Trials@uspto.gov
571-272-7822

Paper 8
Date: September 6, 2022

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

AMAZON.COM, INC. and
AMAZON WEB SERVICES, INC.,
Petitioner,

v.

SWARM TECHNOLOGY LLC,
Patent Owner.

IPR2022-00633
Patent 10,592,275 B2

Before MICHAEL R. ZECHER, GREGG I. ANDERSON, and SCOTT B. HOWARD, *Administrative Patent Judges.*

ZECHER, *Administrative Patent Judge*.

DECISION
Granting Institution of *Inter Partes* Review
35 U.S.C. § 314

I. INTRODUCTION

A. Background and Summary

Amazon.com, Inc. and Amazon Web Services, Inc. (collectively "Amazon") filed a Petition requesting an *inter partes* review ("IPR") of claims 1–17 of U.S. Patent No. 10,592,275 B2 (Ex. 1001, "the '275 patent"). Paper 2 ("Pet."). Swarm Technology LLC ("Swarm") filed a Preliminary Response. Paper 5 ("Prelim. Resp.").

We have authority, acting on the designation of the Director, to determine whether to institute an IPR under 35 U.S.C. § 314 and 37 C.F.R. § 42.4(a). An IPR may not be instituted unless "the information presented in the petition filed under section 311 and any response filed under section 313 shows that there is a reasonable likelihood that the petitioner would prevail with respect to at least 1 of the claims challenged in the petition." 35 U.S.C. § 314(a) (2018). "When instituting [an IPR], the Board will authorize the review to proceed on all of the challenged claims and on all grounds of unpatentability asserted for each claim." 37 C.F.R. § 42.108(a) (2021).

For the reasons set forth below, upon considering the preliminary record, we determine that the information presented in the Petition establishes a reasonable likelihood that Amazon will prevail with respect to at least one of the challenged claims. Accordingly, we institute an IPR on all of the challenged claims based on all of the grounds raised in the Petition.

B. Related Matters

The parties identify the following district court cases involving the '275 patent: (1) *Juniper Networks, Inc. v. Swarm Technology LLC*, No. 3:20-cv-03137 (N.D. Cal.) and (2) *Swarm Technology, LLC v Amazon.com, Inc.*, No. 2:21-cv-00438 (D. Az.). Pet. 75–76; Paper 3 (Swarm's Mandatory Notices Pursuant to 37 C.F.R. § 42.8), 1–2. The

parties also identify the following pending IPR proceedings involving the '275 patent: (1) *Juniper Networks, Inc. v. Swarm Technology LLC*, IPR2021-01317 (involving U.S. Patent No. 9,146,777 B2), (2) *Juniper Networks, Inc. v. Swarm Technology LLC*, IPR2021-01445 (involving U.S. Patent No. 9,852,004 B2 ("the '004 patent")), (3) *Juniper Networks, Inc. v. Swarm Technology LLC*, IPR2022-00141 (involving the '275 patent), and (4) *Amazon.com, Inc. and Amazon Web Services, Inc. v. Swarm Technology, LLC*, IPR2022-00283 (involving the '004 patent). Pet. 75–76; Paper 3, 1–2. In addition, the parties identify a number of patents and patent applications related to the '275 patent. Pet. 75; Paper 3, 2–3.

C. The '275 Patent

The '275 patent is titled "System and Method for Swarm Collaborative Intelligence Using Dynamically Configurable Proactive Autonomous Agents." Ex. 1001, code (54). The '275 patent generally is directed to "parallel-process computing, and collaborative intelligence, and particularly relates to a processing architecture which involves autonomous co-processors (such as robotic vehicles, Internet of Things (IoT) components, and networked devices) configured to proactively retrieve tasks from a task pool populated by a central processing unit." *Id.* at 1:16–24.

According to the '275 patent, "[c]omputer processors traditionally execute machine coded instructions serially. To run a plurality of applications concurrently, a single processor interleaves instructions from various programs and executes them serially, although from the user's perspective the applications appear to be processed in parallel." Ex. 1001, 1:49–54. The '275 patent further states that "[t]rue parallel or multi-core processing, on the other hand, is a computational approach that breaks large computational tasks into individual blocks of computations and distributes

them among two or more processors." *Id.* at 1:54–57. "A typical multiprocessor system includes a central processing unit ('CPU') and one or more co-processors. The CPU partitions the computational requirements into tasks and distributes the tasks to co-processors. Completed threads are reported to the CPU, which continues to distribute additional threads to the co-processors as needed." *Id.* at 1:63–2:1.

The '275 patent identifies a problem with using the CPU to control the distribution of tasks:

> Presently known multiprocessing approaches are disadvantageous in that a significant amount of CPU bandwidth is consumed by task distribution; waiting for tasks to be completed before distributing new tasks (often with dependencies on previous tasks); responding to interrupts from co-processors when a task is completed; and responding to other messages from co-processors. In addition, co-processors often remain idle while waiting for a new task from the CPU.

Id. at 2:1–10. The '275 patent purportedly addresses this problem using a system that reduces CPU management overhead and which "more effectively harnesses and exploits available co-processing resources." *Id.* at 2:11–15.

Figure 1 of the ’275 patent is reproduced below.

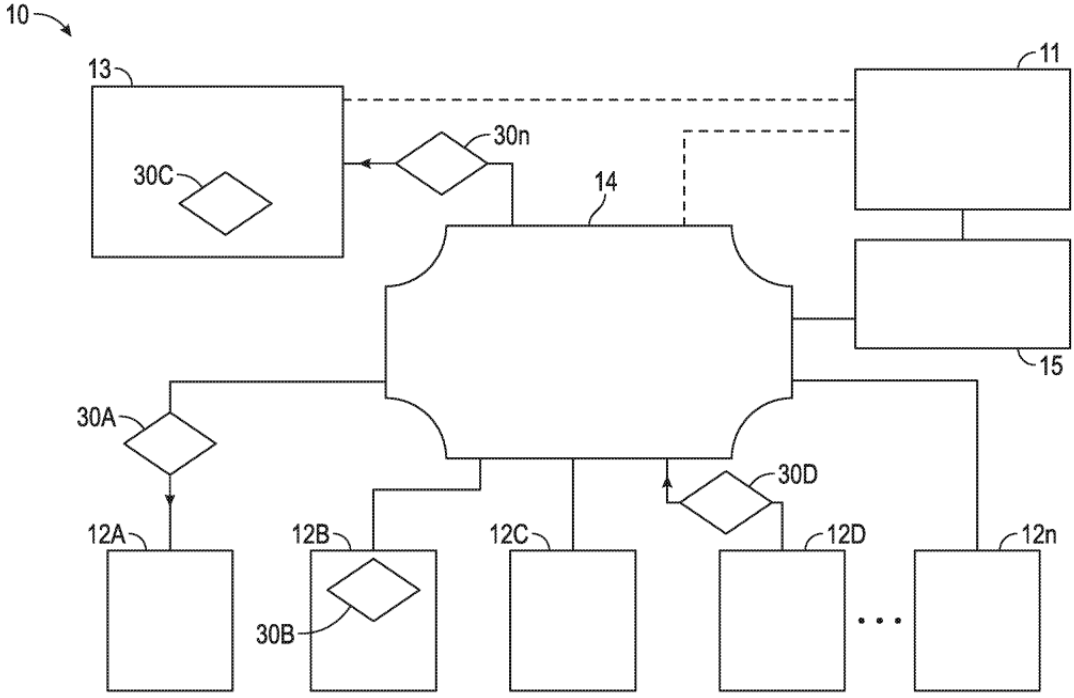

FIG. 1

Ex. 1001, Fig. 1. Figure 1, reproduced above, “is a schematic block diagram of a parallel processing architecture including a CPU, memory, task pool, and a plurality of co-processors configured to communicate through a fabric.” *Id.* at 3:64–67. More specifically, Figure 1 shows:

> a single or multi-core CPU 11 and one or more solidarity or co-processing cells 12A-12[n] configured to communicate with a task pool 13 through a cross-bar switching fabric 14. The solidarity cells 12 may also communicate with each other through the switching fabric 14 or through a separate cell bus (not shown).

Id. at 4:43–49. “CPU 11 may communicate with the task pool 13 directly or through the switching fabric 14. One or more memory units 15 each contain data and/or instructions.” *Id*. at 4:49–52.

D. Illustrative Claim

Claims 1, 6, and 11 are independent. Independent claims 1 and 6 are each directed to "[a] collaborative intelligence system," whereas independent claim 11 is directed to "[a] method of using collaborative intelligence to achieve an objective." Ex. 1001, 14:24, 15:8, 16:25–26. Claims 2–5 directly depend from independent claim 1; claims 7–10 directly or indirectly depend from independent claim 6; and claims 12–17 directly depend from independent claim 11. *Id.* at 14:50–15:7, 16:8–11, 16:17–24, 16:42–64. Independent claim 1, reproduced below, is illustrative of the challenged claims.

> 1. A collaborative intelligence system, comprising:
>
> a task pool;
>
> a controller configured to populate the task pool with a plurality of first tasks and a plurality of second tasks;
>
> a first co-processor configured to successively: proactively retrieve a first task from the task pool; process the first task; generate first resulting data; and update the task pool to reflect completion of the first task, all without any communication between the first co-processor and the controller; and
>
> a second co-processor configured to successively: proactively retrieve a second task from the task pool; process the second task; generate second resulting data; and update the task pool to reflect completion of the second task, all without any communication between the second co-processor and the controller;
>
> wherein the collaborative intelligence system is configured to dynamically accept the first co-processor, the second co-processor, and an additional co-processor into the processing system on a plug-and-play basis without any communication with the controller;

the plurality of first tasks and the plurality of second tasks are associated with a common objective;

the first and second co-processors autonomously work together in solidarity with the task pool to complete the common objective.

Id. at 14:24–49.

E. Prior Art and Asserted Grounds

Amazon relies on the prior art references set forth in the tables below.

Name[1]	Reference	Dates	Exhibit No.
Lurie	US 2007/0124363 Al	published May 31, 2007	1007
Chow	US 2005/0132380 A1	published June 16, 2005	1008
Sander	US 2012/0192201 A1	published July 26, 2012	1010

Printed Publication	Exhibit No.
The Ethernet: A Local Area Network, Data Link Layer and Physical Layer Specifications (Sept. 30, 1980) ("Ethernet Specification"). All citations are to the native pagination.	1009
Gregory R. Andrews, Foundations of Multithreaded, Parallel, and Distributed Programming (2000) ("Andrews"). All citations are to the native pagination.	1011
Ali Marjovi et al., *Robotic Clusters: Multi-Robot Systems as Computer Clusters: A Topological Map Merging Demonstration*, 60 Robotics & Autonomous Sys. 1191 (2012) ("Marjovi"). All citations are to the native pagination.	1026

[1] For clarity and ease of reference, we only list the first named inventor.

Amazon also relies on the testimony of David Lowenthal (Ex. 1003 (Declaration of Dr. Lowenthal)) and Ingrid Hsieh-Yee, Ph.D. (Ex. 1005 (Declaration of Ingrid Shieh-Yee, Ph.D.)).

