

UNITED STATES PATENT AND TRADEMARK OFFICE

---

BEFORE THE PATENT TRIAL AND APPEAL BOARD

---

GOOGLE LLC,  
Petitioner,

v.

DIALECT, LLC,  
Patent Owner.

---

Case IPR2024-00749  
Patent 7,693,720

---

**PATENT OWNER'S PRELIMINARY RESPONSE**

**TABLE OF CONTENTS**

|      |                                                                                                                                                                                                                                    |    |
|------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| I.   | INTRODUCTION .....                                                                                                                                                                                                                 | 1  |
| II.  | THE ’720 PATENT DISCLOSES AND CLAIMS DOMAIN AGENTS<br>THAT ARE NOT DISCLOSED OR SUGGESTED BY THE<br>ASSERTED ART .....                                                                                                             | 4  |
| A.   | The ’720 Patent Selects Domain Agents to Process User<br>Requests and Commands .....                                                                                                                                               | 6  |
| B.   | The ’720 Patent Expressly Claims the Selection of a Domain<br>Agent to Receive, Process, and Respond to a User Request and<br>Command .....                                                                                        | 8  |
| III. | THE BOARD NEED NOT REACH ANY CLAIM CONSTRUCTION<br>ISSUES TO DENY THE PETITION.....                                                                                                                                                | 10 |
| IV.  | PETITIONER’S CHALLENGES RELY SOLELY ON COFFMAN TO<br>DISCLOSE THE CLAIMED “DOMAIN AGENT” .....                                                                                                                                     | 13 |
| A.   | Summary of Petitioner’s Challenges.....                                                                                                                                                                                            | 13 |
| B.   | Coffman’s Non-Proprietary Speech Engine and Application<br>Program Interfaces (APIs) Receive Speech Inputs from<br>Con conversationally-Aware Applications But Do Not Select or<br>Invoke Conversationally-Aware Applications..... | 15 |
| C.   | Coffman’s Context Stacks are Unique to Each Application.....                                                                                                                                                                       | 25 |
| V.   | PETITIONER’S GROUND 1 THEORY IMPROPERLY<br>CHARACTERIZES COFFMAN’S CONVERSATIONALLY-AWARE<br>APPLICATION AS THE CLAIMED “DOMAIN AGENT” AND FAILS<br>ON THE MERITS .....                                                            | 26 |
| A.   | Petitioner’s Ground 1 Fails to Disclose or Suggest Selecting a<br>Con conversationally-Aware Application “Based on the Determined<br>Meaning” of a User’s Utterance.....                                                           | 27 |
| B.   | Petitioner’s Ground 1 Also Fails to Demonstrate that a “Parser”<br>selects a Conversationally-Aware Application .....                                                                                                              | 32 |

IPR2024-00749 Patent Owner’s Preliminary Response

|     |                                                                                                                                                                                       |    |
|-----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| C.  | Petitioner’s Ground 1 Fails to Demonstrate “the selected domain agent uses the communicatively coupled services to create a response to the formulated question or command ...” ..... | 34 |
| D.  | Ground 1 Also Fails to Demonstrate that Claim 31 is Obvious Over Coffman Alone or in Combination with the Remaining Ground 1 Prior Art .....                                          | 40 |
| E.  | Ground 1 Fails, and Petitioner Presents No Alternative Theory for the Claimed “Domain Agent” .....                                                                                    | 40 |
| VI. | CONCLUSION.....                                                                                                                                                                       | 41 |

**TABLE OF AUTHORITIES**

**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        | Reserved                                                                                                               |
| 2003        | U.S. Patent 8,015,006 (“the ’006 Patent”)                                                                              |
| 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 1-16, 18-25, 27, 29-43, 52, and 53 of U.S. Patent No. 7,693,720 (the “’720 Patent”), as requested by Google LLC (“Petitioner”), should be denied.<sup>1</sup>

## **I. INTRODUCTION**

The ’720 Patent is entitled, “Mobile Systems and Methods for Responding to Natural Language Speech Utterance.” Central to the disclosed and claimed invention, the ’720 Patent discloses a mobile 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 ’720 Patent discloses and claims a parser that selects a proper domain agent for processing a user’s question or command based on a determined context of a user’s utterance. Per the ’720 Patent, this “environment makes significant use of context, prior information, domain knowledge, and user specific profile data to achieve a natural environment for one or more users making queries or commands in multiple domains.” ’720 Patent, 2:54-57.

---

<sup>1</sup> 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).

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 '720 Patent's system receives user utterances through a common user interface, determines a context of the user utterance, and then selects a domain agent based on the determined meaning of the utterance to process the utterance's question or command. Unlike Coffman, where the context stack to be searched depends on the conversationally-aware application initially accessed by the user, the '720 Patent first determines a context of the user's utterance, and then selects the appropriate domain agent to process the utterance's

question or command based on the determined meaning. The invoked domain agent generates results to the formulated request, and a speech unit in the '720 Patent presents results of the processed request to the user.

While Petitioner attempts to conflate Coffman's conversationally-aware application with the '720 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 the challenged claims. For this reason, Petitioner's challenge of independent claims 1 and 31, and all the dependent challenged claims, must fail.

