

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

AMERICAN AIRLINES, INC. AND SOUTHWEST AIRLINES CO.,

Petitioners,

v.

INTELLECTUAL VENTURES II LLC,

Patent Owner.

Case No. IPR2025-00987

U.S. Patent No. 8,407,722

PATENT OWNER'S PRELIMINARY RESPONSE

TABLE OF CONTENTS

PATENT OWNER’S EXHIBIT LIST	iv
I. Introduction.....	1
II. Summary Of The ‘722 Patent.....	3
A. Overview	3
B. Representative Claims	7
III. Claim Construction.....	9
IV. A Person of Ordinary Skill in the Art (POSITA).....	10
V. Prior Art Relied On.....	10
A. Lawson (EX1005).....	10
B. Tsutsumitake (EX1006).....	12
C. Choquier (EX1007).....	12
D. Bird (EX1008).....	14
E. Herz (EX1009).....	14
VI. Ground 1: Petitioner Ignores the Distinct Claim Limitations and Fails to Show Either the Claimed Mapping or the Claimed Registration.....	15
A. Petitioner has not shown the claimed “mapping ... controlling an amount of update message traffic through nodes” (Claims 1 and 7).....	17
1. Lawson is targeted delivery and consumer filtering—not a category→node-type mapping that controls node traffic...18	
2. Choquier is service-group dispatch and load balancing—not category→node-type mapping that controls update traffic through nodes.....	23
3. The combination is based on hindsight and never supplies the missing mapping-based control	26
B. Petitioner has not shown the claimed “registering ... provides client connection information to a node” (Claims 14/20/26/32)	32
1. The Petition never identifies, let alone maps, “client connection information,” and that failure is dispositive.....	33

2.	The References Do Not Disclose Client Registration That Provides Connection Information to a Node	34
3.	What the Petition offers instead only highlights why Ground 1 fails as a matter of law.	36
C.	The dependent claims (2–6, 8–13, 15–19, 21–25, 27–31, 33–37) are patentable at least by virtue of their dependency—and, independently, for their additional limitations.....	38
VII.	Ground 2: Adding Herz does not cure the core defects in Petitioner’s case on the independent claims and therefore the challenged dependents stand at least by virtue of their dependency.....	40
VIII.	Ground 3: Petitioner Conflates Distinct Claim Embodiments and Fails to Show (i) the Mapping that Controls Node Traffic (Claims 1–13) or (ii) Registration that Provides Client Connection Information (Claims 14–37)	43
A.	Choquier , Bird , and Tsutsumitake still do not teach the claim-critical gateway mapping that controls update traffic through routing nodes as recited in Claims 1-13.....	44
1.	Bird is a content-processing publish/subscribe broker—not a typed-node routing network with a mapping that controls node throughput	45
2.	Choquier describes CPU-based session/load balancing and group-wide replication—neither is the claimed mapping-based control	46
3.	Tsutsumitake is a direct server↔client event-request push architecture—no routing network, no node types, and no gateway-kept category→node-type mapping.....	47
4.	The combination still fails because the claimed mapping mechanism is nowhere to be found	48
B.	The combination does not disclose “registering ... provides client connection information to a node” (claims 14/20/26/32).....	49
1.	Ground 3 Fails to Show that Client Registration Provides Client Connection Information to a Node	50

2.	Petitioner’s reliance on Tsutsumitake’s (event-ID, client-ID) pair does not satisfy the claims’ “client connection information to a node”	53
C.	The dependent claims (2–6, 8–13, 15–19, 21–25, 27–31, 33–37) are patentable at least by virtue of their dependency.....	57
IX.	Ground 4: Adding Herz to Choquier, Bird, and Tsutsumitake still does not supply the missing limitations of the independent claims and therefore the challenged dependents stand at least by virtue of their dependency.....	60
X.	Conclusion	62

PATENT OWNER’S EXHIBIT LIST

Ex. 2001	American Airlines Webpage – Information Technologies
Ex. 2002	Abstract of Title for U.S. Patent No. 8,407,722
Ex. 2003	Agreed Scheduling Order in Southwest Litigation
Ex. 2004	Docket from <i>Intellectual Ventures I LLC et al. v. American Airlines, Inc.</i> , 4:24-cv-980 (EDTX)
Ex. 2005	Order Denying Motion To Sever And Stay in from <i>Intellectual Ventures I LLC et al. v. American Airlines, Inc.</i> , 4:24-cv-980 (EDTX)
Ex. 2006	Defendant American Airlines, Inc.’s P.R. 3-3 Invalidity Contentions in <i>Intellectual Ventures I LLC et al. v. American Airlines, Inc.</i> , 4:24-cv-980 (EDTX)
Ex. 2007	Defendant Southwest Airlines Co.’s Preliminary Invalidity Contentions in <i>Intellectual Ventures I LLC et al. v. Southwest Airlines, Co.</i> , 7:24-cv-277 (WDTX)
Ex. 2008	Docket from <i>Intellectual Ventures I LLC et al. v. Southwest Airlines, Co.</i> , 7:24-cv-277 (WDTX)

I. Introduction

The Board should deny institution because the Petition never engages with the claim-critical architecture that the US 8,407,722 (the '722 patent) actually claims and, as a result, fails to demonstrate a reasonable likelihood of prevailing on any challenged claim. The patent discloses multiple embodiments for scalable, targeted delivery of live-object updates, and the claims are drafted to reflect different aspects of those embodiments. Claims 1–13 recite a gateway-applied routing mechanism in which categories are assigned to update messages, types are assigned to routing nodes, and a gateway-kept mapping from categories to node types is used to route the update—“the mapping” itself controlling how much update traffic flows through the routing nodes. Claims 14–37, by contrast, omit the "traffic control" limitations in Claims 1-13 and add a client-facing registration requirement: when a client registers “for updates of the live object with the routing network,” that act of registration “provides client connection information to a node,” enabling node-side delivery of subsequent updates to the correct client connection.

The Petition does not rigorously apply the prior art to the different requirements of these two sets of claims. Instead, the Petition treats all claims as if they rose or fell on generic notions like “traffic control” or “subscription.” That approach never addresses the limitations that are uniquely claimed for each embodiment: for claims 1–13, the Petition identifies no gateway-held

category→node-type mapping that is actually applied to route an update and thereby control traffic (e.g., regulate throughput) through typed routing nodes; for claims 14–37, the Petition identifies no disclosure where client registration with a routing network provides client-connection information that the routing network later uses to push the update to the correct client connection. This lack of rigor in the Petition pervades every ground and leaves claim elements unmapped.

Ground 1 (Lawson/Choquier/Tsutsumitake) challenges claims 1-37 and is framed around load-balancing, endpoint selection, and single-server push, but those are not the claimed gateway mapping or node-resident connection records.

Ground 2 adds Herz to challenge dependent claims 3, 5, 9, 11, 12, 15, 21, 27 and 33. However, the multicast, caching, or replication taught in Herz is not equivalent to a mapping of category to node-type maintained in a node of the routing network that controls message traffic (e.g. throughput) in nodes of the routing network, nor do they supply registration-provided connection information to the routing network.

Ground 3 (Choquier/Bird/Tsutsumitake) challenges claims 1-37 and re-labels content processing and handler selection as “traffic control,” but fails to explain the mapping-based routing of claims 1–13 or the provision of connection information to the routing network when the client registers with the network for live updates as recited in claims 14–37.

Ground 4 layers Herz onto the same deficient mappings in Ground 3 in attempting to challenge dependent claims 3, 5, 9, 11, 12, 15, 21, 27 and 33.

Legally, Petitioner's burden at institution is to identify, with particularity, where each and every claim element is taught in the prior art or why it would have been obvious. It cannot rely on high-level characterizations, hindsight reconstruction, or inherency to fill gaps where the cited references neither disclose nor necessarily result in the claimed features. Because the Petition proceeds from the incorrect premise that the challenged independent claims are substantively the same and never addresses the limitations unique to each claim, the Petition fails to show a reasonable likelihood of success on any challenged claim.

For these reasons, and as further detailed in the sections that follow, the Board should deny institution under 35 U.S.C. § 314(a).

II. Summary Of The '722 Patent

A. Overview

The '722 patent tackles the core inefficiencies of the then-prevailing Web delivery model, where stateless servers and client-driven refresh forced users or scripts to repeatedly reload pages to discover changes, wasting bandwidth and server resources and still missing timely updates (EX1001, 16:1–65). The specification explains that techniques such as periodic meta-refresh or bespoke applets are wasteful and not scalable, because the server must resend entire pages (often

unchanged) and the client never knows when data actually changed (EX1001, 16:1–65). The invention replaces that with a dynamic content routing network that delivers targeted update messages for “live objects” displayed at the client. Each live object has an object ID, and an update message “identif[ies] a live object and contain[s] data for updating a property of the live object,” enabling precise, real-time changes without full-page reloads (EX1001, 8:9–35; 9:45–56; 10:11–36).

To do this at scale, the patent discloses two complementary routing architectures. First, the patent discloses a hierarchy-of-registries approach in which nodes maintain a node registry (732) keyed to the object IDs for which their connected clients have registered; the registry stores connection-level delivery information sufficient for the node to address and deliver the update to the correct, current client connection (EX1001, 17:9–13; 20:23–38). An example of this embodiment is shown in Fig. 8 (reproduced below).

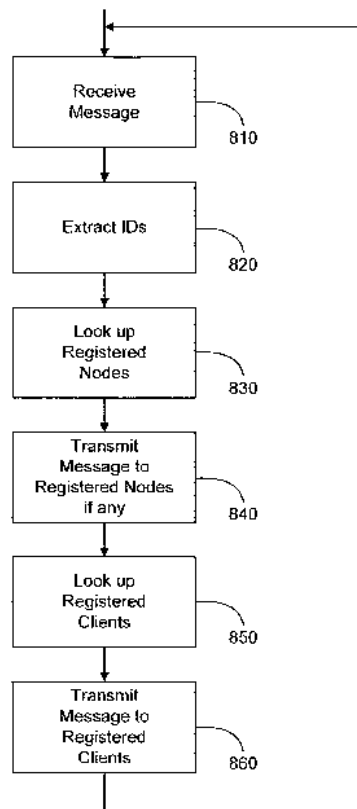


Fig. 8

Second, the patent discloses a category / node-type specialization approach that explicitly assigns messages to categories, assigns nodes to types, and maintains “mappings... that specify which categories of messages are forwarded to which types of nodes. The mappings allow control over the amount of traffic processed by the nodes.” A gateway “holds” and applies these mappings: on receipt of an update message it identifies the message’s category → identifies the mapped node type(s)

→ routes accordingly (EX1001, 18:26–29; 21:66–22:16). An example of this embodiment is shown in Fig. 10 (reproduced below).

