

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

SAMSUNG ELECTRONICS CO., LTD.,
Petitioner,

v.

HEADWATER RESEARCH LLC,
Patent Owner.

IPR2024-01407
Patent 9,179,359

PATENT OWNER'S RESPONSE

TABLE OF CONTENTS

I.	Introduction	1
II.	Overview of the '359 Patent	1
III.	Level of Ordinary Skill in the Art.....	3
IV.	Claim Construction	4
V.	The Petition Fails to Show Unpatentability of Claim 1.	4
	A. The Petition fails to show obviousness of denying or blocking piggy back network connection requests.	5
	B. The Petition fails to show that Shell-Cole would indicate unavailability of the alleged Internet data service to an application that is “currently available” to another application.	12
VI.	The Petition additionally fails to show unpatentability of claim 26.	14
VII.	Conclusion	20

PATENT OWNER’S EXHIBIT LIST

No.	Description
Ex. 2001	<i>Headwater Research LLC v. Samsung Elecs. Co.</i> , No. 2:23-cv-641-JRG-RSP, Dkt. No. 44, Docket Control Order (E.D. Tex. June 20, 2024)
Ex. 2002	<i>Headwater Research LLC v. Samsung Elecs. Co.</i> , No. 2:23-cv-641-JRG-RSP, Dkt. No. 56, Order Granting Motion to Amend Docket Control Order (E.D. Tex. Dec. 23, 2024)
Ex. 2003	Eastern District of Texas Case Management Statistics
Ex. 2004	<i>Headwater Research LLC v. Samsung Elecs. Co.</i> , No. 2:23-cv-641-JRG-RSP, Dkt. No. 21, Unopposed Motion for Extension of Time to Move, Answer or Otherwise Respond to Plaintiff’s Complaint and Waiver of Foreign Service Requirement (E.D. Tex. Jan. 8, 2024)
Ex. 2005	<i>Headwater Research LLC v. Samsung Elecs. Co.</i> , No. 2:23-cv-641-JRG-RSP, Dkt. No. 22, Order Granting Extension of Time to Respond to Complaint (E.D. Tex. Jan. 15, 2024)
Ex. 2006	<i>Headwater Research LLC v. Samsung Elecs. Co.</i> , No. 2:23-cv-641-JRG-RSP, Plaintiff Headwater Research LLC’s Amended Disclosure of Asserted Claims and Infringement Contentions (E.D. Tex. Aug. 20, 2024)
Ex. 2007	<i>Headwater Research LLC v. Samsung Elecs. Co.</i> , No. 2:23-cv-641-JRG-RSP, Samsung Electronics Co. Ltd. and Samsung Electronics America, Inc.’s P.R. 3-3 and 3-4 Invalidity and Patent Ineligibility Contentions (E.D. Tex. July 17, 2024)
Ex. 2008	U.S. Patent Application Publication No. 2008/0080457 by Cole (“Cole-2”)
Ex. 2009	Redline comparison of Cole (Ex. 1006) and Cole-2 (Ex. 2008)
Ex. 2010	Declaration of Erik de la Iglesia in support of Patent Owner’s Preliminary Response
Ex. 2011	<i>Curriculum vitae</i> of Erik de la Iglesia
Ex. 2012	Declaration of Dr. Michael C. Brogioli in support of Patent Owner’s Response
Ex. 2013	<i>Curriculum vitae</i> of Dr. Michael C. Brogioli
Ex. 2014	June 12, 2025 deposition of Dr. Kevin R. B. Butler

I. Introduction

Claim 1 of the '359 patent requires processors configured to “a network access condition that indicates the unavailability to the particular application of an Internet data service that is currently available via the WWAN modem to a different application.” In other words, claim 1 requires that the same “Internet data service” be both “currently available” to the “different application” and indicates as unavailable “to the particular application.” The Petition alleges that this is satisfied based on denying “piggy back” connection requests, which are requests that relate to an application that would like to access a network while another application already has an open connection to the network). However, the Petition identifies no teaching in the prior art that it would *ever* deny a “piggy back” connection request, and the Petition identifies no motivation to modify the prior art to do so.

