

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

GOOGLE LLC,
Petitioner,

v.

DIALECT, LLC,
Patent Owner.

Case IPR2024-00748
Patent 8,015,006

PATENT OWNER'S PRELIMINARY RESPONSE

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	THE ’006 PATENT DISCLOSES AND CLAIMS PROCESSING OF USER REQUESTS USING DOMAIN AGENTS THAT ARE NOT DISCLOSED OR SUGGESTED BY THE ASSERTED ART.....	4
A.	The ’006 Patent Invokes Domain Agents to Process User Requests and Commands	6
B.	The ’006 Patent Expressly Claims a Domain Agent “Processing” the Formulated User Request	9
III.	THE BOARD NEED NOT REACH ANY CLAIM CONSTRUCTION ISSUES TO DENY THE PETITION.....	11
IV.	PETITIONER’S CHALLENGES RELY SOLELY ON COFFMAN TO DISCLOSE THE CLAIMED “DOMAIN AGENT”	14
A.	Summary of Petitioner’s Challenges.....	14
B.	Coffman’s Non-Proprietary Speech Engine and Application Program Interfaces (APIs) Receive Speech Inputs from Conversationally-Aware Applications But Do Not Select or Invoke Conversationally-Aware Applications.....	16
C.	Coffman’s Context Stacks are Unique to Each Application.....	26
V.	PETITIONER’S GROUND 1 THEORY IMPROPERLY CHARACTERIZES COFFMAN’S CONVERSATIONALLY-AWARE APPLICATION AS THE CLAIMED “DOMAIN AGENT”	27
A.	Ground 1 Fails to Establish Obviousness of Claim 1	28
1.	Petitioner’s Ground 1 Theory for Claim 1 Misunderstands Coffman and Fails to Establish any “Domain Agent(s)” in Coffman.....	28
2.	Petitioner’s Ground 1 Fails to Demonstrate “associating the one or more recognized words or phrases and a	

	pronunciation ... with the determined identity and the request contained in the utterance in response to the one or more recognized words or phrases satisfying a predetermined confidence level”	36
3.	Ground 1 Fails to Establish Unpatentability of Claim 1	40
B.	Petitioner Also Fails to Demonstrate that Claims 4, 5, and 6 are Obvious Over Coffman in Combination with the Other Asserted Art in Grounds 2-4	40
1.	Petitioner Fails to Demonstrate that Claim 4 is Obvious Over Coffman in Combination with the Other Asserted Art in Ground 2.....	40
2.	Petitioner Fails to Demonstrate that Claim 5 is Obvious Over Coffman in Combination with the Other Asserted Art in Ground 3.....	41
3.	Petitioner Fails to Demonstrate that Claim 6 is Obvious Over Coffman in Combination with the Other Asserted Art in Ground 4.....	41
4.	Ground 2, 3, and 4 Also Fail to Establish Unpatentability of Any Challenged Claim.....	42
VI.	CONCLUSION.....	42

TABLE OF AUTHORITIES

Cases

<i>K/S Himpp v. Hear-Wear Techs., LLC</i> , 751 F.3d 1362 (Fed. Cir. 2014)	37
---	----

Statutes

35 U.S.C. § 311(b)	37
--------------------------	----

Rules

37 C.F.R. § 42.107	1
37 C.F.R. § 42.120(a).....	1

PATENT OWNER’S EXHIBIT LIST

Exhibit No.	Description
2001	Patent Owner’s Power of Attorney
2002	U.S. Patent 7,693,720 (“the ’720 Patent”)
2003	Reserved
2004	U.S. Patent 8,195,468 (“the ’468 Patent”)
2005	U.S. Patent 9,263,039 (“the ’039 Patent”)
2006	U.S. Patent 9,495,957 (“the ’957 Patent”)
2007	Order (Construing Claim Terms), Dialect, LLC v. Amazon.com Inc., Civ. No. 1:23cv581, Dkt. 213 (E.D. Va. Apr. 29, 2024)

Pursuant to 37 C.F.R. § 42.107, Patent Owner Dialect, LLC (“Dialect” or “Patent Owner”) files this preliminary response to the Petition, setting forth reasons why the Petition for *inter partes* review (“IPR”) of claims 1-6 and 9 of U.S. Patent No. 8,015,006 (the “’006 Patent”), as requested by Google LLC (“Petitioner”), should be denied.¹

I. INTRODUCTION

The ’006 Patent is entitled, “Systems and Methods for Processing Natural Language Speech Utterances with Context-Specific Domain Agents.” Central to the disclosed and claimed invention, the ’006 Patent discloses a system having domain agents, acting on behalf of other software or entities, that can receive, process, and respond to user requests initiated by speech commands. More specifically, the ’006 Patent discloses and claims that a domain agent associated with a determined context of a user utterance performs the step of processing the user’s request to generate a response to the utterance. According to the ’006 Patent, using a domain agent associated with the determined context can account “for the domain of expertise required, the context for the question or command, domain specific information, the history of the user’s interaction, user preferences,

¹ By submitting this Preliminary Response, no waiver of any argument is intended by Patent Owner. Patent Owner will have a right to file a complete response to the Petition, should the Board institute *inter partes* review. 37 C.F.R. § 42.120(a).

information sources or commands available, and responses obtained from the sources” in generating a response to a user’s request. ’006 Patent, 4:15-20.

The Petition relies upon a single reference—Coffman—to disclose the claimed “domain agent.” But Petitioner incorrectly identifies Coffman’s “conversationally-aware application” as a “domain agent” without properly accounting for the interaction between Coffman’s conversationally-aware applications, which serve as the user-facing frontend software of the system, and Coffman’s Conversational Virtual Machine, or “CVM,” which stores prior dialogues with a user in an application-specific context stack on the system backend. Petitioner would have the Board believe that Coffman’s CVM receives a user request via a conversationally-aware application, searches a context stack for the active context between the user and the conversationally-aware application, and then selects the conversationally-aware application to carry out the user’s request. This is not how Coffman operates.

Fundamentally, Coffman provides a speech engine, including context stacks a dialog manager, and conversational sub-systems, that can be accessed by a conversationally-aware application without requiring a developer to build its own proprietary speech engine for every application. Thus, in Coffman, when a user begins its interaction with a conversationally-aware application, the conversationally-aware application accesses Coffman’s CVM via a

conversationally-aware application protocol interface (“API”). Upon receiving the user’s utterance, the CVM will search a stored and application-specific context stack for a context—or discourse—that matches the user’s utterance, and will treat that context as the active context for that conversationally-aware application.

Where the system includes many conversationally-aware applications providing different user interfaces for receiving user utterances and each accessing the CVM, each application has its own context stack. As a result, Coffman’s CVM may receive user utterances, requests, or commands from various applications, but each are handled by Coffman’s CVM by searching application-specific context stacks. The CVM also possesses conversational sub-systems to process the requests and return responses to the conversationally-aware applications. Those responses will be presented to the user by the conversationally-aware applications. Thus, in Coffman, users interact with conversationally-aware applications via speech, but the conversationally-aware applications rely on the CVM backend to process requests, commands, and queries from a user.

On the other hand, the ’006 Patent’s system receives user utterances through a common user interface, determines a meaning and context of the user utterance, and then selects a domain agent based on the determined meaning of the utterance to process the utterance’s request, command, or query. Unlike Coffman, where the context stack to be searched depends on the conversationally-aware application

initially accessed by the user, the '006 Patent first determines a meaning of the user's utterance, and then selects the appropriate domain agent to process the utterance's request based on the determined meaning. The invoked domain agent generates results to the formulated request, and a speech unit in the '006 Patent presents results of the processed request to the user.

While Petitioner attempts to conflate Coffman's conversationally-aware application with the '006 Patent's domain agent, Coffman's conversationally-aware application is more akin to a speech interface with a user. This limited role of a conversationally-aware application in Coffman fails to satisfy every claim requirement of a "domain agent" in challenged claim 1. For this reason, Petitioner's challenge of claim 1 and all the dependent challenged claims must fail.

