UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

CISCO SYSTEMS, INC.,
Petitioner,

v.

NETFUEL, INC.,
Patent Owner.

Patent No. 9,663,659
Filing Date: October 10, 2012
Issue Date: May 30, 2017
MANAGING COMPUTER NETWOK RESOURCES

PATENT OWNER'S NOTICE OF APPEAL

Case No. IPR2019-00992

Pursuant to 35 U.S.C. §§ 141 and 142 and 37 C.F.R. §§ 90.2 and 90.3, Patent Owner Netfuel, Inc. ("Netfuel" or "Patent Owner") hereby provides notice that it appeals to the United States Court of Appeals for the Federal Circuit from the Final Written Decision entered November 5, 2020 (Paper 41) and from all underlying orders, decisions, rulings, and opinions regarding U.S. Patent No. 9,663,659 (the "'659 patent") in Case No. IPR2019-00992. This notice is timely under 37 C.F.R. § 90.3, having been filed within 63 days after the date of the Final Written Decision. For the limited purpose of providing the Director with the information requested in 37 C.F.R. § 90.2(a)(3)(ii), Patent Owner anticipates that the issues on appeal may include, but are not limited to: the Board's claim constructions, its application of those constructions, its obviousness determinations including that claims 1–3, 6-9, 13-15, and 18 of the '659 patent are unpatentable under 35 U.S.C. § 103; the findings, rulings and conclusions supporting or relating to those determinations; the constitutionality of the appointments of Administrative Patent Judges Karl D. Easthom, Kalyan K. Deshpande, and Brian J. McNamara under U.S. Const. art. II, § 2, cl. 2. in view of *Arthrex v. Smith & Nephew*, No. 18-2140 (Fed. Cir. 2019); and any other issues decided adversely to Patent Owner in any orders, decisions, rulings, or opinions in IPR2019-00992.

Simultaneous with this submission, a copy of this Notice of Appeal is being filed with the Clerk of the United States Court of Appeals for the Federal Circuit

and being submitted electronically through the Court's CM/ECF system, together with the requisite fee in the amount of $500.00. In addition, a copy of this Notice of Appeal is being filed with the Patent Trial and Appeal Board and served upon counsel of record for Cisco Systems, Inc.

Respectfully submitted,

Dated: January 5, 2021

/Vincent J. Rubino, III /

Vincent J. Rubino, III (Reg. No. 68,594)
Lead Counsel for Patent Owner
FABRICANT LLP
230 Park Avenue, 3rd Floor W.
New York, NY 10169
Telephone: 212-257-5797
Facsimile: 212-257-5796
Email: vrubino@fabricantllp.com

CERTIFICATE OF SERVICE

A copy of Patent Owner's Notice of Appeal and related documents are being served via email on Petitioner's attorneys of record on the 5th day of January, 2021, the same day as the filing of the above-identified document in the United States Patent and Trademark Office/Patent Trial and Appeal Board, by serving Petitioner's correspondence address of record with the USPTO as follows:

May Eaton
PERKINS COIE LLP
Email: may.eaton@perkinscoie.com; CiscoNetFuel@perkinscoie.com

Reza Dokhanchy
KIRKLAND & ELLIS LLP
Email: reza.dokhanchy@kirkland.com; Cisco-NetfuelIPR@kirkland.com

Eugene Goryunov
Dina Blikshteyn
HAYNES AND BOONE, LLP
Email: eugene.goryunov.ipr@haynesboone.com;
dina.blikshteyn.ipr@haynesboone.com

Dated: January 5, 2021

By: /*Vincent J. Rubino, III*/
Vincent J. Rubino, III (Reg. No. 68,594)
Lead Counsel for Patent Owner
FABRICANT LLP
230 Park Avenue, 3rd Floor W.
New York, NY 10169
Telephone: 212-257-5797
Facsimile: 212-257-5796
Email: vrubino@fabricantllp.com

Trials@uspto.gov
571-272-7822

Paper 41
Date: November 5, 2020

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

CISCO SYSTEMS, INC.,
Petitioner,

v.

NETFUEL, INC.,
Patent Owner.

IPR2019-00992
Patent 9,663,659 B1

Before KARL D. EASTHOM, KALYAN K. DESHPANDE, and BRIAN J. MCNAMARA, *Administrative Patent Judges*.

EASTHOM, *Administrative Patent Judge*.

JUDGMENT
Final Written Decision
Determining All Challenged Claims Unpatentable
35 U.S.C. § 318(a)

I. INTRODUCTION

Cisco Systems, Inc. ("Petitioner") filed a Petition pursuant to 35 U.S.C. §§ 311–319 to institute an *inter partes* review of claims 1–3, 6–9, 13–15, and 18 of U.S. Patent No. 9,663,659 B1 (Ex. 1003, the "'659 patent"). Paper 2 ("Petition" or "Pet."). NetFuel, Inc. ("Patent Owner") filed a Preliminary Response. Paper 7 ("Prelim. Resp."). The Board instituted an *inter partes* review of the challenged claims pursuant to 35 U.S.C. § 314. Paper 10 ("Inst. Dec.").

After institution, Patent Owner filed a Patent Owner Response (Paper 22, "PO Resp."); Petitioner filed a Reply (Paper 27); and Patent Owner filed a Sur-reply (Paper 30, "Sur-reply" or "PO Sur-reply"). Thereafter, the parties presented oral arguments via a video hearing (August 12, 2020), and the Board entered a transcript into the record. Paper 40 ("Tr.").

For the reasons set forth in this Final Written Decision pursuant to 35 U.S.C. § 318(a), we determine that Petitioner demonstrates by a preponderance of evidence that the challenged claims are unpatentable.

A. Real Party-In-Interest

Petitioner identifies Cisco Systems, Inc. as the real party-in-interest. Pet. 2.

B. Related Proceedings

Petitioner identifies the following pending district court litigation involving the '659 patent: *NetFuel, Inc. v. Cisco Systems, Inc.*, No. 5:18-cv-02352-EJD (N.D. Cal.). Pet. 2–3; *see also* Paper 4, 2.

C. The ’659 Patent

The ’659 patent, titled “Managing Computer Network Resources,” “pertains to the management of computer networks.” Ex. 1003, code (54), 2:11–12. The ’659 patent describes prior art systems as managing computer networks “to ensure smooth and efficient operation,” including ensuring robustness, quality of service (QoS), and scalability. *Id.* at 1:14–19. However, these prior art systems typically involve humans, which “is undesirable, particularly, for example, in the case of a network having a large number of nodes because of the time it would take a human to identify and fix a failed node.” *Id*. at 1:20–24.

The ’659 patent explains “[i]t is therefore desirable that networks run themselves as much as possible,” and proposes a method of managing a computer network using software agents that “operate within an agent runtime environment (. . . ‘ARE’)[,] which is hosted on a particular network (host) device,” where “[e]ach ARE allows agents operating therein to communicate with the host device, with other agents, and with an agent control mechanism.” Ex. 1003, 1:25–26, 2:13–20.

With reference to Figure 1 of the ’659 patent, reproduced below, a vertically layered system 10 for managing a network comprises an ARE 12 loaded on a host platform 14 defined on a network device. Ex. 1003, 2:27–28, Fig. 1.

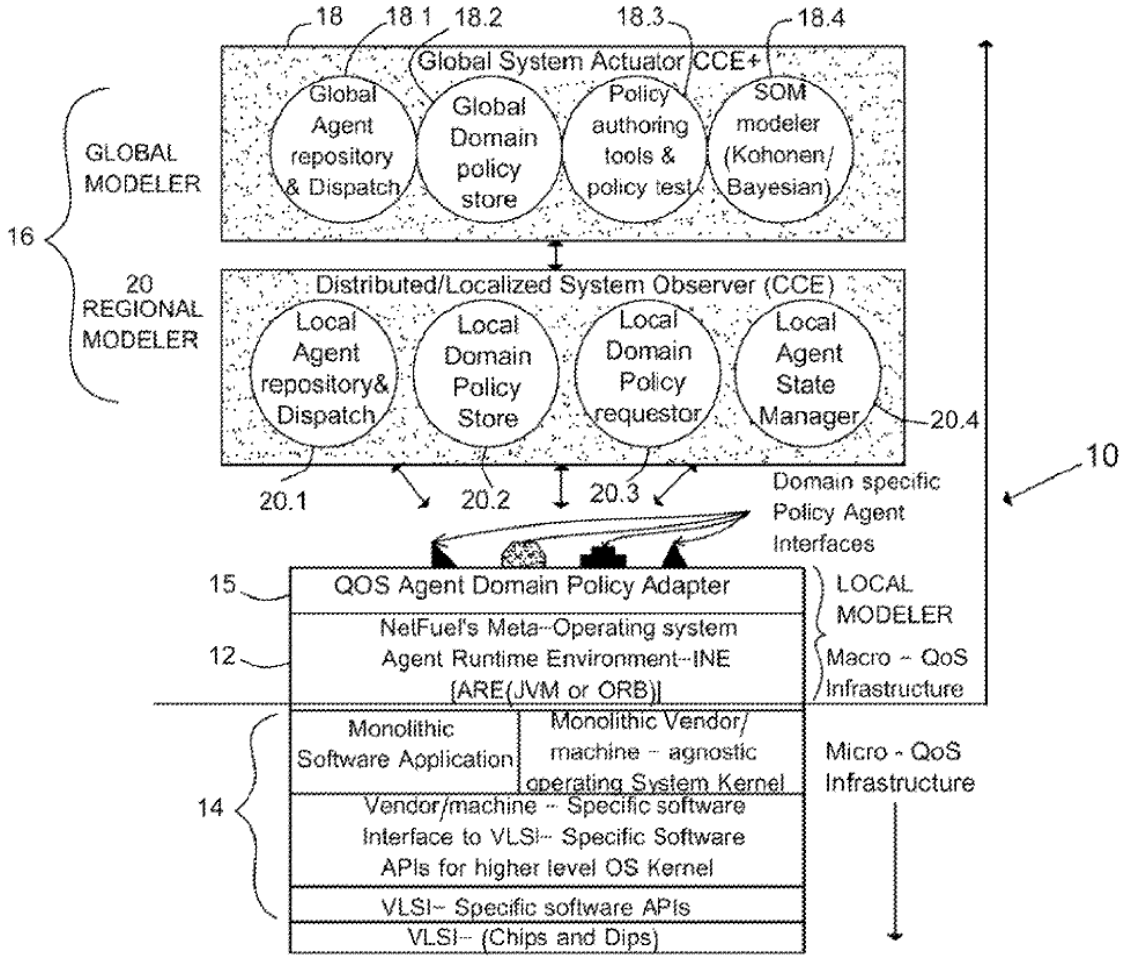

FIG.1

Figure 1 illustrates a schematic drawing of vertically layered system 10 for managing a network, including agent control mechanism 16 as a global modeler, with ARE 12 that includes agents layered on top of monitored device 14. Ex. 1003, 1:30–31, 2:11–44. In general, system 10 manages a large network comprising a number of network devices 14, each hosting an ARE. *Id*. at 2:51–53.

Agents operating within AREs 12 perform discovery, monitoring, and policy enforcement functions, and each ARE 12 provides a uniform context and set of services upon which agents can rely and/or acts as an operating

system for the agents. Ex. 1003, 2:29–31, 4:28–30. AREs 12 also can create agents, suspend operation of the agents, and terminate the agents. *Id*. at 2:53–54. Each agent has an "assigned goal which is expressed in the form of policy and which can be dynamically modified based on desired operational characteristics of the network," where a "[p]olicy may be seen as the desired operational characteristics of a network on which system 10 is installed." *Id*. at 2:20–23, 59–61. The agents monitor various operational parameters of system 10 and apply a corrective policy to change these operational parameters if they deviate from established limits. *Id*. at 3:40–44.

Each ARE uses a proxy thread manager to control a "master thread group that starts, stops and resets agents within its control." Ex. 1003, 6:35–36. Each ARE, using the thread group, prioritizes, boot straps, suspends, and renders active, agents in its environment "since an agent does not need to be active until required." *See id.* at 6:38–62. "[E]ach agent upon instantiation is assigned a thread priority level by the ARE . . . ." *Id.* at 11:20–21. "Each agent can be pacified, by the proxy thread manager, and [its] thread made available to active agents in the event of such pacification." *Id.* at 6:51–54. "[E]ach ARE has the ability to provide feedback on the per-thread utilization of each thread running within it. . . . to, for example, help determine load balancing on the AREs" (by CCE- 20). *Id.* at 11:27–30. A proxy thread manager subsystem administers thread utilization and acts as a thread controller and mediator for interaction between agents and threads. *Id.* at 6:48–51. "Each agent executes its own thread, which is separate from the main thread of the ARE and any other agents." *Id.* at 7:15–17. "AREs 12 load the agent code, and any code the agent code is dependent on, into

memory." *Id.* at 6:66–67. Different types of agents include static, mobile, complex, intelligent, or simple agents. *Id.* at 7:51–8:48. "Agents are used to enforce specific actions/behavior on hardware/software devices controlled by the system 10." *Id.* at 11:32–33.

"A monitoring agent may monitor more than one feature on a device. . . . In some embodiments, the monitoring agents comprise two separate agents running sending information between each other or it may be physically the same monitoring agent acting as both sender/[l]istener." Ex. 1003, 16:19–22.

Agents operating within an ARE can monitor network characteristics and make information available to a "global modeler." Ex. 1003, 2:29–3:5. In system 10 of Figure 1, the uppermost layer includes an agent control mechanism 16 which comprises the global modeler (coherent computing entity CCE^+ layer 18) on top of a regional modeler (CCE^- layer 20). *Id*. at 2:32–36. Policies are input into the global modeler (CCE^+ layer 18) and then disseminated through the various layers until they reach the various agents seized with the task of implementing each policy. *Id*. at 2:66–3:2. The agents additionally monitor various operational characteristics of the network devices, and report these characteristics to CCE^+ layer 18, which models network behavior to write optimal network policies. *Id*. at 3:3–15.

Figure 15 of the ’659 patent follows:

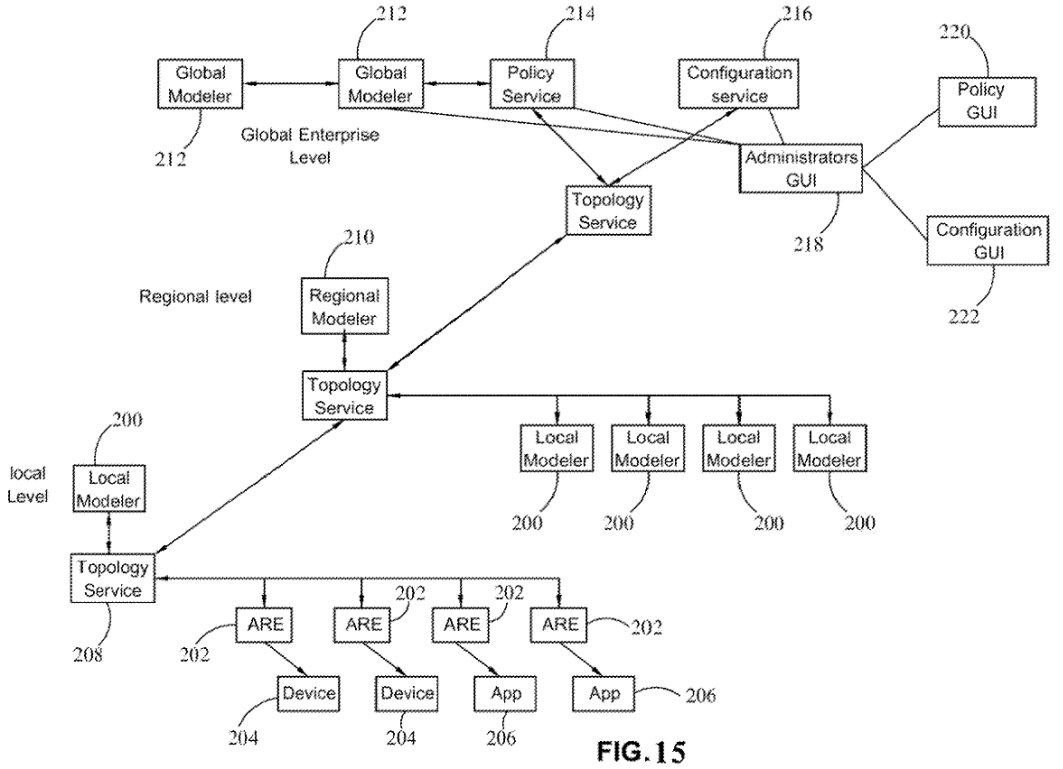

FIG. 15

Figure 15 illustrates the hierarchy of global and regional modelers within system 10, including local AREs 202 monitoring devices 204 or applications 206 on the lowest level. *Id.* at 20:35–38. The modeler hierarchy also includes a regional level defined by regional modeler 210, which controls a number of local modelers 200, and a global level, which defines a global modeler 212 communicating with a policy service 214 and coordinating overall policy interaction and control in system 10. *Id.* at 20:36–56, Fig. 15.

D. The Challenged Claims

Petitioner challenges claims 1–3, 6–9, 13–15, and 18 of the ’659 patent. Pet. 1. Claims 2, 3, and 6 depend from independent claim 1. Claims 8 and 9 depend from independent claim 7. Claims 14, 15, and 18 depend

from independent claim 13. Independent claim 7, a "computer-readable medium" claim, and independent claim 13, a "system" claim, each mirror the limitations of independent claim 1, a "method" claim. Claim 1, reproduced below, illustrates the subject matter of the challenged claims:

> 1. [a] A computer-implemented method, comprising:
>
> [b] running at least one thread in a first runtime environment;
>
> [c] monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;
>
> [d] detecting if there is an abnormality in the monitored operational parameters; and
>
> [e] performing a corrective action to fix any detected abnormalities,
>
> [f] wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,
>
> [g] wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,
>
> [h] wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.

Ex. 1003, 36:26–49 (brackets [a]–[h] added by Petitioner).[1]

[1] *See, e.g.*, Pet. 35–58; Ex. 1001 ¶¶ 106–132.

E. Asserted Grounds of Unpatentability

Petitioner asserts the following grounds of unpatentability:

Claims Challenged	35 U.S.C. §	References
1–3, 7–9, 13–15	103(a)[2]	Lindskog[3], Lindskog II[4], Lindskog III[5], Turek[6]
6, 18	103(a)	Lindskog, Lindskog II, Lindskog III, Turek, Dijkstra[7]

Pet. 5, 35, 67.[8] Petitioner relies on the Declaration of Dr. Kevin Almeroth, (Ex. 1001). Patent Owner relies on the Declaration of Vijay Madisetti, Ph.D. (Ex. 2001).

---

[2] Because the effective filing date of the challenged claims pre-dates March 16, 2013, the 35 U.S.C. § 103 provisions of the Leahy-Smith America Invents Act ("AIA"), Pub. L. No. 112-29, §§ 3(b)–3(c), 3(n)(1), 125 Stat. 284, 285–87, 293 (2011) do not apply. *See* Ex. 1003, code (63) (listing continuation application filing date Jan. 31, 2012), Certificate of Correction (listing effective filing date June 28, 2002 through a string of continuation applications).

[3] Lindskog et al., U.S. Patent No. 6,665,262 B1, issued Dec. 16, 2003, filed Feb. 16, 1999 (Ex. 1007).

[4] Lindskog et al., U.S. Patent No. 6,370,572 B1, issued Apr. 9, 2002 (Ex. 1008).

[5] Newcombe et al., U.S. Patent No. 6,349,325 B1, issued Feb. 19, 2002 (Ex. 1010). For convenience, we maintain Petitioner's nomenclature and refer to U.S. 6,349,325 B1 as "Lindskog III" (*see* Pet. 5), even though U.S. 6,349,325 B1 does not list "Lindskog" as one of the inventors (as noted by Patent Owner, *see* Prelim. Resp. 17).

[6] Turek et al., U.S. Patent No. 6,460,070 B1, issued Oct. 1, 2002, filed June 3, 1998 (Ex. 1012).

[7] "Self-stabilizing Systems in Spite of Distributed Control," Communications of the ACM, Edsger Dijkstra (Nov. 1974) (Ex. 1013).

[8] For the ground of unpatentability pertaining to claims 1–3, 7–9, and 13–15, Petitioner refers to the combination of "*Lindskog* (Ex. 1007) in view of *Turek* (Ex. 1012)" with "*Lindskog II* and *Lindskog III* . . . [being]

II. ANALYSIS

A. Claim Construction

Because Petitioner filed the Petition on April 14, 2019, the Board applies the same claim construction standard used in a civil action under 35 U.S.C. § 282(b), as articulated in *Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005) (en banc). *See* Changes to the Claim Construction Standard for Interpreting Claims in Trial Proceedings Before the Patent Trial and Appeal Board, 83 Fed. Reg. 51,340, 51,340, 51,358 (Oct. 11, 2018) (amending 37 C.F.R. § 42.100(b) effective November 13, 2018) (now codified at 37 C.F.R. § 42.100(b) (2019)).

Petitioner submits the claims "must be construed consistent with [their] ordinary and customary meaning as understood by a POSITA" (person of ordinary skill in the art), and further submits that "[a]s part of the co-pending litigation, the parties agreed to . . . constructions" for the claim terms of "agent" and "thread" and other terms. *See* Pet. 10–11 (citing Ex. 1005; Ex. 1006); *supra* § I.B (related district court litigation). Petitioner also submits "the [c]ourt in the co-pending litigation issued its Claim Construction Order, construing two terms of the '659 Patent" including the claimed "global modeler" and the claimed "first making a request for a

---

incorporated by reference in *Lindskog* . . . such that their disclosures are part of *Lindskog* for purposes of Ground[] 1." Pet. 5. Petitioner alternatively provides that "[t]o the extent there is any doubt about that [i.e., the incorporation by reference], the Challenged Claims are further rendered obvious by *Lindskog* (Ex. 1007) in view of *Lindskog II* (Ex. 1008), *Lindskog III* (Ex. 1010), and *Turek* (Ex. 1012)." *Id.* Petitioner provides similar alternative listings for the ground of unpatentability pertaining to claims 6 and 18. *Id.* at 5–6.

corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment." *Id.* at 11–18 (citing Ex. 1006). Petitioner urges the Board to adopt its construction of these terms, "which are the same as the [district c]ourt's constructions—for purposes of this [P]etition." *Id.* at 12.

