

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

GOOGLE LLC,
Petitioner,

v.

DIALECT, LLC,
Patent Owner.

IPR2024-00746
Patent 7,398,209 B2

**PETITIONER'S REQUEST FOR DIRECTOR REVIEW
OF INSTITUTION DECISION¹**

¹ This IPR is one of five involving challenges to related patents that were denied for substantially the same reason (IPR2024-00746, IPR2024-00747, IPR2024-00748, IPR2024-00749, IPR2024-00751). Similar requests are being filed in all five proceedings.

I. INTRODUCTION

Petitioner Google LLC respectfully requests Director Review of the Board's decision not to institute IPR of U.S. Patent No. 7,398,209 B2 ("209 patent"). (Paper 11 ("Decision").) Director Review is warranted because the Decision rejects the un rebutted testimony of Petitioner's expert, and credits Patent Owner's pure attorney argument. At most, that attorney argument raises a factual dispute that the Board has repeatedly ruled is best resolved on a full record through trial. Moreover, the Decision's reliance on that attorney argument is based on a clearly erroneous understanding and application of the Petition's primary reference, Coffman², to the challenged claims. The Decision's understanding of Coffman is not correct and does not withstand scrutiny. But even if correct, the proper course would have been to institute IPR and resolve any supposed factual dispute on a full record.

II. FACTUAL BACKGROUND

The '209 patent and Coffman both involve controlling a computer using natural language voice commands, for example, in a vehicle. To determine which of several applications the user wants to invoke, the '209 patent and Coffman both rely on the same mechanism: evaluating the user's request in order to determine a matching "context" and associated application that should carry out the request.

² Int'l Patent App. Pub. No. WO 00/20962 A2 (EX1003).

A. The Challenged Patent and Claims

The challenged '209 patent relates to retrieval of online information or processing of commands through a speech interface that allows users to submit natural language speech questions and commands. (EX1001, 1:13-17.) Using information supplied by agents, a “context” for a user’s utterance is determined, and recognized words and phrases in the utterance are transformed into commands and questions that are processed by an agent associated with the context. (*Id.*, 10:49-58, 17:45-51, 18:8-10.) Claim 1, with the key disputed limitations emphasized below, is illustrative:

1. A method responsive to a user generated natural language speech utterance, comprising:
 - receiving the user generated natural language speech utterance, the received user utterance containing at least one request;
 - maintaining a dynamic set of prior probabilities or fuzzy possibilities usable at each stage of processing the received user utterance;
 - recognizing words and phrases contained in the received utterance using information in one or more dictionary and phrase tables;
 - parsing the recognized words and phrases to determine a meaning of the utterance, wherein determining the meaning includes determining a context for the at least one request contained in the utterance based on one or more keywords contained in the recognized words and phrases;
 - selecting at least one domain agent based on the determined meaning**, the selected domain agent being an autonomous executable that receives, processes, and responds to requests associated with the determined context;
 - formulating the at least one request contained in the utterance in accordance with a grammar used by the selected domain agent

to process requests associated with the determined context;
invoking the selected domain agent to process the formulated request; and

presenting results of the processed request to the user, the presented results generated as a result of the invoked domain agent processing the formulated request.

(EX1001, 25:46-26:9.)

B. The Petition and Institution Decision

The Petition challenged claims as obvious over Coffman in view of various references, but relied primarily on Coffman for the key disputed limitations, **“selecting at least one domain agent based on the determined meaning”** and **“invoking the selected domain agent to process the formulated request.”** (Pet. at 14, 36, 41.) The Petition was supported by un rebutted expert testimony (EX1002).

Coffman discloses a multi-modal “conversational” computing system that is remarkably similar to the ’209 patent. (Pet. at 17.) The system includes four key components for purposes of IPR (and this request): a conversational virtual machine (CVM), conversational subsystems, conversationally aware applications, and a context stack. The CVM “provides the capability to have natural dialog with NLU [natural language understanding], NLG [natural language generation], contexts and mixed-initiatives sorted across multiple tasks, processes and discourses (with multiple domains).” (*Id.* at 20 (citing EX1003, 6:6-8); *see also id.* at 34 (citing EX1003, 39:29-31); EX1002 ¶¶89, 44 (citing EX1003, 10:23-25, 11:10-13 (CVM

“allows a computer...to converse with a user”).) The CVM does this by using conversational subsystems 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. (Pet. at 20 (citing EX1003, 5:1-4); EX1002 ¶¶44, 89, 93 (citing EX1003, 19:16-18, 29:1-4).)