## **II. THE '720 PATENT DISCLOSES AND CLAIMS DOMAIN AGENTS THAT ARE NOT DISCLOSED OR SUGGESTED BY THE ASSERTED ART**

The '720 Patent discloses and claims a mobile system and method to process and respond to a user generated natural language speech utterance including, in relevant part, selecting a domain agent based on a determined context of a user utterance. The domain agent is selected to process the user's question or command, and create a response to the question or command. FIG. 5 of the '720 Patent, reproduced below, is an overall diagrammatic view of an embodiment of the '720 Patent:

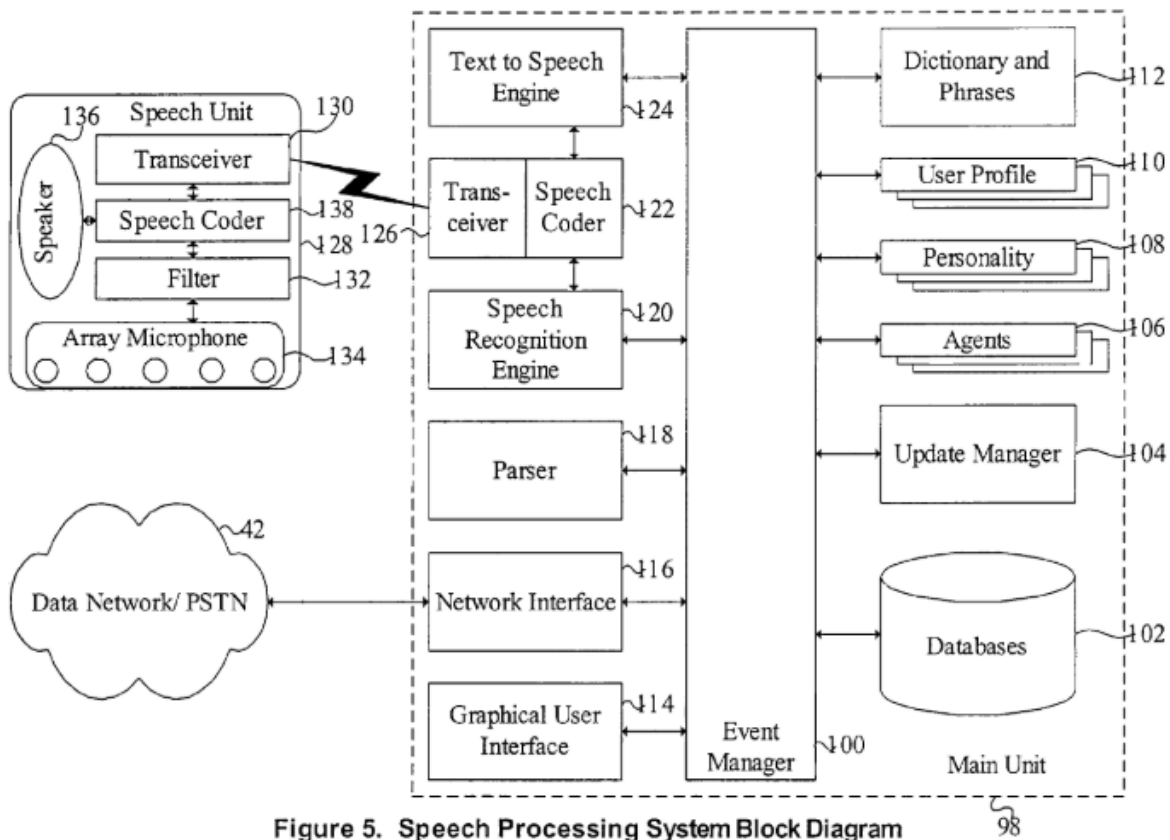


Figure 5. Speech Processing System Block Diagram

'720 Patent, FIG. 5, 15:44-46. 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.*, 20:6-21:10. The system also includes multiple agents 106, discussed in more detail below. *Id.* 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.*, 20:28-30. 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.*, 20:31-33, 20:62-64.

**A. The '720 Patent Selects Domain Agents to Process User Requests and Commands**

The '720 Patent's system 90 includes different types of agents 106. *Id.*, 21:61-22:6. The agents 106 “contain packages of both generic and domain specific behavior for the system,” and play a critical role in the efficient processing of the speech received at the main unit. *Id.*, 20:12-11. As shown in the exemplary agent architecture of FIG. 6, there are domain agents 156, a system agent 150, and an agent library 158 included within FIG.5's “agents 106.” *Id.*, 21:61-22:6. “A system agent ... provides default functionality and basic services.” *Id.*, 21:40-41. “[G]eneric and domain specific behavior and information may be organized into agents,” also referred to as “domain specific agents.” *Id.*, 21:38-45.

Describing domain agents 156 in more detail, the '720 Patent discloses that “domain specific agents may provide complete, convenient and re-distributable packages or modules for each application area. In other words, an agent may include everything needed to extend or modify the functionality of the system in a current or new domain.” *Id.*, 21:41-45.

“The speech recognition engine 120 may recognize words and phrases, using for example, information in the dictionary and phrase tables 112, and pass these to the parser 118 for interpretation.” *Id.*, 27:2-5. “The parser may determine a

context for an utterance by applying prior probabilities or fuzzy possibilities to keyword matching, user profile 110, dialog history, and context stack contents.”