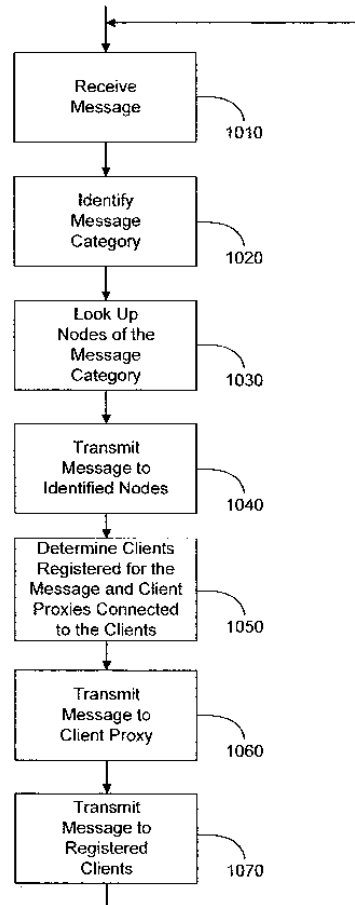


Fig. 10

These mechanisms solve two different—but related—problems. On the delivery side, the hierarchy-of-registries ensures reliable, one-to-one targeting of the exact client connection that registered for a given object, minimizing unnecessary traffic (EX1001, 17:9–13; 20:23–38). On the throughput side, the category→node-type mappings give the gateway an ex-ante traffic-shaping rule that determines

which specialized nodes receive which classes of updates, thereby controlling load through the routing fabric instead of broadcasting or overfeeding nodes (EX1001, 18:26–29; 21:66–22:16). The claims mirror this division of labor: claims 1–13 recite a gateway-applied category→node-type mapping that controls the amount of update-message traffic through routing nodes, while claims 14–37 add the client-facing registration requirement that registration “provides client connection information to a node,” i.e., connection-level delivery information at the node that enables the node to target the correct client connection when an update arrives—more than a mere client identifier of the kind seen in the cited prior art (EX1001, 18:26–29; 21:66–22:16; 24:8–36; 17:9–13; 20:23–38).

B. Representative Claims

<u>Mapping-driven routing</u> (e.g. Claim 1)	<u>Client Registration</u> (e.g. Claim 14)
1. A method comprising:	14. A method comprising:
receiving, using a processing device, an update message from an input source, the update message identifying a live object and containing data for updating a property of the live object;	providing, using a processing device of an input source, a data representation to a client device, different from the input source, coupled to a routing network, wherein the data representation includes at least one live object recognizable by the client device, and causing the client device to respond to the live object of the data representation by determining an object identifier (ID) of the live object and to register for updates of the

	live object with the routing network, <u>such that registering the client device with the routing network provides client connection information to a node in the routing network; and</u>
identifying a category of the update message based on the input source;	sending, using the processing device of the input source, an update message to the routing network, wherein the update message identifies the live object and contains update data that updates a property of the live object,
<u>determining a node having a node type to which the update message is to be routed based on a mapping of categories of update messages to node types, the mapping controlling an amount of update message traffic through nodes of a routing network;</u>	wherein a gateway device at the routing network is configured to identify a category of the update message based on the input source, to determine a node type to which the identified category maps, and to route the update message to the node, having the node type, at the routing network,
routing, using the processing device, the update message to the node having the determined node type;	wherein the node is configured to identify the client device as a registered device and to route the update message to the client device, and
causing the node, through the update message, to determine a client, different from the input source, that has registered for updates of the live object;	wherein the client device processes the update message upon receipt to update the property of the live object at the client device.
causing the node to route the update message from the node to the client; and	
causing the client to process the update message and to update the property of the live object	

III. Claim Construction

Consistent with Patent Owner’s positions in the related district court litigations, the Board should apply the plain and ordinary meaning to all challenged claim terms under the Phillips standard. The governing regulation provides that, in inter partes review, “a claim of a patent ... shall be construed using the same claim construction standard that would be used to construe the claim in a civil action under 35 U.S.C. 282(b),” i.e., the standard articulated in *Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005) (en banc). See 37 C.F.R. § 42.100(b). Under Phillips, claim terms are given their ordinary and customary meaning as understood by a person of ordinary skill in the art at the time of the invention, in the context of the intrinsic record. 415 F.3d at 1312–14. Extrinsic evidence “is less significant than the intrinsic record” and may not contradict it. *Id.* at 1317–19; see also *Vitronics Corp. v. Conceptronic, Inc.*, 90 F.3d 1576, 1582–84 (Fed. Cir. 1996).

Applying these principles, the terms used in the challenged claims (including, for example, “routing network,” “client connection information,” “node type,” and “live object”) should be afforded their plain and ordinary meaning in view of the specification and prosecution history. To the extent Petitioner advances extra-textual constructions that broaden or narrow the claims beyond what a skilled artisan would understand from the intrinsic record, those proposals should be rejected as

inconsistent with *Phillips* and its progeny. See *Phillips*, 415 F.3d at 1312–14; *Vitronics*, 90 F.3d at 1582–84.

Accordingly, and in alignment with Patent Owner’s positions in the parallel district court actions, the Board should resolve this IPR under the plain-and-ordinary-meaning standard without adopting any special constructions beyond what is necessary to address a concrete dispute

IV. A Person of Ordinary Skill in the Art (POSITA)

A person of ordinary skill in the art as of the priority date would have held at least a bachelor’s degree in computer science, computer engineering, or an equivalent field, and ~2–4 years of experience building distributed systems and client–server/web messaging architectures (or a graduate degree with less experience), with familiarity in publish/subscribe models, socket-based connection management, and gateway-mediated routing.

V. Prior Art Relied On

A. Lawson (EX1005)

Lawson addresses event-notification systems that relied on broad dissemination of notifications, noting that “events tend to be broadcast events,” which results in notifying many uninterested recipients and unnecessary network traffic (EX1005, 2:21–36). To avoid broadcast-style traffic, Lawson implements a targeted endpoint-selection architecture driven by registries rather than any

intermediate routing-fabric controls. Lawson organizes event delivery around two complementary registries: a global event registry listing which servers should receive notification for a given event, and a local event registry at each server listing which local event consumers at that server should be notified (EX1005, 10:22–31; 10:36–64). Operationally, when a server detects an event, it consults the global event registry to identify the specific remote server(s) to be notified, and the destination server then consults its local event registry to notify the appropriate local consumers (EX1005, 10:63–67). The event packet carries at least a “type of event” field that is used to extract from the global event registry the servers that should receive the event and from the local event registry the local consumers that should be notified (EX1005, 12:36–43).

Lawson’s exemplar shows server B notifying server E “without broadcasting” to other servers and explains that, by operating in this way, message traffic is reduced relative to broadcast-type events (EX1005, 10:40–46). That reduction flows from endpoint targeting via registry lookups—first to identify destination server(s), then to identify local consumer(s)—not from any intermediate routing policy or traffic shaping through routing nodes, as confirmed by the same passage reiterating the global-then-local registry flow (EX1005, 10:63–67). Accordingly, Lawson teaches a server/consumer registry system that performs targeted endpoint notification by using event-type information in the event packet to identify destination servers

through a global registry and then notifying local consumers identified in a local registry at those servers (EX1005, 10:22–31; 10:36–64; 10:63–67; 12:36–43).

B. Tsutsumitake (EX1006)

Tsutsumitake describes a single-server, event-request push system in which a web page displayed at the client can be dynamically updated by event notifications sent back from the same server. On the client, a page receiving unit 112 parses the downloaded page to detect whether an event request is embedded; if so, it notifies an event request unit 113, which sends the event request to the server (EX1006, 10:56–67; 4:22–29). On the server, an event request receiving unit 104 coordinates with a connection management unit 105 that checks whether a (event ID, client ID) pair is already registered and, if not, stores that pair as a subscription record (EX1006, 11:45–55). The server then pushes event notifications to the client over a maintained, per-client connection, with keep-alive “system events,” and the connection is cut or resumed when the user navigates (EX1006, 12:1–17; 12:33–63; 21:24–43). The reference illustrates page updates such as current value and device state changes driven by incoming events (Fig. 7) (EX1006, 13:46–57; 13:32–14:3).

C. Choquier (EX1007)

Choquier discloses a service-oriented data-center architecture that distributes client service requests across a pool of application servers using a service map that

records, for each service and server, dynamic CPU LOAD/CPU INDEX values. Gateways consult the service map to choose a server ID to handle an incoming client request under the rubric “DYNAMIC LOAD BALANCING BY ROUTING OF SERVICE REQUESTS.” (EX1007, 11:2–22; 11:15–22.) The reference explains that service descriptions in the service map are “used by the Gateways ... to locate servers ... for handling client service requests,” and that the map is periodically broadcast so gateways keep a current view of server availability and load (EX1007, 12:33–36; 12:56–61). In short, Choquier’s “routing” concerns session/request dispatch to an application server based on server load, not message-level routing through a typed routing fabric.

For updates, Choquier introduces an Arbiter associated with a service group of servers. When any server in the group generates an update transaction, the Arbiter “dispatches each update transaction to every server in the service group”—including the originating server—so that all servers process the update in the same serialized order; the Arbiter tracks outcomes and can remove outlier servers that diverge (EX1007, 21:27–50; 21:51–60). The Arbiter learns group membership via a global registry that maps servers to service groups (EX1007, 21:40–43). Notably, this design replicates each update to all servers in the group, which is the opposite of selectively limiting which nodes process a given update.

D. Bird (EX1008)

Bird discloses a content-aware publish/subscribe broker that not only routes messages by topic but also analyzes message content, selects a message-handler module (often multimedia processing), and generates one or more derived messages that are then sent to subscribers whose stored requirements/capabilities are satisfied. The reference explains that the broker identifies message-content characteristics and uses that identification to determine which of a set of message-handler modules to pass the message to; the selected handler processes the message and returns a derived message for delivery to the appropriate subscribers (EX1008, ¶[0035]; see also claims 21–24 (identify non-alphanumeric content → select handler → generate derived message(s) → send to subscribers)). Bird’s processing flow is depicted in Fig. 4 (steps 180–280): topic check, byte-sequence analysis to identify content type, comparison against subscriber requirements and system capabilities, formation of a processing task, handler selection, and finally derived-message delivery (EX1008, Fig. 4; claim 17 (analysing byte sequences ... to identify multimedia content); claim 21).

