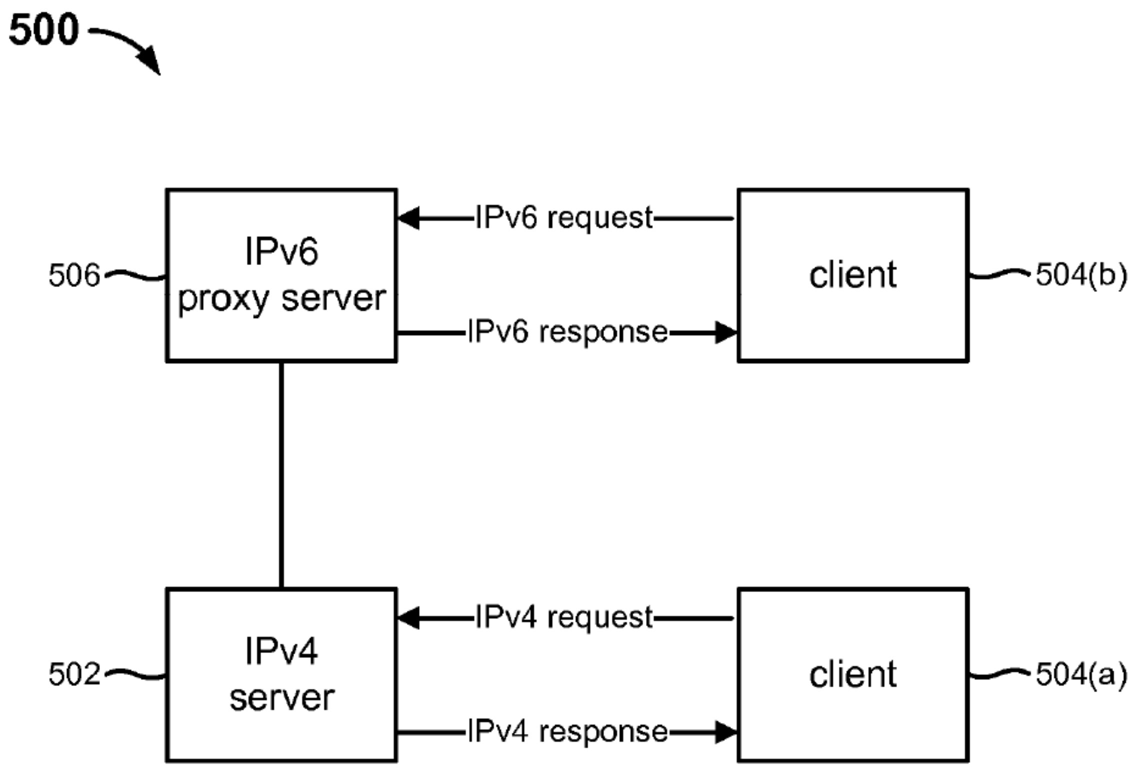

FIG. 5

Figure 5 is a block diagram of a network environment in which proxy server 506 translates content from a first format (e.g., IPv4) to a second format (e.g., IPv6). *Id.* ¶¶ 28–29, Fig. 6. IPv4 server 502 responds to an IPv4 request from client 504(a) with an IPv4 response. *Id.* ¶ 28. On its back end, proxy server 506 is configured to proxy content from IPv4 server 502 via an IPv4 network. *Id.* On its front end, proxy server 506 is connected to and configured to serve IPv6 content natively via an IPv6 network. *Id.* For example, proxy server 506 responds to IPv6 requests from client 504(b) with

IPv6 responses. *Id.* Proxy server 506 also can receive client requests for and serve IPv4 content. *Id.*

Overview of Niinomi

Niinomi relates to executing packet transfer communication using communication protocols such as Transmission Control Protocol/Internet Protocol (TCP/IP) to reduce loading in a host computer by the fragmentation and reassembly of transmitted and received packets and to reduce transmission delay time of the transmitted and received packets. Niinomi ¶¶ 2, 13.

In a transmission side communication controller, IP fragment packets are formed by the fragmentation of an IP packet, and an IP fragment header is added to each individual IP fragment packet. *Id.* ¶ 58. A network controller at the reception side analyzes the IP fragment headers, reassembles the IP fragment packets, generates a reassembly header, and passes the reassembly header and reassembled packet to a receiving host. *Id.* ¶ 59. In the process of generating the reassembly header, a higher layer header, such as a TCP header, is extracted from the first IP fragment packet and included in the reassembly header. *Id.* ¶ 104. The reception side communication controller can send a reassembly completion notifier to the transmission side communication controller. *Id.* ¶ 113.