*Id.*, 27:60-63. “The context of a question or command may determine the domain and thereby, the domain agent 156, if any, to be invoked.” *Id.*, 27:63-65.

In particular, the parser may invoke a domain agent 156 among the agents 106 to process the user's question or command. “Once the context and the criteria are determined, the parser 118 may form the question or command in a standard format or hierarchical data structure used for processing by the agents 150, 156.” *Id.*, 28:63-66.

As disclosed, “[s]pecific questions may be generally directed to one of the domain agents 156. ... Based on the question, command or context and the parameters or criteria, the domain agent [156] may create one or more queries to one or more local or external information sources.” *Id.*, 29:19-24. For example, the agent 106 is responsible for “formulat[ing] one or more queries to one or more local and/or network data sources or sending appropriate commands to local or remote devices or the system itself.” *Id.*, 6:47-50; *see also id.*, 21:3-6. In this way, the “agents can then process the commands or questions.” *Id.*, 21:2-3.

Further, the “agents 106 may return results of questions as responses to users.” *Id.*, 21:11-12. “The agent may 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.*, 21:16-19.

“Once coded, the utterances may be transmitted from the main unit 98 ... to the transceiver 130 on the speech unit 128. The utterance may then be decoded and decompressed by the speech coder 138 and output by the speaker 136” to the user. *Id.*, 21:19-23.

**B. The ’720 Patent Expressly Claims the Selection of a Domain Agent to Receive, Process, and Respond to a User Request and Command**

Independent claims 1 and 31 are the sole independent claims challenged by the Petition. Consistent with the ’720 Patent’s disclosure introduced above, independent claim 1 of the ’720 Patent recites that a “parser interprets the recognized words or phrases by determining a context for the natural language speech utterance; [and then] selecting at least one of the plurality of domain agents based on the determined context.” In full, claim 1 recites:

1. A mobile system responsive to a user generated natural language speech utterance, comprising:
  - a speech unit connected to a computer device on a vehicle, wherein the speech unit receives a natural language speech utterance from a user and converts the received natural language speech utterance into an electronic signal; and
  - a natural language speech processing system connected to the computer device on the vehicle, wherein the natural language speech processing system receives, processes, and responds to the electronic signal using data received from a plurality of

domain agents, wherein the natural language speech processing system includes:

a speech recognition engine that recognizes at least one of words or phrases from the electronic signal using at least the data received from the plurality of domain agents, wherein the data used by the speech recognition engine includes a plurality of dictionary and phrase entries that are dynamically updated based on at least a history of a current dialog and one or more prior dialogs associated with the user;

a parser that interprets the recognized words or phrases, wherein the parser uses at least the data received from *the plurality of domain agents* to interpret the recognized words or phrases, wherein the parser interprets the recognized words or phrases by:

determining a context for the natural language speech utterance;

*selecting at least one of the plurality of domain agents based on the determined context;* and

transforming the recognized words or phrases into at least one of a question or a command, wherein the at least one question or command is formulated in a grammar that the selected domain agent uses to process the formulated question or command; and

an agent architecture that communicatively couples services of each of an agent manager, a system agent, the plurality of domain agents, and an agent library that includes one or

more utilities that can be used by the system agent and the plurality of domain agents, wherein *the selected domain agent uses the communicatively coupled services to create a response to the formulated question or command and format the response for presentation to the user.*

'720 Patent, claim 1 (emphasis added). As Petitioner explains, claim 31's elements predominantly recite "a method counterpart" to the features of claim 1. Pet., 79-80. Therefore, claim 31 will not be separately presented or discussed in detail in this Preliminary Response.

As discussed below, Petitioner's challenge fails to identify any "domain agent" in the asserted combinations that satisfies each of these claim requirements.

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

The Petitioner identifies no terms as requiring constructions and presents no constructions. *See* Pet., 23. The United States District Court for the Eastern District of Virginia has issued constructions for the following terms (Ex. 2007):

| <b>Term</b>                       | <b>Patent(s)</b> | <b>Claim(s)</b> | <b>Construction</b>                                                                                                                 |
|-----------------------------------|------------------|-----------------|-------------------------------------------------------------------------------------------------------------------------------------|
| <b>context</b>                    | '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          |                                                                                                                                     |
| <b>context stack</b>              | '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      |                                                                                                                                     |
| <b>entries in a context stack</b> | '468 Patent      | 19              | "elements of a context stack that store or reference contexts"                                                                      |
| <b>context entries</b>            | '957 Patent      | 1, 3, 4, 5, 7   | "elements of a context stack that store or reference contexts"                                                                      |
| <b>grammar</b>                    | '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        |                                                                                                                                     |

|                                                       |             |             |                                                                                                                                                                                 |
|-------------------------------------------------------|-------------|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>context description grammar</b>                    | '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" |
| <b>domain agent</b>                                   | '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      |                                                                                                                                                                                 |
| <b>procedures sensitive to the determined context</b> | '006 Patent | 5           | "procedures that differ based on which context has been determined"                                                                                                             |
| <b>parser</b>                                         | '720 Patent | 1, 35       | "software that analyzes a string of words"                                                                                                                                      |
|                                                       | '006 Patent | 1, 5, 10    |                                                                                                                                                                                 |

In the table above, U.S. Patent 7,693,720 (“the ’720 Patent”) refers to the patent-at-issue in this proceeding. U.S. Patent 8,015,006 (“the ’006 Patent”) (Ex. 2003), 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 ’720 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 V below, Petitioner’s grounds include fundamental flaws and fail to establish the disclosure of every claimed element in at least independent claim 1. 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 three Grounds in its Petition.