The CVM system’s conversationally aware applications, which the Petition mapped to the claimed “**domain agent**,” are applications (e.g., spreadsheet application) specifically programmed for operating with a CVM via application program interfaces (APIs). (Pet. at 33-34 (citing EX1003, 13:26-29, 58:12-20, 58:30-32 (spreadsheet application)); *see also* EX1002 ¶¶45, 120-121.) A conversationally aware application registers and operates with a CVM by providing its data files (such as active vocabulary and language models) to the CVM for use by the conversational subsystems (such as the speech recognition engine). (Pet. at 24-25 (citing EX1003, 57:22-58:11), 37 (citing EX1003, 57:22-27, 36:31-32); *see also* EX1002 ¶¶45, 101.)

The CVM allows the user to run multiple tasks by using a “context stack” comprising entries for “contexts” that the Petition (unsurprisingly) mapped to the claimed “**context**.” (Pet. at 31-32 (citing EX1003, 29:18-30:10, 41:2-5, 42:1-5); *see also* EX1002 ¶¶46, 115-116.) Coffman’s context stack “comprises all the information associated with an application” and its structure allows “maintain[ing]

non-ambiguous conversational interactions with multiple tasks, threads or processes.” (EX1002 ¶46 (quoting EX1003, 21:23-24, 42:16-17); *see also* Pet. at 31-32 (citing EX1003, 29:18-30:10).) Coffman explains that “each new task, process, or thread creates a new stack entry and is associated with a discourse” and that “[e]ach application may be associated with multiple discourses[.]” (Pet. at 31-32 (citing EX1003, 30:4-10); *see also* EX1002 ¶116.) Context stack entries are searched to determine a context matching user input and to activate the corresponding process. (Pet. at 32 (citing EX1003, 41:2-5, 42:1-5), 36 & 41 (citing EX1003, 37:2-4); *see also* EX1002 ¶¶46, 117-118, 124, 136.) The search includes context stack entries of *multiple* applications: “The first task to have its dialog completed by producing a non-ambiguous query is executed and the corresponding query as interpreted by the other application is stored to activate if the recognized query is rejected by the user.” (Pet. at 36 & 41 (citing EX1003, 37:2-4); *see also* EX1002 ¶¶46, 124, 136.)³

Thus, as to the key disputed limitations, **“selecting at least one domain agent based on the determined meaning”** and then **“invoking the selected domain agent to process the formulated request,”** the Petition explained that these steps are met by Coffman’s disclosure of activating an application based on the application’s association with the context and meaning of the user’s input, and

³ All emphasis added to quoted material unless noted.

having the application carry out the user’s request. (Pet. at 36 (citing EX1003, 42:1-7 (“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”)), 41 (citing EX1003, 37:2-4 (“The first task to have its dialog completed by producing a non-ambiguous query is executed....”)); EX1002, ¶¶124, 136.)

The Decision narrowly focused on these “**selecting**” and related “**invoking**” limitations. (Decision at 22-29.) Contrary to unrebutted evidence and without the benefit of a full record, the Decision found that the Petition failed to sufficiently show it was a conversationally aware application (“**domain agent**”) that was **selected** and **invoked** (as opposed to something else getting selected and invoked). (*Id.* at 25-28, 29.) The key issue appears to be that Coffman’s disclosures that the “tasks,” “processes” or “threads” associated with context stack entries and executed or activated as a result of matching a context stack entry were not, in the Panel’s view, sufficiently tied to corresponding conversationally aware applications. (*Id.* at 27 (“Petitioner does not explain how Coffman’s disclosure of activating a process after traversing context stack 405 in the CVM teaches a conversationally aware application executing software that selects a domain agent.”).)

As explained below, Petitioner respectfully submits that the Petition and supporting evidence amply demonstrated the tie between tasks/processes/threads and conversationally aware applications—and that Coffman therefore discloses and

renders obvious the claimed “**selecting**” and “**invoking**” steps. Any factual dispute should have been resolved on a full post-institution record.