The Board adopted the constructions for purposes of institution. Inst. Dec. 9–10. Patent Owner does not challenge the constructions. *See* PO Resp. 8.

The claim construction agreed upon by the parties and supported by the record follows:

Claim term	Construction
agent	program that performs some type of operation, which may be information gathering or some processing task, in the background
thread	the execution of a section of code independently within a software program
global modeler	a module that enforces policies upon subordinate modelers
first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment	if the corrective policy is not available to an agent operating within the first runtime environment, first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment

See Pet. 10–16; PO Resp. 8; Inst. Dec. 9–10.

Only those terms in controversy need to be construed and only to the extent necessary to resolve the controversy. *See Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co.*, 868 F.3d 1013, 1017 (Fed. Cir. 2017) ("we need only construe terms 'that are in controversy, and only to the extent necessary to resolve the controversy'" (quoting *Vivid Techs., Inc. v. Am. Sci. & Eng'g, Inc.*, 200 F.3d 795, 803 (Fed. Cir. 1999))). On this record, no further construction of claim terms is necessary. No constructions raise any material dispute here. PO Resp. 8 ("Any disagreements that Patent Owner may have with these constructions are not material to the arguments in this Response."). For purposes of this Final Written Decision, the record supports the constructions listed above. Accordingly, we maintain the district court's claim constructions in the co-pending litigation. *See* Ex. 1005; Ex. 1006; PO Resp. 8 (listing construction of terms).

B. Level of Ordinary Skill in the Art

The level of ordinary skill in the art is "a prism or lens" through which we view the prior art and the claimed invention. *Okajima v. Bourdeau*, 261 F.3d 1350, 1355 (Fed. Cir. 2001).

Factors pertinent to a determination of the level of ordinary skill in the art include "(1) educational level of the inventor; (2) type of problems encountered in the art; (3) prior art solutions to those problems; (4) rapidity with which innovations are made; (5) sophistication of the technology; and (6) educational level of workers active in the field." *Envtl. Designs, Ltd. v. Union Oil Co*., 713 F.2d 693, 696–97 (Fed. Cir. 1983) (citing *Orthopedic Equip. Co. v. All Orthopedic Appliances, Inc.,* 707 F.2d 1376, 1381–82 (Fed. Cir. 1983)). "These factors are not exhaustive but are merely a guide to

determining the level of ordinary skill in the art." *Daiichi Sankyo Co. Ltd, Inc. v. Apotex, Inc*., 501 F.3d 1254, 1256 (Fed. Cir. 2007). In determining a level of ordinary skill, we also look to the prior art, which may reflect an appropriate skill level. *Okajima*, 261 F.3d at 1355. Additionally, the Court informs us that "[a] person of ordinary skill is also a person of ordinary creativity, not an automaton." *KSR Int'l Co. v. Teleflex Inc.*, 550 U.S. 398, 421 (2007).

Petitioner asserts that a person having ordinary skill in the art at the time of the invention (1) "would have the equivalent of a four-year degree from an accredited institution in computer science, computer engineering or the equivalent and experience with, or exposure to, software development for networking applications or systems," (2) "would also have approximately two to three years of professional experience in software development for networking applications or systems," and (3) "[a]dditional graduate education could substitute for professional experience, while significant experience in the field might substitute for formal education." Pet. 9–10 (citing Ex. 1001 ¶ 50). Patent Owner does not dispute Petitioner's characterization of the person of ordinary skill in the art. PO Resp. 7. We adopt Petitioner's definition of the level of ordinary skill in the art as supported by the prior art of record and the '659 patent.

Patent Owner contends that Petitioner and Dr. Almeroth relied upon the wrong priority date, because they relied on a priority date of January 31, 2012, but the "actual priority date of the '659 Patent is in fact June 28, 2002." PO Resp. 6; *see* Ex. 1003 (Certificate of Correction). After noting this discrepancy, Patent Owner stated that "[t]he Board . . . need not resolve this dispute as it does not impact the resolution of this review." *Id.* at 7. The

Board hereby accepts Patent Owner's characterization that any discrepancy "does not impact the resolution of this review." *See id.*[9]

C. Principles of Law

If "the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains," 35 U.S.C. § 103 renders the claim obvious. *KSR*, 550 U.S. at 406. The question of obviousness involves resolving the underlying factual determinations, including (1) the scope and content of the prior art; (2) any differences between the claimed subject matter and the prior art; (3) the level of ordinary skill in the art; and (4) when available, evidence such as commercial success, long felt but unsolved needs, and failure of others. *Graham v. John Deere Co*., 383 U.S. 1, 17–18 (1966); *see KSR*, 550 U.S. at 407 ("While the sequence of these questions might be reordered in any particular case, the [*Graham*] factors continue to define the inquiry that controls.").

The Court sets forth "an expansive and flexible approach" to the question of obviousness. *KSR*, 550 U.S. at 415. Whether a patent claiming the combination of prior art elements would have been obvious involves determining whether any improvement amounts to more than the predictable use of prior art elements according to their established functions. *Id*. at 417. Reaching this determination, however, requires more than merely showing that the prior art includes separate references covering each separate

[9] Dr. Almeroth acknowledges that Patent Owner asserts the earlier (2002) priority date. *See* Ex. 1001 ¶ 3.

limitation in a challenged claim. *Unigene Labs., Inc. v. Apotex, Inc*., 655 F.3d 1352, 1360 (Fed. Cir. 2011). Rather, obviousness additionally requires that a person of ordinary skill at the time of the invention "would have selected and combined those prior art elements in the normal course of research and development to yield the claimed invention." *Id*.

"Obviousness may be defeated if the prior art indicates that the invention would not have worked for its intended purpose or otherwise teaches away from the invention." *Meiresonne v. Google*, 849 F.3d 1379 (Fed. Cir. 2017) (citing *DePuy Spine, Inc. v. Medtronic Sofamor Danek, Inc.*, 567 F.3d 1314, 1326 (Fed. Cir. 2009)). A reference teaches away "when a person of ordinary skill, upon reading the reference, would be discouraged from following the path set out in the reference, or would be led in a direction divergent from the path that was taken" in the claim. *Galderma Labs., L.P. v. Tolmar, Inc.*, 737 F.3d 731, 738 (Fed. Cir. 2013); *see also In re Kahn*, 441 F.3d 977, 990 (Fed. Cir. 2011) ("Nothing in Stanton can be said to discourage a person having ordinary skill in the art from using the visual-input control taught in Garwin in the claimed combination or to lead the skilled artisan in a direction divergent from the path taken by Kahn.")

A reference that "merely expresses a general preference for an alternative invention but does not criticize, discredit, or otherwise discourage investigation into" the claimed invention does not teach away. *Galderma Labs.*, 737 F.3d at 738. "An inference of nonobviousness is especially strong where the prior art's teachings undermine the very reason being proffered as to why a person of ordinary skill would have combined the known elements." *DePuy Spine*, 567 F.3d at 1326–27 ("The district court

thus found that Puno's teachings undermine the very reason Medtronic proffers as to why it would have been obvious to combine Puno and Anderson, viz., the creation of a rigid screw."); *see also In re Kahn*, 441 F.3d 977, 990 (Fed. Cir. 2011) ("Nothing in Stanton can be said to discourage a person having ordinary skill in the art from using the visual-input control taught in Garwin in the claimed combination or to lead the skilled artisan in a direction divergent from the path taken by Kahn.").

D. Overview of the Asserted References

1. Lindskog (Ex. 1007)

Lindskog, titled "Distributed Fault Management Architecture," relates to "a system and method for distributing control of the fault management functions throughout a communications network." Ex. 1007, code (54), 1:13–16. Lindskog's "distributed fault management architecture . . . decentralizes the fault management operations and increases the robustness of the fault management system, while permitting faults to be addressed in real time" by distributing fault management capabilities throughout the various levels of a communications network. *Id.* at 4:44–50.

Lindskog's Figure 1 follows:

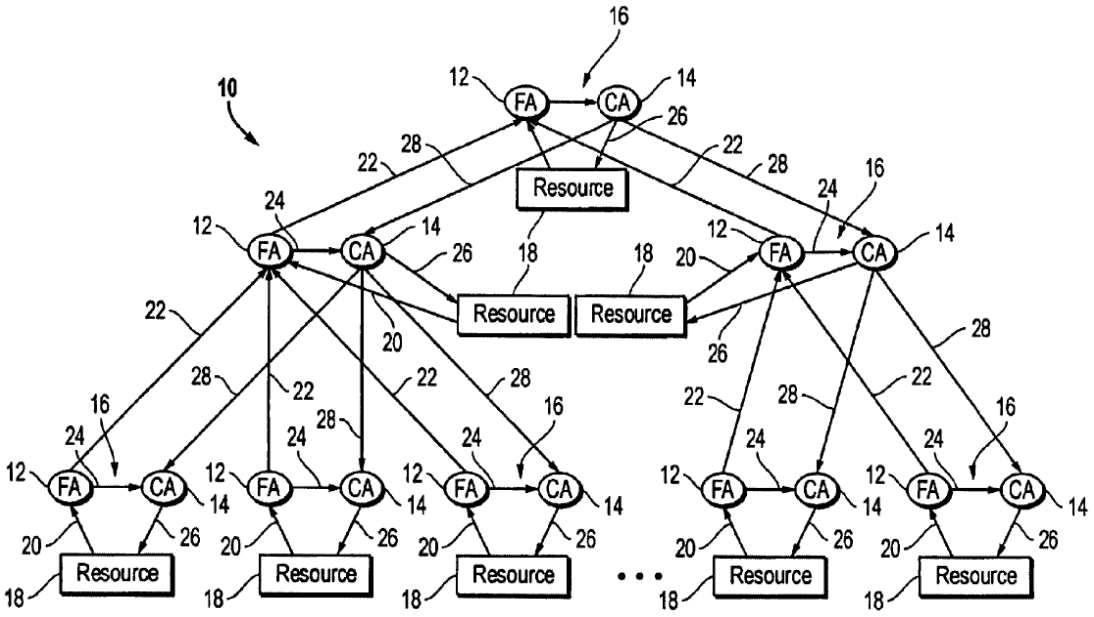

FIG. 1

Figure 1 illustrates distributed fault management system 10, using fault agents (FAs) 12 and configuration agents (CAs) 14 arranged in pairs and operating on different levels throughout system 10. Ex. 1007, 4:17–18, 58–62.

Lindskog's Figure 6 follows:

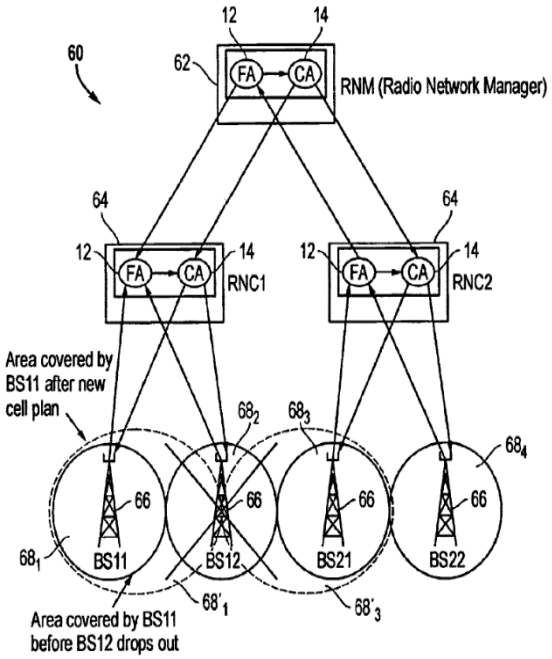

FIG. 6

Similar to Figure 1, Figure 6 illustrates a fault management architecture that also employs pairs 16 FA-CA of agents FA 12 and CA 14 at different levels and nodes, albeit in cellular telecommunications network 60. Ex. 1007, 4:32–33, 10:48–50. In Figures 1 and 6, the network "perform[s] distributed fault management functions in a communications network" that "includes a plurality of nodes. . . . usually arranged in a hierarchy and can comprise physical devices, such as base stations, radio network controllers, radio network managers, and the like." *Id.* at 3:12–18. At nodes with agent pairs FA-CA 16, "each node . . . supervise[s] one or more network resources, which can comprise logical resources (e.g., a cell border, a location area, etc.) and/or physical devices," for example resource 18 in Figure 1 and BS11 in Figure 6. *See id*. at 3:20–23, 4:67–5:1, Fig. 1.

When FA 12 detects an alarm, "either from a subordinate fault agent or from a network resource, the fault agent analyzes the received alarm data to identify a cause of the alarm and to determine if the underlying fault that caused the alarm can be handled at the current node." Ex. 1007, 3:27–32, 5:5–12, Fig. 1. If the underlying fault that caused the alarm cannot be handled at the current node, then fault agent FA 12 at the current node produces new alarm 22 that summarizes the received alarm data, and passes that new alarm 22 to an interconnected (or higher level) fault agent FA 12. *Id*. at 3:31–35, 5:5–13, Fig. 1. Once the alarm data reaches a fault agent FA 12 that can handle the underlying fault, that fault agent FA 12 forwards the fault information to its associated configuration agent CA 14. *Id*. at 3:35–38, 5:10–17, Fig. 1. Configuration agent CA 14 then processes the fault information to generate reconfiguration data for correcting or otherwise reducing the effect of the fault, and sends the reconfiguration data (26, 28 in Figure 1) to corresponding network resource(s) 18, or to subordinate configuration agent(s) CA 14, depending on which nodes the proposed reconfiguration affects. *Id*. at 3:37–44, 5:55–65, Fig. 1.

In one exemplary embodiment shown in Lindskog's Figure 6, cellular telecommunications network 60 uses Lindskog's fault management architecture to generate an alarm (indicating a coverage problem, such as a base station BS 12 malfunctioning), pass the alarm to FA 12 of Radio Network Manager (RNM) 62 via a FA 12 of a Radio Network Controller (RNC 1 or RNC 2), and deliver the alarm to the associated CA 14 in RNM 62. Ex. 1007, 11:5–24, Fig. 6. The CA 14 of RNM 62 resolves the alarm by generating a new cell plan that reconfigures some base stations 66 (e.g., BS 11 and BS 21) to provide cellular coverage in expanded cell coverage areas

$68'_1$ and $68'_3$ relative to original cell coverage areas 68_1 and 68_3. *Id*. at 11:25–46, Fig. 6.

Each agent includes an event receiver that receives alarm data from subordinate fault agents and an event generator that performs alarm correlation and filtering functions to identify the cause of the underlying fault that created the alarm. Ex. 1007, 3:51–55. Each agent's event generator communicates with a database:

> When the event generator identifies the cause of a particular alarm or set of alarms, the event generator updates the fault information *in an event database*, and, as a result, the event dispatcher sends the fault information to the associated configuration agent (if it is determined that the current node can handle the fault) or to a supervising fault agent, as higher level alarm data (if it is determined that the current node cannot handle the fault).

Id. at 3:56–63 (emphasis added); *see* Fig. 2. When CA 14 of FA-CA pair 16 can handle the alarm at the corresponding level and below, that CA 14 uses "corrective measures and reconfigurations" that "can include *the performance of complicated calculations (e.g.*, *involving various planning algorithms in knowledge-based systems*) that affect a number of underlying resources 18 or can be very simple, such as a decision not to act at all (i.e., a simple type of alarm filtering)." *Id.* at 5:50–55 (emphasis added). "Once" CA 14 "determine[s]" "the appropriate corrective measures and system reconfigurations," it sends

> *corresponding instructions* . . . to the underlying network resources 18 . . . and/or to the subordinate CAs 14, *depending on where corrective actions are needed*. Upon receipt of *such instructions* at a subordinate CA 14, *additional processing can be performed*, with any additional instructions being sent to the

> underling resources 18 . . . and/or even to more subordinate CAs 14.

Id. at 5:55–65 (emphases added).

Each supervising FA 12 also may use a subscription request system that forwards requests to subordinate FAs 12 for information only about "alarms . . . of interest." Ex. 1007, 6:2–9. Then, subordinate agents FA 12 each employ "processing unit 36 [that] interacts with an events database 38 to determine what actions need to be taken to properly respond to the request" from the upper level supervising agent FA 12. *Id.* at 6:40–43, *see id.* at Fig. 2. By forwarding only requested information, this provides "maximum flexibility" and "avoids the consumption of bandwidth," by avoiding sending data "not relevant for the moment. This can be thought of as an alarm filtering operation." *Id.* at 6:6–14.

Lindskog's system improves upon prior art "centralized fault management systems" that also detect "fault information (e.g., alarms), for both hardware and software faults." Ex. 1007, 2:16–21. According to Lindskog, prior art systems attempted to reduce "processor load" at nodes by sending locally stored periodic reports (e.g., after gathering statistics over a 5 or 15 minute interval) as an output file to a centralized management fault node. *See id.* at 2:1–19. Lindskog states that these prior art systems minimize the "sheer volume of information generated," but they do not provide "network performance information . . . in real time." *Id.* at 2:7–15.

2. Lindskog II (Ex. 1008)

Lindskog, discussed above, "incorporate[s] by reference herein [Lindskog II (Ex. 1008)] in its entirety." Ex. 1007, 1:6–9. Lindskog II, titled "Performance Management and Control System for a Distributed

Communications Network," relates to "a system and method for improving the performance management of a distributed communications network." Ex. 1008, code (54), 1:13–17. Lindskog II's management and control architecture "handle[s] numerous problems arising in the telecommunications field" and "protect[s] various network resources from overload caused by too high traffic levels being offered." *Id.* at 9:19–31. The system solves mobile station load problems by controlling the barring and unbarring of access to different access "classes" within cells that mobile stations seek to access. *See id.* at 15:4–16:26.

The system employs computers to handle real-time data. *See id.* at Fig. 2 ("computer interface," "[c]omputing platform 112"). Similar to Lindskog, Lindskog II's computer based system employs "control agents" in a hierarchical arrangement of higher and (subordinate) lower level CAs. Ex. 1008, 7:3–16. Each control agent includes "computing platform" 112, "object base" 104, and a subscription and event handler 108 that uses knowledge base 102 with numbered "application-specific control functions" each including a number of control tasks 1–n. *See id.* at Fig. 2, 8:36–67. Each CA employs "control logic" and "control algorithms." *See id.* at 8:51, 9:2.

> In particular, regarding Figure 2,
>
> a CA 100 includes a Knowledge Base (KB) 102, which is preferably a database that contains descriptions of the different control applications performed by the CA. The application-specific control functions are in turn specified in terms of one or more control tasks (1-n), each of which is given a priority (specifying the relative importance of the task in comparison with the other tasks). Associated with a control task is information about the data needed (e.g., what to subscribe to,

> with what periodicity, etc.), and to which external control commands to react. A control task (1-n) also contains the control logic that realizes the actual control algorithms. This logic may be encoded, for example, in terms of rules, neural networks, cases, or some other mechanisms used for representing knowledge which are appropriate for the considered class of control problems. An example of a control task involving a state transition diagram and a number of associated rules is described in more detail below.

Ex. 1008, 8:39–57.

In general, the CAs

> receive real time performance information associated with the distributed communications network from a plurality of performance agents, analyze the real time performance information, and in due time, output control commands that control the performance of the distributed communications network based on the real time performance data received.

Id. at code (57).

Also, each CA, distributed throughout the network,

> can handle many control tasks, each of which is defined by the priority of the task, the data needed (subscribed to), what external control commands to react to, and last but not least, the control logic. The control logic can be separated into two parts: the control actions (e.g., a control rule, a neural network mapping, etc.); and the sequence (or timing) between these actions.

Id. at 14:20–26.

> Interactions between CAs are performed by means of control commands, which specify the targets or constraints that a particular CA must meet, and set the goals of the control algorithms. The control commands are set by superior CAs in the hierarchy, or by a human operator through a graphical user interface. In the latter case, the person is in effect a superior CA.

Id. at 7:59–65.

Like Lindskog's system, Lindskog II's distributed communications network management and control system "provides a hierarchical control approach which is the decomposition of control responsibilities into cell, region, or sub-network domains, thereby allowing access control on the most appropriate level." Ex. 1008, 3:64–67, 4:58–60.

3. Lindskog III (Ex. 1010)

Lindskog II, discussed above, "incorporate[s] by reference [Lindskog III (Ex. 1010)] herein in its entirety." Ex. 1008, 1:6–9. Lindskog III, titled "Prioritized Agent-Based Hierarchy Structure for Handling Performance Metrics Data in a Telecommunication Management System," relates to "the performance management of a telecommunications network" to "ensure that corrective actions are taken when necessary and so that network design can be improved according to usage and performance." Ex. 1010, code (54), 1:8–12.