Regarding claim 26, which requires that an active application determination be based on user interaction with the application. In alleging the obviousness of this claim, the Petition mixes and matches two distinct embodiments of the “Shell” prior art reference. However, the Petition identifies no motivation (or any other obviousness rationale) supporting this proposed modification to the prior art, and thus cannot establish obviousness.

II. Overview of the '359 Patent

Claim 26 of the '359 patent is directed at a high level to an end-user device

that indicates unavailability of WWAN access to one application while WWAN access is available to a different application, wherein the “different application” has been determined by the system to be an “active application based on a user interaction with the different application.” Claims 1 and 26 are reproduced below.

1. A wireless end-user device, comprising:

- [1.1] a wireless wide area network (WWAN) modem to receive and transmit Internet data between the device and at least one WWAN when configured for and connected to the at least one WWAN;
- [1.2] a wireless local area network (WLAN) modem to receive and transmit Internet data between the device and at least one WLAN when configured for and connected to the at least one WLAN; and
- [1.3] one or more processors configured to apply a stored WWAN differential traffic control policy to Internet data service provided using the WWAN modem and the at least one WWAN, and
- [1.4] indicate to a particular application, via an application program interface (API), one or more network access conditions based on the applied WWAN differential traffic control policy,
- [1.5] wherein the one or more network access conditions includes a network access condition that indicates the unavailability to the particular application of an Internet data service that is currently available via the

WWAN modem to a different application, but that is indicated as unavailable to the particular application.

26. The wireless end-user device of claim 1, *wherein the one or more processors are further configured to determine that the different application is an active application **based on a user interaction with the different application.***

Ex. 1001, claims 1, 26.

As shown above, Claim 26 requires that the system be configured to “determine that the different application” (i.e., the application for which the Internet data service *is* currently available to) “is an active application based on a user interaction with the different application.” Claim 26 thus elaborates on the application of claim 1’s policy, because the “different application” is the application to which “Internet data service... is currently available via the WWAN modem,” while the system “indicates the unavailability” of the same Internet data service to “a particular application” (distinct from the claimed “different application” determined to be “an active application”). In other words, the active application determination impacts how claim 1’s “differential traffic control policy” is applied, including affecting the application to which “Internet data service... is currently available.” Ex. 2012, ¶¶33-34.

III. Level of Ordinary Skill in the Art

The Petition proposes that a POSITA in the field of the ’359 patent “would

have had (1) at least a bachelor's degree in computer science, computer engineering, electrical engineering, or a related field, and (2) at least two years of industry experience in services and application implementation in communication networks. Additional graduate education could substitute for professional experience, and vice versa." *See* Ex. 1003 ¶15; Pet. at 13 & n.2 (citing Ex. 1003 ¶¶14-15). For purposes of this Response, Patent Owner does not challenge that definition.

IV. Claim Construction

The Petition states that "[n]o formal claim constructions are necessary." Pet. at 2. For purposes of this Response, Patent Owner agrees.

V. The Petition Fails to Show Unpatentability of Claim 1.

The Petition alleges that limitation [1.5] is satisfied by the combined Shell-Cole system "denying/blocking [a] 'piggy back' network connection request." *See* Pet. 32-33.

However, as explained below, Shell and Cole do not disclose (or render obvious) denying a piggy back network connection request. Ex. 2012 ¶¶35-48. And as also explained below, even if such an implementation *were* disclosed or obvious in view of Shell and Cole, the Petition fails to show how any network connection response indicates unavailability to the particular application of the *same* Internet data service that is currently available to the "different application." Ex. 2012, ¶¶49-52.