III. ARGUMENT

*A. The Petition Sufficiently Demonstrated, For Purposes of Institution, that Coffman Discloses and Renders Obvious “**Selecting**” and “**Invoking**” a “**Domain Agent**”*

To start, the Petition sufficiently showed that a task/process/thread associated with a context stack entry in Coffman corresponds to an *application*. As discussed in II.B above, the Petition, supported by a full expert report, noted that Coffman discloses that context stack entries are created by “each new task, process, or thread”—which the Petition specifically linked to an application by noting that an “application process” would be an example of an “active process.” (Pet. at 32 (citing EX1003, 29:18-30:10); *see also* EX1003, ¶166.) The Petition and corresponding un rebutted expert testimony linked a task/process/thread to an application by noting Coffman’s statements that each task/process/thread is “associated with a discourse” and that “[e]ach application can be associated with multiple discourses.” (Pet. at 31-32 (citing EX1003, 30:4-10); EX1002 ¶116.) Of course, a task/process/thread is not *exactly* the same as an application—rather, they describe a *running* application. This would have been common knowledge even to everyday Microsoft Windows users who have typed “CTRL-ALT-DEL” to open the “Task Manager” to view information about (and force quit) application processes. This common knowledge

is reflected in Coffman's "BACKGROUND" section, which describes how Microsoft Windows provides "multi-tasking" by "displaying each process as a separate window." (EX1003, 1:26-28.) Coffman explicitly links a task/process/thread to an application by explaining, as the Petition noted, that "[t]he first task to have its dialog completed by producing a non-ambiguous query is executed and the corresponding query as interpreted by the other application is stored to activate...." (Pet. at 36 & 41 (citing EX1003, 37:2-4).) Coffman's similar statements about having "a natural dialog...across multiple tasks, processes and discourses" and having "a natural dialog...across multiple applications" likewise show this link. (*Compare* EX1003, 6:6-8 (cited in Pet. at 20) *with* EX1003, 36:29-31 (cited in Pet. at 34).)

The Petition also sufficiently showed that a task/process/thread associated with a context stack entry in Coffman corresponds not only to an application generally, but to a *conversationally aware application*. As the Petition noted, Coffman explains that the "core of the CVM 401 is the context stack" and that conversationally aware applications are specifically programmed for operating with a CVM. (Pet. at 32 (citing EX1003, 29:18-30:3), 33-34 (citing EX1003, 13:26-29, 58:12-20).) The Petition noted how conversationally aware applications register with a CVM by providing data files to the CVM, which Coffman elsewhere discusses in connection with a task/process. (*Compare* Pet. at 24-25 (citing EX1003,

57:22-58:11 (conversationally aware application “register[s] all its conversational capabilities and requirements” including “[a]ctive vocabulary, finite state grammar and language models to control the application” and “[s]ymbolic mapping if NLU is supported...”) *with id.* at 31-32 (citing EX1003, 29:18-30:3 (“...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)”)).) The Petition noted that a context “discourse”—with which an application can be associated—is a “conversational interaction associated with a given task/process/thread.” (*Id.* at 31-32 (citing EX1003, 29:18-30:3); EX1002 ¶116.)

These disclosures, all detailed in the Petition and unrebutted expert testimony, should leave little doubt that Coffman teaches that it is a *conversationally aware application* that is selected (and ultimately invoked)—and that Coffman therefore discloses and renders obvious the “**selecting**” and “**invoking**” steps.

B. The Decision’s Understanding of Coffman Is Wrong, But Even If Correct, It Should Not Have Resulted in Institution Denial.