Ground 1 asserts that claims 1-16, 22-25, 27, 29-31, 35-43, 52, 55 are obvious in view of Coffman,<sup>2</sup> Kanevsky,<sup>3</sup> Kneser,<sup>4</sup> and Zadrozny.<sup>5</sup> Collectively, Coffman, Kanevsky, Kneser, and Zadrozny are referred to in the Petition as “Ground 1 Prior Art.” Pet., 20.

Ground 2 asserts that claims 18, 20-21, and 32-33 are obvious in view of the Ground 1 Prior Art further in view of DeLine.<sup>6</sup> Collectively, the Ground 1 Prior

---

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

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

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

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

<sup>6</sup> Ex. 1010, U.S. Patent No. 6,420,975 (“DeLine”).

Art plus DeLine—five references—are referred to in the Petition as “Ground 2 Prior Art.” Pet., 20.

Ground 3 asserts that claims 19 and 34 are obvious in view of the Ground 2 Prior Art, further in view of Gerson.<sup>7</sup> Pet., 20.

In each Ground, Petitioner relies upon a conversationally-aware application from Coffman to disclose the claimed “domain agent” in claims 1 and 31. But this assertion misunderstands Coffman, because Coffman’s conversationally-aware applications cannot satisfy each claimed requirement of a “domain agent” as recited in these 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 ’720 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 conversationally-aware application simply engages in a dialog with a user; this is

---

<sup>7</sup> Ex. 1011, U.S. Patent No. 6,937,977 (“Gerson”).

what makes the application “conversationally-aware” and distinguishes it from a conventional application. As will be explained, Coffman’s conversationally-aware applications are not selected based on a determined meaning, and are not then invoked to process a user’s request.

As noted in the Petition, Coffman’s issued U.S. Patent (U.S. Patent No. 7,137,126) was known to the Examiner. Pet., 20-21; *see also* Ex. 1001, 3 (citing U.S. Patent No. 7,137,126 among the ’720 Patent’s References Cited). The Examiner’s allowance of the ’720 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

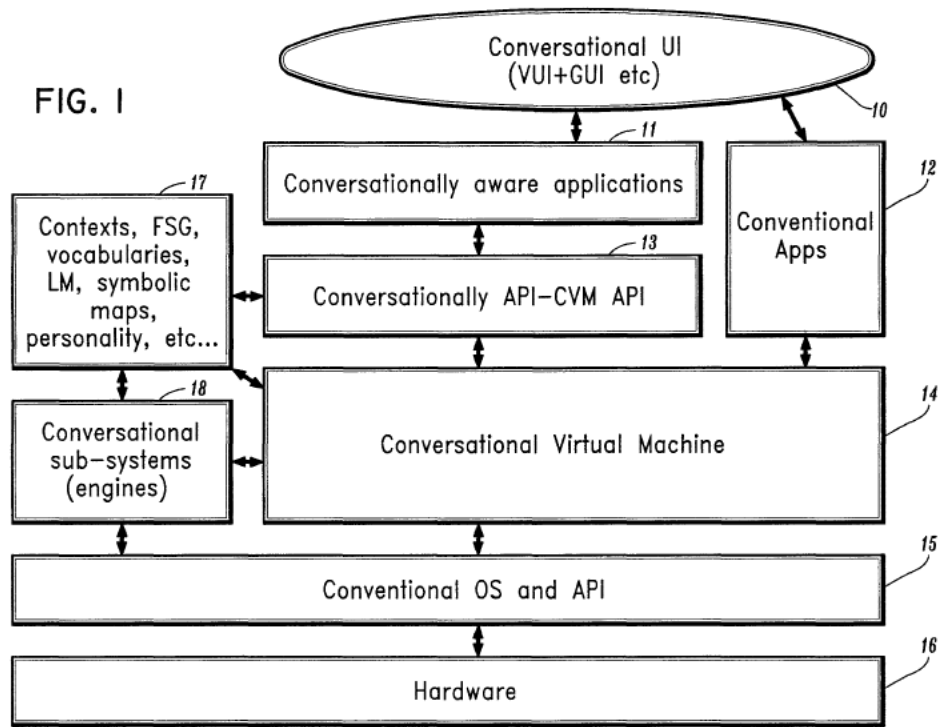
... user and machine.” Coffman, 5:1-4. Coffman refers to these aspects of the CVM as a “subsystem,” and notes that the “subsystem [is] accessed through the CVM.” Coffman, 5:4-5. As an aid to software developers, Coffman explains that the “development environment offered by the CVM ... allows a developer 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.