A. The Petition fails to show obviousness of denying or blocking piggy back network connection requests.

The Petition’s analysis regarding how denying or blocking a “piggy back” request satisfies the relevant limitations is reproduced in its entirety below:

Additionally, in Shell, an application may “wish[] to ‘piggy back’ on other connection requests[,]” i.e., “an application may request to only access resources in a destination network when another application is already accessing resources.” SAMSUNG-1005, [0070]. A “piggy back” connection request may be denied in some instances, e.g., due to limited network bandwidth. *Id.*, [0092]. In these instances, a first application (*the particular application*) sends a “piggy back” network connection request based on an established connection of a second application (*different application*). SAMSUNG-1003, ¶104.

In denying/blocking the “piggy back” network connection request, the connection manager modules would provide notification data to the first application indicating decision information regarding the “piggy back” network connection request (*network access condition*). SAMSUNG-1003, ¶105. Further, the decision information would indicate that the “piggy back” network connection request is denied/blocked (*Internet data service being indicated as unavailable to the particular application*), but that network access by the first application is available (*Internet data service [] is currently available via the WWAN modem to a different application*). *Id.*

Pet. 32-33 (emphases in original). The Petition has no theory regarding how the requirements of limitation [1.5] are satisfied *other* than by denying or blocking a

“piggy back” connection request. For instance, the Petition presents no theory for how claim 1’s requirements relating to the “different application” (including that the Internet data service be available to that “different application”) are met, other than with respect to the piggy back request. *See generally* Pet. 32-33.

Shell teaches that its “piggy back” request occurs when an “application... notif[ies] connection planner 211 it wishes to ‘piggy back’ on other connection requests”

In other embodiments, *an application may notify connection plan[n]er 211 it wishes to “piggy back” on other connection requests. That is, an application may request to only access resources in a destination network when another application is already accessing resources.* For example, sync engine 202 may wish to synchronize an electronic mail account only when a connection to the Internet has been established for another application. *Applications that wish to piggy back, may call APIs associated with connection manager modules 210 and be given an open handle. The open handle facilitates notification back to the application that a connection on which the application may want to piggy back has been established.*

Ex. 1005, [0070] (emphasis added). A POSITA would thus understand that a piggy back connection request is simply a request to open a handle, which will notify the application at a later time, whenever a connection suitable to piggyback upon—i.e.,

“a connection on which the application may want to piggy back”—has been established. Ex. 2012, ¶¶36-37.

The understanding of a piggy back connection request as a request to open an API handle is consistent with Dr. Butler’s testimony at deposition, in which “a piggyback request may call APIs to open the handle” such that the application can be informed in the future when a suitable connection is established. Ex. 2014 at 11:14-12:6 (“[I]n the context of Shell’s Paragraph 70, is the API call that is made as part of a piggyback connection request the same API call that results in the creation of an open handle? A.... [C]ertainly, there is a way in which a piggyback request may call APIs to open the handle.”).

In other words, an application’s piggyback connection request is an application’s request (sent to connection planner 211 via an API) to open a handle, such that “[t]he open handle facilitates notification back to the application that a connection on which the application may want to piggy back has been established.” Ex. 1005, [0070]. Ex. 2012, ¶38.

Importantly, a POSITA would have recognized that these piggyback connection requests have several advantages. Ex. 2012, ¶39. For example, by allowing applications to connect to the network only when another application is accessing the network, significant power savings can be achieved by reducing the number of times the network modem switches states (e.g., active vs inactive, or

transmitting vs not transmitting). Additionally, because an open handle generated as a result of a piggy back connection request informs the application when a suitable connection is available for the application to piggyback upon, the application can know to initiate a network connection only when conditions are suitable for doing so. Thus, piggy backing prevents scenarios where the application might attempt to initiate a connection under less-than-ideal circumstances, such as when no other connection exists (which may be inefficient by causing additional power state changes in the modem), or when no connection can be established.

In light of these advantages, it would not have been obvious to ***block or deny*** an application's piggyback connection request (i.e., a request to open a handle so that the application can be informed when a connection meeting the application's needs ultimately becomes available). Ex. 2012, ¶40.