II. THE '006 PATENT DISCLOSES AND CLAIMS PROCESSING OF USER REQUESTS USING DOMAIN AGENTS THAT ARE NOT DISCLOSED OR SUGGESTED BY THE ASSERTED ART

The '006 Patent discloses and claims a system to process and respond to a user generated natural language speech utterance including, in relevant part, processing the formulated request with a domain agent associated with a determined context to generate a response to the user's utterance. FIG. 1 of the '006 Patent, reproduced below, is an overall diagrammatic view of an embodiment of the '006 Patent:

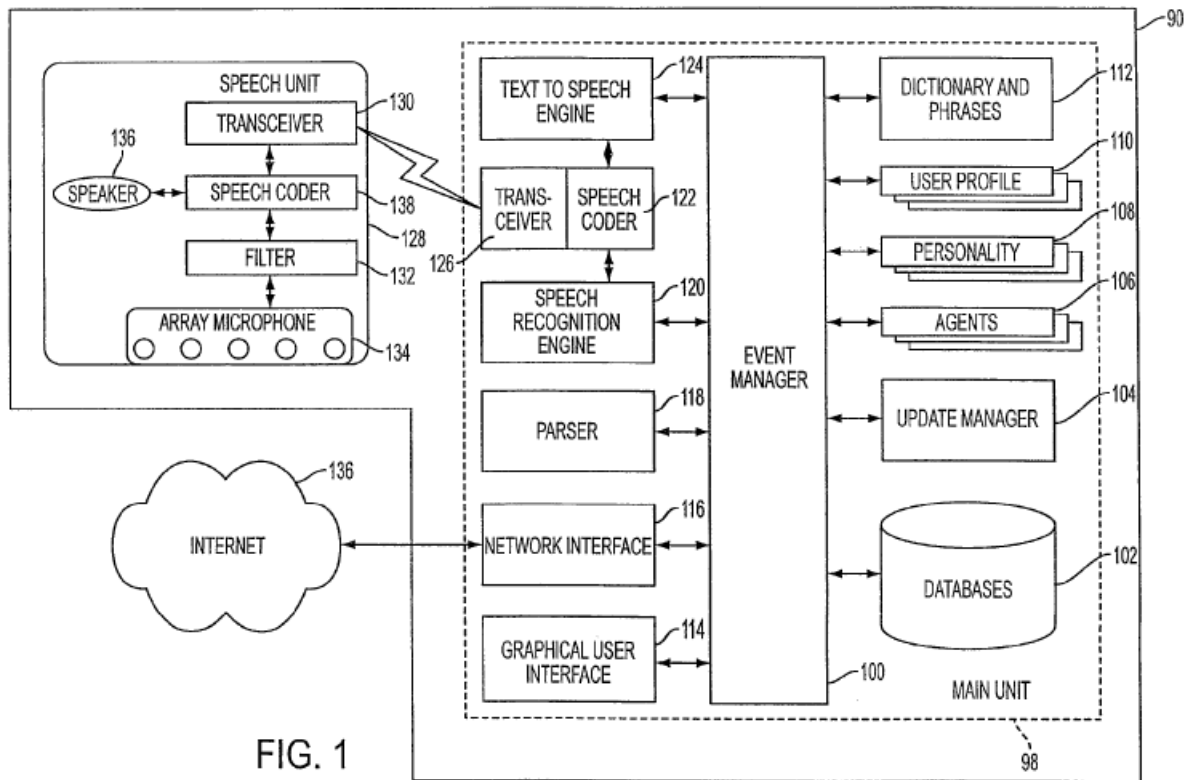


FIG. 1

'006 Patent, FIG. 1, 8:54-55. As shown in the figure above, the system includes, among other things, speech unit 128 and main unit 98, transceivers 126 and 130, speech recognition engine 120, dictionary and phrases module 112, parser 118, and event manager 100. *Id.*, 10:6-11:4. The system also includes multiple agents 106, discussed in more detail below. *Id.*, 11:28-51. This system is equipped to receive, process, and respond to requests and commands conveyed by user speech. As shown, the “speech unit 128 includes one or more microphones, for example array microphone 134, to receive the utterances of the user.” *Id.*, 10:30-32. The speech coder 138 in the speech unit 128 may compress and encode the user’s speech, and

the speech coder in the main unit 98 may decode and decompress the speech. *Id.*, 10:32-39, 10:56-58.

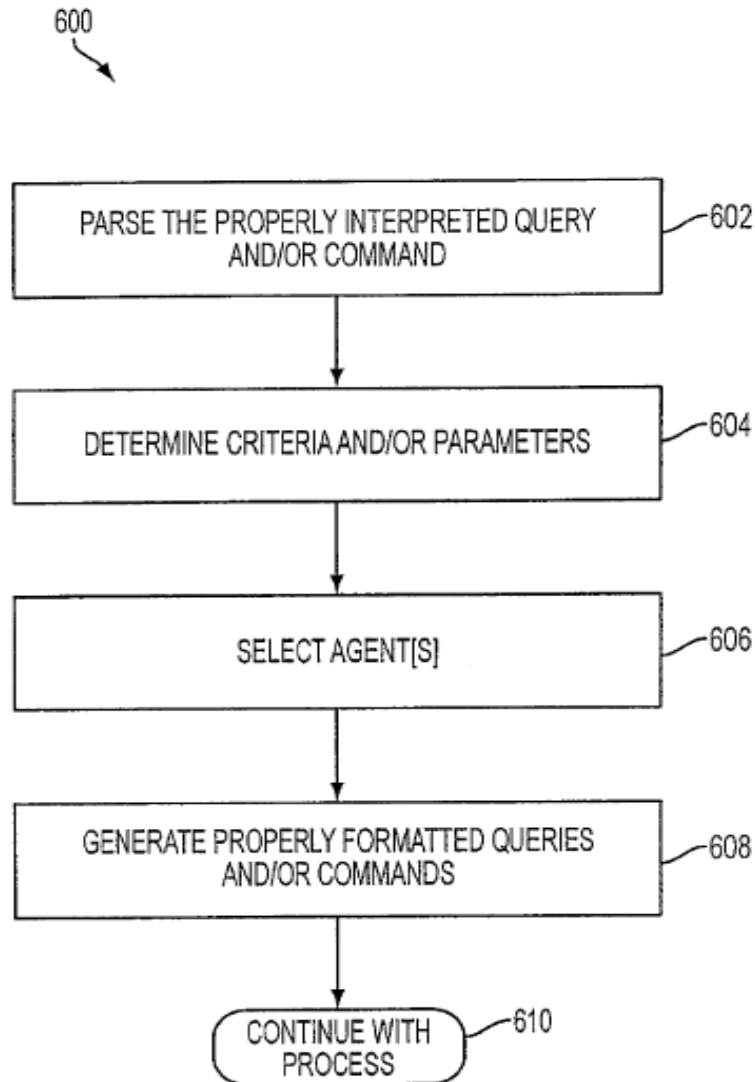
A. The '006 Patent Invokes Domain Agents to Process User Requests and Commands

The '006 Patent's system 90 includes different types of agents 106. *Id.*, 11:2-510. The agents 106 “contain packages of both generic and domain specific behavior for the system 98,” and play a critical role in the efficient processing of the speech received at the main unit. *Id.*, 10:15-27. As shown in the exemplary agent architecture of FIG. 2, there are domain agents 156, a system agent 150, and an agent library 158 included within FIG. 1's “agents 106.” *Id.*, 11:52-12:10. “A system agent ... provides default functionality and basic services.” *Id.*, 11:31-32. “[G]eneric and domain specific behavior and information may be organized into domain agents.” *Id.*, 11:29-31.

Describing domain agents 156 in more detail, the '006 Patent discloses that “domain specific agents provide complete, convenient and redistributable packages or modules for each application area. In other words, a domain agent includes everything needed to extend or modify the functionality of the system 90 in a current or new domain.” *Id.*, 11:32-37. “New commands, keywords, information, or information sources can be added to any domain agent 156 by changing agent data or scripting.” *Id.*, 13:9-15.

Speech “is processed by the speech recognition engine 120 using data in the dictionary and phrases module 112 and received from the agents 106.” *Id.*, 10:58-60. “The recognized words and phrases are processed by the parser 118, which transforms them into complete commands and questions using data supplied by an agent 106.” *Id.*, 10:61-63. “Based on keywords in the questions and commands and the structures of the questions and commands, the parser invokes the required agent[s].” *Id.*, 11:54-57.

