

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

GOOGLE LLC,
Petitioner,

v.

DIALECT, LLC,
Patent Owner.

IPR2024-00752
Patent 8,849,652 B2

Before KRISTEN L. DROESCH, HYUN J. JUNG, and
GEORGE R. HOSKINS, *Administrative Patent Judges*.

HOSKINS, *Administrative Patent Judge*.

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

I. INTRODUCTION

Petitioner Google LLC filed a Petition (Paper 2, “Pet.”) pursuant to 35 U.S.C. §§ 311–319 to institute an *inter partes* review of U.S. Patent No. 8,849,652 B2 (Ex. 1001, “the ’652 patent”), claims 1–27. The Board issued an Institution Decision (Paper 11, “Inst. Dec.”) granting the petitioned review.

Patent Owner Dialect, LLC then filed a Patent Owner Response (Paper 15, “PO Resp.”). Petitioner filed a Reply (Paper 22, “Pet. Reply”) and Patent Owner filed a Sur-reply (Paper 28, “PO Sur-reply”).

An oral hearing was held on August 5, 2025. A transcript of the hearing is in the record (Paper 33, “Tr.”).

We have jurisdiction under 35 U.S.C. § 6(b)(4) and § 318(a). This Decision is a final written decision under 35 U.S.C. § 318(a) and 37 C.F.R. § 42.73 as to the patentability of claims 1–27 of the ’652 patent. We determine Petitioner has demonstrated the unpatentability of claims 1–7, 12–17, and 19–23, but not claims 8–11, 18, and 24–27.

II. BACKGROUND

A. *Real Parties-in-Interest and Related Proceedings*

Petitioner identifies itself (Google LLC) as the real party-in-interest. *See* Pet. 11 & n.1. Patent Owner identifies itself (Dialect, LLC) as the real party-in-interest. *See* Paper 5, 1.

Petitioner filed several other IPR proceedings to challenge patents owned by Patent Owner. *See* Pet. 11–12. They are: IPR2024-00746 (U.S. Patent No. 7,398,209 B2); IPR2024-00747 (U.S. Patent No. 7,502,738 B2); IPR2024-00748 (U.S. Patent No. 8,015,006 B2);

IPR2024-00749 (U.S. Patent No. 7,693,720 B2); IPR2024-00750 (U.S. Patent No. 9,031,845 B2); IPR2024-00751 (U.S. Patent No. 8,447,607 B2); and IPR2024-00753 (U.S. Patent No. 7,640,160 B2).

The parties also identify three district court litigations as possibly related to the '652 patent. *See* Pet. 11; Paper 5, 1; PO Resp. 12–14.

B. The '652 Patent Specification

The '652 patent describes a plurality of mobile devices, each having a speech interface enabling natural language human-machine interactions. *See* Ex. 1001, 1:20–25. That is, each device allows “mobile users to submit natural language speech and/or non-speech questions or commands in a wide range of domains” and then “present[s] responses in a natural manner.” *Id.* at 1:25–29, 3:33–38. Figure 1 of the '652 patent is reproduced below.

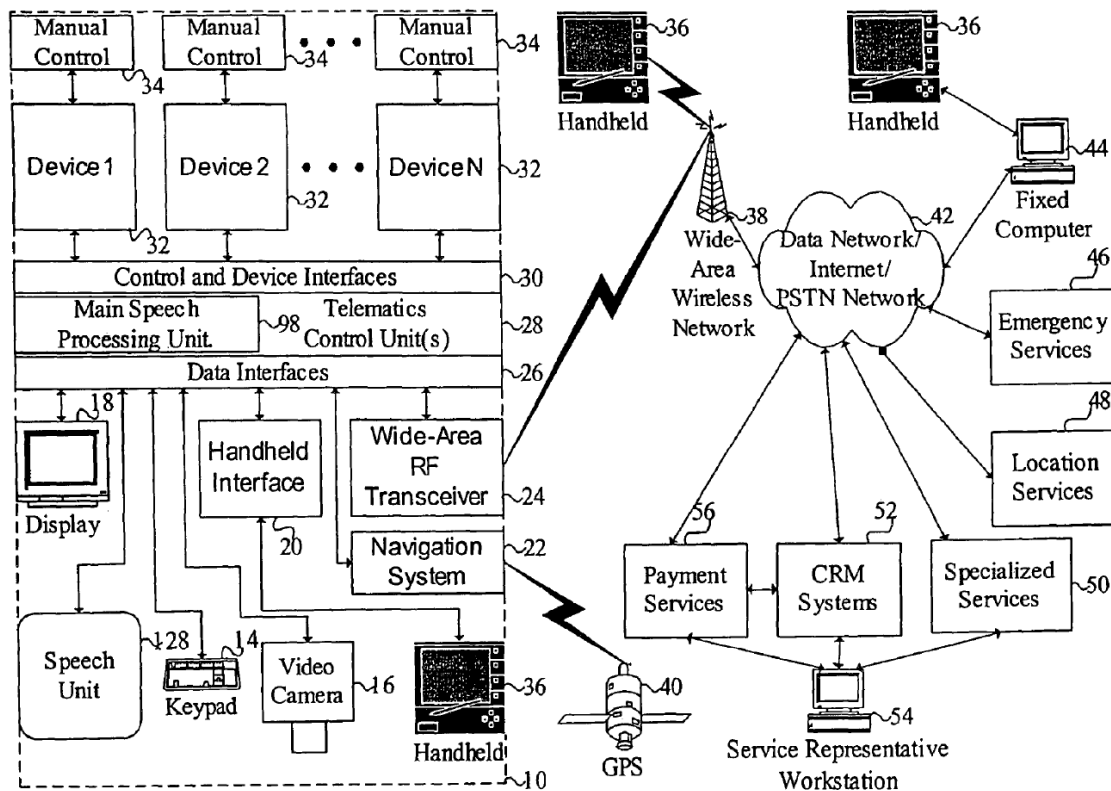


Figure 1

'652 Patent, Figure 1.

Figure 1 is a block diagram of a speech interface system according to one embodiment of the '652 patent's invention. *See id.* at 15:30–31, 15:58–64. The system includes mobile structure 10, several mobile devices 36, and fixed home or office computer system 44, all in communication with each other via wireless network(s) 38 and other data network(s) 42 such as the Internet. *See id.* at 15:65–16:1, 17:5–14.

Mobile structure 10 can be a vehicle such as an automobile, and includes telematics control unit 28 to control various devices 32 (e.g., door locks, interior temperature controls, cruise control, etc.) to execute commands spoken by users. *See id.* at 15:61–63, 16:7, 16:37–41, 16:64–17:4. Mobile structure 10 accordingly has speech unit 128 including microphone(s) and related circuitry to record natural language utterances by the user, and main speech processing unit 98 that parses the recorded utterances to discern and respond to user questions or commands. *See id.* at 16:6–12, 16:37–41, 25:48–54, 26:18–31.

Mobile devices 36 can include mobile telephones or personal digital assistants. *See id.* at 16:1–3. Each mobile device 36 has its own speech unit 128 and main speech processing unit 98 to enable its local processor to respond to natural language questions and commands from users. *See id.* at Fig. 3, 18:33–41, 18:54–62, 19:24–32. Fixed computer system 44 similarly has its own speech unit 128 and main speech processing unit 98. *See id.* at Fig. 4, 18:33–41, 21:61–22:12.

Each networked device 10, 36, and 44 tracks the “context” of verbal utterance questions and commands received and processed by the device, such as “time, location, numbers, dates, categories (e.g., music, movies, television, addresses, etc.) and other context.” *Id.* at 19:32–33, 19:49–55,

20:22–26, 22:12–13, 23:11–35. Each device maintains this context information in a “context stack” stored on the device, which “includes an ordered list of command contexts.” *Id.* at 20:10–12, 20:19–22. The context information utilized to determine the intent of subsequent verbal utterance questions and commands. *See id.* at 2:11–22, 2:58–67, 4:50–51, 6:3–16, 20:13–22, 29:42–51, 32:55–64, 33:67–34:18.

The context information maintained by each networked device 10, 36, and 44 can be synchronized between the devices. *See id.* at 17:15–21. Figure 7 shows how context manager 702 may maintain the contexts tracked by the various networked devices. *See id.* at 15:43–44, 19:33–41. Then, each networked device “may be informed of context changes through context tracking module 714, thereby synchronizing the context across the . . . devices.” *Id.* at 19:41–44; *see also id.* at 4:29–32 (“[M]obile devices may be informed of context changes through a context tracking module to enable synchronizing of context across the [devices].”). “The synchronization may be performed automatically when any two or more of the computers are connected to these networks,” or “the synchronization may be performed on demand under user control.” *Id.* at 17:21–25.

C. The '652 Patent Claims

The '652 patent includes twenty-seven claims, all of which are challenged here. *See Ex. 1001, 38:38–42:63; Pet. 13–14.* Claim 1 recites, with our emphases added:

1. A system for processing natural language utterances where recognized words of the natural language utterances alone are insufficient to completely determine one or more commands or requests, the system comprising:

one or more physical processors programmed with one or more computer program instructions which, when executed, cause the one or more physical processors to:

generate a first context stack associated with a first device, the first context stack comprising context information that corresponds to a plurality of prior utterances;

synchronize the first context stack with a second context stack associated with a second device such that the context information of the first context stack is updated based on related context information of the second context stack;

receive a natural language utterance associated with a command or request;

determine one or more words of the natural language utterance by performing speech recognition on the natural language utterance; and

determine the command or request based on the one or more words and the updated context information.

Ex. 1001, 38:39–61 (emphases added).

D. Asserted Prior Art

Petitioner relies on the following three references as prior art against all claims of the '652 patent. *See* Pet. 13–14.

Name	Reference	Exhibit No.
Coffman	WO 00/20962	1003
Kennewick	US 2004/0044516 A1	1004
Lee	US 2002/0087315 A1	1006

Patent Owner does not dispute that these references are prior art against all claims of the '652 patent. *See* PO Resp.

E. Asserted Grounds

Petitioner asserts the following three grounds of unpatentability. *See* Pet. 13–14.

35 U.S.C. § ¹	References	Claims Challenged
103(a)	Coffman	1–9, 14–18, 20–26
103(a)	Coffman, Kennewick	10–13, 19, 27
103(a)	Coffman, Kennewick, Lee	13, 19

F. Testimonial Evidence

Petitioner relies on the expert witness testimony of Mr. Christopher Schmandt (Exhibits 1002, 1017, 2011, 2014, and 2015). Patent Owner relies on the expert witness testimony of Mr. Benedict Occhiogrosso (Exhibits 1018 and 2013).

III. PETITIONER’S GROUNDS OF UNPATENTABILITY

A. Legal Standards

1. Petitioner’s Burden

“In an [*inter partes* review], the petitioner has the burden from the onset to show with particularity why the patent it challenges is

¹ Amendments to 35 U.S.C. § 103 enacted by the Leahy-Smith America Invents Act, Pub. L. No. 112-29, 125 Stat. 284 (2011) (“AIA”) took effect in March 2013. The ’652 patent issued from an application filed in May 2013, asserting priority to three applications filed before March 2013. *See* Ex. 1001, codes (22) & (60). Therefore, we cite to the pre-AIA version of § 103. The outcome of this Decision is not affected by which version applies.

unpatentable.” *Harmonic Inc. v. Avid Tech., Inc.*, 815 F.3d 1356, 1363 (Fed. Cir. 2016) (citing 35 U.S.C. § 312(a)(3) (requiring *inter partes* review petitions to identify “with particularity . . . the evidence that supports the grounds for the challenge to each claim”)). This burden of persuasion never shifts to the patent owner. *See Dynamic Drinkware, LLC v. Nat’l Graphics, Inc.*, 800 F.3d 1375, 1378 (Fed. Cir. 2015). The petitioner must prove unpatentability by a preponderance of the evidence. *See* 35 U.S.C. § 316(e); 37 C.F.R. § 42.1(d).

2. Obviousness

A patent claim is unpatentable under 35 U.S.C. § 103 if the differences between the claimed subject matter 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. *KSR Int’l Co. v. Teleflex Inc.*, 550 U.S. 398, 406 (2007). The question of obviousness is resolved on the basis of 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) objective evidence of nonobviousness, if made available in the record, which has not happened here. *See Graham v. John Deere Co.*, 383 U.S. 1, 17–18 (1966).

B. Level of Ordinary Skill in the Art

Petitioner contends a person having ordinary skill in the art pertaining to the ’652 patent “would have possessed a bachelor’s degree in electrical engineering or computer science, and two years of work experience in the field of speech- and natural language-interfaces for interacting with and

directing computer systems and software, including speech recognition and processing (and more experience could substitute for less formal education, and vice versa).” Pet. 16; Ex. 1002 ¶¶ 11–15.

Patent Owner does not oppose Petitioner’s formulation of ordinary skill. *See* PO Resp.; Ex. 2013 ¶¶ 28–31.

We are satisfied that Petitioner’s unopposed formulation of ordinary skill generally comports with the level of skill necessary to understand and implement the teachings of the ’652 patent and the asserted prior art. Therefore, we apply that formulation here.

C. Claim Construction

We interpret the ’652 patent claims “using the same claim construction standard that would be used to construe the claim in a civil action under 35 U.S.C. 282(b).” 37 C.F.R. § 42.100(b). This “includ[es] construing the claim in accordance with the ordinary and customary meaning of such claim as understood by one of ordinary skill in the art and the prosecution history pertaining to the patent.” *Id.*

Petitioner argues: “Express claim construction is unnecessary because the prior art renders the challenged claims obvious under any reasonable construction.” Pet. 18.

Patent Owner argues “Petitioner’s grounds include fundamental flaws” such that ruling in Patent Owner’s favor “do[es] not require the Board to conduct any construction of” claim terms. PO Resp. 14.

In Sections III.D.2(b)(ii) and (iii) below, we construe the claim terms “synchronize” and “context stack” as part of comparing claim 1 with Coffman.

Otherwise, based on the arguments and evidence presented, we determine no explicit construction is needed to resolve the parties' controversy. *See Realtime Data, LLC v. Iancu*, 912 F.3d 1368, 1375 (Fed. Cir. 2019) ("The Board is required to construe 'only those 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)).

D. Ground One — Obviousness over Coffman

In Ground One, Petitioner asserts claims 1–9, 14–18, and 20–26 of the '652 patent are unpatentable as having been obvious over Coffman. *See* Pet. 13–14, 18–60. Patent Owner disagrees. *See* PO Resp.

For reasons that follow, we determine Petitioner has demonstrated claims 1–7, 14–17, and 20–23 are unpatentable as having been obvious over Coffman, but has not demonstrated claims 8, 9, 18, and 24–26 are unpatentable based on this ground. First, we summarize the disclosure of Coffman, then we consider the parties' opposing contentions concerning the challenged claims.