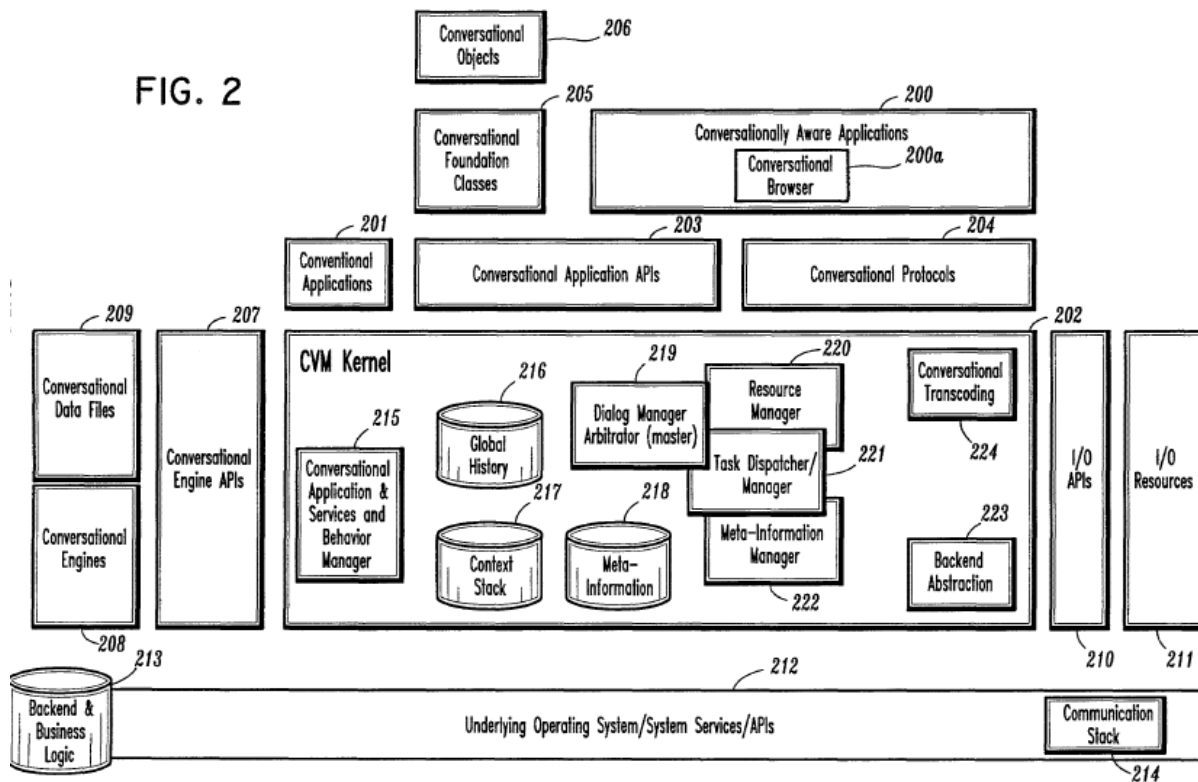
FIG. 1



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):

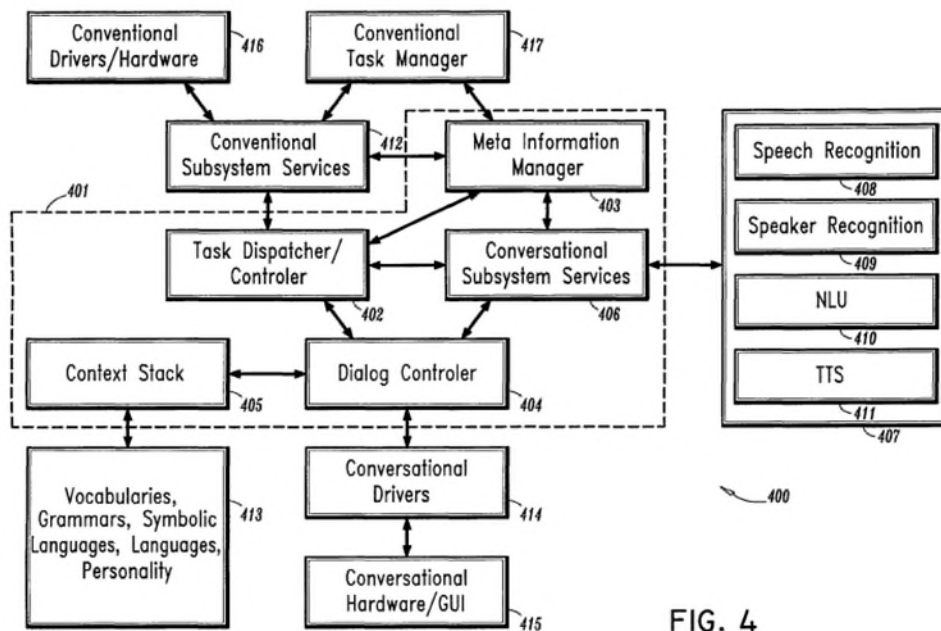


FIG. 4

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 and initiate a new process to be performed by a different conversationally-aware application based on the active context.

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

Ground 1 is facially deficient in establishing the unpatentability of any challenged claim. 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. 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.

Contrary to Petitioner’s assertion, Coffman does not disclose a parser that “selects” any conversationally-aware application based on a meaning of an utterance based on the determined context. Indeed, it is a user who selects a conversationally-aware application to be used. 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. There is no selecting of a conversationally-aware application based on a determined meaning by Coffman’s CVM or any component thereof, nor does Coffman’s conversationally-aware application process a request.