E. Herz (EX1009)

Herz (EX1009) discloses a content-distribution and personalization architecture that groups users/content by topic or community and pre-positions content in the network for faster retrieval by selecting core servers for a topical group

and constructing an (often unrooted) multicast tree spanning those cores; files (or descriptors) are then broadcast along the tree to the cores for caching/availability (EX1009, 49:19–27; 49:31–46; 50:21–37). The reference details a tree-construction procedure (e.g., a minimum-spanning tree over the core servers) and a broadcast “message M” carrying the cluster profile, list of core servers, and tree topology, with each proxy storing either its subtree (if a core) or a list of nearby cores (if not) (EX1009, 51:1–16; 51:16–28). Under “Global Requests to Multicast Trees,” a global request is forwarded so all core servers eventually receive a copy and act on it—again, broadcast over the multicast tree (EX1009, 53:1–8; 53:16–38). For “virtual communities,” Herz maintains a separate multicast tree MT(V) for each community; when a user joins, the proxy may be designated a core server and added to the tree, and messages on that tree are delivered to all core servers for that community (EX1009, 77:7–18; 78:35–43).

VI. Ground 1: Petitioner Ignores the Distinct Claim Limitations and Fails to Show Either the Claimed Mapping or the Claimed Registration

Ground 1 rests on a single, over-generalized “traffic control” narrative. That narrative never engages with the claim-specific routing mechanisms the ’722 patent actually claims, nor with the fact that the patent teaches distinct embodiments and the claims select different features from those embodiments. The result is a petition

that treats all claims as if they rose or fell on the same generic notion of “less messaging,” while skipping the elements that matter.

For claims 1–13, the heart of the invention is a routing network that performs routing “based on a mapping of categories of update messages to node types,” where “the mapping [controls] an amount of update message traffic through nodes of a routing network.”(EX1001, 22:36-38.) The specification is explicit: messages are assigned to categories, nodes are assigned to types, and “[m]appings are created that specify which categories of messages are forwarded to which types of nodes. The mappings allow control over the amount of traffic processed by the nodes.” (EX1001, 18:26–29.) The routing network holds the mapping data structures and, upon receiving a message, identifies the category → identifies the mapped node type(s) → routes accordingly (EX1001, 21:66–22:16; Fig. 10, steps 1020→1030→1040).

For claims 14–37, the claims retain that category→node-type routing but add a different, client-facing requirement directed to registration and delivery: when a client “registers for updates of the live object with the routing network,” that act of registering “provides client connection information to a node” (EX1001, 24:8–36).

Petitioner never grapples with these different claim thrusts. For claims 1–13, the petition recasts Lawson’s non-broadcast endpoint notification and Choquier’s CPU-based server selection as “traffic control,” yet never identifies the claimed

category→node-type mapping in the routing network that controls how much traffic flows through routing nodes (see EX1001, 18:26–29; 21:66–22:16). For claims 14–37, Petitioner reuses the same combination but never even identifies—much less maps—any prior-art disclosure of the claimed “client connection information” supplied to a node by client registration (EX1001, 24:8–36), i.e., the node-resident client connection information the node uses to put the update on the correct client connection (EX1001, 17:9–13; 20:23–38).

By failing to appreciate that the two claim sets are directed to different aspects/embodiments—and by not addressing the elements unique to each—the Petition never meets its burden for Ground 1.

A. Petitioner has not shown the claimed “mapping ... controlling an amount of update message traffic through nodes” (Claims 1 and 7)

Claims 1 and 7 require routing “based on a mapping of categories of update messages to node types,” and they further require that “the mapping [controls] an amount of update message traffic through nodes of a routing network.” In the specification, the patent explains exactly how this works: the system “assign[s]” messages to categories and nodes to types, and “Mappings are created that specify which categories of messages are forwarded to which types of nodes. The mappings allow control over the amount of traffic processed by the nodes.” (EX1001, 18:26–

29). The routing network “holds” the data structures that specify node types and store the mappings; when an update message is received, the routing network “determines the category” and then “determines the node type(s) to which the message category maps,” and routing takes place on that basis. (see EX1001, 21:66-22:16; Fig. 10 (steps 1020→1030→1040)).

Accordingly, the claimed “mapping” itself is the ex-ante routing rule at the mapping network that decides which node types will receive which categories, thereby governing how much update traffic each node type must process. It is not a generic observation that some traffic is lower; it is a specific mapping between category and node type that is used in routing and, by doing so, controls the amount of update traffic that flows through the nodes that belong to those types.

1. Lawson is targeted delivery and consumer filtering—not a category→node-type mapping that controls node traffic

The '722 Patent, “assigns messages to one or more categories, assigns nodes to one or more types, and maintains mappings between categories and types” EX1001, 18:24–29. For example, “all messages in the routing network 110 are assigned to one or more of N categories, and all of the nodes are assigned to one or more of M types. Mappings are created that specify which categories of messages are forwarded to which types of nodes. The mappings allow control over the amount

of traffic processed by the nodes.” Id., 18:24-29. In connection with this disclosure, claim 1 of the ’722 Patent recites in pertinent part:

determining a node having a node type to which the update message is to be routed based on a mapping of categories of update messages to node types, *the mapping controlling an amount of update message traffic through nodes of a routing network*;

Independent claim 7 contains a similar limitation. These limitations in claims 1 and 7 require that the claimed mapping of categories of update messages to node types function to control the amount of update message traffic through nodes of the network. Put another way, the claim language does not merely require control of the amount of update message traffic through the nodes, but additionally specifies that it is the mapping of categories of update messages to node types controls the amount of update message traffic through the nodes of the network.

With respect to the claim limitations requiring control of the amount of update message traffic through the nodes, the Petition fails to explain how, in the relied on references, such control *results from a mapping of categories of update messages to node types*, as required by claims 1 and 7. According to the Petition:

... Lawson’s example above regarding Fig. 2 specifies the “event notification was transferred from server B to server E without broadcasting the event to servers A, C and D,” and “[t]hus, such a method can globalize an event to required locations while, simultaneously, ***reducing message traffic*** due to broadcast type events.” Petition, 42 (emphasis in original).

Assuming *arguendo* that Lawson’s reduction of message traffic as compared to broadcast events equated to controlling an amount of update message traffic through nodes of a routing network, the Petition fails to clearly articulate how such control results from a mapping of categories of update messages to node types, as required by claims 1 and 7.

Petitioner points to Lawson’s example where “event notification was transferred from server B to server E without broadcasting ... reducing message traffic due to broadcast,” and argues that this shows the claimed “controlling ... message traffic.” (EX1005, 10: 40–46.) But Lawson achieves reduced broadcast by checking a global registry to see which specific servers have registered consumers, and then sending the event to those servers—i.e., it is server-specific delivery based on consumer registration and filtering. (See, e.g., EX1005, 10:63–67, 12:36–43). The Petition fails to explain how Lawson discloses a mapping from message categories to node types, and fails to explain any use of such a mapping to control the amount of update traffic processed by the node.

Instead, Lawson’s core mechanism is endpoint selection using registries, not typed-node routing. The specification describes a global event registry that lists which servers should be notified for each event, and local event registries at each server that list the local event consumers. In operation, the local server that detects an event checks the global event registry to identify the specific remote servers to

notify and then, at each notified server, consults its local registry to notify local consumers (EX1005, 10:63–67). Lawson’s figures and narrative likewise depict networks with global registries 34/36 and local registries 38/40 and explains that these registries need only contain information sufficient to identify which servers should receive which events (EX1005, 10:22–31; 10:36–64).

Lawson’s worked example underscores that this is targeted server-specific delivery—not a gateway-kept category→node-type mapping. Lawson explains that server B transfers the event to server E “without broadcasting ... reducing message traffic due to broadcast,” because all server B needed to know (from the registry) was that server E required notification (EX1005, 10:40–46). The example then reiterates the two-step endpoint flow: check the global registry to find server E; server E checks its local registry to notify the local consumer (EX1005, 10:63–67). Nothing in this example (or elsewhere in Lawson) describes routing based on node types or a gateway’s category→node-type mapping used to route messages through a routing fabric.

Lawson also implements consumer-side filtering, further confirming its endpoint paradigm. The specification describes storing filtering criteria so that only events meeting the stored criteria are sent to a given consumer, and even supports “directed events” where the intended consumer’s identity drives filtering (EX1005, 20:31–47, 22:15–20, 17:52–55). Filtering is expressly tied to the local registry /

event consumer context, not to any gateway mapping that controls throughput through routing nodes (EX1005, 17:44–55).

Even the way Lawson packages events confirms endpoint lookup rather than typed-node routing: the event packet includes a “type of event” field that is used to extract from the global registry the servers that should receive the event and from the local registry the consumers that should receive it (EX1005, 12:36-43). That is a registry-keyed destination selection workflow—event-type → servers/consumers—not category (based on input source) → node type as required by claims 1 and 7 (which relies on gateway-held category→node-type mappings to control the amount of update traffic through routing nodes).

Finally, Lawson’s problem statements emphasize reducing broadcasts and registration burdens in event notification systems—again, problems solved by server/consumer registries and filtering, not by typed-node traffic control. The specification criticizes prior systems for “broadcast events” that notify many uninterested users and proposes registries/filters to avoid such broadcasts (EX1005, 2:21–36). That is a different concept from the “mapping” of claims 1 and 7, where a gateway controls which node types receive which message categories to control traffic through the nodes.

2. Choquier is service-group dispatch and load balancing—not category→node-type mapping that controls update traffic through nodes

Petitioner also cites Choquier’s “service groups,” “service maps,” and load balancing, asserting this shows “controlling traffic.” (See, e.g., EX1007, 11:2–22; 12:33–36; 12:56–61.) But Choquier’s “service map” contains server IDs, service types, and load data and is used by gateways to select a server within a service group for client service requests and by the system to perform dynamic load balancing/allocation—that is classical resource balancing, not the claimed category→node-type mapping that controls the amount of update message traffic through nodes. (EX1007, 11:2–22; 12:33–36; 12:56–61.) Indeed, when Choquier handles update transactions, the Arbiter “dispatches each update transaction to every server in the service group,” i.e., it replicates updates to **all** servers in the group, which is the opposite of using a category→node-type mapping to limit which nodes must process the update messages. (EX1007, 21:27–50; 21:51–60).