F. Asserted Grounds of Unpatentability

Amazon asserts that claims 1–17 of the '275 patent would have been unpatentable based on the grounds set forth in the table below. Pet. 1–2, 14–71.

Claim(s) Challenged	35 U.S.C. §[2]	Reference(s)/Basis
1, 3, 6, 11, 12, 14, 16, 17	103(a)	Lurie, Ethernet Specification
2, 7, 10, 13	103(a)	Lurie, Ethernet Specification, Andrews
4, 9, 15	103(a)	Lurie, Ethernet Specification, Andrews,[3] Chow

[2] The Leahy-Smith America Invents Act ("AIA") included revisions to 35 U.S.C. §§ 102, 103 that became effective on March 16, 2013. Because the '275 patent claims the benefit of the priority date of U.S. Patent Application No. 13/750,696, which was filed before March 16, 2013, and neither party has argued that the provisions of the AIA apply, we apply the pre-AIA versions of the statutory bases for unpatentability. *See* Ex. 1001, code (63).

[3] Although Amazon contends that "[c]laims 4 and 15 are unpatentable under 35 U.S.C. § 103 over Lurie in view of Ethernet Specification and Chow," whereas "[c]laim 9 is unpatentable under 35 U.S.C. § 103 over Lurie in view of Ethernet Specification, *Andrews*, and Chow," Amazon relies on Lurie, Ethernet Specification, Andrews and Chow in its substantive analysis of dependent claims 4, 9, and 15. *See* Pet. 2, 58–63. Accordingly, we consider Amazon's reliance on Andrews for each of these dependent claims.

Claim(s) Challenged	35 U.S.C. §[2]	Reference(s)/Basis
5	103(a)	Lurie, Ethernet Specification, Sander
8	103(a)	Lurie, Ethernet Specification, Andrews, Marjovi

II. ANALYSIS

A. Legal Standards

In *Graham v. John Deere Co. of Kansas City*, 383 U.S. 1 (1966), the Supreme Court set out a framework for assessing obviousness under 35 U.S.C. § 103(a) that requires consideration of four factors: (1) the "level of ordinary skill in the pertinent art," (2) the "scope and content of the prior art," (3) the "differences between the prior art and the claims at issue," and (4) when in evidence, "secondary considerations" of non-obviousness such as "commercial success, long[-]felt but unsolved needs, failure of others, etc." *Id.* at 17–18. "While the sequence of these questions might be reordered in any particular case," *KSR Int'l Co. v. Teleflex Inc.*, 550 U.S. 398, 407 (2007), the U.S. Court of Appeals for the Federal Circuit has repeatedly emphasized that "it is error to reach a conclusion of obviousness until all those factors are considered," *WBIP, LLC v. Kohler Co.*, 829 F.3d 1317, 1328 (Fed. Cir. 2016).[4]

B. Level of Ordinary Skill in the Art

In determining whether an invention would have been obvious at the time it was made, we consider the level of ordinary skill in the pertinent art at the time of the invention. *Graham*, 383 U.S. at 17. "The importance of

[4] At this preliminary stage of the proceeding, because neither party addresses objective evidence of non-obviousness, we focus solely on the first three *Graham* factors.

resolving the level of ordinary skill in the art lies in the necessity of maintaining objectivity in the obviousness inquiry." *Ryko Mfg. Co. v. Nu-Star, Inc.*, 950 F.2d 714, 718 (Fed. Cir. 1991). The "person having ordinary skill in the art" is a hypothetical construct, from whose vantage point obviousness is assessed. *In re Rouffet*, 149 F.3d 1350, 1357 (Fed. Cir. 1998).

Factors pertinent to a determination of the level of ordinary skill in the art include "(1) the 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 active workers in the field." *Env't Designs, Ltd. v. Union Oil Co. of Cal.*, 713 F.2d 693, 696 (Fed. Cir. 1983) (citing *Orthopedic Equip. Co. v. All Orthopedic Appliances, Inc.*, 707 F.2d 1376, 1381–82 (Fed. Cir. 1983)). "Not all such factors may be present in every case, and one or more of these or other factors may predominate in a particular case." *Id.* at 696–97.

In its Petition, Amazon argues that a person having ordinary skill in the art "would have had a bachelor's degree in computer science, computer engineering, or a related field, and about five years of industry or academic research experience in parallel or distributed computing, system architecture, or an equivalent." Pet. 4–5 (citing Ex. 1003 ¶¶ 42–44). Amazon further argues that "[a] post-graduate degree or additional formal education can compensate for less experience, and vice-versa." *Id.* In its Preliminary Response, Swarm does not address the level of skill in the art. *See* Prelim. Resp.

Accordingly, for purposes of institution, we adopt Amazon's proposed formulation of the level of ordinary skill in the art because it appears to be consistent with the '275 patent and the asserted prior art.[5]

C. Claim Construction

We apply the same claim construction standard used in the federal courts, in other words, the claim construction standard that would be used to construe the claim in a civil action under 35 U.S.C. § 282(b), which is articulated in *Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005) (en banc). *See* 37 C.F.R. § 42.100(b). Under the *Phillips* standard, "the words of a claim 'are generally given their ordinary and customary meaning,'" which is "the meaning that the term would have to a person of ordinary skill in the art in question at the time of the invention, i.e., as of the effective filing date of the patent application." *Phillips*, 415 F.3d at 1312–13.

In its Petition, Amazon states that "[t]he prior art relied on in this Petition discloses the subject matter of the challenged claims under any reasonable construction, including their plain meaning." Pet. 5 (citing Ex. 1003 ¶ 45). In its Preliminary Response, Swarm contends "that all claim terms of the '275 Patent should be given their ordinary and customary meaning as understood by one of ordinary skill in the art based on their plain language or, alternatively, based on their plain language in light of specification and prosecution history." Prelim. Resp. 7. Further, Swarm specifically argues that the parties "dispute the 'ordinary and customary' meaning of at least the following claim terms: i) a plurality of first tasks,

[5] If Swarm proposes a different level of ordinary skill in the art in its Response, the parties are encouraged to address whether there are any material differences between the two proposals and what impact, if any, the different level has on the obviousness analysis.

ii) a plurality of second tasks, iii) a respective task type, and iv) proactively retrieve." *Id.* at 8. Nonetheless, Swarm does not propose any express constructions for these terms.

Based on the preliminary record, because there is no readily discernible controversy between the parties that requires us to construe any claim terms for purposes of institution, we need not construe any claim terms at this time. *See Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co.*, 868 F.3d 1013, 1017 (Fed. Cir. 2017) (noting that "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))). We encourage the parties to develop any arguments related to claim construction during trial, including what impact, if any, a different claim construction would have on the obviousness analysis.

D. Asserted Obviousness in View of Lurie and Ethernet Specification

Amazon argues that claims 1, 3, 6, 11, 12, 14, 16, and 17 would have been obvious over Lurie and Ethernet Specification. *See* Pet. 14–55. Based on the preliminary record, we are sufficiently persuaded that Amazon has established a reasonable likelihood of prevailing on this asserted obviousness ground with respect to at least one of claims 1, 3, 6, 11, 12, 16, and 17.

1. Overview of Lurie

Lurie is titled "Instrument-Based Distributed Computing Systems." Ex. 1007, code (54). Lurie is directed to "providing an instrument-based distributed computing system that accelerates the measurement, analysis,

verification and validation of data in the distributed computing environment." *Id.* ¶ 2.

Lurie describes an "automatic [task] distribution mode" of a distributed computing environment. Ex. 1007 ¶ 19. Lurie includes technical computing client 250 ("TCC"), which "can be a technical computing software application that provides a technical computing and graphical modeling environment for generating block diagram models and to define mathematical algorithms for simulating models." *Id.* ¶ 55. Lurie further includes a number of technical computing workers 270A–N ("TCWs"), which "can be a technical computing software application that provides a technical computing environment for performing technical computing of tasks, such as those tasks defined or created by the technical computing client 250." *Id.* ¶ 58.

In one embodiment, Lurie further describes Automatic Task Distribution Member 260 ("ATD Mechanism"), which can be "one or more application software components to provide for the automatic [task] distribution of tasks from the [TCC] to the [TCWs]." Ex. 1007 ¶ 61. This embodiment is shown in Figure 3B, which is reproduced below.

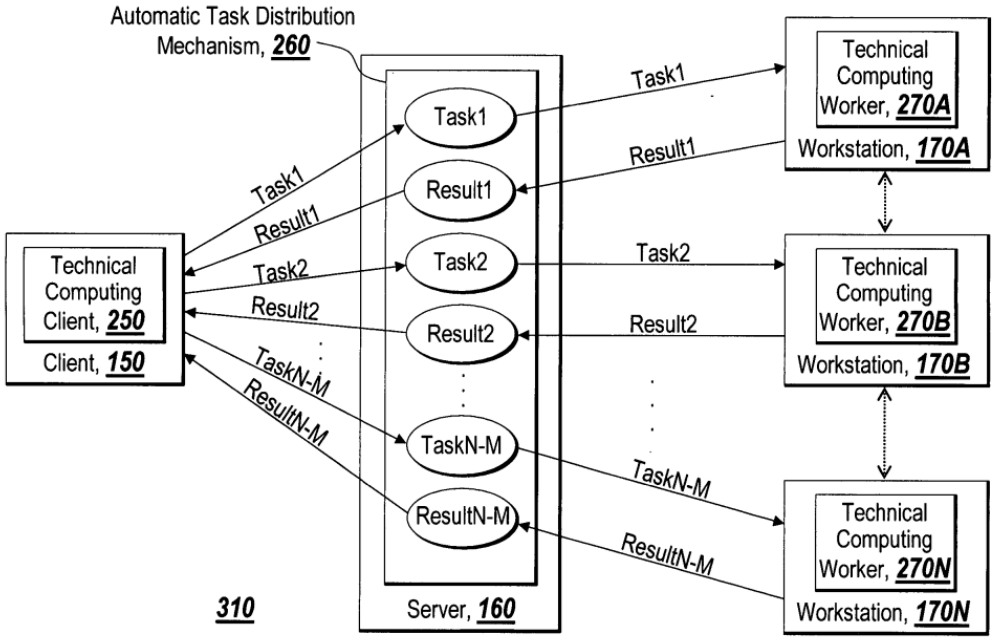

Fig. 3B

Figure 3B, reproduced above, "is a block diagram of [Lurie's] automatic [task] distribution mode" embodiment, which depicts (1) tasks and results flowing between the TCC and the ATD Mechanism and (2) tasks and results flowing between the ATD Mechanism and the TCWs. Ex. 1007 ¶ 19, Fig. 3B. More specifically, the ATD Mechanism of the system "is intended to provide a configuration where the user does not want to control which [TCW] executes a particular task." *Id.* ¶ 71. Instead, the TCC is in communication with the ATD Mechanism, which, in turn, is in communication with the various TCWs. *Id.* "Under this mode of operation, the [TCC] is not required to have any specific knowledge of the [TCWs], e.g., the name of the workstation running a [TCW], or the availability of the [TCW] to perform technical computing of a task." *Id.* The TCC submits task to the ATD Mechanism, which, in turn, makes them available to the TCWs. *Id.* ¶ 72. "A [TCW] takes a task from the [ATD Mechanism] for

technical computing of the task, computes a result for the task and provides the result to the [ATD Mechanism]." *Id.* The TCC is then able to obtain the results from the ATD Mechanism. *Id.*

An alternative "batch mode" embodiment is illustrated in Figure 3C, which is reproduced below. Ex. 1007 ¶ 20.