Against the Petition’s detailed analysis and expert support, Patent Owner offered only attorney argument, with no opposing expert. The Decision improperly credited Patent Owner’s attorney argument, and engaged in its own unsupported speculation, about the significance of isolated disclosures in Coffman, over the Petition and the considered testimony of Petitioner’s expert. *Icon Health & Fitness*,

Inc. v. Strava, Inc., 849 F.3d 1034, 1043 (Fed. Cir. 2017) (“attorney argument is not evidence”); *Vizio, Inc. v. Maxell, Ltd.*, IPR2022-01463, Paper 7 at 22 (PTAB) (crediting uncontested expert testimony and granting institution); *Tesla, Inc. v. Graphite Charging Co.*, IPR2024-00388, Paper 14 at 15 (PTAB Aug. 27, 2024) (same); *Hilti AG v. Specified Techs. Inc.*, IPR2024-00192, Paper 16 at 26-27, 33-34 (PTAB May 14, 2024) (instituting IPR, rejecting argument that petition and supporting expert with similar verbiage were “conclusory” because prior art disclosures were sufficiently identified and mapping was reasonable); *Hum Indus. Tech., Inc. v. Amsted Rail Co.*, IPR2023-00540, Paper 10 at 37 (PTAB Aug. 11, 2023) (expert testimony not conclusory because testimony supported by citations to record evidence). As explained below, Patent Owner’s and the Decision’s understanding of Coffman is wrong; but even if correct, it at most raised factual issues to resolve on a more complete post-institution record. See, e.g., *Samsung Elecs. Am., Inc. v. Snik LLC*, IPR2020-01324, Paper 9 at 20 (PTAB Mar. 9, 2021) (“At most, Patent Owner’s alternative reading of [prior art] raises a factual dispute best resolved on a full record.”); *Permian Global Inc. v. Fuel Automation Station, LLC*, IPR2023-01238, Paper 11 at 17 (PTAB Feb. 28, 2024) (instituting IPR where dispute that petition’s “conclusory” statement did not include “much analysis” but was “supported by unrebutted testimony” was “best resolved after further development of the record”).

Turning to the points raised in the Decision, it first focused on Coffman’s statement that “[e]ach action, completed query and conversational argument of an active task as well as each returned value/result on stored on the context stack 405” and asserted that this statement “suggests that traversing the context stack concerns determining the context based on applications that have already been selected.” (Decision at 27.) Although not entirely clear, the Decision may be focusing on the words “active task” and equating a task being “active” with a task being “selected.” The Decision’s point may therefore be that an application having a context stack entry cannot be further “selected” because it was already “selected” in connection with simply having a context stack entry.

But Coffman distinguishes between a task being “active” and the “activation” and execution that occurs as a result of matching a corresponding context stack entry that the Petition pointed to in connection with the “**selecting**” step. The Decision ignores, for example, Coffman at 37:2-4, which was cited repeatedly in the Petition. That passage explains that “[t]he first task to have its dialog completed by producing a non-ambiguous query is executed and the corresponding query as interpreted by the other application is stored to activate if the recognized query is rejected by the user.” (EX1003, 37:2-4 (cited in Pet. at 36 & 41).) What Coffman is describing is selecting from among running applications (“active tasks”) to determine which of those running applications should process a user’s request—consistent with

Coffman’s teaching that “[i]t is the conversational computing paradigm that allows a user to seamlessly control multiple Windows applications, for example, running in parallel....” (EX1003, 4:13-16.) This is similar, for example, to your e-mail, spreadsheet, and web browser applications all being open, and determining which of those running applications should be in “focus” to perform a particular action. (See EX1003, 3:14-16 (describing how speech applications “allow[] a user to change the focus” and “give voice commands to the task in focus”).)

The Decision next asserted that “Coffman’s traversal of a context stack merely concerns which process to execute next” and that “the selection of which process to execute next occurs after the application has been selected.” (Decision at 27-28.) It appears that, in the Panel’s view, Coffman’s “processes” are associated with tasks and threads performed at the CVM, and they do not correspond to conversationally aware applications. (Decision at 28.) But the Panel’s view makes little sense because, as the Petition noted, Coffman describes a process becoming active *after* CVM and conversational subsystem processing (such as NLU and context stack search) has been performed:

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.

(EX1003, 42:1-7 (cited in Pet. at 36).) As discussed in III.A above, the totality of the Petition’s analysis and collected citations to Coffman, supported by Petitioner’s expert, showed that the applications/tasks/processes/threads associated with context stack entries correspond to conversationally aware applications. (*See also* EX1003, 31:24-29 (“A dialog controller 404 manages the dialog...across all the conversational and conventional applications (registered with the task dispatcher/controller 402).... The dialog controller 404 manages and determines the active context and application.”), 65:12-13 (claim 11) (“a dialog manager for managing dialog across the applications and for selecting an active dialog, context and application based on input queries”).)