Specifically, Choquier’s host data center maintains a service map that lists services, the servers currently providing them, and each server’s CPU LOAD and CPU INDEX. Gateways read this service map to choose a specific server ID to handle a client service request, and the point of the map is to distribute session load among servers—i.e., dynamic load balancing—not to route update messages by message category to types of routing nodes. Choquier describes how CPU LOAD/CPU

INDEX are used for server selection and places the selection logic under “DYNAMIC LOAD BALANCING BY ROUTING OF SERVICE REQUESTS.” (EX1007, 11:2–22; 11:15–22). It further explains that service descriptions in the service map are “used by the Gateways ... to locate servers ... for handling client service requests” (EX1007, 12:33–36), and that the service map is periodically broadcast to all gateways so they remain current on which servers should receive sessions (EX1007, 12:56–61). None of this is a gateway-kept mapping of message categories to node types used to route update messages.

In addition, Choquier handles updates in the exact opposite manner of Claims 1 and 7. For updates, Choquier uses an Arbiter. When any server in a service group generates an update transaction, the Arbiter “dispatches each update transaction to every server in the service group,” including the originator, so all servers process the update in the same serialized order (EX1007, 21:27–50). The Arbiter tracks outcomes per server and removes outliers from service if they diverge (EX1007, 21:51–60). This is explicit replication to all servers of the group, which increases the number of servers that must process each update; it does not “control the amount of update message traffic through nodes” by selectively routing updates to a subset of node types.

Where Choquier does maintain a registry-style mapping, it is a “global registry” that maps servers to service groups, enabling the Arbiter to know the

servers comprising each group (EX1007, 21:40–43). That is not a mapping of message categories to node types, and it is not used by a gateway to perform category-based routing of update messages.

Accordingly, the only “traffic control” Choquier performs is distributing client service requests among servers using CPU LOAD/INDEX (EX1007, 11:15–22; 12:33–36) and periodically broadcasting updated service maps to gateways (EX1007, 12:56–61). That is session/request dispatch, not the category→node-type routing rule required by the claims—i.e., a rule that determines which nodes receive which update messages in order to control the amount of update-message traffic through nodes.

Therefore, Petitioner’s reliance on Choquier’s “service groups,” “service maps,” and “Arbiter” cannot satisfy the claim requirement of routing “based on a mapping of categories of update messages to node types,” where “the mapping [controls] an amount of update message traffic through nodes.” In Choquier:

- (i) Gateways select servers for client service sessions using CPU-based load balancing (EX1007, 11:15–22; 12:33–36; 12:56–61);
- (ii) Updates are replicated to every server in the service group (EX1007, 21:27–50), not selectively routed to a subset of typed nodes; and
- (iii) the Arbiter’s mapping is servers↔service groups (EX1007, 21:40–43), not categories↔node types.

Even if one tries to equate Choquier’s “service groups” with “node types,” Choquier’s architecture still does not route update messages using a category→type mapping at the gateway. Instead, it replicates each update to all servers in the group (EX1007, 21:27–50) and uses the service map to load-balance client sessions (EX1007, 11:15–22; 12:33–36; 12:56–61). That is the opposite of the claim-required mapping-based control of update-message traffic through routing nodes.

3. The combination is based on hindsight and never supplies the missing mapping-based control

The Petition’s purported motivation for combining Lawson and Choquier does not explain why a POSITA would have assigned Lawson’s servers A, B, C, D according to “topic” or set of “topics” as allegedly “disclosed in Choquier, and for Lawson’s servers to determine which server groups (node types) should receive an event update based on this server groups to event producer mapping.” Petition, 44 (emphasis added). According to the Petition, a POSITA would have been motivated to combine Choquier with Lawson to “provide Lawson’s system with “a high degree of scalability,” “dynamic[] balancing of processing load among...servers,” and redundancy regarding server failure or overloading.” Petition, 23. The Petition fails to tie these generic design goals to the specific functionality that the Petition seeks to cherry-pick from Choquier and import into Lawson. Put another way, the Petition does not explain what about Choquier's scalability, load balancing and redundancy

techniques would have motivated a POSITA to have configured Lawson's servers to "determine which server groups (node types) should receive an event update based on this server groups to event producer mapping." That is because this aspect of the Petition is based on hindsight using the '722 Patent claims as a checklist rather than on anything in Lawson or Choquier, neither of which uses a category to node type mapping to control the amount of traffic through the nodes.

This gap in the Petition's arguments purporting to combine Lawson with Choquier is exemplified in, e.g., the Petition's analysis of [1.2]/[7.2]. There, the Petition alleges that "Choquier discloses assigning servers to server groups (node types) based on message categories ("a specific topic or set of topics") using service maps." Petition, 43 (emphasis added). However, when attempting to combine Choquier with Lawson, the Petition abandons Choquier's teaching of message categorization based on "a specific topic or set of topics," and substitutes (without explanation) Lawson's so called "event producer" based filtering in place of Choquier's message categorization based on a specific topic. From the Petition:

... It would have been obvious based on the combination of Lawson and Choquier for Lawson's servers to be included in server groups to provide for server groups/clusters (e.g., providing servers groups at each of servers A, B, C, D in Lawson) assigned according to "topic" or set of "topics" (message category of "event producer" based on Lawson's filtering) as disclosed in Choquier, and for Lawson's servers to determine which server groups (node types) should receive an event

update based on this server groups to event producer mapping. Petition, 43-44 (emphasis added).

Completely absent from the Petition is any explanation why a POSITA would have been motivated to substitute Lawson's so called "'event producer based filtering' in place of Choquier's message categorization based on 'a specific topic or set of topics.'" None of the purported motivations identified in the Petition -- i.e., such as increasing "efficiency, redundancy or scalability of Lawson's routing network" (see, Petition, 44) -- explain why a POSITA would have been motivated to substitute Lawson's so called "'event producer" based filtering' in place of Choquier's message categorization based on 'a specific topic or set of topics.'"

Besides being based on hindsight, Ground 1 re-labels Lawson's non-broadcasting as "controlling ... message traffic" and Choquier's service-map balancing as further "traffic control," but neither reference—alone or as actually combined—discloses or suggests that the specific routing mechanism the claims require: a gateway-kept mapping from message categories (which may be based on input source) to node types, used to route the update message and thereby control the amount of update-message traffic through routing nodes. The '722 patent explains that messages are assigned to categories, nodes to types, and "mappings are created that specify which categories of messages are forwarded to which types of nodes. The mappings allow control over the amount of traffic processed by the

nodes.” (EX1001, 18:26–29.) The patent further teaches that the gateway holds the data structures that specify node types and the mappings; on receipt of a message the gateway identifies the category, then identifies the node type(s) to which the category maps, and routes accordingly (EX1001, 21:66–22:16; see also Fig. 10, steps 1020→1030→1040).

Lawson does not provide this missing mapping. Lawson routes by endpoint selection—a server that detects an event consults a global event registry to identify which servers should be notified, and the destination server consults its local registry to notify local consumers (EX1005, 10:63–67). Lawson’s event packet simply provides a type-of-event field that is used to extract recipients from the global and local registries (EX1005, 12:36–43). That is registry-keyed destination delivery to endpoints—not a gateway-kept category→node-type mapping used to route update messages through typed routing nodes. There is no disclosure in Lawson of (i) message “categories” (e.g., based on input source), (ii) “node types” inside a routing fabric, or (iii) a mapping held at a gateway that is applied ex ante to determine which routing nodes must receive which update messages (EX1005, 10:22–31; 10:36–64; 10:40–46; 12:36–43).

Choquier likewise does not provide the missing mapping. Choquier’s service map lists services, servers, and CPU LOAD/CPU INDEX, and gateways use the map to pick a server for a client service request—“DYNAMIC LOAD BALANCING BY

ROUTING OF SERVICE REQUESTS” (EX1007, 11:2–22; 11:15–22). The service descriptions “are used by the Gateways ... to locate servers ... for handling client service requests,” and the map is periodically broadcast to gateways (EX1007, 12:33–36; 12:56–61). For updates, Choquier’s Arbiter “dispatches each update transaction to every server in the service group,” ensuring replication/serialization, and removes divergent servers if needed (EX1007, 21:27–50; 21:51–60). Thus, Choquier’s “traffic control” is session load balancing and map broadcast, plus update replication to all servers in a group—not category-based selective routing of update messages to node types. There is no disclosure of message categories, no node types keyed by message category, and no gateway-held category→node-type mapping (EX1007, 11:2–22; 12:33–36; 12:56–61; 21:27–50; 21:40–43; 21:51–60).

In fact, Petitioner never identifies how Lawson’s registries of servers/consumers become the claimed gateway-kept category→node-type mapping (EX1005, 10:63–67; 12:36–43). Nor does Petitioner show that Choquier’s service map—not keyed by message category and used for CPU-based server selection—somehow supplies the missing category→node-type routing rule that controls update-message traffic through routing nodes (EX1007, 11:2–22; 12:33–36; 12:56–61). To the contrary, Choquier’s update path increases processing by replicating each update to every server in the service group (EX1007, 21:27–50), which is the

opposite of the claim's mapping-based control that limits which nodes must process a given update.

Therefore, even taken together, Lawson's endpoint registries and Choquier's service-map load balancing do not disclose or suggest a gateway-kept mapping of message categories to node types used to route update messages to control the amount of update-message traffic through nodes.

Accordingly, Ground 1 fails because it tries to substitutes labels for substance. Lawson discloses endpoint-based notification using registries keyed by event type, and Choquier discloses service maps for CPU-based load balancing and replication to all servers in a group. Neither reference, alone or as combined by Petitioner, teaches or suggests the claimed routing mechanism in which a gateway maintains mappings between message categories and node types and applies those mappings to control the amount of update-message traffic that traverses routing nodes. The Petition does not identify any disclosure of message categories, node types, or a gateway-kept category→node-type mapping, nor does it explain how registries of recipients or CPU-indexed server maps could be transformed into such a routing rule. The result is that Petitioner's theory leaves a core limitation of the claims entirely unsupplied. Ground 1 should therefore be rejected.

