

Petition for *Inter Partes* Review of
U.S. Patent No. 7,502,738 B2

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

GOOGLE LLC,
Petitioner

v.

DIALECT, LLC,
Patent Owner

Case IPR2024-00747
U.S. Patent No. 7,502,738 B2
Issue Date: March 10, 2009

Title: SYSTEMS AND METHODS FOR RESPONDING TO
NATURAL LANGUAGE SPEECH UTTERANCE

**PETITION FOR *INTER PARTES* REVIEW
OF U.S. PATENT NO. 7,502,738 B2**

Table of Contents

	Page
I. MANDATORY NOTICES UNDER §42.8(A)(1)	10
A. Real Party-In-Interest under §42.8(b)(1).....	10
B. Related Matters under §42.8(b)(2)	10
C. Lead and Back-Up Counsel under §42.8(b)(3)	11
D. Service Information	12
II. FEE PAYMENT	12
III. REQUIREMENTS UNDER §§ 42.104 AND 42.108 AND CONSIDERATIONS UNDER §§ 314(A) AND 325(D).....	12
A. Grounds for Standing	12
B. Identification of Challenge and Statement of Precise Relief Requested	12
C. Considerations Under §§ 314(a) and 325(d).....	13
IV. OVERVIEW OF THE CHALLENGED PATENT.....	14
A. Person of Ordinary Skill in the Art (POSITA)	14
B. Specification Overview	14
V. CLAIM CONSTRUCTION	16
VI. THE CHALLENGED CLAIMS ARE UNPATENTABLE.....	16
A. Overview of Grounds	16
B. Ground 1: Claims 1-11, 18, 20-23, 26, 31-37 and 39 are Obvious Over Coffman in View of Kanevsky and Ronning.	18
1. Claim 1: “A system responsive to a user generated natural language speech utterance, comprising:”	18
(a) “an agent architecture that includes a plurality of domain agents, each of the plurality of domain agents being an autonomous executable configured to receive, process, and respond to requests associated with a respective context;” (1[a]).....	21

Table of Contents (continued)

Page

(b)	“a parser configured to determine a context for one or more keywords contained in the utterance and to determine a meaning of the utterance based on the determined context, wherein the parser 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;” (1[b])	26
(c)	“an event manager configured to coordinate interaction between the parser and the agent architecture; and” (1[c]).....	34
(d)	“an update manager that enables the user to purchase one or more domain agents from a third party on a one-time or subscription basis.” (1[d]).....	37
2.	Claim 2: “The system according to claim 1, further comprising a speech unit that receives the utterance from the user.”	42
3.	Claim 3: “The system according to claim 2, wherein the speech unit includes a speech coder that creates a digitized speech signal from the utterance.”	44
4.	Claim 4: “The system according to claim 1, wherein the utterance contains at least one request that includes at least one of a query or a command.”	46
5.	Claim 5: “The system according to claim 1, further comprising a speech recognition engine that recognizes words and phrases contained in the received user utterance using information in one or more dictionary and phrase tables, wherein the parser examines the recognized words and phrases to identify the one or more keywords.”	47

Table of Contents
(continued)

	Page
6. Claim 6: “The system according to claim 5, wherein the speech recognition engine further recognizes an identity of the user based on information contained in one or more user profiles.”	53
7. Claim 7: “The system according to claim 1, further comprising a text to speech engine that creates one or more utterances to be presented as output to the user.”	55
8. Claim 8: “The system according to claim 7, further comprising a personality module that includes one or more characteristics for formatting the utterances to be presented as the output to the user.”	56
9. Claim 9: “The system according to claim 1, wherein the parser determines the context for the one or more keywords by applying prior probabilities or fuzzy possibilities to one or more of keyword matching, a user profile, or a dialog history.”	57
10. Claim 10: “The system according to claim 1, wherein at least one of the plurality of domain agents creates a query to at least one local or external information source when processing a request that includes a question.”	59
11. Claim 11: “The system according to claim 1, wherein at least one of the plurality of domain agents directs a command to the system or an external entity when processing a request that includes a command.”	61
12. Claim 18: “The system according to claim 1, wherein the agent architecture further includes a system agent that provides default functionality and basic services, wherein each of the plurality of domain agents can use the functionality and services of the system agent.”	62

Table of Contents
(continued)

Page

13.	Claim 20: “The system according to claim 18, wherein the agent architecture further includes an agent library that contains one or more utilities for commonly used functions, wherein each of the system agent and the plurality of domain agents can use the utilities contained in the agent library.”	66
14.	Claim 21: “The system according to claim 20, wherein the utilities contained in the agent library include at least one of text and string handling, network communications, database lookup and management, fuzzy and probabilistic evaluation, or text to speech formats.”	67
15.	Claim 22: “The system according to claim 20, wherein the update manager uses a network interface to download, distribute, and install updates relating to one or more of the system agent, the plurality of domain agents, the agent library, one or more databases available to the agent architecture, or entries in the dictionary and phrase tables.”	69
16.	Claim 23: “The system according to claim 22, wherein the updates include one or more of a new domain agent, additional domain knowledge for a domain agent, new keywords for a domain agent, preferred information sources for a domain agent, updated domain information for a domain agent, or updated content for a domain agent.”	70
17.	Claim 26	70
18.	Claim 31	70
19.	Claim 32: “The system according to claim 1, further comprising an extension and modification facility that allows users and content developers to configure behavior of the agent architecture.”	70

Table of Contents

(continued)

	Page
20. Claim 33: “The system according to claim 32, wherein the extension and modification facility can be used to extend or modify behavior of the plurality of domain agents.”	72
21. Claim 34: “The system according to claim 33, wherein the users and content developers can add one or more of new commands, keywords, information, or information sources to the agent architecture to extend or modify the behavior of the plurality of domain agents.”	72
22. Claim 35: “The system according to claim 32, wherein the extension and modification facility provides a generic agent that can be used to create a new domain agent.”	73
23. Claim 36: “The system according to claim 32, wherein the users and content developers can configure the behavior of the agent architecture using one or more of a scripting language, a graphical user interface, or natural language utterances.”	74
24. Claim 37: “The system according to claim 32, wherein the users and content developers can create one or more of special reports, packages of queries, alerts, or output formats using the extension and modification facility.”	75
25. Claim 39	76
C. Ground 2: Claims 16, 17, 25, 30 and 38 Are Obvious Over Ground 1 Prior Art in Further View of Zadrozny	76
1. Claim 16: “The system according to claim 1, wherein the parser formulates at least one request in accordance with a grammar that the selected domain agent uses to process requests associated with the context for the one or more keywords contained in the utterance.”	76

Table of Contents
(continued)

	Page
2. Claim 17: “The system according to claim 16, wherein the parser provides the formulated request to the event manager to invoke the selected domain agent, and the event manager sends and receives one or more events to the agent architecture to process the formulated request, the event manager thereby coordinating interaction between the parser and the agent architecture.”	81
3. Claim 25	83
4. Claim 30	83
5. Claim 38	83
VII. CONCLUSION.....	83
CERTIFICATE OF SERVICE	86

List of Exhibits

Exhibit No.	Description of Document
1001	U.S. Patent No. 7,502,738 B2 to Robert Kennewick et al. (filed May 11, 2007, issued March 10, 2009) (“ 738 ” or “ 738 patent ”)
1002	Declaration of Christopher M. Schmandt (“ Schmandt ”)
1003	International Patent Application Publication No. WO 00/20962 to Daniel Coffman, Stephane H. Maes et al. (filed October 1, 1999, published April 13, 2000) (“ Coffman ”)
1004	U.S. Patent No. 5,897,616 to Dimitri Kanevsky and Stephane Herman Maes (filed June 11, 1997, issued April 27, 1999) (“ Kanevsky ”)
1005	<i>RESERVED</i>
1006	Wlodek Zadrozny et al., “Natural Language Understanding with a Grammar of Constructions”, Proc. 15th Int’l Conf. on Computational Linguistics (Vol. 2) (1994) (“ Zadrozny ”)
1007	International Patent Application Publication No. WO 00/21075 to Stephane H. Maes et al. (filed October 1, 1999, published April 13, 2000) (“ Maes ”)
1008	Excerpts from Dafydd Gibbon et al., <i>Handbook of Multimodal and Spoken Dialogue Systems : Resources, Terminology, and Product Evaluation</i> (2000) (“ Gibbon ”)
1009	Excerpts from <i>Microsoft Computer Dictionary</i> (5th ed. 2002)
1010	U.S. Patent No. 9,817,650 B2 to Joel A. Ronning et al. (filed April 27, 2012, issued November 14, 2017) (“ Ronning ”)
1011	U.S. Patent Application Ser. No. 09/492,846, filed January 27, 2000 (parent/priority application to Ronning) (“ ’846 Application ”)
1012	Redline comparison between Ronning and ’846 Application