Figure 1 of Lindskog III follows:

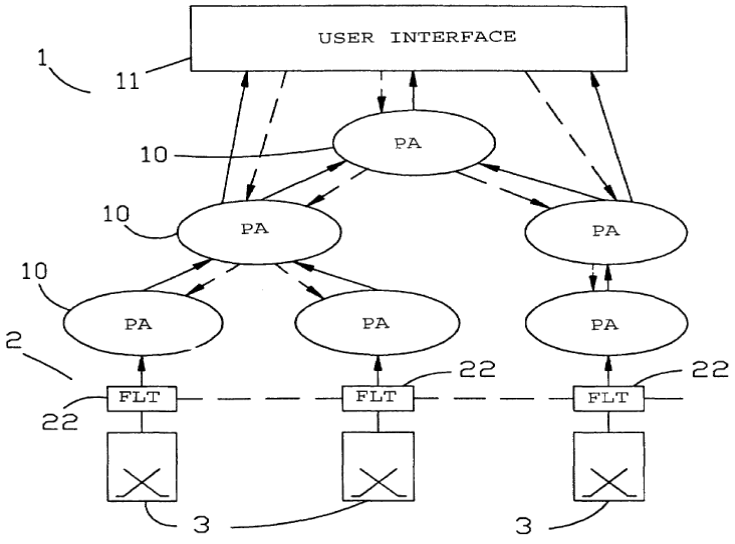

FIG 1

Lindskog III's Figure 1 schematically represents performance management system 1 for measuring performance of managed telecommunications network 2 having traffic machines 3. Ex. 1010, 3:63–64, 4:10–15. System 1 comprises a hierarchy of performance agents (PA) 10, with "[e]ach agent 10 . . . responsible for the processing, analysis and abstraction of the performance data received from the traffic machines 3 or lower level agents 10." *Id*. at 4:23–26. Each agent PA 10 reports a "pre-defined set of performance metrics as a minimum to its parent in the hierarchy," where the "set of metrics taken together with a set of defined threshold values specify a set of states for the network." *Id*. at 4:49–54. "Events relating to failure conditions, . . . called exception events[,] . . . are generated either by the traffic machines when they detect a *hardware/software* fault, or by an agent [PA 10] when a performance threshold is crossed." *Id.* at 5:31–35 (emphasis added). "Once an agent 10 receives an exception event, it correlates it with the performance information to determine how severe the exception condition is." *Id*. at 5:28–37.

4. Turek (Ex. 1012)

Turek, titled "Mobile Agents for Fault Diagnosis and Correction in a Distributed Computer Environment," relates to "managing a large distributed computer enterprise environment" and "diagnosing and correcting network faults in such an environment using mobile software agents." Ex. 1012, code (54), 1:7–11. The system seeks to dispatch "the minimum amount of code that may be necessary to rectify a given network fault." *Id.* at 2:12–14. Accordingly, the dispatched "software agent is a minimum set of tasks that are identified for use in diagnosing and/or correcting the fault." *Id.* at 2:19–21.

Turek's Figure 1 follows:

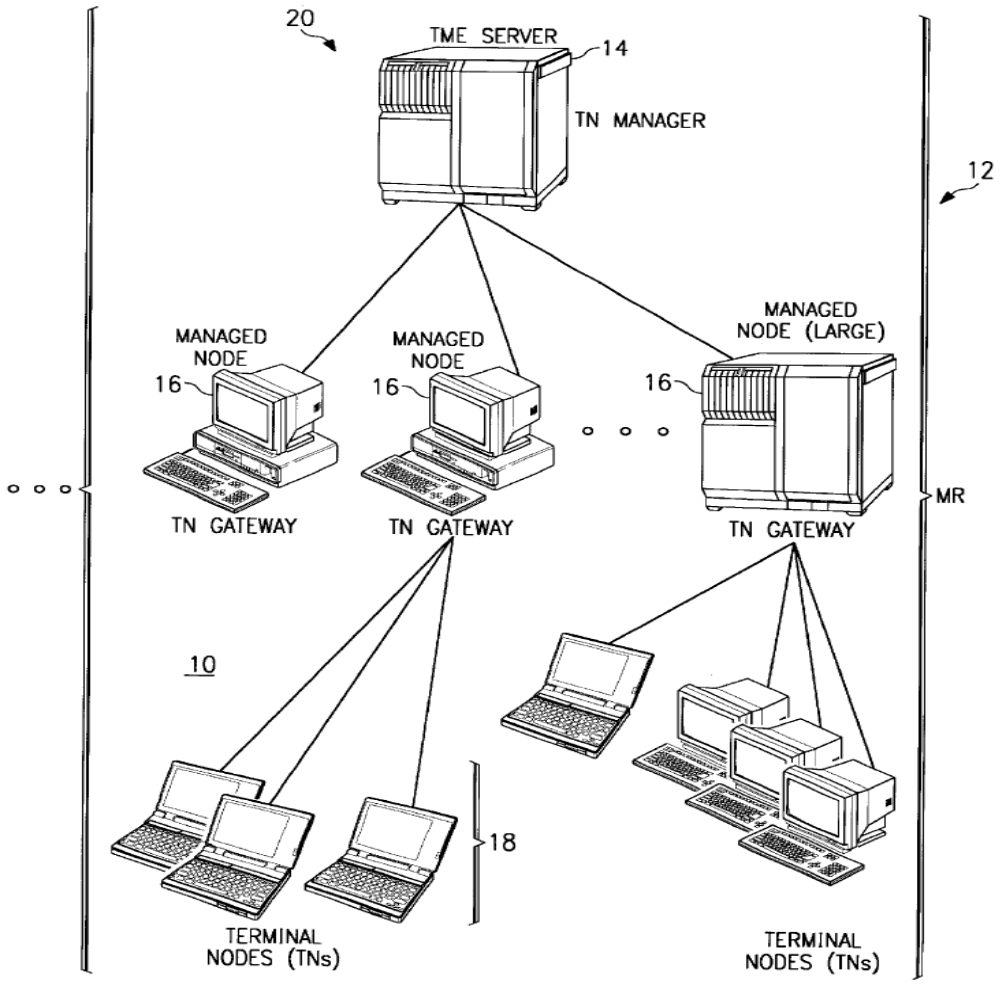

FIG. 1

Figure 1 illustrates distributed computer environment 10 comprising "up to thousands of 'nodes'" managed in a distributed manner. Ex. 1012, 3:20–23, 47–51. The managed environment in Turek's Figure 1 "logically br[eaks] down into a series of loosely-connected managed regions (MR) 12, each with its own management server 14 for managing local resources with the MR." *Id*. at 3:52–55. Multiple servers 14 coordinate activities across the enterprise and permit remote site management and operation, each server 14 serving a number of gateway machines 16, each of which in turn supports

a plurality of terminal nodes/endpoints 18. *Id*. at 3:59–63. Server 14 coordinates all activity within the MR using a terminal node manager 20. *Id*. at 3:63–64. "[W]hen a network error or 'fault' is reported whose cause and location are not apparent or readily ascertainable, an appropriate agent is identified and dispatched to determine this information." *Id*. at 5:43–48.

Figure 5 of Turek follows:

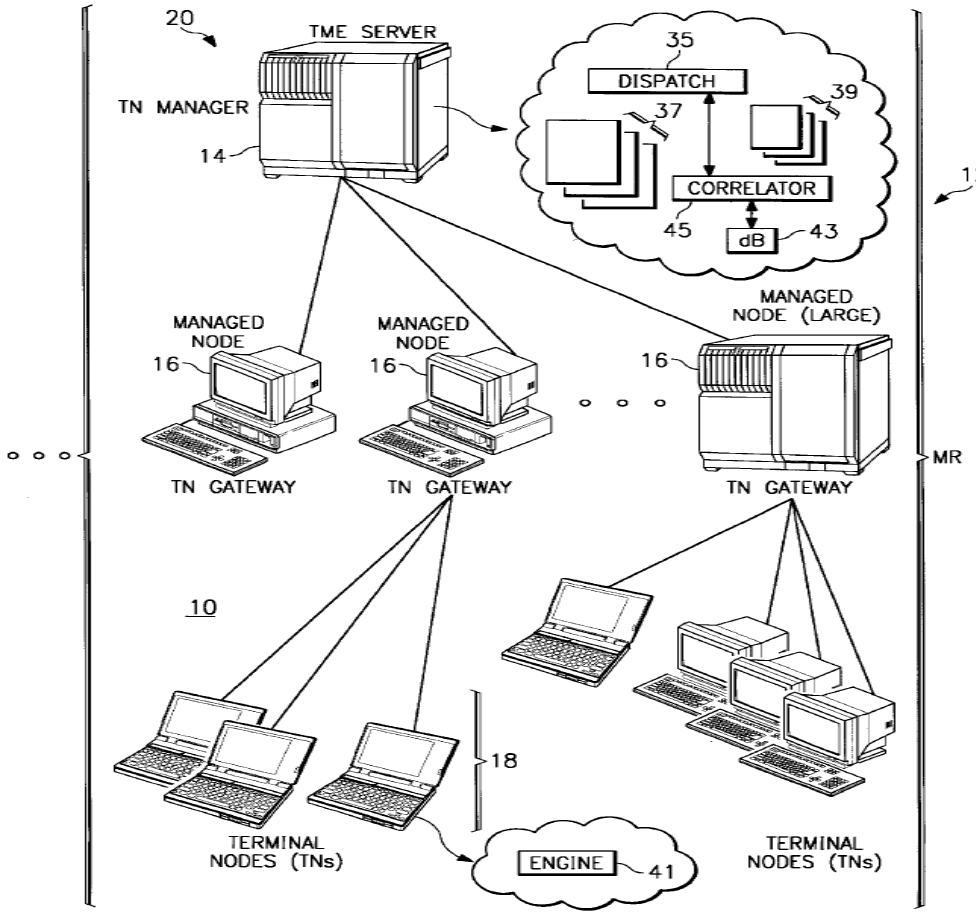

FIG. 5

Figure 5 illustrates a distributed computer network environment including manager server 14 with dispatch mechanism 35 employing "software agents 37" to manage nodes 16 and 18. Ex. 1012, 3:37–40, 5:65–67. Manager server 14, via dispatch mechanism 35, dispatches an agent

from "a set of software agents 37." *Id*. at 5:65–6:2, Fig. 5. Manager server 14 attempts to dispatch agent(s) 37 to the actual node in the network at which the fault condition occurs, but "[i]f the agent does not find the fault at the initial location to be examined, the agent then migrates through the network to locate the error." *Id*. at 5:46–53. A "particular triggering 'event' is used to provide clues as to the network location to which the agent should be sent, as well as the type of agent to send." *Id.* at 2:44–46.

"Alternatively, dispatch mechanism 35 may include a set of configurable software tasks 39 from which one or more agents are constructed." *Id.* at 5:67–6:2. Runtime "engine 41 provides a runtime environment for the software agent" at each managed node. Ex. 1012, 6:9–10.

> In one embodiment,
>
> both the runtime engine and the software agent(s) are conveniently written in Java. As is known in the art, Java is an object-oriented, *multithreaded*, portable, platform-independent, secure programming environment used to develop, test and maintain software programs. Java programs have found extensive use on the World Wide Web, which is the Internet's multimedia information retrieval system. These programs include full-featured interactive, *standalone applications, as well as smaller programs, known as applets*, that run in a Java enabled Web browser.

Ex. 1012, 6:38–48 (emphases added).

> At each node, *the software agent is preferably run by the runtime engine previously deployed there.* Alternatively, the software agent runs as a standalone process using existing local resources. As noted above, when the event is a fault, the software agent locates the fault and attempts to rectify it. If necessary, the software agent may obtain additional code from the dispatch

> mechanism or some other network source. Such additional code may be another software agent.

Ex. 1012, 2:63–3:3 (emphases added). Sending "event driven" software agents with a minimal amount of code "conserves network bandwidth." *Id.* at 9:2–10, 38–42 ("By distributing some of the diagnostic functionality in the engine, network bandwidth is conserved because only a small amount of code needs to be dispatched to the target site.").

5. Dijkstra (Ex. 1013)

Dijkstra discloses an algorithm for "self-stabilizing systems in spite of distributed control," where a system is "'self-stabilizing' if and only if, regardless of the initial state and regardless of the privilege selected each time for the next move . . . the system is guaranteed to find itself in a legitimate state after a finite number of moves." Ex. 1013, 643, Title (capitalization altered). In contrast, non-self-stabilizing systems "when once (erroneously) in an illegitimate state, they could—and usually did[]—remain so forever." *Id.* at 643.

E. Lindskog, Lindskog II, Lindskog III, and Turek, Claims 1–3, 7–9, and 13–15

Petitioner argues that claims 1–3, 7–9, and 13–15 would have been obvious over the combined teachings of Lindskog and Turek, with Lindskog II incorporated by reference in Lindskog, and Lindskog III incorporated by reference in Lindskog II. Pet. 35–67. Patent Owner opposes. PO Resp. 10–27; PO Sur-reply 1–27.

1. Independent Claim 1

a. Limitation [a] ("a computer-implemented method, comprising")

Petitioner contends that Lindskog teaches limitation [a]. Pet. 35–36. Limitation [a], the preamble, recites "[a] computer-implemented method, comprising." Petitioner relies on disclosed computer functionality in Lindskog including "performing distributed fault management functions in a communications network," via GRAFCET computer programming charts used by agents 12 and 14 and also standard communication mechanisms such as CORBA.[10] *See* Pet. 35–36 (citing Ex. 1001 ¶¶ 106–107; Ex. 1007, code (54), Fig. 2 (showing processor 36, event generator 44, and events data base 38), Fig. 3 (GRAFCET computer programming function chart), 11:65–67 (describing a "standard communication mechanisms such as CORBA"). Petitioner refers to other software functionality in addressing limitations [b] and [c] as discussed further below.

Patent Owner does not challenge Petitioner's showing that Lindskog teaches a computer-implemented method.

After considering the full record, Petitioner persuasively shows that Lindskog discloses the subject matter of the preamble even if it is limiting.

b. Limitation [b] ("running at least one thread in a first runtime environment")

Petitioner asserts Lindskog (and Lindskog II) each disclose a thread and runtime environment as claimed, "but to the extent *Lindskog* does not

[10] Petitioner and Dr. Almeroth do not provide the full name corresponding to the GRAFCET and CORBA acronyms but generally describe the two respectively as standards for computer control and communication. *See* Ex. 1003 ¶ 109.

expressly disclose 'thread' and 'runtime environment,'" Petitioner contends that "*Turek* expressly discloses those limitations" and renders the subject matter obvious. *See* Pet. 36–37; *infra* § II.E.i (discussing Petitioner's motivation to combine Turek and Patent Owner's responses).

Regarding Lindskog and Lindskog II, Petitioner argues that (i) "each agent pair [(i.e., a fault agent (FA) and an associated configuration agent (CA))] can be included in a network node, and each network node may be associated with a different device," so "a POSITA would understand that each of *Lindskog's* network nodes has a runtime environment supporting the agent pairs running therein" and that (ii) each of Lindskog II's sub-network "domains incorporates a runtime environment that runs the code of the agent pairs operating within the domain." *Id.* at 37 (citing Ex. 1001 ¶¶ 109–111; Ex. 1007, 3:11–13, 19–20, 4:62–65, Fig. 1; Ex. 1008, 3:64–66); *see also id.* at 36 (quoting Ex. 1007, 11:65–67 ("Flexibility can also be achieved by allowing the agents 12 and 14 to be interconnected in almost any way by using, for example, standard communication mechanisms such as CORBA.")).

Petitioner also asserts that Lindskog discloses the claimed "at least one thread," where Lindskog's "***network resources*** . . . ***can comprise logical resources***" and involve "***software faults.***" Pet. 37–38 (emphasis by Petitioner) (quoting Ex. 1007, 2:15–16. 3:20–23). Petitioner reasons that threads run in Lindskog, because Lindskog "discloses that its network resources can receive corrective measures independently of other network

resources that did not receive instructions from the supervisory agent pair." *Id.* at 38 (citing Ex. 1001 ¶ 112).[11]

Petitioner also reasons that Lindskog discloses threads because "corrective measures can include the performance of complicated calculations . . . that ***affect a number of underlying resources 18***[,]" and "[o]nce the appropriate corrective measures and system reconfigurations are determined by the CA 14, ***corresponding instructions are sent from the CA 14 to the underlying network resources 18***." Pet. 38 (emphasis by Petitioner) (quoting Ex. 1007, 5:60–65). Petitioner also reasons that Lindskog discloses that "[e]ach FA-CA pair 16 ***supervises one or more underlying network resources 18*** and/or subordinate FA-CA pairs 16" and contends this "can be software-based and constitute threads." Pet. 37, 42 (emphasis by Petitioner) (quoting Ex. 1007, 4:67–5:1).

In summary, according to Petitioner,

> [b]ecause one or more selected network resources implements instructions and configuration information, while other non-selected network resources do not, *each of the selected network resources executes independently of non-selected network resources*. Ex. 1001 ¶112. A POSITA would therefore recognize that each of *Lindskog's* network resources executing within a first domain underlying a lower level agent pair is a "thread" running within a corresponding first runtime environment because each of the network resources can include the execution of a section of code independently within a software program running in the corresponding runtime environment.

[11] As noted above (*see supra* §§ I.B, II.A), the Board adopts the parties' and district court's construction of a "thread" as "the execution of a section of code independently within a software program." *See* Pet. 11, 37; Ex. 1006, 2.

Pet. 39 (emphasis added) (citing Ex. 1001 ¶ 112; Ex. 1007, Fig. 1). In other words, Petitioner relies on Lindskog's teachings that each agent pair controls and monitors alarm *data* within each of one or more underlying resources that each include independently running software to show that Lindskog's system implies that threads run in that runtime environment. *See id.* (citing Ex. 1001 ¶¶ 111–112; Ex. 1007, 2:15–16, 3:20–23, 26–28, 4:67–5:1, 5:60–65, Fig. 1; Ex. 1001 ¶¶ 111–112).

Petitioner also relies on Turek's multi-threaded architecture to supplement Lindskog's teachings with respect to the running of runtime engines, threaded agents, and resources. *See* Pet. 31–33 (discussing motivation, citing Ex. 1003 ¶¶ 104–103), 40–41. For example, Petitioner asserts that "*Turek* . . . teaches running at least one thread in a runtime environment," because Turek describes a Java-based "***multi-threaded***" "***runtime environment (e.g., an engine)***" and "***software agent***" wherein "at each node, the ***software agent is preferably run by the runtime engine.***" *Id.* at 40 (emphasis by Petitioner) (quoting Ex. 1012, 2:33–34, 2:63–65, 3:33–36), 3:65, 6:38–44, Fig. 2). Petitioner contends that running at least one resource thread with software threaded agents in a multi-threaded runtime environment in Lindskog's system, as Turek suggests, would have been obvious to improve network efficiency, *inter alia*, by decentralizing the system, conserving bandwidth, limiting the amount of code required to be executed, and by allowing parallel processes to run simultaneously. *See id.* at 30–33 (discussing motivation to combine Turek), 40–41 (citing Ex. 1001 ¶ 113). Petitioner also asserts that it would have been obvious to employ Turek's multi-threaded agents for "multiple resources running as threads" so that the resources "could be implemented concurrently in parallel to achieve

a corrective measure, thus saving time and more efficiently resolving detected faults." *Id.* at 33 (citing Ex. 1001 ¶ 103).

Patent Owner contests Petitioner's showing regarding the claimed "runtime environments," asserting that Lindskog I's and II's domains "are not runtime environments because a domain is not something that is hosted on a particular device." PO Resp. 19 (quoting Ex. 2001 ¶¶ 51–52). Patent Owner equates the claimed "runtime environments" with "domains" described in Lindskog II. *See id.* Patent Owner quotes Dr. Madisetti's testimony that "[a] person of ordinary skill in the art would not understand a runtime environment to include multiple separate network devices that simply have some logical association with each other, such as a 'domain.'" *Id.* (quoting Ex. 2001 ¶ 51). Patent Owner contends the '659 patent supports this testimony, because "[t]he 'agent runtime environments' of the '659 Patent are described as being '*hosted* on a particular network (host) device.'" *Id.* (quoting Ex. 1003, 2:15–17; citing Ex. 2001 ¶ 51).

As Petitioner points out, however, nothing requires the claimed runtime environment to be hosted on a particular device. *See* Reply 2–4. Relying on the combined teachings, Petitioner persuasively shows that "each node has a runtime environment supporting the agents running therein" (Reply 2) with each runtime environment running a thread supervised by those agents (Pet. 36–40). Patent Owner bases its arguments on the allegation that runtime environments do not read on Lindskog II's domains and runtime environments and domains must be hosted on a device. *See* PO Resp. 18–22. In an attempt to avoid repetition, this Final Written Decision discusses Patent Owner's arguments that attempt to equate runtime environments with domains in the related discussion of limitation [f] below

and incorporates that discussion by reference here to the extent it applies. *See* PO Resp. 18–22 (addressing domains in the context of runtime environments); PO Resp. 42–45 (similar arguments); *infra* § II.E.1.e.

Regarding Patent Owner's argument that the '659 patent requires a runtime environment to be hosted on a device (*see* PO Resp. 19), Petitioner characterizes the '659 patent as indicating that an ARE (agent runtime environment) "*may be* 'hosted on a particular network (host) device.'" Reply 3–4 (emphasis added) (quoting Ex. 1003, 2:17). To show that the '659 does not require an ARE to be hosted on a device, Petitioner points out that the '659 patent states that "[i]n some cases a host device may be unable to support an ARE or an agent running on it." *Id.* at 4 (quoting Ex. 1003, 4:20–22).

In its Sur-reply, Patent Owner contends "this [column 4] citation [by Petitioner] confirms that a runtime environment is hosted or supported on a device." PO Sur-reply 2. Patent Owner points out that the '659 patent also states that "[i]n these cases [in which a host device cannot support an ARE], *the nearest ARE . . . is used to provide a proxy for that host device*," and Patent Owner also contends that "in the next paragraph," the '659 patent "describes that these [nearest proxy] AREs are 'hosted on a host device 30.'" *Id.* at 2–3 (emphasis by Board) (quoting Ex. 1003, 4:21–33). Patent Owner otherwise "agrees that 'runtime environment' has its plain and ordinary meaning." *Id.* at 1.

As Petitioner argues, however, the '659 patent does not describe all AREs or runtime environments as hosted on a device. For example, in the paragraph that Patent Owner relies upon, the '659 patent states that "FIG. 2 shows a schematic representation of the components of an ARE 12 hosted

on a host device 30." Ex. 1003, 4:31–32; *see* PO Sur-reply 2–3. Contrary to Patent Owner's arguments, this sentence does not state that the nearest or "proxy" ARE is "hosted on a host device." *See* Sur-reply 2–3. Also, the '659 patent states that AREs may include "a vendor specific interface which lacks the capacity to host an ARE." Ex. 1003, 4:16–17 (emphasis added); *see* Reply 4 (citing Ex. 1003, 4:17–21 (citing a CORBA embodiment)). The '659 patent also describes "[p]olicy agents [that] operate on . . . remote devices (*devices which cannot run an ARE* for agents to execute in)," further supporting Petitioner. *See* Ex. 1003, 12:62–65 (emphasis added).