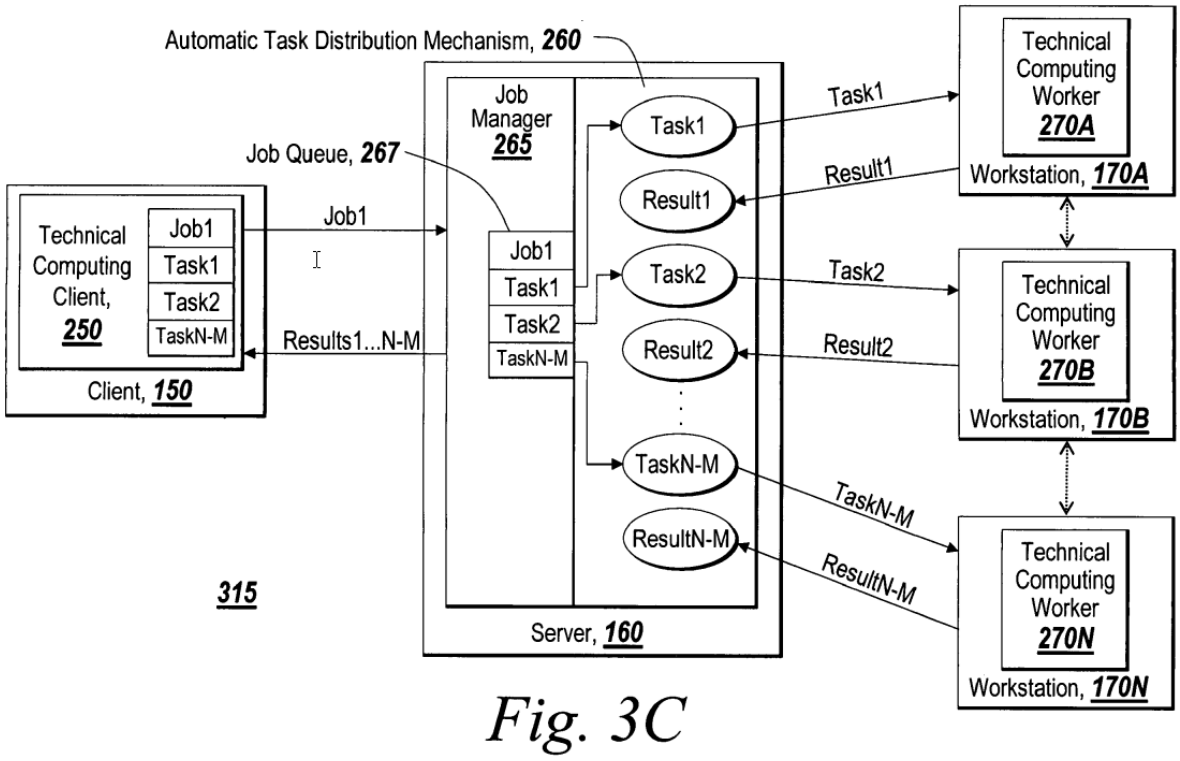

Fig. 3C

Figure 3C, reproduced above, "is a block diagram of the batch automatic distribution mode of operation." *Id.* The "batch mode . . . is intended to provide a configuration where the user can specify a group of related tasks as a job and provide the batch of tasks, or the job, to the [ATD Mechanism]." *Id.* ¶ 73. In this embodiment, the TCC "is in communication with the job manager 265 on the server 160" and "[t]he job manager 265 interfaces and communicates with the [ATD Mechanism] running on the same server 160." *Id.* As with the Figure 3B embodiment, "[e]ach of the [TCWs] 270A-270N is in communication with the [ATD Mechanism]." *Id.*

In batch mode, "[i]nstead of submitting each task separately as depicted in FIG. 3B, the [TCC] submits the job containing all the associated tasks to the job manager 265." Ex. 1007 ¶ 74. "The job manager 265 obtains the tasks from the job and provides the tasks to the [ATD Mechanism] for [TCWs] 270A-270N to take and compute results." *Id.* Lurie describes the operation of the components as follows:

> For example, [TCC] defines a job, Job1, with a set of three tasks: Task1, Task2 and TaskN-M. The [TCC] then submits Job1 to the job manager 265. The job manager 265 obtains Job1 and obtains each of the tasks, Task1-TaskN-M from Job 1. Then, . . . the job manager 265 submits each of the tasks to the [ATD Mechanism] for technical computing by a [TCW]. [TCW] 270A may take Task1 from the [ATD Mechanism], compute a Result1 for Task1 and provide the Result1 to the [ATD Mechanism]. [TCW] 270B and [TCW] 270N, in a similar fashion, compute and provide results for Task2 and TaskN-M respectively. The job manager 265 then obtains the set of results for the completed job of Job1 and provides the results of each of the tasks to the [TCC].

Id. When there are multiple jobs, "the job manager 265 may handle jobs in a first-in first-out (FIFO) manner. In this case, the job manager 265 does not process the next job until all the tasks from the current job have been processed by the [ATD Mechanism]." *Id.* ¶ 75.

2. Overview of Ethernet Specification

Ethernet Specification "contains the specification of the Ethernet, a local area network." Ex. 1009, i. Ethernet Specification "is intended as a design reference document, rather than an introduction or tutorial." *Id.*

Relevant to Amazon's argument, chapter 6 provides the "Ethernet Data Link Layer Specification." Ex. 1009, 19–44. Figure 6-1 is reproduced below.

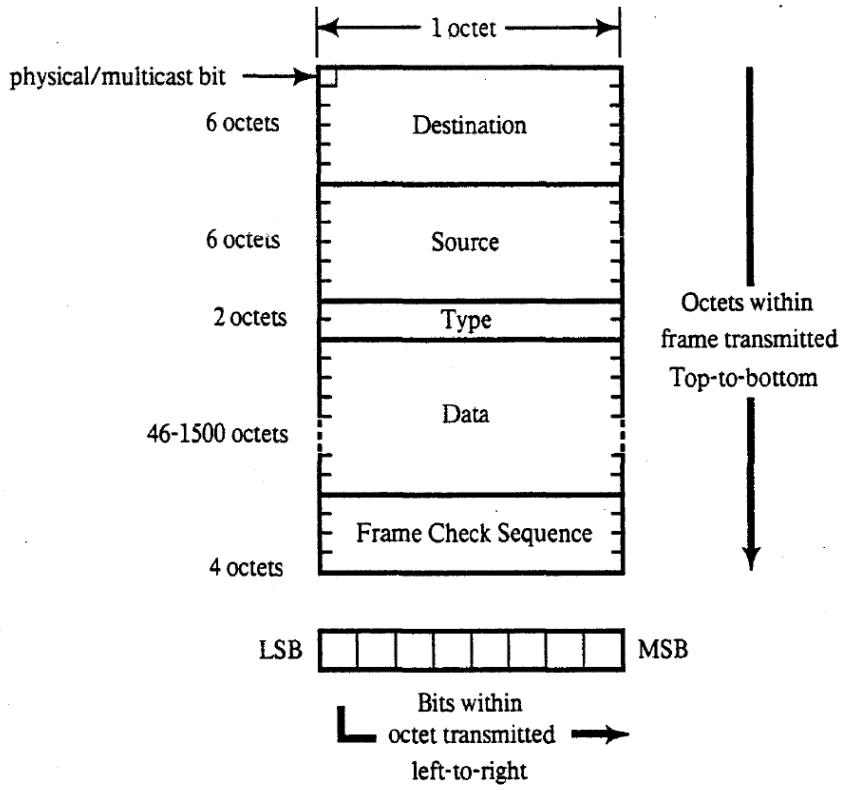

Figure 6-1: Data Link Layer Frame Format

Ex. 1009, 20. Figure 6-1, reproduced above, depicts a "Data Link Layer Frame Format," which includes a destination, source, type, data, and frame check sequence. *Id.*

3. Analysis of Independent Claim 1

a) Undisputed Limitations

The preamble of independent claim 1 recites "[a] collaborative intelligence system." Ex. 1001, 14:24. To the extent the preamble of independent claim 1 is treated as limiting, Amazon argues that Lurie teaches the feature recited in the preamble. Pet. 15–16. Specifically, Amazon argues that "Lurie's 'computing system' discloses a 'collaborative intelligence system' . . . as its TCWs execute individual 'tasks that are processed and/or managed collectively,' in order to complete a larger 'job,'

which is 'a group of one or more tasks.'" *Id.* at 15 (quoting Ex. 1007, code (57), ¶ 43) (citing Ex. 1003 ¶ 133).

Independent claim 1 further recites "a task pool." Ex. 1001, 14:25. Amazon argues that Lurie teaches this limitation. Pet. 16–18. Specifically, Amazon argues that Lurie's ATD Mechanism is "a task pool because it is a data or memory structure configured to hold, store, monitor, populate with, and update tasks, and make them available to TCWs." *Id.* at 16 (citing Ex. 1003 ¶¶ 137–38). Amazon also argues that the ATD Mechanism "can be implemented using a technology known as Javaspace, creating a 'space' or 'object exchange repository' in which tasks reside, i.e., a task pool." *Id.* at 17 (quoting Ex. 1007 ¶¶ 90–91) (citing Ex. 1007 ¶¶ 26, Fig. 6A; Ex. 1003 ¶¶ 140–44).

Independent claim 1 further recites "wherein the collaborative intelligence system is configured to dynamically accept the first co-processor, the second co-processor, and an additional co-processor into the processing system on a plug-and-play basis without any communication with the controller." Ex. 1001, 14:40–44. Amazon argues that Lurie teaches this limitation. Pet. 31–32. Specifically, Amazon contends that Lurie has multiple TCWs and that Lurie "discloses dynamic acceptance of its TCWs (first, second, and additional co-processors) on a plug-and-play basis without any communication with the TCC (controller)." *Id.* at 31 (citing Ex. 1003 ¶ 200). According to Amazon, "the TCC need not have any 'specific knowledge' of a TCW, including its 'name' or 'availability,' for a coprocessor to participate in the system." *Id.* (quoting Ex. 1007 ¶ 71) (citing Ex. 1003 ¶ 202). Amazon also argues that "Lurie's TCWs are started and stopped using an activation daemon," and that "[r]egistration with the activation daemon, and subsequent activation of the TCW, permits the TCW

to be dynamically accepted at any time, and constitutes acceptance on a plug-and-play basis." *Id.* at 32 (citing Ex. 1007 ¶ 108; Ex. 1003 ¶¶ 203–05).