Petition for *Inter Partes* Review of
U.S. Patent No. 7,502,738 B2

List of Exhibits

Exhibit No.	Description of Document
1013	<i>RESERVED</i>
1014	U.S. Patent Application Publication No. 2003/0144846 to Lawrence A. Denenberg and Christopher M. Schmandt (filed January 31, 2002, published July 31, 2003) (“ Denenberg ”)
1015	Excerpts from <i>Merriam-Webster’s Collegiate Dictionary</i> (10th ed. 1994)
1016	Excerpts from Richard F. E. Sutcliffe et al., <i>Industrial Parsing of Software Manuals</i> (1996) (“ Sutcliffe ”)
1017-1019	<i>RESERVED</i>
1020	Declaration of Ingrid Hsieh-Yee, Ph.D.
1021	Prosecution history of U.S. Patent No. 7,502,738 B2
1022	Proof of Service of Complaint, <i>Dialect, LLC v. Google LLC</i> , Case No. 1:23-cv-00378 (D. Del.)
1023	International Preliminary Examination Report for Maes (EX1007) application

I. MANDATORY NOTICES UNDER §42.8(A)(1)

A. Real Party-In-Interest under §42.8(b)(1)

Google LLC (“Petitioner”) is the real party-in-interest to this IPR petition.¹

B. Related Matters under §42.8(b)(2)

The ’738 patent is the subject of pending litigation in which Petitioner is named as a defendant, *Dialect, LLC v. Google LLC*, Case No. 1:23-cv-00378 (D. Del.) (filed April 3, 2023). Petitioner was first served with a complaint alleging infringement of the ’738 patent on April 6, 2023. (EX1022, p.002.) On March 28, 2024, Petitioner’s motion to transfer to the Northern District of California was granted. *See* Dkt. 32, Case No. 1:23-cv-00378 (D. Del.). The ’738 patent was also the subject of *Dialect, LLC v. Samsung Electronics, Co.*, Case No. 2:23-cv-00061-JRG (E.D. Tex.), which was terminated on September 11, 2023.

Petitioner is filing IPR petitions against eight patents asserted by Patent Owner in the pending litigation involving Petitioner:

U.S. Patent No.	IPR Case
7,398,209 B2	IPR2024-00746
7,502,738 B2	IPR2024-00747
8,015,006 B2	IPR2024-00748

¹ Google LLC is a subsidiary of XXVI Holdings Inc., which is a subsidiary of Alphabet Inc. XXVI Holdings Inc. and Alphabet Inc. are not real parties in interest to this proceeding.

Petition for *Inter Partes* Review of
U.S. Patent No. 7,502,738 B2

U.S. Patent No.	IPR Case
7,693,720 B2	IPR2024-00749
9,031,845 B2	IPR2024-00750
8,447,607 B2	IPR2024-00751
8,849,652 B2	IPR2024-00752
7,640,160 B2	IPR2024-00753

C. Lead and Back-Up Counsel under §42.8(b)(3)

Petitioner provides the following designation of counsel.

LEAD COUNSEL	BACK-UP COUNSEL
Heidi L. Keefe (Reg. No. 40,673) hkeefe@cooley.com COOLEY LLP ATTN: Patent Group 1299 Pennsylvania Ave. NW, Suite 700 Washington, DC 20004 Tel: (650) 843-5001 Fax: (650) 849-7400	Andrew C. Mace (Reg. No. 63,342) amace@cooley.com COOLEY LLP ATTN: Patent Group 1299 Pennsylvania Ave. NW, Suite 700 Washington D.C. 20004 Tel: (650) 843-5287 Fax: (650) 849-7400
	Mark R. Weinstein (Admission <i>pro hac vice</i> to be requested) mweinstein@cooley.com COOLEY LLP ATTN: Patent Group 1299 Pennsylvania Avenue NW Suite 700 Washington D.C. 20004 Tel: (650) 843-5007 Fax: (650) 849-7400
	Reuben Chen (Admission <i>pro hac vice</i> to be requested) rchen@cooley.com COOLEY LLP

LEAD COUNSEL	BACK-UP COUNSEL
	ATTN: Patent Group 1299 Pennsylvania Avenue NW Suite 700 Washington D.C. 20004 Tel: (650) 843-5007 Fax: (650) 849-7400

D. Service Information

This Petition is being served by Federal Express to the correspondence address of record for the '738 patent, 174899 - The Law Offices of David A. Gerasimow, P.C., 211 W. Wacker Dr., Ste. 1717, Chicago, IL 60606. Petitioner consents to electronic service at the addresses provided above for lead and back-up counsel.

II. FEE PAYMENT

Petitioner requests review of 30 claims, with a \$52,750 payment.

III. REQUIREMENTS UNDER §§ 42.104 AND 42.108 AND CONSIDERATIONS UNDER §§ 314(A) AND 325(D)

A. Grounds for Standing

Petitioner certifies that the '738 patent is available for IPR and that Petitioner is not barred or otherwise estopped.

B. Identification of Challenge and Statement of Precise Relief Requested

Petitioner requests IPR institution based on:

Petition for *Inter Partes* Review of
U.S. Patent No. 7,502,738 B2

Ground	Claims	Basis for Challenge under §103
1	1-11, 18, 20-23, 26, 31-37, 39	Coffman (EX1003), Kanevsky (EX1004), and Ronning (EX1010)
2	16, 17, 25, 30, 38	Ground 1 Prior Art + Zadrozny (EX1006)

C. Considerations Under §§ 314(a) and 325(d)

§314(a): Petitioner is unaware of any previous post-grant challenge against the '738 patent. The pending litigation involving Petitioner is still in early stages, with no discovery or claim construction having occurred and no trial date set, and as noted, was recently transferred to the Northern District of California. If IPR is instituted, Petitioner intends to bring a motion to stay the pending litigation.

§325(d): During prosecution of the '738 patent, an applicant-submitted IDS identified Coffman's U.S. counterpart (i.e., 7,137,126). (EX1021, p.00112.) It was not substantively discussed. Why the Examiner did not appreciate its significance is unclear, but it was listed among over 100 other references. (EX1021, pp.00108-00112.) The Examiner also did not have the additional references that this Petition relies on, which provide further context and detail for understanding Coffman and how its disclosures provide compelling grounds of obviousness.

IV. OVERVIEW OF THE CHALLENGED PATENT

A. Person of Ordinary Skill in the Art (POSITA)

A POSITA as of June 3, 2002² would have possessed a bachelor's degree in electrical engineering or computer science, and two years of work experience in the field of speech- and natural language-interfaces for interacting with and directing computer systems and software, including speech recognition and processing (and more experience could substitute for less formal education, and vice versa). (EX1002 ("Schmandt"), ¶¶11-15.)

B. Specification Overview

The '738 patent states that it "relates to the retrieval of online information or processing of commands through a speech interface," as illustrated:

² Petitioner does not concede that the claims are entitled to the June 3, 2002 date, but that issue is irrelevant here because the prior art here predates that date.