Therefore, the record supports Petitioner's contention that the '659 patent does not require the claimed "runtime environments" (or the disclosed AREs) to be hosted on a device. Furthermore, Patent Owner relies on an ARE embodiment (as opposed to the claimed generic runtime environments), but even if a patent describes just a single embodiment, claims are not necessarily limited to all the features of that embodiment as Petitioner argues. *See* Reply 3 (*citing Superguide Corp. v. DirecTV Enterprises, Inc.*, 358 F.3d 870, 875 (Fed. Cir. 2004) ("Though understanding the claim language may be aided by the explanations contained in the written description, it is important not to import into a claim limitations that are not a part of the claim.")).

Further supporting Petitioner, the '659 patent states that "[t]he software agents operate within an agent runtime environment (*hereinafter referred to as an 'ARE'*) which is hosted on a particular network (host) device." Ex. 1003, 2:14–17 (emphasis added); Reply 3 (citing Ex. 1003, 2:15–18). Specifying an "*agent* runtime environment" followed by the "hereinafter . . . 'ARE'" phrase shows that the '659 patent distinguishes

between the disclosed "ARE" or "agent runtime environment" that requires "software agents" and the claimed runtime environment. *See* Ex. 1003, 2:15–18 (emphasis added).

In other words, the challenged claims do not recite an "agent runtime environment" or "ARE." Rather, claim 1 recites a more general "runtime environment" and it also recites an "agent" operating therein. Specifically, limitation [b] of claim 1 does not require an agent, but limitation [f] recites "a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to *an agent operating within the first runtime environment*." These recitations in limitations [b] and [f] imply that the claim construction of a "first runtime environment" does not *require* an *agent* runtime environment (i.e., an ARE) as described with all its attributes in the '659 patent, because the latter limitation uses other language to require an "agent" to be part of (i.e., "operating within") the generic "runtime environment." Construing the "runtime environment" itself to require an agent based on the construction of a runtime environment renders the "agent" recitation in limitation [f] superfluous.[12] Therefore, by similar logic, even if the '659 patent requires the disclosed AREs to be hosted on a device, for this additional reason, it does not require the generically claimed runtime environment to be hosted thereon.

In further support of this interpretation, the '659 patent describes several requirements for AREs such as "service libraries 12.1," "a device

[12] Claim 1 also recites "running at least one thread in a first runtime environment." In analogous fashion, this does not mean that a runtime environment must be a threaded runtime environment or include a thread running therein based on the construction of a "runtime environment."

adaptor 12.2,", "a security manager 12.3," "a unique ID," and "a unique communications port ID." *Id.* at 4:47–5:5. The '659 patent also states "[the] AREs are able to create/instantiate, suspend operation of the agents, and terminate the agents." *Id.* at 2:53–54. These disclosures further indicate that the claimed "runtime environment" represents a broader environment than the more specific AREs described in the '659 patent and does not require all of the disclosed attributes of an ARE. Patent Owner essentially attempts to limit the claimed "runtime environment" to be hosted on a device by selectively picking one possible feature of a disclosed but unclaimed ARE without accounting for the slew of other disclosed ARE features.

Petitioner also persuasively relies on combining the teachings of Turek to suggest "threads can be run in the runtime environments." Pet. 37. As Petitioner notes, Turek discloses that "*at each node*, *the software agent is preferably run by the runtime engine*." *Id.* at 40 (emphasis added) (quoting Ex. 1012, 2:63–65). Petitioner also notes that Turek teaches that "***the runtime environment (e.g., an engine)*** is preferably part of a distributed framework supported on each managed node of the distributed enterprise environment." *Id.* (emphasis by Petitioner) (quoting Ex. 1012, 3:33–36). Petitioner adds that Turek's multi-threaded network creates a predictable result of a distributed, multi-threaded network that improves efficiency. Patent Owner agrees that Turek teaches runtime environments on a node. *See* PO Resp. 18 (discussing Turek's runtime environments as supported on or at individual nodes or endpoint machines).[13]

[13] The parties agree that a runtime environment carries its plain and ordinary meaning, and as noted above, Patent Owner agrees (or at least does not dispute) that Turek discloses a runtime environment at each node. *See* PO

Patent Owner addresses Lindskog by asserting that the references do not disclose "running at least one thread in a first runtime environment" as limitation [b] requires. PO Resp. 23–25. According to Patent Owner, Lindskog refers to "'logical resources (*e.g.*, a cell border, a location area, etc.)' to mean a logical grouping that can include multiple physical devices." PO Resp. 23 (citing Ex. 2001 ¶ 58). Patent Owner also contends that Lindskog "specifically describes a 'physical device[]' as an example of a 'network resource[].'" PO Resp. 23 (citing Ex. 1007, 5:2–3 ("a network resource 18 such as a single circuit in a cellular base station")). Patent Owner contends nothing in the Lindskog references "provide[s] any examples of 'logical resources' . . . that are software programs or the execution of a section of code independently within a software program." *Id.* at 24 (citing Ex. 2001 ¶ 58). Patent Owner also argues that "because no operational parameters relating to the FA-CA pairs are monitored, the FA-CA pairs cannot be the 'threads' 'running within a first runtime environment.'" *Id.* at 25 (noting that limitation [c] requires "monitoring operational parameters *relating to the each thread*").

Patent Owner then turns to Turek and contends it does not suggest the claimed threads, because "the agents of Turek are not deployed into the

---

Resp. 18; Sur-reply 1–4; Reply 3 (citing Ex. 1006, 2 (district court construction)). Given that Patent Owner does not dispute that Turek disclose a runtime environment at each node (PO Resp. 19–22; Sur-reply 2), and Petitioner contends that "each node [including Lindskog's nodes as modified by Turek] has a runtime environment supporting the agents running therein" (*see* Reply 2; Pet. 37; Ex. 1001 ¶¶ 109–111), no further (implicit) construction of the term is necessary. *See also* Ex. 1003, 4:28–30 ("Each ARE 12 provides a uniform context and set of services that agents can rely on, thereby acting as an operating system for the agents.").

network to its nodes until *after* an event is detected by the central dispatch mechanism." PO Resp. 25–26 (citing Ex. 1012, 2:36–49, 6:55–59, 7:49–59; Ex. 2001 ¶ 60). Regarding the latter argument, Patent Owner contends that "[a]s with the FA-CA pairs of Lindskog I, Petitioner does not assert that the 'agents' of Turek are the claimed 'threads.'" *Id.* at 25 n.5. Patent Owner explains that "Patent Owner addresses" Turek's alleged agents and Lindskog I's FA-CA pairs as threads only "because the Board cited them as relevant to this limitation in its Institution Decision," but "Petitioner did not." *Id.* at 25 n. 5 (citing Inst. Dec. 32–33).

Patent Owner's arguments do not undermine Petitioner's showing. As discussed above and throughout this Final Written Decision, Petitioner contends that Lindskog and Turek combine to teach or suggest *threaded runtime environments, threaded agents, and threaded (monitored) resources* in addressing limitations [b]–[h]. *See, e.g.*, Reply 1 ("[T]hreads, as construed, are disclosed by *Lindskog's* software-based logical resources and *Turek's* multi-threaded software agents"), 5–6 (discussing Turek's multi-threaded Java agents and runtime engines as threads); Pet. 33 (obviousness of running multiple threads in parallel including "multiple resources running as threads"), 41 (obviousness of using Turek's multi-threaded environment and running at least one thread).[14]

[14] *See also* Reply 4 ("Petitioner explained that the logical resources that issue these software faults are claimed threads, as construed" (citing Pet. 38)); Reply 6 ("Petitioner explained that *Turek's* multi-threaded software agent that is run by the runtime environment discloses a 'thread' running within a first runtime environment." (citing Pet. 40)); Pet. 42 (citing supervision by Lindskog's upper FA-CA pair of both underlying "***network resources* 18** and/or subordinate FA-CA pairs 16" (including "subordinate FAs 12") in

With respect to Patent Owner's arguments and Dr. Madisetti's testimony that Lindskog only monitors physical devices and not software, the record does not support the arguments and testimony. *See* PO Resp. 23–26 (citing Ex. 2001 ¶¶ 57–60). As one specific example, Dr. Madisetti acknowledges that Lindskog III discloses a "processor . . . that processes data," but dismisses it as a "physical device." Ex. 2001 ¶ 57 n.3.

This cited testimony does not address Petitioner's showing based on the combination of the Lindskog references and Turek wherein the resources (including processors) run software threads that the software agents monitor. *See* Ex. 2001 ¶¶ 57–60; Pet. 30–33, 36–45. Moreover, Lindskog discloses that its system employs software agents and monitors software and hardware resources, by disclosing, for example, algorithms and knowledge-based systems, which require software (and by incorporating by reference Lindskog II and III). *See supra* §§ II.D.1, II.D2, II.D3 (summarizing Lindskog, Lindskog II, and Lindskog III (the "Lindskog references")), II.E.h (discussing incorporation by reference of Lindskog II and III into Lindskog); Pet. 36–39 (persuasively describing software in Lindskog and the Lindskog references). For example, in addition to describing "both hardware and software faults" as typical in fault management systems (*see* Ex. 1007, 2:1–

---

discussing monitoring operational parameters relating to threads with respect to related limitation [c]); Pet. 40–41 (quoting Turek's multi-threaded software agent and runtime teachings and asserting the obviousness of running agents in a multi-threaded environment); Reply 6–7 (addressing Patent Owner's argument that "Turek's agents cannot be the claimed threads," referring to "multi-threaded software agent[s]," and citing to its reliance on Turek's agents as threaded in the Petition (citing Pet. 31–33, 40–41, 53; Ex. 1001 ¶ 113)).

17), Lindskog teaches "corrective measures and reconfigurations" that "can include *the performance of complicated calculations (e.g.*, *involving various planning algorithms in knowledge-based systems*) that *affect a number of underlying resources 18* or can be very simple, such as a decision not to act at all (i.e., a simple type of alarm filtering)." Ex. 1007, 5:50–55 (emphasis added); Pet. 38–39 (quoting Ex. 1007, 5:50–55 (i.e., portions of the same sentence in Lindskog)). The '659 patent similarly discloses that "[a]gents are used to enforce specific actions/behavior on *hardware*/software devices controlled by the system 10." Ex. 1003, 11:32–33 (emphasis added).

As noted, Petitioner also contends that Lindskog teaches monitoring alarms for "for both hardware and ***software faults***." Reply 4 (emphasis by Petitioner) (quoting Ex. 1007, 2:15–16; citing Pet. 38). In the Sur-reply, Patent Owner contends that Lindskog "does not state where the faults come from," and contends *inter alia*, that even if the faults are software, "this is insufficient" as a showing of threads. *See* PO Sur-reply 5–6.

Dr. Madisetti's testimony does not address Petitioner's showing related to these software faults. *See, e.g.*, Ex. 2001 ¶¶ 56–60. Patent Owner's arguments similarly ignore that Lindskog implies that faults emanate from independently monitored and controlled hardware circuits and software. They also fail to rebut Petitioner's reliance on Turek's teachings, as discussed further below.

In addition to the citations in the previous paragraph, Lindskog refers to processing or providing data or information with respect to monitored resources (e.g., monitoring for faults). Petitioner for example, reproduces Lindskog's Figure 2 in connection with monitoring operational parameters. *See* Pet. 43 (also citing Ex. 1010, 5:28–35 (Lindskog III)). With respect to

Figure 2, Lindskog teaches that its agents employ "*processing unit* 36 [that] interacts with an events *database* 38 to determine what actions need to be taken to properly respond to the request" from the upper level supervising agent FA 12. Ex. 1007, 6:40–43 (emphasis added), *see* Fig. 2. As another example, in the Summary of the Invention, Lindskog discloses that "[t]he configuration agent *processes* the fault *information* to generate reconfiguration *data* for correcting or otherwise reducing the effect of the fault and sends the reconfiguration *data to a network resource (ordering it to perform some action)*." *Id.* at 3:38–42.

As Petitioner also notes, Lindskog employs a GRAFCET computer functioning chart, and the agent pairs communicate with each other at nodes using a standard computer communication protocol CORBA to interconnect the pair. *See* Pet. 36 (citing Ex. 1007, Fig. 3, 11:65–67; Ex. 1001 ¶¶ 106–107).[15]

Describing related art in the "BACKGROUND OF THE INVENTION" section, Lindskog discloses using "a number of applications residing on a software platform" for monitoring hardware and software faults, and performing network management, using storage of files and processors, etc., as typical in prior art systems. *Id.* at 1:43–2:15. Lindskog then discloses in the "DETAILED DESCRIPTION OF THE INVENTION" section improving upon those centralized systems by decentralizing some of

[15] Dr. Almeroth testifies that based at least on CORBA (a standard communication mechanism) and GRAFCET (a standard for sequential control), "a POSITA would understand that a common way to implement agents is through software," and "a POSITA would understand that the agent pairs are, at a minimum, implemented in a combination of software running on hardware." Ex. 1001 ¶ 109 (citing Ex. 1007, 7:56–62, 11:64–67).

the functionality in a hierarchical node and agent based system. *See id.* at 4:37–57. Therefore, Lindskog's teachings advance the state of the prior art without limiting the improved system to monitoring or using only hardware, contrary to Patent Owner's arguments. *See* PO Sur-reply 5 (arguing Lindskog only discloses "two short passages" and the first one (Ex. 1007, 2:15–16) appears in the "Description of Related Art" section, and the second one (*id.* at 3:20–23) "shows that the logical resources are not software").

At the first passage cited by both parties, Lindskog discuses as part of the "BACKGROUND OF THE INVENTION" section, that "[d]etailed fault information (e.g., alarms), *for both hardware and software faults*, is also gathered by the various network elements and is sent up to a centralized fault management node, which is responsible for alarm filtering and alarm correlation." Ex. 1007, 2:16–20. Later, at the second passage cited by both parties, Lindskog discusses supervising "one or more network resources, which can comprise logical resources (e.g., a cell border, a location area, etc.) and/or physical devices." *Id.* at 3:20–23.

Therefore, contrary to Patent Owner's arguments, as Petitioner argues, Lindskog does not limit logical resources to the listed examples and implies software by referring to "and/or physical devices" and earlier describing software faults in the "BACKGROUND OF THE INVENTION" section. *See id.*; Reply 4–5. Lindskog also describes analyzing "the received *alarm data* to identify the cause of the alarm and to determine if the underlying fault that caused the alarm can be handled at the current node." *Id.* at 3:28–32; *see also id.* at 11:11–12 ("The FA12 of RNC1 correlates (in an event generator 44) *the received data* to determine the cause of the alarms." (emphasis added)). In context, Lindskog's use of the generic term "fault" in

conjunction with receiving alarm data and processing the data and other similar teachings discloses monitoring and determining hardware and software faults on a per thread basis. In the context of Lindskog, receiving alarm data from one or more monitored resources implies running software threads (when viewed in light of the knowledge of a PHOSITA). *See, e.g.*, Ex. 1001 ¶ 111 ("A POSITA would further understand that one or more network resources in a single supervised domain are the execution of one or more sections of software code in the runtime environment of the supervised domain.").

According to Petitioner's persuasive showing, as discussed above, Lindskog's nodes each include a runtime environment that allows agents within an agent pair to communicate with one another (and also monitor underlying threaded resources for faults as discussed below in connection with limitation [c]), thereby teaching or suggesting the running of at least one thread in a first runtime environment. *See* Ex. 1007, 3:36–39; Pet. 37–38 (citing Ex. 1001 ¶¶ 111–112; Ex. 1007, 2:15–16, 3:20–23; Ex. 1001 ¶¶ 111–112). By employing specific fault agents in a node tied to one or more specific alarms, faults, operational parameters, underlying agents, or software resources, below the node (Ex. 1007, 5:66–6:14), this teaches or at least suggests (in combination with Turek) running at least one thread in a first runtime environment at that node. *See* Pet. 33, 37–41. As Petitioner also explains, resolving one or more selected faults independently and not resolving other non-selected faults implies or teaches distinct threads in a runtime environment. *Id.* at 38–39 (citing Ex. 1007, 5:60–65; Ex. 1001 ¶ 112).

As noted above, Petitioner also shows that Turek discloses and suggests running at least one thread using a software agent or agents in a run-time environment. *See* Pet. 40 ("*Turek* discloses that 'at each node, ***the software agent is preferably run by the runtime engine***.'" (quoting Ex. 1010, 2:63–65)), 41 (citing Ex. 1001 ¶ 113), 44–45 (contending Turek's automatic deployment of one or more agents in a multi-threaded network to monitor operational parameters suggests per-thread utilization); Ex. 1001 ¶ 113 (testimony, supported by citations to Turek, showing that Turek suggests threads in both the background runtime engine and the software agents); Ex. 1012, 6:9–10, 38–48.

As Petitioner also persuasively shows, Turek teaches employing agents of stand-alone Java applications or applets, running in a multi-threaded, platform independent, programming environment, suggesting a run-time environment including an engine at each node (or domain) in Lindskog's system and including the running of a resource thread (of a monitored resource). *See* Ex. 1012, 6:38–48; Pet. 27, 32, 33 ("A POSITA would understand that multiple threads can run in parallel using multithreading techniques, so multiple resources running as threads could be implemented concurrently in parallel to achieve a corrective measure, thus saving time and more efficiently resolving detected faults.") (citing Ex. 1001 ¶ 103), 40 ("*Turek* additionally discloses that 'both the runtime engine and ***the software agent(s)*** are conveniently written in Java' and that 'as is known in the art, Java is an object oriented, *multi-threaded*, portable, platform-independent, secure programming environment used to develop, test and maintain software programs.'" (quoting Ex. 1012, 6:38–44)). After relying on both Turek's agents and runtime environments as Java-oriented multi-

threaded platforms, Petitioner persuasively concludes that "[i]t would have been obvious to modify *Lindskog's* fault management system so that each of the agent pairs operates in *Turek's* multithreaded runtime environment architecture so as to achieve the predictable result of scalable fault management in a distributed, multi-threaded network and to improve network efficiency." Pet. 41.

In other words, Petitioner shows persuasively that it would have been obvious to employ Turek's thread-based software agents in Lindskog's agent based system as a "known technique" including the use of "a multithreaded agent architecture in order to achieve a more robust network management system" and to improve efficiency and for scalability. *See* Pet. 30, 41. Petitioner asserts other valid and persuasive reasons, including that "a POSITA would recognize that in a multithreaded architecture, a process can still run even if certain threads are blocked/consumed by other tasks." *Id.* at 31 (citing Ex. 1001 ¶ 104).[16] And Petitioner also persuasively contends it would have been obvious to run resources as threads (in addition to agents and runtime environments): "A POSITA would understand that multiple threads can run in parallel using multithreading techniques, so multiple resources running as threads could be implemented concurrently in parallel to achieve a corrective measure, thus saving time and more efficiently resolving detected faults." *Id.* at 33 (citing Ex. 1001 ¶ 103).

As Petitioner notes, Patent Owner "does not dispute that *Turek* teaches a 'runtime environment.' . . . . Instead, [Patent Owner] argues that

[16] This Final Written Decision further addresses Patent Owner's arguments regarding motivation to combine Turek below. *See infra* § II.E.1.i.

Turek's software agents are not 'threads' because they 'are not deployed into the network to its nodes until *after* an event is detected by the central dispatch mechanism' and thus cannot monitor operational parameters to detect abnormalities." Reply 6 (citing or quoting PO Resp. 18–22, 25–26); *see* PO Sur-reply 6 (arguing that "Turek's agents cannot be the claimed threads because they are not deployed into the network to its nodes until *after* an event is detected by the central dispatch mechanism").

Petitioner persuasively replies as follows:

> Petitioner relies on a subsequent ***fault origination stage*** where the multi-threaded software agent "arrives" and is run by the runtime environment "at a given node" (steps 66–68 of Figure 6). Ex. 1012, 8:24–27, 2:63–65; Pet. at 40. [The Petition] explain[s] that *Turek*'s multi-threaded software agent that is run by the runtime environment discloses a "thread" running within a first runtime environment. Pet. at 40.

Reply 6.

In other words, Petitioner relies on the combined teachings of running Lindskog's agents in a threaded architecture to monitor resource threads prior to and during fault detection as suggested by Turek (even though Turek deploys its mobile agents after detecting a fault at an undetermined node).[17] *See* Reply 12 (asserting that Patent Owner's argument "that *Turek's* agents do not monitor operational parameters because they are deployed only after a fault occurs. . . . is irrelevant to Petitioner's case." (citing PO Resp. 33)); Reply 6 (relying on Turek's monitoring regardless of the timing of agent deployment: "*Turek's* multi-threaded software agent that is run by the

[17] Like Turek, the '659 patent similarly describes "*mobile* agents [that] can migrate under their own control from machine to machine in a heterogeneous network." Ex. 1003, 8:11–12 (emphasis added).

runtime environment discloses a 'thread' running within a first runtime environment."); Pet. 40–41 (asserting the obviousness of running threaded software agents in a threaded runtime environment); Pet. 31 ("In particular, a POSITA would recognize that implementing *Lindskog's* network resources as threads would minimize the bandwidth consumed by, without sacrificing the performance of, a distributed network management system) (citing Ex. 1001 ¶103)).[18]

Based on the combined teachings of the references, as discussed above, Petitioner persuasively shows that running Lindskog's agents to monitor software threads of resources in a multi-threaded system would have been obvious in order to run concurrent parallel operations for making corrections, thereby saving time and promoting efficiency, where Lindskog and Turek disclose the use of agents for monitoring hardware or software. *See* Pet. 30–33, 41–44. Accordingly, Petitioner shows persuasively that Turek suggests running threaded agents in threaded runtime environments at individual nodes of Lindskog's system to monitor threaded resources, where Turek teaches that threaded Java applets in a multithreaded architecture or runtime environment were well-known. *See* Pet. 40–41 (citing Ex. 1001 ¶ 113; Ex. 1012, 2:33–36, 63–65, 3:65, 6:38–44, Fig. 2); Ex. 1012, 6:9–10, 38–48.