The Petition supposes that a piggyback connection request may be blocked or denied “due to limited network bandwidth.” Pet. 32. But “limited network bandwidth” is a state in which the *current* network bandwidth is not suitable. Ex. 2012, ¶41. And a *current* lack of bandwidth is not a reason to deny an application's request to open an API handle so that it can be informed *in the future* when network bandwidth *is* suitable for the application to access the network in the manner requested. *Id.* The point of a piggyback connection request is not to ***immediately*** establish a connection, but rather to request an open handle so that the application

can be informed whenever conditions are ideal for establishing a piggyback connection. *Id.*

Indeed, a POSITA would be strongly motivated to ***not*** deny or block piggyback connection requests, because a POSITA would understand that, by connecting to the network only when another application is accessing the network, significant power savings can be achieved by reducing the number of times the network modem switches states (e.g., active vs inactive, or transmitting vs not transmitting). Ex. 2012, ¶42. A POSITA would understand that if a piggyback connection request (which opens a handle informing the application when ideal conditions for a piggyback connection arise) is denied, the application would be prevented from receiving an open handle such that it can learn when ideal connection conditions exist for establishing a piggyback connection. Without such an open handle, the application would attempt network access ***regardless*** of preexisting connections, which would unnecessarily increase power consumption of the device.

A POSITA would understand that such indiscriminate network access attempts would frequently cause connections to the network even when no other connections existed, thus frequently switching a modem from an inactive state to an active state for the purpose of the new connection and causing significantly more battery drain than the application “piggybacking” on a preexisting connection. Ex. 2012, ¶¶39, 43. Accordingly, a POSITA would avoid blocking or denying piggyback

connection requests, as doing so would lead to significantly more power inefficiency than creating an open handle to inform the application of when a suitable connection is established. Ex. 2012, ¶43.

Moreover, to the extent the Petition contends that a “piggy back” connection request would be a request for a connection *after* receiving information via the open handle that a suitable connection is available, the Petition does not explain why it would be obvious for Shell’s system to deny such a connection request while another application retained Internet access. Ex. 2012, ¶¶44-47.

The Petition’s only support for the proposition that a “‘piggy back’ connection request may be denied in some instances” is a citation to paragraph [0092] of Shell (*see* Pet. 32), but paragraph [0092] does not disclose denying a *piggy back* connection request to the extent such a request is responsive to an open handle notification from Shell’s connection manager modules 210 that a suitable connection is available. Ex. 2012, ¶44. Because Shell does not disclose denying “piggy back” connection requests, the Petition must establish the obviousness of doing so. The Petition, however, does not make such an allegation, and thus fails to set forth any plausible theory of obviousness. *See generally* Pet. 32-33.

Moreover, a POSITA would *not* have found it obvious to deny an application’s connection request made subsequent to Shell’s notification that a suitable connection would be available to the application (to the extent that is what

the Petition intends to argue). Ex. 2012, ¶¶45-47. Specifically, a POSITA would understand that, in the context of Shell's piggy back teachings, that the connection manager modules 210 would only notify the application that a connection is available *if and when* a connection would be allowed if requested. *Id.* ¶45. For example, there would be no reason for Shell's connection manager modules to inform an application via an open handle that it would be a good time for the application to initiate a connection, only for the connection manager modules to immediately thereafter deny a resulting connection request. *Id.* This would lead to unnecessary and inefficient application behavior (e.g., waking an inactive application, causing that application to initiate a futile connection request, and have the connection manager modules 210 process unnecessary connection requests that could have been avoided if the appropriate open handle behavior had been implemented), and the Petition does not allege that its proposed combination would operate in such a manner (much less provide a motivation to do so).

The fact that a POSITA would ***not*** find it obvious for the Shell-Cole combination to inform potentially piggybacking applications of a network connection upon which that the application ***would not*** be able to piggyback is confirmed by Shell itself. *See* Ex. 2012, ¶46. Specifically, Shell teaches that an "open handle facilitates notification back to the application that *a connection on which the application may want to piggy back has been established.*" Ex. 1005, [0070]. Of

course, if a connection initiated by a different application is on a network where insufficient bandwidth exists to handle both the different application's connection and the potentially piggybacking application's connection, then that would not be a "connection on which the application may want to piggy back," because no application would want to piggy back on a connection on which it would be unable to piggy back. As such, it would only be obvious for applications to be informed that a connection is possible via the open handle when a connection would actually be *allowed*. It certainly would not be obvious to *deny* or *block* any connection attempts initiated as a result of receiving a piggyback open handle notification, because receiving a notification via an open handle that a piggyback connection is possible would indicate that responsive connection attempts will be *allowed*. See Ex. 2012, ¶¶46-47.