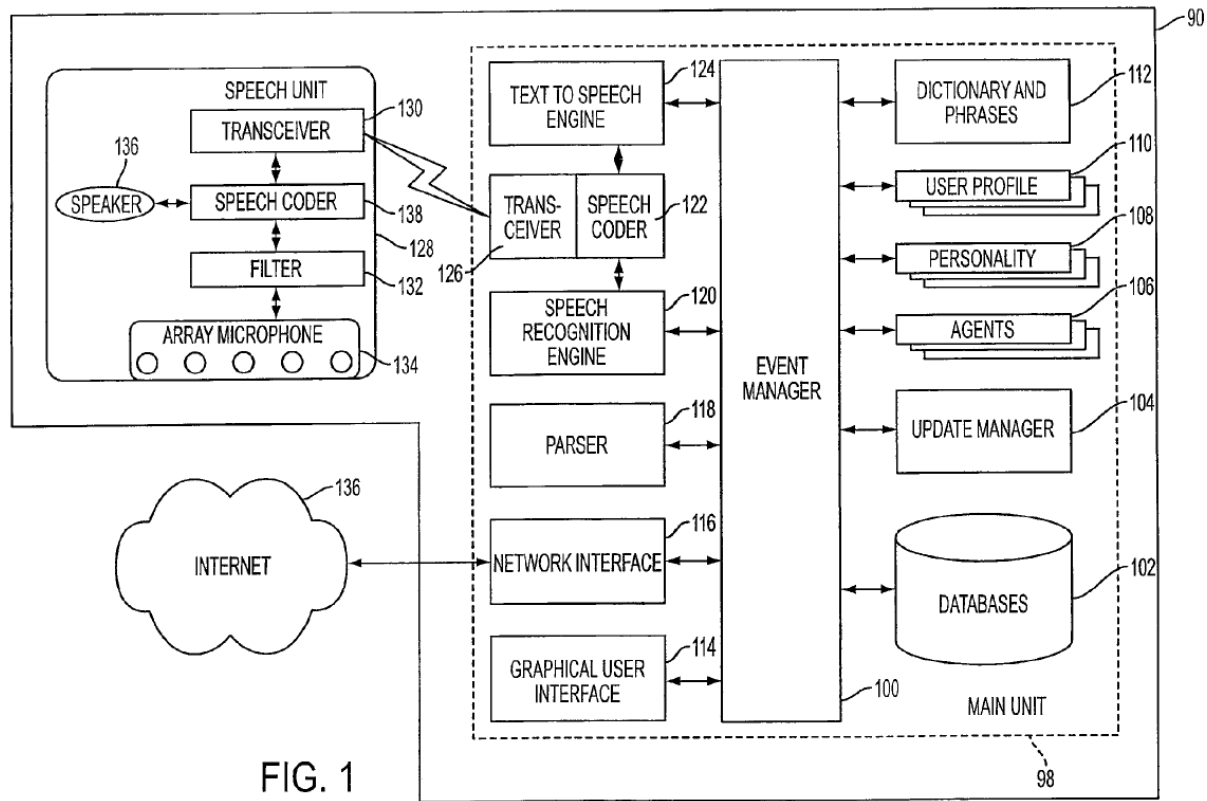


FIG. 1

(’738, Fig. 1, 1:13-15.) As shown, system **90** includes speech unit **128**, speech recognition engine **120**, dictionary and phrases **112**, parser **118**, and agents **106**. Speech unit **128** includes a microphone to receive a spoken utterance. (’738, 10:29-31.) After initial processing, the speech is passed to speech recognition engine **120** for processing “using data in the dictionary and phrases module **112** and received from the agents **106**.” (’738, 10:57-59.) Then, “[t]he recognized words and phrases are processed by the parser **118**, which transforms them into complete commands and questions using data supplied by an agent[s] **106**.” (’738, 10:60-62.) “The agent[s] **106** then process the commands or questions.” (’738, 10:62-63.) To select the correct agent to process the command or question, “parser **118** determines a

context for an utterance by applying prior probabilities or fuzzy possibilities to keyword matching, user profile **110**, and dialog history.” (’738, 17:48-50.) “Once the context for the question or command has been determined, the parser **118** can invoke the correct agent **156, 150.**” (’738, 18:11-13.)

V. CLAIM CONSTRUCTION

Express claim construction is unnecessary because the prior art renders the challenged claims obvious under any reasonable construction, but Petitioner reserves its right to respond to claim construction arguments by Patent Owner.

VI. THE CHALLENGED CLAIMS ARE UNPATENTABLE

A. Overview of Grounds

The challenged claims attempt to lay claim to well-known techniques for speech-enabled interfaces for interacting with and directing computer systems. The challenged claims employ standard techniques that were known long before the ’738 patent, such as using speech recognition and natural language processing to select an application for carrying out a query and/or command included in a spoken utterance.

All grounds rely primarily on **Coffman** (**EX1003**), which despite being filed nearly three years before the ’738 patent, discloses a remarkably similar conversational computing system. All grounds combine Coffman with **Kanevsky** (**EX1004**), which discloses known techniques related to speech recognition and

natural language processing, and **Ronning** (EX1010), which describes techniques for using a software agent for purchasing, downloading, installing and updating applications on a computer system. Ground 2 combines these references with **Zadrozny** (EX1006) for features recited in claim 16. As detailed below, Coffman, Kanevsky and Zadrozny share links to each other and thus comprise a collective body of work describing aspects of the same systems and methods—for example, Coffman expressly incorporates Kanevsky by reference (Coffman, 37:10-13), and Kanevsky in turn discusses Zadrozny (Kanevsky, 14:22-25); Coffman and Kanevsky share a common named co-inventor (Stephane Maes); and the named inventors and authors of Coffman, Kanevsky, and Zadrozny are identified as associated with IBM. *See, e.g., Paice, LLC v. Ford Motor Co.*, 881 F.3d 894, 906-907 (Fed. Cir. 2018) (incorporated material “effectively part of the host document as if it were explicitly contained therein”); *Boston Scientific Scimed, Inc. v. Cordis Corp.*, 554 F.3d 982, 991 (Fed. Cir. 2009) (“[c]ombining two embodiments disclosed adjacent to each other in a prior art patent does not require a leap of inventiveness”).

The references are prior art to the ’738 patent under at least pre-AIA §§102(b) (Coffman, Kanevsky, Zadrozny) and 102(e) (Ronning). (*See* Schmandt, ¶54 (Ronning); EX1012; EX1020, ¶¶25-48 (Zadrozny).)

B. Ground 1: Claims 1-11, 18, 20-23, 26, 31-37 and 39 are Obvious Over Coffman in View of Kanevsky and Ronning.

1. Claim 1: “A system responsive to a user generated natural language speech utterance, comprising:”

As explained in more detail for the limitations below, Coffman discloses a conversational computing “**system.**” (Coffman, 36:29-31, 5:1-4, 13:8-11.) The system is “**responsive to a user generated...speech utterance**”—Coffman explains that the system performs processing in response to receiving spoken input from a person. (Coffman, e.g., 53:8-10 (“Voice input from a microphone (e.g. the standard voice input) can be arguments of function calls with the sequence of words, phones, or user identity or queries (symbolic language representation provided by NLU).”), 37:14-22 (“Spoken input from a speech client (e.g., telephone 600) is subject to a speech recognition process 604 and other input (e.g., keyboard, mouse clicks etc) are subject to NLU processing 605.”), 41:2-5 (“speech inputs being transcribed/understood”), 53:22-26 (“speech or any other input stream”).)³ Figure 6 of Coffman illustrates:

³ All underlining added to quoted material unless noted.

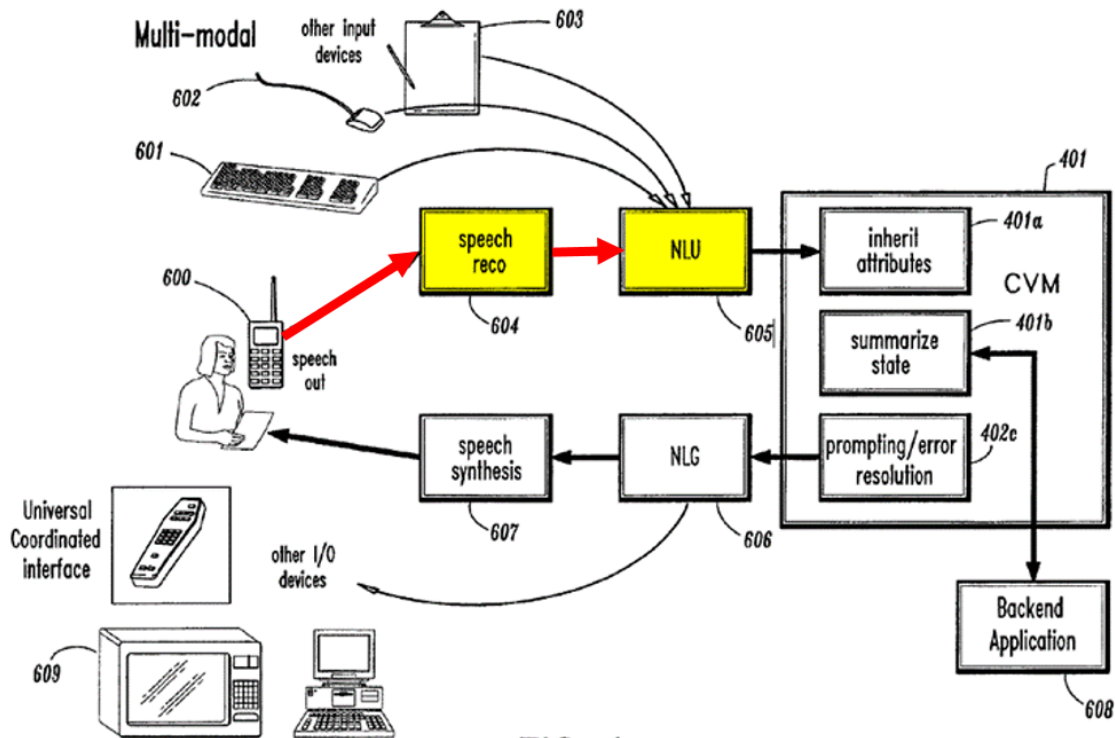


FIG. 6

(Coffman, Fig. 6 (annotations added); *see also id.*, 37:14-22.)