**A. Petitioner’s Ground 1 Fails to Disclose or Suggest Selecting a Conversationally-Aware Application “Based on the Determined Meaning” of a User’s Utterance**

As introduced above, claim 1 recites, in relevant part, a “parser interprets the recognized words or phrases by determining a context for the natural language speech utterance; [and] selecting at least one of the plurality of domain agents based on the determined context.” Petitioner alleges that Coffman’s conversationally-aware application is a “**domain agent.**” Pet., 33 (citing Coffman, 13:26-29; Schmandt, ¶89) (emphasis in original). However, this characterization

fails because Coffman fails to disclose selecting a conversationally-aware application, much less selecting such a conversationally-aware application based on a “determined context.”<sup>8</sup>

Coffman searches for an active context in a stack of contexts by comparing “new queries or arguments” to the stack “until an acceptable match is obtained and optionally confirmation is obtained from the user.” See Pet., 43-44 (citing Coffman, 42:1-7). Petitioner’s argument for the parser’s selecting of at least one domain agent in claim 1 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. These two things are different elements of Coffman, and Coffman does not support Petitioner’s characterization of a conversationally-aware 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

---

<sup>8</sup> Section V.B below also addresses Petitioner’s failure to establish that any selection of a conversationally-aware application is performed by a “parser” in Coffman.

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 query. 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 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 discloses that “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.* Thus, Petitioner is incorrect that identification of a context in a context stack yields the activation or selection of a conversationally-aware application. 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. Coffman

fails to disclose the claimed feature of a “parser ... selecting at least one of the plurality of domain agents based on the determined context.”

Not to be overlooked, the Petition also points to Coffman’s disclosure at 37:2-4 to allege that Coffman’s parser selects a conversationally-aware application. Pet., 44; *see also* Pet., 33-34 (alleging that “task executes query”). But this disclosure’s relevance is not explained by Petitioner or Schmandt, and the relied-upon disclosure from Coffman at 37:2-4 is anything but clear. This disclosure also does not support the selection of a conversationally-aware application.

Finally, confirming that Coffman’s determination of an active context in the context stack does *not* lead to a selection of a conversationally-aware application based on the context, 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. Similarly, Coffman explains that a “context stack 217 comprises all the information associated with *an application*.” Coffman, 21:23-24 (emphasis added). As such, a search of an application’s own context stack 217 may identify an active context and active process for a discourse in that application, but does not support Petitioner’s expansive reading that Coffman will then select a conversationally-aware application among many applications to process a user’s request based on the context stack search. Rather, a user interacting with a

conversationally-aware application in Coffman results in, at most, an identification of an active context for that user’s current discourse with that application. *See* Coffman, 30:4-5 (“[E]ach new task, process, or thread creates a new stack entry and is associated with a discourse.”); 29:4-5 (“Each application may be associated with multiple discourses ....”); 30:23-24 (“[F]or 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.”) (emphasis added). This does not disclose or suggest a parser in Coffman selecting a conversationally-aware application based on a determined meaning, contrary to Petitioner’s theory. That theory is wrong.

**B. Petitioner’s Ground 1 Also Fails to Demonstrate that a “Parser” selects a Conversationally-Aware Application**

Claim 1 of the ’720 Patent further requires that the claimed parser is the entity “selecting at least one of the plurality of domain agents based on the determined context.” Petitioner asserts that Coffman discloses a “parser” as part of Coffman’s NLU technology. Pet., 42 (pointing to Coffman’s NLU system 410 within conversational resource systems (engines) 407). But the Petition fails to establish that any selecting of a conversationally aware application is performed by Coffman’s NLU system 410. *Id.*

Indeed, Petitioner’s single-paragraph analysis of this claim language does not even try to explain how Coffman’s NLU system 410 allegedly selects a conversationally-aware application. Instead, Petitioner avoids discussion of this operative detail using passive voice, stating that a conversationally-aware application *is* selected, not that the NLU system 410 selects it. Pet., 43. Petitioner states: “A conversationally-aware application (**‘domain agent’**) is **‘select[ed]...based on the determined context’** because activation of an application for processing the user’s query is based on the application’s association with the determined context.” Pet., 43 (citing Coffman, 42:1-7; Schmandt, ¶116) (emphasis in original). As already discussed above, Coffman’s disclosure at 42:1-7’s does not support a selection of a conversationally-aware application. Schmandt’s paragraph 116 also offers no further insight, and instead parrots the Petition’s unsupported conclusory language without providing any technical explanation.

As a result, Petitioner also fails to establish that any parser, or even Coffman’s NLU system 410, performs the claimed “selecting at least one of the plurality of domain agents based on the determined context,” as required by claim 1. This failure is yet another fatal deficiency in the Petition, and independently warrants denial.

**C. Petitioner's Ground 1 Fails to Demonstrate “the selected domain agent uses the communicatively coupled services to create a response to the formulated question or command ...”**

As noted above, per claim 1 of the '720 Patent, “the selected domain agent uses the communicatively coupled services to create a response to the formulated question or command and format the response for presentation to the user.” '720 Pat., 33:3-6. Contrary to the Petition's “domain agent” analysis, a conversationally-aware application in Coffman does not use “communicatively coupled services to create a response to the formulated question or command ....”