In particular, the parser may invoke a domain agent 156 among the agents 106. Specifically, the “parser 118 determines a context for an utterance by applying prior probabilities or fuzzy possibilities to keyword matching, user profile 110, and dialog history. ***The context of a question or command determines the domain and thereby, the domain agent 156, if any, to be [in]voked.***” *Id.*, 17:49-53 (emphasis added). The process of selecting a “proper” domain agent to invoke is illustrated in FIG. 6 of the '006 Patent, inserted below. *Id.*, 9:3-6.



As disclosed, “[s]pecific questions are generally directed to one of the domain agents 156. Based on the question or context and the parameters or criteria, the domain agent creates one or more queries to one or more local or external information sources.” *Id.*, 19:8-11. For example, the agent 106 may “create queries to local databases 102 or th[r]ough the network interface 116 to data

sources on the Internet 136 or other networks.” *Id.*, 10:64-67. In this way, the “agent 106 then process[es] the commands or questions.” *Id.*, 10:63-64.

Further, the “agents 106 return results of questions as responses to users.” *Id.*, 11:5-6. “The agents 106 create a response string, which is sent to the text to speech engine 124;” this engine 124 “creates the required utterances, which may be encoded and compressed by the speech coder 122.” *Id.*, 11:9-13. “Once coded, the utterances are transmitted from the main unit 98 [to] the speech unit 128. The utterance is then decoded and decompressed by the speech coder 138 and output by the speaker 136” to the user. *Id.*, 11:13-17.

B. The '006 Patent Expressly Claims a Domain Agent “Processing” the Formulated User Request

Consistent with the '006 Patent's disclosure introduced above, independent claim 1 of the '006 Patent recites the step of “processing the formulated request with the domain agent associated with the determined context to generate a response to the utterance.” This feature appears in substantially similar format in independent claims 4, 5, 6. *See, e.g.*, '006 Patent, 26:17-19, 27:9-11, 27:57-59, 28:8-10.

Additionally, claim 1 includes the steps of “determining an identity associated with a user that spoke the utterance based on voice characteristics associated with the utterance; and associating the one or more recognized words or phrases *and a pronunciation* associated with the one or more recognized words or

phrases with the determined identity” *Id.*, 25:63-26:4 (emphasis added).

Although these features only appear in claim 1, Petitioner's failure to establish these features in the asserted art also dooms Petitioner's Ground 1 challenge.

In full, claim 1 recites:

1. A method for processing natural language speech utterances with context-specific domain agents, comprising:

receiving, at a speech unit coupled to a processing device, a natural language speech utterance that contains a request;

recognizing, at a speech recognition engine coupled to the processing device, one or more words or phrases contained in the utterance using information in one or more dictionary and phrase tables, wherein recognizing the one or more words or phrases contained in the utterance includes:

dynamically updating the information in the one or more dictionary and phrase tables based on a dynamic set of prior probabilities or fuzzy possibilities;

determining an identity associated with a user that spoke the utterance based on voice characteristics associated with the utterance; and

associating the one or more recognized words or phrases and a pronunciation associated with the one or more recognized words or phrases with the determined identity and the request contained in the utterance in response to the one or more recognized words or phrases satisfying a predetermined confidence level;

parsing, at a parser coupled to the processing device, the one or more recognized words or phrases to determine a meaning associated with the utterance and a context associated with the request contained in the utterance, wherein the one or more recognized words or phrases are further associated with the determined context in response to the one or more recognized words or phrases satisfying the predetermined confidence level; formulating, at the parser, the request contained in the utterance in accordance with a grammar used by a domain agent associated with the determined context;

processing the formulated request with the domain agent associated with the determined context to generate a response to the utterance; and

presenting the generated response to the utterance via the speech unit.

'006 Patent, claim 1 (emphasis added). As discussed below, Petitioner's challenges fail to identify any "domain agent" in the asserted combinations that satisfies the "processing" claim requirements in each challenged independent claim.

III. THE BOARD NEED NOT REACH ANY CLAIM CONSTRUCTION ISSUES TO DENY THE PETITION

The Petitioner identifies no terms as requiring constructions. The United States District Court for the Eastern District of Virginia has issued constructions for the following terms (Ex. 2007):

Term	Patent(s)	Claim(s)	Construction
context	'720 Patent	1, 31	"the subject matter area to which a particular user input is directed and which is used to determine the meaning of the user input"
	'006 Patent	1, 2, 4, 5, 10	
	'468 Patent	19, 28	
	'039 Patent	17, 18	
context stack	'006 Patent	2	"a data structure that stores or references contexts in a manner that can be ordered"
	'468 Patent	19, 28	
	'039 Patent	17	
	'957 Patent	1, 3, 4, 7	
entries in a context stack	'468 Patent	19	"elements of a context stack that store or reference contexts"
context entries	'957 Patent	1, 3, 4, 5, 7	"elements of a context stack that store or reference contexts"
grammar	'720 Patent	1, 31	"a set of rules governing how inputs and requests to a domain agent should be structured"
	'006 Patent	1, 5, 10	

context description grammar	'039 Patent	13	"a data structure containing entries constituting or referencing sets of rules, wherein each of those sets describes the structure of natural language in a particular context"
domain agent	'720 Patent	1, 31	"executables associated with a specific domain that receive, process and respond to user questions, queries and commands"
	'006 Patent	1, 2, 5, 10	
	'468 Patent	19, 30, 32	"software with domain-specific behavior and information"
	'039 Patent	13, 14	
procedures sensitive to the determined context	'006 Patent	5	"procedures that differ based on which context has been determined"
parser	'720 Patent	1, 35	"software that analyzes a string of words"
	'006 Patent	1, 5, 10	

In the table above, the '006 Patent corresponds to the challenged patent. Ex. 1001. U.S. Patent 7,693,720 ("the '720 Patent") (Ex. 2002), U.S. Patent 8,195,468 ("the '468 Patent") (Ex. 2004), U.S. Patent 9,263,039 ("the '039 Patent") (Ex. 2005), and U.S. Patent 9,495,957 ("the '957 Patent") (Ex. 2006) are each assigned to Patent Owner and share at least one inventor in common with the '006 Patent.

Without waiving its right to submit constructions in a Patent Owner Response, Patent Owner submits that the Board need not engage in any pre-institution claim construction. For the reasons presented in Section III below, Petitioner’s grounds include fundamental flaws and fail to establish the disclosure of every claimed element in at least independent claims 1, 4, 5, and 6. These findings do not require the Board to conduct any construction of any claim terms.

IV. PETITIONER’S CHALLENGES RELY SOLELY ON COFFMAN TO DISCLOSE THE CLAIMED “DOMAIN AGENT”

A. Summary of Petitioner’s Challenges

Petitioner asserts four Grounds in its Petition.

Ground 1 asserts that claims 1-3 are obvious in view of Coffman,² Kanevsky,³ Kneser,⁴ and Zadrozny.⁵ Collectively, Coffman, Kanevsky, Kneser, and Zadrozny are referred to in the Petition as “Ground 1 Prior Art.” Pet., 18.

Ground 2 asserts that claim 4 is obvious in view of the Ground 1 Prior Art further in view of Maes ’101⁶ and Franceschi.⁷ Pet., 18.

² Ex. 1003, International Patent Application Publication No. WO 00/20962 (“Coffman”).

³ Ex. 1004, U.S. Patent No. 5,897,616 (“Kanevsky”).

⁴ Ex. 1005, U.S. Patent No. 6,157,912 (“Kneser”).

⁵ Ex. 1006, “Natural Language Understanding with a Grammar of Constructions,” Proc. 15th Int’l Conf. on Computational Linguistics (Vol. 2) (1994) (“Zadrozny”).

⁶ Ex. 1010, U.S. Patent No. 6,073,101 (“Maes ’101”).

⁷ Ex. 1011, U.S. Patent No. 6,321,196 (“Franceschi”).

Ground 3 asserts that claim 5 is obvious in view of the Ground 1 Prior Art, further in view of Chai.⁸ Pet., 18.

Ground 4 asserts that claims 6 and 9 are obvious in view of the Ground 1 Prior Art, further in view of Redfern.⁹ Pet., 18.