After reviewing Amazon's arguments and evidence, including the Declaration of Dr. Lowenthal (Ex. 1003), which Swarm does not address at this stage (*see* Prelim. Resp.), we are persuaded that Amazon sufficiently demonstrates, for purposes of institution, that Lurie teaches each of the undisputed limitations of independent claim 1 discussed in this subsection.

b) "a controller configured to populate the task pool with a plurality of first tasks and a plurality of second tasks"

Independent claim 1 further recites "a controller configured to populate the task pool with a plurality of first tasks and a plurality of second tasks." Ex. 1001, 14:26–27.

(1) Amazon's Arguments

Amazon argues that Lurie teaches this limitation. Pet. 19–22. Specifically, Amazon argues that Lurie's TCC is a controller that is "configured to populate the task pool (ATD Mechanism) with a plurality of first tasks and a plurality of second tasks." *Id.* at 19 (citing Ex. 1003 ¶¶ 150–51). According to Amazon, Lurie teaches that the TCC "'submits one or more tasks' to the ATD Mechanism, which 'obtains the tasks . . . to

make the[m] available for distribution to any of the [TCWs].'" *Id.* (quoting Ex. 1007 ¶ 72) (citing Ex. 1007, Fig. 3C; Ex. 1003 ¶¶ 149, 152).[6]

Amazon sets forth three theories regarding the plurality of first tasks and plurality of second tasks. First, Amazon argues that Lurie's Task1 through TaskN-M can be subdivided in many different ways to create a plurality of first tasks and a plurality of second tasks. Pet. 20 (citing Ex. 1003 ¶ 155). Further, Amazon contends that Lurie's "TCWs are configured to perform certain tasks based on the 'operational and/or performance characteristics of the [TCWs],' and 'the system can determine or select' a TCW that is appropriate 'to work on a task' based on matching those characteristics with an appropriate task." *Id.* at 20 (quoting Ex. 1007 ¶ 81) (citing Ex. 1007 ¶¶ 43, 91; Ex. 1003 ¶ 156). Accordingly, Amazon argues that "[a] [person having ordinary skill in the art] would have understood that because TCWs have different characteristics, Lurie's tasks have different corresponding types and would be grouped accordingly, into distinct pluralities of tasks." *Id.*

Second, Amazon focuses on the batch mode illustrated in Lurie's Figure 3C. *See* Pet. 20–22. Amazon argues that, in batch mode, the TCC populates the ATD Mechanism with a plurality of jobs and that each job has a plurality of tasks:

---

[6] Amazon cites paragraph 72 of Lurie to teach the "controller" limitation in the context of Lurie's "automatic task distribution mode," but Amazon also cites Figure 3C, which depicts Lurie's "batch mode." To the extent that Swarm argues that Amazon conflates Lurie's automatic task distribution mode with its batch mode, we determine that Amazon has shown sufficiently that Lurie teaches the "controller" limitation with respect to Lurie's automatic task distribution mode.

> In "batch mode," a "group of related tasks" is collectively referred to as a "job," such that ["]a plurality of first tasks["] associated with one job (e.g., Task1 through TaskN-M associated with Job1 in Figure 3C), and "a plurality of second tasks" associated with a second job . . . , are populated in the task pool by the TCC. . . . In this mode, the TCC submits a job "containing all the associated tasks to the job manager," and the job manager "obtains the tasks from the job and provides the tasks to the [ATD Mechanism] for [TCWs] to take and compute results."

Id. at 20–21 (quoting Ex. 1007 ¶¶ 73–75) (citing Ex. 1007, Fig. 3C; Ex. 1003 ¶¶ 158–61). Amazon also argues that the "job manager and ATD Mechanism manage multiple jobs, including a second job consisting of a plurality of second tasks." *Id.* at 21 (citing Ex. 1007 ¶¶ 73–75).

Third, Amazon contends that "Lurie's tasks reside in a 'space' in the ATD Mechanism," and that "[a] [person having ordinary skill in the art] would have understood that tasks residing in the same space may be considered to be the same type." Pet. 22 (citing Ex. 1007 ¶ 91, Fig. 6A; Ex. 1003 ¶¶ 248–50; Ex. 1022, 394).

(2) Swarm's Arguments

Swarm disputes each of Amazon's three theories. Prelim. Resp. 27–43. With regard to Amazon's first theory, Swarm argues that "[Amazon] impermissibly modifies Lurie by arbitrarily partitioning Lurie's tasks into 'first tasks' (Task1 and Task2) and 'second tasks' (Task3 and Task4 or TaskN-M-1 and Task-N-M . . .)." *Id.* at 30. Specifically, Swarm contends that Amazon conflates Lurie's automatic task distribution mode with its batch mode, and further fails to show that Lurie "discloses any embodiment where a TCC divides Tasks1–TaskN-M into groups of tasks." *Id.* In addition, Swarm argues that the testimony of Dr. Lowenthal is not supported by the teachings of Lurie and, instead, "improperly substitute[s] Lowenthal's

own hindsight reconstruction for Lurie's lack of disclosure." *Id.* at 31 (citing Ex. 1003 ¶¶ 154–55). According to Swarm,

> Dr. Lowenthal's declaration violates [United States Court of Appeals for the] Federal Circuit case law and the [Patent Trial and Appeal Board]'s published position that "in an obviousness analysis, conclusory assertions from a third party about general knowledge in the art cannot, without supporting evidence of record, supply a limitation that is not evidently and indisputably within the common knowledge of those skilled in the art."

Id. (quoting Consolidated Trial Practice Guide 36 (Nov. 2019))[7] (citing *K/S Himpp v. Hear-Wear Techs., LLC*, 751 F.3d 1362, 1365 (Fed. Cir. 2014)).

Swarm further argues that Amazon mischaracterizes Lurie by arguing that "its tasks have [an] indicia of task type (i.e., indicator of the command to be executed, or indicator of the 'operational and/or performance characteristics' that a particular TCW must have to complete the task)." Prelim. Resp. 38 (citing Pet. 22). Swarm contends that "Lurie's tasks contain: (i) the executable computing command, (ii) the quantity of input arguments for the command, and (iii) the data that constitutes the input arguments themselves," but "no suggestion of also including an 'indicia of task type.'" *Id.* at 38–39 (citing Ex. 1007 ¶¶ 47, 83). Swarm further argues that an indicia of task type would not be useful because "Lurie's TCWs do not search the ATD Mechanism for a task that has a task type that they are capable of performing." *Id.* at 39. Rather, Swarm contends that Lurie's ATD Mechanism "determine[s] the [TCW] to work on a task *by any configuration or properties set on the [TCW]*," not the task itself. *Id.* at 39 (citing Ex. 1007 ¶ 81 (emphasis added)).

[7] *Available at* https://www.uspto.gov/TrialPracticeGuideConsolidated.

Even if the TCWs have different characteristics, Swarm argues that Lurie's tasks do not have different corresponding types. Prelim. Resp. 36 (citing Pet. 20). Rather, Swarm contends that "Lurie teaches away from the task types described in the '275 Patent, where the co-processor searches 'the task pool and retrieve[s] an available task 22 that requires completion, is not locked, and *that has a task type that can be performed by the [co-processor]*." *Id.* at 37 (quoting Ex. 1001, 8:41–43 (emphasis added)). Specifically, Swarm argues that "Lurie's ATD Mechanism assigns the tasks to the TCWs," "exclud[ing] both the TCC and the TCWs from determining which task will be performed by a particular TCW," so there "is no motivation to modify Lurie to provide a 'controller that populates a task pool with a plurality of first tasks and a plurality of second tasks.'" *Id.* at 37–38 (quoting Ex. 1001, 14:26–27).

With regard to Amazon's second theory, Swarm argues that, contrary to Amazon's argument, Lurie does not teach that the TCC—which Amazon maps to the claimed "controller"—populates the ATD Mechanism with tasks. Prelim. Resp. 32–36. According to Swarm, in batch mode, the TCC is not in communication with the ATD Mechanism; instead, "Lurie teaches that the TCC is in communication with a separate and distinct module called the 'job manager,'" which "communicates with the TCC and subsequently populates tasks to the ATD mechanism." *Id.* at 33 (quoting Ex. 1007 ¶ 73). Swarm further argues that, even in batch mode, Lurie does not teach "a task pool 'populated with a plurality of first tasks and a plurality of second tasks.'" *Id.* at 35–36 (quoting Ex. 1001, 14:26–27). According to Swarm, Lurie teaches that "the Job Manager acts as a buffer/queue that stages jobs, unpacks tasks, and only submits a single batch of tasks to the ATD at a

time," so the ATD Mechanism is only populated with the tasks associated with a single job at a time. *Id.* at 34–36 (citing Ex. 1007 ¶¶ 74–75, 95).

Finally, regarding Amazon's third theory, Swarm contends that "Lurie does not teach or suggest the use of JavaSpace to configure different objects spaces, where each object space houses groups of task objects having the same task type." Prelim. Resp. 40. Specifically, Swarm argues that, "[w]hen using a JavaSpace environment, Lurie still restricts its operation to one of the automatic task distribution mode or the batch mode," so "[n]othing about the JavaSpace environment would alter Lurie's fundamental modes of operation." *Id.* at 41.

(3) Our Analysis

Based on the preliminary record, Amazon has sufficiently shown that Lurie teaches that the TCC (controller) populates the ATD Mechanism (task pool) with a plurality of first tasks and a plurality of second tasks. As shown in Lurie Figure 3B, the TCC populates the ATD Mechanism with various tasks Task1 through TaskN-M. Ex. 1007 ¶¶ 71, 72, Fig. 3B. Amazon has sufficiently shown that a person having ordinary skill in the art would have understood that the tasks could be divided in various ways, including "a plurality of first tasks (*e.g.*, Task1 and Task2) and a plurality of second tasks (*e.g.*, Task3 and Task4, or TaskN and TaskM)." Ex. 1003 ¶ 155. Although we agree with Swarm that this division is arbitrary, Swarm has not proposed a construction of first tasks or second tasks that would preclude such an arbitrary grouping.

To the extent that Amazon's discussion of this element explains Lurie's automatic task distribution mode while citing Figure 3C—representing Lurie's batch mode—we do not find that the discussion amounts to a "misleading characterization of Lurie" or destroys the

arguments set forth by Amazon. Prelim. Resp. 27. Rather, we treat the two modes as independent theories—as otherwise indicated by the parties—and determine that Amazon has shown sufficiently that Lurie teaches the "controller" limitation with respect to Lurie's automatic task distribution mode.

In addition, we do not agree with Swarm's argument that Amazon is improperly relying on general knowledge in the art or that *K/S Himpp* applies. *See* Prelim. Resp. 31. To the contrary, Dr. Lowenthal is simply providing testimony regarding Lurie's teachings from the perspective of a person of ordinary skill in the art. *See* Ex. 1003 ¶ 155. This is not an improper use of expert testimony. *See Sundance, Inc. v. DeMonte Fabricating Ltd.*, 550 F.3d 1356, 1361 n.3 (Fed. Cir. 2008) ("What a prior art reference discloses or teaches is determined from the perspective of one of ordinary skill in the art."); Consolidated Trial Practice Guide (Nov. 2019) at 35 ("Experts may testify about the teachings of the prior art and how they relate to the patentability of the challenged claims."); *see also id.* at 36 ("Expert testimony may be presented to establish the scope and content of the prior art for determining obviousness and anticipation.").