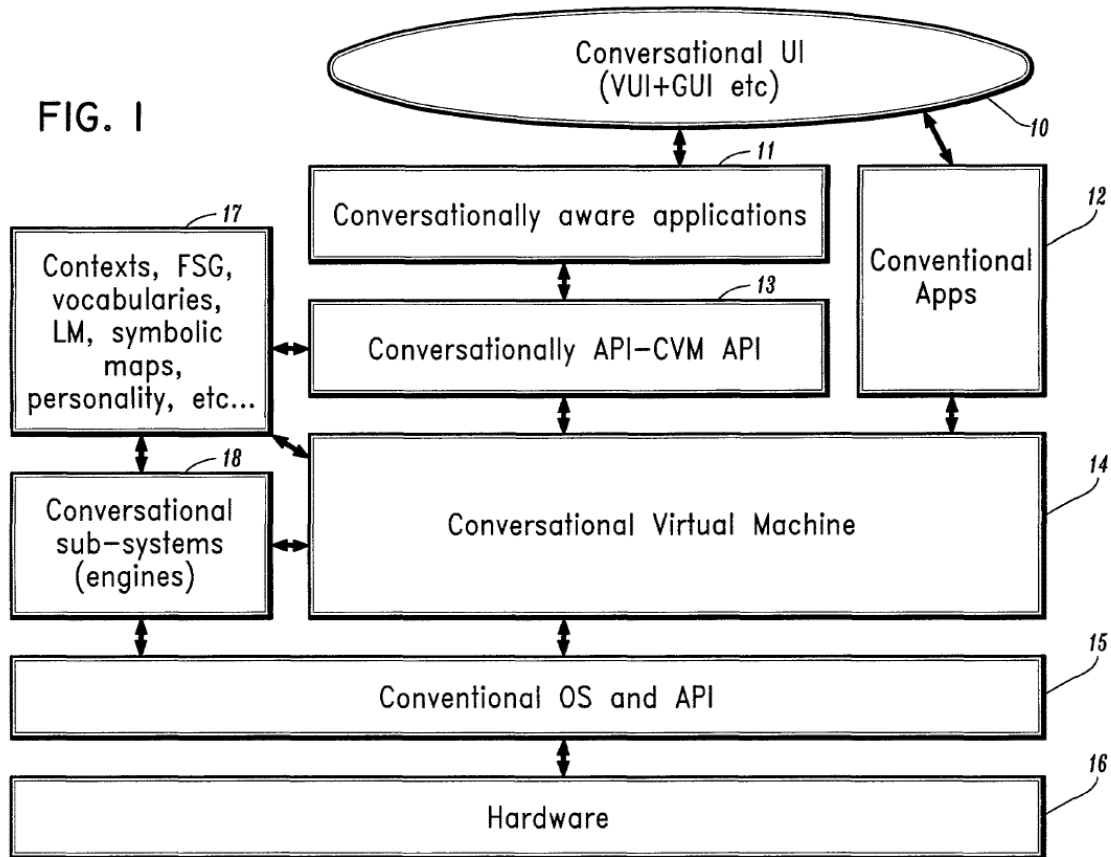
1. Coffman

Coffman discloses a "conversational virtual machine" (CVM) that "exposes conversational APIs (application program interface), conversational protocols and conversational foundation classes to application developers." Ex. 1003, 1:6–17.² Coffman's CVM "provides a kernel layer that is responsible for implementing conversational computing

² When citing to Coffman, we refer to the native pagination at the bottom center of each page.

by managing dialog and context, conversational engines and resources, and conversational protocols / communication across platforms and devices having different conversational capabilities to provide a universal CUI (conversational user interface).” *Id.* at 1:14–17.

Figure 1 of Coffman is reproduced below.

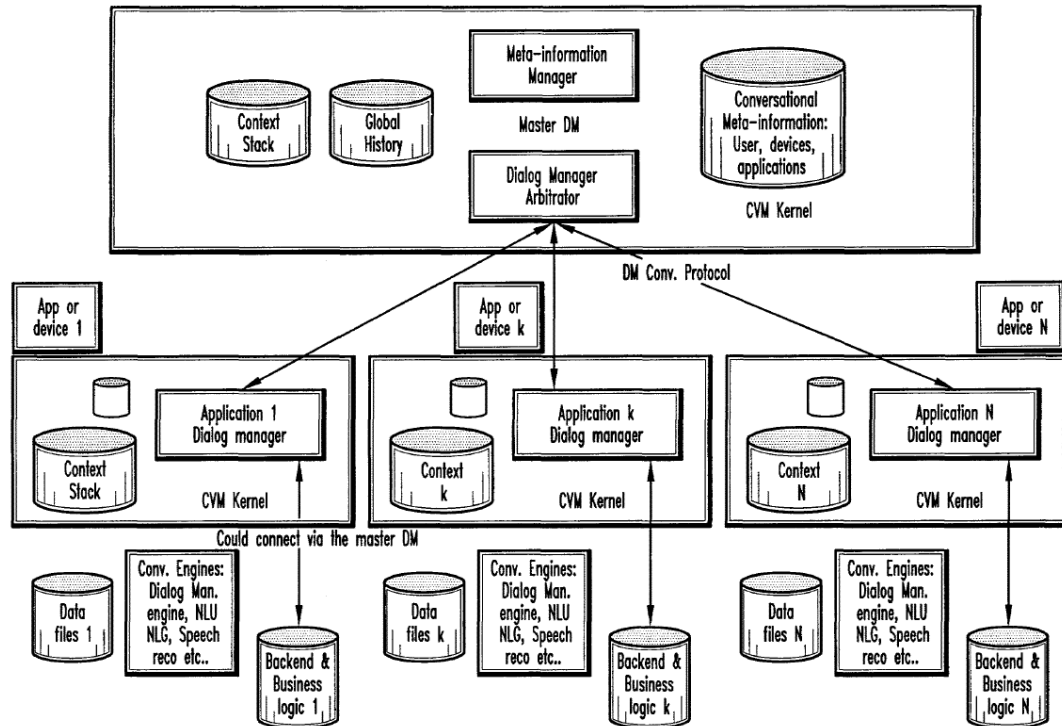


Coffman, Figure 1.

Figure 1 of Coffman is a block diagram of a conversational computing system. *See id.* at 9:27–28, 12:5–7. The system includes CUI 10, conversationally-aware applications 11 with APIs 13 to operate with CVM 14, conventional applications 12 managed by CVM 14 without APIs, and other components. *See id.* at 12:7–10, 12:26–13:13.

Figure 10 of Coffman is reproduced below.

FIG. 10



Coffman, Figure 10.

Figure 10 is a diagram of a dialog management system and process, including “protocols for information exchange between dialog managers (DMs) . . . of networked devices” wherein “different devices, CVMs or different applications will have their own dialog manager [and] context stack.” *Id.* at Figs. 9–10, 10:13–16, 26:28–27:2, 27:23–24. In particular, Figure 10 “illustrates a hierarchical dialog between multiple dialog managers (i.e, the master arbitrator, and the slave dialog managers 1, k, and N) of various devices / applications (1, k and N)” that follows “a typical master slave topology . . . formed by exchanging the relevant information via the DM protocols.” *Id.* at 27:24–28. The master dialog manager is shown at the top of Figure 10 (“Master DM”), and three slave

dialog managers are shown across the middle of Figure 10 (“Dialog manager[s]”). *See id.* at 27:2–5.

The master DM is “responsible for managing the flow of I/O to the different [slave] managers to decide the active dialog and appropriately execute a query and update the context and history.” *Id.* at 27:5–7. Various types of information can be exchanged between the master and slave devices, including “context” information. *Id.* at 27:7–20.

2. *Claim 1 of the '652 Patent*

Petitioner provides arguments and evidence in support of contending claim 1 is unpatentable as having been obvious over Coffman, including supporting testimony from Mr. Schmandt. *See* Pet. 19–37; Ex. 1002 ¶¶ 41–46, 56–80. Patent Owner and Mr. Occhiogrosso disagree. *See* PO Resp. 1–9, 11–12, 14–19, 29–36; Ex. 2013 ¶¶ 51–77. For the following reasons, we conclude Petitioner has demonstrated claim 1 is unpatentable by a preponderance of the evidence.

(a) *Petitioner’s Unopposed Contentions*

Petitioner contends and Mr. Schmandt testifies that Coffman discloses, as recited in the preamble of claim 1 if it is limiting, a system for processing natural language utterances where recognized words of the utterances alone are insufficient to completely determine one or more commands or requests. *See* Pet. 19–24 (citing Ex. 1003, Fig. 6, 5:1–4, 6:6–8, 6:32–7:3, 11:23–24, 13:8–11, 29:20–32, 36:19–21, 36:29–31, 37:14–32, 41:2–5, 42:1–5, 48:1–3, 53:8–10, 53:22–26, 56:25–57:2); Ex. 1002 ¶¶ 57–62.

Petitioner also contends and Mr. Schmandt testifies that Coffman’s system includes, as required by claim 1, one or more physical processors programmed with instructions to cause the processors to take certain actions. *See* Pet. 24–25 (citing Ex. 1003, 13:1–3); Ex. 1002 ¶ 63. Petitioner and Mr. Schmandt alternatively assert this claim requirement “describes nothing more than routine and conventional components and operation of a generic computer system,” which a person of ordinary skill in the art would have found obvious to utilize in Coffman’s system. Pet. 24–25, 29–30, 33, 35, 37 (citing Ex. 1009, 16, 24–25, 28, 33); Ex. 1002 ¶¶ 63, 68, 75, 77, 80.

Further according to Petitioner and Mr. Schmandt, the actions taken by Coffman’s processors include, as recited in claim 1, “generat[ing] a first context stack associated with a first device, the first context stack comprising context information that corresponds to a plurality of prior utterances.” Ex. 1001, 38:46–49; *see* Pet. 25–29 & n.3 (citing Ex. 1003, Figs. 3 and 10, 6:8–10, 6:26–28, 6:32–7:3, 22:32–23:3, 27:1–7, 27:24–28, 28:24–27, 29:18–30:10, 37:20–26, 41:2–5); Ex. 1002 ¶¶ 64–67.

Petitioner and Mr. Schmandt further conclude the actions taken by Coffman’s processors include, as recited in claim 1: “receiv[ing] a natural language utterance associated with a command or request”; “determin[ing] one or more words of the natural language utterance by performing speech recognition on the natural language utterance”; and “determin[ing] the command or request based on the one or more words and . . . context information.” Ex. 1001, 38:55–61; *see* Pet. 32–37 (citing Ex. 1003, Figs. 4 and 6, 6:32–7:3, 10:28–30, 13:8–13, 29:1–4, 30:8–10, 37:14–26, 38:9–11, 38:26–27, 40:12–15, 41:2–5, 42:1–5, 48:1–3, 51:12–14, 56:25–57:2); Ex. 1002 ¶¶ 73–80.

We conclude the foregoing contentions, which Patent Owner does not oppose, are supported by the Coffman disclosures and other evidence cited by Petitioner and Mr. Schmandt.

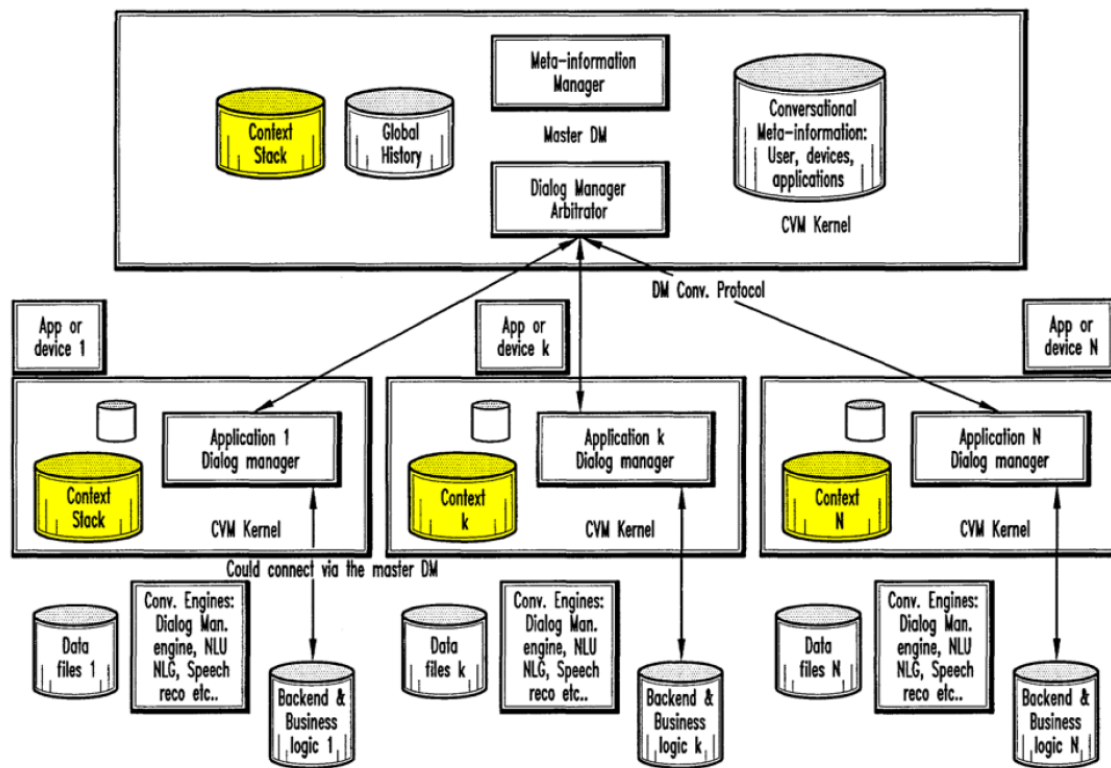
(b) Petitioner's Opposed Contentions

Patent Owner opposes Petitioner's reliance on Coffman as disclosing, per claim 1, *synchronizing* a first *context stack* associated with a first device with a second *context stack* associated with a second device, such that the context information of the first context stack is updated based on related context information of the second context stack. *See* Ex. 1001, 38:50–54; Pet. 30–32; PO Resp. 1–9, 11–12, 14–19, 29–36; Ex. 1002 ¶¶ 69–72; Ex. 2013 ¶¶ 51–77.

We summarize the parties' competing contentions in this regard, then we explain why we conclude Petitioner's contentions are supported by a preponderance of the evidence.

(i) The Petition

Petitioner contends Coffman's system uses “‘**context stack[s]**’ where the state and history of each active task / process, such as an application process, is tracked in its own ‘context’ . . . derived from a user's spoken input.” Pet. 25–27 & n.3 (citing Ex. 1003, Fig. 3, 6:8–10, 6:26–28, 6:32–7:3, 22:32–23:3, 29:18–30:10, 37:20–26, 41:2–5); Ex. 1002 ¶¶ 64–66 & n.4. To support this contention, Petitioner annotates Coffman's Figure 10, as reproduced below (on the next page):



Coffman, Figure 10 (As Annotated at Pet. 28).

Figure 10 is a diagram of a dialog management system and process, as summarized above in Section III.D.1. Petitioner has annotated (in yellow) the “Context Stack” maintained by CVM 14 with a master dialog manager or “Master DM,” as well as the “Context Stack[s]” respectively maintained by three “[D]evice[s]” 1, k, and N with respective slave “Dialog manager[s].” Ex. 1003, Fig. 10, 26:28–27:28; *see also* Pet. 27–29 (“Coffman further explains that the devices in its conversational system ‘typically . . . will have their own . . . context stack 217, 217a’ — i.e., each device has a ‘**context stack associated with [that] device**’ [per claim 1].”) (citing Ex. 1003, Fig. 10, 27:1–7, 27:24–26); *id.* at 29 (arguing Coffman’s “context stack operates and is managed under control of dialog manager / controller”) (citing Ex. 1003, 28:24–27, 29:18–20); Ex. 1002 ¶¶ 66–67.

Petitioner asserts the “first” device and context stack of claim 1 correspond to Coffman’s master device (at the top of Figure 10 above), and the “second” device and context stack of claim 1 correspond to one of Coffman’s slave devices (across the middle of Figure 10 above). *See* Pet. 29 (citing Ex. 1003, 27:5–7, 27:26–28, 28:24–27, 29:18–20); *id.* at 30 (citing Ex. 1003, Figs. 3 and 10, 27:1–7, 27:24–26); Ex. 1002 ¶¶ 67, 69.