Petitioner's theory on this claimed feature is presented at pp. 53-55 of the Petition. Petitioner alleges that “Coffman teaches, and a POSITA would have found it obvious, that the selected conversationally-aware application ... uses the services of these components of the CVM and conversational subsystem ... ‘to create a response to the formulated question or command’ in a few different ways.” Pet., 53. None of these alleged “ways” discloses the required claim element.

First, Petitioner alleges that the conversationally-aware application registers with the system “so that the application can be available to be selected to respond to the user.” Pet., 53 (citing Schmandt, ¶137). This is unsupported by Coffman, which does not disclose the selection of a conversationally-aware application to respond to a user. Both Schmandt and the Petition cite to Coffman's disclosure at 19:26-20:2 for this assertion, but that portion of Coffman does not support the

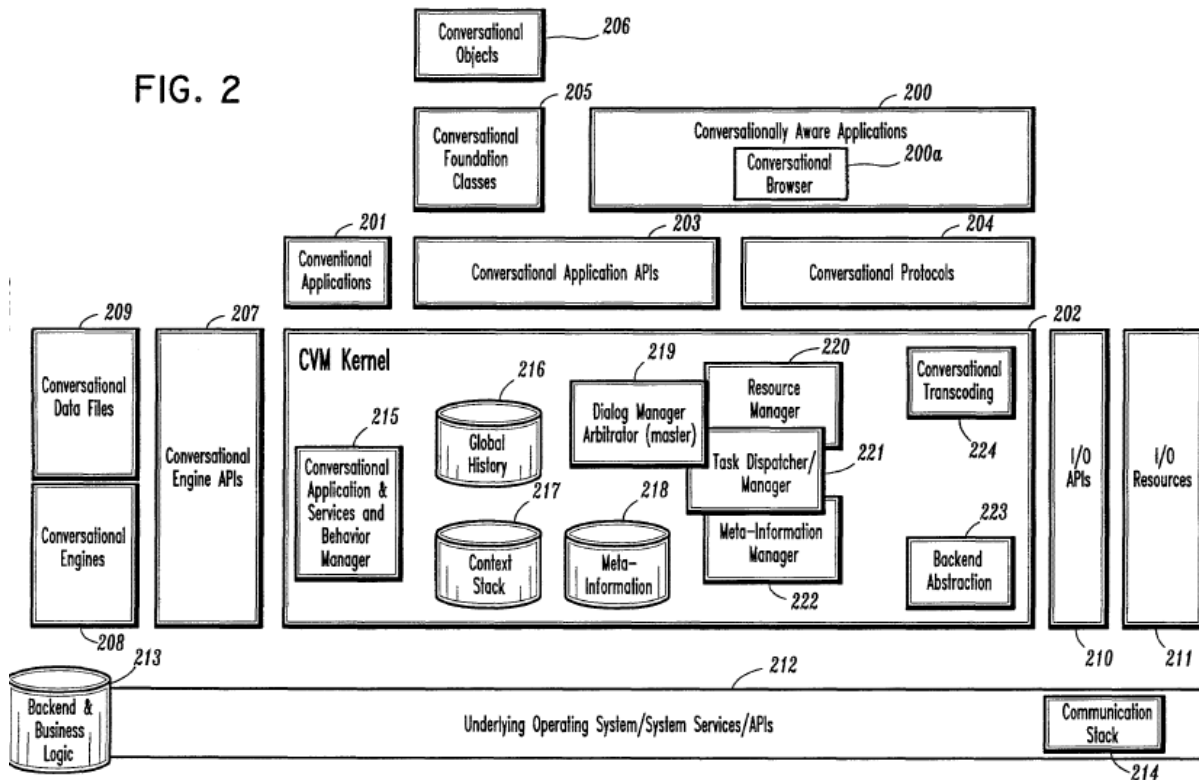
alleged selection of a conversationally-aware application to respond to a user.

Rather, those portions of Coffman confirm that Coffman’s CVM—not the conversationally-aware application—responds to user questions or commands.

Coffman’s CVM 401 includes context stack 405, task dispatcher/controller 402, dialog controller 404, and conversational subsystem services 406. *See* Coffman, FIG. 4; *see also* Coffman, 58:8-9 (“the CVM automatically handles all the input/output issues, including the conversational and conventional subsystems.”). 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. Moreover, “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 an application. *Id.* (emphasis added). 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.