In each Ground, Petitioner relies upon a conversationally-aware application from Coffman to disclose the claimed “domain agent” in claims 1, 4, 5, and 6. But this assertion misunderstands Coffman, because Coffman’s conversationally-aware applications cannot satisfy each claimed requirement of a “domain agent” as recited in these challenged claims. As discussed in the Petition, Petitioner suggests that Coffman’s Conversational Virtual Machine, or “CVM,” selects one or more conversationally-aware applications to carry out a process based on a context or meaning of a user’s utterance. This is incorrect, and fails to recognize that Coffman’s conversationally-aware applications allow for speech commands or requests from users, and provide speech output responses back to a user—they are the frontend of the system. In this way, they are unlike the ’006 Patent’s domain agents, which are selected to process a request only after a user input is parsed and the proper context of an utterance is determined. Coffman’s conversationally-aware applications rely on the CVM to process a user request. The

⁸ Ex. 1012, U.S. Patent No. 6,829,603 (“Chai”).

⁹ Ex. 1013, U.S. Patent No. 6,078,914 (“Redfern”).

conversationally-aware application simply engages in a dialog with a user; this is what makes the application “conversationally-aware” and distinguishes it from a conventional application. As will be explained, Coffman’s conversationally-aware applications do not process a user’s request contained in an utterance.

Moreover, as noted in the Petition, Coffman’s issued U.S. Patent (U.S. Patent No. 7,137,126) was known to the Examiner. Pet., 18; *see also* Ex. 1001, 3 (citing U.S. Patent No. 7,137,126 among the ’006 Patent’s References Cited). The Examiner’s allowance of the ’006 Patent over Coffman was proper for at least the reasons discussed below, and should not be revisited in this proceeding.

B. Coffman’s Non-Proprietary Speech Engine and Application Program Interfaces (APIs) Receive Speech Inputs from Conversationally-Aware Applications But Do Not Select or Invoke Conversationally-Aware Applications

As a background to Coffman’s disclosure, Coffman was seeking to provide a backend system that would more easily allow a developer to program a voice-based interactive telephone application to converse with a user, without having to build an entire backend speech engine or API. Coffman notes that “voice command and control (voice C&C) UIs [user interfaces] are emerging everywhere computers are used.” Coffman, 3:1-2. “For instance, telephone companies, call centers and IVR have implemented speech interfaces to automate certain tasks, reduce their operator requirements and operating costs and speed-up call processing.” Coffman, 3:5-7. However, Coffman also notes that interactive voice

response (“IVR”) “application developers offer their own proprietary speech engines and APIs (application program interface),” and that “proprietary applications are typically not portable from vendor to vendor.” Coffman, 3:7-8.

Coffman also states that “speech interfaces for GUI based OSs have been implemented using commercially available continuous speech recognition applications for dictation and command and control.” Coffman, 3:12-14. Coffman describes these “applications” as “add-ons to the GUI based OSs in the sense that such applications allow for the replacement of keyboard and mouse and allows a user to change the focus, launch new tasks, and give voice commands to the task in focus.” Coffman, 3:14-16.

As explained, Coffman’s purpose is to provide a “conversational virtual machine” with a speech engine and application program interfaces (APIs) usable by application developers “for implementing conversational computing” with conventional applications and conversationally-aware applications. Coffman, 1:7-17. The point of Coffman, therefore, is to provide a “universal conversational user interface.” Coffman, 3:30.

Coffman’s Conversational Virtual Machine, or “CVM,” “utilizes conversational subsystems (which may be local or distributed) including speech recognition, speaker recognition, text-to-speech, natural language understanding and natural dialog generation engines to understand and generate dialog between

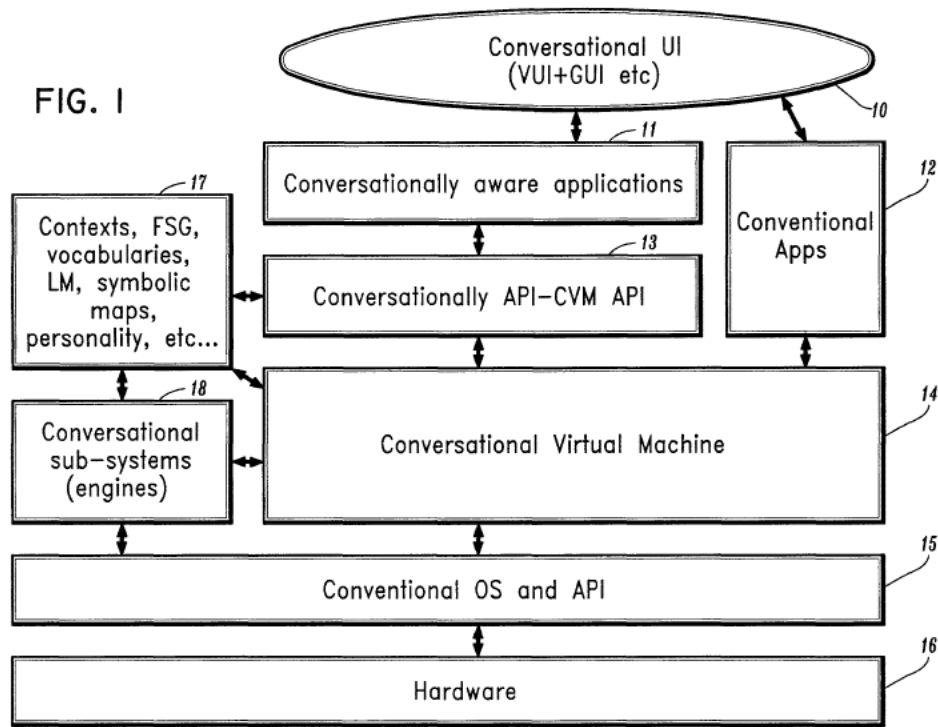
... user and machine.” Coffman, 5:1-4. Coffman refers to these aspects of the CVM as a “subsystem,” and notes that the “subsystem [is] accessed through the CVM.” Coffman, 5:4-5. As an aid to software developers, Coffman explains that the “development environment offered by the CVM ... allows a developer to build, simulate and debug conversational aware application [*sic*] for CVM.” Coffman, 18:16-18. An application developer does not need to build a speech engine or APIs, and instead accesses the CVM through the system’s established APIs. The system’s speech “engines are hidden to the application through the conversational application APIs.” Coffman, 5:5-6.

The relationship of Coffman’s CVM to conventional applications, conversationally-aware applications, and the APIs is illustrated in Coffman’s FIG.

1. Coffman’s FIG. 1 is described as a “block diagram of a conversational computing system according to an embodiment of the present invention.”

Coffman, 9:27-28.