Analysis

Regarding claim 1, the Examiner finds that Lyon's disclosure of proxy server 506 identifying requests for IPv6 content from client 504(b)[3]

---

[3] Although Lyon describes two clients, 504(a) and 504(b), the Examiner simply refers to client 504 in several instances. For each instance, we

teaches “identify a first communications protocol of a network interface for ingress data packet communications,” as recited in claim 1. Final Act. 4 (citing Lyon ¶ 28, Fig. 5). In particular, the Examiner finds that IPv6 is “a first communications protocol.” *Id.*

The Examiner further finds that Lyon’s description of proxy server 506 receiving from client 504(a), via IPv4 server 502, an IPv4 communication protocol request teaches:

> responsive to the identified first communications protocol of the network interface differing from a second communications protocol used for fragmenting the received ingress data packet communications, derive fragmentation data according to a data packet fragmentation protocol corresponding to the second communications protocol,

as recited in claim 1. *Id.* at 4–5 (citing Lyon ¶¶ 28–29, Figs. 4–5). Specifically, the Examiner finds that IPv4 is “a second communications protocol use for fragmenting the received ingress data packet communications.” *Id.* at 4. According to the Examiner, by obtaining data using a protocol that is compliant with packet fragmentation associated with the received IPv4 packet and sending a response packet in that format to client 504(a), proxy server 506 “derive[s] fragmentation data according to a data packet fragmentation protocol corresponding to the second communications protocol.” *Id.* at 4–5.

The Examiner concedes that Lyon does not teach identifying the first communications protocol of the network interface for ingress data packet communications “received by the network interface,” deriving fragmentation data according to the data packet fragmentation protocol corresponding to

---

identify the most logical client, 504(a) or 504(b), consistent with the Examiner’s findings and Lyon’s disclosure.

the second communications protocol "from an application layer header of the data packet communications," or "re-assemble a logical packet from the data packet communications using the data packet fragmentation protocol and the fragmentation data," as recited in claim 1. *Id.* at 5. The Examiner cites Niinomi for these teachings. *Id.* Specifically, the Examiner finds: that Niinomi's transceiver is a "network interface" that receives ingress data packet communications (*id.* (citing Niinomi ¶ 54, Figs. 1–2)); that Niinomi's description of including a higher layer (e.g., TCP header) in a reassembly header teaches deriving fragmentation data "from an application layer header of the data packet communications" (*id.* (citing Niinomi ¶ 104)); and that Niinomi's description of reassembling IP fragment packets teaches "re-assembl[ing] a logical packet from the data packet communications using the data packet fragmentation protocol and the fragmentation data," as recited in claim 1 (*id.* (citing Niinomi ¶¶ 58–59, Fig. 6A)).

The Examiner concludes that it would have been obvious to derive fragmentation data from an application layer header and re-assemble a logical packet in Lyon using the data packet fragmentation protocol and the fragmentation data "in order to improve efficiency of the system by performing re-assembly and fragmentation processing in a manner that [achieves] improvement of communication throughput by reduction of transmission delay time as taught by Niinomi." *Id.* at 6 (citing Niinomi ¶¶ 8, 10, 13).

The Examiner makes substantially the same findings for independent claims 6 and 11. *See id.* at 7–9 (claim 6), 10–13 (claim 11).

Appellant argues that two aspects of claim 1 are missing from Lyon and Niinomi. First, Appellant argues that Lyon does not teach "responsive to the identified first communications protocol of the network interface differing from a second communications protocol used for fragmenting the received ingress data packet communications, derive fragmentation data according to a data packet fragmentation protocol corresponding to the second communications protocol," as recited in claim 1. Appeal Br. 5–7. Second, Appellant argues that Niinomi does not teach that any deriving of fragmentation data is "from an application layer header of the data packet communications," as recited in claim 1. *Id.* at 7–8. Appellant incorporates these arguments for independent claims 6 and 11. *Id.* at 8–9.