Coffman further discloses receiving spoken input that is a “**natural language** **speech utterance.**” In the context of the ’738 patent, and speech recognition and processing generally, “**natural language**” generally refers to the forms of language used by a person for communication, as opposed to artificial or formal languages such as programming languages. (See, e.g., Schmandt, ¶64 (citing Gibbon (EX1008), p.0035 (definition of “natural language”)).) For example, English is a natural language, while Java is a programming language. (*Id.*) Consistent with this understanding, Coffman discloses several examples of natural language speech utterances, such as “I would like to open this,” “it should be synchronized with my PDA,” and “e-mail from my mother.” (Coffman, 6:32-7:3 (“For example, a mouse

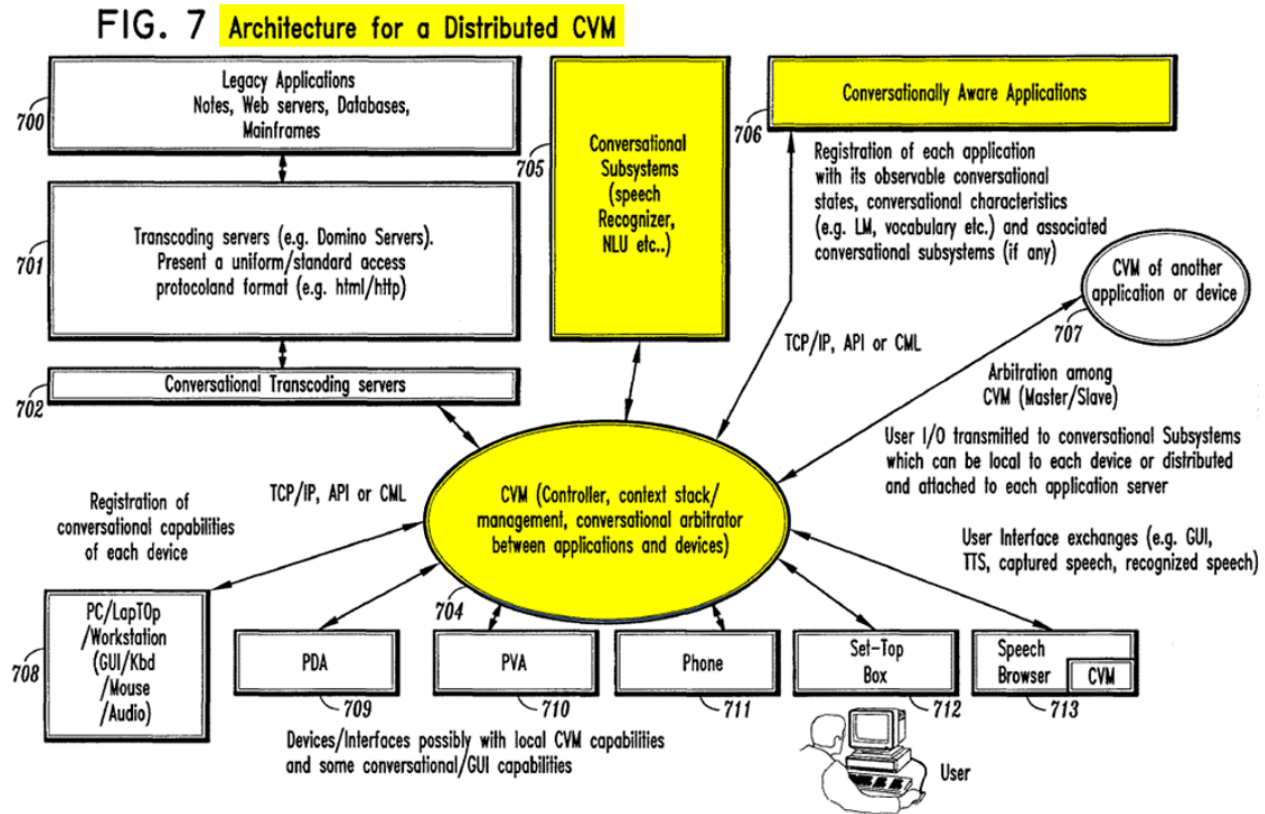
click or pointer/stylus pointing action followed by the command ‘I would like to open this’ is disambiguated into a set of attribute value pairs: Command: open, Object: Windows or task selected by the last mouse click.”), 48:1-3 (“e-mail from my mother is correctly understood, independently of the user.”), 56:25-57:2 (“grocery list,” “it should be synchronized with my PDA,” and “Do it!”); Schmandt, ¶64.)

Coffman’s receipt (and use) of spoken “**natural language**” input is also confirmed by Coffman explaining that the spoken input, once recognized, is subjected to natural language parsing and understanding (“NLU”) processes, as illustrated by Figure 6. (Schmandt, ¶65; Coffman, e.g., Fig. 6, 53:22-26 (“User inputs can be represented by their transcriptions or their mappings into a symbolic language after parsing and NLU.”), 53:8-10 (“Voice input from a microphone (e.g. the standard voice input) can be arguments of function calls with the sequence of words, phones, or user identity or queries (symbolic language representation provided by NLU).”), 37:14-22 (“NLU processing 605”), 6:6-8 (“natural dialog with NLU”), 5:1-4, 37:29-32.)

Coffman therefore discloses “[a] **system responsive to a user generated natural language speech utterance**,” and as shown below, the claimed system is obvious. (Schmandt, ¶66.)

- (a) **“an agent architecture that includes a plurality of domain agents, each of the plurality of domain agents being an autonomous executable configured to receive, process, and respond to requests associated with a respective context;” (1[a])**

Coffman’s conversational computing system comprises **“an agent architecture that includes a plurality of domain agents”** that includes a conversational virtual machine (“CVM”) and conversational subsystems (such as a speech recognition engine), and a plurality of “conversationally aware applications” (**“plurality of domain agents”**). (Schmandt, ¶67.) Coffman explains, for example, that its 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 conversationally aware applications are “specifically programmed for operating with a CVM core layer (or kernel) 14 via conversational application APIs 13.” (Coffman, 12:28-30.) Figure 7 of Coffman, for example, “illustrates an architecture for a distributed CVM,” where CVM 704 “acts as the conversational arbitrator between a plurality of applications 706, devices, 708-713, other CVM applications or devices 705”:



(Coffman, Fig. 7 (highlighting added), 55:12-16; *see also id.*, Fig. 4, 28:10-15.)

The “**plurality of domain agents**” corresponds to the plurality of conversationally aware applications in Coffman’s system. The ’738 patent describes “agents” as “autonomous executables that receive, process and respond to user questions, queries and commands” and states that they “provide complete, convenient and re-distributable packages or modules of functionality, typically for a specific domain or application.” (’738, 2:51-58.) In the software context, an “agent” generally broadly refers to a program that performs actions on behalf of another entity, such as a user or another software program or computing system. (Schmandt,

¶68 (examples of web browser as “HTTP user agent” and desktop email software as “mail user agent” or MUA).)

The executable software of a conversationally aware application (and associated data, such as vocabulary, grammar, language models and NLU symbolic mapping) is a “**domain agent**.” (Schmandt, ¶69; Coffman, 13:26-29.) Coffman’s conversationally-aware application receives, processes, and responds to user questions, queries and commands. (Coffman, 13:8-13 (queries transformed into “application calls” and application output returned to user), 37:2-4 (task executes query), 40:9-19 (“execute completed queries” and “output of executed or running processes”).) Coffman’s conversationally-aware application can also provide functionality related to a specific domain, such as a particular subject matter (such as email or spreadsheet) or device (such as a PDA or smart phone). (Coffman, 16:35-17:7 (listing “application specific” foundation classes, including “Manage_e-mail,” “Manage_calendar,” and “Manage_addressbook/director”), 56:11-14 (e-mail), 58:30-32 (spreadsheet), 56:22-25 (“conversational PDA,” “conversational car PC (PVA, Personal Vehicle Assistant),” and “conversational smart phone”).) Coffman also describes modular development tools for conveniently building complete and re-distributable applications. (Coffman, 58:12-20; *see also id.*, 58:27-28, 7:29-8:9.) The applications can be distributed for installation by conversational systems.

(Coffman, 52:9-11 & 52:18-21 (describing agents to upgrade or install applications).)

A conversationally-aware application executable is a “**domain agent**” because, as explained, it provides functionality for a particular area or realm, such as a particular subject matter or a device. (Schmandt, ¶70; Coffman, 16:35-17:7, 56:11-14, 58:30-32, 56:22-25, 36:29-31 (“[T]he present invention provides the capability to have a natural dialog with NLU, NLG and mixed initiative across multiple applications, with multiple domains.”).)

1[a] further recites “**each of the plurality of domain agents being an autonomous executable.**” A conversationally-aware application (“**domain agent**”) is an “**autonomous executable**” because it automatically (without user involvement) registers with a conversational system to be able to receive, process and respond to requests, and moreover, operates automatically in response to the system selecting it to handle a request. (Schmandt, ¶71.) Coffman explains, for example, that “conversational devices or applications [] register their conversational capabilities” and describes “automatically discover[ing] local and network conversationally aware systems.” (Coffman, 7:12-14, 25:18-24; *see also id.*, 23:16-20.) And as discussed in more detail for 1[b], Coffman also explains how “[t]he first task to have its dialog completed by producing a non-ambiguous query is executed” (Coffman, 37:2-4), making clear that an application is automatically selected and

carries out a request as a result of processing by the conversational system. (*See also id.*, 42:1-7; Schmandt, ¶71.)