B. Petitioner has not shown the claimed “registering ... provides client connection information to a node” (Claims 14/20/26/32)

Independent claims 14/20/26/32 require that when the client device receives the data representation with a live object, the client registers with the routing network and, as part of that registration, it “provides client connection information to a node” so that subsequent update messages can be routed from the node to the correct client connection. For example, the ’722 Patent discloses that “[e]ach entry contains ... a socket identifier or some other information for identifying how to route the message to the named entity.” EX1001, 20:27-30 (emphasis added). Significantly, the claim language does not simply recite that registering the client device with the routing network provides “client information,” but instead requires that such registering provide “client connection information.” The adjective “connection” before information has an independent meaning that cannot be read out of the claim. While a list of persons wishing to receive notifications may constitute “client information,” such a list does not expressly or inherently include “client connection information,” which, for example, provides “information for identifying how to route the message to the named entity.” EX1001, 20:27-30. As explained more fully below, the Petition (at 31-33) at best identifies “client information” comprising a client name (e.g., an “event consumer” or “client ID”) in the relied on references; however, the Petition

fails to allege -- let alone demonstrate -- that the relied on references expressly or inherently disclose “client connection information,” as claimed.

1. The Petition never identifies, let alone maps, “client connection information,” and that failure is dispositive

Petitioner bears the burden to “identify, with particularity, the evidence that supports the grounds for the challenge” and to map each claim element to the prior art. 35 U.S.C. § 312(a)(3); 37 C.F.R. § 42.104(b)(4)–(5); see also *Harmonic Inc. v. Avid Tech., Inc.*, 815 F.3d 1356, 1363–65 (Fed. Cir. 2016) (the petition must present the evidence and arguments with particularity); *In re Magnum Oil Tools Int’l, Ltd.*, 829 F.3d 1364, 1375–78 (Fed. Cir. 2016) (burden of persuasion never shifts; Board may not supply missing arguments); *Intelligent Bio-Systems, Inc. v. Illumina Cambridge Ltd.*, 821 F.3d 1359, 1369–71 (Fed. Cir. 2016) (conclusory assertions are insufficient). The Board is confined to the theories in the Petition; it cannot rescue an omission with a new theory in reply or by its own reconstruction. 37 C.F.R. § 42.23(b); *Magnum Oil Tools*, 829 F.3d at 1380.

Here, the Petition never identifies what “client connection information” is, never says where the prior art discloses it, and never maps it to the claims. Indeed, in Petitioner’s own element-by-element mapping (see Petition, 30-34), the phrase “client connection information” does not appear at all. That omission alone defeats

Ground 1 for independent claims 14/20/26/32 (and their dependents that rise and fall with this limitation).

Because the Petition never identifies—let alone maps—any prior-art disclosure of “client connection information” as claimed, Ground 1 cannot meet Petitioner’s particularity burden. See 35 U.S.C. § 312(a)(3); 37 C.F.R. § 42.104(b)(4)–(5); *Harmonic*, 815 F.3d at 1363–65; *Magnum Oil Tools*, 829 F.3d at 1375–78; *Intelligent Bio-Systems*, 821 F.3d at 1369–71. The Board should decline to supply the missing theory or accept any new mapping first offered in reply.

2. The References Do Not Disclose Client Registration That Provides Connection Information to a Node

Lawson does not provide this node-side registration/connection requirement. Lawson’s model is endpoint selection: a server that detects an event consults a global event registry to identify which servers to notify, and a notified server consults its local registry to identify which local consumers to notify (EX1005, 10:63–67). Lawson’s event packet merely includes a type-of-event field that the system uses to extract recipients from the global and local registries (EX1005, 12:36–43). There is no disclosure that the act of client “registration” provides client connection information to a node.

Choquier likewise does not provide this node-side registration/client connection information requirement. In Choquier, gateways use a service map

(listing services/servers with CPU LOAD/CPU INDEX) to pick a server for a client service request—the selection logic is expressly described as “DYNAMIC LOAD BALANCING BY ROUTING OF SERVICE REQUESTS” (EX1007, 11:2–22; 11:15–22). The service descriptions “are used by the Gateways ... to locate servers ... for handling client service requests,” and the service map is periodically broadcast to keep gateways current (EX1007, 12:33–36; 12:56–61). That is session/request dispatch, not client registration with a routing network that provides client connection information to a node for later object-ID-based update routing. And when updates occur, Choquier’s Arbiter dispatches each update transaction to every server in the service group and monitors outcomes (EX1007, 21:27–50; 21:51–60)—again, nothing about node-resident client connection information..

Tsutsumitake does not fill the gap. Tsutsumitake teaches a server-push, event-request model: the client issues an event request, the server sends events over that maintained per-client connection, and the client may cut or resume that connection when navigating. (EX1006, 10:56–67, 4:22–29. 11:45–55). That is a direct client↔server mechanism, not registration with a routing network that provides client connection information to a node. In fact, there are no “nodes,” “node types,” or gateway-kept category→node-type mappings in that architecture. Nothing in Tsutsumitake converts Lawson’s endpoint registries or Choquier’s service-map

dispatch into the claimed client connection information provided to a node as part of client registration.

Accordingly, Petitioner never identifies any disclosure in Lawson, Choquier, or Tsutsumitake where a client device's act of registering with a routing network and providing client connection information to a node as the claims require. Therefore, the Petition fails to explain how the relied on references meet the claimed client device registration feature that "provides client connection information to a node" (EX1001, 24:8–36).

3. What the Petition offers instead only highlights why Ground 1 fails as a matter of law.

Petitioner first relies on Lawson, but Lawson is directed to endpoint selection, not node-side connection information. Petitioner cites Lawson's consumer "registration" keyed to event types and event consumers. Yet Lawson's registries function only for endpoint selection: a server that detects an event looks up which servers to notify in a global event registry, and the destination server looks up which local consumers to notify in a local event registry (EX1005, 10:63–67). The event packet merely includes a type-of-event field used to extract recipients from those registries (EX1005, 12:36–43). Nothing in Lawson is registration with a routing network that provides client connection information to a node, and Lawson never discloses a node-side registry storing socket identifiers or equivalent connection

metadata to deliver object-ID-based update messages (EX1005, 10:22–31; 10:36–64; 10:40–46; 12:36–43). Petitioner therefore provides no identification, and no mapping, of the claims’ “client connection information.”

Petitioner next points to Tsutsumitake, but that reference also fails. Tsutsumitake describes that if a client is “not registered,” the server stores a pair of (event ID, client ID) in a connection management unit. (EX1006, 11:45–55). This occurs entirely within a direct server-to-client event-request push architecture. (EX1006, 10:56–67; 4:22–29). A server-side “client ID,” however, is not the claims’ node-resident connection metadata that client registration provides to a node so the node can route an update message to the correct client connection. (EX1001, 24:8–36; 17:9–13; 20:23–38). Tsutsumitake contains no routing network, no nodes or node types, and no gateway-kept category→node-type mapping. Petitioner therefore identifies no node-side “client connection information” and offers no mapping of any Tsutsumitake field to that claim term.

Choquier fares no better. Choquier’s service map supports CPU-based session and load balancing, and its Arbiter replicates each update to every server in the service group. (EX1007, 11:2–22; 12:33–36; 12:56–61; 21:27–50). This has nothing to do with client registration providing client connection information to a node.

The consequence of Petitioner’s omission is decisive. None of the cited references discloses or suggests that registering a client device with a routing

network provides client connection information to a node. Absent this claim-critical registration feature, Petitioner’s combination cannot meet the requirements of the independent claims.

C. The dependent claims (2–6, 8–13, 15–19, 21–25, 27–31, 33–37) are patentable at least by virtue of their dependency—and, independently, for their additional limitations

Ground 1 challenges all claims 1–37, but it never establishes the unpatentability of the independent claims. As shown in Parts A and B, Petitioner does not supply (i) the claim-critical gateway-kept mapping of categories of update messages to node types that is used to route the update and thereby controls how much update traffic flows through routing nodes (independent claims 1/7), nor (ii) the later-claim requirement that “registering the client device with the routing network provides client connection information to a node” for subsequent node→client delivery (independent claims 14/20/26/32). Having failed to meet the independent claims, the Petition necessarily fails as to each dependent claim, which by definition includes all limitations of its respective independent claim.

In any event, Petitioner also does not carry its burden on the additional limitations of the dependents. For claims 2 and 8 (method/apparatus), the Petition never identifies prior art in which a node (not an endpoint server) extracts an object identifier from an update message and uses it to make node-side

registration/connection determinations (EX1001, 22:60–65 (Fig. 10 steps 1020→1030→1040)). For claims 3 and 9, nothing in the cited art discloses the claimed client-proxy path in which the node determines at least one client proxy for the client and routes the update to the proxy for delivery to the client (EX1001, 19:20–33; 20:23–38; Figs. 9–12). For claims 4/5 and 10/11, the references do not teach where the client-connection registration information resides—at the node (claim 4/10) versus at the client proxy (claim 5/11)—or that either stores connection-level delivery information enabling later node→client transmission (EX1001, 17:9–13; 20:23–38). Claim 6 (and its apparatus analog claim 12) recites the live-object property semantics and, in claim 12, a gateway that routes to a node adapted to receive messages of more than one message category with the node then determining a connection with at least one client proxy—features the Petition never shows in the prior art’s session-dispatch/load-balance or endpoint-registry models (EX1001, 18:26–29; 21:66–22:16).

Turning to the later dependent sets, claims 15–19 (and their apparatus/article/CRM counterparts 21–25, 27–31, 33–37) add requirements the combination likewise does not teach—most notably the activation-module behavior that executes at the client to register live objects and, in certain embodiments, determine the message category and/or node type and cause registration accordingly (EX1001, 19:20–33; 20:24–33; 24:49–62). The cited art has no client-side module

that reads page content to (i) identify live objects for a routing network, and (ii) select category/node-type registration targets as taught by the '722 patent. Nor does the combination disclose the apparatus (claims 21–25) or article/CRM forms (claims 27–31, 33–37) configured to perform these same functions—i.e., a gateway that applies a category→node-type mapping for update messages, and a node/proxy that maintains connection-level delivery information supplied by registration (EX1001, 18:26–29; 21:66–22:16; 24:8–36; 17:9–13; 20:23–38).

Accordingly, because Ground 1 fails on the independent claims 1, 7, 14, 20, 26, and 32, the challenges to all dependent claims—2–6, 8–13, 15–19, 21–25, 27–31, and 33–37—fail at least by virtue of their dependency. And even apart from that dispositive point, Petitioner has not shown where the prior art teaches the additional limitations that each dependent claim layers on top of its parent.