Although we agree with Swarm that Lurie does not define expressly tasks by an indicia of task type, Swarm has not shown that Lurie precludes tasks of different corresponding types. According to Lurie, "[a] task defines a technical computing command, such as a MATLAB® command, to be executed, and the number of arguments and any input data to the arguments." Ex. 1007 ¶ 47. In addition, "[a] job is a logical unit of activities, or tasks that are processed and/or managed collectively." *Id.* Because Lurie's tasks may contain different technical computing commands, arguments, and input data, we find that Lurie teaches tasks of different types.

Similarly, we do not agree that the ATD Mechanism's determination of the TCW to work on a task based on the TCW's characteristics teaches away from modifying Lurie to use task types. Lurie teaches that "the system can determine or select the [TCW] to work on a task by operational and/or performance characteristics of the [TCW]," similar to the '275 patent's teaching that "[t]he task type indicates which cells 12 are capable of performing the task 22." Ex. 1007 ¶ 81; Ex. 1001, 7:31–32. Although the '275 patent uses the *task type* to inform the co-processors of which tasks they are capable of performing, we find that Lurie's ATD Mechanism would be unable to identify appropriate tasks for the TCWs based on each TCW's characteristics unless the tasks were also distinguishable. Therefore, we agree with Amazon that "[a] [person having ordinary skill in the art] would have understood that because TCWs have different characteristics, Lurie's tasks have different corresponding types and would be grouped accordingly, into distinct pluralities of tasks." Pet. 20 (citing Ex. 1007 ¶¶ 43, 81, 91; Ex. 1003 ¶ 156).

Based on the preliminary record, Amazon has presented sufficient evidence that would support a finding that Lurie teaches "a controller configured to populate the task pool with a plurality of first tasks and a plurality of second tasks," as recited in independent claim 1. Having found that Amazon presents sufficient evidence regarding its first theory, we need not address its additional theories. However, we do so in order to provide guidance to the parties for the trial.

We do not agree with Swarm's argument that, under Lurie's batch mode, the ATD Mechanism only processes one plurality of tasks at a time. Although Swarm accurately describes how Lurie operates in batch mode (*see* Ex. 1007 ¶ 75), it is inapposite that the job manager does not process

the second job and its second plurality of tasks until after the first job and its first plurality of tasks have been completed. There is nothing in independent claim 1 that requires both the plurality of first tasks and the plurality of second tasks to be in the task pool at the same time; instead, independent claim 1 does not have any temporal component and simply recites "a controller configured to populate the task pool with a plurality of first tasks and a plurality of second tasks." Ex. 1001, 14:26–27.

Nonetheless, based on the preliminary record, we question whether Amazon has shown that Lurie teaches the "controller" limitation under its second theory. Although Amazon identifies the TCC as the controller, based on Lurie's teachings it appears that the job manager in Lurie's batch mode populates the ATD Mechanism with tasks, not the TCC. *See* Ex. 1007 ¶ 74 ("Instead of submitting each task separately as depicted in FIG. 3B, the [TCC] submits the job containing all the associated tasks to the job manager 265," which "obtains the tasks from the job and provides the tasks to the [ATD Mechanism] for [the TCWs] to take and compute results."); *id.* Fig. 3C (showing job manager 265 populating the ATD Mechanism). Because the TCC does not populate the ATD Mechanism with tasks, it is unclear how the TCC satisfies this disputed limitation.

Finally, regarding Amazon's third theory, we question whether Amazon has shown that Lurie's JavaSpace teaches "that tasks residing in the same space may be considered to be the same type." Pet. 22 (citing Ex. 1007 ¶ 91, Fig. 6A; Ex. 1003 ¶¶ 248–50; Ex. 1022, 394). Based on the preliminary record, we agree with Swarm that Lurie's JavaSpace environment still requires the use of the automatic task distribution mode or batch mode. Prelim. Resp. 40–41. Nonetheless, we find that Amazon has presented sufficient evidence that would support a finding that Lurie teaches

the limitation based on the automatic task distribution mode, regardless of whether the operation is performed within the JavaSpace environment.

c) "a first co-processor"

Independent claim 1 further recites "a first co-processor configured to successively: proactively retrieve a first task from the task pool; process the first task; generate first resulting data; and update the task pool to reflect completion of the first task, all without any communication between the first co-processor and the controller." Ex. 1001, 14:28–33.

(1) Amazon's Arguments

Amazon argues that Lurie alone or in combination with the Ethernet Specification teaches this limitation. Pet. 23–29. Specifically, Amazon argues that one of Lurie's TCWs is a first co-processor which "'takes,' without being instructed by the TCC, 'a task from the [ATD Mechanism]' (i.e., task pool), to execute." *Id.* at 23 (quoting Ex. 1007 ¶ 72) (citing Ex. 1003 ¶ 165). Amazon argues that the "TCW (co-processor) then processes its task and generates the resulting data." *Id.*; Ex. 1007 ¶ 72. Amazon further argues that the "TCW then 'provides the result [back] to the [ATD Mechanism],' thereby updating the ATD Mechanism (task pool) to reflect completion." *Id.* (quoting Ex. 1007 ¶ 72) (citing Ex. 1003 ¶¶ 168–73). According to Amazon, because the ATD Mechanism "makes the results . . . available to the [TCC]," "the TCC (controller) and the TCWs (co-processors) do not communicate (as required by the claim), but instead rely on the ATD Mechanism as an intermediary." *Id.* at 24 (quoting Ex. 1007 ¶ 61) (citing Ex. 1007 ¶¶ 61, 72, Figs. 3C, 6A; Ex. 1003 ¶¶ 167–74). In discussing the ATD Mechanism, Amazon directs us to Lurie's Figure 6A, which shows Object Exchange Repository 662. *Id.* at 24.

Further, Amazon argues that Lurie in combination with Ethernet Specification teaches that the "TCWs 'proactively retrieve' tasks because Lurie is implemented over a network where each TCW has an agent that identifies available tasks appropriate to that TCW from the task pool . . . and retrieves an available, appropriate task to deliver to the TCW." Pet. 27 (citing Ex. 1007 ¶¶ 43, 81, 91; Ex. 1003 ¶ 183). Specifically, Amazon argues that "[a] [person having ordinary skill in the art] would have understood that[,] when Lurie is implemented over Ethernet (or similar networking protocol), data frames act as agents of a TCW to retrieve tasks as described in the '275 patent." *Id.* at 28 (footnote omitted) (citing Ex. 1007 ¶ 93; Ex. 1003 ¶¶ 128, 184, 188–90).

(2) Swarm's Arguments

Swarm presents three arguments. First, Swarm argues that Lurie teaches that the ATD Mechanism distributes the tasks to the TCWs, thus "[w]hether by direct distribution, automatic distribution via the ATD mechanism, or in batch mode, Lurie affirmatively teaches away from a co-processor configured to autonomously and proactively 'retrieve' a task." Prelim. Resp. 49 (citing Ex. 1007 ¶¶ 6, 43–44).

Second, Swarm argues the "Lurie's Object Exchange Repository embodiment is substantively the same as the conventional 'controller/responder' paradigm that the '275 Patent expressly overcomes." Prelim. Resp. 51. According to Swarm, "a TCW must register with the Object Exchange Repository and only retrieves a task in response to a notification from the Object Exchange Repository. Lurie's TCW is *not* proactive or autonomous like the co-processors of the '275 Patent." *Id.*

Third, Swarm contends that "none of Lurie's operational modes (direct, automatic, batch) disclose a TCW retrieving a task without being

prompted to do so." Prelim. Resp. 48. Specifically, Swarm argues that, in Lurie's direct distribution mode, "the TCWs do not retrieve/take tasks because the tasks are directly sent to the TCW by the ATD," and in Lurie's automatic task distribution mode and batch mode, "the TCWs do not proactively retrieve/take tasks because they require a notification from the TCW before obtaining a task." *Id.* at 53–54 (citing Ex. 1007, Figs. 5A, 5B, 5C). In addition, Swarm argues that "Lurie's JavaSpace (Object Exchange Repository) embodiments do not alter any of the above modes of operation, but rather simply describe the environment in which the modes are implemented." *Id.* at 54.

(3) Our Analysis

Based on the preliminary record, Amazon has shown sufficiently that Lurie teaches or suggests the recited "first co-processor" limitation. Specifically, Lurie teaches how a TCW "takes a task from the [ATD Mechanism] 260 for technical computing of the task, computes a result for the task and provides the result to the [ATD Mechanism] 260." Ex. 1007 ¶ 72. A person having ordinary skill in the art would have realized that providing the results back to the ATD Mechanism teaches "updating the task pool to reflect completion of the first task." *Id.* ¶ 170. Additionally, Lurie shows that the TCWs communicate only with the ATD Mechanism and not the TCC. *See id.* ¶¶ 61, 72, 94, Figs. 3B, 5B, 6A; *see also* Ex. 1003 ¶¶ 173–80.

We do not agree with Swarm's arguments. With regard to Swarm's first argument, it is true that in some places Lurie describes "distributing" the tasks to the co-processors. *See* Ex. 1007 ¶ 6 ("The method further includes the step of *distributing* the job to remote computing workers for the distributed execution of the job." (emphasis added)); *id.* ¶ 43 ("The task

can be *directly distributed* by the technical computing client to one or more technical computing workers." (emphasis added)); *id.* ¶ 44 ("Additionally, a task or a group of tasks, in a job, can be submitted to an automatic task distribution mechanism to *distribute* the one or more tasks automatically to one or more technical computing workers providing technical computing services." (emphasis added)). However, Lurie also teaches other embodiments in which the TCW takes the tasks. *See* Ex. 1007 ¶ 72 ("A technical computing worker 270A-270N *takes* a task from the automatic task distribution mechanism 260 for technical computing of the task . . . ." (emphasis added)). Accordingly, based on the preliminary record, Lurie appears to describe two different embodiments, one in which the TCWs take tasks (which Amazon relies upon) and another in which the tasks are distributed. The embodiment which Amazon relies on is sufficient for purposes of institution when considered in combination with Ethernet Specification.

Specifically, we agree with Amazon that "Lurie's TCWs 'proactively retrieve' tasks because Lurie is implemented over a network where each TCW has an agent that identifies available tasks appropriate to that TCW from the task pool . . . and retrieves an available, appropriate task to deliver to the TCW." Pet. 27 (citing Ex. 1007 ¶¶ 43, 81, 91; Ex. 1003 ¶ 183). The '275 patent states that "[t]he term proactive suggests that each co-processor may be configured (e.g., programmed) to periodically send an agent to monitor the task pool for available tasks appropriate to that co-processor." Ex. 1001, 2:48–51. In addition, the '275 patent states that "an agent is generally analogous to a data frame in the networking sense, in that an agent may be equipped with a source address, a destination address, and a payload." Ex. 1001, 8:45–48; *see id.* at 3:21–25 (explaining that "the term

agent refers to a software module, analogous to a network packet"). In Lurie, "[t]he [TCW], in response to the notification, performs a take operation on the object exchange repository 662 to retrieve the submitted task object." Ex. 1007 ¶ 93. As Swarm highlights in its third argument, the take operation occurs only in response to the notification. Prelim. Resp. 52–53. However, Amazon contends that "Lurie's TCWs communicate with the ATD Mechanism via 'LAN, 'WAN,' or 'the Internet,'" and Lurie's JavaSpace "does not rely on passing messages directly between processes or invoking methods directly on remote objects." Pet. 27–28 (citing Ex. 1007 ¶¶ 51, 52, 71, 93, Fig. 1B; Ex. 1003 ¶ 184); Ex. 1007 ¶ 91.