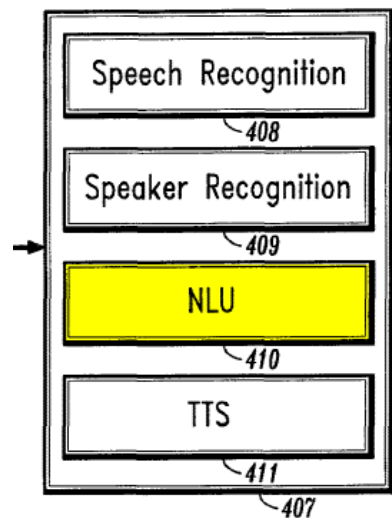
The autonomous nature of Coffman’s conversationally-aware applications is confirmed by **Maes (EX1007)**, which Coffman incorporates by reference and which claims priority to the same provisional patent applications as Coffman. (Coffman, 22:25-29; Coffman 1:2-3 & Maes, 1:4-5; EX1023 (Maes application corresponds to “YO999-113P”); Schmandt, ¶72.) Coffman specifically points to Maes for aspects of its conversational computing system. (Coffman, 22:25-29, 26:21-25, 45:11-22, 61:5-6.) Maes describes techniques for conversationally-aware applications to automatically communicate and coordinate with each other. (Maes, 3:4-10 (describing “automatically shar[ing] conversational resources automatically and in a coordinated and synchronized manner so as to provide a seamless conversational interface through an interface of one of the network devices.”).)

1 [a] further recites that each domain agent is “**configured to receive, process, and respond to requests associated with a respective context**.” As detailed for 1 [b] below, Coffman explains that its conversational system associates an application and user request with a “context” to determine the application and formulate the associated query/command that should be carried out in response to a user’s request. (Schmandt, ¶73; Coffman, e.g., 29:18-30:3 (describing “context stack” and “context”), 30:4-10 (“Each context associated with a given discourse

comprises the latest requests made to the corresponding process/task/thread as well as the latest output.”), 42:1-5, 41:2-5.)

- (b) **“a parser configured to determine a context for one or more keywords contained in the utterance and to determine a meaning of the utterance based on the determined context, wherein the parser 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;” (1[b])**

Coffman discloses **“a parser.”** (Schmandt, ¶74.) As mentioned for 1[a], Coffman’s conversational system includes various “conversational subsystems”—including “a natural language understanding and natural language parsing system [NLU] 410” (illustrated at right (Fig. 4 (excerpt))). (Coffman, 29:1-4, Fig. 4.) Coffman explains that, “using NLU technology,” “[q]ueries can be decoded, parsed and then translated into a logic combination of queries (symbolic query).” (Coffman, 51:12-14; *see also id.*, 40:12-15 (“different statistical parsers” for operating on a query).)



A POSITA would have further understood and found it obvious that Coffman’s parser is **“configured to determine a context for one or more keywords contained in the utterance.”** As explained below, Coffman uses NLU

(i.e., parsing) as part of the process to determine a “context” for the user’s request that matches keywords recognized in the user’s utterance. (Schmandt, ¶75.)

Coffman’s system includes a “context stack” where the “context,” including the state and history of each active process (such as an application process) is tracked. (Coffman, 29:18-30:3, 30:4-10.) Coffman further discloses that, in order to determine a “context” for processing a user’s request, NLU processing (i.e., parsing) is used to match recognized words in the user’s utterance to a context. (Coffman, 42:1-5, 41:2-5, 30:8-10, 51:12-14, 37:20-26; Schmandt, ¶77.)

A POSITA would have understood and found it obvious that matching recognized words in the user’s utterance to a context would involve “**keywords contained in the utterance.**” (Schmandt, ¶78.) For example, Coffman describes matching the word “open” (“**keyword**”) in user’s spoken utterance “I would like to open this” to an “open” command in an application’s symbolic language associated with a context. (Coffman, 6:32-7:3.) This example illustrates how Coffman’s teachings about “fit[ting] the utterance at the level of the NLU symbolic language” and using NLU “symbolic language (attribute value n-uple) [to map] the actual query” involve “**keywords contained in the utterance.**”⁴ (Schmandt, ¶78; Coffman, 42:1-5, 30:20-22; *see also id.*, 37:20-26.) Coffman’s example is an

⁴ An “n-uple” refers to a collection of “n” elements; for example, a “4-uple” (or “quadruple”) is a collection of four elements. (Schmandt, ¶78 n.5.)

instance of the well-known concept of “keyword detection,” also known as “word spotting.” (Schmandt, ¶78 (citing EX1008, p.0029 (defining “keyword detection”))).) Moreover, Coffman incorporates by reference U.S. Patent No. 5,897,616 (**Kanevsky (EX1004)**) (Coffman, 37:10-13), which explains that NLU can be implemented using “a combination of classical parsing capabilities with key word spotting and speech recognition.” (Kanevsky, 14:1-7.) As explained below, it would have been obvious to combine Coffman and Kanevsky. A POSITA would have therefore understood and found it obvious for Coffman’s “natural language understanding and natural language parsing system” (i.e., “**parser**”) to be “**configured to determine a context for one or more keywords contained in the utterance.**” (Schmandt, ¶78.)

Rationale and Motivation to Combine (Coffman and Kanevsky). It would have been obvious to combine Coffman and Kanevsky, predictably resulting in Coffman’s conversational system in which a parser used keywords in the user’s utterance—i.e., “key word spotting”—for performing natural language understanding. (Schmandt, ¶79.) Combining the teachings of Coffman and Kanevsky in this manner would have therefore resulted in “**a parser configured to determine a context for one or more keywords contained in the utterance**” as claimed. (*Id.*)

Petition for *Inter Partes* Review of
U.S. Patent No. 7,502,738 B2

Both Coffman and Kanevsky are analogous references in the same field of speech recognition and processing as the '738 patent, and they would have been reasonably pertinent to a number of problems facing the patent inventors, including how to implement a system for a person to interact with a computer system using speech and techniques for performing speech recognition. (Schmandt, ¶80; '738, 1:13-15; Coffman, e.g., 1:7-8, Figs. 1, 4 & 6; Kanevsky, e.g., 1:15-21.)

A POSITA would have been motivated to combine. Coffman expressly incorporates Kanevsky (Coffman, 37:10-13) and is therefore part of Coffman's disclosures for purposes of applying it as prior art. *See, e.g., Paice*, 881 F.3d at 906-907 ("...effectively part of the host document as if it were explicitly contained therein"); *see also Boston Scientific Scimed*, 554 F.3d at 991 ("[c]ombining two embodiments disclosed adjacent to each other in a prior art patent does not require a leap of inventiveness" (quoting *Boston Scientific Scimed, Inc. v. Cordis Corp.*, 554 F.3d 982, 991 (Fed. Cir. 2009))). Moreover, both references share a named co-inventor, Stephane Maes, and identify IBM as the assignee/applicant. (Coffman, Kanevsky face pages.) Therefore, a POSITA would have consulted both of these references to understand the features and functionality of Coffman's conversational system and/or to actually implement or adapt the system and features it provides. (Schmandt, ¶81.)

Moreover, Coffman already suggests the use of keywords in connection with natural language parsing, by explaining that recognized words from a user's utterance are mapped to a symbolic language of an application. (Schmandt, ¶82; Coffman, e.g., 53:25-26 ("User inputs can be represented by their transcriptions or their mappings into a symbolic language after parsing and NLU.")) For example, as noted, Coffman describes an example where the word "open" in the user's command "I would like to open this" is recognized and mapped to an "open" command. (Coffman, 6:32-7:3.) A POSITA would have understood that the word "open" in this context was an example of a "**keyword**," consistent with the well-known concept of "keyword detection." (Schmandt, ¶82 (citing EX1008, p.0029 (defining "keyword detection"))). Kanevsky provides express confirmation, and makes obvious, that parsing the recognized words and phrases of a user's utterance in connection with natural language understanding used "keywords" included in the utterance. (Kanevsky, 14:1-7.) Moreover, Kanevsky's description of "classical parsing capabilities with key word spotting" makes clear that implementing keyword spotting as taught by Kanevsky in Coffman's natural language parsing would have involved, at most, the application of a known technique to implement or improve Coffman's similar functionality in the same way. (Schmandt, ¶82.)

A POSITA would have had a reasonable expectation of success, as the combination would have involved applying well-known technologies and

techniques without undue experimentation. (Schmandt, ¶83 (quoting Kanevsky, 14:1-7).)

1[b] further recites that the parser is also **“configured...to determine a meaning of the utterance based on the determined context.”** A POSITA would have also understood and found it obvious for Coffman’s “natural language understanding and natural language parsing system” (i.e., “parser”) to be **“configured...to determine a meaning of the utterance based on the determined context.”** (Schmandt, ¶84.) This is because, as explained above, Coffman uses NLU as part of the process to determine a “context” for the user’s request in order to properly carry out the request. Using an example from Coffman, using NLU, Coffman’s system determines that the meaning of the user utterance “I would like to open this” is a request to perform the “open” command with a particular application, which is determined by matching the recognized words and phrases to an “open” command in the application’s symbolic language associated with a context. (*Id.* (citing Coffman, 6:32-7:3).)