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

Second, Petitioner alleges that Coffman's "conversational subsystems ... include text-to-speech services to generate a spoken response to the user's input based on a text response from the application." Pet., 54. This theory is also flawed, because Petitioner fails to substantiate what "text response" it is referring to, or where Coffman discloses such a "text response" from an application. The truth is that Coffman discloses no such thing. Coffman does disclose a text-to-speech synthesis (TTS) system 411 (Coffman, 29:4) as part of the conversational system 400. This is illustrated in FIG. 4, reproduced below:

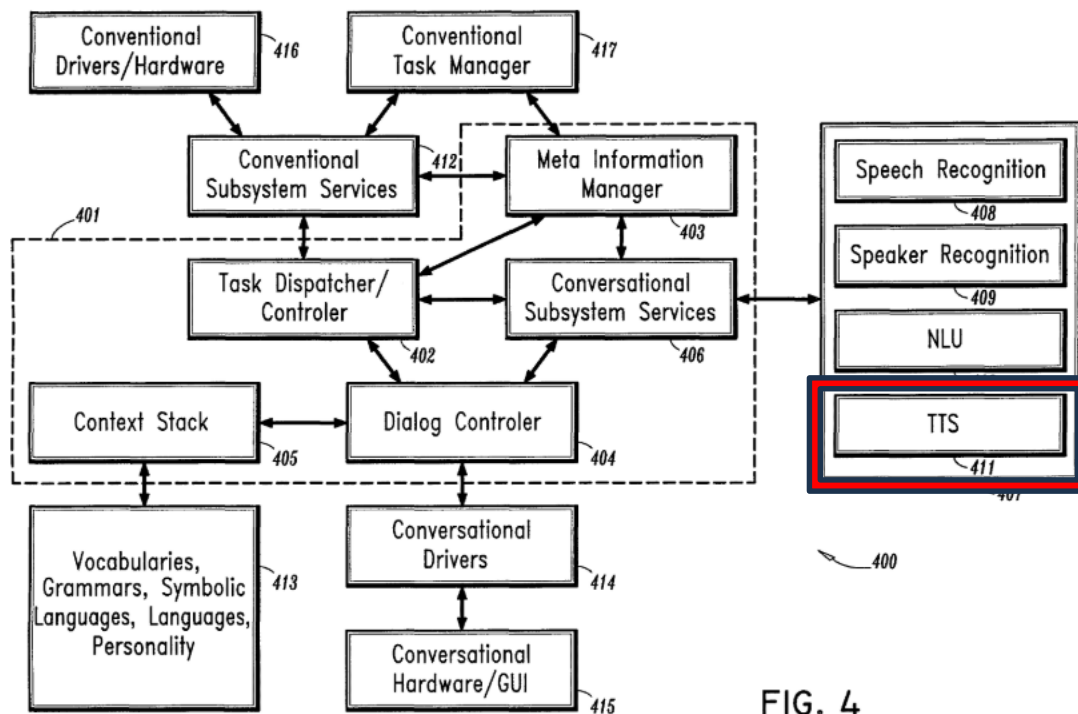


FIG. 4

However, Coffman does not disclose or suggest that the TTS 411 converts an application's text to speech. That unsupported conclusion is Petitioner's alone, and is based on Petitioner's unfounded view on how Coffman's conversationally-aware applications interact with the CVM 401. Pet., 54 (relying on an "application's text-based response" for both theories of how Coffman allegedly discloses the claimed "response"). Moreover, as already discussed above, Coffman discloses that it is the CVM 401, not a conversationally-aware application, that performs such functions as "executing a task, listening for new input, and producing an output/feedback." Coffman, 40:12.

While not directly on point for element 1[b][iii], Petitioner’s argument for element 1[b] asserts that a “conversationally-aware application receives, processes, and responds to user questions, queries and commands.” Pet., 33. Without discussion, Petitioner cites to Coffman’s disclosures at 13:8-13, 37:2-4, and 40:9-19 for this assertion.

These citations do not support Petitioner’s assertion, however. For the same reasons explained in Section V.A, Coffman’s disclosure at 37:2-4 relates to queries, and does not support Petitioner’s assertion regarding a conversationally-aware application. Similarly, 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 meaning 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.

Thus, Petitioner is incorrect in arguing that a conversationally-aware application “uses the communicatively coupled services to create a response to the formulated question or command,” as claimed. The conversationally-aware application receives processed results for presenting to the user, but is not selected and does not to create a response to a formulated question or command.

Petitioner’s argument for this claimed element is incorrect, and warrants denial of the Petition.

**D. Ground 1 Also Fails to Demonstrate that Claim 31 is Obvious Over Coffman Alone or in Combination with the Remaining Ground 1 Prior Art**

As Petitioner explains, claim 31’s elements predominantly recite “a method counterpart” to the features of claim 1. Pet., 79-80. Petitioner therefore substantially copies the failed challenge of claim 1 for its challenge of claim 31. *See Id.* For the same reasons discussed above with respect to claim 1, Petitioner fails to establish that claim 31 or any of the claims that depend therefrom are obvious.

**E. Ground 1 Fails, and Petitioner Presents No Alternative Theory for the Claimed “Domain Agent”**

For each of these reasons presented above, Coffman—whether considered alone or in any combination with the other cited art—fails to disclose at least a parser that “selects at least one of the plurality of domain agents based on the determined meaning, wherein the selected domain agent is configured to receive,

process, and respond to requests associated with the determined context.”

Petitioner's reliance on Coffman's conversationally-aware application to disclose the claimed “domain agent” fails to satisfy at least these features of claim 1.

Further, Petitioner advances no alternative theory in any asserted Ground to shore up the deficiencies in this regard. Thus, the Petition fails with regard to Ground 1 and independent claims 1 and 31. Since all other challenged claims depend directly or indirectly from these independent claims 1 and 31, this defect is fatal to the Petition.

## 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](mailto:whelge@nossaman.com)

Email: [jwilson@nossaman.com](mailto: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 7,734 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