VII. Ground 2: Adding Herz does not cure the core defects in Petitioner’s case on the independent claims and therefore the challenged dependents stand at least by virtue of their dependency

Petitioner’s addition of Herz does not supply either of the independent-claim limitations the Lawson/Tsutsumitake/Choquier combination still lacks. Herz is about what content to distribute and where to cache/replicate it, not how to route an update message through typed routing nodes under a gateway’s category→type policy, and not how client registration with a routing network provides connection information to a node for later node→client delivery.

Herz uses similarity/interest information to select “core servers” for a set of target objects and then builds a topical multicast tree to broadcast files (not per-message updates) to those core servers for later retrieval. See, e.g., the description of selecting core servers and constructing an unrooted multicast tree for file broadcast to those servers, and of maintaining separate multicast trees for each topical group (EX1009, 49:19–27; 49:31–46; 50:21–37). Herz then describes the multicast tree construction procedure (minimum-spanning tree over core servers; message M describing the cluster profile, core servers, and tree topology), and the broadcast of message M to all proxy servers, with each proxy storing either a subtree (if it is a core) or a list of nearby cores (if it is not) (EX1009, 51:1–16; 51:16–28). Herz separately explains “Global Requests to Multicast Trees,” in which a global request message is forwarded so that “all core servers eventually receive a copy” and act on the embedded request—again, a broadcast across the multicast tree (EX1009, 53:1–8; 53:16–38). In another embodiment for “virtual communities,” Herz explicitly maintains a separate multicast tree MT(V) for each community; when a new user joins, the proxy may be designated a core server and added to the tree, and messages are then broadcast to all core servers in that tree (EX1009, 77:7–18; 78:35–43). None of this is the gateway-kept mapping of categories of update messages to node types that the ’722 patent claims and uses to route the update and thereby control the amount of update traffic through routing nodes. Herz’s “mappings,”

where present, are content/topic→core-server (or proxy↔core-server) arrangements in service of file distribution and caching, not category→node-type routing of update messages.

Nor does Herz supply the later-claim requirement that registration with the routing network “provides client connection information to a node.” Herz’s records concern user interest summaries, proxy-mediated/pseudonymous participation, and which servers belong on which multicast tree, and its “routing” either clusters/filter-selects content to show to users or replicates/broadcasts files to core servers for later retrieval (EX1009, 49:37–46; 50:1–6; 51:13–28; 53:1–38; 77:7–18; 78:35–43). Herz nowhere teaches that a client’s act of registering with a routing network deposits node-resident client connection information that a node later uses to target the correct client connection for a live-object update as the ’722 patent requires. Because Herz does not provide (i) a gateway-held category→node-type mapping applied to route update messages and control node throughput (independent claims 1/7), or (ii) registration-supplied client connection information to a node (independent claims 14/20/26/32), adding Herz cannot remedy the combination’s failure to meet the independent claims. It follows that dependent claims 3, 5, 9, 11, 12, 15, 21, 27, and 33—challenged only in view of that same defective theory—are patentable at least by virtue of their dependency on allowable independent claims.

VIII. Ground 3: Petitioner Conflates Distinct Claim Embodiments and Fails to Show (i) the Mapping that Controls Node Traffic (Claims 1–13) or (ii) Registration that Provides Client Connection Information (Claims 14–37)

As in Ground 1, Petitioner again treats all claims as if they rose or fell on the same generic notion of “traffic control,” but the ’722 patent does not claim a one-size-fits-all mechanism. The patent discloses distinct embodiments, and the claims draw different limitations from those embodiments. Claims 1–13 are directed to the gateway-applied routing architecture that assigns categories to update messages, assigns types to routing nodes, and uses a gateway-kept mapping from categories to node types to route the update message—with that mapping itself controlling how much update traffic flows through the routing nodes (EX1001, 18:26–29; 21:66–22:16 (Fig. 10, steps 1020→1030→1040)). By contrast, claims 14–37 add a different, client-facing registration requirement: when the client registers “for updates of the live object with the routing network,” that act of registration “provides client connection information to a node,” enabling the node to deliver subsequent update messages to the correct client (EX1001, 24:8–36).

Petitioner’s arguments never grapple with these different claim thrusts. With respect to claims 1–13, the Petition recasts Choquier’s CPU-based session dispatch, Bird’s content-handler selection, and Tsutsumitake’s single-server push as “traffic control,” but nowhere identifies the claimed gateway-held category→node-type

mapping that itself regulates update traffic through routing nodes (EX1001, 18:26–29; 21:66–22:16). And with respect to claims 14–37, the Petition never identifies—let alone maps—any disclosure in Choquier, Bird, or Tsutsumitake where client registration within a routing network provides “client connection information to a node” i.e., the node-resident connection information the node later uses to push the update to the correct client connection (EX1001, 24:8–36; 17:9–13; 20:23–38).

Petitioner’s conflation of these different embodiments leaves the claim-critical limitations unique to each set unaddressed.

A. Choquier, Bird, and Tsutsumitake still do not teach the claim-critical gateway mapping that controls update traffic through routing nodes as recited in Claims 1-13.

Claim 1 requires a specific, gateway-applied routing mechanism: messages are assigned to categories, routing nodes are assigned to types, and the gateway maintains a mapping of categories of update messages to node types; when an update message arrives, the gateway identifies the message’s category, identifies the mapped node type(s), and routes accordingly, with “the mapping [controlling] an amount of update message traffic through nodes of a routing network.” The specification states this expressly: the system assigns messages to categories, assigns nodes to types, and maintains mappings between categories and types, and “[t]he mappings allow control over the amount of traffic processed by the nodes.”

(EX1001, 18:26–29.) The gateway then performs the category→node-type determination and routing as set out in Fig. 10 (steps 1020→1030→1040). (EX1001, 21:66–22:16; Fig. 10).

1. Bird is a content-processing publish/subscribe broker—not a typed-node routing network with a mapping that controls node throughput

Bird is directed to examining message content, selecting a message-handler module, generating one or more derived messages, and delivering those derived messages to subscribers whose requirements/capabilities are met. The Abstract and the summary explain that the broker identifies message-content characteristics and uses that identification to determine which of a set of message-handler modules to pass the message to, and that processing then yields derived message(s) that are sent to subscribers based on stored requirements. (See, e.g., EX1008, ¶[0035]; claims 21–24 (“identify non-alphanumeric message content ... determine which handler module ... generate one or more derived messages ... send derived messages to subscribers.”)).

Bird further describes scanning message bytes to recognize content and then comparing the identified content with stored subscriber requirements and system capabilities, culminating in a processing-task definition and selection of a handler module to produce the derived message that will be sent to subscribers. (See EX1008, Fig. 4 (steps 180–280); claim 17 (“analysing byte sequences ... to identify

... multimedia message content”); claim 21 (steps for generating and sending derived messages.)).

Where Bird references a “distributed broker infrastructure,” it is to select among processing systems (for example, a system near a database for SQL, or a system with image-processing hardware for images) based on capabilities and identified content—i.e., handler/processor placement for content processing, not category→node-type routing of update messages in a typed routing fabric, and certainly not a mapping that controls update-traffic through nodes of a routing network. (EX1008, ¶[0037]) (distributed collection of brokers; selection by capabilities and identified content). Bird’s “selectors” choose processing modules/systems, not routing node types, and nowhere does Bird disclose a routing network that holds a category→node-type mapping or applies such a mapping to route update messages.

2. Choquier describes CPU-based session/load balancing and group-wide replication—neither is the claimed mapping-based control

Choquier’s service map lists services, servers, and each server’s CPU LOAD/CPU INDEX; gateways consult this map to choose a server to handle a client service request under “DYNAMIC LOAD BALANCING BY ROUTING OF SERVICE REQUESTS.” (EX1007, 11:2–22; 11:15–22.) Service descriptions are “used by the Gateways ... to locate servers ... for handling client service requests,”

with periodic service map broadcasts to gateways to keep them current. (EX1007, 12:33–36; 12:56–61.) For updates, an Arbiter “dispatches each update transaction to every server in the service group,” enforcing serialized replication and removing outliers when necessary. (EX1007, 21:27–50; 21:51–60). None of that is message category→node-type routing, and the replication step is the opposite of limiting node-throughput via a category→type mapping.

3. Tsutsumitake is a direct server↔client event-request push architecture—no routing network, no node types, and no gateway-kept category→node-type mapping

Tsutsumitake teaches that the client detects an event request in page content and causes the event request to be sent; the server then sends event notifications back over a maintained, per-client connection, with the client able to cut or resume that connection when navigating. (EX1006, 10:56–67; 4:22–29). In managing those server-side pushes, Tsutsumitake describes that, if the client is “not registered,” the server stores a pair of IDs (event ID, client ID) in a connection management unit. (EX1006, 11:45–55). That server-side “client ID” in a single-server push model is not the claim’s gateway-applied category→node-type routing and is not a mapping that controls update-message traffic through routing nodes. Nor does Tsutsumitake disclose any routing network of typed nodes or a gateway that holds and applies a category→node-type mapping. The architecture is direct client/server push, not

gateway-routed updates through node types. (See also EX1006, 12:1–17; 12:33–63; 21:24–43.

4. The combination still fails because the claimed mapping mechanism is nowhere to be found

Nothing in Bird, Choquier, or Tsutsumitake discloses or suggests the gateway-kept, category→node-type mapping—and the use of that mapping to route update messages—that Claim 1 requires. Bird chooses processing handlers and emits derived messages in a publish/subscribe system, including optionally distributing broker functions across processing systems according to capabilities and identified content, but it is not a routing network and does not control node throughput via any category→type mapping held in a routing network. (EX1008, ¶[0037]; claims 21–24). Choquier chooses servers for client sessions via CPU metrics and replicates updates to all servers in a group; it neither recognizes message “categories (based on input source)” nor assigns “node types,” and it certainly does not use a gateway mapping to limit which nodes must process a given update. Tsutsumitake operates as a direct server-push system and adds no gateway-held category→node-type mapping or typed-node routing to the combination.

There is no articulated reason to combine these disparate systems in a way that would yield the claimed architecture without fundamentally altering their operation. Bird’s core utility lies in content analysis and transformation for

subscribers; Choquier's in CPU-based server selection and consistency via replication; Tsutsumitake's in client-driven event requests and server push over a single connection. Re-engineering them to implement a gateway-maintained category→node-type mapping that controls update traffic through routing nodes is not a predictable combination of prior-art elements; it is a reconstruction no reference proposes and that runs counter to Choquier's choice to replicate each update to every server in a group.