Petitioner next asserts Coffman makes clear that the master device’s context stack “is ‘**synchronized**’ with a slave device’s context stack” per claim 1. Pet. 30–31 (citing Ex. 1003, 25:10–14, 27:7–20); Ex. 1002 ¶ 70. Petitioner argues a person of ordinary skill in the art “would have further understood and found it obvious that, as a result of synchronizing device context stacks in Coffman’s system, **‘the context information of the first context stack is updated based on related context information of the second context stack.’**” Pet. 31; Ex. 1002 ¶ 71. Petitioner asserts “[t]he context information in the two context stacks would be ‘**related**’” per claim 1 because: (1) Coffman explains the context information “can comprise ‘**context updates**,’ which would update previous, related context information in the device’s context stack”; and (2) “the associated devices themselves are related as a result of the master slave topology relationship.” Pet. 31; Ex. 1002 ¶ 71. Further according to Petitioner: “The context information of the first stack would be ‘**updated**’ based on the context information of the second context stack both by adding information to the first context stack and by updating existing information in the first context stack.” Pet. 31–32; Ex. 1002 ¶ 71.

Petitioner and Mr. Schmandt finally conclude Coffman discloses or alternatively suggests utilizing physical processors to synchronize the first

context stack with the second context stack, substantially as discussed above in Section III.D.2(a) in relation to Petitioner’s unopposed contentions. *See* Pet. 32 (“for substantially the same reasons as those explained for 1[a][i]”); Ex. 1002 ¶ 72.

(ii) *Claim Construction — Synchronize a First Context Stack with a Second Context Stack to Update Related Context Information*

Claim 1 recites one or more processors programmed with instructions to “*synchronize the first context stack with a second context stack associated with a second device such that the context information of the first context stack is updated based on related context information of the second context stack.*” Ex. 1001, 38:43–45, 38:50–54 (emphases added).

Patent Owner argues Coffman does not disclose the claimed *synchronizing of context stacks* because Coffman merely “*add[s]* or *update[s]* context in” the master context stack based on a slave context stack. PO Resp. 1–2. In Patent Owner’s view, such adding or updating “is not sufficient to establish that two stacks are synchronized, as each stack would possess a number of varying attributes including the contents, size, and order of the stacks.” *Id.* at 2. Patent Owner asserts the ’652 patent specification indicates that “[t]he synchronization of contexts . . . preserves the ordered nature of the context stack or the set of context entries (or ‘contexts’) in the context stack.” *Id.* at 7 (citing Ex. 1001, 20:10–12, 39:23–27 (claim 5)). Patent Owner further asserts the ’652 patent “seeks to determine ‘which version of a data element . . . or an agent are the newest or most up-to-date and propagates that element’ . . . and which can entail ‘complete replacement,’ rather than a series of incremental updates.” *Id.* at 29–30 (citing Ex. 1001, 17:26–32). The ’652 patent’s invention further

“synchronizes a context stack across different devices” according to Patent Owner. *Id.* at 30 (citing Ex. 1001, 4:29–32, 19:41–44).

Patent Owner also asserts that the reason for the Examiner’s allowance of the ’652 patent focused on synchronizing context *stacks* rather than synchronizing context more generally. *Id.* at 2–3 (citing Ex. 1012, 447); *see also id.* at 11–12 (citing Ex. 1012, 127, 170–171, 186, 191, 194, 198, 446–447). Patent Owner notes that the Examiner considered “Coffman’s U.S. patent counterpart” prior to allowing the ’652 patent. *Id.* at 12, 15 (citing Pet. 14–15; Ex. 1012, 175, 216, 224); *see* Pet. 14–15 (identifying Coffman’s U.S. counterparts). According to Patent Owner: “[t]he distinction expressly emphasized in the Examiner’s reason for allowance — that even synchronizing a context entry does not show synchronization of context stacks [in independent claim 1] or synchronization of context sets [in independent claim 20] — is a distinction between Coffman and the challenged claims as well.” PO Resp. 15, 35 (citing Ex. 1012, 447). Patent Owner concludes that although “Coffman may add context to a context stack or update context in a context stack,” that “does not establish that Coffman discloses **synchronization** of a context stack or context set, *which also requires synchronization of order of the context stack or context set.*” *Id.* at 15 (emphasis added in part).

Patent Owner further informs the Board that the U.S. District Court for the Eastern District of Virginia, in *Dialect, LLC v. Amazon.com, Inc. and Amazon Web Services, Inc.*, Case No. 1:23-cv-00581, construed several claim terms in Patent Owner’s patents. *See* PO Resp. 12–14 & n.1 (citing Exs. 2002–2007). Those patents did not include the ’652 patent, but they did include a continuation parent of the ’652 patent (U.S. Patent

No. 8,195,468 B2) and a continuation child of the '652 patent (U.S. Patent No. 9,495,957 B2). *See id.* at 11 n.1; Ex. 1001, code (60); Ex. 2006, code (60). The District Court construed the claim term “context stack” in both of those related patents, and concluded it refers to “a data structure that stores or references contexts in a manner *that can be ordered*.” PO Resp. 13 (emphasis added) (quoting Ex. 2007, 1).

Patent Owner moreover asserts both expert witnesses, Petitioner’s Mr. Schmandt and Patent Owner’s Mr. Occhiogrosso, agree with Patent Owner’s understanding of the claim term “synchronize” as requiring an *identical order* of entries in the two stacks. *See* PO Resp. 2, 7–8, 33–34 (citing Ex. 2011, 13:10–17; Ex. 2013 ¶¶ 39, 73). Patent Owner argues “Mr. Schmandt’s testimony confirms that for two stacks to be synchronized, it is not enough that they contain the same entries; they also must contain those entries in an identical order because order is an attribute of a stack.” *Id.* at 33–34 (citing Ex. 2011, 13:10–17, 14:13–16, 14:25–15:6).

Mr. Occhiogrosso also “explains that a [person of ordinary skill in the art] would not interpret two stacks containing different context entries, or even equivalent context entries but in different orders, as being synchronized.” *Id.* at 34 (citing Ex. 2013 ¶¶ 70, 73–74). Mr. Occhiogrosso testifies further that “the activities of ‘addition’ to a context stack or updating existing information for a context entry in [the] context stack” based on another context stack do not “synchronize” the two stacks per claim 1, because “a stack is an ordered list.” *Id.* at 34–35 (citing Ex. 2013 ¶¶ 70, 73–74).

In reply, Petitioner characterizes Patent Owner’s claim construction as requiring “perfect synchronization” which the Board “correctly rejected in the Institution Decision.” Pet. Reply 1, 5 (citing PO Resp. 33–34;

Inst. Dec. 25–27). Petitioner adds that “[t]he claim language itself confirms that perfect synchronization is not required” in “stat[ing] that context stacks are ‘synchronized’ **‘such that the context information of the first context stack is updated based on related context information of the second context stack.’”** *Id.* at 1, 5–6 (citing Ex. 1001, 38:46–54); Ex. 1017 ¶ 10. Claim 1 “does not state, for example, that synchronizing causes the first context stack’s content and ordering to be identical with the second context stack.” Pet. Reply 5; Ex. 1017 ¶ 10. According to Petitioner, this understanding is consistent with the ’652 patent specification. *See* Pet. Reply 6 (citing Ex. 1001, 17:25–32, 19:36–44 (describing Fig. 7); Inst. Dec. 25); Ex. 1017 ¶ 11. Petitioner also asserts Patent Owner mischaracterizes the prosecution history of the ’652 patent. *See* Pet. Reply 2–3 (citing Pet. 15; PO Resp. 11–12, 15–16, 35; Inst. Dec. 37–39; Ex. 1012, 447).

Patent Owner responds that its construction should be adopted because it “gives full credit to every claimed feature, including the claimed ‘synchronize’ requirement,” whereas Petitioner’s construction “completely omit[s]” the term “synchronize.” PO Sur-reply 1–2; *see also id.* at 10–11 (“Petitioner elevates [the ‘update’] phrase above the preceding clause that requires that the stacks are synchronized, and as a result interprets the claim such that any update to a context stack would satisfy the entire claimed phrase.”); Tr. 25:17–27:24 (argument by Patent Owner’s counsel).

Patent Owner asserts “Petitioner does not dispute that a stack possesses many attributes including the contents, size, and order of the stacks” and “Petitioner fails to show that even one of these attributes would be synchronized between two stacks according to Coffman.” *Id.* at 2.

Patent Owner also asserts the post-Reply deposition of Petitioner’s expert Mr. Schmandt “confirmed additional defects in the Petition’s theories related to claim construction.” *Id.* Patent Owner contends Mr. Schmandt testified that whether two context stacks are synchronized is a “*nuanced*” inquiry depending on “what the application is [omitted] and what the content is,” and “is in the eye of the beholder.” *Id.* at 2–3 (citing Ex. 2014, 26:14–25, 28:2–9, 38:9–42:2; Ex. 2015). Mr. Schmandt also allegedly “spent nearly four pages of deposition transcript discussing factors that would be relevant to how a [person of ordinary skill in the art] would understand the term, ‘synchronizing,’” none of which was reflected in Mr. Schmandt’s declaration testimony. *Id.* at 3 (citing Ex. 2014, 38:9–42:2). Based on these contentions, Patent Owner concludes the Petition failed to satisfy the requirement to “identify . . . [h]ow the challenged claim is to be construed” (37 C.F.R. § 42.104(b)(3)), because it did not address the “many factors” identified during Mr. Schmandt’s deposition. PO Sur-reply 3 (citing Pet. 18); *id.* at 4–7 (same).

Patent Owner further argues Mr. Schmandt testified that “synchronization may be required to be perfect,” thereby acknowledging “synchronization of two stacks as claimed requires more than just adding an entry to one stack.” *Id.* at 6 (citing Ex. 2014, 38:17–18).

Patent Owner moreover asserts Petitioner fails to recognize the importance of the Examiner’s stated reasons for allowing the ’652 patent, allegedly indicating that “synchronizing a context *stack/set* across devices is a narrower limitation than just synchronizing a context across devices.” *Id.* at 7 (emphasis added) (citing Ex. 1012, 171, 447).

We begin our claim construction with the claim language. Claim 1 recites one or more processors programmed with instructions to “synchronize the first context stack with a second context stack . . . such that the context information of the first context stack is updated based on related context information of the second context stack.” Ex. 1001, 38:43–45, 38:50–54. This language focuses on synchronizing the two context stacks by *updating the context information* of the first stack based on *related context information* of the second stack. *See id.*

The only potential textual support in claim 1 for Patent Owner’s position that the claim requires duplicating the *order* of the context entries in the two stacks are the terms “synchronize” and “context stack.” *See id.* We discern nothing in the plain and ordinary meaning of those terms that would require duplicating the order of context entries in the two stacks as part of the claimed synchronizing.

Patent Owner cites claim 5 as supporting its claim construction. *See* PO Resp. 7. Claim 5 depends indirectly from claim 1 to add “wherein *the plurality of context entries are ordered* in the first context stack” and “*updat[ing] the ordering of the plurality of context entries* in the first context stack.” Ex. 1001, 39:23–27 (emphasis added). We conclude that claim differentiation suggests claim 1 is broader than that, so it does not require duplicating the order of context entries in the two stacks. *See* 35 U.S.C. § 112(d) (requiring a dependent claim to “specify a further limitation of the subject matter claimed”); *Phillips v. AWH Corp.*, 415 F.3d 1303, 1314–15 (Fed. Cir. 2005) (en banc) (“[T]he presence of a dependent claim that adds a particular limitation gives rise to a presumption that the limitation in question is not present in the independent claim.”).

Patent Owner’s claim construction that synchronizing context stacks must include duplicating the order of entries in the stacks also lacks support in the ’652 patent specification. The only passage cited by Patent Owner that mentions the order of entries states: “Agents 106 may update a context stack, that includes an ordered list of command contexts, to enable follow-up requests.” Ex. 1001, 20:10–12; *see also id.* at 4:29–32, 17:25–32, 19:41–44 (the other passages cited by Patent Owner). That disclosure, however, is permissive (“may update”) rather than mandatory or limiting, and does not include the “synchronization” term upon which Patent Owner’s limiting claim construction relies. Indeed, the ’652 patent specification’s usage of the term “synchronize” focuses on synchronizing *data* such as context without referring to the *order* of the data. *See id.* at 4:13–16 (“the context”), 14:32–34 (“[u]ser specific data”), 17:15–32 (“data and agents”), 19:41–44 (“the context”), 26:66–27:1 (“the context”).

Turning to the ’652 patent prosecution history, in the record here as Exhibit 1012, the applicant presented application claims 50 and 63 as follows with our emphases added.

50. A system for processing natural language utterances where recognized words of the natural language utterances alone are insufficient to completely determine one or more commands or requests, the system including one or more processors executing one or more computer program modules which, when executed, cause the one or more physical processors to:
generate a context stack comprising context information that corresponds to a plurality of prior utterances;
receive a natural language utterance associated with a command or request;

determine one or more words of the natural language utterance by performing speech recognition on the natural language utterance; and

determine the command or request based on the one or more words and the context information.

63. The system of claim 50, wherein *the context stack is a first context stack of a first device*, and wherein the one or more processors are further caused to:

synchronize the context stack with a second context stack of a second device.

Ex. 1012, 123, 127 (emphases added).

The Examiner rejected claim 50 under 35 U.S.C. § 103 as having been obvious over Ross³. *See id.* at 156–160. The Examiner objected to claim 63 as dependent upon the rejected claim 50, but concluded claim 63 would be allowable if rewritten in independent form. *See id.* at 170. The Examiner included a statement of reasons for allowance of claim 63, as follows:

As described in the Notice of Allowability mailed 11/1/10 in Application 12/617506[⁴], Mumick et al. (US 2003/0046346) teaches synchronizing plural browsers on a plurality of mobile devices including a state manager that stores navigation histories in a data store so that a user can switch from a visual browser on one device to a voice browser on a second device without losing navigation state, but synchronization of navigation state is not the claimed synchronization of context across multiple devices (page 2).

While the claim language at issue is not identical, the limitation in claim 63 is directed to a similar concept which is also more narrow in that not only is context synchronized, but context stacks of distinct devices are synchronized.

³ U.S. Patent App. Pub. No. 2002/0133354 A1.

⁴ Nothing from U.S. Patent App. No. 12/617,506 has been entered into the record of this proceeding.

Id. at 171.

The applicant then amended claim 50 and the other independent claims to incorporate the allowable subject matter of claim 63. *See id.* at 186 (independent claim 50), 191 (independent claim 64), 194 (independent claim 70), 198.

The Examiner then allowed the '652 patent. *See id.* at 441–449 (Notice of Allowance). The Examiner provided another statement of reasons for allowance, identical to the first statement (reproduced above) except that the reference in the final sentence to “stacks” was changed to “stacks / sets” to account for the addition of new independent claim 70. *Id.* at 447. The '652 patent thereafter issued, with independent application claims 50, 64, and 70 being re-numbered respectively as claims 1, 14, and 20 in the patent. *See id.* at 480.

Based on the foregoing summary of the '652 patent prosecution history, we find the Examiner provided reasons for allowance over Ross and Mumick, but not over Coffman or any counterpart of Coffman. Also, we find the Examiner cited Mumick's synchronization of *navigation histories* as being different from the claimed synchronization of *context stacks* (claim 1) or *context sets* (claim 20). *See id.* at 170–171 (“synchronization of navigation state is not the claimed synchronization of context across multiple devices”), 447 (same). Coffman, by contrast, specifically discloses *context stacks / sets*. *See supra* Section III.D.2(a) (discussing Petitioner's unopposed contentions).