We agree with Amazon that "[a] [person having ordinary skill in the art] would have understood that when Lurie is implemented over Ethernet (or similar networking protocol), data frames act as agents of a TCW to retrieve tasks as described in the '275 patent." Pet. 28 (footnote omitted) (citing Ex. 1007 ¶ 93; Ex. 1003 ¶¶ 128, 184, 188–90). Specifically, Amazon argues that "a [person having ordinary skill in the art] would have understood that information or data is transmitted in accordance with any standard network protocol utilizing a data frame, network packet, or equivalent protocol data unit that constitutes an 'agent.'" *Id.* at 27–28 (citing Ex. 1007, ¶¶ 51–52, 71, 91, Fig. 1B; Ex. 1003 ¶ 184). Amazon further argues that "Ethernet Specification describes just one such protocol that would have been obvious to use to implement Lurie, and that nearly all such protocols commonly known in the art . . . utilize data frames or network packets ('agents') having source, destination, and payload or data fields." *Id.* at 28 n.9 (citing Ex. 1003 ¶ 187; Ex. 1018, 1, 34–35; Ex. 1019, 1; Ex. 1023, 1338–40; Ex. 1024, 18–19). Therefore, based on the preliminary record, we find that a person having ordinary skill in the art would have had

sufficient reasons to implement Lurie's automatic task distribution mode over the Ethernet, as taught by the Ethernet Specification, thereby creating a "proactive" TCW "configured (e.g., programmed) to periodically send an agent to monitor the task pool for available tasks appropriate to that co-processor." Ex. 1001, 2:48–51; Ex. 1003 ¶ 188.

Finally, Swarm's second argument addresses Lurie's Object Exchange Repository embodiment implemented in JavaSpace and argues that Lurie does not teach a TCW proactively retrieving a task "without a predicate instruction or notification." Prelim. Resp. 52. Swarm contends that "[t]he only difference is that in Lurie's Object Exchange Repository embodiment, it is the Object Exchange Repository that becomes consumed with 'task distribution,'" not the ATD Mechanism. *Id.* We agree with Swarm that Lurie's JavaSpace implementation of an Object Exchange Repository still requires that "[a] [TCW] *registers with the object exchange repository 662 to be notified when a task object appears* in the object exchange repository 662." *Id.* at 51–52 (quoting Ex. 1007 ¶ 93 (emphasis added)). However, as discussed above, we find that Amazon has presented sufficient evidence that would support a finding that the combination of Lurie and Ethernet Specification teaches the "first co-processor" limitation, particularly when Lurie's automatic task distribution mode is implemented over the Ethernet. *See* Pet. 27 (citing Ex. 1007 ¶¶ 43, 81, 91; Ex. 1003 ¶ 183).

d) "a second co-processor"

Independent claim 1 further recites "a second co-processor configured to successively: proactively retrieve a second task from the task pool; process the second task; generate second resulting data; and update the task pool to reflect completion of the second task, all without any communication between the second co-processor and the controller." Ex. 1001, 14:34–39.

(1) The Parties' Arguments

Amazon argues that Lurie teaches this limitation. Pet. 29–31. Specifically, Amazon argues that "Lurie discloses multiple TCWs." *Id.* at 29 (citing Ex. 1007 ¶ 71 (Lurie's ATD Mechanism "is in communications with a plurality of" TCWs), Fig. 3C; Ex. 1003 ¶¶ 193–94). Amazon further argues that for the same reason Lurie teaches:

> "a *first* co-processor" (any one TCW), configured to proactively retrieve a *first* task from the task pool, process the *first* task, generate *first* resulting data, and update the task pool to reflect completion of the *first* task for Element [1c], Lurie also discloses a "*second* co-processor" (any other TCW), configured to proactively retrieve a *second* task from the task pool, process the *second* task, generate *second* resulting data, and update the task pool to reflect completion of the *second* task.

Id. at 30–31 (citing Pet. § V.A.1.c; Ex. 1003 ¶¶ 195–97).

Swarm does not directly dispute this element, but argues that "a version of [the first co-processor] claim element is recited expressly, or by incorporation, by every claim of the '275 Patent." Prelim. Resp. 54. Accordingly, Swarm's arguments with respect to the "first co-processor" limitation equally apply to the "second co-processor" limitation as well. *See id.* 46–54.

(2) Our Analysis

Based on the preliminary record, Lurie teaches using multiple TCWs. Ex. 1007 ¶ 71 ("The [ATD] mechanism 260 is in communications with a plurality of technical computing workers 270A–270N."); *id.* Fig. 3B. Amazon distinctly identifies a first TCW as the first processor and a second TCW as the second co-processor. Pet. 23 n.8. For the same reasons discussed in the proceeding section (*see supra* Part II.D.3.c.3), we find that Amazon presents sufficient evidence that would support a finding that

Lurie's second TCW meets the requirements of the "second co-processor" limitation.

e) "the plurality of first tasks and the plurality of second tasks are associated with a common objective"

Independent claim 1 further recites "the plurality of first tasks and the plurality of second tasks are associated with a common objective." Ex. 1001, 14:45–46.

(1) The Parties' Arguments

Amazon argues that Lurie teaches this limitation. Pet. 32–33. Specifically, Amazon contends that, in Lurie's "'batch mode,' tasks are associated with a job (e.g., Task1 through TaskN-M associated with Job1)," and that "[a] [person having ordinary skill in the art] would have understood that Lurie's jobs reflect a common objective with which pluralities of first tasks and pluralities of second tasks are collectively or respectively associated." *Id.* at 33 (citing Ex. 1003 ¶¶ 212–13). In its Preliminary Response, Swarm does not address this limitation.[8] *See* Prelim. Resp.

(2) Our Analysis

Based on the preliminary record, Lurie teaches the "common objective" limitation. Apart from the claims, the '275 patent only discusses a common objective when explaining the term "solidarity," which "implies that co-processing cells share a *common objective* in monitoring and executing all available tasks within the task pool." Ex. 1003 ¶ 212 (quoting

[8] Although Swarm does not address this "common objective" limitation, we analyze this limitation as if it were disputed because Amazon expressly relies on Lurie's batch mode to teach this limitation and, as we discuss above, we question whether Amazon has shown that Lurie's batch mode teaches the other limitations of independent claim 1.

Ex. 1001, 2:51–54). Lurie describes "distribut[ing] a portion of the job to one or more remote workers on a network," where "[t]he workers execute the received portion of the job and may return execution results to the client." Ex. 1007, code (57), ¶ 43 (describing Lurie's "distributed computing system," configured to complete a "job," which is "a logical unit of activities, or tasks that are processed and/or managed collectively"); Ex. 1003 ¶ 133. Moreover, Lurie states that "the present invention allows the use of instrument-based distributed system on a network to conduct the job and facilitate decreasing the time for executing the job." Ex. 1007, code (57). Therefore, because Lurie describes both a common objective of decreasing the time for executing a job and a system for distributing *portions* of the job to several co-processors, we find that Lurie teaches a plurality of first and second tasks associated with a common objective.

Although Amazon expressly relies on Lurie's batch mode to teach the "common objective" limitation, we question whether Amazon has shown that Lurie's batch mode teaches the other limitations of independent claim 1 and instead rely on Lurie's automatic task distribution mode to teach those limitations. *See supra* Part II.D.3.b.3. Specifically, we find that Lurie's automatic task distribution mode teaches the "common objective" limitation because the "[TCWs] can also communicate and interact with each other . . . to execute tasks both serially and in parallel by submitting a task to another [TCW]." Ex. 1007 ¶ 72. This is because, according to Lurie, "a large computational job can be performed in a distributed fashion . . . to facilitate decreasing the time for executing the job." *Id.* ¶ 5.

f) "the first and second co-processors autonomously work together in solidarity with the task pool to complete the common objective"

Independent claim 1 further recites "the first and second co-processors autonomously work together in solidarity with the task pool to complete the common objective." Ex. 1001, 14:47–49.

(1) The Parties' Arguments

Amazon argues that Lurie alone or in combination with Ethernet Specification teaches this limitation. Pet. 33–35. Specifically, Amazon contends that "[t]he TCWs (co-processors) work together in solidarity with the ATD Mechanism (task pool) because the TCWs (via their agents) retrieve available tasks from the ATD Mechanism and execute all such tasks to complete the job." *Id.* at 34 (citing Ex. 1003 ¶ 216). In addition, Amazon argues that Lurie's co-processors work together in solidarity because "Lurie discloses that all TCWs (co-processors) share the common objective as all TCWs monitor and execute all available tasks within the ATD Mechanism (task pool)." *Id.* (citing Ex. 1003 ¶ 217). Finally, Amazon contends that Lurie's TCWs are autonomous because "Lurie teaches that the TCWs (co-processors) interact with the ATD Mechanism (task pool) without being instructed to do so by the TCC or the ATD Mechanism because the TCWs send agents to retrieve tasks for execution." *Id.* at 34–35 (citing Ex. 1007, Fig. 5B; Ex. 1003 ¶ 218). In its Preliminary Response, Swarm does not

address this limitation with respect to independent claim 1.[9] *See* Prelim. Resp.

(2) Our Analysis

We find that Amazon has shown sufficiently that Lurie in combination with Ethernet Specification teaches this limitation. As discussed above, we find that Lurie's TCWs work together in solidarity to complete a common objective because Lurie describes both a common objective of decreasing the time for executing a job and a system for distributing *portions* of the job to several TCWs. *See* Ex. 1007, code (57), ¶ 43; Ex. 1003 ¶ 133.

Further, we find that the combination of Lurie and Ethernet Specification teaches co-processors autonomously working together. The '275 patent states that "the term autonomous implies that a co-processor may interact with the task pool without being instructed to do so by the CPU or by the task pool." Ex. 1001, 2:45–48. As discussed in relation to the claim term "proactive," we find that Lurie's TCWs may interact with the ATD Mechanism without being instructed to do so by the TCC or by the ATD Mechanism. Specifically, Lurie teaches that the "[TCC] is in communication with the [ATD Mechanism] running on the server," which "is in communications with a plurality of [TCWs]." Ex. 1007 ¶ 71, Fig. 5B. Lurie also teaches that "the [TCC] is not required to have any specific

[9] Swarm contends that neither Lurie nor the Ethernet Specification, alone or in combination with any other cited reference, teach or suggest "autonomous co-processors [that] operate . . . by: proactively retrieving a task having a compatible task type from the task pool" as required by each of claims 11–17. Prelim. Resp. 54–55. However, Swarm focuses on the "proactively retrieving" language and relies on the same reasons discussed above for the "first co-processor" limitation. *See id.*; *see also supra* Part II.D.3.c.

knowledge of the [TCWs]." Further, we agree with Amazon that "[a] [person having ordinary skill in the art] would have understood that when Lurie is implemented over Ethernet (or similar networking protocol), data frames act as agents of a TCW to retrieve tasks as described in the '275 patent." Pet. 27–28 (footnote omitted) (citing Ex. 1007 ¶ 93; Ex. 1003 ¶¶ 128, 184, 188–90). Therefore, Lurie's TCWs may interact with the ATD Mechanism without being instructed to do so by the TCC or ATD Mechanism and may do so to work together to complete a common objective, particularly when Lurie's automatic task distribution mode is implemented over the Ethernet.

g) Conclusion Regarding Independent Claim 1

For the reasons set forth above, Amazon has demonstrated a reasonable likelihood of prevailing on its assertion that the subject matter of independent claim 1 would have been obvious over the combined teachings of Lurie and Ethernet Specification.