As a result, swapping in Bird, while reusing Choquier and Tsutsumitake, does not cure the central deficiency. No reference discloses a mapping from message categories to node types held in the routing network that is applied to route the update message and thereby control the amount of update-message traffic through routing nodes. The record still lacks the claimed mapping called out in the '722 patent. (EX1001, 18:26–29; 21:66–22:16). Claims 1–13 are therefore not shown obvious under Ground 3.

B. The combination does not disclose “registering ... provides client connection information to a node” (claims 14/20/26/32)

As with Ground 1, Petitioner never joins issue with the element actually recited in the later independent claims. Claims 14/20/26/32 do not merely repeat the mapping from message categories to node types held in the routing network of claims 1–13; they add a different, client-facing requirement: when a client registers

“for updates of the live object with the routing network,” that very act “provides client connection information to a node,” enabling the node to route subsequent update messages to the correct client connection (EX1001, 24:8–36).

1. Ground 3 Fails to Show that Client Registration Provides Client Connection Information to a Node

Claim 14 (and its counterparts 20/26/32) require more than a client merely expressing interest in updates. The claim language ties registration with the routing network to a specific consequence: “registering the client device with the routing network provides client connection information to a node” so that subsequent update messages can be routed from that node to the correct client connection (EX1001, 24:8–36).

Nothing in Petitioner’s Ground 3 combination of Choquier (EX1007), Bird (EX1008), and Tsutsumitake (EX1006) discloses or suggests that the claimed registration provides client connection information to a node. In Choquier, “registration” in the sense relevant here does not occur at all. Choquier’s gateways do not collect client connection information for later node-side delivery; they read a service map that lists services/servers with CPU LOAD/CPU INDEX and pick a server to handle a client service request under “DYNAMIC LOAD BALANCING BY ROUTING OF SERVICE REQUESTS” (EX1007, 11:2–22; 11:15–22; 12:33–36; 12:56–61). When updates occur, Choquier’s Arbiter “dispatches each update

transaction to every server in the service group,” enforcing a serialized order and removing outliers (EX1007, 21:27–50; 21:51–60). None of this has anything to do with a client device registering with a routing network in a manner that provides client connection information to a node to use later for update delivery.

Bird likewise does not supply the missing limitation. Bird’s broker analyzes message content, selects a handler module, generates derived messages, and sends those derived messages to subscribers whose stored requirements/capabilities are satisfied (EX1008, ¶[0035]; Fig. 4 (steps 180–280); claims 17, 21–24). Even where Bird discusses a distributed broker infrastructure, it is about choosing among processing systems based on capabilities and identified content (e.g., SQL near a database; images to a system with image-processing hardware)—that is handler/processor placement, not node-type routing in a mapping-controlled routing fabric (EX1008, ¶[0037]). Bird’s repository of subscriber “requirements/capabilities” (see, e.g., EX1008, ¶[0039]–[0042]) is not node-resident client connection information provided by client registration with a routing network, and Bird nowhere teaches a node that stores socket identifiers (or other client connection information) to later deliver update messages.

Tsutsumitake confirms the point from the other direction. Its architecture is direct server↔client event-request push, not a multi-node routing network. The page receiving unit 112 on the client detects an event request embedded in the page and

causes an event request to be sent; the server then pushes event notifications back over a maintained, per-client connection, and the client can cut or resume the connection on navigation (EX1006, 10:56–67; 4:22–29; see also 12:1–17; 12:33–63; 21:24–43). If the client is “not registered,” the server records a pair of (event ID, client ID) in a connection management unit (EX1006, 11:45–55). A server-side “client ID” in a single-server push model is not the claims’ “client connection information to a node” in a routing network—and Tsutsumitake discloses no nodes, no node types, and no gateway-kept category→node-type mapping. Even if one were to place Tsutsumitake’s “client ID” in a routing network, the “client ID” at best identifies “client information.” However, the Petition fails to allege -- let alone demonstrate -- that the “client ID” expressly or inherently disclose “client connection information,” such as “a socket identifier or some other information for identifying how to route the message to the named entity.” EX1001, 20:27-30. Finally, even if one were to re-label Tsutsumitake’s server as a “node,” the cited registration is server-local account/consumer identification, not the node-resident connection pointer (e.g., a socket identifier) that the ’722 patent requires the registration to provide (EX1001, 17:9–13; 20:23–38; 24:8–36).

Nor can the deficiency be cured “in combination.” Choquier provides no client-to-node registration channel and no node-side storage of client connection information; Bird provides no routing-network node or node-resident connection

state, only subscriber requirements/capabilities used for content processing; and Tsutsumitake provides server-local (event ID, client ID) pairs in a direct push model. None of these disclosures—alone or together—teaches registration with a routing network that, as part of that act, stores node-resident client connection information that the node later uses to route an update message to the correct client connection (EX1001, 24:8–36). The “client connection information” required by the claims is absent from the combination.

2. Petitioner’s reliance on Tsutsumitake’s (event-ID, client-ID) pair does not satisfy the claims’ “client connection information to a node”

Petitioner asserts that “connection information is also provided as part of the registration request,” pointing to Tsutsumitake’s statement that the server checks whether a pair consisting of the event ID and the client ID is already registered and, if not, registers the pair in a “connection management unit.” (Petition, 74) That reading misconstrues both Tsutsumitake and the ’722 claim language.

First, the claims require that the act of registering with the routing network “provides client connection information to a node” so that the node can later route a subsequent update message to the correct client connection. By contrast, Tsutsumitake is a single-server, direct event-request push architecture. A page component on the client detects an event request and causes that request to be sent; the server then pushes event notifications back over a maintained, per-client

connection, and the client may cut or resume that connection on navigation. (EX1006, 10:56–67; 4:22–29; see also 12:1–17; 12:33–63; 21:24–43.) In that context, the “connection management unit” merely stores a logical pair—(event ID, client ID)—to record that a particular client wishes to receive events of a particular type. (EX1006, 11:45–55). This is a server-local subscription state, not node-resident connection information as used in the ’722 Patent . Moreover, as explained above, Tsutsumitake’s “client ID” at best identifies “*client information*.” However, the Petition fails to allege -- let alone demonstrate -- that this “client ID” expressly or inherently disclose “*client connection information*,” such as “a socket identifier or some other information for identifying how to route the message to the named entity.” EX1001, 20:27-30.

Second, the claims require registration “with the routing network” that provides...to a node the connection information. Tsutsumitake contains no routing network, no node types, and no gateway-kept category→node-type mapping; it is a single-server push model. (EX1006, 10:56–67; 4:22–29.) The very focus and purpose of the “registration” are different: Tsutsumitake’s pair is used by the server to correlate a client and an event type, not to equip a routing-network node with information about the connection to the client. (EX1001, 24:8–36).

Third, Petitioner’s characterization—“connection information is also provided as part of the registration request”—does not appear in EX1006. What

EX1006 says is that the server asks whether the (event ID, client ID) pair has already been registered, and if not, the server registers the pair. (EX1006, 11:45–55.) The passage describes server-side subscription bookkeeping, not a client’s provision of client connection information to a routing-network node as part of registration. That distinction matters under the claim language.

Finally, to the extent that Petitioner is suggesting that a “client ID” inherently constitutes the claimed “client connection information,” this suggestion fails as a matter of law and fact. The passage on which Petitioner relies in Tsutsumitake merely says that the server will store a pair (event ID, client ID) when a client is “not registered” (EX1006, 11:45–55). Petitioner offers no explanation as to why a client ID is anything other than an identity label. Further, the Petition doesn’t even explain how the Client ID is client connection information as required by the claims.

As a legal matter, inherency cannot be established by “probabilities or possibilities.” The Federal Circuit has long held that a missing limitation is inherent only if it is “necessarily present” in the prior art disclosure, not if it “may” or “probably” is present. *In re Robertson*, 169 F.3d 743, 745 (Fed. Cir. 1999) (quoting *In re Oelrich*, 666 F.2d 578, 581 (C.C.P.A. 1981)); *Continental Can Co. USA, Inc. v. Monsanto Co.*, 948 F.2d 1264, 1268 (Fed. Cir. 1991) (inherent feature must be “necessarily present” in the prior art structure). That standard applies with equal, and often greater, force when inherency is invoked to fill gaps in an obviousness theory

under § 103: inherency “may not be established by probabilities or possibilities” and may supply a missing limitation only when the property or feature is necessarily present or the natural result of the combination of prior-art elements. *Par Pharm., Inc. v. TWI Pharms., Inc.*, 773 F.3d 1186, 1195–96 (Fed. Cir. 2014) (reversing obviousness where the alleged inherent feature was not shown to necessarily result from the combination). See also *Honeywell Int’l Inc. v. Mexichem Amanco Holding S.A. de C.V.*, 865 F.3d 1348, 1354–56 (Fed. Cir. 2017) (cautioning that inherency cannot be a shortcut for missing limitations in § 103 absent necessity).

Here, Petitioner offers no evidence—let alone the required showing of necessity—that a client ID in Tsutsumitake’s server-local table is, or must be, the claimed “client connection information.” At most, Petitioner posits that a system could look up a connection from a client ID. That is precisely the kind of “mere possibility” that cannot carry the day.

As a result, Petitioner’s reliance on Tsutsumitake does not cure the missing limitation. The ’722 Patent’s claims require that registration with the routing network provides client connection information to a node, thereby equipping the node to route subsequent updates to the correct client connection. Tsutsumitake, by contrast, is a single-server push model in which the server maintains a local subscription table keyed by event ID and client ID. That server-local pairing is not node-resident client connection information, is not provided to a node in a routing network, and does not

enable the routing of update messages through nodes based on client registration. Because Petitioner has not shown that Tsutsumitake necessarily discloses or results in the claimed “client connection information,” the inherency theory fails. Absent this critical element, Ground 3 cannot meet its burden with respect to the independent claims.