1[b] further recites **“wherein the parser 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.”** Coffman’s NLU system (“parser”) **“selects at least one of the plurality of”** conversationally aware application

(“**domain agent**”) “**based on the determined meaning**” because it activates an application for processing the user’s query in accordance with the determined symbolic mapping and call/query to perform for the application. (Schmandt, ¶85.)

Coffman explains:

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.

(Coffman, 42:1-7; *see also id.*, 37:2-4 (“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.”).) For example, based on the NLU system’s determination that the user utterance “I would like to open this” means that the user is requesting to perform “open” command with a particular application, that application is “**selected**” to perform the command. (Schmandt, ¶85 (citing Coffman, 6:32-7:3).)

Lastly, the selected conversationally aware application (“**selected domain agent**”) “**receives, processes, and responds to requests associated with the determined context**” because, as explained, it is activated to carry out a call/query

associated with the determined context in accordance with the user's request and it responds by returning output to the conversational system for output to the user. (Schmandt, ¶86.) For example, Coffman describes “return[ing] output of executed or running processes” and presenting them to the user using, for example, text-to-speech, as Coffman illustrates in Figure 6:

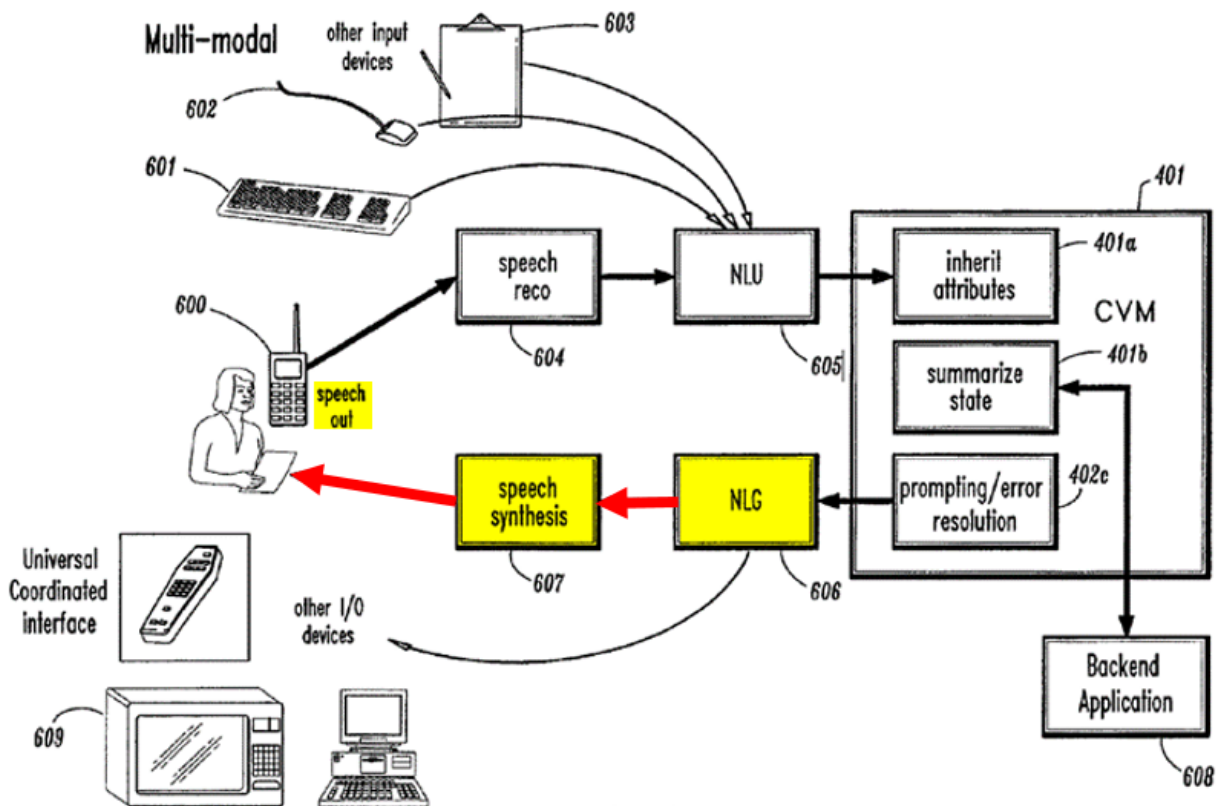


FIG. 6

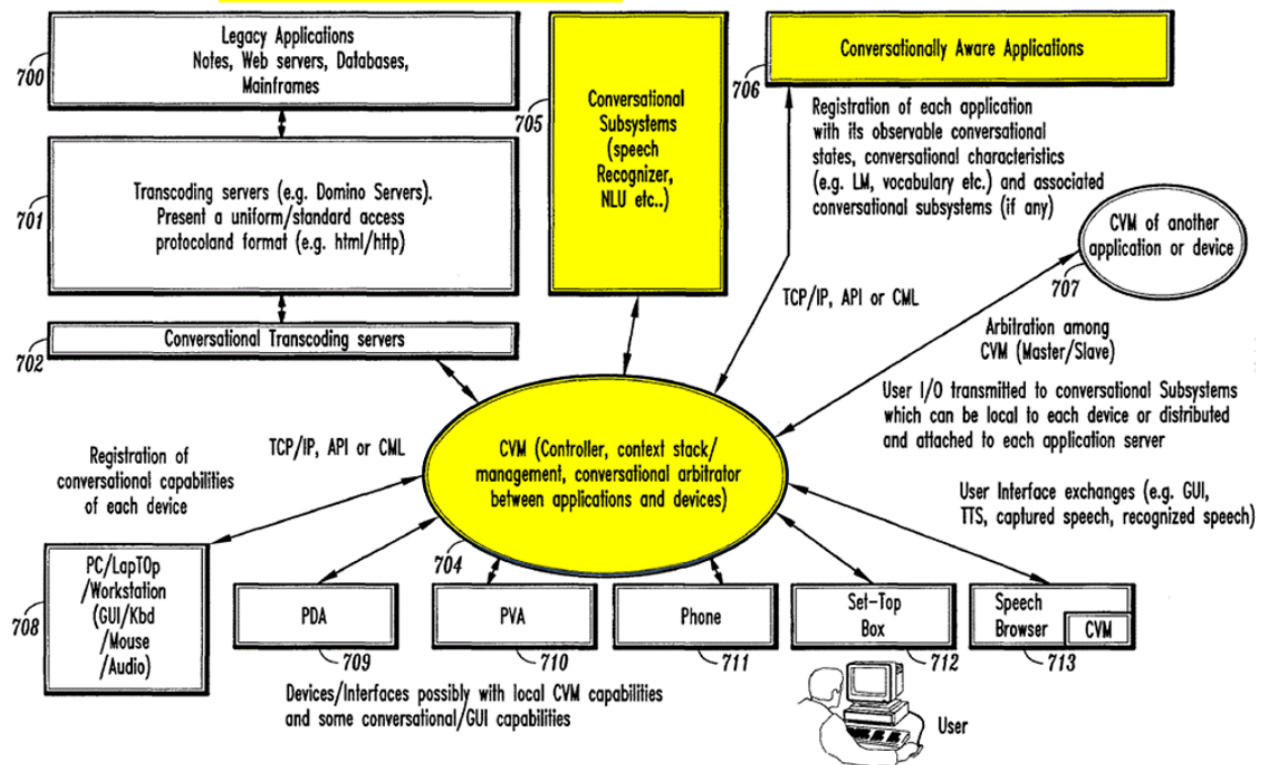
(Coffman, Fig. 6 (annotations added), 40:9-19; *see also id.*, 13:8-13 & 58:9-11 (CVM converts application “output into speech...provided to the user”), 39:3-5, 37:2-4.)

- (c) **“an event manager configured to coordinate interaction between the parser and the agent architecture; and”**
(1[c])

The '738 patent describes an **“event manager”** as a software module for managing events. (Schmandt, ¶87; '738, 2:33-40, 11:51-52.) The '738 patent's use of the term **“event”** is consistent with its ordinary understanding in the art as broadly referring to an action or occurrence to which a program might respond. (Schmandt, ¶87 (citing EX1009, p.0015 (defining “event”)).)