4. Claims 3, 6, 11, 12, 14, 16, and 17

Because Amazon has demonstrated a reasonable likelihood of success in demonstrating that at least one claim of the '275 patent is unpatentable, we institute on all challenged claims and all grounds raised in the Petition. *See* 37 C.F.R. 42.108(a). Therefore, at this preliminary stage, it is not necessary for us to provide an assessment of every challenge raised by Amazon. However, we address the additional argument that Swarm advances with respect to independent claim 11 below.

a) "each of the tasks has a respective task type"

Independent claim 11 recites, in relevant part, "populating a task pool, with a controller, such that the task pool includes a plurality of tasks

associated with the objective, wherein each of the tasks has a respective task type." Ex. 1001, 16:27–30.

(1) The Parties' Arguments

Amazon argues that Lurie teaches this limitation. Pet. 48. Specifically, Amazon contends that "Lurie's TCC is a controller that populates the ATD Mechanism with a plurality of 'tasks,'" and "each task has an indicia of task type that matches the respective TCW's 'operational and/or performance characteristics.'" *Id.* (quoting Ex. 1007 ¶ 81) (citing Ex. 1003 ¶ 287). In response, Swarm contends that Lurie does not teach a controller populating the task pool with a plurality of tasks or that the tasks have any indicia of task type. Prelim. Resp. 43–46. Moreover, Swarm argues that an indicia of task type would not "be useful given that Lurie's TCWs do not search the ATD Mechanism for a task that has a task type that they are capable of performing." *Id.* at 45.

(2) Our Analysis

We find that Amazon has shown sufficiently that Lurie teaches the "each of the tasks has a respective task type" limitation. As discussed above, we find that Amazon has presented sufficient evidence that would support a finding that Lurie teaches that the TCC populates the ATD Mechanism with a plurality of first tasks and a plurality of second tasks. *See supra* Part II.D.3.b.3; Ex. 1007, Fig. 3B, ¶¶ 71, 72. In addition, we find that Amazon has presented sufficient evidence that would support a finding that each of Lurie's tasks has a task type. The '275 patent states that "[a] task 22 may have a task type and a descriptor. The task type indicates which cells 12 are capable of performing the task 22." Ex. 1001, 7:30–32. Based on the preliminary record, we find that Lurie's tasks may contain different technical computing commands, arguments, and input data that constitute a task type

or descriptor, especially because Swarm has not proposed constructions of task type or descriptor that would preclude such classification. *See supra* Part II.D.3.b.3; Ex. 1001 ¶ 47 ("A task defines a technical computing command, such as a MATLAB® command, to be executed, and the number of arguments and any input data to the arguments.").

Lurie also teaches that "the system can determine or select the [TCW] to work on a task by operational and/or performance characteristics of the [TCW]." Ex. 1007 ¶ 81. We find that Lurie's ATD Mechanism would not be able to identify appropriate tasks for the TCWs based on each TCW's characteristics unless the tasks were also distinguishable by a task type or descriptor. Therefore, based on the preliminary record, we find that Lurie teaches that each of the tasks has a task type.

b) Conclusion Regarding Claims 3, 6, 11, 12, 14, 16, and 17

We note that Amazon provides detailed explanations supported by the testimony of Dr. Lowenthal and specific citations to the relevant references indicating where in each reference Amazon argues the limitations of claims 3, 6, 11, 12, 14, 16, and 17 are taught, and why a person having ordinary skill in the art would have combined the various teachings of Lurie and Ethernet Specification with a reasonable expectation of success. *See* Pet. 14–71. Accordingly, we are persuaded that the information presented in the Petition establishes that there is a reasonable likelihood that Amazon would prevail on its assertion that the subject matter of claims 3, 6, 11, 12, 14, 16, and 17 would have been obvious over the combined teachings of Lurie and Ethernet Specification.

E. Asserted Obviousness in View of Lurie, Ethernet Specification, Andrews, and/or Chow

Amazon argues that claims 2, 4, 7, 9, 10, 12, and 15 would have been obvious over Lurie, Ethernet Specification, Andrews and/or Chow. *See* Pet. 56–65. Based on the preliminary record, we are sufficiently persuaded that Amazon has established a reasonable likelihood of prevailing on this asserted obviousness ground with respect to at least one of claims 2, 4, 8, 9, 10, 12, and 15.

1. Overview of Andrews

Andrews is entitled "Foundations of Multithreaded, Parallel, and Distributed Programming" and is a book directed to "the history of concurrent programming." Ex. 1011, v–vi. Andrews describes a "way to implement parallel computations using what is called a *bag of tasks*," where "[a] task is an independent unit of work" and "[t]asks are placed in a bag that is shared by two or more worker processes." *Id.* at 132. Andrews explains that this approach can "be used to solve iterative problems having a fixed number of independent tasks." *Id.* Andrews also explains that "[t]he bag-of-tasks paradigm has several useful attributes," including easy implementation, scalability, and load balancing. *Id.* at 132–33.

2. Overview of Chow

Chow is entitled "Method for Hiding Latency in a Task-Based Library Framework for a Multiprocessor Environment." Ex. 1008, code (54). Chow is directed to "a task-based library framework for dynamic load balancing in a multiprocessor environment, and to a method of latency hiding in this framework." *Id.* ¶ 1.

More specifically, Chow discloses "[a] task-based library framework for load balancing using a system task queue in a tightly-coupled

multiprocessor system. The system memory holds a queue of system tasks. The library processors fetch tasks from the queue for execution." Ex. 1008, code (57). Chow further discloses that "[t]he library processors fetch the tasks when they have a light load. A library processor can fetch a task while executing another task." *Id.*

Chow's Figure 1 is reproduced below.

FIG. 1

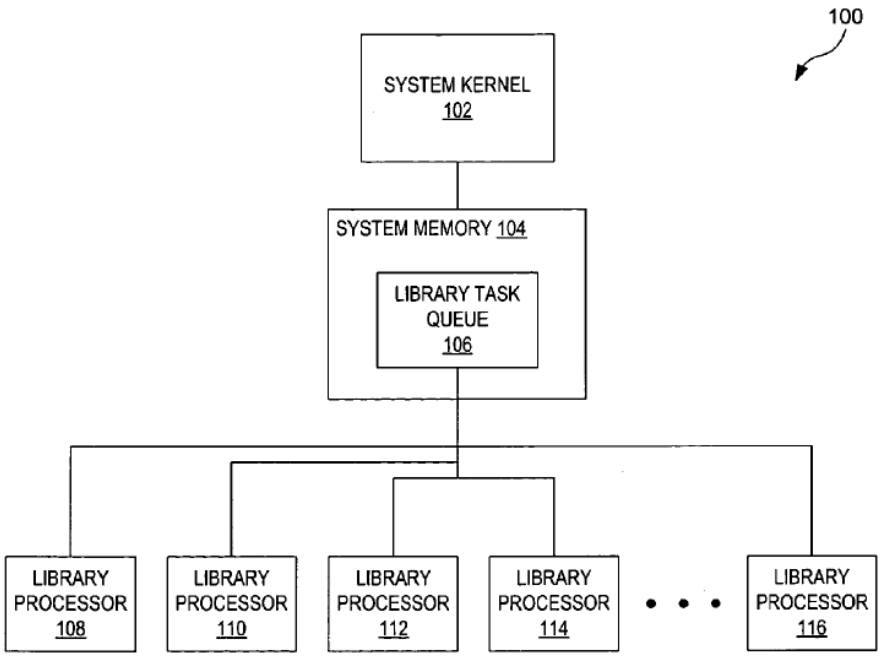

Figure 1, reproduced above, "schematically depicts a tightly-coupled multiprocessor system with a task-based library framework and library processors." Ex. 1008 ¶ 7. In connection with Figure 1, Chow discloses the following:

> the reference numeral 100 generally designates a tightly-coupled multiprocessor system with a task-based library framework. The system 100 comprises a system kernel 102, a system memory 104, and a number of library processors, 108, 110, 112, 114, and 116. The ellipsis indicates that the system 100 can comprise additional library processors. The system memory comprises a queue of tasks 106 to be assigned to the library processors

> (library task queue 106). Each library processor has access to the library task queue 106. When tasks arrive at the system kernel 102 for processing, they are subdivided into subtasks and placed into the library task queue 106. The library processors 108, 110, 112, 114, and 116 fetch the subtasks from the library task queue 106.

Id. ¶ 13.

3. Dependent Claims 2, 7, 10, and 13

For example, claim 2 depends directly from independent claim 1 and claim 7 depends directly from independent claim 6. Dependent claim 2 further recites "wherein the first co-processor is configured to perform at least one of depositing a new task into the task pool and modifying an existing task within the task pool." Ex. 1001, 14:50–53. Dependent claim 7 recites substantially the same limitation as claim 2. *Compare id.* at 14:50–53, *with id.* at 16:8–11.

Amazon argues that Lurie alone or in combination with Andrews teaches these limitations because Lurie's "TCWs are configured to 'write' a result to task objects within the ATD Mechanism, thereby modifying an existing task." Pet. 56 (citing Ex. 1003 ¶ 357). In addition, Amazon argues that "Andrews teaches that 'executing one task' in a bag-of-tasks architecture (such as Lurie), 'will usually lead to producing two smaller tasks, which a worker [co-processor] needs to put back into the bag [task pool].'" *Id.* at 57 (quoting Ex. 1011, 135) (citing Ex. 1011, 132; Ex. 1003 ¶¶ 99–100). Amazon further contends that "[a] [person having ordinary skill in the art] would have been additionally motivated to combine Lurie (alone or [together] with Ethernet Specification), with Andrews such that Lurie's TCWs deposit tasks in the ATD Mechanism." Pet. 57 (citing Ex. 1003 ¶ 30). According to Amazon, "[a] [person having ordinary skill in the art]

would have implemented this task re-depositing in Lurie in accordance with the details provided in Andrews and to optimize Lurie's system by increasing load balancing and system efficiency." *Id.* (citing Ex. 1003 ¶¶ 331–34; Ex. 1011, 135).