Patent Owner cites the Examiner's additional statement that “[w]hile the claim language at issue is not identical” (apparently comparing U.S. Patent App. No. 12/617,506 and the '652 patent claims), “the limitation in

claim 63 [‘synchronize the context stack with a second context stack of a second device’] is directed to a similar concept which is also more narrow in that not only is context synchronized, but context stacks / sets of distinct devices are synchronized.” Ex. 1012, 171, 447. That statement does, as Patent Owner argues, appear to differentiate between synchronizing “context” generally and synchronizing “context stacks” or “context sets” specifically. *See id.* Nonetheless, the Examiner did not articulate the exact property of a “context stack” or “set” that might be important in this regard, much less indicate that the order of the context entries is important. *See id.* Moreover, the Examiner cited only Mumick as the prior art that was overcome by the ’652 patent claims, reducing the relevancy of the Examiner’s claim comparison to Coffman as at issue here. *See id.* Thus, we conclude Patent Owner’s reliance on the ’652 patent prosecution history is misplaced.

Next, the U.S. District Court for the Eastern District of Virginia construed the claim term “context stack” in patents related to the ’652 patent as referring to “a data structure that stores or references contexts in a manner *that can be ordered.*” Ex. 2007, 1 (emphasis added). As explained in an accompanying Memorandum Opinion (Exhibit 3002), that construction adopted Patent Owner (Dialect)’s position that a “context stack” *is not limited* to an *ordered list*, and rejected Amazon’s contrary position. *See* Ex. 3002, 20–24; Ex. 2007, 1 (referring to “the reasons set forth in the accompanying Memorandum Opinion”). In other words, Patent Owner’s position in the present IPR that order is an attribute of a context stack and so must be part of synchronizing two context stacks (*see* PO Resp. 2, 7–8) is the direct opposite of Patent Owner’s successful position in its District Court

litigation against Amazon.⁵ Our present Decision that synchronizing context stacks does not require duplicating the order of entries in the stacks is consistent with, and is supported by, the District Court’s claim construction of the term “context stack.”

In particular, the District Court considered essentially the same intrinsic evidence that is at issue here, and reached essentially the same conclusions we have reached above. That is, the District Court concluded the specification disclosures cited by Patent Owner here do not support the position that a context stack must be limited to an ordered list. *See* Ex. 3002, 21 (citing Ex. 2004, 20:10–11; Ex. 2006, 20:44–46); Ex. 1001, 20:10–12 (the corresponding ’652 patent disclosure). The District Court also relied on claim differentiation, in a similar fashion to our reliance on dependent claim 5 in the ’652 patent. *See id.* at 21–22 & n.20 (citing Ex. 2006, 40:9–10). The District Court further considered extrinsic evidence proffered by Amazon “to demonstrate that a ‘stack’ must have a particular kind of internal structure,” but concluded other extrinsic evidence proffered by Patent Owner “rebut[ted] that contention.” *Id.* at 22–24 & n.21. Thus, our conclusion that synchronizing context stacks does not require duplicating the order of entries in the context stacks is supported by the District Court’s claim construction analysis.

Patent Owner suggests the District Court’s claim construction does not apply to the ’652 patent or to Patent Owner’s arguments in this IPR,

⁵ We are troubled by Patent Owner’s submission of the District Court’s Order Construing Claim Terms (Exhibit 2007) without the Memorandum Opinion in support (Exhibit 3002). *See* Paper 30; 37 C.F.R. § 42.11(a) (“Parties and individuals involved in the proceeding have a duty of candor and good faith to the Office during the course of a proceeding.”).

because the claims construed by the District Court recited only “context stack” and did not include the term “synchronize.” *See* Tr. 33:20–34:18. We do not agree. As discussed above, Patent Owner’s position in this IPR is that *order is an attribute of a context stack* so order must be part of synchronizing two context stacks. *See* PO Resp. 2, 7–8. That is the direct opposite of Patent Owner’s successful position in its District Court litigation against Amazon that a context stack need not be ordered. *See, e.g.,* Tr. 33:20–34:13.

We have also considered the expert testimony. Patent Owner’s expert Mr. Occhiogrosso opines that synchronizing context stacks must include duplicating the order of entries in the stacks. *See* Ex. 2013 ¶¶ 36–41, 50, 67–74. However, the only documentary evidence cited to support this opinion is the ’652 patent specification disclosures we have addressed above. *See id.* ¶¶ 39, 70, 72 (citing Ex. 1001, 20:10–12, 20:19–22). Thus, Mr. Occhiogrosso’s extrinsic testimony on this point is overly conclusory, and we accordingly give it little weight here. Moreover, although extrinsic evidence such as expert witness testimony “can shed useful light on the relevant art,” it is “less significant than the intrinsic record in determining ‘the legally operative meaning of claim language.’” *Phillips*, 415 F.3d at 1317 (citations omitted); *see also id.* at 1318 (“We have viewed extrinsic evidence in general as less reliable than the patent . . . in determining how to read claim terms.”); *Seabed Geosolutions (US) Inc. v. Magseis FF LLC*, 8 F.4th 1285, 1287 (Fed. Cir. 2021) (“Given the clarity of the intrinsic evidence, resort to extrinsic evidence is unnecessary” so “to the extent the Board relied on extrinsic evidence to alter the meaning of ‘fixed’ that is clear from the intrinsic evidence, that was error.”).

Petitioner's expert Mr. Schmandt agreed during cross examination that two context stacks are not "identical" if they have "the same content but in different orders," "because order is a -- order is an attribute of stack." Ex. 2011, 13:10–17, 14:13–15:6; *see* PO Resp. 2, 33–34. Also, when asked whether a first context stack with entries "ABCD" and a second context stack with entries "DCAB" would be considered to be synchronized, he stated "[t]hat would depend on how the POSITA understood synchronized, which is nuanced by what the application is (inaudible) and what the context is." Ex. 2014, 26:14–27:21, 28:10–15; *see* PO Sur-reply 2–3, 5.

Mr. Schmandt explained that "whether or not order is a requirement for synchronization depends on what the stacks are being used for" and "a stack with different ordering is not identical, but that doesn't mean it is not synchronized and depending on what's going on with the way in which the stack is being used, synchronization may or may not require -- be required to be absolutely perfect." Ex. 2014, 27:22–28:9; Ex. 2015 (errata at 28:5).

Later, immediately following a lunch break, Patent Owner's counsel asked Mr. Schmandt whether he could explain the "nuance in the term synchronizing." Ex. 2014, 37:4–38:8. Mr. Schmandt responded that "synchronizing is in the eye of the beholder," and provided a lengthy discussion of synchronizing data. *Id.* at 38:9–42:2; *see* PO Sur-reply 2–3, 5–6. He concluded that "order of a context stack could be important" but he "would not say it has to be" important. Ex. 2014, 42:3–11.

In short, Mr. Schmandt's testimony is that synchronization of two context stacks may include duplicating the order of entries in the stacks in some scenarios, but does not need to do so in all scenarios. That is consistent with the District Court's claim construction that context stacks

“*can* be ordered” (Ex. 2007, 1 (emphasis added)) as well as our conclusion here that synchronizing context stacks does not require duplicating the order of entries in the stacks. Thus, we disagree with Patent Owner’s suggestion that the expert testimony proffered here should lead to a different claim construction from the District Court’s. *See* Tr. 34:19–35:19. Further, as noted above, extrinsic evidence such as expert witness testimony is less significant than intrinsic evidence when construing claims. *See Phillips*, 415 F.3d at 1317, 1318; *Seabed Geosolutions*, 8 F.4th at 1287.

Finally, we disagree with Patent Owner’s contention that the Petition fails to comply with 37 C.F.R. § 42.104(b)(3). The claim construction issues addressed above arose due to arguments in the Patent Owner Response seeking to distinguish claim 1 from Coffman, opening the door for Petitioner to respond to those arguments.

For the foregoing reasons, we conclude that synchronizing context stacks per claim 1 does not require duplicating the order of entries in the stacks.

(iii) *Whether Coffman Discloses or Renders Obvious Synchronizing a First Context Stack with a Second Context Stack to Update Related Context Information*

Patent Owner argues Coffman does not synchronize context stacks per claim 1. *See* PO Resp. 29–36. Patent Owner concedes “Coffman provides a system with context stacks and allows for additions and updates to those context stacks” per Petitioner’s contentions. *Id.* at 29–30, 32–33 (citing Pet. 30–31; Ex. 1002 ¶ 70; Ex. 1003, 25:10–14, 27:7–20). Patent Owner nonetheless contends Coffman does not “synchronize” context stacks,

applying Patent Owner’s claim construction of claim 1 discussed above in Section III.D.2(b)(ii). *See id.* at 30, 32–33, 35; Ex. 2013 ¶¶ 55–64, 67–77.

Patent Owner further argues and Mr. Occhiogrosso testifies that Coffman lists “items of information that *can be — not must be —* exchanged between the different dialog managers.” PO Resp. 30–31 (citing Ex. 1003, 27:7–22); Ex. 2013 ¶ 65. Patent Owner and Mr. Occhiogrosso assert “Coffman even allows the individual dialog managers the choice to act *or not act* on received information when managing their own context stack,” so “different devices, CVMs or different applications ***will have their own . . . context stack 217, 217a.***” PO Resp. 31, 35 (citing Ex. 1003, 27:1–2, 27:12–13 (item 5), 27:16–18 (item 11), 27:19–20 (item 15)); Ex. 2013 ¶¶ 66, 75. Patent Owner and Mr. Occhiogrosso conclude Coffman is “inconsistent” with synchronizing context stacks per claim 1, because Coffman’s process “does not even require any synchronization of individual context items, much less synchronization of the context stack.” PO Resp. 3, 31, 35 (citing Ex. 1003, 27:1–2, 27:16–20 (items 12 & 14)); Ex. 2013 ¶¶ 58, 63–77.

Petitioner replies that Coffman, “just like” claim 1, synchronizes two context stacks by updating the context information of one stack based on related context information of the other stack. Pet. Reply 1 (citing Ex. 1003, 25:10–14). Petitioner asserts “Patent Owner’s argument that Coffman fails to disclose or render obvious the claimed ‘**synchroniz[ing]**’ is nearly identical to” arguments the Board “correctly rejected in the Institution Decision.” *Id.* at 4–5 (citing Paper 9 (Patent Owner’s Preliminary Response), 20–23; PO Resp. 32–36; Inst. Dec. 26–27).

Petitioner also addresses Patent Owner’s argument “that Coffman does not disclose ‘synchronization’ because Coffman allows for permissive, not mandatory, context stack updates.” *Id.* at 4 (citing PO Resp. 30–31). Petitioner asserts we should reject that argument for reasons already articulated in the Institution Decision. *See id.* at 8 (citing PO Resp. 35; Inst. Dec. 26–27; Ex. 1001, 17:15–25, 38:50–54); Ex. 1017 ¶ 12. According to Petitioner, Patent Owner’s expert Mr. Occhiogrosso “agreed at his deposition.” Pet. Reply 8 (citing Ex. 1018, 25:20–22).

Petitioner adds: “Even if devices in Coffman’s system only sometimes [choose] to act in response to exchanging information between devices, that is enough to disclose the claimed ‘synchronizing,’” and “Patent Owner appears to agree that synchronization can include replacing ‘a portion of a database and can be performed ‘automatically’ or ‘on demand.’” Pet. Reply 8–9 (citing *Hewlett-Packard Co. v. Mustek Sys., Inc.*, 340 F.3d 1314, 1326 (Fed. Cir. 2003); PO Resp. 7). Accordingly, Petitioner concludes “Coffman’s disclosures of adding and updating a device’s context stack with context information from another device’s context stack discloses the claimed ‘**synchroniz**[ing]’ step.” *Id.* at 9–10 (citing Pet. 28–31; Inst. Dec. 20–25; Ex. 1003, 25:10–14).

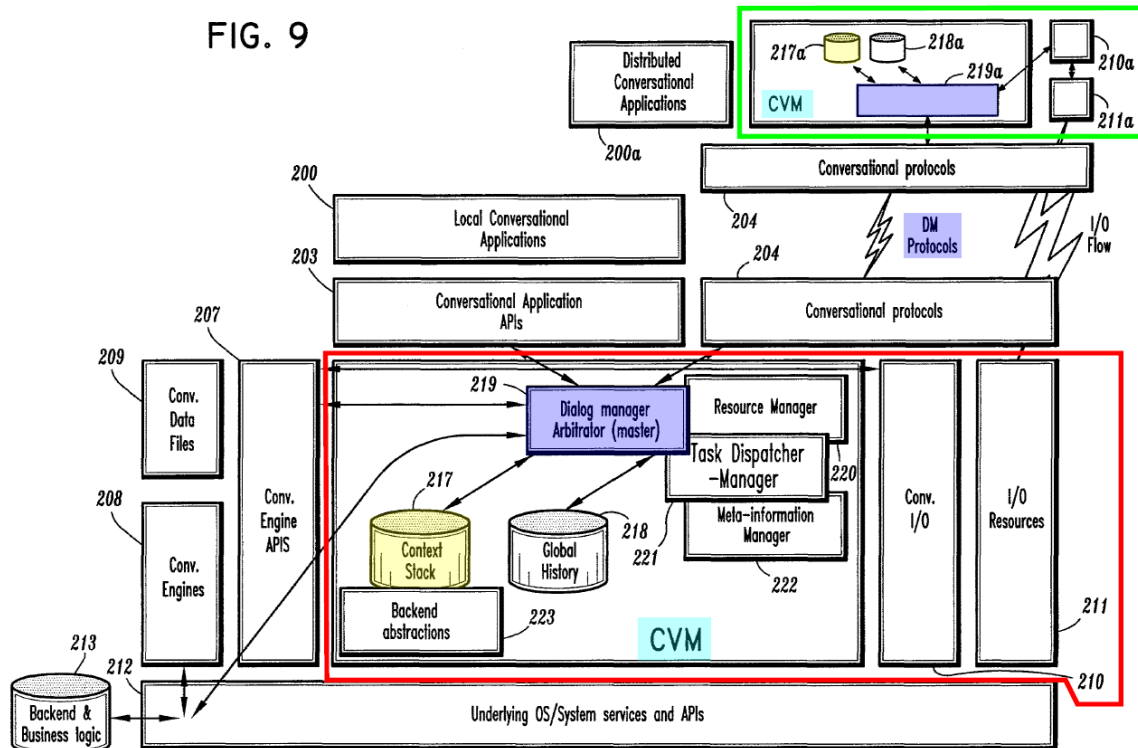
Petitioner alternatively contends, “even if [claim 1] did require ‘perfect’ synchronization,” then “a person of ordinary skill in the art would have understood and found it obvious that Coffman’s synchronization could at least sometimes be ‘perfect’ — which is enough to meet the claim.” *Id.* at 7–8 (citing Ex. 1003, 26:1–3, 27:5–7; *Hewlett-Packard*, 340 F.3d at 1326); Ex. 1017 ¶ 11.

Patent Owner replies largely by reiterating positions we have summarized above. *See* PO Sur-reply 7–12. Patent Owner also disputes Petitioner’s alternative obviousness argument, proffered in the event claim 1 is construed to require perfect synchronization. *See id.* at 11 (lines 8–18).

We conclude a preponderance of the evidence supports Petitioner’s contention that Coffman discloses synchronizing a first context stack associated with a first device (i.e., Coffman’s master device) with a second context stack associated with a second device (i.e., one of Coffman’s slave devices), such that the context information of the first stack is updated based on related context information of the second stack.

We begin with Coffman’s Figure 3, which illustrates distributed conversational protocols 300 that “allow, in general, remote applications and resources [to] register their conversational capabilities and proxies” and “exchange information [such as] *context* with each other” including “notification of state and *context changes*.” Ex. 1003, 22:24–23:15 (emphasis added), Fig. 3 (illustrating conversational protocols 300 intermediating between a local client device (on the left) and a server device (on the right)); *see* Pet. 26–27, 30 (citing Ex. 1003, 22:32–23:3, Fig. 3). Conversational protocols 300 “include Dialog Manager (DM) protocols” and “allow the exchange of information to coordinate the conversation involving multiple devices or applications including [a] master/[slave] conversational network” such as shown in Figures 9 and 10. Ex. 1003, 23:5–9, 26:28–30, 27:23–28.