B. The Petition fails to show that Shell-Cole would indicate unavailability of the alleged Internet data service to an application that is "currently available" to another application.

Claim 1 requires "one or more processors configured to... indicate to a particular application, via an [API], one or more network access conditions..., wherein the one or more network access conditions includes a network access condition that indicates the unavailability to the particular application of *an Internet data service that is currently available via the WWAN modem to a different application, but that is indicated as unavailable to the particular application.*" In

other words, the **same** “Internet data service that is currently available via the WWAN modem to a different application” must be “indicated as unavailable to the particular application.” Ex. 2012, ¶48.

The Petition does not identify a consistent “Internet data service... that is indicated as unavailable to the particular application,” which Internet data service is “currently available via the WWAN modem to a different application.” Instead, the Petition presumes that a “*‘piggy back’ network connection request*” would be “*denied/blocked* [to the particular application]..., but that *network access by the [different] application is available*.” Pet. 32-33. In other words, the Petition improperly identifies two **different** Internet data services for its claim mapping: (1) a piggyback network access request made by the “particular application,” and (2) general “network access” by the “different application.” Ex. 2012, ¶49. Accordingly, the Petition fails to allege that the Shell-Cole combination “indicates the unavailability to the particular application ***of an Internet data service that is currently available via the WWAN modem to a different application***,” because the Petition points to two **different** Internet data services as being (1) “currently available” to the different application and (2) indicated as unavailable to the particular application.

The Petition does not allege, for example, that the combined Shell-Cole system indicates unavailability of ***all*** WWAN network access to the “particular

application” that issued a piggyback connection request. Nor does the Petition allege that the “different application” would have access to the same “‘piggy back’ network access request” made by the “particular application” whose piggyback request was denied or blocked. Accordingly, the Petition does not set forth any plausible theory how the *same* Internet data service is both (1) available to the different application, and (2) indicated as unavailable to the particular application.

VI. The Petition additionally fails to show unpatentability of claim 26.

As summarized above in Section II, claim 1 recites:

1. A wireless end-user device, comprising:

- [1.1] a wireless wide area network (WWAN) modem to receive and transmit Internet data between the device and at least one WWAN when configured for and connected to the at least one WWAN;
- [1.2] a wireless local area network (WLAN) modem to receive and transmit Internet data between the device and at least one WLAN when configured for and connected to the at least one WLAN; and
- [1.3] one or more processors configured to apply a stored WWAN differential traffic control policy to Internet data service provided using the WWAN modem and the at least one WWAN, and
- [1.4] indicate to a particular application, via an application program interface (API), one or more network access

conditions based on the applied WWAN differential traffic control policy,

[1.5] wherein the one or more network access conditions includes a network access condition that indicates the unavailability to the particular application of an Internet data service that is currently available via the WWAN modem to a different application, but that is indicated as unavailable to the particular application.

Ex. 1001, Claim 1. In other words, claim 1 specifies that there is a “WWAN differential traffic control policy” that is applied “to Internet data service provided using the WWAN modem.” Claim 1 further specifies that the policy applies differently to two different applications. To a first application (the “particular application”), an Internet data service is indicated as unavailable, and to a second application (the “different application”) the same Internet data service is made “currently available.”

Claim 26 further expands on the operation of claim 1, in specifying that the “different application” (i.e., the application to which Internet data service is made available via the applied policy) was determined to be an “active application” “based on a user interaction” with that application.