After considering the full record including reasons to combine the references as discussed further below, Petitioner persuasively shows that the

[18] As discussed below (*infra* § 2.E.1.i (motivation)), Turek deploys its multi-threaded AREs prior to deploying its mobile agents.

combination of the Lindskog references and Turek renders obvious the subject matter of this limitation.

c. Limitations [c] ("monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread") and [d] ("detecting if there is an abnormality in the monitored operational parameters")

For limitation [c], Petitioner relies on Lindskog (with the above-noted incorporation of Lindskog II and Lindskog III) or the combined teachings of Lindskog and Turek. Pet. 41–45. To a certain extent, the findings regarding the "running at least one thread" limitation [b] in the previous sections overlap with some findings with respect to limitations [c] and [d], as arguments by the parties show.[19]

With respect to limitation [c], Petitioner contends that "[e]ach agent pair [in Lindskog] specifies which events (e.g., operational parameters) associated with what network resources (e.g., threads) the pair desires to monitor via a subscription request," and therefore "*Lindskog's* agent pairs can monitor operational parameters of these threads (software-based network resources)." *Id.* at 42–43 (citing Ex. 1007, 4:67–5:1, 5:6–10, 6:17–22, 12:60–63, Fig. 2). Petitioner also asserts that Lindskog III describes monitored operational parameters. *Id.* at 42 (citing Ex. 1010, 5:28–35).

Petitioner further asserts that "*Lindskog* further teaches that the monitored operational parameters can include per-thread utilization,"

[19] Many of Patent Owner's arguments regarding limitations [c] and [d] overlap with arguments regarding limitation [b]. To avoid unnecessary repetition, we incorporate our findings and rationale across the sections of the Final Written Decision.

including the incorporation or suggestion of Lindskog II's "***real time performance information***" for "overload" protection, and Lindskog III "***raw traffic data***." *See* Pet. 43–44 (emphasis by Petitioner) (quoting Ex. 1008, 4:51–54; citing Ex. 1008, 9:28–30, 12:56–59). Petitioner also cites to Lindskog II's teachings of "[n]etwork overload protection control . . . to ***protect various network resources from overload caused by too high traffic levels*** being offered." *Id.* at 43 (emphasis by Petitioner) (quoting Ex. 1008, 9:28–30). Petitioner contends that "*Lindskog II* further explains that '[t]he ***detection of overload on each level can be based on measurements directly from the managed resources (in network 26)***.'" *Id.* at 43–44 (emphasis by Petitioner) (quoting Ex. 1008, 12:56–59). Petitioner adds that Lindskog II additionally discloses that the agents "receive ***real time performance information*** associated with the distributed communications network from a plurality of performance agents." *Id.* at 44 (emphasis by Petitioner) (quoting Ex. 1008, 5:51–54). And Petitioner points to Lindskog III, as teaching that "this performance information includes '***raw traffic data*** which is used as the basis for performance data.'" *Id.* (emphasis by Petitioner) (quoting Ex. 1010, 4:14–15).

Based on these teachings, Petitioner contends "a POSITA would understand that *Lindskog* monitoring per-resource traffic and load data is monitoring 'per-thread utilization'" as claimed. *Id.* at 44 (citing Ex. 1001 ¶¶ 115–116).

With respect to Turek, Petitioner asserts it "teaches monitoring operational parameters including per-thread utilization," because "[a]s Turek explains, . . . 'the present invention envisions ***automatic deployment of one or more software agents to locate a particular network fault***, and the use of

such agent (once the fault is located) to rectify the problem.'" Pet. 44 (quoting Ex. 1012, 6:22–30). Petitioner also relies on management tasks by Turek, with Turek "disclos[ing] that 'the system management framework facilitates execution of system ***management tasks required to manage the resources*** in the [managed region]" and that 'such tasks are quite varied and include, without limitation, file and data distribution, ***network usage monitoring***, user management, printer or other resource configuration management.'" *Id.* at 44–45 (emphasis by Petitioner) (quoting Ex. 1012, 4:22–29).

Petitioner states that "[i]t would have been obvious to modify *Lindskog's* monitoring of operational parameters of network resources to including the 'network usage monitoring' of *Turek* in order to further identify, diagnose, and correct utilization-related faults and other conditions at each network resource in the distributed network." Pet. 45 (citing Ex. 1001 ¶¶ 114–118). In summary, Petitioner contends that running threaded runtime engine and agents to diagnose and rectify specific faults in Lindskog's software threaded resources as suggested by Turek would have rendered obvious the claim step of "monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread." *See id*. at 45–46.

Limitation [d] refers back to limitation [c] and recites "detecting if there is an abnormality in the monitored operational parameters." Accordingly, limitation [d] alternatively allows an abnormality detection of one or both of two major categories in the "monitored operational parameters." The first more generic type includes an abnormality in "operational parameters relating to the each thread" as limitation [c]

specifies. The second more specific type requires an abnormality in monitored "operational parameters relating to the each thread" for operational parameters of "a per-thread utilization for the each thread" as limitation [c] also specifies. As discussed below, Petitioner persuasively shows that the references teach or suggest both types.

Petitioner relies on Lindskog or the combined teachings of Lindskog and Turek to teach limitation [d]. Pet. 45. Petitioner contends that Lindskog's "agent pairs monitor events and alarms (i.e., monitored parameters) to 'identify faults that occur in the system 10, so that corrective actions can be taken.'" *Id.* at 45–46 (quoting Ex. 1007, 5:5–9). Petitioner submits that Lindskog's "event generator 44 is the intelligence of the agent 12 or 14' and is used to determine whether the monitored operational parameters and events constitute a problem." *Id.* at 46 (quoting Ex. 1007, 7:44–45). Petitioner explains that "[t]he event generator 44 consumes the received events, one after another, and ***performs event filtering and correlation.*** In other words, ***the event generator 44 keeps track of whether the requirements for a particular event have been fulfilled***." *Id.* at 46 (emphasis by Petitioner) (quoting Ex. 1007, 7:40–43).

Petitioner also explains that to determine fault, "a particular event might require the occurrence of three different events at certain subordinate FAs 12" or "[i]n other cases, the requested event might merely require the existence of a specific occurrence." Pet. 46 (quoting Ex. 1007, 6:61–65). Petitioner also explains that Lindskog III's agents detect if a failure exists by considering when "traffic machines . . . detect a hardware/software fault or . . . when a performance threshold is crossed." *Id.* at 42 (quoting Ex. 1010, 5:34–35; citing Ex. 1001 ¶¶ 119–121).

Petitioner also relies on Turek's teachings of performing a test at a dispatch mechanism and further cites to Turek's "software agent" for performing "fault detection and diagnosis." *See* Pet. 46 (citing Ex. 1012, 7:1–4, Fig. 6). According to Petitioner, "[i]t would have been obvious to modify Lindskog's fault detection method to incorporate *Turek's* test to determine if an event constitutes a network fault in order to increase *Lindskog's* ability to ascertain such conditions." *Id.* at 47 (citing Ex. 1001 ¶¶ 119–121).

Patent Owner contests Petitioner's assertions regarding limitations [c] and [d], arguing that

> the 'alarms or events' that are received by the FA-CA pairs of Lindskog I are not "operational parameters relating to . . . thread[s]," and Lindskog I does not disclose that there are ever 'abnormalit[ies]' in the 'alarms or events such that 'an abnormality in the monitored operational parameters' could be detected, as required by claim limitation [d].

PO Resp. 27. According to Patent Owner, Lindskog's FAs "do not actually monitor any operational parameters" because "the FAs merely 'receive[] alarms or events,' Ex. 1007 at 5:6, that are in-fact 'generated in the network resources 18,' *id.* at 5:18." *Id.* (citing Ex. 2001 ¶ 63). In a footnote, Patent Owner also argues that "[t]he fact that the alarms are generated in the network resources 18 highlights yet another issue with Petitioner's identification of the network resources 18 as the claimed 'threads.'" *Id.* at 28, n.6. Patent Owner explains that "[t]he network resources 18 cannot be both the threads (which are 'the execution of a section of code independently within a software program') as well as the element that monitors operational parameters of the claimed threads." *Id.*

Regarding the latter argument, and as partly discussed in connection with limitations [b], Petitioner primarily and persuasively relies on the combined teachings of the Lindskog references and Turek to teach the threaded agents as monitoring the claimed threads running as software threads in a network resource connected to the monitoring agents. *See* Pet. 31–33, 36–45. Patent Owner's other arguments attempt to restrict "monitoring of operational parameters relating to the each thread" to exclude receiving alarm data specified by Lindskog's subscription requests and also do not account for Petitioner's reliance on monitoring real time or raw data in the Lindskog references. *See* Pet. 41–43; Ex. 1007, 6:17–22. Even if monitoring operational parameters excludes receiving the subscribed software fault alarm data described in Lindskog, Petitioner persuasively shows that the Lindskog references collectively teach monitoring real or raw data and determining abnormalities in the data, for example, "***when a performance threshold is crossed***." *See* Pet. 46 (emphasis by Petitioner) (quoting Ex. 1010, 5:28–35).

Moreover, Patent Owner concedes that Lindskog II teaches monitoring resources "based on *measurements directly from the managed resources*," but relies on its assertion that the "monitoring is only of resources as a whole" (i.e., instead of "relating to the each thread"):

> Lindskog II discloses that its monitoring is only of resources as a whole, not of parameters relating to "the execution of a section of code independently within a software program." Lindskog II describes monitoring "to protect various network resources from overload caused by too high traffic levels being offered." Ex. 1008 at 9:28–31. Accordingly, Lindskog II describes that "[t]he

> detection of overload on each level can be based on *measurements directly from the managed resources*." *Id.* at 12:56–59.

PO Resp. 30 (emphasis added).

Again, Patent Owner's "monitoring as a whole" argument does not account for Petitioner's persuasive showing, as discussed partly above in connection with limitation [b], that it would have been obvious to run and monitor Lindskog's resources as threads "to achieve a corrective measure" including addressing overloads. *See* Pet. 33 ("A POSITA would understand that multiple threads can run in parallel using multithreading techniques, so multiple resources running as threads could be implemented concurrently in parallel to achieve a corrective measure, thus saving time and more efficiently resolving detected faults.") 33 (citing Ex. 1001 ¶103)), 43 (citing Ex. 1008, 9:28–30 (correcting overloads)).

Lindskog teaches monitoring a device's operational parameters at least by receiving alarm data as a result of that monitoring, and then correlating that alarm data to determine an underlying fault. Claim 1 does not specify *receiving* operational parameters, it requires *monitoring* them. So claim 1 broadly contemplates receiving raw data, real data, alarm data or other information in order to monitor operational parameters.[20]

[20] Patent Owner cites to Dr. Almeroth's testimony responding he "wouldn't think so, though it, you know, might be in some system" after being asked whether an "alarm or condition isn't the parameter, itself." *See* Ex. 2003, 26:14–18; PO Resp. 29 (quoting Ex. 2003, 26:14–18). Regarding "network usage monitoring" to detect an abnormality in operational parameters, Dr. Almeroth gave an example of monitoring "things like number of packets per second," "faults that can happen in software," and other "conditions as it relates to the particular software." *Id.* at 26:4–13. Dr. Almeroth also

The '659 patent supports this interpretation and Petitioner's showing. It broadly describes "sending information" from a monitored device (using an agent at the device) to a monitoring agent in a push monitoring embodiment: "[A]n active agent running on the [monitored] device *send[s] information out in a controlled and timely manner*." Ex. 1003, 15:13–14 (emphasis added), 56–57 ("[T]he monitoring agent assumes the device has an active agent running upon it which sends out information to its associated local modeler"). Also, "[e]ach monitoring agent . . . has the ability to alert the local modelers to *monitored responses*. These *responses are called policy events and are used to detect any aberrant behavior based on operational ranges specified by the policy*." *Id.* at 15:35–38 (emphasis added). The '659 patent also specifically describes "*alerts to be monitored.*"[21] *See also* Ex. 1003, 13:43–4–51 ("Each monitoring agent has monitoring policy which determines the operation or set of operations it should perform to monitor the device, and a set of thresholds, within which *the monitoring results* should fall. . . . Complex monitoring agents may be written *to combine monitored values in a variety of ways using threshold ranges, time periods, rates of change*, etc.").

As Petitioner notes, Patent Owner "does not dispute that a POSITA would have understood that 'alarms' '*relate to the operation of the network*

---

testified in his deposition that a parameter could be alarm data "in some system[s]," but "I'd have to see." *See* Ex. 2003, 26:18–19.

[21] "Agents use an event notification system whereby polices are setup *as alerts to be monitored* within the local modelers, which notifies the appropriate agents that a policy SLA is not within its desired operational range." Ex. 1003, 11:37–41 (emphasis added).

resource as a whole.'" Reply 8 (emphasis added) (quoting PO Resp. 27; citing Ex. 2001 ¶¶ 63–64). Stated differently, Patent Owner and Dr. Madisetti at least agree that alarms relate to operational parameters as Petitioner argues. *See* Reply 8.

In testifying about limitations [b], [c], and [d], Dr. Madisetti's testimony focuses on Lindskog's hardware instead of addressing Petitioner's showing that Lindskog discloses monitoring for software faults. *See, e.g.,* Ex. 2001 ¶ 65 ("An alarm that relates to whether or not a base station is malfunctioning describes the operation of the base station as a whole (either the base station is malfunctioning, or it is not.")), ¶¶ 47–85. Contrary to this testimony that Lindskog only determines whether "the base station as a whole" is either "malfunctioning," or "not," Lindskog specifically discloses monitoring on a much smaller individual scale, including for "single circuit" failure. *See* Ex. 1007, 5:2–3. Dr. Madisetti's testimony regarding detecting abnormalities also does not account for Petitioner's persuasive showing that Lindskog and Turek together show the obviousness of running Lindskog's resources as threads, as discussed above in connection with limitation [b]. *See id.,* Pet. 33. Without addressing these key features, Dr. Madisetti's testimony does not help to resolve key disputes here, including with respect to thread monitoring and detecting abnormalities.

In its Sur-reply, Patent Owner repackages its argument above and contends that "[b]ecause the claims require 'detecting if there is an abnormality in the monitored operational parameters,' Lindskog's 'alarms' and 'events' cannot be 'operational parameters' because Lindskog never discloses detecting abnormalities in the alarms and events themselves." PO Sur-reply 7. Similarly, Patent Owner argues that "an alarm is what identifies

that an abnormality exists somewhere else in the system" and "by the time an alarm reaches the event generator, it is merely something that exists indicating that a fault was detected elsewhere." *Id.* at 13 (citing Ex. 1007, 7:40–49). Patent Owner also contends that Lindskog's event generator 44 simply communicates messages during filtering and correlation, which does not pertain to monitoring and detecting. PO Resp. 36. In its Sur-reply, Patent Owner similarly contends that "when an FA performs event filtering and correlation it is 'tracking' its own handling of events." Sur-reply 9 (citing Ex. 1007, 7:45–19; Ex. 2003, 108:18–20). Patent Owner also contends that "[a]n FA's event filtering and correlation can be used where the events relate to the FA's handling of an alarm, for example, to determine the cause of alarm after-the-fact." *Id.* (citing Ex. 1007, 11:10–14).

Contrary to these arguments, regardless of whether the event generator communicates messages during filtering and correlation, it monitors operational parameters by filtering and correlating alarm data to detect abnormalities. As Petitioner explains and as discussed above, Lindskog's agent's event generator 44, "the intelligence of the agent" (Ex. 1007, 7:44–45), receives alarm information that event generator 44 processes during filtering and correlation so that it can decide to correct the fault or not (i.e., determine if an abnormality exists). *See* Ex. 1007, 5:6–10, 5:42–65, 7:40–56, Fig. 2; Pet. 45–46; Reply 14–16.

Correlating and filtering the event (alarm) data ordered by the CA to determine the cause of an underlying fault shows an abnormality in a (correlated) pattern of the monitored operational parameters, including an abnormality in the pattern of the alarm data itself. *See* Ex. 1007, 6:17–22, 7:45–50, 8:55–58; Reply 10–11; Pet. 42–46. For example, Lindskog teaches

that "the appropriate CAs 14 . . . are able to request and receive alarm information (events), in response to which the receiving CA 14 is potentially able to take the necessary corrective actions." Ex. 1007, 8:55–58. The necessary corrective actions signify an abnormality in the alarm data and its underlying operational parameters—hence the correction. And as Petitioner argues, Lindskog teaches that a "requested event might require the occurrence of three different events at certain subordinate FAs 12," but "[i]n other cases, the requested event might merely require the existence of a specific occurrence at the FA12 or network resource 18 that receives the request." *Id.* at 6:63–65; Pet. 46 (quoting Ex. 1007, 6:61–65). Similarly, as part of the decision making process in Lindskog to determine the cause of a fault, a "simple type of alarm filtering" includes "a decision not to act at all." Ex. 1007, 5:54–55; *see* Pet. 38–39 (quoting Ex. 1007, 5:50–55 (i.e., portions of the same sentence in Lindskog)). Pet. 46 (quoting Ex. 1007, 6:61–65).

Accordingly, as Petitioner shows, Lindskog describes complicated and simple filtering and correlating of a pattern alarm or event data to determine an abnormality in that data thereby further signifying an abnormality in the operational parameters underlying that received data in order to perform a corrective action. *See* Ex. 1007, 5:42–55, 7:40–62. Stated differently, abnormalities occur in a series or pattern group of that alarm data to signify an abnormality in monitored operational parameters underlying the received alarm data, which Lindskog's system evaluates in order to determine the cause of the fault.

Patent Owner also contends that "[b]ecause any abnormality must already have been detected prior to the deployment of an agent to a node, any testing done by Turek's agent on that node necessarily cannot satisfy the

claimed 'detecting' step." PO Sur-reply 15.[22] Contrary to this argument, as similarly discussed above in connection with limitation [b], Petitioner does not rely on Turek's teachings to show when to deploy an agent or diagnose a fault. Rather, Petitioner generally relies on the combined teachings of Lindskog and Turek to further teach or suggest detecting abnormalities in monitored operational parameters, including by detecting an abnormality in a pattern or group of alarm data or by detecting threshold crossings related to monitored raw data or real data. *See* Pet. 45–46. As noted above, Patent Owner concedes that "Lindskog II describes that '[t]he detection of overload on each level can be based on measurements directly from the managed resources.'" PO Resp. 30 (citing Ex. 1008, 12:56–59).

Further regarding Turek, Patent Owner states that "while Turek generically describes 'diagnosing and rectifying errors that arise in [a] computer network' and 'deploy[ing] . . . one or more software agents to

[22] Patent Owner asserts that Petitioner relies "only relied on Turek's disclosure of "a test . . . performed at the *dispatch mechanism*" to reach the abnormality limitation, instead of software agents running on a node. *See* PO Sur-reply 15 (citing Pet. 46–47; Reply 17). While the Petition relies on Turek's test at the dispatch mechanism as Patent Owner argues, on the same page, the Petition also specifically points to "a software agent fault detection and diagnosis process" with respect to Figure 6, showing "AGENT ARRIVES AT GIVEN NODE" at step 66. *See* Pet. 46–47 (citing Ex. 1001 ¶¶ 119–121; Ex. 7:1–4). (If Turek's test indicates a fault has occurred based on an "event" which "may comprise a plurality of alarms" and the location of the fault is unknown, the dispatch mechanism deploys a software agent into the network to perform more diagnostics and fault correction. *See* Ex, 1012, 7:16–8–9.)

locate a particular network fault,' Ex. 1012 at 6:22–30, it does not describe how the faults are actually detected or what they relate to." *Id.* (citing Ex. 2001 ¶¶ 68–69). This argument does not undermine Petitioner's showing, based on the combined teachings of the Lindskog references, and Turek, as teaching the monitoring of software resources for per-thread utilization. *See* Pet. 33.

As indicated above, the argument that Turek does not specifically disclose how to detect a fault "or what they relate to" ignores that Petitioner relies on the combined teachings of the references, including Lindskog's monitoring and correction of independent software faults as supplemented by Turek's threaded architecture and monitoring of data. The record shows that artisans of ordinary skill readily would have recognized how to detect an abnormality in per-thread utilization, for example, by comparison to thresholds for monitored data related to load balancing or overload protection. *See, e.g.*, Ex. 1008, 9:28–30; Ex. 1010, 5:28–35; Ex. 1001 ¶¶ 119–121; Pet. 41–46, 46 (citing Ex. 1007, 3:26–29, 3:52–54); Ex. 2003, 26:7–12 (Dr. Almeroth citing an example for determining an abnormality in "packets per second" of conditions or parameters in "particular software" in response to questions by Patent Owner); Reply 11 (citing Ex. 1008, 12:56–59 (discussing per-thread utilization via operating parameters including number of access requests to detect and determine overload of a network resource)).

As also discussed above and further below, Petitioner provides several persuasive reasons for monitoring resources using Lindskog's agent pairs with multiple threads and providing per-thread utilization based on Turek's threaded runtime environment, including increased robustness, speed,

efficiency, balancing overloads, and to aid in correcting and diagnosing faults. *See* Pet. 28–33, 43–45. As another more specific reason, as noted above, Petitioner contends that employing Turek's specific teaching of a test "to determine whether a given event as occurred" (i.e., a test for abnormalities) to modify Lindskog's system would have been obvious in order to enhance Lindskog's ability to detect an abnormality. *See* Pet. 46 (citing Ex. 1001 ¶¶ 119–121; quoting Ex.1012, 7:1–4).

In its Sur-reply, Patent Owner also contends that "Petitioner's Reply does not address that 'it would not be possible' to combine Lindskog with Turek because 'even if one were to make the agents of Lindskog multi-threaded, one would still be left with network resources that are not threads.'" PO Sur-reply 27 (citing PO Resp. 54–55). Patent Owner also argues that "[a] fault or an alarm relating to the traffic levels or other similar characteristics of a network device or resource does not provide any information regarding the utilization of a thread of a 'per-thread' basis." PO Resp. 32.