Figure 9 of Coffman is reproduced below, with our color annotations.



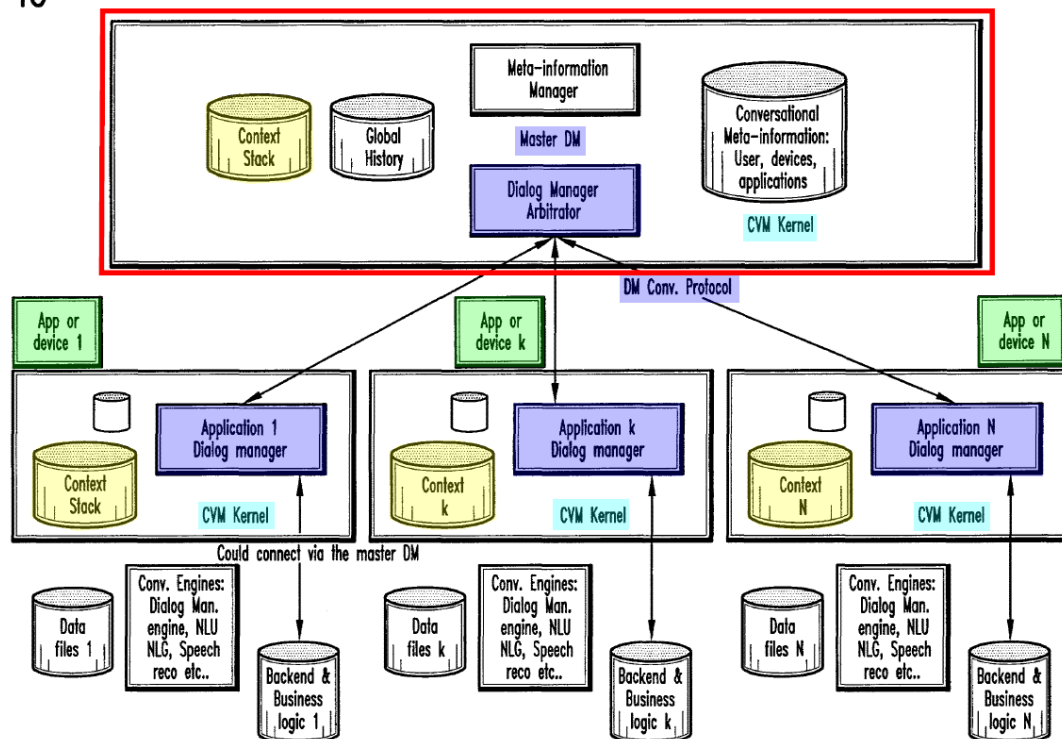
Coffman, Figure 9 (Annotated).

Figure 9 is a diagram of a dialog management process, including conversational protocols 204 for information exchange between networked devices (colored red and green), wherein each device has its own CVM (colored light blue), dialog manager 219 or 219a (colored purple), and context stack 217 or 217a (colored yellow). *Id.* at 10:13–14, 26:28–27:2. “Through the dialog manager DM protocols [colored purple] . . . the different dialog managers [colored purple] will negotiate a topology with a master dialog manager [at the red device] and slave or peer dialog managers [at the green device],” wherein “[t]he active master dialog manager [219] will be responsible for managing the flow of I/O to the different [slave] managers” to “*update the context.*” Ex. 1003, 27:2–7 (emphasis added); *see* Pet. 27–28, 30 (citing Ex. 1003, 27:1–7, 27:24–26). “For instance, the

following information can be exchanged:” “(4) data files (conversational arguments) if applicable i.e., if engines are used that are controlled by a master DM); “(5) notification of I/O events such as user input, outputs to users for transfer to engines and/or *addition to contexts*”; “(11) transfer of confidence and ambiguity results, proposed feedback and output, proposed expectation state, proposed action, *proposed context changes*, proposed new dialog state”; “(12) decision notification, *context update*, action update, state update, etc;” “(14) notification of *context changes*”; and “(15) data files, *context and state updates due to action.*” Ex. 1003, 27:7–20 (emphases added); see Pet. 31 (citing Ex. 1003, 27:7–20).

Figure 10 of Coffman is reproduced below, with our color annotations.

FIG. 10



Coffman, Figure 10 (Annotated).

Figure 10 is a diagram of a dialog management system and process similar to the process shown in Figure 9, and is annotated in color in the same fashion as Figure 9 above, but adds the additional detail of showing three different slave devices (colored green) in network communication with the master device (colored red). *See* Ex. 1003, 10:13–16, 27:24–28. Coffman describes how “[w]ith a master/slave network configuration” such as shown in Figure 10, “the master dialog manager can compile all the lists and match the functions and needs with conversational capabilities,” and “*context information* may be transmitted by indicating passing or pointing to the *context stack* / history of the device or application that the controller can access and *add to its context stack*.” *Id.* at 25:10–14 (emphases added); *see* Pet. 28–29, 30–31 (citing Ex. 1003, Fig. 10, 25:10–14, 27:5–7, 27:24–28, 28:24–27, 29:18–20).

Coffman further discloses that its various devices each have their own context stack, and that each device utilizes its context stack to store context information. *See* Ex. 1003, Figs. 9 and 10, 5:15–18 (“The context stack accumulates the context.”), 25:10–14, 27:1–20, 29:18–30:3 (describing Fig. 4); Ex. 1002 ¶¶ 64–67.

Thus, based on the evidence presented, we are persuaded by Mr. Schmandt’s testimony that a person of ordinary skill in the art would have understood Coffman’s various descriptions relating to updating context information between its various devices to satisfy the synchronization requirement of claim 1. *See* Ex. 1002 ¶¶ 64–67, 69–71. In particular, we find Coffman’s system “synchronizes” the master device’s context stack with a slave device’s context stack, such that the master device’s context stack is updated based on related context information of the slave device’s

context stack. *See id.* ¶¶ 66–67, 69–71; Pet. 26–32. For example, Mr. Schmandt cites Coffman’s statement that the master device’s dialog manager 219 “can compile all the lists and match the functions and needs with conversational capabilities,” wherein “context information may be transmitted by indicating passing or pointing to the context stack / history of the device or application that the controller can access and add to its context stack.” Ex. 1003, 25:10–14 (cited at Ex. 1002 ¶ 70); *see* Pet. 31–32. Mr. Schmandt also testifies persuasively that “Coffman further explains that, using dialog manager protocols, devices can exchange various kinds of information, including notification of ‘addition to contexts,’ ‘context update’ and ‘notification of context changes.’” Ex. 1002 ¶¶ 70–71 (citing Ex. 1003, 27:7–20); *see* Pet. 31.

Patent Owner’s opposing argument and expert testimony are not persuasive for the following reasons.

First, Patent Owner’s attempt to distinguish between the claimed “synchronizing” of context information and Coffman’s “updating” of context information relies heavily on Patent Owner’s claim construction of synchronizing context stacks in claim 1. We have rejected that claim construction above (*see supra* Section III.D.2(b)(ii)), so the bulk of Patent Owner’s discussion of Coffman and Mr. Occhiogrosso’s testimony in support is unavailing.

Patent Owner and Mr. Occhiogrosso also overlook that Coffman describes its “updating” of context information as “the *exchange* of information to *coordinate* the conversation involving multiple devices,” such as Coffman’s master device and slave devices. Ex. 1003, 22:32–23:15 (emphases added), 25:10–14, 26:28–27:20. We are persuaded by

Mr. Schmandt’s testimony that this results in synchronization of context information between the devices, such that (for example) the context stack of Coffman’s master device is updated based on the context stack of Coffman’s slave device(s). *See* Ex. 1002 ¶¶ 70–71. This is consistent, for example, with the ’652 patent’s description of “[t]he synchronization process” as “attempt[ing] to determine which version of a data element or an agent are the newest or most up-to-date and propagates that element,” including replacing “a portion of the database.” Ex. 1001, 17:25–32. Mr. Occhiogrosso’s contrary testimony relies on Patent Owner’s rejected claim construction of synchronizing context stacks. *See* Ex. 2013 ¶¶ 61–64, 67–77.

Second, Patent Owner argues and Mr. Occhiogrosso testifies that Coffman falls short of the claimed synchronization because each device in Coffman’s system can choose to act or not to act in response to exchanging information between devices. *See, e.g.*, Ex. 2013 ¶¶ 65–66, 75. However, we find that, even if there are instances where a given device in Coffman may not be synchronized with updated context information,⁶ still there are other instances when Coffman’s devices are synchronized with updated context information. *See* Ex. 1003, 25:10–14, 27:1–20. That is all that claim 1 requires.

For example, the ’652 patent specification correspondingly contemplates that “[t]he synchronization may be performed automatically when any two or more of the computers are connected to these networks,” or “[a]lternatively . . . the synchronization may be performed on demand under

⁶ *See, e.g.*, Ex. 1003, 27:17 (referring to “*proposed* context changes” (emphasis added)).

user control.” Ex. 1001, 17:15–25. Both of those scenarios contemplate points in time when the devices are not synchronized — in the first scenario when the computer is not connected to the network, and in the second scenario before the user initiates the synchronization. The ’652 patent specification similarly provides that “[s]ynchronization is an incremental change process” in some embodiments. *Id.* at 17:28–32. Thus, Coffman’s disclosure that each device can choose whether or not to synchronize its context information does not distinguish Coffman from the synchronization of claim 1, which can in some embodiments be performed only intermittently or incrementally.

Patent Owner suggests the “synchronization” of “context stacks” recited in claim 1 refers *only* to the “complete replacement of” a context stack as described in the ’652 patent specification at column 17, line 29. *See* Tr. 31:20–32:11 (discussing Ex. 1019, Slide 34). We disagree. The ’652 patent specification at column 17, lines 15–32 discusses synchronization of “data and agents” generically (Ex. 1001, 17:15–16), which includes context stacks specifically. There is no reason why a person of ordinary skill in the art would conclude that synchronization of contexts stacks would be limited to “complete replacement” of stacks (*id.* at 17:29) when that description is read in context (*see id.* at 17:15–32).

Next, because we find Coffman discloses the synchronization of contexts stacks as set forth above, we need not and we do not reach Petitioner’s alternative obviousness theory, proffered in the event that claim 1 is construed to require perfect synchronization. *See* Pet. Reply 7–8; PO Sur-reply 11 (lines 8–18).

And lastly, the final clause of claim 1 recites “determin[ing] the command or request based on the one or more words and *the updated context information*” introduced by the previously recited synchronization step. Ex. 1001, 38:60–61 (emphasis added). Petitioner accordingly asserts it would have been obvious to utilize such updated context information in the determination step. *See* Pet. 37. That assertion is not opposed by Patent Owner separately from whether Coffman discloses synchronizing two context stacks in the first instance. We conclude Petitioner’s contention is supported by a preponderance of the evidence. *See* Ex. 1002 ¶¶ 78–79.

For the foregoing reasons, we find Coffman discloses synchronizing a master device’s context stack with a slave device’s context stack such that the context information of the former is updated based on related context information of the latter, per claim 1.

(c) Conclusion as to Claim 1

We conclude Petitioner has shown by a preponderance of the evidence that claim 1 is unpatentable as having been obvious over Coffman.

3. Claims 2–7 of the ’652 Patent

Petitioner provides arguments and evidence in support of contending claims 2–7, each of which depends from independent claim 1, are unpatentable as having been obvious over Coffman, including supporting testimony from Mr. Schmandt. *See* Pet. 37–53; Ex. 1002 ¶¶ 81–110.

Patent Owner does not argue for the patentability of these claims separately from claim 1. *See, e.g.*, PO Resp. 14–16, 36. Therefore, Patent Owner has waived any such argument. *See* Inst. Dec. 40 (“Any arguments not raised by Patent Owner in a timely-filed response may be deemed

waived.”); Paper 12, 10 (“Patent Owner is cautioned that any arguments for patentability not raised in the response may be deemed waived.”).

We adopt Petitioner’s arguments as to claims 2–7 having been obvious over Coffman as our own in this Decision. *See* Pet. 37–53; Ex. 1002 ¶¶ 81–110. Accordingly, we conclude a preponderance of the evidence establishes claims 2–7 are unpatentable as having been obvious over Coffman.

4. Claims 8 and 9 of the ’652 Patent

Petitioner provides arguments and evidence in support of contending claims 8 and 9 are unpatentable as having been obvious over Coffman, including supporting testimony from Mr. Schmandt. *See* Pet. 53–59; Ex. 1002 ¶¶ 41–46, 111–119. Patent Owner and Mr. Occhiogrosso disagree. *See* PO Resp. 3, 19–29, 37–47; Ex. 2013 ¶¶ 55–60, 78–124.

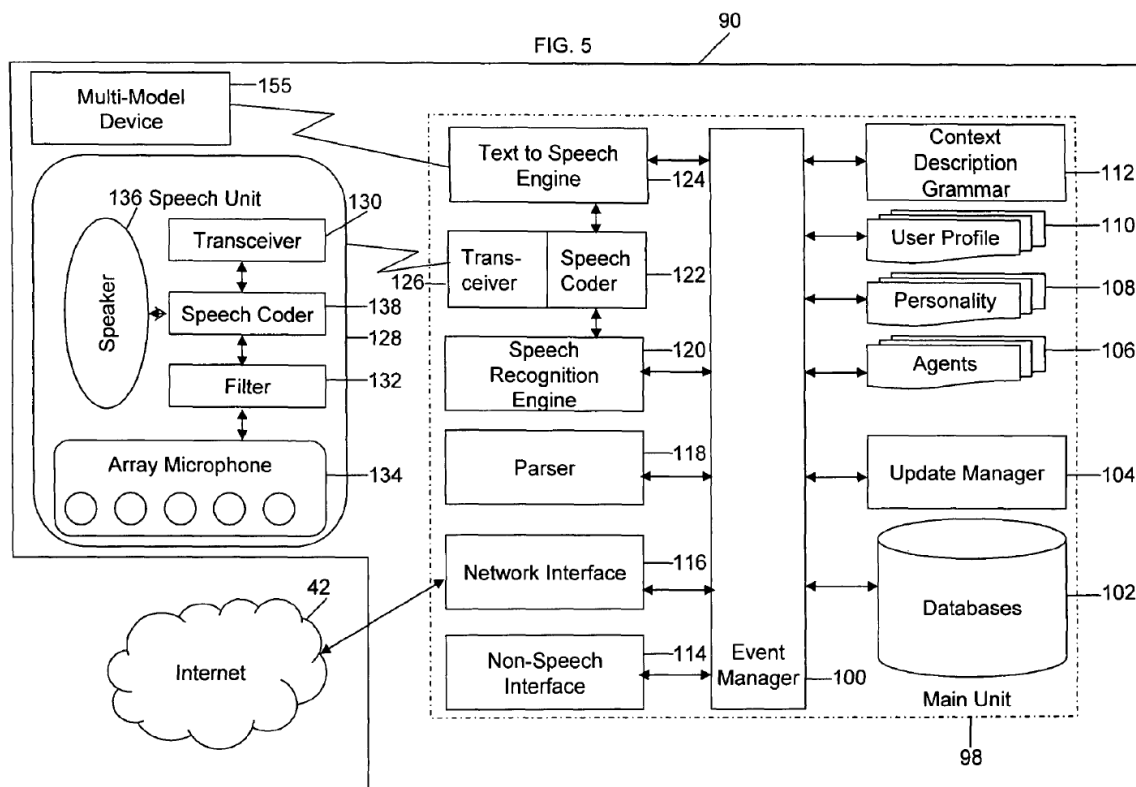
We conclude Petitioner has not demonstrated claims 8 and 9 are unpatentable by a preponderance of the evidence. Specifically, as discussed below, Petitioner has not demonstrated Coffman discloses one or more processors that identify one or more domain agents associated with one or more context entries in a context stack, per claim 8. Further, Petitioner does not rely on obviousness to address this claim limitation, so Coffman’s deficient disclosure is fatal to Petitioner’s case.