C. The dependent claims (2–6, 8–13, 15–19, 21–25, 27–31, 33–37) are patentable at least by virtue of their dependency.

Ground 3 challenges all claims 1–37, but the combination of Choquier, Bird, and Tsutsumitake does not render any of the independent claims obvious. As shown in our Ground-3 analysis, Petitioner never supplies the claim-critical (i) gateway-kept mapping of categories of update messages to node types that is used to route the update and thereby controls how much update traffic flows through routing nodes (independent claims 1/7) (EX1001, 18:26–29; 21:66–22:16), nor (ii) the later-claim requirement that “registering the client device with the routing network provides client connection information to a node” (independent claims 14/20/26/32) (EX1001, 24:8–36; see also 17:9–13; 20:23–38). Nothing in Choquier—which performs CPU-based session/request dispatch from a service map and replicates each update to every server in the service group—teaches category→node-type routing of update messages (EX1007, 11:2–22; 12:33–36; 12:56–61; 21:27–50; 21:51–60). Bird is a content-aware broker that selects handler modules and emits

derived messages based on subscriber capabilities; its “selectors” choose processing systems by capabilities/identified content, not typed routing nodes (EX1008, ¶[0035]; Fig. 4 (steps 180–280); claims 17, 21–24; ¶[0037]). Tsutsumitake is a single-server event-request/push model with no routing network, no node types, and no gateway mapping, and its server-local (event ID, client ID) pair is not node-resident connection information provided by registration with a routing network (EX1006, 4:22–29; 10:56–67; 12:1–17; 12:33–63; 21:24–43). Because Petitioner has not met its burden on claims 1, 7, 14, 20, 26, and 32, the challenges to all dependent claims—2–6, 8–13, 15–19, 21–25, 27–31, and 33–37—fail at least by virtue of their dependency, as each dependent necessarily includes all limitations of its parent independent claim.

In any event, Petitioner also does not carry its burden on the additional limitations of these dependents. By way of example only: claims 2/8 (method/apparatus) require a node to extract an object ID from the update message and use it in node-side determinations—nowhere taught by CPU-load dispatch, endpoint registries, or single-server push (EX1001, 21:66–22:16; EX1007, 11:2–22; EX1008, ¶[0035]; EX1006, 10:56–67). Claims 3/9 require the client-proxy routing path (node→proxy→client) that the cited art does not disclose (EX1001, 19:20–33; 20:23–38). Claims 4/10 and 5/11 specify where “client registration information concerning the client connection” resides—at the node versus at the proxy—and the

combination nowhere teaches node- or proxy-resident, connection-level delivery information enabling later node→client transmission (EX1001, 17:9–13; 20:23–38). Claims 6/12 add the live-object property semantics and (for the apparatus) a gateway that routes to a node adapted to receive messages of more than one message category; again, Petitioner identifies no “message categories” or “node types,” much less a node adapted to receive multiple categories (EX1001, 18:26–29; 21:66–22:16; EX1007, 11:2–22; 12:33–36). Claims 15–19 (and their apparatus/article/CRM counterparts 21–25, 27–31, 33–37) add the activation-module behavior that executes at the client to register live objects and, in certain embodiments, determine the message category and/or node type and cause registration accordingly—a client-side function absent from Choquier/Bird/Tsutsumitake (EX1001, 19:20–33; 20:24–33; 24:49–62). Thus, even apart from the dispositive failure on the independent claims, Petitioner has not shown where the prior art teaches the additional limitations layered by each dependent claim. All dependent claims challenged in Ground 3 therefore remain patentable at least by virtue of their dependency, and on their own added limitations.

IX. Ground 4: Adding Herz to Choquier, Bird, and Tsutsumitake still does not supply the missing limitations of the independent claims and therefore the challenged dependents stand at least by virtue of their dependency

Petitioner’s Ground 4 swaps in Herz alongside Choquier/Bird/Tsutsumitake, but—as with Ground 3—Herz does not provide either of the claim elements missing from Petitioner’s case on the independent claims. First, nothing in the four-reference combination discloses a mapping of categories of update messages to node types maintained in the routing network that is used to route the updates and thereby controls how much update traffic flows through routing nodes (claims 1/7). Choquier’s “routing” is CPU-based session/request dispatch using a service map with CPU LOAD/CPU INDEX, and its update path involves replication to every server in the service group (EX1007, 11:2–22; 11:15–22; 12:33–36; 12:56–61; 21:27–50; 21:51–60). Bird is a content-aware broker that analyzes message content, selects handler modules, and emits derived messages for subscribers; its “selectors” choose processing systems based on capabilities/identified content, not typed routing nodes (EX1008, ¶[0035]; Fig. 4; claims 17, 21–24; ¶[0037]). Tsutsumitake is single-server push via a maintained per-client connection and contains no routing network, no node types, and no gateway mapping (EX1006, 4:22–29; 10:56–67; 12:1–17; 12:33–63; 21:24–43). Herz likewise does not supply the missing mapping: Herz selects core servers for topical groups, constructs multicast trees, and

broadcasts files (not per-message updates) to all core servers; its “mappings” are content/topic→core server(s) for replication/caching, not category→node-type mappings in a routing network that throttle update traffic through routing nodes (EX1009, 49:19–27; 49:31–46; 50:21–37; 51:1–16; 51:16–28; 53:1–8; 53:16–38; 77:7–18; 78:35–43). In short, the quartet still lacks the claimed category→node-type routing rule.

Second, nothing in the combination discloses that client registration with the routing network “provides client connection information to a node” (claims 14/20/26/32). Choquier has no client-to-node registration channel and no node-side connection state; it is service-map-based session dispatch (EX1007, 11:2–22; 12:33–36; 12:56–61). Bird’s subscriber requirements/capabilities repository is not node-resident connection information for update delivery; Bird does not even teach a routing-network node that stores such data (EX1008, ¶[0035]; ¶[0039]–[0042]). Tsutsumitake’s “registration” is server-local bookkeeping of a (event ID, client ID) pair in a connection management unit—an identity label, not the client connection information the claims require to be provided to a node at registration (EX1006, 11:45–55; 10:56–67). Herz adds no client-registration-to-node connection information either; its records concern user interest summaries, proxy-mediated/pseudonymous participation, and which servers belong on which multicast

tree, not client connection information supplied by registration for later update routing (EX1009, 49:37–46; 50:1–6; 51:13–28; 53:1–38; 77:7–18; 78:35–43).

Because Herz does not remedy either missing limitation in the independent claims, Ground 4 still fails to show those independent claims obvious. It follows that dependent claims 3, 5, 9, 11, 12, 15, 21, 27, and 33, challenged only on the same defective theory, are patentable at least by virtue of their dependency on allowable independent claims.

X. Conclusion

For all four grounds, Petitioner never engages with—let alone carries its burden on—the claim-critical that architecture the '722 Patent actually claims.

Independent Claims 1 and 7 require gateway-applied routing “based on a mapping of categories of update messages to node types,” with “the mapping” itself controlling how much update traffic flows through routing nodes. Petitioner’s combinations substitute generic “traffic reduction” narratives for the specific, gateway-kept category→node-type mapping the claims and specification require. Lawson is endpoint selection via registries; Choquier is CPU-based session dispatch plus group-wide update replication; Bird is content processing and handler selection; and Herz is multicast/broadcast to core servers. None discloses, and none reasonably suggests, the claimed mapping-based control of update-message throughput through typed routing nodes.

Independent claims 14/20/26/32 add a different and equally dispositive requirement: client registration with the routing network must “provide[] client connection information to a node.” The Petition never even identifies what in the prior art constitutes that “client connection information,” much less maps it with particularity, and the cited references do not supply it. Lawson’s registries identify servers and local consumers, not node-resident client-connection handles; Choquier has no client→node registration at all; and Tsutsumitake’s (event-ID, client-ID) bookkeeping in a single-server push model is not node-resident connection information provided by registration with a routing network. Petitioner’s ipse dixit cannot cure missing elements or re-engineer these systems after the fact.

Because Petitioner fails to show a reasonable likelihood of prevailing on any independent claim, every challenged dependent claim stands at least by virtue of its dependency—and, in any event, for additional limitations the combinations likewise do not teach (including node-side object-ID extraction and determinations, the proxy delivery path, the locus of connection-registration information at the node or proxy, and client-side activation/registration behaviors). The Board may not supply missing theories or accept new mappings for the first time in reply to patch these foundational gaps.

Accordingly, the Board should deny institution under 35 U.S.C. § 314(a) as to all challenged claims.

Respectfully submitted,

Dated: September 15, 2025

By: /Brandon R. Theiss/

Brandon R. Theiss

Registration No. 70,507

*Counsel for Patent Owner
Intellectual Ventures II LLC*

CERTIFICATE OF COMPLIANCE WITH WORD COUNT

Pursuant to 37 C.F.R. § 42.24(d), I certify that this **Patent Owner Preliminary Response** complies with the type-volume limits set forth in the Memorandum and 37 C.F.R. § 42.24 because it contains 13,850 words, excluding the parts of the Brief that are exempted by 37 C.F.R. § 42.24(a), according to the word processing system used to prepare this Brief.

Respectfully submitted,

Dated: September 15, 2025

By: /Brandon R. Theiss/
Brandon R. Theiss
Registration No. 70,507

*Counsel for Patent Owner
Intellectual Ventures II LLC*

CERTIFICATE OF SERVICE

The undersigned certifies that pursuant to 37 C.F.R. § 42.6(e), a copy of the foregoing **PATENT OWNER'S PRELIMINARY RESPONSE** was served via email (as consented to by counsel) on September 15, 2025 to lead and backup counsel of record for Petitioner as follows:

Keith D. Harden
kharden@munckwilson.com
Munck Wilson Mandala, LLP
2000 McKinney Ave., Ste. 1900
Dallas, TX 75201

Daniel E. Venglarik
dvenglarik@munckwilson.com
Munck Wilson Mandala, LLP
2000 McKinney Ave., Ste. 1900
Dallas, TX 75201

S. Wallace Dunwoody
wdunwoody@munckwilson.com
Munck Wilson Mandala, LLP
2000 McKinney Ave., Ste. 1900
Dallas, TX 75201

Michael C. Wilson
mwilson@munckwilson.com
Munck Wilson Mandala, LLP
2000 McKinney Ave., Ste. 1900
Dallas, TX 75201

John B. Campbell
jcampbell@McKoolSmith.com
McKool Smith, P.C.
303 Colorado Street
Suite 2100

Austin, TX 78701

Casey Shomaker
cshomaker@McKoolSmith.com
McKool Smith, P.C.
300 Crescent Court
Suite 1200
Dallas, TX 75201

Alan Block
ablock@McKoolSmith.com
McKool Smith, P.C.
300 South Grand Avenue
Suite 2900
Los Angeles, CA 90071

SWA-IV@munckwilson.com

AA_Intellectual_Ventures@mckoolsmith.com

Dated: September 15, 2025

By: /Brandon R. Theiss/
Brandon R. Theiss
Registration No. 70,507

*Counsel for Patent Owner
Intellectual Ventures II LLC*