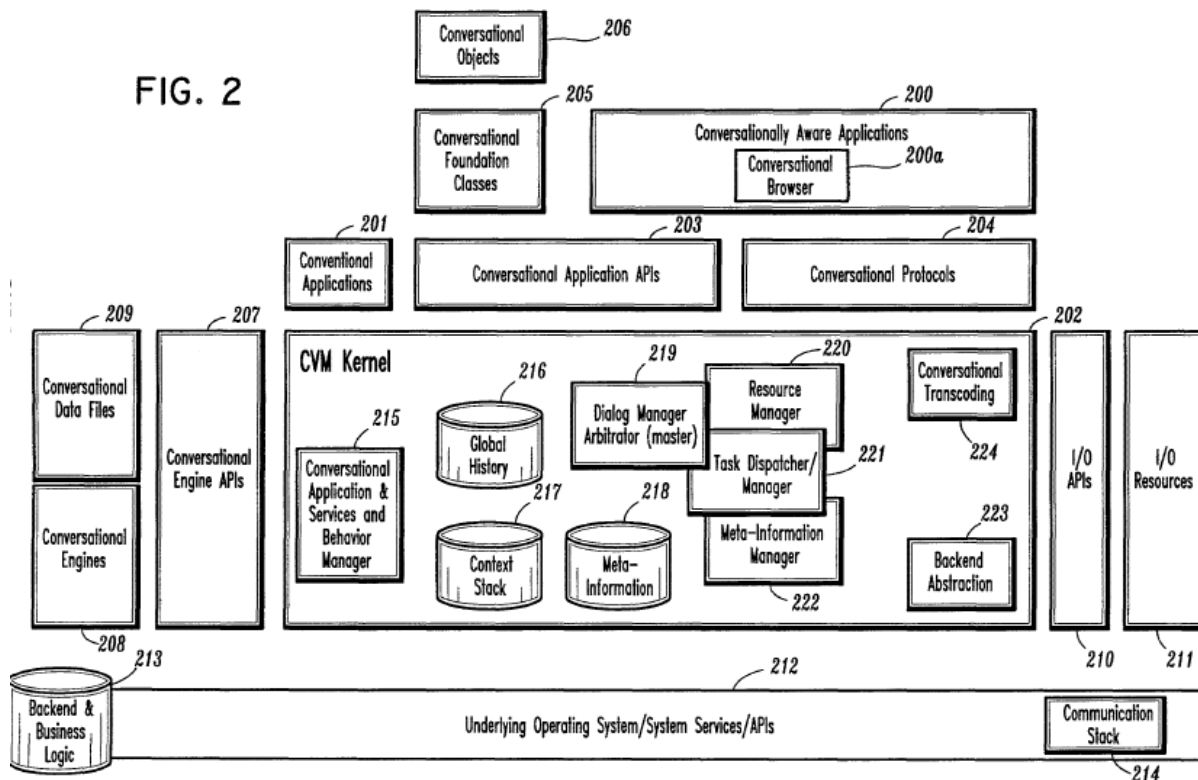
FIG. 1



Per Coffman, the CVM system “comprises a plurality of applications including conversationally aware applications 11 (i.e., applications that ‘speak’ conversational protocols) and conventional applications 12.” Coffman, 12:26-28. The difference between these types of applications, per Coffman, is that “conversationally aware applications 11 are applications that are specifically programmed for operating with a CVM core layer (or kernel) 14 via conversational application APIs 13.” *Id.*, 12:28-30. Conventional applications interact with the CVM, but do not need to do so through the conversational application APIs 13. Thus, for both conventional applications and conversationally-aware applications, the “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 user interface ... to provide conversational system behaviors.” *Id.*, 12:30-13:1. In particular, Coffman’s conversationally-aware applications 11 are directly connected to the User Interface 10 to receive a user’s utterance, and interface with the CVM 14 via the conversational application API.

Indeed, the conversational application API serves as an intermediary between the conversationally-aware applications on the frontend, and the backend system including CVM 14, conversational sub-systems 18, and backend logic. The CVM and its components are illustrated and described in more detail with respect to Coffman’s FIG. 2, reproduced below. The CVM Kernel 202 in FIG. 2 corresponds to the CVM Kernel 14 illustrated in FIG. 1.



Relevant to the current Petition, the CVM Kernel 202 includes a Context Stack 217, a Dialog Manager Arbitrator (master) 219 (which is frequently referred to in Coffman as the Dialog Manager 219), and a Backend Abstraction 223. There is also a “conversational application and behavior/service manager layer 215,” which “comprises functions for managing the conventional and conversationally aware applications 200 and 201.” *Id.*, 19:30-31. Coffman describes these management functions as including “for example, keeping track of which applications are registered (both local and network-distributed), what are the dialog interfaces (if any) of the applications, and what is the state of each application.” *Id.*, 19:31-20:2.

Describing the context stack 405, Coffman discloses that “[t]he core of the CVM 401 is the context stack 405 which operates and is managed under the control of the dialog controller 404.” *Id.*, 29:19-20. Coffman defines the term “context” to refer “to the state of each discourse (whether active or nonnative), which keeps track of the past history of the discourse, its current state, and the specific characteristics and full query arguments of the corresponding task (e.g, vocabulary file, language model, parsing, tags, voiceprint, TTS rules, grammar, NLU etc. of each active task/process) along with any data needed for input understanding.” *Id.*, 29:27-32.

The application-specific “context stack 217 is managed by the dialog manager 219.” *Id.*, 21:23; *see also id.*, 27:1-2 (disclosing that each application will have its “own ... context stack ...”). “More specifically, each new task, process, or thread creates a new stack entry and is associated with a discourse.” *Id.*, 30:4-5. “Each context associated with a given discourse comprises the latest requests made to the corresponding process/task/thread as well as the latest output.” *Id.*, 30:6-8.

The dialog manager 219 (or dialog controller 404) “manages the context stack 405 by creating a new stack entry in the context stack 405 for each new task/process/thread that is spawned either local or remotely from a networked device.” *Id.*, 30:13-15. “The context stack 405 is associated with the machine state stack so that for any new input from a user, the dialog controller 404 may

traverse the context stack 405 until the input context can be appropriately established. This essentially amounts to finding and selecting the active discourse between the user and machine among the last and past discourses.” *Id.*, 30:22-25; *see also id.*, 41:9-13.

“Each application may be associated with multiple discourses (e.g. the application management discourse and the application content navigation discourses).” *Id.*, 29:4-8. An application’s context stack 217 “comprises all the information associated with an application.” *Id.*, 21:23-24. “Such information includes all the variable, states, input, output and queries *to the backend* that are performed in the context of the dialog and any extraneous event that occurs during the dialog.” *Id.*, 21:24-26 (emphasis added). The CVM Kernel 202 also includes a “global history 216,” which “includes information that is stored beyond the context of each application,” such as “the information that is associated with all the applications and actions [taken] during a conversational session (i.e., the history of the dialog between user and machine for a current session (or from when the machine was activated).” *Id.*, 21:28-32.

Importantly, Coffman’s mention in this passage of “queries to the backend” is explained further by the inclusion of Coffman’s “backend abstraction layer 223” in the CVM Kernel 202. The “backend abstraction layer 223 ... allows access to backend business logic 213 via the dialog manager 219 (rather than bypassing the

dialog manager 219).” Coffman, 22:1-3. The “backend abstraction layer 223 can translate input and output to and from the dialog manager 219 to database queries. This layer 223 will convert standardized attribute value n-uples into database queries and translate the result of such queries into tables or sets of attribute value n-uples back to the dialog manager 219.” *Id.*, 22:4-7. As such, Coffman’s backend can process queries formulated at the CVM, and provide the results back to the CVM for communicating them to a conversationally-aware application, which presents the results of the processed query to the user.

This understanding of Coffman is also supported by FIG. 4. FIG. 4 of Coffman, reproduced below, provides a block diagram of Coffman’s system 400 and CVM 401 (Coffman, FIG. 4 and at 28:10-12):

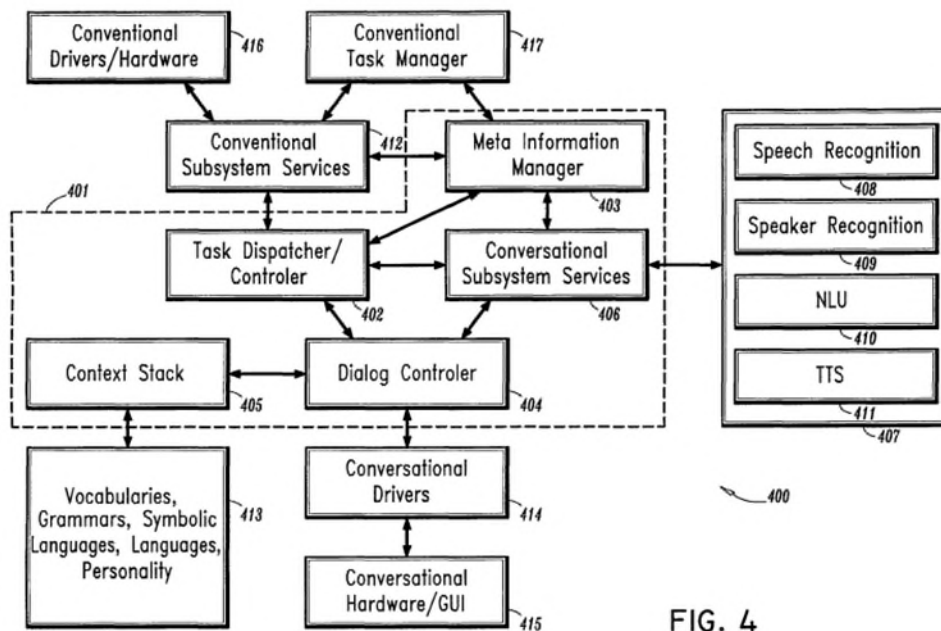


FIG. 4

The conversational system 400 comprises a combination of conventional subsystems and conversational subsystems which are executed and managed by a CVM 401. The CVM 401 comprises a task dispatcher/controller 402, a meta information manager 403, a dialog controller/manager 404, a context stack 405, and a conversational subsystem services manager 406. The “CVM controller” of Coffman is understood to refer collectively to the task dispatcher/controller 402 and the dialog controller 404. In CVM 401, “instructions/commands that are initiated by the user are explicitly *sent to the background* of the system.” *Id.*, 43:1-2 (emphasis added).