Again, as discussed above, these arguments ignore Petitioner's reliance on the combined teachings of Lindskog and Turek to suggest running and monitoring Lindskog's resources as threads. As also discussed above in connection with limitation [b], Lindskog discloses "that each FA-CA pair 'supervises one or more underlying network resources.'" *See* Reply 7 (quoting Ex. 1007, 4:67–51); *infra* § II.E.1 (addressing motivation). And Lindskog begins the process because "the FA or CA 'that desires event information from network resources . . . subscribes to event data . . . by sending one or more subscription requests." *See id.* (quoting Ex. 1007, 6:17–22). Therefore, according to Petitioner, "[e]ach agent pair specifies

which events (e.g., operational parameters) associated with what network resources (e.g., threads) the [FA-CA] pair desires to monitor via a subscription request." Pet. 42; *see* Reply 8 (citing Pet. 42). Petitioner correlates Lindskog's events or alarm data as including software faults in Lindskog. *See* Pet. 38; Reply 4; 1007, 2:15–16, 3:20–23, Ex. 1001 ¶¶ 111–112. Therefore, as Petitioner persuasively explains, "[a] POSITA would have understood that a logical network resource that issues 'software faults' is software and is a 'thread.'" Reply 5 (citing Pet. 11, 38; Ex. 1001 ¶¶ 111–112).

Petitioner also persuasively asserts that "threads, as construed, are disclosed by *Lindskog's* software-based logical resources and *Turek's* multi-threaded software agents." *See* Reply 1. Combining Turek with Lindskog and asserting obviousness, and as noted above, Petitioner persuasively shows that "[a] POSITA would understand that multiple threads can run in parallel using multithreading techniques, so *multiple resources running as threads* could be implemented concurrently in parallel to achieve a corrective measure, thus saving time and more efficiently resolving detected faults. Pet. 33 (emphasis added) (citing Ex. 1001 ¶ 103).

At the cited paragraph, Dr. Almeroth testifies that programming Lindskog's "resources as threads to the extent they are not already implemented in this manner," in view of Turek, "would have been a simple matter of applying a known technique (i.e., multithreading) to the network resources to yield a predictable result—a more efficient system—with a high expectation of success." Ex. 1001 ¶ 103. Dr. Almeroth explains further:

> In particular, a POSITA would have understood that the employing multi-threading with Lindskog's fault management

> system would result in improved processing power utilization, because multi-threading enables simultaneous thread execution. For example, if Lindskog's network resources were running as threads, multiple resources could be executed simultaneously *to detect or correct faults in the managed system, thus improving utilization*.

Ex. 1001 ¶ 103 (emphasis added).

Focusing on a "POSITA['s]" knowledge armed with Lindskog's and Turek's teachings, Petitioner explains how Lindskog's network resources imply or suggest threads:

> Because they can comprise "logical resources" and involve "software faults," a POSITA would understand that *Lindskog's* network resources can include the execution of at least a section of software code within a program, therefore disclosing the claimed threads. Ex. 1001 ¶111–112. A POSITA would further understand that these threads associated with the network resources could execute in the runtime environments described immediately above for running code. Ex. 1001 ¶¶ 111–112.

Pet. 37. Therefore, Petitioner persuasively shows that it would have been obvious to run and monitor Lindskog's network resources as threads, where Lindskog discloses monitoring information alarms for software and hardware faults, thereby implying logical resources as including threads. *See also id.* at 38 ("*Lindskog* also explains that '***a fault agent receives alarm data***, either from a subordinate fault agent or ***from a network resource***'" (quoting Ex. 1007, 3:26–28), "and that 'fault information (e.g., alarms)' can be 'for both hardware ***and software faults***.'" (quoting Ex. 1007, 2:15–16)). As Petitioner also persuasively argues, Lindskog "discloses that network resources can comprise '***logical resources***' that are the claimed threads, as

construed, because they issue 'software faults.'" Reply 8–9 (citing Ex. 1007, 2:15–16, 3:20–23; Pet. 38; PO Resp. 23).[23]

Further regarding "utilization," this limitation generally relates to comparing resources of a thread relative to another thread—e.g., with the goal of performing load balancing. For example, the '659 patent teaches that "each ARE has the ability to provide feedback on the per-thread utilization of each thread running within it. . . . to, for example, help determine load balancing on the AREs." Ex. 1003, 11:27–30; Tr. 17:17–19:3 (Patent Owner discussing the passage in response to questions by the Board).[24]

As indicated above, addressing this "utilization" limitation, Petitioner contends that the Lindskog references teach handling overload problems—similar to the load balancing problem addressed in the '659 patent. *See* Pet. 43–44 (citing Ex. 1008, 9:28–39). For example, Petitioner argues that Linkskog II "explains that '[t]he ***detection of overload on each level can be based on measurements directly from the managed resources (in network 26)***.' [Ex. 1008] 12:56–59. *Lindskog II* additionally discloses that the agents

[23] To the extent Petitioner also contends that "network resources . . . are the claimed threads" (*see, e.g.*, Reply 25), the Board interprets such statements as short-hand for Petitioner's showing that Lindskog's network resources comprise threads—execution of independent software code that issues software faults that agents monitor. *See, e.g.*, Pet. 37; Reply 9.

[24] In response to how the '659 patent measures per-thread utilization, Patent Owner stated at the Oral Hearing that "in the context of thread utilization, that's a fairly established concept particularly when it comes to something like CPU utilization in the thread, how much of a CPU's time is being utilized by a thread or how much of the physical resource is being utilized by that thread." Tr. 19:17–24. The challenged claims do not recite CPUs.

'receive ***real time performance information*** associated with the distributed communications network from a plurality of performance agents.'" *Id.* at 43–44 (citing Ex. 1008, 5:51–54). Petitioner adds that Linkskog III's system monitors "***raw traffic data*** which is used as the basis for performance data." *Id.* (quoting Ex. 1010, 4:14–15). According to Petitioner, "a POSITA would understand that *Lindskog* monitoring per-resource traffic and load data is monitoring 'per-thread utilization.'" *Id.* at 44 (citing Ex. 1001 ¶¶ 115–116).

Petitioner also relies on Turek to teach "***network usage monitoring***," and "execution of system ***management tasks required to manage the resources*** in the [managed region]." Pet. 45 (quoting Ex. 1012, 4:22–29). And as explained above, Petitioner relies on Turek's "runtime engine [that] interacts with a software agent for this purpose." *Id.* at 44 (quoting Ex. 1012, 6:22–30). Based on these teachings and the teachings outlined in connection with running resources as threads according to limitation [b], Petitioner contends that "[i]t would have been obvious to modify *Lindskog's* monitoring of operational parameters of network resources to including the 'network usage monitoring' of *Turek* in order to further identify, diagnose, and correct utilization-related faults and other conditions at each network resource in the distributed network." *Id.* at 45.

Patent Owner challenges this motivation and argues in its Sur-reply it is "conclusory" and "unsupported," and the Petition provides "no explanation how a POSITA would successfully accomplish such a modification." Sur-reply 12. Contrary to Patent Owner's Sur-reply argument, prior to advancing the particular motivation statement, Petitioner quotes or cites Turek as disclosing "run time engine 41," "software agents," and "multi-threaded" architecture, and also quotes Turek for its "***network***

usage monitoring," as indicated above. *See* Pet. 44–45 (emphasis by Petitioner) (quoting Ex. 1012, 4:22–29, 6:22–30; citing Ex. 1001 ¶¶ 114–118). Lindskog describes determining and correcting the cause of a fault as noted above, and Petitioner cites Lindskog II for its monitoring of "real time performance information" and for its teaching of "***protect***[ing] ***various network resources from overload caused by too high traffic levels*** being offered." Pet. 43 (emphasis by Petitioner) (quoting Ex. 1008, 5:51–54, 9:28–30).

These combined teachings show that artisans of ordinary skill would have been able to implement the modifications set forth by Petitioner by monitoring "real time performance information" (Ex. 1008, 9:28–30) or "raw traffic data" (Ex. 1010, 4:14–15) and applying the well-known techniques for fault determination that Lindskog and Turek describe (*see supra* §§ II.D.1–4), for the reasons advanced by Petitioner. And as noted above, Patent Owner argued during the Oral Hearing that "thread utilization" is "a fairly established concept particularly when it comes to something like CPU utilization in the thread, how much of a CPU's time is being utilized by a thread or how much of the physical resource is being utilized by that thread." Tr. 19:17–24; note 24. Petitioner's showing applies well-known thread utilization techniques to well-known threaded resources for the known purpose of correcting utilization related faults like overloads. *See* Pet. 43–45 (citing Ex. 1001 ¶¶ 114–118).

In addition, during his deposition, in context of the reference teachings at issue here and his declaration testimony, Dr. Almeroth testified that

> examples of operational parameters could be conditions as it relates to the particular software. There's examples of faults that can happen in software. So operational behavior. Abnormalities. So network performance as it relates to network usage monitoring. So it could be things like number of packets per second.

Ex. 2003, 26:7–12.

Dr. Almeroth also testified that in the context of Lindskog III, "performance data based on raw traffic data is an example of an operational parameter" (Ex. 2003:21–22):

> the raw traffic data is used as the basis for performance data. So it's the performance data, itself, which would be the operational parameter. And it could be based on, say, statistics about the raw traffic data; but I wouldn't think the raw traffic data, by itself, would be an operational parameter. That's not what I've described here from Lindskog III.

Ex. 2003, 35:13–19. He explained further as follows:

> So, for example, if you have processes that are responsible for processing raw traffic data, and as a result of an analysis of that raw traffic data you're able to determine the per-thread utilization of the thread that is operating on that traffic data, then that would be a good example that a person of skill in the art would understand from the collective disclosures here.

Ex. 2003, 36:7–13.

Patent Owner also argues in its Sur-reply that Lindskog III's monitoring of raw traffic data involves events data generated "by traffic machines when they detect a hardware/software fault." PO Sur-reply 9 (quoting Ex. 1010, 5:28–35). This statement further shows monitoring of software faults and supports Petitioner. Nevertheless, Patent Owner contends that Lindskog III's "traffic machines" correspond to Lindskog's "network resources," which Lindskog's FAs monitor, so "the traffic

machines cannot be both what does the monitoring and what is being monitored." PO Sur-reply 10.

Patent Owner's arguments overly restrict Lindskog III's teachings and fail to undermine Petitioner's showing. Dr. Almeroth testified, as noted above, that "the raw traffic data is used as the basis for performance data." Ex. 2003, 35:13–19. Claim 1 does not require an agent to receive real or raw data in order to monitor operational parameters for abnormalities, as Dr. Almeroth's deposition testimony quoted above implies. Lindskog III states that "[e]vents relating to failure conditions are called exception events and these events are generated either by the traffic machines when they detect a hardware/software fault, or by an agent 10 when a performance threshold is crossed." Ex. 1010, 5:31–35. As set forth by Petitioner, Lindskog (*see* claim 8 thereof), Lindskog II, and Lindskog III each generally disclose and suggest monitoring of real time, raw data, or alarm data, as well-known ways to determine faults and monitor operational parameters, with Lindskog, for example, disclosing that agents monitor operational parameters, including by receiving "real time fault data" (Ex. 1007, 12:60–64) from "at least one network resource." *See* Pet. 43–44; Reply 10–11.

Accordingly, Petitioner persuasively shows that it would have been obvious to employ Turek's network usage monitoring using Turek's multi-threaded architecture (Ex. 1012, 4:22–29; 6:22–30) to monitor per-thread utilization in Lindskog's system on a per-thread basis "to ***protect various network resources from overload caused by too high traffic levels***" as Lindskog II teaches (Pet. 43 (quoting Ex. 1008, 9:28–30)), and "to further identify, diagnose, and correct utilization-related faults" and thereby avoid blocking or overloading threads (Pet. 45). *See* Pet. 31 ("In particular, a

POSITA would recognize that in a multithreaded architecture, a process can still run even if certain threads are blocked/consumed by other tasks." (citing Ex. 1001 ¶ 104)), 43–45.

As Petitioner also argues, Lindskog II specifically teaches measuring a load L by determining the number of access requests M, lengths of input and output queues Q, and the delay of packets I, with respect to a specific level of network resource, to alleviate overloads. *See* Reply 9 (citing Ex. 1008, 12:62–13:7). Similar to other arguments and testimony, Patent Owner argues and Dr. Madisetti testifies that this overload protection relates to monitoring of resources as a whole. PO Resp. 30; Ex. 2001 ¶ 66. In its Sur-reply, Patent Owner also contends Lindskog II's monitored network resource "is a physical device." PO Sur-reply 11.

Patent Owner's contentions do not address the combination and isolate Turek's thread teachings from the contentions. Patent Owner, citing Dr. Madisetti, concedes that "the monitoring of a network resource for the detection of 'overload caused by too high traffic' [in Lindskog II] would involve, for example, *the monitoring and detection of whether* more than a specified amount *of data* is being received by the resource." PO Resp. 30 (emphasis added) (citing Ex. 2001 ¶ 66). Petitioner applies this teaching in the context of a thread: In context, the monitored devices employ software threads to process received data as Petitioner contends, and as Dr. Almeroth testified during his deposition with regard to the combined teachings at issue here, monitoring the amount of received data in the context of load balancing correlates to monitoring a software thread for an abnormality. *See* Ex. 2003, 26:7–13, 35:13–19, 36:7–13 (quoted above).

After considering the full record including reasons to combine the references as discussed further below, Petitioner persuasively shows that the combination of the Lindskog references and Turek render obvious the subject matter of limitations [c] and [d].

d. Limitation [e] ("performing a corrective action to fix any detected abnormalities")

For limitation [e], Petitioner relies on Lindskog's fault correction and monitoring system, which includes fault agents and configuration agents to correct faults and address alarms as corrective actions for any detected abnormalities. *See id.* at 47–48; Ex. 1007, code (54), Fig. 2. Patent Owner does not challenge Petitioner's showing with respect to limitation [e]. *See* PO Resp. 35–42 (discussing limitations [d] and [f]). To the extent Patent Owner's arguments related to limitation [d] and [f] impact the "detected abnormalities" aspect of limitation [e], we incorporate our related discussion of those limitations here.

After considering the full record including reasons to combine the references as discussed further below, Petitioner persuasively shows that the combination of the Lindskog references and Turek render obvious the subject matter of this limitation.

e. Limitation [f] ("wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment")

For limitation [f], Petitioner relies on the Lindskog references and Turek. Pet. 48–53. Petitioner contends "*Lindskog* teaches requesting a

corrective policy from an entity external to the first runtime environment" as claimed. *Id.* at 53. Particularly, Petitioner argues a skilled artisan "would recognize that a lower level FA-CA agent pair [in Lindskog] running in a corresponding domain is an 'agent' operating within the first runtime environment" and "would recognize that a request [for a corrective policy] made by a lower-level agent pair to a higher level agent pair in the hierarchical control system [of Lindskog] constitutes a request to an 'entity external to the first runtime environment.'" *Id.* at 48–50, 53 (citing Ex. 1001 ¶¶ 123–124; Ex. 1007, 3:24–35, 51–63, 5:6–15, Fig. 1). With respect to Lindskog II, Petitioner asserts it discloses "superior control agents that passively delegate corrective actions," and regarding Turek, Petitioner argues "*Turek* teaches management of a distributed network consisting of '***a runtime environment*** supported on given nodes of the network.'" *Id.* at 52–53 (emphasis by Petitioner) (quoting Ex. 1012, 2:33–34; citing Ex. 1008, 7:20–27).

Patent Owner contends that Lindskog's "FA-CA pair" is not an agent "as the parties have agreed it should be construed: 'a program that performs some type of operation, which may be information gathering or some processing task, in the background.'" *See* PO Resp. 39 (citing Ex. 2001 ¶¶ 78–85). According to Patent Owner, "the FA and CA of the FA-CA pair, as suggested by their names 'fault agent' and 'configuration agent,' are each independent agents that work together." *Id.* at 39–40. Patent Owner lists different functions performed by Lindskog's FA and CA in support of its position, and also points to teachings about agents in Lindskog II and III. *Id.* at 41–42.

These arguments partially involve the interpretation of the agreed-upon claim construction of an "agent," as Patent Owner's arguments recognize. However, the agreed upon construction does not preclude "an agent" from being an agent "pair," each with different tasks.[25] As discussed above, the '659 patent describes a large number of various tasks performed by a single agent or multiple agents in an ARE. *See supra* § I.C; Ex. 1003, Fig. 7, 8:50–12:23 (describing myriad functions and structures).[26] In addition, the '659 patent specifically envisions that an agent does more than one task and encompasses "two separate agents": "*A monitoring agent may monitor more than one feature on a device*. . . . In some embodiments, the monitoring agents comprise two separate agents running sending information between each other or *it may be physically the same monitoring*

[25] Typically, "an" signifies "one or more" in patent claims. *See TiVo, Inc. v. EchoStar Commc'ns Corp.,* 516 F.3d 1290, 1303 (Fed. Cir. 2008) ("As a general rule, the words 'a' or 'an' in a patent claim carry the meaning of 'one or more.'"). "The exceptions to this rule are extremely limited: a patentee must 'evince[] a clear intent' to limit 'a' or 'an' to 'one.'" *Baldwin Graphic Sys., Inc. v. Siebert, Inc.,* 512 F.3d 1338, 1342 (Fed. Cir. 2008) (citation omitted).

[26] Each agent includes several different modules, implementing different protocols, including communication/messaging adaptor 112, domain adapter 115, policy repository 114, policy repository 116, and agent control unit 118. *See* Ex. 1003, Fig. 7, 8:50–12:23. Messaging adapter 112 may be implemented with a Transport Layer Security protocol, which "is application protocol independent." *Id.* at 8:59–60. The policy repository 114 "may be implemented using a Lightweight Directory Access Protocol . . . directory structure." *Id.* at 9:2–3. The agents include Java agents, similar to Turek's agents. *See id.* at 9:24–26.

agent acting as both sender/[l]istener." Ex. 1003, 16:18–22 (emphases added).

During the Oral Hearing, after being asked about portions of this passage in the '659 patent and whether the claim construction excludes a dual agent embodiment (*see* Tr. 35:8–12), Patent Owner stated that "we can imply that agent's plural in the first [part of the second] sentence and then the second part of the sentence relates to each of those agents individually, but I don't think we are excluding any specific embodiments, but we are working with the construction that we agreed" upon (*id.* at 35:14–18). In that same colloquy, Patent Owner explained that Petitioner

> didn't say there are two agents [in Lindskog]. There's a FA and a CA. Each of which is called an agent. Each of which is a separate software module. *Maybe they're working together to be agent, plural, which would satisfy the claim limitation*. [Petitioner] just said that they are an agent together so I think that Petitioner is indicating that to meet the claim limitation, you have to have a single agent.

Tr. 35:21–26 (emphasis added). In other words, if the agents "work together," similar to the embodiments in the '659 patent that include agents that communicate and work together as sender and receiver, Patent Owner concedes that "would satisfy the claim limitation." *See id.* And Patent Owner does not argue that Lindskog's agents do not "work together"—rather, Patent Owner admits that they "are each independent agents *that work together*." PO Resp. 40 (emphasis added).

In any event, Petitioner persuasively shows that "the FA-CA pair is a single program that performs a supervising operation," where "*Lindskog* discloses that '[e]ach FA-CA pair 16 supervises one or more underlying network resources.'" Reply 18 (quoting Ex. 1007, 4:67–5:1); *see* Pet. 48–49

(citing Ex. 1001 ¶¶ 123–124). Patent Owner argues that that the FA and CA both operate using GRAFCET function charts, but this indicates they employ the same program to interpret and process different GRAFCET function charts. *See* PO Resp. 40 (citing Ex. 1007, Fig. 4, Fig. 5). Patent Owner's Sur-reply argument that different GRAFCET charts show each agent runs a different program ignores the '659 patent disclosure that describes how each agent runs a multiplicity of different modules to handle myriad functions. *See supra* note 26. Running different GRAFCET cards corresponds to different modules using the same program language. Therefore, the FA-CA pair is an agent under the agreed-upon construction and as interpreted in light of the '659 patent specification.[27]

Most importantly, however, this claim interpretation does not raise a material issue given the recited limitation and Petitioner's showing. Only limitation [f] recites an agent. Specifically, limitation [f] recites "wherein performing the corrective action comprises first making a request for a

[27] As noted above, the parties agree that an agent is "a program that performs some type of operation, which may be information gathering or some processing task, in the background." *See supra* § II.a. Applying the normal rule for a claim construction, and similar to the claimed "an agent," the antecedent "a" in "a program" does not necessarily mean a single program. *Supra* note 25 (citing *TiVo,* 516 F.3d at 1303 ("As a general rule, the words 'a' or 'an' in a patent claim carry the meaning of 'one or more.'"); *Baldwin Graphic,* 512 F.3d at 1342 ("[t]he exceptions to this rule are extremely limited: a patentee must 'evince[] a clear intent' to limit 'a' or 'an' to 'one.'"). Nevertheless, even if it the claim construction requires a single program, Petitioner shows persuasively that Lindskog's agent pair satisfies the condition, or as explained below, Lindskog teaches the only agent recited in claim 1.

corrective policy to correct a detected abnormality from an entity external to the first runtime environment *if the corrective policy is not available to an agent operating within the first runtime environment*." For this limitation, Petitioner contends that "*Lindskog* teaches that the event generator within a particular fault agent of an agent pair makes the determination as to *whether or not its own configuration agent has the policy to correct the problem before escalating or resolving the problem*." Pet. 50 (emphasis added). So Petitioner contends the corrective policy is not available to a single agent of the pair—namely the *corrective agent* (CA) of the FA-CA pair. Also, Petitioner shows that fault agents 14 do not apply corrective policies, because they determine if the CA "has the policy," so a corrective policy would not be available to that single fault agent either. *See id.* Therefore, for two separate reasons, Petitioner relies on a single agent in Lindskog for the only agent recited in claim 1, rendering moot Patent Owner's claim interpretation arguments.