(a) Claims 8 and 9 and the Supporting ’652 Patent Specification

Claim 8 depends from claim 1, and adds that the one or more processors are programmed with instructions to: “identify one or more domain agents associated with the one or more context entries [in a context stack], wherein the one or more domain agents are configured to process the

command or request; and generate a response to the command or request using the one or more domain agents.” Ex. 1001, 39:47–54; *see also id.* at 39:5–6, 39:47 (claims 3 and 8 together recite that the “context entries” are maintained in the “context stack” of claim 1). Claim 9 depends from claim 8. *See id.* at 39:55.

The ’652 patent specification correspondingly provides various networked devices 10, 36, and 44 (discussed above in Section II.B) that “enable[] mobile users to submit natural language speech and/or non-speech questions or commands *in a wide range of domains*” and then “present responses in a natural manner.” *Id.* at 1:20–29 (emphasis added), 3:33–38. Figure 5 of the ’652 patent is reproduced below.



’652 Patent, Figure 5.

Figure 5 is a diagrammatic view of natural language processing system 90. *See id.* at 15:38–40, 23:36–37. The system includes main unit 98 and speech unit 128. *See id.* at 23:38–41.

Speech unit 128 includes array microphone 134 to record natural language utterances by the user. *See id.* at 25:48–52. Speech unit 128 filters, encodes, and transmits the recorded utterances to transceiver 126 of main unit 98. *See id.* at 25:52–56, 25:66–26:6, 31:33–50, 31:66–32:28.

Main unit 98 includes speech coder 122 to decode the recorded utterances received from speech unit 128. *See id.* at 26:15–18. The decoded utterances are transcribed into a text message by speech recognition engine 120 utilizing context description grammar module 112. *See id.* at 19:49–58, 23:56–61, 26:18–20, 33:7–11.

The speech transcription is sent to parser 118 for interpretation. *See id.* at 26:18–21, 33:8–11, 33:65–67. Parser 118 “transforms information into complete algorithms and questions using data supplied by knowledge agents . . . selected from the plurality of agents 106 using a grammar stack, wherein the knowledge agents provide information for generating a response to [a] question or command” contained within the transcription. *Id.* at 26:20–26; *see also id.* at 27:40–47 (“Based on keywords in the questions and commands and the structures of the questions and commands, the parser may invoke selected agents.”), 6:16–23 (“For each user utterance, including a question or query or set of questions or queries, the system may . . . determin[e] the domain of expertise required and the context to invoke the proper resources, including agents[.]”). “The knowledge agents [106] may then process the commands or questions” to result in actions taken by system 90. *Id.* at 26:27–35.

Agents 106 may include “domain specific agents” that provide “domain specific behavior.” *Id.* at 27:13–23, Fig. 6 (illustrating domain agents 156). In particular, parser 118 examines recorded utterances to “determine a context for an utterance” based on, for example, “context stack contents.” *Id.* at 33:65–34:5. “The context of a question or command may determine the domain and, thereby, the domain agent 156, if any, to be invoked.” *Id.* at 34:5–15; *see also id.* at 34:34–36 (“Once the context for the question or command has been determined, the parser 118 can invoke the correct agent 156, 150.”).

(b) The Petition

In the Petition, Petitioner argues: “A ‘**domain agent**’ in Coffman corresponds to the executable software of *a conversationally-aware application . . . and associated data.*” Pet. 53–54 (italicized emphasis added) (citing Ex. 1001, 27:14–17, 28:11–12; Ex. 1003, 13:26–29); Ex. 1002 ¶¶ 112–113. Petitioner asserts, referring to Coffman’s Figure 1, that the “[c]onversationally-aware applications [11] ‘are applications that are specifically programmed for operating with a CVM core layer (or kernel) 14 via conversational application APIs 13.’” Pet. 54 (citing Ex. 1003, 12:28–30); Ex. 1002 ¶ 113. Further according to Petitioner, “[a] conversationally-aware application operates with a CVM by providing its data files to the CVM for use by the conversational subsystems [of the CVM].” Pet. 54 (citing Ex. 1003, 36:31–32, 57:23–27); Ex. 1002 ¶ 113. Petitioner contends Coffman’s conversationally-aware applications moreover receive, process, and respond to user questions, queries and commands to “provide[] functionality related to a specific domain, such as a

particular subject matter (such as email or spreadsheet) or device (such as a PDA or smart phone).” Pet. 54–55 (citing Ex. 1003, 13:8–13, 16:35–17:7, 36:29–31, 37:2–4, 40:9–19, 56:11–14, 56:22–25, 58:30–32); Ex. 1002 ¶ 114.

Petitioner next argues “[a] conversationally-aware application in Coffman is ‘**associated with the one or more context entries**’” because “Coffman explains that context stack entries are created for and associated with applications.” Pet. 55 (citing Pet. 25–30; Ex. 1003, 29:18–30:10); Ex. 1002 ¶ 115. Moreover: “A conversationally-aware application in Coffman is ‘**configured to process the command or request**’” because “Coffman describes completing and activating a context entry’s command / query for the associated application to process.” Pet. 55–56 (citing Pet. 35–37; Ex. 1003, 37:2–4, 40:12–15, 41:2–5, 42:1–5); Ex. 1002 ¶ 116.

Petitioner lastly argues Coffman’s processors “**generate a response to the command or request using the one or more domain agents**” per the final limitation of claim 8. Pet. 56–57; Ex. 1002 ¶ 117.

(c) The Patent Owner Response

Patent Owner argues Petitioner’s challenge to claim 8 here fails for the same reasons the Board denied institution of Petitioner’s petitions in IPR2024-00746, IPR2024-00747, IPR2024-00748, IPR2024-00749, and IPR2024-00751 challenging other patents owned by Patent Owner. *See* PO Resp. 3, 15–16. In particular, according to Patent Owner, “Petitioner’s interpretation of Coffman’s relationship to the domain agents claimed in the patents-at-issue in those cases was deficient,” and the present Petition’s analysis of claim 8 “related to the selection of a domain agent to process a user’s utterance” (*id.* at 3) and “correlat[ing] Coffman’s conversationally-

aware applications with the claimed ‘domain agent’” (*id.* at 16) reflect the same deficiencies.

Patent Owner particularly asserts: “Coffman’s CVM does not identify any conversationally-aware application, based on an identified context entry, configured to process a command or request.” PO Resp. 38. Instead, according to Patent Owner, “in Coffman it is a user who ‘selects’ a conversationally-aware application to be used.” *Id.* at 38, 43–44; Ex. 2013 ¶¶ 85, 98, 110–111.

In support, Patent Owner argues “Coffman’s conversationally-aware applications are gateways to use the system.” PO Resp. 37–38. Specifically, Patent Owner contends Coffman’s conversationally-aware applications provide “an interface (UI) for a user to provide the speech-based content for a given conversationally-aware application to interact with users” and then “access[] the tools provided by Coffman’s conversational virtual machine (‘CVM’) . . . and await[] a response from the CVM to continue a communication with a user.” *Id.* at 38. The CVM, in Patent Owner’s view, “activates a processing thread that accesses a context stack specific to [the initiating] conversationally-aware application, identifies the active discourse (or conversation) from that context stack, and carries out a process associated with the active discourse’s context.” *Id.* at 38–40 (citing Ex. 1003, 42:1–7).

Patent Owner argues, referring to Coffman’s Figures 2 and 4, that CVM 202 / 401 generates a response to a command or request via various backend subsystems 213 / 223 / 406 / 407 of the CVM, *without* “using” Coffman’s frontend conversationally-aware applications 200. *See* PO Resp. 44, 45–47 (citing Ex. 1003, 13:8–13, 22:1–7, 29:10–11, 58:9–11); *see also*

id. at 24–28 (further citing Ex. 1003, Figs. 2 and 4, 5:5–6, 13:8–10, 21:23–32, 22:1–7, 27:1–2, 28:10–12, 29:4–11, 29:19–20, 29:27–32, 30:3–8, 30:13–15, 30:22–25, 41:9–13, 43:1–2); Ex. 2013 ¶¶ 97–98, 112, 117. This is demonstrated, according to Patent Owner, by Coffman’s descriptions that “‘each new task, process, or thread creates a new stack entry and is associated with a discourse,’ and that each ‘application may be associated with multiple discourses’” wherein “each application has its own context stack.” PO Resp. 44–45 & n.5 (citing Ex. 1003, 10:4–6, 22:1–3, 27:1–2); *see also id.* at 28–29 (further citing Ex. 1003, Fig. 4, 11:2–4, 27:1–2, 29:18–27); Ex. 2013 ¶¶ 98, 108, 113.

Patent Owner particularly asserts that Coffman’s description of a CVM dialog manager identifying a “corresponding process” in a context stack (Ex. 1003, 42:1–7) “does not disclose that the ‘process’ . . . occurs at a ‘conversationally-aware application.’” PO Resp. 40; Ex. 2013 ¶¶ 95, 104. “Indeed,” according to Patent Owner, “Coffman associates a ‘process’ with tasks and threads performed *at the CVM*” not at a conversationally-aware application, and “Coffman discloses that ‘all conversational subsystem calls are hidden to the application.’” PO Resp. 40–41 (citing Ex. 1003, Fig. 2, 5:5–6, 21:1–3, 29:7–11, 30:1–5, 30:26–28, 42:16–17); Ex. 2013 ¶¶ 95, 98, 104–106. Patent Owner concludes “Petitioner is incorrect that identification of a context in a context stack yields the identification of a conversationally-aware application” in Coffman, because the CVM “subsystems are invisible to the conversationally-aware application, which receives the final results to be reported to the user.” PO Resp. 41–42.

This is confirmed, urges Patent Owner, by Coffman’s disclosure “that where there are different applications, ‘different devices, CVMs or different

applications will have their own dialog manager [and] context stack 217.” *Id.* at 42 (citing Ex. 1003, 21:23–24, 27:1–2). Thus, although “a search of an application’s own context stack 217 may identify an active context and active process for that application,” such local processing “does not support Petitioner’s expansive reading that Coffman’s CVM will then identify a conversationally-aware application among many applications to process a user’s request based on the context stack search.” *Id.* at 42–43 (citing Ex. 1003, 29:4–5, 30:4–5, 30:22–25); Ex. 2013 ¶¶ 108–109.

(d) The Petitioner Reply

Petitioner replies that “Patent Owner’s arguments are all founded on the same misreading of Coffman, that a context stack is not used to invoke a conversationally-aware application (the claimed ‘**domain agent**’) to process a command / request.” Pet. Reply 1, 10; Ex. 1017 ¶ 15.

Petitioner firstly disputes Patent Owner’s assertion that the user of Coffman’s system selects a conversationally-aware application as a frontend gateway to the system, “much like how users often manually toggle among applications in a Microsoft Windows environment,” as “mak[ing] no sense.” Pet. Reply 10–11 (citing PO Resp. 37–38). Petitioner asks, if Patent Owner’s view is correct, then “why would there be more than one conversationally-aware application for a user to interact with?,” as “[o]ther conversationally-aware applications would simply be redundant, and there would be no need to toggle among them.” *Id.* at 11; Ex. 1017 ¶ 19.

Referring to Figure 1 of Coffman, Petitioner asserts Patent Owner confuses Coffman’s conversational user interface 10, which serves as the gateway to Coffman’s system, with Coffman’s conversationally-aware

applications 11. Pet. Reply 11–12 (citing Ex. 1003, 64:3); Ex. 1017 ¶ 19. Petitioner further asserts that it is *Coffman’s CVM dialog manager* that “allows a user to seamlessly control multiple Windows applications, for example, running in parallel” and “manag[e] dialog across the applications and . . . select[] an . . . application based on input queries” including identifying and selecting a conversationally-aware application. Pet. Reply 12–13 (citing Pet. 23, 55; Ex. 1003, 3:14–16, 4:13–16, 20:19–23, 31:24–29, 37:18–20, 56:11–14, 65:12–13 (claim 11)); Ex. 1017 ¶ 20.

Petitioner secondly disputes Patent Owner’s contention that “Coffman’s context stack search . . . identifies a process performed at the CVM” rather than at Coffman’s conversationally-aware application(s). Pet. Reply 13–14 (citing PO Resp. 39–40; Ex. 1003, 42:16–17); Ex. 1017 ¶ 21. According to Petitioner, “Patent Owner’s citation to Coffman’s statement that task dispatcher / controller 402 ‘dispatches and coordinates different tasks and processes that are spawned (by the user and machine) . . . actually confirms that the tasks, threads and processes are associated with applications launched by the user — such as conversationally-aware applications.” Pet. Reply 14 (citing PO Resp. 41; Pet. 22, 55; Ex. 1003, 4:13–16, 4:27–29, 6:6–8, 30:26–28, 36:29–31); Ex. 1017 ¶ 22. Also, “even if the tasks, processes and threads Coffman describes could be understood as encompassing processes performed at the CVM,” Petitioner argues “it is enough [to demonstrate obviousness] that . . . a conversationally aware application would *at least sometimes* be identified to process a query.” Pet. Reply 15 (emphasis added) (citing *Unwired Planet, LLC v. Google Inc.*, 841 F.3d 995, 1002 (Fed. Cir. 2016)).

Petitioner dismisses Coffman’s disclosures that “all conversational subsystem calls are hidden to the application” (Ex. 1003, 29:7–11) as “relat[ing] to services such as speech recognition and text-to-speech synthesis” that “are used to search the context stack and determine the command / request,” rather than “the results of performing a command / request.” Pet. Reply 15–16 (citing Pet. 36; Ex. 1003, 29:1–6, 41:2–5, 42:27–29); Ex. 1017 ¶ 24.

Petitioner thirdly disputes Patent Owner’s reliance on Coffman’s statements that “different devices, CVMs or different applications will have their own dialog manager [and] context stack 217, 217a” (Ex. 1003, 27:1–2) and “context stack 217 comprises all the information associated with an application” (*id.* at 21:23–24). *See* Pet. Reply 16 (citing PO Resp. 42). Petitioner asserts those “statements simply do not require that every context stack is limited to a single application.” *Id.*; Ex. 1017 ¶¶ 25–26. Petitioner emphasizes Coffman states “different devices, CVMs or different applications will have their own . . . context stack 217, 217a.” Pet. Reply 16 (citing Ex. 1003, 27:1–2). That statement, in Petitioner’s view, “is entirely consistent with two devices that each have their own context stack that includes context entries for multiple applications,” as allegedly shown by Figure 9 of Coffman “depict[ing] just one context stack associated with each device and the CVM and ‘conversational applications’ on that device.” *Id.* at 16–17 (annotating Figure 9); Ex. 1017 ¶ 26. Petitioner asserts “Coffman’s statement is actually referring to . . . an optional, alternative ‘stand-alone’ type of conversationally-aware application . . . carrying its own CVM,” and “Coffman does not require that every application have its own, dedicated

context stack.” Pet. Reply 17–19 (citing Pet. 44; Ex. 1003, Fig. 4, 28:10–18, 32:27–28, 59:12–13; Ex. 1018, 32:11–23); Ex. 1017 ¶ 27.