This disclosure is critical to a proper understanding of Coffman's CVM, and how it relates to the conversationally-aware applications disclosed in Coffman. In particular, a user who interacts with a conversationally-aware application on the front end of Coffman's system will communicate with the conversationally-aware application via speech, and the conversationally-aware application will access the CVM layer (or kernel) via conversational application APIs so that the CVM can perform the backend operations. And, as noted, Coffman's CVM and backend operations are hidden from the conversationally-aware applications. *See* Coffman, 5:5-6, 29:10-11. As disclosed by Coffman: “conversational sub-systems 18 are responsible for converting voice requests into queries and converting outputs and results into spoken messages using the appropriate data files 17 (e.g., contexts,

finite state grammars, vocabularies, language models, symbolic query maps etc.).”

Coffman, 13:8-10.

C. Coffman's Context Stacks are Unique to Each Application

Coffman discloses that a CVM may interact with multiple applications.

Coffman discloses that the CVM provides applications and devices with “conversational services and behaviors as well as conversational protocols for interaction with *multiple applications* and devices also equipped with a CVM layer, or at least, conversationally aware.” Coffman, 11:2-4 (emphasis added). However, Coffman discloses that where there are different applications, “different devices, CVMs or different applications will have their own dialog manager, context stack 217, 217a and global history 218, 218a.” Coffman, 27:1-2.

A context stack for an application is further described with respect to FIG. 4's context stack 405, “which operates and is managed under control of the dialog controller 404.” *Id.*, 29:18-19. The context stack 405 “accumulates the context (i.e., full query arguments list of attribute value n-uples, and state/mode) of each active process with an activated discourse (i.e., conversational interaction associated with a given task/process/thread) along with any data files 413 (or at least identifiers of such conversational arguments) for the different engines that may be needed for input understanding (e.g., files or arguments that the engines use for performing their respective tasks such as active FSG, topic, vocabulary,

HMM (hidden markov models), voiceprints, language models or possible queries for a speech input).” *Id.*, 29:21-27. As such, a search of an application’s own context stack 217a may identify an active context and active process for that application. But this search does not support Petitioner’s assertion that a CVM would then select a conversationally-aware application to process a user’s request based on the active context.

V. PETITIONER’S GROUND 1 THEORY IMPROPERLY CHARACTERIZES COFFMAN’S CONVERSATIONALLY-AWARE APPLICATION AS THE CLAIMED “DOMAIN AGENT”

Ground 1 is facially deficient in establishing the unpatentability of any challenged claim. In particular, Petitioner’s challenge incorrectly looks to Coffman’s conversationally-aware application to disclose the claimed “domain agent” of the challenged claims. Petitioner’s characterization of Coffman’s conversationally-aware application as a “domain agent” is unsupported by Coffman, and overlooks that Coffman’s conversationally-aware applications are merely gateways to use the system, providing an interface (UI) for a user to provide the speech-based content for a given conversationally-aware application to interact with users. In response to a particular conversationally-aware application making an API call containing a user’s utterance, Coffman’s CVM activates a processing thread that accesses a context stack specific to that 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. Thus, a conversationally-aware application in Coffman accesses the tools provided by Coffman’s conversational virtual machine (“CVM”) via Coffman’s application program interfaces (APIs), and awaits a response from the CVM to continue a communication with a user. That is all that Coffman’s conversationally-aware applications do, and it is not at all comparable to what the claimed “domain agent” is required to do when claim 1 is viewed in its entirety.

A. Ground 1 Fails to Establish Obviousness of Claim 1

1. Petitioner’s Ground 1 Theory for Claim 1 Misunderstands Coffman and Fails to Establish any “Domain Agent(s)” in Coffman

Petitioner’s fundamental misunderstanding of Coffman undercuts Petitioner’s allegations regarding how Coffman allegedly discloses the claimed “domain agent.” As noted above, claim 1 recites, in relevant part:

1. A method for processing natural language speech utterances with *context-specific domain agents*, comprising:

...

processing the formulated request with the domain agent associated with the determined context to generate a response to the utterance;

Petitioner asserts that Coffman’s “conversationally-aware application executable is a ‘domain agent.’” Pet., 43 (citing Schmandt, ¶121) (emphasis in original). Petitioner also cites to Coffman to allege that “an application is

automatically *selected* and carries out a request as a result of processing by the conversational system.” Pet., 44 (citing Coffman, 37:2-4, 42:1-7; Schmandt ¶125) (emphasis added). Further, Petitioner contends that “Coffman’s conversationally-aware application receives, processes, and responds to user questions, queries, and commands.” Pet., 42 (citing Coffman, 13:8-13, 37:2-4, 40:9-19). None of these citations support Petitioner’s assertions regarding Coffman’s conversationally-aware applications.

Addressing Coffman’s disclosure at 42:1-7 first, Petitioner’s argument for the processing of a user request by a conversationally-aware application appears to be founded on Petitioner’s unsupported conflation of the “process” described with respect to Coffman’s context stack with the conversationally-aware applications disclosed in Coffman.¹⁰ These two things are different elements of Coffman, and Coffman does not support Petitioner’s characterization of a conversationally-aware

¹⁰ The Petition attempts a similar misleading slight-of-hand while discussing the claim 1’s recitation of “context-specific domain agents.” Pet., 45. Petitioner asserts that “Coffman describes a ‘context stack’ where the state and history of each active process, such as an application process, is tracked in its own ‘context.’” *Id.* The slight-of-hand maneuver is Petitioner’s treatment of the “active process,” described with respect to Coffman’s context stack, as identical to Coffman’s conversationally-aware applications. That characterization is never explained or supported in the Petition.

application as being the same as a process. Relevant to this point, Coffman's disclosure at 42:1-7 is reproduced below:

Completion/search for the active context is performed from context to context down the stack. That is, new queries or arguments are compared by the dialog engine by going down the stack until an acceptable match is obtained and optionally confirmation is obtained from the user. As soon a context is found that fits the utterance at the level of the NLU symbolic language, the *context becomes active and the corresponding process becomes active*. Until the active command is completed, or until a new command is provided, the selected context is marked active, and pushed to the top of the context stack 405.

Coffman, 42:1-7 (emphasis added). Contrary to Petitioner's assumption, Coffman does not disclose that the "process" in the passage above occurs at a "conversationally-aware application," and does not disclose that activating a process is the same as selecting a conversationally-aware application. Indeed, Coffman associates a "process" with tasks and threads performed *at the CVM*. See Coffman, 42:16-17 ("This stack structure allows to maintain non-ambiguous conversational interactions with multiple tasks, threads or processes."); 21:1-3 ("The task dispatcher/manager 221 dispatches and coordinates different tasks and processes that are spawned (by the user and machine) on local and networked conventional and conversational resources"); 30:1-2 ("context stack (as well as

other CVM components)"); 30:4-5 (“[E]ach new task, process, or thread creates a new stack entry and is associated with a discourse.”); FIG. 2 (illustrating CVM components including task dispatcher/manager 221).

Nothing about this disclosure suggests that the CVM activates or selects a conversationally-aware application for processing a user's request. To the contrary, Coffman expressly discloses that the task dispatcher/controller 402 of the CVM 401 “dispatches and coordinates different tasks and processes that are spawned” Coffman, 30:26-28. Nothing in Coffman's discussion of the task dispatcher/controller 402 suggests or discloses activating or selecting a conversationally-aware application to process a user's request based on a match in the context stack, or even that the task dispatcher/controller 402 is capable of selecting a conversationally-aware application. Instead, Coffman selects a context stack based on the user's interaction with a conversationally-aware application.