Similar to its argument that an agent runtime environment must be hosted on a device addressed above in connection with limitation [b], Patent argues that Lindskog's "domains are not 'hosted on particular network (host) device' as contemplated by the '659 Patent." PO Resp. 44 (citing Ex. 1003, 2:15–17; annotating Ex. 1003, Fig. 1). Patent Owner also argues that "because Petitioner has identified 'domains' [in Lindskog] as the claimed 'runtime environments,' a request for a corrective policy cannot be made by an FA-CA pair to another FA-CA pair that is an 'entity external to the first runtime environment.'" PO Resp. 44 (annotating Ex. 1007, Fig. 1). Patent Owner explains that "[b]oth the pair making the request and the pair the request is directed t[o] would be within the same domain of the pair that the

request is direct to." *Id.* at 44–45. Patent Owner argues that Lindskog I's and Lindskog II's "domains *are not something that is hosted on an individual device*," because "the 'domains' are logical associations of multiple devices." *Id.* at 22 (emphasis added). Dr. Madisetti testifies that "[a] person of ordinary skill art would not understand the 'domains' of Lindskog I and II to be runtime environments." Ex. 2001 ¶ 50.

First, claim 1 does not recite domains. Second, like an ARE, the '659 patent does not require a "domain [to be] hosted on an individual device," contrary to Patent Owner's argument. *See* PO Resp. 22; *supra* § II.E.1.b. To the contrary, the '659 patent describes multiple AREs in a domain: "If this ARE does not have the corrective policy then an attempt will be made to obtain the corrective policy from other agents and AREs operating within the particular functional domain." Ex. 1003, 3:53–57. This disclosure shows that a domain need not be hosted on a device, because the '659 patent allows (but does not require) each ARE to be hosted on a single device *see* s*upra* § II.E.1.b.—so multiple AREs signify multiple devices as possibly residing in a single domain of the '659 patent. Also, this disclosure shows that the '659 patent does not require a one-to-one correspondence between domains and runtime environments.

Dr. Madisetti also testifies that "[a] person of ordinary skill art would not understand the 'domains' of Lindskog I and II to be runtime environments." Ex. 2001 ¶ 50. This testimony does not contradict Petitioner's showing. Dr. Almeroth agrees that one cannot assume that "domains are the equivalent of runtime environments." *See* Ex. 2003, 131:12–13.

In any event, Patent Owner's focus on domains does not address Petitioner's specific and persuasive showing that each of Lindskog's nodes and underlying monitored resource threads, as modified by Turek's runtime engine teachings, each represent a separate runtime environment.

To support its showing, Petitioner points to flexibility in domains or "***network sub-domains***" of Lindskog II and contends that a logical association for a runtime environment includes a node with an agent pair supervising one or more software resources below it (i.e., "a thread . . . running in a first runtime environment"). *See* Pet. 37 (emphasis by Petitioner) (quoting Ex. 1008, 3:64–66; citing Ex. 1001 ¶¶ 109–111; Ex. 1007, 4:67–51). For example, Petitioner argues that "[b]ecause each agent pair can be included in a network node, and each network node may be associated with a different device, a POSITA would understand that each of *Lindskog's* network nodes has a runtime environment supporting the agent pairs running therein." *Id.* at 37 (citing Ex. 1001 ¶¶ 109–111); *see* Reply 2 (citing Pet. 37; Ex. 1001 ¶¶ 109–111).

Similarly, relying on specific teachings in Lindskog, Dr. Almeroth testifies that Lindskog "discloses that each agent pair is hosted at a particular network node and that each network node may be associated with a different device or logical entity," "each domain constitutes a 'runtime environment' for an associated agent pair," and "[a] POSITA would further understand that one or more network resources in a single supervised domain are the execution of one or more sections of software code in the runtime environment of the supervised domain." Ex. 1001 ¶¶ 110–111 (citing Ex. 1007, 2:15–16, 3:20–23, 4:60–51, 5:2–5, 5:28–46).

Similar to Dr. Almeroth's testimony, Petitioner bolsters its showing by relying on Lindskog II, which discloses using a "hierarchical control approach which is the ***decomposition of control responsibilities into cell, region, or subnetwork domains***." *See* Pet. 37 (citing Ex. 1001 ¶¶ 109–111); Reply 2 (citing Ex. 1008, 3:64–66, Pet. 37). This reference to Lindskog II's "control responsibilities" for "cell, region, or subnetwork domains" shows the versatility of Lindskog's system controlling resources (including software) underlying nodes with agent pairs, instead of requiring all the nodes in a given hierarchy to contain the same runtime environment, which Patent Owner urges. *See* PO Resp. 18–22.

Patent Owner cites Dr. Almeroth's deposition testimony as supporting Patent Owner, but it supports Petitioner and shows the flexibility of domains. *See* PO Resp. 20 (quoting Ex. 2003, 129:17–19; 131:8–11 ("Q. Does each of the CAs in Figure 4 [of Lindskog II] exist within its own domain? A. It depends. And it varies. . . . there might be instances in an implementation that's consistent with the disclosures of Lindskog II, where you might have multiple CAs within the same")). Dr. Almeroth also answered "No" when asked if one can assume that "domains are the equivalent of runtime environments?" Ex. 2003, 131:12–13.

To illustrate its point about runtime environments, Petitioner annotates Figure 1 of Lindskog as follows:

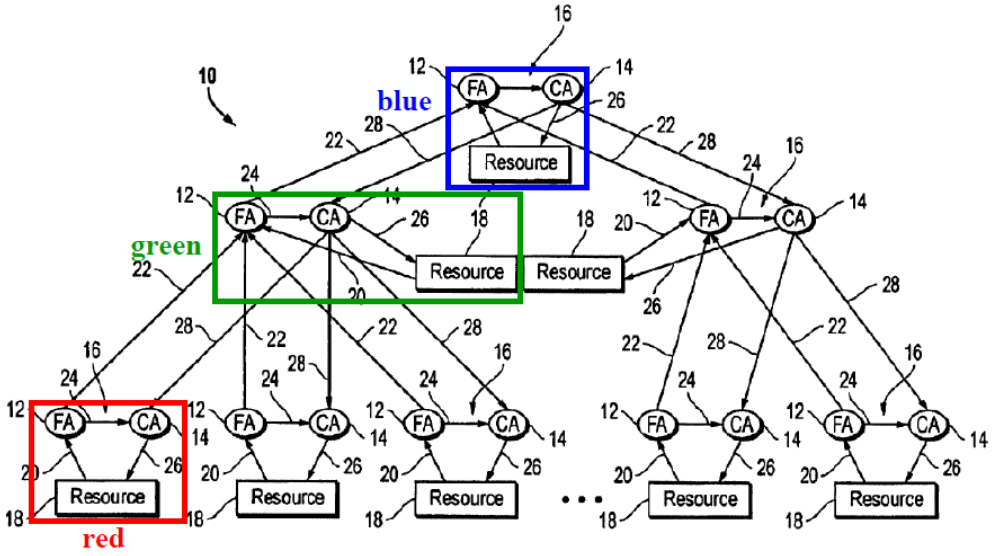

FIG. 1

As annotated above by Petitioner, Figure 1 of Lindskog shows three separate runtime environments 16 (red, green, blue), each with FA–CA pair 12 and monitored resource 18. Reply 20; *see also* Pet. 51 (similar annotation of Lindskog Fig. 1).

As annotated Figure 1 depicts, Petitioner shows that Lindskog, or the combined teachings, teach the first runtime environment, for example, including lower-level agent pair and monitored network resources within a logical domain or runtime environment. *See* Pet. 48–53; Ex. 1001 ¶¶ 123–124; Reply 21–22. Petitioner also shows that upper level agent pair 12 in runtime environment 16 (blue or green) constitutes an entity external to first runtime environment 16 (red), as limitation [f] requires. *See id.*; Reply 19–21; Ex. 1007 code (54), Fig. 1; Ex. 1001 ¶ 124; Pet. 37 ("[E]ach of *Lindskog's* network nodes has a runtime environment supporting the agent pairs running therein.").

In reference to Figure 6 of Lindskog and a discussion of domains (*see* supra § II.D.1), Lindskog states "BS12 is at the border of the domain of the

RNC1." Ex. 1007, 11:15–16. It also refers to the managed domain of "[t]he CA 14 of the RNM 62" at the top of Figure 1. *Id.* at 11:25–27. This signifies that even though the lower domain of Lindskog (starting at RNC1) falls within the "managed domain" of RNM 62 at the top, the region above the lower domain (starting at RNCI) is separate from the lower RNCI domain region. Therefore, to the extent these domains and sub-domains help to resolve the claimed external runtime issue, Figure 6 (*supra* § II.D.1) reveals that the "[t]he CA 14 of the RNM 62" is outside the domain of the RNC1 at the bottom, as Petitioner essentially argues with respect to annotated Figure 1. *See* Reply 19–22 (asserting, for example, the blue domain includes an entity external to the first run time environment of the red domain).

Dr. Madisetti produces the following annotation of Lindskog Figure 1 (Ex. 2001 ¶ 87):

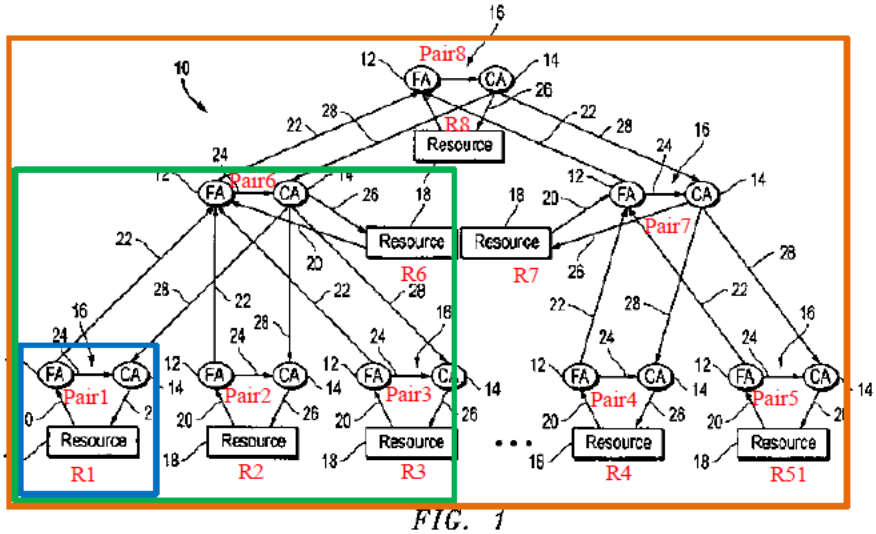

FIG. 1

Figure 1, as annotated by Dr. Madisetti, reveals "multiple overlapping 'domains' in the system of Lindskog I," and includes resources R1–R8 and FA-CA pairs (agents) as Pair1–Pair8. Ex. 2001 ¶¶ 87–88.

Dr. Madisetti testifies that Pair 1 (blue box) is inside the domain of Pair 6 (green box) so that they are "within in the same domain (the domain boxed in green)." Ex. 2001 ¶ 90 (emphasis omitted). This testimony is not relevant to a claim limitation or to Petitioner's showing.

As Petitioner argues, even Dr. Madisetti's testimony shows that Pair 6 is outside the blue domain of Pair 1. *See* Reply 21–22 (color emphasis omitted) ("The blue and green domains are different domains because Pair6 is in the green domain but not in the blue domain. . . . Accordingly, when Pair1 in the blue domain sends an alarm (request) to Pair6, the request is *external* to the blue domain"). Accordingly, limitation [f] reads on both parties' annotated versions of Figure 1. *See* Ex. 2001 ¶¶ 87–90; Reply 21–23.

Patent Owner argues in its Sur-reply that "Petitioner relies on Lindskog's 'domains' rather than its 'nodes' as the claimed 'runtime environment.'" PO Sur-reply 23 (citing Reply 1). The record does not support Patent Owner's argument for the reasons noted above and further below with respect to employing Turek's runtime engine at nodes. As determined above, and as the annotated version of Lindskog's Figure 1 verify (Reply 20; Pet. 51), Petitioner consistently refers to Lindskog's nodes as demarcating the upper bound for a runtime environment with underlying monitored resources as the lower bound.

Patent Owner alternatively contends that "if Petitioner were to change its identification of the 'runtime environment' to Lindskog's nodes, Lindskog would no longer disclose 'running at least one thread in a first runtime environment . . . each of the runtime environments running multiple threads.'" PO Sur-reply 23–24. Patent Owner explains that "Lindskog's

'network resources' are not part of or run on its nodes. The nodes are separate from and 'supervise one or more network resources.'" *Id.* at 24 (quoting Ex. 1007, 3:19–24 (nodes "generally include[] a [FA] and an associated [CA]")). Again, these arguments do not address Petitioner's showing that relies on a runtime environment to include a node and a monitored software resource below the node. *See* Pet. 37–39, 48–53. Reply 19–22.

In addition, Petitioner bolsters its reliance on nodes by turning to Turek's disclosure of multi-threaded runtime environments at each node running a corresponding software agent to supplement Lindskog's teachings and modify its architecture, with the combination persuasively suggesting runtime environments at each node to ensure a scalable and efficient system including multitasking. *See* Pet. 27, 30–33, 40, 48; Ex. 1001 ¶¶ 103–104. Similar to Lindskog, as Petitioner notes, Turek teaches "[i]f the software agent cannot rectify the problem [at the fault location], then 'a call is made at step 60 to obtain additional help.'" Pet. 26 (quoting Ex. 1010, 8:10–14). As Petitioner also notes, other than challenging the motivation to combine Turek with the Lindskog references, Patent Owner does not challenge Petitioner's showing that "*Turek* discloses 'a runtime environment supported on given nodes of the network.'" *See* Reply 23 (quoting Ex. 1012, 2:33–34).

After considering the full record including reasons to combine the references as discussed further below, Petitioner persuasively shows that the combination of the Lindskog references and Turek render obvious the subject matter of limitation [f].

f. Limitation [g] ("wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment")

For limitation [g], Petitioner contends that Lindskog's "subordinate agent pair running in a domain is an 'agent running within the first runtime environment'" as claimed, and Lindskog's "***reconfiguration data for correcting*** or otherwise reducing the effect of [a] fault" teaches the claimed "corrective policy." Pet. 54 (emphasis by Petitioner) (quoting Ex. 1007, 3:39–44, 4:50–52). Petitioner further explains that Lindskog's "reconfiguration data (i.e., corrective policy)" is received by the subordinate pair's configuration agent from a configuration agent of an upper level agent pair, and is applied at the subordinate agent pair. *Id.* at 54–55 (citing Ex. 1001 ¶ 128; Ex. 1007, 3:39–50, 4:50–52, 5:55–60).

Patent Owner does not contest Petitioner's showing with respect to limitation [g]. *See* PO Resp. 42–45.

After considering the full record including reasons to combine the references as discussed further below, Petitioner persuasively shows that the combination of the Lindskog references and Turek render obvious the subject matter of limitation [g].

g. Limitation [h] ("wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads")

For limitation [h], Petitioner relies partly on its showing with respect to limitation [f]. Pet. 55. Petitioner contends that "*Lindskog* discloses a hierarchical fault management system with a lower-level agent pair running

in a corresponding domain on the lower level (i.e., an agent running in a first runtime environment) that makes a request to a higher level agent pair running within a different domain on the higher level (i.e., an entity external to the first runtime environment)." *Id.* (citing Ex. 1001 ¶ 130). Petitioner explains that "[t]he higher level agent pair is a global modeler: it supervises multiple subordinate agent pairs and their corresponding domains and is configured to 'draw conclusions based on information from only ***a subset of its subordinate FAs***.'" *Id.* at 55–56 (quoting Ex. 1007, 5:12–14; citing Ex. 1001 ¶ 130).

Petitioner further contends that Lindskog II teaches limitation [h] via a CA having global knowledge, with Turek disclosing and suggesting "multiple threads" and "plurality of runtime environments." *See* Pet. 27, 55–58 (citing Ex. 1012, 3:67–44, 3:67–4:4, 6:38–44 ("[B]oth the runtime engine and the software agent(s) are conveniently written in Java. As is known in the art, Java is an object-oriented, multi-threaded, portable, platform-independent, secure programming environment used to develop, test and maintain software programs.").

Patent Owner groups the arguments for limitation [h] together with its unpersuasive arguments for limitation [f]. PO Resp. 42–45 (equating domains identified by Patent Owner as runtime environments).

After considering the full record including reasons to combine the references as discussed further below, Petitioner persuasively shows that the combination of the Lindskog references and Turek render obvious the subject matter of limitation [h].

h. Incorporation by Reference and Reasons to Combine the Lindskog References

Petitioner asserts "*Lindskog II* and *Lindskog III* are incorporated by reference in *Lindskog* . . . such that their disclosures are part of *Lindskog*." Pet. 5. Particularly, Petitioner asserts (i) "*Lindskog* incorporates by reference *Lindskog II* . . . which is a patent stemming from a prior-filed, commonly-assigned application that was expressly 'incorporated by reference herein in its entirety' in the *Lindskog* specification," and (ii) "*Lindskog* incorporates by reference *Lindskog III* . . . which is a patent stemming from a prior-filed, commonly-assigned application that was expressly 'incorporated by reference herein in its entirety' in the *Lindskog II* specification," and "[b]ecause *Lindskog II* is incorporated by reference in *Lindskog* and thus forms part of the original disclosure of *Lindskog*, *Lindskog III* also forms part of the original disclosure of *Lindskog*." *Id.* at 21, 23 (citing Ex. 1007, 1:5–10; Ex. 1008, 1:7–9).

Petitioner further states "[l]ike *Lindskog*, *Lindskog II* is directed to management of distributed communication networks and an agent-based system for network management," and "[l]ike *Lindskog*, *Lindskog III* is directed to management of distributed network using 'a hierarchy of performance agents 10.'" *Id.* at 21, 23 (citing Ex. 1008, 1:13–42, 2:57–3:4; Ex. 1010, 4:23–24).

Petitioner also asserts that "[a] POSITA would additionally have been motivated to combine *Lindskog*, *Lindskog II*, and *Lindskog III*." Pet. 28 (citing Ex. 1001 ¶ 105). Petitioner asserts that reasons for the combination (of the three Lindskog, Lindskog II, and Lindskog III references) and attendant beneficial results include the following: (i) the three references

having been "each filed only a few months apart by the same applicant and . . . directed to the same problem—management of complex networks using agents"; (ii) the three references "were filed only a few months apart, assigned to the same entity, and there is an express teaching to combine—namely, *Lindskog II* and *Lindskog III* are expressly cited in *Lindskog* and *Lindskog II*, respectively"; (iii) "all three [references] pertain to the same subject matter and seek to address the same issue—management of distributed networks by applying an almost identical methodology, namely, agent-based scalable solutions"; (iv) the hierarchical network management systems of the three references are combinable with "a reasonable expectation of success" as the references "share the same goals, and apply similar agent-based methodology"; and (v) the combination would "achieve a more scalable and efficient network management system, which is recognized as a common goal of all three [Lindskog] patents." *Id.* at 28–30 (citing Ex. 1007, 3:5–8, 4:48–57; Ex. 1008, 3:23–28; Ex. 1010, 1:9–12, 2:1–4; Ex. 1001 ¶ 105).

Persuasively supporting and summarizing its position regarding reasons for combining including scalability, flexibility, and improved and similar methodology in fault control, Petitioner quotes Lindskog, Lindskog II, and Lindskog III, respectively, as follows:

> *See, e.g.*, Ex. 1007, 4:48–57 ("Essentially, fault management capabilities are distributed throughout the various levels of a communications network. Maximum efficiency is then achieved by performing fault processing, and making necessary corrections, at the lowest possible level. In addition, the decentralization of the fault management operations allows scalability (i.e., the ability to continue operations as new equipment or levels are added to the network) of the system to a

> much larger degree than with a centralized fault management solution."); Ex. 1008, 3:23–28 ("Another important technical advantage of the present invention is that it provides a system that is both flexible and scalable, with the possibility of configuring the system to suit the needs of individual operators, the size of the particular networks, and/or the particular control problems to be deployed."); Ex. 1010, 1:9–12 ("Such management is very important to ensure that corrective actions are taken when necessary and so that network design can be improved according to usage and performance."), 2:1–4 ("It will be appreciated that because the agents are distributed in a hierarchy, the capacity of the management system can be scaled to meet the managed network raw data output and the activities can be distributed in a flexible manner."); *see also* Ex. 1001 ¶ 105. Because Lindskog II and Lindskog III are incorporated by reference by Lindskog, share the same goals, and apply similar agent-based methodology, a POSITA would find a high likelihood of success in combining these three references. Ex. 1001 ¶ 105.

Pet. 29–30 (quoting Ex. 1007 (Lindskog); Ex. 1008 (Lindskog II); Ex. 1010 (Lindskog III); citing Ex. 1001 (Almeroth testimony)).

Patent Owner does not dispute that one of ordinary skill in the art would have had a reason to combine Lindskog, Lindskog II, and Lindskog III or that Lindskog incorporates by reference Lindskog II and Lindskog III.

Based on a review of the record, Petitioner articulates persuasive reasons, based on rational underpinnings, to combine the teachings of Lindskog, Lindskog II, and Lindskog III. In contemplating Lindskog's system, an artisan or ordinary skill would have consulted the teachings of Lindskog II and Lindskog III, part of prior related "agent-based scalable solutions" in "hierarchical network management systems," incorporated by reference entirely and respectively into Lindskog and Lindskog II, as Petitioner argues. *See* Pet. 29–30 (citing Ex. 1001 ¶ 102). An artisan of

ordinary skill would have contemplated the inter-related teachings to understand fully the related agent-based system taught by Lindskog. For example, an artisan of ordinary skill would have consulted the three related teachings to shed more light on what Lindskog teaches, including teachings related to software and load control by monitoring data, and in turn, use the understanding to implement a more efficient system using a scalable solution to improve corrective actions. *See id.* at 29–30, 42 (citing Lindskog III, Ex. 1010, 5:28–25 (detecting hardware/software faults)), 43 (citing Lindskog II, Ex. 1008, 5:28–25 (monitoring real time performance information), citing Lindskog III, Ex. 1010, 4:14–15 (monitoring raw traffic data)). Determining obviousness involves "an expansive and flexible approach." *See KSR*, 550 U.S. at 415. Petitioner primarily points to teachings in Lindskog II and Lindskog III that apply to Lindskog's similar system or that support or buttress Lindskog's methods and systems.