Petitioner moreover asserts that “even if Patent Owner were correct that a context stack in Coffman is limited to a single application,” still “Coffman explains that user input may not necessarily match any entry in a context stack, and if not, other context stacks are searched to find a match.” Pet. Reply 19–20 (citing Pet. 44; Ex. 1003, 55:27–56:3); Ex. 1017 ¶ 25. Then, “when a matching entry is actually found in a context stack, Coffman’s system has ‘**identified**’ the associated application for processing the command/request.” Pet. Reply 20 (citing Pet. 47; Ex. 1003, 37:2–4); Ex. 1017 ¶ 25.

Petitioner fourthly maintains Coffman’s CVM dialog manager generates a response to the command or request using one or more of the conversationally-aware applications. *See* Pet. Reply 21; Ex. 1017 ¶ 28. Petitioner asserts Coffman’s CVM dialog manager allows the user to control multiple Windows applications running in parallel, such as a spreadsheet application and an email application, wherein the applications are utilized to access information and generate a response. *See* Pet. Reply 21–22 (citing Pet. 55; Ex. 1003, 4:13–16, 35:19–36:24, 56:11–14, 65:12–13 (claim 11); Ex. 1018, 36:13–24); Ex. 1017 ¶ 28. According to Petitioner, “Coffman never suggests that, in order to make the spreadsheet application ‘conversationally aware,’ the application’s core functionality would be moved to the CVM or conversational subsystems.” Pet. Reply 22 & n.1 (citing Ex. 1003, 13:29–32, 14:8–12, 42:27–29); Ex. 1017 ¶ 28.

Petitioner finally argues that other passages in Coffman “are consistent with the admissions of Patent Owner’s own expert

[Mr. Occhiogrosso] that a conversationally aware application would be invoked [by the CVM] to respond to a user command / request, based on determining the meaning of the user’s utterance.” Pet. Reply 22–23 (citing Ex. 1003, 13:8–13, 42:27–29, 58:9–11; Ex. 1018, 35:19–36:24); Ex. 1017 ¶ 29.

(e) The Patent Owner Sur-reply

Patent Owner replies that the Petition contains five (5) pages of argument directed to claim 8, which in the Petitioner Reply balloons to nearly fourteen (14) pages of “new” and “untimely” argument. PO Sur-reply 4, 13.

Patent Owner then reiterates that “Petitioner is incorrect that identification of a context entry in a context stack [in Coffman] yields the identification of a conversationally-aware application configured to process a user’s request or command” per claim 8. *Id.* at 13. Patent Owner asserts the Petitioner Reply “create[s] endless confusion rather than clearing up how Coffman operates.” *Id.* Patent Owner argues the Coffman disclosures cited by Petitioner fail to “indicate[] that Coffman identifies a context entry in a context stack, and then identifies which conversationally-aware application can be configured to process the user’s request or command based on the identified context entry.” *Id.* at 13–16 (citing Pet. Reply 12–13, 19, 21; Ex. 1003, 4:13–16, 30:26–31:23, 55:27–56:3, 65:12–13 (claim 11)).

(f) Analysis and Conclusion

We conclude a preponderance of the evidence does not support Petitioner’s contention that Coffman discloses one or more processors that *identify* a domain agent (i.e., one of Coffman’s conversationally-aware

applications 11 per Petitioner’s mapping) *associated with* one or more context entries in a context stack per claim 8. Further, Petitioner does not rely on obviousness to address this claim limitation, so Coffman’s deficient disclosure is fatal to Petitioner’s case.

(i) *The Petition and Mr. Schmandt’s Initial Testimony*

The Petition and Mr. Schmandt’s Declaration in support (Exhibit 1002) entirely fail to address how Coffman’s processor(s) might be programmed to “identify” one or more conversationally-aware applications per claim 8. *See* Pet. 53–57; Ex. 1002 ¶¶ 64–68, 112–116. Petitioner and Mr. Schmandt stop short of that claim requirement by concluding merely that “[a] conversationally-aware application in Coffman is ‘**associated with the one or more context entries.**’” Pet. 55; Ex. 1002 ¶ 115.⁷ Claim 8 requires more than mere association; it requires the processor to identify a domain agent that is associated with one or more entries in a context stack. *See* Ex. 1001, 38:43–45 (parent claim 1), 39:47–50 (claim 8). The Petition and Mr. Schmandt’s initial testimony do not address that claim requirement.

Next, we consider the argument and testimony that *is* proffered. Details are provided below, but in overview, we determine Coffman’s disclosure is too high-level and non-specific, and Mr. Schmandt’s supporting testimony is too conclusory, to satisfy Petitioner’s burden to demonstrate obviousness by a preponderance of the evidence.

⁷ By contrast, in the related IPR2024-00751, Petitioner and Mr. Schmandt address the “**associated**” claim term and then address the “**identify**[ing]” claim term. *See* IPR2024-00751, Paper 2 at 49–50, Ex. 1002 ¶¶ 101–102.

Mr. Schmandt firstly testifies that Coffman’s conversationally-aware applications 11 are “**domain agents**” per claim 1 because they include “executable software . . . and associated data, such as vocabulary, grammar, language models and [Natural Language Understanding] symbolic mapping” so they correspond to a “program that performs actions on behalf of another entity.” Ex. 1002 ¶¶ 112–114 (citing Ex. 1001, 27:14–17, 28:11–12); *see* Pet. 53–55; *see also* Ex. 1003, 3:18 (defining the “NLU” acronym). We conclude this testimony, which Patent Owner does not dispute, is supported by the ’652 patent specification passages and Coffman disclosures cited by Mr. Schmandt. *See* Ex. 1002 ¶¶ 112–114.

Mr. Schmandt then testifies that Coffman’s applications 11 are “**associated with the one more context entries**” in a context stack per claim 8 because:

As discussed for [the “generate a first context stack” limitation of parent claim 1], *Coffman explains that context stack entries are created for and associated with applications.* (Coffman, 30:4–10 (“More specifically, each new task, process, or thread creates a new stack entry and is associated with a discourse. Each application may be associated with multiple discourses (e.g. the application management discourse and the application content navigation discourses).”); *see also id.*, 29:18–30:3.

Ex. 1002 ¶ 115 (emphasis added); *see* Pet. 55.

The cited disclosure describes Coffman’s conversational system 400, shown in Figure 4, which is managed by CVM 401 utilizing context stack 405. *See* Ex. 1003, 28:22–30:12. We do not discern anything there to indicate context stack 405 is created for and associated with *applications* as Mr. Schmandt testifies, much less that such applications might include

Coffman's *conversationally-aware applications* which Petitioner specifically maps to the domain agents of claim 8. *See id.*

Indeed, Coffman describes how “each of these conversational subsystems 407 *may be accessed through API calls to the CVM 401,*” in response to which the CVM “will locate the requested conversational subsystem 407 . . . , drive its execution and return appropriately the results” wherein “all conversational subsystem calls are *hidden to the application*” that initiated the API call. *Id.* at 29:6–13 (emphasis added). That disclosure supports Patent Owner's position that Coffman's conversationally-aware applications utilize APIs to invoke CVM 401 to process a command or request, and contradicts Petitioner's position that Coffman's CVM 401 utilizes context stack 405 to identify a conversationally-aware application to be invoked to process a command or request. *See also* Ex. 1003, Fig. 1 (illustrating communication between conversationally-aware application 11 and CVM 14 via API 13).

Further, the specific disclosure quoted by Mr. Schmandt states “each new task, process, or thread creates *a new stack entry* and is associated with *a discourse,*” as opposed to being associated with an “application” as Mr. Schmandt testifies, and also states “[*e*]ach application may be associated with multiple discourses.” Ex. 1003, 30:4–6 (emphases added); *see also id.* at 29:27–32 (“In other words, the term ‘context’ refers to the state of each *discourse.*” (emphasis added)), 30:6–8 (“*Each context associated with a given discourse* comprises the latest requests made to the corresponding process / task / thread as well as the latest output.” (emphasis added)). We cannot discern, absent persuasive explanation from Mr. Schmandt, how Coffman's disclosure of determining a matching

discourse in a context stack can correspond to determining a matching *application* such as one of Coffman’s conversationally-aware applications 11, especially when each application can be associated with multiple discourses. Such explanation is lacking in the paragraph of Mr. Schmandt’s testimony quoted above (Ex. 1002 ¶ 115).

Such explanation also is lacking in Mr. Schmandt’s previous testimony comparing the “generate a first context stack” limitation of claim 1 and Coffman, which he cross-references in the testimony quoted above. *See* Ex. 1002 ¶¶ 64–68 (citing Ex. 1003, 6:8–10, 6:26–28, 6:32–7:3, 19:26–29 (Fig. 2), 22:32–23:3 (Fig. 3), 27:1–7 (Fig. 9), 27:24–28 (Fig. 10), 28:24–27 (Fig. 4), 29:18–30:10 (Fig. 4), 37:18–26 (Fig. 6), 41:2–5). The various Coffman disclosures cited by Mr. Schmandt there most pertinently describe Coffman’s context stack(s) as being associated with a “discourse” as discussed above, or perhaps with a non-specific “task” or “process” or “thread” or “engine” or “network service” or “conversational subsystem.” *See id.* Mr. Schmandt does not explain how such non-specific disclosures lead him to conclude that Coffman’s conversationally-aware applications 11 (specifically) are “associated with the one or more context entries” per claim 8. *See id.* ¶¶ 64–68, 115. Indeed, Coffman’s description that “task dispatcher 402 [of CVM 401] *redirects the stream to the appropriate conversational subsystems or conventional subsystem*” appears to refer to conversational resource subsystems 407 and conventional subsystem services accessed via manager 412 in Figure 4, not the conversationally-aware applications 11 in Figure 1 which access CVM 14 / 401 through APIs 13. Ex. 1003, 28:22–29:17, 40:30–41:7 (emphasis added).

Mr. Schmandt goes on to testify that Coffman’s conversationally-aware applications 11 are “configured to process the command or request” per claim 8 because “Coffman describes completing and activating *a context entry’s command / query for the associated application to process.*” Ex. 1002 ¶ 116 (emphasis added) (citing Ex. 1002 ¶¶ 78–80; Ex. 1003, 37:2–4, 40:12–15, 41:2–5, 42:1–5); *see* Pet. 55–56. Once again, however, the cited disclosures fail to indicate that Coffman’s context entries are associated with *applications* as Mr. Schmandt testifies, much less that such applications might include Coffman’s *conversationally-aware applications*. *See* Ex. 1003, 37:2–4, 40:12–15, 41:2–5, 42:1–5. And, the cross-referenced testimony again most pertinently describes Coffman’s context stack(s) as being associated with non-specific “processes” or “conversational subsystems,” and Mr. Schmandt does not explain how such non-specific disclosures lead him to conclude that Coffman’s conversationally-aware applications 11 (specifically) are associated with the one or more context entries per claim 8. *See* Ex. 1002 ¶¶ 78–80 (citing Ex. 1003, 30:8–10, 37:20–26, 40:12–15, 41:2–5, 42:1–5, 51:12–14, 55:27–30).

Finally, Petitioner’s argument and Mr. Schmandt’s testimony concerning the final “generate a response” limitation of claim 8 do not add anything material to our foregoing consideration of the “identify one or more domain agents” limitation of claim 8. *See* Pet. 56–57; Ex. 1002 ¶ 117. Similarly, Petitioner’s argument and Mr. Schmandt’s testimony concerning claim 9 also do not add anything material to our foregoing consideration of claim 8. *See* Pet. 58–59; Ex. 1002 ¶¶ 118–119.

Thus, we conclude the Petition and Mr. Schmandt’s supporting testimony fail to demonstrate, by a preponderance of the evidence, that

Coffman discloses one or more processors that identify one or more domain agents associated with one or more context entries in a context stack, per claim 8. As noted by Patent Owner, our analysis above is substantially the same as the Board’s reasoning behind denying institution in IPR2024-00746 (Paper 11, at 22–30), IPR2024-00747 (Paper 11, at 15–21), IPR2024-00748 (Paper 11, at 15–20), IPR2024-00749 (Paper 11, at 14–19), and IPR2024-00751 (Paper 11, at 18–22) concerning Petitioner’s challenges to other patents owned by Patent Owner. By contrast, the Institution Decision granting review in the present IPR concluded Petitioner had established a reasonable likelihood of prevailing on claim 1 (*see* Inst. Dec. 1, 13, 27, 40), and did not address the merits concerning claim 8 (*see id.* at 28–29).

(ii) *The Petitioner Reply and Mr. Schmandt’s Reply Testimony*

As noted above in Section III.D.4(f)(i), the Petition and Mr. Schmandt’s initial Declaration (Exhibit 1002) entirely fail to address how Coffman’s processor(s) might be programmed to “identify” one or more conversationally-aware applications per claim 8. As a result, the additional argument in the Petitioner Reply concerning this claim requirement is untimely and should not be considered. *See* PO Sur-reply 4, 13; *e.g.*, *Corephotonics, Ltd. v. Apple Inc.*, 84 F.4th 990, 1002 (Fed. Cir. 2023) (“[T]he expedited nature of IPRs bring with it an obligation for petitioners to make their case in their petition” and “the Board is not permitted to entertain ‘[s]hifting arguments’ but must, instead, reject any ‘entirely new theory of *prima facie* obviousness absent from the petition,’ if the petitioner attempts to inject such a theory into the proceeding post-petition.” (citations omitted)); *Wasica Fin. GmbH v. Cont’l Auto. Sys.*,

Inc., 853 F.3d 1272, 1286 (Fed. Cir. 2017) (“Here, [the petitioner] did not make out its obviousness case in its petition” and “[a]fter [the patent owner] pointed out the flaws of [the petitioner’s] position, [the petitioner’s] ensuing arguments to the Board and to us effectively abandoned its petition in favor of a new argument” thereby “amount[ing] to an entirely new theory of *prima facie* obviousness” which “is foreclosed by statute, our precedent, and Board guidelines.”).

Nonetheless, in order to provide a complete record for federal court review if sought by Petitioner, we will go on to consider the substantive merit of the arguments in the Petitioner Reply. We conclude those arguments and Mr. Schmandt’s supporting testimony (Exhibit 1017) do not satisfy Petitioner’s burden to demonstrate obviousness by a preponderance of the evidence, for the following reasons.

First, the bulk of Petitioner’s argument seeks to establish that Patent Owner’s and Mr. Occhiogrosso’s various readings of Coffman are incorrect, rather than why Petitioner’s reading is correct. *See supra* Section III.D.4(d) (summarizing the Petitioner Reply). This misses the mark by attempting to shift Petitioner’s burden to Patent Owner. *See* 35 U.S.C. § 316(e); 37 C.F.R. § 42.1(d); *Harmonic*, 815 F.3d at 1363; *Dynamic Drinkware*, 800 F.3d at 1378.

Second, even though Patent Owner does not have the burden here, we agree with Patent Owner’s position that Coffman *does not* utilize its context stacks to identify and invoke a conversationally-aware application. We accordingly disagree with Petitioner’s opposite position that Coffman *does* utilize its context stacks to identify and invoke a conversationally-aware application.

Referring to Figure 1, Coffman describes “conversationally aware applications 11 (i.e., applications that ‘speak conversational protocols’)” as “applications that are specifically programmed for operating with a CVM core layer (or kernel) 14 via conversational application APIs 13.” Ex. 1003, 12:26–30. Coffman further discloses that “CVM kernel 14 controls the dialog across applications and devices on the basis of their registered conversational capabilities and requirements and provides a unified conversational interface.” *Id.* at 12:30–32. Conversational sub-systems / engines 18 (*not* conversationally-aware applications 11) are responsible for converting voice requests into queries and outputs into spoken messages. *See id.* at 13:8–13. Although Coffman discloses that “[c]onversational application API 13 conveys all the information for the CVM 14 to transform queries into application calls and conversely output into speech” (*id.* at 13:11–13), Petitioner has not demonstrated Coffman’s conversationally-aware applications 11 are able to identify another conversationally-aware application 11 to process a command or request.