Based on the preliminary record, Amazon's arguments and supporting evidence with respect to dependent claims 2, 7, 10 and 13 are sufficient for purposes of institution. At this stage of the proceeding, Swarm does not address separately Amazon's arguments and supporting evidence as to how the teachings of Lurie, Ethernet Specification, and Andrews account for the limitations of these dependent claims. *See* Prelim. Resp. When considering Amazon's unrebutted arguments and supporting evidence, Amazon has demonstrated a reasonable likelihood that it will prevail on its assertions that the subject matter of dependent claims 2, 7, 10, and 13 would have been obvious over the teachings of Lurie, Ethernet Specification, and Andrews.

4. Dependent Claims 4, 9, and 15[10]

Claim 4 depends directly from independent claim 1, claim 9 depends indirectly from independent claim 6, and claim 15 depends directly from independent claim 11. Dependent claim 4 further recites "wherein the first co-processor is configured to determine when it has available processing capacity, and to dispatch the first agent to the task pool in response to the determination." Ex. 1001, 14:66–15:3. Dependent claims 9 and 15 recite substantially the same limitation as dependent claim 4. *Compare id.* at 14:66–15:3, *with id.* at 16:17–21, *and id.* at 16:53–56.

[10] Amazon relies on Lurie, Ethernet Specification, Andrews and Chow in its substantive analysis of dependent claims 4, 9, and 15. *See* Pet. 2, 58–63. Accordingly, we consider Amazon's reliance on Andrews for each of these dependent claims. *See* discussion *supra* note 3.

Amazon contends that "Lurie alone or [in combination] with Chow teaches" these limitations because "[a] [person having ordinary skill in the art] would have implemented Lurie so that its co-processors do not fetch new tasks from the pool unless or until the co-processors had sufficient capacity." Pet. 59 (citing Ex. 1003 ¶¶ 335–36, 358–61). Further, Amazon argues that a person having ordinary skill in the art would have found it obvious to check the availability of each co-processor to take on new tasks "to avoid the undesirable scenarios of having either (1) too few processors for the work load, which wastes time while processors work through their backlog, or (2) too many processors, which results in idle processors wasting processing cycles or power." *Id.* at 59–60 (citing Ex. 1003 ¶¶ 338–40; Ex. 1011, 132–33 (noting that a "useful attribute[]" of Andrews's bag-of-tasks architecture is that "one can use any number of processors," and therefore it is "easy to implement load balancing")).

Specifically, Amazon contends that, "[w]hen combined with Chow, Lurie's co-processors (TCWs) retrieve and process tasks by writing the data associated with each task to a buffer within the TCW. As in Chow, each TCW increases efficiency by checking whether it has an existing task within its buffer before the TCW dispatches an agent to the task pool to retrieve a new task." *Id.* at 62 (citing Ex. 1003 ¶¶ 367–71); Ex. 1008 ¶¶ 18–21, Figs. 1, 2, 4; Ex. 1003 ¶¶ 341–43, 363–66, 367–71. Amazon further argues that a person having ordinary skill in the art would have been motivated to combine the teachings of Lurie and Chow because "[i]mplementing Chow's buffers to check the available processing capacity of each TCW would further Lurie's goal of 'decreasing the time for executing [a] job.'" Pet. 63 (quoting Ex. 1007 ¶ 5) (citing Ex. 1008 ¶ 21; Ex. 1007, code (57); Ex. 1003 ¶ 343).

Based on the preliminary record, Amazon's arguments and supporting evidence with respect to dependent claims 4, 9, and 15 are sufficient for purposes of institution. At this stage of the proceeding, Swarm does not address separately Amazon's arguments and supporting evidence as to how the teachings of Lurie, Ethernet Specification, Andrews, and Chow account for the limitations of these dependent claims. *See* Prelim. Resp. When considering Amazon's unrebutted arguments and supporting evidence, Amazon has demonstrated a reasonable likelihood that it will prevail on its assertions that the subject matter of dependent claims 4, 9, and 15 would have been obvious over the teachings of Lurie, Ethernet Specification, Andrews, and Chow.

F. Asserted Obviousness in View of Lurie, Ethernet Specification, and Sander

Amazon argues that claim 5 would have been obvious over Lurie, Ethernet Specification, and Sander. *See* Pet. 65–68. Based on the preliminary record, we are sufficiently persuaded that Amazon has established a reasonable likelihood of prevailing on this asserted obviousness ground with respect to dependent claim 5.

1. Sander

Sander is entitled "Dynamic Work Partitioning on Heterogeneous Processing Devices." Ex. 1010, code (54). Sander is directed to "an architecture for unifying the computational components within a computer system." *Id.* ¶ 3.

Sander describes various elements of a computer system including a CPU and an accelerated processing device ("APD"). Ex. 1010 ¶¶ 5, 12–13. According to Sander, both the CPU and the APD can be "formed on a single

silicon die or package, combining CPU 102 and APD 104 to provide a unified programming and execution environment." *Id.* ¶ 28.

2. Dependent Claim 5

Claim 5 depends directly from independent claim 1 and further recites "wherein the controller and the task pool reside on a monolithic integrated circuit (IC) that is not a component of either the first or second co-processors." Ex. 1001, 15:4–7. Amazon argues that the combination of Lurie and Sander teaches this limitation because "[p]lacing Lurie's TCC and ATD Mechanism on the same monolithic IC would decrease potential latency and prevent against loss of data over a network connection," while "plac[ing] Lurie's TCWs on separate circuits from the TCC and ATD Mechanism" would "preserv[e] the scalability of its system." Pet. 67 (citing Ex. 1003 ¶¶ 392–93; Ex. 1007 ¶ 73). Amazon contends that "[a] [person having ordinary skill in the art] would therefore consider it obvious to use 'remote' TCWs on 'workstations' as described in Lurie, preserving the ability to later connect specialized hardware, while keeping the TCC and ATD Mechanism on the same chip, which preserves both flexibility and scalability of the system." *Id.* at 68 (citing Ex. 1007, Fig. 3C; Ex. 1003 ¶ 396). Amazon also argues that "[a] [person having ordinary skill in the art] would have understood that the interaction among and roles of the components in Sander's system are analogous to those of Lurie's, and likely to be successful for that reason." *Id.* at 66 (citing Ex. 1010 ¶¶ 8, 11, 14; Ex. 1003 ¶¶ 378–80, 390–91).

Based on the preliminary record, Amazon's arguments and supporting evidence with respect to dependent claim 5 are sufficient for purposes of institution. At this stage of the proceeding, Swarm does not address separately Amazon's arguments and supporting evidence as to how the

teachings of Lurie, Ethernet Specification, and Sander account for the limitation of this dependent claim. *See* Prelim. Resp. When considering Amazon's unrebutted arguments and supporting evidence, Amazon has demonstrated a reasonable likelihood that it will prevail on its assertions that the subject matter of dependent claim 5 would have been obvious over the teachings of Lurie, Ethernet Specification, and Sander.

G. Asserted Obviousness in View of Lurie, Ethernet Specification, Andrews, and Marjovi

Amazon argues that dependent claim 8 would have been obvious over Lurie, Ethernet Specification, and Marjovi. *See* Pet. 68–71. Based on the preliminary record, we are sufficiently persuaded that Amazon has established a reasonable likelihood of prevailing on this asserted obviousness ground with respect to dependent claim 8.

1. Overview of Marjovi

Marjovi is entitled "Robotic Clusters: Multi-Robot Systems as Computer Clusters: A Topological Map Merging Demonstration." Ex. 1026, 1191. Marjovi is directed to "the novel concept of *robotic clusters* to empower these systems in their problem solving." *Id.*

Marjovi describes multi-robot systems and robotic clusters, which are "group[s] of individual robots which are able to share their processing resources." Ex. 1026, 1191. Marjovi explains that "the robots can solve difficult problems by using the processing units of other robots," which leads to "advantages over single robot systems, namely higher task accomplishment speed, higher accuracy and fault tolerance." *Id.*

2. Dependent Claim 8

Claim 8 depends indirectly from independent claim 6 and recites "[t]he collaborative intelligence system of claim 7, wherein the first co-

processor is an unmanned autonomous vehicle configured to operate as at least one of a ground vehicle and an aerial vehicle in connection with defense field operations." Ex. 1001, 16:12–16. Amazon contends that the combination of Lurie and Marjovi teaches this limitation because "Lurie discloses that each TCW is associated with a computing device, which 'can be any type of computing device . . . capable of computing and communicating the operations described.'" Pet. 70 (quoting Ex. 1007 ¶ 52) (citing Ex. 1007, Fig. 1B). Accordingly, Amazon argues that "[a] [person having ordinary skill in the art] would have been motivated to implement Marjovi's architecture according to Lurie's teachings by using one of Marjovi's unmanned autonomous ground vehicles (robots) as a TCW." *Id.* (citing Ex. 1003 ¶ 398). Specifically, Amazon argues that the proposed combination of Lurie and Marjovi would achieve a specific common objective ("topological map merging for unmanned autonomous vehicles") and "eliminate[] the need for any robot to spend processing resources distributing subtasks." *Id.* at 71 (citing Ex. 1003 ¶ 400, 402–03; Ex. 1026, 1191–95).

Based on the preliminary record, Amazon's arguments and supporting evidence with respect to dependent claim 8 are sufficient for purposes of institution. At this stage of the proceeding, Swarm does not address separately Amazon's arguments and supporting evidence as to how the teachings of Lurie, Ethernet Specification, Andrews, and Marjovi account for the limitation of this dependent claim. *See* Prelim. Resp. When considering Amazon's unrebutted arguments and supporting evidence, Amazon has demonstrated a reasonable likelihood that it will prevail on its assertions that the subject matter of dependent claim 8 would have been

obvious over the teachings of Lurie, Ethernet Specification, Andrews, and Marjovi.

III. CONCLUSION

Following § 314, we have determined that the totality of the information presented at this preliminary stage shows there is a reasonable likelihood that Amazon would prevail with respect to at least one of the challenged claims, and we institute on all challenged claims and all grounds raised in the Petition.

Our factual findings, conclusions of law, and determinations at this stage of the proceeding are preliminary and based on the evidentiary record developed thus far. This is not a final decision as to the patentability of claims for which IPR is instituted. Our final decision will be based on the record as fully developed during trial.

IV. ORDER

In consideration of the foregoing, it is hereby:

ORDERED that an IPR of all challenged claims of the ’275 patent is instituted with respect to all grounds raised in the Petition; and

FURTHER ORDERED that pursuant to 35 U.S.C. § 314(a), an IPR of the ’275 patent is hereby instituted commencing on the entry date of this Decision, and pursuant to 35 U.S.C. § 314(c) and 37 C.F.R. § 42.4, notice is hereby given of the institution of a trial.

FOR PETITIONER:

Jonathan M. Strang
Adam M. Greenfield
David Zucker
Kimberly Q. Li
Latham & Watkins LLP
jonathan.strang@lw.com
adam.greenfield@lw.com
david.zucker@lw.com
kimberly.li@lw.com

FOR PATENT OWNER:

Daniel R. Pote
Jennings Strouss & Salmon PLC
p@jsslaw.com
dpote@jsslaw.com

Michael K. Kelly
Beus Gilbert McGroder PLLC
mkelly@beusgilbert.com