Accordingly, on a review of the record, Petitioner provides persuasive reasons to combine the related teachings of Lindskog II and III with, and to consider them incorporated by reference into, Lindskog. *See* Pet. 28–33.

i. Reasons to Combine Turek with Lindskog

Supplementing the motivation discussed above, Petitioner asserts "[a] POSITA would . . . have been motivated to combine *Lindskog* (together with *Lindskog II* and *Lindskog III*) with *Turek*, which was filed only a few months later and is directed to the same problem—management of complex networks using agents." Pet. 30. Petitioner asserts that reasons for the combination and attendant beneficial results include the following: (i) "applying *Turek's* multithreading to *Lindskog's* fault management system would have been a simple matter of using a known technique to implement

the network resources in *Lindskog* as threads to achieve the predictable result of a more robust and efficient [network management system] system" with "a high expectation of success in doing so"; (ii) to "enable decentralization of the management system" using the "multithreaded architecture of *Turek*"; (iii) "to achieve a more efficient network management system" that conserves network bandwidth and achieves "lowered bandwidth consumption"; (iv) to "allow[] for simultaneous execution of multiple parts of a program, thus improving processor utilization via multitasking," by "applying *Turek's* multithreaded architecture to *Lindskog's* resources"; and (v) "to increase the concurrency and speed of *Lindskog's* corrective measures" by implementing "multiple resources running as threads . . . concurrently in parallel to achieve a corrective measure, thus saving time and more efficiently resolving detected faults." *Id.* at 30–33 (citations omitted).

Patent Owner disputes that one of ordinary skill in the art would have had a reason to combine Turek with the Lindskog references, asserting that "Lindskog and Turek are incompatible systems with opposite goals." PO Resp. 45 (citing Ex. 2001 ¶¶ 93–95). According to Patent Owner, "the Lindskog references describe a system in which fault management functions are distributed in a hierarchical manner to each node so as to minimize the distance that information relating to faults needs to travel." *Id.* at 46. Patent Owner asserts the Lindskog references "teach[ ] against centralization of fault management, describing such systems as lacking 'robustness' and as requiring information relating to faults having to travel larger distances to reach the centralized management." *Id.*

In contrast, according to Patent Owner, "Turek . . . describes a system of fault management that relies on a dispatch mechanism preferably located at a central location so as to reduce the costs of running fault management resources at each node, even when they are not needed. It teaches against distribution of fault management resources to each node in a systems as inefficient and costly." PO Resp. 46. Therefore, based on these incompatible goals, Patent Owner asserts that "[a] person of ordinary skill in the art . . . would not have been motivated to combine" the references. *Id.* (citing Ex. 2001 ¶¶ 93–104).

More particularly, Patent Owner asserts that Lindskog teaches decentralization over centralization. PO Resp. 47. Patent Owner quotes Lindskog as teaching fault management at the "lowest possible level" as follows:

> Essentially, *fault management capabilities are distributed throughout the various levels of a communications network.* Maximum efficiency is then achieved by performing fault processing, and making necessary corrections, at the lowest possible level. In addition, *the decentralization of the fault management operations allows scalability (i.e.*, the ability to continue operations as new equipment or levels are added to the network) of the system to a much larger degree than with a centralized fault management solution.

Id. at 47 (quoting Ex. 1007, 4:48–57) (emphases by Patent Owner).

Patent Owner contends that Lindskog "details with specificity what the problems with a centralized fault management system are. 'First, unacceptably large amounts of bandwidth [would be] consumed [in a centralized management system] by the alarm information that must be sent to the highest level of the fault management system.'" PO Resp. 47 (quoting

Ex. 1007, 2:65–67; citing Ex. 2003, 69:24–70:6). Patent Owner quotes Lindskog as teaching another problem with centralized systems: "[A] fully centralized operation and management (O&M) system . . . lacks robustness; if the centralized O&M system breaks down, handling of fault management tasks will be suspended." *Id.* at 48 (quoting Ex. 1007, 3:3–8).

Patent Owner contends that Turek "avoids distributing resources, including agents, throughout the nodes of its network." PO Resp. 48. Patent Owner asserts that Turek teaches overcoming problems with prior art management systems, teaching that distributing resources to each node "is quite costly" and "as the number of managed nodes increases, the system maintenance problems also increase, as do the odds of a machine failure or other fault." *Id.* (quoting Ex. 1007, 1:27–34).

Contrary to Patent Owner's arguments, Petitioner provides persuasive reasons to combine Turek and the Lindskog references in a manner that maintains the central functionality of Lindskog's system, and Turek's multi-threaded architecture does not teach against Lindskog's method or render it incompatible with Lindskog's goals. As one example of a valid reason to combine the teachings, Petitioner explains that "[t]he combination would have resulted in a more robust and efficient network management system, with increased concurrency and speed when implementing corrective measures." Reply 23 (citing Pet. 30–33; Ex. 1001 ¶¶101–114).

Petitioner explains that Turek does not teach only a centralized architecture, but instead teaches "'a large distributed enterprise' with gateways 'at a plurality of locations.'" Reply 24 (quoting Ex. 1012, 5:35–36, 5:62–64). Petitioner concedes that Turek's fault management dispatch mechanism may "either be centrally located (as [Patent Owner] points out)

or 'may also be located at a particular gateway node.'" *Id.* (quoting Ex. 1012, 8:55–58; citing Ex. 1012, 2:30–32, 5:34–36 ("a set of 'software agents' are available at a central location (e.g., manager 14) or at a plurality of locations (e.g. the gateways 16)"), 5:65–67 ("the manager 14 includes the dispatch mechanism 35 having a set of software agents 37 associated therewith.").

The cited portions of Turek in the previous paragraph support Petitioner. Although Turek "includes a dispatch mechanism preferably located at a central location" as Patent Owner contends, Turek also supports "a runtime environment supported on given nodes of the network" dispatched from a non-central location (i.e., "a particular gateway node"), similar to the system of Lindskog—as Petitioner argues. *See* Ex. 1012, 2:31–33, 5:34–36, 8:55–58 (dispatch mechanism "may also be located at a particular gateway node"); Reply 24.

Patent Owner's Sur-reply disputes that Turek teaches a distributed dispatch or fault management system. Patent Owner contends that Turek's "dispatch mechanism is also limited to one location, such as 'the system manager[]' or 'a *particular* gateway node[].'" PO Sur-reply 25 (emphasis by Patent Owner) (quoting Ex. 1012, 8:55–58). Patent Owner also contends that Turek teaches that "'software agents' are available at" a limited number of "locations (e.g., the gateways[])" from which they are "dispatched … and then migrate throughout the network environment." *Id.* (quoting Ex. 1012, 5:33–40). This second argument by Patent Owner contradicts the first argument. Turek teaches making "a set of 'software agents' available for dispatch from "a plurality of gateways," supporting Petitioner's showing of a distributed fault management system. *See* Ex. 1012, 5:32–40.

Moreover, Patent Owner's allegation that Turek's centralized fault management system would be incompatible with, or teach against, Lindskog's system, does not account for Turek's use of at least local runtime resources at each node and decentralization of some of the management functions, similar to the system of Lindskog. *See* Pet. 40. Also, Turek does not "teach[] against distribution of fault management resources to each node in a systems as inefficient and costly" as Patent Owner argues. *See* PO Resp. 46.

Rather, as noted, Turek teaches employing resources at each node—including at least a runtime engine at each node. For example, "[a]t each node [in Turek's system], the agent is preferably incorporated into or otherwise executed by the previously deployed runtime environment." *Id.* at 9:15–17. Similarly, Turek teaches "as a large portion (namely, the runtime engine) of the diagnostic capability is already at the system to be diagnosed, network traffic is further minimized," as Petitioner argues. Ex. 1010, 9:17–20; *see* Pet. 40 ("Turek teaches that '***the runtime environment*** ***(e.g., an engine)*** is preferably part of a distributed framework supported on each managed node of the distributed enterprise environment.'" (quoting Ex. 1012, 3:33–36)).

Turek's use of minimal resources (runtime engines) at each node is similar to Lindskog's system of minimizing the functionality of agents at lower nodes. *Compare* Ex. 1012, 2:12–15 ("A still further object of the invention is to dispatch, into a large distributed computer network, the minimum amount of code that may be necessary to rectify a given network fault."), 2:33–36 ("[T]he runtime environment (e.g., an engine) is preferably part of a distributed framework supported on each managed node of the

distributed enterprise environment."), 2:22–26 ("Yet another object of the present invention is to collect information about network conditions as mobile software agents are dispatched and migrated throughout a large computer network to correct network faults, wherein such information is then useful in diagnosing new faults."), *with* Ex. 1007, 5:2–17 ("The lowest level of the system 10 is assumed to be a network resource 18 such as a single circuit in a cellular base station, for example, while the highest level of the system 10 typically is the network operator. . . . The receiving FA 12 then decides whether the alarm can be handled on the current level or if it must be forwarded (as indicated at 22) to the next higher agent level. ").

Moreover, the challenged claims do not require an agent at every node (or at any node), large networks, and a vast amount of (or any) resources at each node. Turek discourages against using vast amounts of management resources at each node, including power and memory (megabytes), which typically occurred in prior art centrally controlled networks with high numbers of nodes. *See* Ex. 1012, 1:14–36; Reply 24 ("*Turek's* invention sought to overcome prior centralized management systems where a central server 'manages a number of nodes.'" (citing Ex. 1012, 1:16–17)); Ex. 1012, 1:14–16, 35–36 ("The problem is exacerbated in a typical enterprise as the node number rises."), 1:19–28 (Each managed node [in the prior art systems] typically includes a management framework, comprising a number of management routines, that is capable of a relatively large number (e.g., hundreds) of simultaneous network connections to remote machines. The framework manages hundreds of megabytes of local storage and can spawn many dozens of simultaneous processes to handle method requests from local or remote users. This amount of power, however, is quite costly.").

Neither Lindskog and Turek, nor the claims, require large amounts of management resources at each node. Accordingly, for the reasons noted above, a person of ordinary skill would not be "discouraged from following the path set out in the reference [such as using threaded runtime environments at nodes] or would be led in a direction divergent from the path that was taken" in the claim. *See Galderma Labs., L.P. v. Tolmar, Inc.*, 737 F.3d 731, 738 (Fed. Cir. 2013); *see also In re Kahn*, 441 F.3d 977, 990 (Fed. Cir. 2011) (("Nothing in Stanton can be said to discourage a person having ordinary skill in the art from using the visual-input control taught in Garwin *in the claimed combination* or to lead the skilled artisan in a direction divergent from the path taken by Kahn." (emphasis added)).

Turek also teaches that by keeping resources at a node (including threaded resources), this conserves network bandwidth (by reducing network traffic): "Thus, as a large portion (namely, the [threaded] runtime engine) of the diagnostic capability is already at the system to be diagnosed, network traffic is further minimized." *Id.* at 9:17–20. This further shows decentralization with respect to that diagnostic capability. Similarly, "[o]nce the actual fault is located, the runtime engine executes the [threaded] software agent to diagnose and/or rectify the problem, or it requests additional help for this purpose if necessary." *Id.* at 8:62–65.

This request for additional help and storage of system resources at nodes also tracks Lindskog's system, which attempts to rectify faults at minimally coded lower level agents and seeks additional help from upper level agents in the hierarchy only if a lower level agent cannot handle the fault, thereby minimizing bandwidth resources by limiting diagnostic data communicated by the agents, as discussed further below and as Petitioner

argues. *See supra* § III.D.4; Reply 25; Pet. 32; Ex. 1007, 6:9–13 ("[T]he subordinate FA 12 only forwards alarm information to agents that have requested such information through a subscription request. This function avoids the consumption of bandwidth for sending data that are not relevant for the moment."); Ex. 1012, 9:39–43. Turek similarly discloses that its system "conserves network bandwidth as software agents migrate through the network. In some cases, two or more agents, or an agent created by the dispatch mechanism from two diagnostic procedures, may be used if indicated." Ex. 1012, 9:7–10.

Moreover, Petitioner combines Turek's software teachings, including its multi-thread runtime environments and agents suggested at each node, in a manner compatible with Lindskog's system. Contrary to the thrust of Patent Owner's arguments alleging incompatibility with Lindskog's decentralized network teachings based on an alleged totally centralized fault management system in Turek, nothing about Turek's software agents, multi-threading, and localized runtime engines limits Turek's well-known Java applet multi-thread teachings to a centralized fault system or renders Turek's multi-thread teachings incompatible with Lindskog's system.

Regarding Patent Owner's argument that Turek's system reduces costs (PO Resp. 49), Petitioner explains that Turek's system provides robustness via the use of threads (*see* Reply 24). Patent Owner's argument that Turek's system "[i]nstead" reduces costs does not undermine Petitioner's showing threads increase robustness. *See* PO Resp. 49. As Petitioner argues, Turek teaches that prior art systems that manage a large number of nodes were costly. *See* Reply 24 (citing Ex. 1012, 1:16–17). Turek supports Petitioner, and teaches that prior art systems include a server

that manages "a number of nodes," and the cost in such prior art systems increases as the number of nodes increase (due to increased maintenance, heavy use, and large memory requirements), whereas Turek's system of multi-threaded runtimes and agents improves upon those prior art systems. *See* Ex. 1012, 1:12–34.

Patent Owner agrees that Lindskog's system fosters robustness and contends robustness is "a result of the *decentralized* structure of the system." PO Resp. 49. But Petitioner applies Turek's multithreaded runtime environments, which Turek dispatches from gateway nodes, into Lindskog's decentralized management structure, which employs a hierarchical structure similar to Turek's gateway structure. This application does not defeat the robustness of Lindskog, but increases it, due to the parallel threads of Turek. *See* Reply 25.

Regarding Patent Owner's related argument that Petitioner does not provide motivation for implementing Turek's threads in Lindskog's system, Petitioner responds that Lindskog II teaches the overload of network resources, and Lindskog teaches its network resources represent threads ("the execution of a section of code independently within a software program") for monitoring operational parameters such that these threads are the "multiple threads that that could be blocked/consumed by tasks." *See* Reply 25 (quoting PO Resp. 50). In other words, Petitioner shows that using a multiple-threaded runtime environment for agents at each node, as Turek suggests, helps to prevent thread blockage and overloaded threads, thereby increasing robustness.

In its Sur-reply, Patent Owner contends that Lindskog does not disclose any threads to be blocked, and queries that if Lindskog does

disclose threads, "[t]his . . . begs the question: if Lindskog *already discloses threads*, why would one look to Turek for multi-threading?" PO Sur-reply 25. As discussed above, Petitioner relies on Turek in the alternative, Turek shows that multi-threading was well-known, and Petitioner relies on Turek and the combined teachings to suggest monitoring of operational parameters including a per-thread utilization. *See supra* §§ II.E.1.c, d.

Regarding Patent Owner's argument that Turek's "'multithreading' relates to the utilization of a processor, while 'more efficient network management' [of Lindskog] relates to network bandwidth" (PO Resp. 51), Petitioner replies that "both *Turek* and *Lindskog* 'conserve[ ] network bandwidth' in a similar way, thereby encouraging the addition of multi-threading to result in a more efficient system." Reply 25. According to Petitioner, Lindskog and Turek "achieve bandwidth-light network management by limiting the amount of unnecessary diagnostic data communicated by its agents." Reply 25 (quoting Pet. 32; citing Ex. 1007, 6:9–13; Ex. 1012, 9:39–43). The record supports Petitioner. As discussed above, both systems handle faults at a local or lower node level if possible, rendering it unnecessary to communicate all manner of fault information up through all the nodes in the system, thereby saving bandwidth and mitigating processing at higher levels in Lindskog's hierarchical system. *See supra* §§ II.D.1, II.D.4.

Patent Owner's related argument that "Turek teaches that the size of the *agents* should be minimized by sending 'only the amount of agent code necessary to address that particular fault through the network'" reinforces the findings here that Turek's system minimizes bandwidth in a similar fashion as Lindskog. *See* PO Resp. 53 (quoting Ex. 1012, 9:5–8).

As Petitioner argues, Patent Owner's argument that Turek minimizes agent size to overcome a bandwidth problem in a "centralized design" so that the "Turek and Lindskog [teachings] with respect to minimizing the use of network bandwidth do not apply to each other," lacks record support, because Turek teaches decentralization of agent distribution as discussed above. *See* PO Resp. 53.

Regarding Patent Owner's arguments that no need exists to address faults "more quickly than 'in real time'" (Reply 26 (quoting Resp. 53)), Petitioner maintains that "[m]ulti-threading in the context of fault management allows multiple 'real time' faults to be addressed at the same time" (*id.*). Petitioner explains that the alternative of not using multi-threading potentially creates more bottlenecks so that eliminating bottlenecks using multi-threads increases overall speed and efficiency. *See id.*; Pet. 33 ("[M]ultiple threads can run in parallel using multi[-]threading techniques, so multiple resources running as threads could be implemented concurrently in parallel to achieve a corrective measure, thus saving time and more efficiently detecting faults").

Accordingly, a full review of the record reveals that Petitioner articulates persuasive reasons based on rational underpinnings to combine Lindskog and Turek.

j. Summary, Claim 1

Based on the foregoing discussion and a review of the full record, Petitioner shows by a preponderance of evidence that claim 1 would have been obvious over Lindskog, Lindskog II, Lindskog III, and Turek.

2. Independent Claims 7 and 13

Independent claim 7 recites a "non-transitory computer readable medium" in the preamble and independent claim 13 recites a "[a] system" in the preamble, a processor, and a memory, and otherwise includes limitations similar to the limitations of independent claim 1. *See* Pet. 60–67 (addressing claims 7 and 13). Petitioner primarily relies on the above-discussed showing with respect to claim 1. *See id*. Patent Owner does not argue the claims separately. *See* PO Resp. 55.

Based on the foregoing discussion of claim 1 and a review of the full record, Petitioner shows by a preponderance of evidence that claims 7 and 13 would have been obvious over Lindskog, Lindskog II, Lindskog III, and Turek.

3. Dependent Claims 2, 3, 8, 9, 14, and 15

Petitioner asserts that the proposed combination teaches each of the additional limitations of dependent claims 2, 3, 8, 9, 14, and 15. Pet. 58–61. These dependent claims generally recite further limitations related to load balancing and monitoring parameters using threshold comparisons. Petitioner shows these network control limitations were typical and known in the prior art and would have been obvious, supporting its showing with citations to the record and the testimony of Dr. Almeroth. *Id.* (citing Ex. 1001). As noted above, Patent Owner argues the claims together.

Based on the foregoing discussion of claims 1, 7, and 13, and a review of the full record, Petitioner shows by a preponderance of evidence that claims 2, 3, 8, 9, 14, and 15 would have been obvious over Lindskog, Lindskog II, Lindskog III, and Turek.

F. Lindskog, Lindskog II, Lindskog III, Turek, and Dijkstra, Dependent Claims 6 and 18

Petitioner contends that claims 6 and 18 would have been obvious over Lindskog, Lindskog II, Lindskog III, Turek, and Dijkstra. Pet. 67–69. These dependent claims add "wherein creating the corrective policy comprises using Dijkstra's Self-Stabilization Algorithm." Petitioner relies on the combination of Lindskog, Turek and Dijkstra to suggest Dijkstra's algorithm. *See id.* Petitioner supports these assertions with citations to the record and the testimony of Dr. Almeroth. *Id.* (citing Ex. 1001). Petitioner also points out "during prosecution of the '659 [p]atent, the examiner took Official Notice that Dijkstra's Algorithm was obvious and well-known at the time of the alleged invention." *Id.* at 69 (citing Ex. 1004, 126).

As noted above, Patent Owner does not argue the claims separately from that of claim 1 or otherwise separately challenge Petitioner's showing with respect to claims 6 and 18.

Based on the foregoing discussion of claims 1, 7, and 13, and a review of the full record, Petitioner shows by a preponderance of evidence that claims 6 and 18 would have been obvious over Lindskog, Lindskog II, Lindskog III, Turek, and Dijkstra.

III. CONCLUSION

The outcome for the challenged claims of this Final Written Decision follows.[28] In summary:

[28] Should Patent Owner wish to pursue amendment of the challenged claims in a reissue or reexamination proceeding subsequent to the issuance of this decision, we draw Patent Owner's attention to the April 2019 Notice Regarding Options for Amendments by Patent Owner Through Reissue or

Claims	35 U.S.C. §	References/ Basis	Claims Shown Unpatent-able	Claims Not shown Unpatent-able
1–3, 7–9, 13–15	103(a)	Lindskog, Lindskog II, Lindskog III, Turek	1–3, 7–9, 13–15	
6, 18	103(a)	Lindskog, Lindskog II, Lindskog III, Turek, Dijkstra	6, 18	
Overall Outcome			1–3, 6–9, 13–15, 18	

IV. ORDER

In consideration of the foregoing, it is hereby

ORDERED that claims 1–3, 6–9, 13–15, and 18 of the '659 patent are unpatentable; and

FURTHER ORDERED that because this is a Final Written Decision, parties to the proceeding seeking judicial review of the Decision must comply with the notice and service requirements of 37 C.F.R. § 90.2.

Reexamination During a Pending AIA Trial Proceeding. *See* 84 Fed. Reg. 16,654 (Apr. 22, 2019). If Patent Owner chooses to file a reissue application or a request for reexamination of the challenged patent, we remind Patent Owner of its continuing obligation to notify the Board of any such related matters in updated mandatory notices. *See* 37 C.F.R. § 42.8(a)(3), (b)(2).

For PETITIONER:

May Eaton
PERKINS COIE LLP
may.eaton@perkinscoie.com

Eugene Goryunov
Dina Blikshteyn
HAYNES & BOONE LLP
eugene.goryunov.ipr@haynesboone.com
dina.blikshetyn.ipr@haynesboone.com

For PATENT OWNER:

Vincent Rubino
Peter Lambrianakos
Enrique Iturralde
FABRICANT LLP
vrubino@fabricantllp.com
plambrianakos@fabricantllp.com
eiturralde@fabricantllp.com