26. The wireless end-user device of claim 1, *wherein the one or more processors are further configured to determine that the different*

*application is an active application **based on a user interaction with the different application.***

Ex. 1001, Claim 26. Taking claims 1 and 26 together, a “user interaction with the different application” is used as an input to make an active application determination, thereby influencing the policy such that “the different application” is provided access to the claimed Internet data service. Ex. 2012, ¶¶52-53.

The Petition contends that this requirement is satisfied because, “[i]n Shell, connection manager modules may assign priorities to network connection requests ‘based on the current state of applications associated with the connection requests.’” Pet. 47 (quoting Ex. 1005, [0057], and citing Ex. 1005, [0092]). The quoted paragraph of Shell, paragraph [0057], teaches that an example of prioritizing connection requests “based on the current state of applications” includes prioritizing “a request from an application that is currently active”:

Priorities may also be assigned to connection requests based on the current state of applications associated with the connection requests. For example, *a request from an application that is **currently active** may be assigned a higher priority than a request from an application running in the background.*

Ex. 1005, [0057].

What is lacking from the above teaching of Shell, however, is any disclosure (as required by claim 26) that a determination that the different application “is an

active application” would be “*based on a user interaction* with the different application.” *See* Ex. 2012, ¶¶54-55.

In an attempt to cure this deficiency of Shell’s teaching regarding a “currently active” application, the Petition points to Shell’s teaching that “voice requests may be assigned a higher priority than data requests,” and on that basis asserts in conclusory fashion that “[a]n application that provides a voice-related connection request... involves a determination that a user is likely to interact with the application.” Pet. 47 (quoting Ex. 1005, [0057]). The Petition sets forth no other theory as to how an active application determination for the alleged “different application” would be based on “user interaction with the different application.” *See generally* Pet. 47.

But what the Petition fails to mention is that assigning “voice requests” a higher priority than “data requests” is described as an *alternative* prioritization algorithm to assigning priorities based on “the current state of applications” (an example of which is assigning priorities to applications that are “currently active”). Specifically, Shell teaches three different ways to prioritize connection requests (“based on applications associated with the connection requests,” “based on the current state of applications associated with the connection requests,” and “based on characteristics associated with the connection requests”):

Illustrated in FIG. 5A are connection requests 500 including new connection request 510 and active connection requests 520 and 530. Connection requests 510, 520, and 530 are prioritized, with connection 510 having the highest priority and connection 530 having the lowest priority. *Priorities may be assigned to connection **requests based on applications** associated with the connection requests.* For example, requests from sync engine 202 may be assigned a higher priority than requests from messaging application 206. *Priorities may also be assigned to connection requests **based on the current state of applications** associated with the connection requests.* For example, a request from an application that is currently active may be assigned a higher priority than a request from an application running in the background. *Priorities may also be assigned to connection requests **based on characteristics** associated with the connection requests.* For example, voice requests may be assigned a higher priority than data requests. It should be understood that these are only examples of assigned priorities. It would be apparent to one skilled in the art, after having reviewed this description, that priorities may be assigned to connection requests using a variety of methods.

Ex. 1005, [0057]; Ex. 2012, ¶56.

Dr. Butler agreed that these three concepts (prioritizing connection requests (1) “based on applications associated with the connection requests,” (2) “based on the current state of applications associated with the connection requests,” and (3) “based on characteristics associated with the connection requests” are three different

techniques described by Shell (Ex. 2014 at 16:17-18:13), and a POSITA would have understood them as being distinct approaches for prioritization. Ex. 2012 at ¶56.

Furthermore, while a POSITA would understand that a system could implement multiple of these different prioritization schemes, Shell does not disclose, for example, that “characteristics associated with the connection requests” would be used to determine “the current state of applications.” Ex. 2012, ¶56. But the Petition assumes exactly that, in asserting that a determination of voice-related connection request (i.e., a type of assigning priority “based on characteristics associated with the connection requests”) would be used to determine whether “an application... is currently active” (i.e., a type of assigning priority “based on the current state of applications”).