Moreover, Coffman expressly discloses that the task dispatcher/controller 402 of the CVM 401 “dispatches and coordinates different tasks and processes that are spawned” Coffman, 30:26-28. Additionally, “all conversational subsystem calls are hidden to the application” Coffman, 29:10-11; *see also id.*, 5:5-6. The CVM 401, not any application, “will locate the requested conversational subsystem 407 (via the conversational subsystem services manager 406), drive its execution and return appropriately the *results*” to the application. *Id.* (emphasis

added). It is the CVM 401 that includes the “context stack 405.” *Id.* Coffman’s conversationally-aware applications serve as the user-facing software of the system, but do not process a user’s request. Rather, that step is being performed by Coffman’s CVM and backend logic. Thus, Petitioner is incorrect that identification of a context in a context stack yields the activation or selection of a conversationally-aware application to process a user’s request. Rather, the search for an active context in Coffman follows the user’s selection of, and interaction with, a conversationally-aware application and the identification of that application’s corresponding context stack. Coffman’s subsystems are invisible to the conversationally-aware application, which receives the final results to be reported to the user.

Addressing Coffman’s disclosure at 37:2-4 next, Petitioner quotes Coffman’s disclosure that “[t]he first task to have its dialog completed by producing a non-ambiguous query is executed” Pet., 44 (quoting Coffman, 37:2-4). Based solely on this thin disclosure, Petitioner concludes that Coffman “*mak[es] clear* that an application is automatically selected and carries out a request as a result of processing by the conversational system.” Pet., 44 (citing also Coffman, 42:1-7, Schmandt, ¶125) (emphasis added).

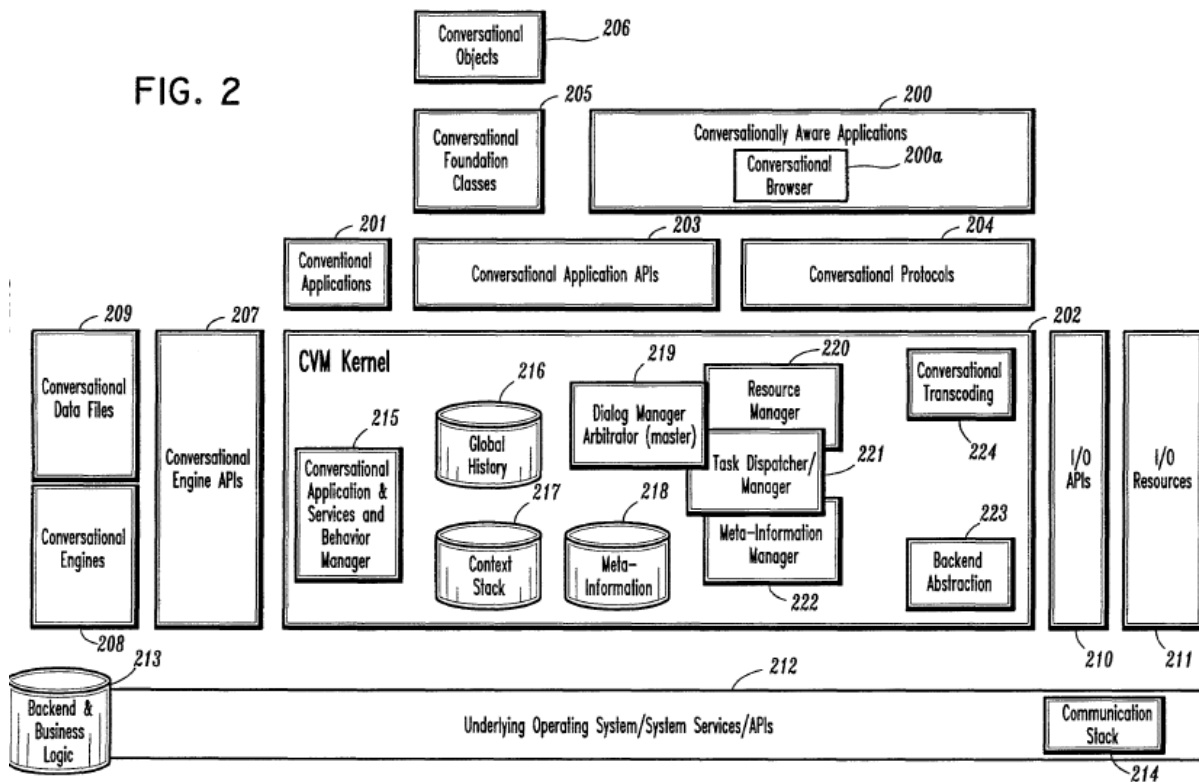
To be certain, the relied-upon disclosure from Coffman at 37:2-4 is anything but clear. Neither the Petition nor paragraph 125 of the Schmandt Declaration

even attempts to explain what this sentence—which is directed to queries, not selecting applications for processing a user request—is attempting to disclose, or why it is relevant to a domain agent being associated with a determined context. Indeed, all Schmandt does differently from the Petition on this theory is to quote—rather than simply cite—a portion of Coffman’s disclosure at 42:1-7. *See* Schmandt, ¶125. That quoted portion of Coffman (“context becomes active and the corresponding process becomes active”), and its lack of relevance to the “processing ... with the domain agent” language, has already been addressed above. Nothing in the Petition or Schmandt explains why or how Coffman’s disclosure at 37:2-4 “mak[es] clear” that an “application is automatically selected and carries out a request as a result of processing by the conversational system,” as alleged by the Petition. Instead, Petitioner appears to hope that the Board will figure out its own theory of relevance, which is improper. This conclusory assertion should be rejected as wholly unsupported and inconsistent with Coffman’s actual operation.

Petitioner’s citations to Coffman’s disclosures at 13:8-13 and 40:9-19 for support are accompanied by no explanation or discussion of those citations. *See* Pet., 42. Indeed, these citations do not support Petitioner’s challenge. Coffman’s disclosure at 13:8-13 describes actions at the CVM 14, not a conversationally-aware application. *See* Coffman, 13:8-13. Coffman’s disclosure at 40:9-19 is

directed to the “input/output processing” of the CVM 401, provided “in further detail.” Coffman, 40:9. Nowhere in this paragraph does Coffman discuss selecting a conversationally-aware application to process a user’s request based on a determined context of a user’s utterance. To the contrary, Coffman expressly discloses that the CVM 401’s actions include “executing a task, listening for new input, and producing an output/feedback.” Coffman, 40:12. This is flatly inconsistent with Petitioner’s theory that a conversationally-aware application processes user questions, queries, and commands.

Further, as shown in Coffman’s FIG. 2, Coffman’s conversationally-aware application interfaces through conversational application APIs to communicate with the CVM Kernel 202, including the dialog manager 219, backend abstraction layer 223, and backend business logic 213.



As introduced in Section IV.B above, the “backend abstraction layer 223 ... allows access to backend business logic 213 via the dialog manager 219 (rather than bypassing the dialog manager 219).” Coffman, 22:1-3. The “backend abstraction layer 223 can translate input and output to and from the dialog manager 219 to database queries. This layer 223 will convert standardized attribute value n-uples into database queries and translate the result of such queries into tables or sets of attribute value n-uples back to the dialog manager 219.” *Id.*, 22:4-7. Coffman’s conversationally-aware applications rely on the CVM, including dialog manager 219 and backend business logic 213, to implement its tools and generate a response to a user’s utterance. The processing of user requests, queries and

commands is performed by the CVM and the backend subsystems, not by Coffman's conversationally-aware applications.

Thus, Petitioner is incorrect in arguing that a conversationally-aware application processes "requests associated with the determined context." The conversationally-aware application receives processed results for presenting to the user, but is not selected to process those results. Petitioner's argument for this claimed element is incorrect, and warrants denial of the Petition.

2. Petitioner's Ground 1 Fails to Demonstrate "associating the one or more recognized words or phrases and a pronunciation ... with the determined identity and the request contained in the utterance in response to the one or more recognized words or phrases satisfying a predetermined confidence level"

As noted above, method claim 1 of the '006 Patent includes the step of "associating the one or more recognized words or phrases and a pronunciation associated with the one or more recognized words or phrases with the determined identity and the request contained in the utterance in response to the one or more recognized words or phrases satisfying a predetermined confidence level." '006 Pat., 25:66-26:4.