In fact, the whole point of Coffman is to enable voice and multi-modal user interaction with conversationally aware applications. For example, Coffman describes how a “personal vehicle assistant” (PVA) “runs an embedded CVM while the two other applications are conversationally aware, i.e., you can control them through the CVM running on the PVA.” (EX1003, 56:22-25 (cited in Pet. at 23, 24); *see also id.*, 4:13-16 (“It is the conversational computing paradigm that allows a user to seamlessly control multiple Windows applications, for example, running in parallel....”).) There should be no doubt that such user interaction involves invoking those applications specifically. For instance, Coffman provides an illustration of the

conversational system, explaining how it allows a user to “dial in your PC to consult or modify some spreadsheet information; the same way that you would 20 have consulted it by voice on your desktop while processing your mail.” (EX1003, 57:17-20; *see also* Pet. at 23 & 24 (citing EX1003, 58:30-32 (describing conversationally aware spreadsheet application)).) If a user asks for the value of a particular spreadsheet cell, wouldn’t the CVM select the conversationally aware spreadsheet application to lookup and provide that information? Coffman nowhere suggests such application-specific functionality is moved to or duplicated by the CVM or its subsystems. Coffman does describe “conversational foundation classes,” as the Decision notes (Decision at 27), but those are components that are built or linked into an application to help make the application “conversationally aware”—*not* to move (or duplicate) actual application-specific functionality to the CVM. (EX1003, 13:29-32 (“[T]he conversational application API layer 203 comprises a plurality of conversational foundation classes 205 (or fundamental dialog components) which are provided to the application developer through library functions that may be used to build a CUI or conversationally aware applications 200 according to the present invention.”); *see also id.*, 13:21-28.)

The Decision next asserted that “Petitioner has not shown that the CVM’s decision about which application is already active and has made a call to the CVM constitutes selecting a domain agent based on the determined meaning of an

utterance.” (Decision at 28.) As discussed above and in the Petition, Coffman teaches using a CVM and context stack to select which application from among (running) applications should process a user request. Coffman is not limited to only determining what actions a single application should perform. In fact, Coffman touts its system as an improvement over such prior art systems that were limited to “giv[ing] commands to the task in focus.” (EX1003, 3:14-16.)

Turning to the “**invoking**” limitation, the Decision again asserts that “a conversational subsystem, rather than a conversationally aware application executes the request.” (Decision at 29.) But again, as discussed above and in the Petition, Coffman teaches using a CVM and context stack to select which application from among (running) applications should process a user request. The processing provided by the CVM and conversational subsystems (such as speech recognition, NLU and context stack search) precedes the selection and invoking of a conversationally aware application that will actually carry out the user’s request (such as performing a spreadsheet lookup). In any event, these factual disputes should have been carried forward for a decision on a full record.

IV. CONCLUSION

For the reasons above and in the Petition, Petitioner respectfully requests Director Review of the Decision, and for an order vacating the Decision and instituting *inter partes* review.

Dated: November 14, 2024

Respectfully submitted,

COOLEY LLP
ATTN: Patent Group
1299 Pennsylvania Avenue NW
Suite 700
Washington, DC 20004
Tel: (650) 843-5001
Fax: (650) 849-7400

By: / Heidi L. Keefe /
Heidi L. Keefe
Reg. No. 40,673
Counsel for Petitioner

CERTIFICATE OF SERVICE

I hereby certify, pursuant to 37 C.F.R. Section 42.6, that a complete copy of the attached **PETITIONER'S REQUEST FOR DIRECTOR REVIEW OF INSTITUTION DECISION** is being served via electronic mail on the 14th day of November 2024 upon Patent Owner's counsel of record in this proceeding as follows:

Wayne Helge
James Wilson
BUNSOW DE MORY, LLP
whelge@bdiplaw.com
jwilson@bdiplaw.com
BDIP-DialectIPRs@bdiplaw.com

DATED: November 14, 2024

/ Heidi L. Keefe /
Heidi L. Keefe
Reg. No. 40,673
COOLEY LLP
1299 Pennsylvania Ave. NW
Suite 700
Washington, D.C. 20004
Tel: (650) 843-5001