As to the first argument, Appellant contends that Lyon's proxy server 506 receives IPv4 content from server 502 over an IPv4 network on its back end and IPv6 content from client 504(b) over an IPv6 network on its front end. *Id.* at 5–6. Appellant argues that to send over the IPv6 network IPv4 packets it received on its back end, proxy server 506 translates the IPv4 content to IPv6 content and then transmits the IPv6 content (not IPv4 content) over the IPv6 network. *Id.* at 6. According to Appellant, "in Lyon, neither the frontend nor the backend of the server 506 receives ingress data packets in a fragmentation protocol that is different than the network interface protocol receiving those ingress data packets." *Id.* at 6.

Specifically, Appellant argues that "in Lyon, the IPv6 proxy server 506 does not appear to receive IPv6 content over an IPv4 network or on an IPv4 network interface, nor does it appear to receive IPv4 content over an IPv6 network or on an IPv6 network interface." *Id.* "In other words, for the server 506 in Lyon, ingress communications received on its frontend were not fragmented according to a different protocol than the frontend receiving interface, and ingress communications received on its backend were not fragmented according to a different protocol than the backend receiving interface." *Id.* Appellant reiterates these arguments in the Reply Brief, at 3–4.

In response, the Examiner asserts that "Lyon . . . discloses . . . receiving, from client 504, IPv4 communication protocol requests (in par. [0028], Fig.4, Fig. 5), wherein server 506 is connected to and configured to serve IPv6 content natively via an IPv6 network (in par. [0028], Fig. 5)" and from this concludes that "the requests disclosed in Lyon, which identify a protocol to be used for communication different than the network interface, read on the broadest reasonable interpretation of the language as claimed." Ans. 3.

We agree with Appellant. Although Lyon teaches proxy server 506 receiving ingress IPv4 packets over an IPv4 network from IPv4 server 502 and ingress IPv6 packets over an IPv6 network from client 504(b), Lyon does not teach receiving ingress data packet communications at an interface of a first communications protocol (e.g., IPv6) and deriving fragmentation data from those communications according to a different communications protocol (e.g., IPv4). Lyon does not describe the derivation of fragmentation data. However, because the protocols of the data packets received by proxy server 506 match that of the interface over which they are received (IPv4 packets received at the backend over an IPv4 network and IPv6 packets received at the frontend over an IPv6 network), the derivation of fragmentation data presumably is according to the same protocol of the network interface in each case. Thus, the Examiner has not shown that Lyon teaches "responsive to the identified first communications protocol of the network interface differing from a second communications protocol used for fragmenting the received ingress data packet communications, derive fragmentation data according to a data packet fragmentation protocol corresponding to the second communications protocol," as recited in claim 1.

As to Appellant's second argument, Appellant contends that Niinomi describes IP packet fragments having IP fragment headers. Appeal Br. 8 (citing Niinomi ¶¶ 58, 63). Appellant argues that such disclosure of a network layer header is insufficient to show fragmentation data derived from an application layer header, as recited in claim 1. *Id.* The Examiner responds that Niinomi's disclosure is not limited to extracting fragmentation data from an IP header and, instead, includes "extracting fragmentation data from the higher layer header" discussed in paragraph 104 on Niinomi. Ans. 5–6 (citing Niinomi ¶¶ 104, 113, Fig. 12). In reply, Appellant argues that "although Niinomi appears to indicate that data extracted may include the TCP header and some 'other' undefined data in Niinomi, Niinomi does not disclose the use of, or make any reference to, an 'application layer header' of the packet. To the contrary, Niinomi appears to be referring to the transport layer of the packet." Reply Br. 5.

We agree with Appellant. Niinomi's discussion of fragmenting and reassembling IP packets (Niinomi ¶¶ 58–59) identifies IP and TCP headers, which are network layer and transport layer headers, respectively. Further, Niinomi's discussion of a "higher layer header" is in reference to "TCP header 52 in this case." *Id.* ¶ 104. The Examiner does not identify any examples in Niinomi of deriving information from an application layer header or explain persuasively why Niinomi's description should be viewed as encompassing such derivation. Thus, the Examiner has not shown that Niinomi teaches "deriving fragmentation data . . . from an application layer header of the data packet communications," as recited in claim 1.

In sum, the Examiner has not shown that claim 1 would have been obvious over Lyon and Niinomi. For the same reasons, the Examiner has not shown that independent claims 6 and 11 would have been obvious over Lyon and Niinomi. Claims 2–5 and 16 depend from claim 1; claims 7–10 and 17 depend from claim 6; and claims 12–15 and 18 depend from claim 11. The dependent claims similarly distinguish over Lyon and Niinomi. Accordingly, the Examiner's rejection of claims 1–18 as obvious over Lyon and Niinomi is reversed.