Instead, as shown in Figure 4 of Coffman, CVM 14 / 401 includes task dispatcher / controller 402 which associates each task or device with a conversational engine and with conversational arguments. *See id.* at 40:20–21. Coffman discloses that, based on the modality (e.g., pointer, keyboard, file, or speech), task dispatcher 402 redirects the stream to the appropriate conversational subsystem 407 (or conventional subsystem 412 / 416 / 417) with speech inputs being transcribed and understood. *See id.* at 41:2–4. The outputs of those subsystems are run down context stack 405 in CVM 14 / 401 to extract a query and complete it. *See id.* at 41:4–5. Context stack 405 is traversed until the user input context is established; this

means finding and selecting the active discourse between the user and the machine from among the last and past discourses. *See id.* at 41:8–13. The results of complete and unambiguous commands are returned to conversation engine 407. *See id.* at 41:24–26.

Coffman describes how a new query is compared by a dialog engine (within the CVM) going down the context stack (within the CVM) and, if a context is found that fits the utterance, activating the context and the corresponding process. *See id.* at 42:1–7. Petitioner does not explain how Coffman’s disclosure of activating a process after traversing context stack 405 in the CVM teaches identifying a conversationally-aware application 11 to be invoked. Coffman’s disclosure that “[e]ach action, completed query and conversational argument of an active task as well as each returned value / result are stored on the context stack 405” (*id.* at 40:27–29) suggests that traversing the context stack concerns determining the context based on applications that have already been selected. *See also id.* at 41:8–13 (disclosing that traversing the context stack “essentially amounts to finding and selecting the active discourse between the user and machine among the last and past discourses, possibl[y] going back into history”). Petitioner does not explain how selection of the active discourse teaches selecting a conversationally-aware application 11.

Coffman’s Figure 2 illustrates conversational application API layer 203 comprised of conversational foundation classes 205 as CUI building blocks and conversation platform libraries, dialog modules, and dialog scripts. *See id.* at 13:29–14:6. Noting that data files for the foundation classes are present on the CVM (loadable for embedded platforms), Coffman discloses high-level application specific foundation

classes, e.g., classes that manage a bank account, manage a portfolio, request travel reservations, manage e-mail, manage a calendar, or manage an address book. *See id.* at 15:9–11, 16:35–17:7. Coffman’s traversal of a context stack merely concerns which process to execute next; as the foundational data classes are present on the CVM, the selection of which process to execute next occurs after the application has been selected.

Coffman’s conversationally-aware applications 11 can make calls to CVM 14 / 401 via APIs 13. *See id.* at Figs. 1 and 4, 32:11–13. Based on its traversal of context stack 405, Coffman’s “CVM 401 decides which application is active and how the context is to be modified (. . . with the dialog manager and conversational protocols).” *Id.* at 32:2–4, 41:8–13. When calls are made through CVM 401, they are registered through task dispatcher / controller 402 and the dialog (which can be multi modal and even without any speech input or output) is managed by dialog controller 404. *See id.* at 32:13–16. However, Petitioner has not shown that the CVM’s decision about which application is already active and has made a call to the CVM constitutes selecting a conversationally-aware application 11 based on the determined meaning of an utterance.

Indeed, as Patent Owner points out, calls by Coffman’s CVM 14 / 202 / 401 to conversational sub-systems 208 (in Fig. 2) or 407 (in Fig. 4) are hidden to applications 11 / 200. *See* PO Resp. 41–42 (citing Ex. 1003, 5:5–6, 29:10–11); *see also* Ex. 1003, 29:6–9 (“CVM 401 will locate the requested conversational subsystem 407 (via the conversational subsystem services manager 406), drive its execution and return appropriately the results.”)). As a conversational subsystem 208 / 407, rather than a conversationally-aware application 11 / 200, executes the request, we

determine Petitioner has not demonstrated that Coffman's CVM 14 / 202 / 401 identifies one or more conversationally-aware applications associated with one or more context entries in context stack 405.

Petitioner and Mr. Schmandt attempt to tie together various disparate disclosures in Coffman to allege that *Coffman's CVM dialog manager 219 / 404* (specifically) allows a user to seamlessly control multiple Windows applications running in parallel, including identifying and selecting a conversationally-aware application. *See* Pet. Reply 12–13, 21 (citing Ex. 1003, 3:14–16, 4:13–16, 20:19–23, 31:24–29, 35:19–36:24, 37:18–20, 56:11–14, 65:12–13 (claim 11)); Ex. 1017 ¶¶ 20, 28. However, those disclosures in fact state more generically that “the conversational computing paradigm” of Coffman's system provides that functionality, rather than the CVM dialog manager specifically. Ex. 1003, 4:13–16, 56:11–14. Moreover, to the extent these disclosures refer to interactions between the CVM dialog manager and “applications,” they are too general to demonstrate by a preponderance of the evidence that the CVM dialog manager *identifies* one of the conversationally-aware applications (specifically) *to be invoked to process a command or a request*, as Petitioner alleges. *See id.* at 20:19–23, 31:24–29, 35:19–36:24, 37:18–20, 65:12–13 (claim 11).

We acknowledge claim 11 of Coffman refers to a dialog manager “*selecting an active dialog, context and application based on input queries.*” *Id.* at 65:11–17 (emphasis added); *see also* Tr. 8:1–16, 16:5–9, 17:12–21 (Petitioner's counsel discussing claim 11). However, as discussed above, we find claim 11 thereby refers to applications that have already been selected

by the user and are therefore “active.” *See supra* (paragraph beginning “Coffman describes how a new query is compared”).

Petitioner and Mr. Schmandt also assert that “when a matching entry is actually found in a context stack, Coffman’s system has ‘**identified**’ the associated application for processing the command / request.” Pet. Reply 19–20 (citing Ex. 1003, 37:2–4, 55:27–56:3); Ex. 1017 ¶ 25. However, for reasons provided above, we disagree with Petitioner’s underlying premise that the context entries in Coffman’s context stack are “associated” with conversationally-aware applications. The Coffman disclosures cited here do not alter that determination. *See* Ex. 1003, 37:2–4, 55:27–56:3.

Further, we do not agree with Petitioner’s assertion that Mr. Occhiogrosso admitted that in Coffman “a conversationally aware application would be invoked [by a CVM dialog manager] to respond to a user command / request, based on determining the meaning of the user’s utterance.” Pet. Reply 21–22, 23 (citing Ex. 1018, 35:19–36:24). Mr. Occhiogrosso was asked : “[L]et’s say I have a *conversationally aware spreadsheet application* and I say what’s the value of the cell at Column A, Row 1, what part of Coffman’s system would handle actually looking up that value ?” — “*would it be the conversationally aware spreadsheet application* or something else ?” Ex. 1018, 35:19–36:17 (emphases added). Thus, in the posited scenario, the user has already selected the conversationally-aware application that is at issue. *See id.* Mr. Occhiogrosso did not testify that Coffman’s CVM dialog manager identifies a conversationally-aware application, utilizing a context stack or otherwise. *See id.* at 35:19–36:24.

Finally, Coffman discloses that each conversationally-aware application 11 can have its own local CVM and context stack. *See* Ex. 1003, Figs. 9 and 10, 26:28–27:2, 27:23–28 (distributed conversational applications 200a each have a local dialog manager 219a and a local context stacks 217a). However, Petitioner does not cite any disclosure in Coffman to suggest that the local CVM and context stack of an application 11 operate any differently from the central CVM 14 / 401 and its context stack 405 as discussed above, in relation to whether the local CVM might utilize its local context stack to identify another conversationally-aware application 11 to be invoked.

Thus, we conclude: (1) the arguments in the Petitioner Reply seeking to establish that Coffman’s processor(s) might be programmed to “identify” one or more conversationally-aware applications per claim 8 are untimely and should not be considered; and (2) even if they are considered, they do not satisfy Petitioner’s burden to demonstrate obviousness by a preponderance of the evidence.

(iii) Conclusion as to Claims 8 and 9

We conclude Petitioner has not shown by a preponderance of the evidence that claims 8 and 9 are unpatentable as having been obvious over Coffman.

5. Claims 14–18 of the ’652 Patent

Petitioner provides arguments and evidence in support of contending independent claim 14 and its dependent claims 15–18 are unpatentable as having been obvious over Coffman, including supporting testimony from Mr. Schmandt. *See* Pet. 59; Ex. 1002 ¶¶ 120–124.

As to claims 14–17, Patent Owner argues for patentability based solely on arguments considered above in the context of claim 1. *See, e.g.*, PO Resp. 8–10, 14–16, 32, 36. Therefore, Patent Owner has waived any other argument. *See* Inst. Dec. 40; Paper 12, 10. We adopt Petitioner’s arguments as to claims 14–17 having been obvious over Coffman as our own in this Decision. *See* Pet. 59; Ex. 1002 ¶¶ 120–123. Accordingly, we conclude a preponderance of the evidence establishes claims 14–17 are unpatentable as having been obvious over Coffman.

As to claim 18, Patent Owner relies on its arguments considered above in the context of claim 8, and Petitioner does likewise. *See, e.g.*, PO Resp. 3, 15–16, 37, 47; Pet. 59; Pet. Reply 1, 10; PO Sur-reply 18. Therefore, for the reasons provided above in the context of claim 8, we conclude a preponderance of the evidence does not establish claim 18 is unpatentable as having been obvious over Coffman.

6. *Claims 20–26 of the ’652 Patent*

Petitioner provides arguments and evidence in support of contending independent claim 20 and its dependent claims 21–26 are unpatentable as having been obvious over Coffman, including supporting testimony from Mr. Schmandt. *See* Pet. 59–60; Ex. 1002 ¶¶ 125–131.

As to claims 20–23, Patent Owner argues for patentability based solely on arguments considered above in the context of claim 1, with the term “context stack” in claim 1 treated identically to the term “context set” in claims 20–23. *See, e.g.*, PO Resp. 8–11, 14–16, 32, 36. Therefore, Patent Owner has waived any other argument. *See* Inst. Dec. 40; Paper 12, 10. We adopt Petitioner’s arguments as to claims 20–23 having been obvious over

Coffman as our own in this Decision. *See* Pet. 59–60; Ex. 1002 ¶¶ 125–128. Accordingly, we conclude a preponderance of the evidence establishes claims 20–23 are unpatentable as having been obvious over Coffman.

As to claims 24–26, Patent Owner relies on its arguments considered above in the context of claim 8, and Petitioner does likewise. *See, e.g.*, PO Resp. 3, 15–16, 37, 47; Pet. 60; Pet. Reply 1, 10; PO Sur-reply 18. Therefore, for the reasons provided above in the context of claim 8, we conclude a preponderance of the evidence does not establish claims 24–26 are unpatentable as having been obvious over Coffman.

7. *Conclusion as to Ground One*

We conclude Petitioner has demonstrated claims 1–7, 14–17, and 20–23 are unpatentable as having been obvious over Coffman, but has not demonstrated claims 8, 9, 18, and 24–26 are unpatentable based on this ground.

E. Ground Two — Obviousness over Coffman and Kennewick

In Ground Two, Petitioner asserts claims 10–13, 19, and 27 of the '652 patent are unpatentable as having been obvious over Coffman and Kennewick. *See* Pet. 13–14, 60–78. Petitioner provides arguments and evidence in support of this assertion, including testimony from Mr. Schmandt. *See id.* at 60–78; Ex. 1002 ¶¶ 132–164.

As to claims 10, 11, and 27, which depend from claim 8 and claim 24, respectively, Patent Owner relies on its arguments considered above in the context of Ground One and claim 8, and adds that Ground Two does not remedy the deficiencies of Ground One. *See* Ex. 1001, 39:63, 39:66, 42:53; PO Resp. 3, 14–16, 47. Petitioner replies by similarly grouping claims 10,

11, and 27 with its arguments for claim 8. *See* Pet. Reply 1, 10. Therefore, for the reasons provided above in the context of claim 8, we conclude a preponderance of the evidence does not establish claims 10, 11, and 27 are unpatentable as having been obvious over Coffman and Kennewick.

As to claims 12, 13, and 19, Patent Owner argues for patentability based solely on arguments considered above in the context of claim 1. *See, e.g.*, PO Resp. 14–16, 36. Therefore, Patent Owner has waived any other argument. *See* Inst. Dec. 40; Paper 12, 10. We adopt Petitioner’s arguments as to claims 12, 13, and 19 having been obvious over Coffman and Kennewick as our own in this Decision. *See* Pet. 69–78; Ex. 1002 ¶¶ 146–163. Accordingly, we conclude a preponderance of the evidence establishes claims 12, 13, and 19 are unpatentable as having been obvious over Coffman and Kennewick.

F. Ground Three — Obviousness over Coffman, Kennewick, and Lee

In Ground Three, Petitioner asserts claims 13 and 19 of the ’652 patent are unpatentable as having been obvious over Coffman, Kennewick, and Lee. *See* Pet. 13–14, 78–81. Petitioner provides arguments and evidence in support of this assertion, including testimony from Mr. Schmandt. *See id.* at 78–81; Ex. 1002 ¶¶ 165–170.

Patent Owner argues for patentability of claims 13 and 19 based solely on arguments considered above in the context of claim 1. *See, e.g.*, PO Resp. 14–16, 36. Therefore, Patent Owner has waived any other argument. *See* Inst. Dec. 40; Paper 12, 10. We adopt Petitioner’s arguments as to claims 13 and 19 having been obvious over Coffman, Kennewick, and Lee as our own in this Decision. *See* Pet. 78–81; Ex. 1002 ¶¶ 165–170.

Accordingly, we conclude a preponderance of the evidence establishes claims 13 and 19 are unpatentable as having been obvious over Coffman, Kennewick, and Lee.

IV. SUMMARY OF CONCLUSIONS

In summary, we determine a preponderance of the evidence establishes claims 1–7, 12–17, and 19–23 of the '652 patent are unpatentable, but does not establish that claims 8–11, 18, and 24–27 of the '652 patent are unpatentable, as shown in the following table:⁸

Claim(s)	35 U.S.C. §	Reference(s)/ Basis	Claim(s) Shown Unpatentable	Claim(s) Not Shown Unpatentable
1–9, 14–18, 20–26	103(a)	Coffman	1–7, 14–17, 20–23	8, 9, 18, 24–26
10–13, 19, 27	103(a)	Coffman, Kennewick	12, 13, 19	10, 11, 27
13, 19	103(a)	Coffman, Kennewick, Lee	13, 19	
Overall Outcome			1–7, 12–17, 19–23	8–11, 18, 24–27

⁸ 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 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).

V. ORDER

In consideration of the foregoing, it is hereby:

ORDERED that claims 1–7, 12–17, and 19–23 of the '652 patent have been proven by a preponderance of the evidence to be unpatentable;

FURTHER ORDERED that claims 8–11, 18, and 24–27 of the '652 patent have not been proven by a preponderance of the evidence to be unpatentable; and

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

IPR2024-00752
Patent 8,849,652 B2

FOR PETITIONER:

Heidi L. Keefe
Andrew C. Mace
Mark R. Weinstein
Reuben Chen
COOLEY LLP
hkeefe@cooley.com
amace@cooley.com
mweinstein@cooley.com
rchen@cooley.com

FOR PATENT OWNER:

Wayne M. Helge
James T. Wilson
BUNSOW DE MORY, LLP
whelge@bdiplaw.com
jwilson@bdiplaw.com