The Petition does not set forth any alleged *motivation* (or other obviousness rationale) to combine Shell’s different techniques in this manner, and in fact does not even acknowledge that it *is* combining different techniques within Shell. As such, the Petition fails to set forth any obviousness rationale to implement Shell as alleged, such that a specific type of connection request (i.e., a voice-related

connection request) would be used to determine whether the requesting application “is an active application” as recited by claim 26. *See* Ex. 2012, ¶57.¹

VII. Conclusion

For each of the foregoing reasons, Patent Owner respectfully requests that Claim 26 (the sole non-disclaimed claim of the ’359 patent) be found not unpatentable.

¹ As Patent Owner has previously argued in its Preliminary Response, the theory articulated in the Petition does not explain how an active application determination is “based on a user interaction with the different application” as claimed. Instead, the Petition’s theory relies on the system making an “active application” determination based on an inference that user interaction was “likely” at some point in the future. *See* Pet. 47 (“An application that provides a voice-related connection request involves a determination that a user is likely to interact with the application....”). However, a determination of what the user might do in the future is not “a user interaction,” and thus the Petition’s theory fails for this additional reason. Patent Owner recognizes that the Board has already interpreted the Petition as presenting an inherency theory, where the “user *has to* interact with that application to attempt that voice call” (Paper 9 at 27), and thus touches on this issue only briefly and respectfully submit that no such inherency theory was actually presented in the Petition, and that invalidating Claim 26 on a theory never presented in the Petition would be improper.

Date: June 27, 2025

Respectfully submitted,

/James A. Milkey/

James A. Milkey, Reg. No. 79,503

Reza Mirzaie, Reg. No. 69,138

Marc A. Fenster, *pro hac vice* forthcoming

Dale Chang, Reg. No. 61,476

Neil A. Rubin, Reg. No. 67,030

Philip X. Wang, Reg. No. 74,621

RUSS, AUGUST & KABAT

12424 Wilshire Blvd., 12th Floor

Los Angeles, CA 90025

Counsel for Patent Owner

CERTIFICATION REGARDING WORD COUNT

Pursuant to 37 C.F.R. §42.24(d), Patent Owner certifies that there are 4,483 words in the paper excluding the portions exempted under 37 C.F.R. §42.24(a)(1).

Date: June 27, 2025

/James A. Milkey/

James A. Milkey, Reg. No. 79,503

Reza Mirzaie, Reg. No. 69,138

Marc A. Fenster, *pro hac vice* forthcoming

Dale Chang, Reg. No. 61,476

Neil A. Rubin, Reg. No. 67,030

Philip X. Wang, Reg. No. 74,621

RUSS, AUGUST & KABAT

12424 Wilshire Blvd., 12th Floor

Los Angeles, CA 90025

Counsel for Patent Owner

CERTIFICATE OF SERVICE (37 C.F.R. § 42.6(e)(1))

The undersigned hereby certifies that the above document was served on June 27, 2025, by filing this document through the Patent Trial and Appeal Case Tracking System as well as delivering a copy via electronic mail upon the following attorneys of record for the Petitioner:

W. Karl Renner (Reg. No. 41,265)
Jeremy J. Monaldo (Reg. No. 58,680)
Jennifer J. Huang (Reg. No. 64,297)
Ryan Chowdhury (Reg. No. 74,466)
FISH & RICHARDSON PC
60 South Sixth Street, Suite 3200
Minneapolis, MN 55402
IPR39843-0181IP1@fr.com
PTABInbound@fr.com
axf-ptab@fr.com
jjm@fr.com
jjh@fr.com
rxc@fr.com

Date: June 27, 2025

/James A. Milkey/
James A. Milkey, Reg. No. 79,503
Reza Mirzaie, Reg. No. 69,138
Marc A. Fenster, *pro hac vice* forthcoming
Dale Chang, Reg. No. 61,476
Amy E. Hayden, Reg. No. 64,817
Neil A. Rubin, Reg. No. 67,030
Philip X. Wang, Reg. No. 74,621
RUSS, AUGUST & KABAT
12424 Wilshire Blvd., 12th Floor
Los Angeles, CA 90025

Counsel for Patent Owner