Nonstatutory Double Patenting Rejection of Claims 1–4, 6–9, and 11–14

The Examiner rejects claims 1–4, 6–9, and 11–14 under the doctrine of nonstatutory obviousness-type double patenting over claims 1, 3, 7, and 9 of Ganesh in view of Niinomi. Final Act. 17–21. The Bibliographic Data Sheet lists the current application as a continuation of the application that matured into Ganesh.

Claim 1 of Ganesh, for example, recites generating an application layer header for data packet communications, fragmenting the data packet communications according to a data packet fragmentation protocol, and transmitting the data packet communications, including the application layer header, over a network interface. The Examiner concedes, however, that Ganesh "does not explicitly disclose deriving fragmentation data from an application layer header of the data packet communications; and re-assemble a logical packet from the data packet communications using the data packet fragmentation protocol and the fragmentation data." Final Act. 17. As the Examiner did in the obviousness rejection over Lyon and Niinomi, the Examiner finds that Niinomi discloses these limitations and that a skilled artisan would have combined the claims of Ganesh with the teachings of

Niinomi for the same reasons discussed above for the combination of Lyon and Niinomi. *Id.* at 17–18.

In response, Appellant reiterates the argument, presented for the combination of Lyon and Niinomi, that Niinomi does not teach "deriving fragmentation data . . . from an application layer header of the data packet communications," as recited in claim 1. Appeal Br. 13; Reply Br. 10–11. In the Answer, the Examiner repeats the findings, presented for the combination of Lyon and Niinomi, that Niinomi's teachings are not limited to extracting fragmentation data from the IP header and, instead, include deriving fragmentation data from an application layer header. Ans. 9. For the same reasons given above for the combination of Lyon and Niinomi, we agree with Appellant that Niinomi does not teach deriving fragmentation data from an application layer header. Accordingly, the Examiner has not shown that claim 1 would have been obvious over claims 1, 3, 7, and 9 of Ganesh in view of Niinomi. For the same reasons, the Examiner has not shown that claims 2–4, 6–9, and 11–14 would have been obvious over claims 1, 3, 7, and 9 of Ganesh in view of Niinomi.

Accordingly, the Examiner's nonstatutory obviousness-type double patenting rejection of claims 1–4, 6–9, and 11–14 is reversed.

Nonstatutory Double Patenting Rejection of Claims 5, 10, and 15

The Examiner rejects claims 5, 10, and 15 under the doctrine of nonstatutory obviousness-type double patenting over claims 1, 3, 7, and 9 of Ganesh in view of Niinomi and Lyon. Final Act. 21–24. Claims 5, 10, and 15 depend from claims 1, 6, and 11, respectively. The Examiner incorporates the nonstatutory obviousness-type double patenting rejection of claims 1, 6, and 11 (detailed above), and cites Lyon for the additional limitations of claims 5, 10, and 15. *Id.* at 22–24. Appellant repeats the argument that Niinomi does not teach the aspects of claims 1, 6, and 11 that the Examiner concedes are missing from the claims of Ganesh. Appeal Br. 12–13; Reply Br. 11–13.

For the reasons given above, the Examiner has not shown that claims 1, 6, and 11 would have been obvious over Ganesh's claims in view of Niinomi. The Examiner does not find that Lyon remedies these deficiencies. Thus, the Examiner has not shown that claims 5, 10, and 15 would have been obvious over claims 1, 3, 7, and 9 of Ganesh in view of Niinomi and Lyon.

Accordingly, the Examiner's nonstatutory obviousness-type double patenting rejection of claims 5, 10, and 15 is reversed.

CONCLUSION

The Examiner's rejections are reversed.

More specifically,

The Examiner's rejection of claims 1–18 as obvious over Lyon and Niinomi is reversed.

The Examiner's nonstatutory obviousness-type double patenting rejection of claims 1–4, 6–9, and 11–14 is reversed.

The Examiner's nonstatutory obviousness-type double patenting rejection of claims 5, 10, and 15 is reversed.

DECISION SUMMARY

In summary:

Claims Rejected	**35 U.S.C. §**	**Basis**	**Affirmed**	**Reversed**
1–18	103(a)	Obviousness		1–18
1–4, 6–9, 11–14		Nonstatutory double patenting		1–4, 6–9, 11–14
5, 10, 15		Nonstatutory double patenting		5, 10, 15
Overall Outcome				1–18

REVERSED