Petitioner presents two theories on this claimed feature, but each theory is deficient and fails. Pet., 36-37. Petitioner first alleges that this feature is obvious because "Coffman explains that application registration involves providing baseforms (pronunciation rules) for the active vocabulary to the conversational

system,” and therefore a “POSITA would have therefore understood that successful recognition of words and phrases would have involved successfully matching the user’s utterance to the pronunciation rules.” Pet., 36. However, this is merely a statement that recognition of words or phrases would require a user’s compliance with existing pronunciation rules, regardless of the user’s identity. This argument eliminates at least the “determined identify” from the claim and does not satisfy the “associating” step of claim 1.

Petitioner then takes an illogical leap and asserts that “the subsequent matching of the recognized words and phrases to a context *would also result in associating the pronunciations* specified by the baseforms of those words and phrases *with the user’s identity and the request associated with the context.*” Pet., 36-37 (citing only Schmandt, ¶115, not any prior art) (emphasis added). Nowhere in Coffman—or any other cited reference—is there a disclosed step of associating any pronunciation with a user’s identity. Petitioner’s use of its expert to fill in this gap in the prior art’s disclosure is improper under the law. *K/S Himpp v. Hear-Wear Techs., LLC*, 751 F.3d 1362, 1365 (Fed. Cir. 2014); 35 U.S.C. § 311(b) (*inter partes* review may only be requested “on the basis of prior art consisting of patents or printed publications.”). Further, even assuming that Coffman’s baseforms included pronunciation rules, checking compliance with an existing pronunciation rule as part of recognizing a word or phrase is not

tantamount to associating a pronunciation with a user's determined identity. There is no reason that checking compliance with an existing pronunciation rule as part of recognizing a word or phrase would also necessarily, inherently, or even obviously also include the step of associating a pronunciation with a user's determined identity. Petitioner's argument to the contrary is unfounded and wrong.

In support of its obviousness argument, Petitioner next asserts that a "POSITA would have understood that a user's pronunciation of words and phrases would be reflected in the voice characteristics of their utterance," and therefore it would have been obvious that "by performing speaker identification in accordance with 1[b][ii], the user's pronunciation of the recognized words and phrases would have been 'associat[ed]'" with the user's determined identity as claimed. Pet., 37. This theory also fails, and improperly treats this claimed element as an inherent aspect of speaker identification despite a lack of evidence that a step of pronunciation association necessarily must occur as part of a speaker identification. Like Petitioner's first theory on this claim element, this second theory also fails because Petitioner has made no showing that performing speaker identification necessarily includes the step of "associating the one or more recognized words or phrases and a pronunciation ... with the determined identity and the request contained in the utterance ..." as claimed.

Petitioner’s failure to establish this claimed “associating” element in the Ground 1 challenge is even more apparent when it is considered that the claimed step of “associating” occurs “in response to the one or more recognized words or phrases satisfying a predetermined confidence level.” Petitioner contends that “the concept of a confidence level has been well known in the art since at least the 1970s,” and that satisfying a predetermined confidence level “would help ensure that the system did not incorrectly associate an input with a particular user, context and query” Pet., 37-38. But this argument is flatly inconsistent with Petitioner’s theory why the claimed “associating” step would have been obvious. Specifically, Petitioner argued that the “associating” step occurred as part of the *process* of recognizing words and phrases, without accurately accounting for the cause-and-effect aspect of this claimed feature. Petitioner specifically claimed that a “POSITA would have therefore understood that successful recognition of words and phrases would have involved successfully matching the user’s utterance to the pronunciation rules.” Pet., 36. In other words, Petitioner argued that matching the user’s utterance to the pronunciation rules would have been required as a *condition* for successful recognition of words and phrases.

But under claim 1, the claimed “associating” does not occur as part of satisfying a predetermined confidence level, but rather “*in response to* the one or more recognized words or phrases satisfying a predetermined confidence level.”

Petitioner’s theory fails to accurately address or account for the cause-and-effect aspect of this claimed feature.

3. Ground 1 Fails to Establish Unpatentability of Claim 1

For all the foregoing reasons, claim 1 survives Petitioner’s Ground 1 challenge. Whether considered alone or in combination, Petitioner’s asserted Ground 1 art fails to disclose or suggest every feature of claim 1.

B. Petitioner Also Fails to Demonstrate that Claims 4, 5, and 6 are Obvious Over Coffman in Combination with the Other Asserted Art in Grounds 2-4

Additionally, for the same reasons discussed above in Section V.A.1 with respect to claim 1, Petitioner fails to establish that claims 4, 5, and 6 or any of the claims that depend therefrom are obvious.

1. Petitioner Fails to Demonstrate that Claim 4 is Obvious Over Coffman in Combination with the Other Asserted Art in Ground 2

Like claim 1, claim 4 recites, in relevant part, “processing the formulated request with the domain agent associated with the determined context” Petitioner notes that this element, referred to as 4[e], is “identical” to the corresponding features in claim 1, and therefore simply refers back to its claim 1 analysis for these features. Pet., 62. For the same reasons discussed in Section V.A.1 above, Petitioner has failed to establish these features as obvious over the prior art. Ground 2 fails for at least the same reasons as Ground 1 fails.

2. Petitioner Fails to Demonstrate that Claim 5 is Obvious Over Coffman in Combination with the Other Asserted Art in Ground 3

Like claim 1, claim 5 recites, in relevant part, “processing the formulated request with the domain agent associated with the determined context”

Petitioner notes that this element, referred to as 5[e], corresponds to equivalent features in claim 1, and therefore simply refers back to its claim 1 analysis for these features. Pet., 63, 74. For the same reasons discussed in Section V.A.1 above, Petitioner has failed to establish these features as obvious over the prior art. Ground 3 fails for at least the same reasons as Ground 1 fails.

3. Petitioner Fails to Demonstrate that Claim 6 is Obvious Over Coffman in Combination with the Other Asserted Art in Ground 4

Like claim 1, claim 6 recites, in relevant part, “processing the formulated request with the domain agent associated with the determined context”

Petitioner notes that this element, referred to as 6[d], corresponds to equivalent features in claim 1, and therefore simply refers back to its claim 1 analysis for these features. Pet., 78-79. For the same reasons discussed in Section V.A.1 above, Petitioner has failed to establish these features as obvious over the prior art. Ground 4 fails for at least the same reasons as Ground 1 fails.

4. Ground 2, 3, and 4 Also Fail to Establish Unpatentability of Any Challenged Claim

Therefore, for the same reasons discussed above with respect to claim 1, Petitioner fails to establish that claims 4, 5, and 6 or any of the claims that depend therefrom are obvious.

VI. CONCLUSION

For the reasons presented above, the Petition should be denied, and no *inter partes* review should be instituted.

Dated: July 16, 2024

/Wayne M. Helge/

Wayne M. Helge (Reg. No. 56,905)

James T. Wilson (Reg. No. 41,439)

NOSSAMAN, LLP

1401 New York Ave, NW

Suite 800

Washington, DC 20005

Telephone: (202) 887-1400

Email: whelge@nossaman.com

Email: jwilson@nossaman.com

Counsel for Patent Owner

CERTIFICATE OF WORD COUNT

The undersigned certifies that the foregoing PATENT OWNER'S PRELIMINARY RESPONSE complies with the type-volume limitation in 37 C.F.R. § 42.24(b)(1). According to the word-processing system's word count, the brief contains 7,921 words, excluding the parts of the brief exempted by 37 C.F.R. § 42.24(a).

By: /Wayne M. Helge/
Wayne M. Helge (Reg. No. 56,905)
Attorney for Patent Owner

CERTIFICATE OF SERVICE

I hereby certify that on July 16, 2024, a true and correct copy of the foregoing document was served via email, by consent, to Petitioner by serving the correspondence email addresses of record as follows:

Heidi L. Keefe
hkeefe@cooley.com

Andrew C. Mace
amace@cooley.com

Mark R. Weinstein
mweinstein@cooley.com

Reuben Chen
rchen@cooley.com

COOLEY LLP
ATTN: Patent Group
1299 Pennsylvania Ave. NW, Suite 700
Washington D.C. 20004
Tel: (650) 842-7800

/Wayne M. Helge/
Wayne M. Helge (Reg. No. 56,905)
Attorney for Patent Owner