A POSITA would have understood and found it obvious that Coffman's system includes an **“event manager”** as claimed, which corresponds to software functionality that mediates actions between its NLU system (**“parser”**) and conversationally-aware applications (**“the agent architecture”**). (Schmandt, ¶88) For example, Figure 7 of Coffman depicts a CVM architecture where the connection between conversational subsystems 705 (including NLU) and the rest of the architecture, including conversationally aware applications 706, is mediated by CVM 704:

FIG. 7 Architecture for a Distributed CVM



(Coffman, Fig. 7 (highlighting added).) Likewise, Figure 4 of Coffman depicts the connection between conversational subsystems 407 (including NLU 410) and the rest of the architecture being mediated by conversational subsystem services component 406 of CVM 401:

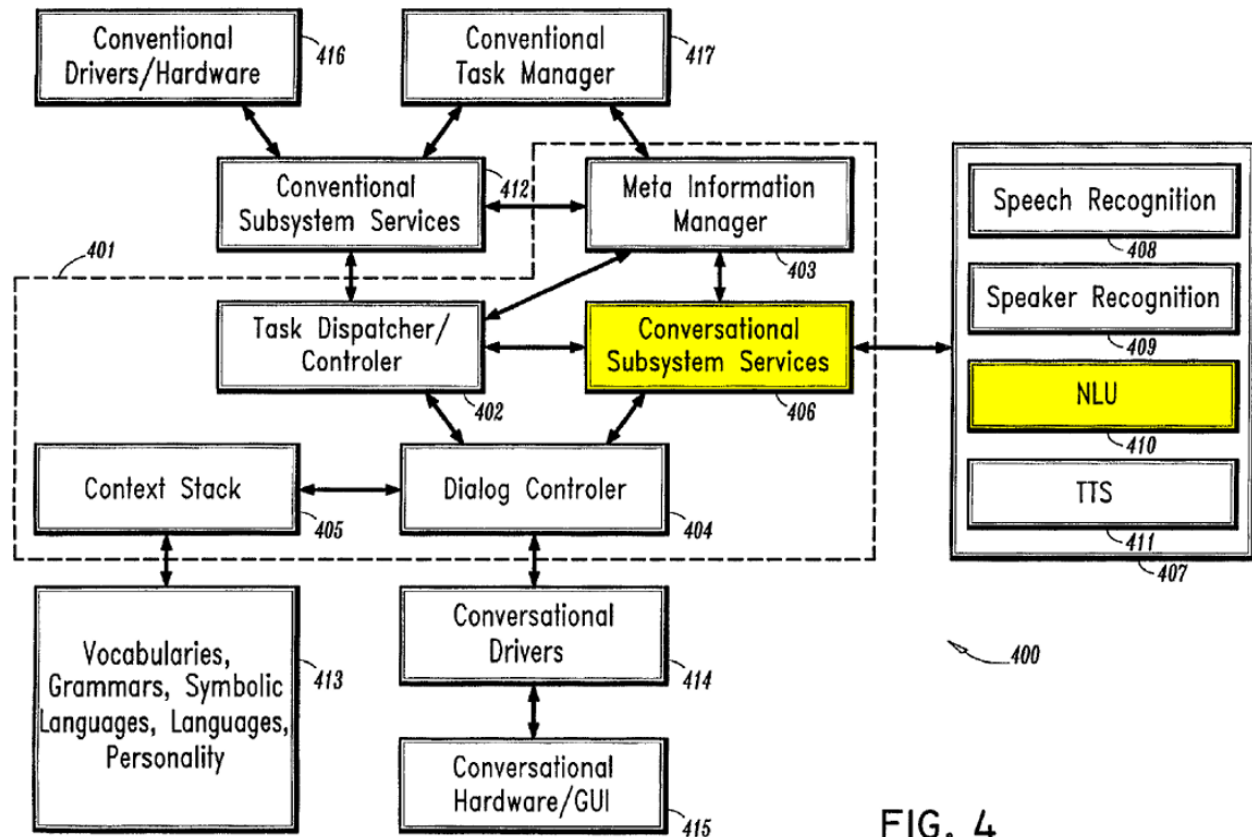


FIG. 4

(Coffman, Fig. 4 (highlighting added).) Coffman explains that the interaction between NLU 410 and conversationally aware applications takes place via an application program interface (“API”) mediated by the CVM. For example, referring to Figure 4, Coffman states that “each of these conversational subsystems 407 may be accessed through API calls to the CVM 401. The CVM 401 will locate the requested conversational subsystem 407 (via the conversational subsystem services manager 406), drive its execution and return appropriately the results.” (Coffman, 29:6-9; *see also id.*, 13:26-28, 12:28-13:13.) A POSITA would have therefore understood and found it obvious that Coffman’s conversational computing system included an **“event manager configured to coordinate interaction**

between the parser and the agent architecture” in the form of CVM software functionality for mediating actions between its NLU system (“**the parser**”) and its conversationally aware applications (“**the agent architecture**”), such as actions to activate an application and execute an application call/query as a result of NLU processing. (Schmandt, ¶88.)

- (d) **“an update manager that enables the user to purchase one or more domain agents from a third party on a one-time or subscription basis.” (1[d])**

The ’738 patent describes an “**update manager**” as a software module that manages agent-related updates. (Schmandt, ¶89; ’738, 2:33-46, 10:22-25.)

Coffman similarly discloses that its conversational computing system includes help and support services providing functionality to install, upgrade and re-install conversationally aware applications:

In yet another aspect, conversational help, manuals and support is provided through a ubiquitous coordinated conversational interface, using local and remote resources, usage history of a user and agents to complete request, guide through procedure, search for information and upgrade/install new applications.

(Coffman, 9:4-7; *see also id.*, 52:7-11 (“local or remote agents (e.g. to upgrade or re-install and application in the background)”); 21:15-18 (“services such as conversational help and assistance, as well as some dialog prompts (introduction, questions, feedback etc) provided by the CVM system...”).) Coffman also describes

Petition for *Inter Partes* Review of
U.S. Patent No. 7,502,738 B2

providing the ability for third parties to develop conversationally aware applications for the system, explaining that “development tools are provided for developer to build, simulate and debug conversational aware application for CVM.” (Coffman, 7:29-30; *see also id.*, 18:16-18 (“The development environment offered by the CVM is referred to herein as SPOKEN AGE™. Spoken Age allows a developer to build, simulate and debug conversational aware application for CVM.”), 13:21-23; Schmandt, ¶90.)

Coffman does not detail the process by which its help/support services enable a user to install (re-install) or upgrade an application (such as a conversationally aware application developed by a third party), but it would have been obvious in view of U.S. Patent No. 9,817,650 B2 (**Ronning (EX1010)**) for Coffman’s system to include software functionality to “**enable[] the user to purchase one or more domain agents [conversationally-aware applications] from a third party on a one-time or subscription basis.**” (Schmandt, ¶90.)

Ronning describes “an agent software application for controlling distribution of files and managing updates to files,” including software applications and data. (Ronning, 2:29-34; *see also id.*, 9:8-12, 3:36-42; Schmandt, ¶91.) Using the agent, a user can interact with an ecommerce system to purchase and download, e.g., software applications. (Ronning, 4:10-25, 3:44-54.) The agent “shields the user from the order process, for example, in order to simplify it” and “may also perform

system management, for example, performing a background process that scans the user's system in order to manage licensed software, perform archival control, and perform other such processing.” (Ronning, 8:59-65.) According to Ronning, “[t]he agent can thus assist a user in managing the downloading of particular digital information, as well as provide recommendations concerning updates to files on the user's machine.” (Ronning, 9:17-20.) Figure 17 of Ronning, shown below, illustrates an example agent interface for a user to download and install various software applications, including Mahjongg Master and MusicMatch Audio Player:

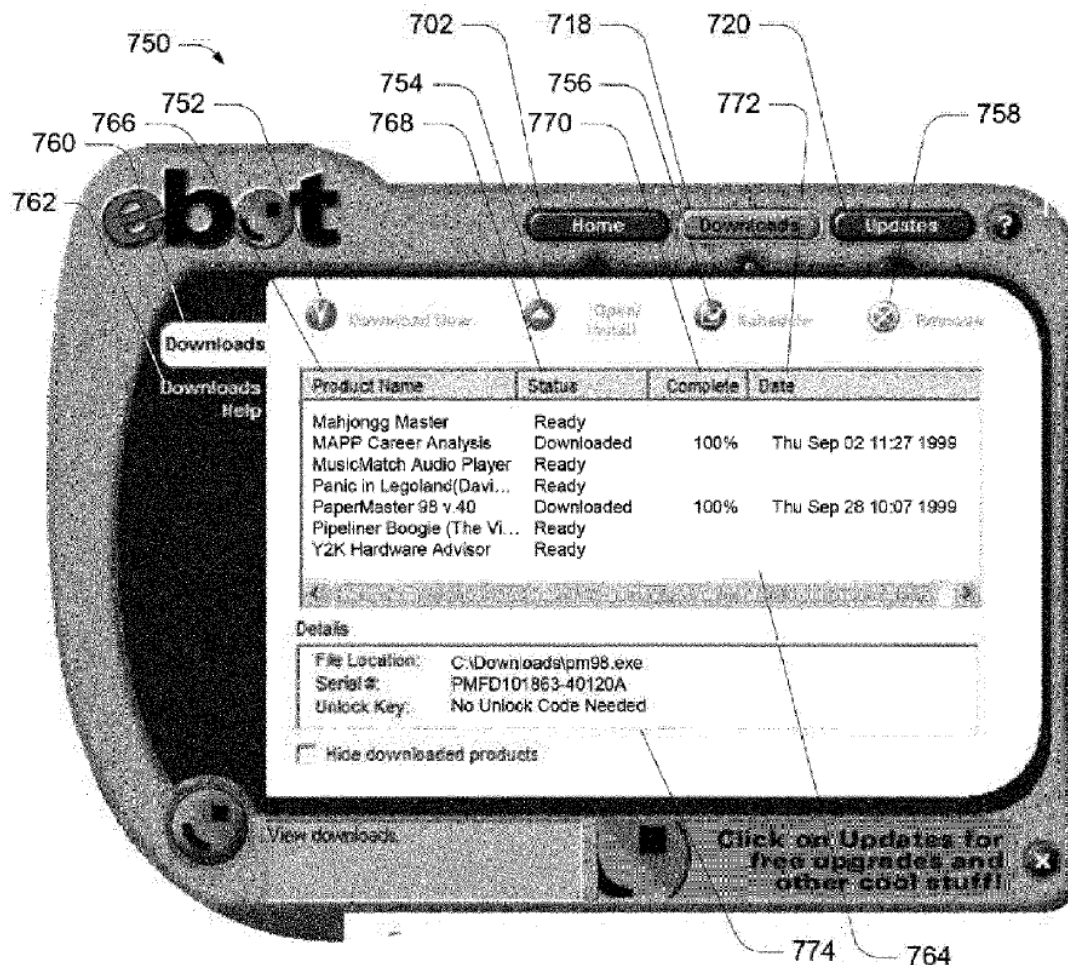


FIG. 17

(Ronning, Fig. 17; *see also id.*, 12:15-17 (“Upon selection of downloads section **718**, the agent displays a downloads screen **750** shown in FIG. 17 (step **568**).”), 12:48-56 (“In the downloads section **750**, selection of an open/install section **754** (step **574**) provides for the user to request installation of a particular file as selected in status section **764**....If the file has been successfully downloaded, the agent installs the file (step **590**).”), 9:24-29 (“FIGS. 5-9 are flow charts of routines for implementing an agent in agent module **203**....FIGS. 10-24 are diagrams of screens supported by the routines in FIGS. 5-9.”).)

Rationale and Motivation to Combine (Coffman and Kanevsky with Ronning). It would have been obvious to combine Ronning with Coffman and Kanevsky, predictably resulting in Coffman’s conversational system having functionality for a user to purchase conversationally-aware applications from a third party on a one-time or subscription basis. (Schmandt, ¶92.) Combining the teachings in this manner would have therefore resulted in “**an update manager that enables the user to purchase one or more domain agents from a third party on a one-time or subscription basis**” as claimed. (*Id.*)

Ronning is an analogous reference to the ’738 patent because it would have been reasonably pertinent to a number of problems facing the patent inventors, including how to incorporate into a computer system functionality to manage the

system's software (such as downloading, installing and updating) and to enable a user to purchase software for the system. (Schmandt, ¶93.)

A POSITA would have been motivated to combine. As discussed, Coffman discloses that its conversational computing system already includes functionality to install, update and re-install conversationally-aware applications (“**domain agents**”). (Coffman, 9:4-7, 52:7-11, 21:15-18.) Coffman mentions, for example, “local or remote agents (e.g. to upgrade or re-install and application in the background).” (Coffman, 52:7-11.) Ronning describes in detail a similar “agent” for managing software applications installed on a computer system. (Ronning, 2:29-34, 3:44-54, 4:10-25, 8:59-65, 9:8-20.) Therefore a POSITA, looking for further detail in implementing Coffman's conversational system would have looked to Ronning for details regarding how to implement functionality for managing conversationally-aware applications. (Schmandt, ¶94.) The combination would have involved the application of these known techniques of Ronning to improve Coffman's similar software functionality in the same way. (*Id.*) Moreover, it would have been obvious to incorporate Ronning's teachings regarding purchasing software, from a third party, into Coffman's conversational system. (Ronning, 4:10-25, 3:44-54.) Coffman already describes tools for developers (i.e., third parties) to build their own conversationally-aware applications. (Coffman, e.g., 58:12-26.) A POSITA would have recognized that allowing third parties to develop their own

applications would benefit Coffman’s system by increasing the number of applications available to users and thereby increase the overall usefulness of the conversational system. (Schmandt, ¶94.) A POSITA would have further appreciated that enabling users to purchase those applications—i.e., financial compensation—would have been an obvious way to incentivize third parties to develop those applications. (*Id.*)

A POSITA would have had a reasonable expectation of success, as the combination would have involved applying well-known technologies and techniques without undue experimentation. (Schmandt, ¶95.)

2. Claim 2: “The system according to claim 1, further comprising a speech unit that receives the utterance from the user.”

The ’738 patent describes a “**speech unit**” as comprising a microphone and speaker, with other optional elements, and connected to a “main unit” processing device. (’738, 10:28-38, 10:50-52, Fig. 1; Schmandt, ¶96.)

A POSITA would have understood and found it obvious that Coffman’s system would comprise “**a speech unit that receives the utterance from the user.**” (Schmandt, ¶97.) Like the ’738 patent, Coffman similarly describes user devices for its conversational computing system having a microphone and speaker—i.e., a “**speech unit**”—for receiving speech input from a user and providing audio output (e.g., via text-to-speech). (Coffman, e.g., 53:8-10 (“Voice input from a microphone

(e.g. the standard voice input).”), 37:14-22 (“Spoken input from a speech client (e.g., telephone 600)”), 35:2-3 (“listen to the output of a task”), 39:12 (“audio only output”), 39:30 (“voice output”), Fig. 6 (telephone 600, “speech synthesis,” “speech out”).) Figure 6 of Coffman, reproduced below, illustrates receiving an utterance from a user via telephone 600:

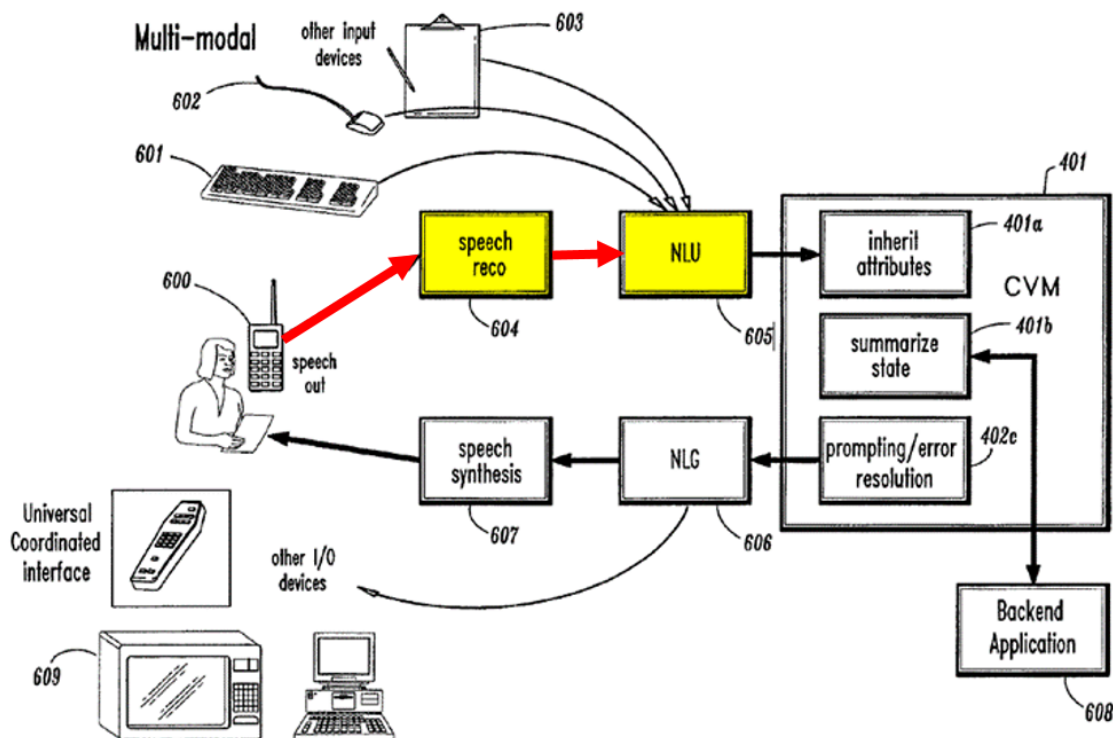


FIG. 6

(Coffman, Fig. 6 (annotations added); *see also id.*, 37:14-22 (“Spoken input from a speech client (e.g., telephone 600)...”).) As explained for the preamble of claim 1, Coffman further discloses that “**the utterance**” recited is “**a user generated natural language speech utterance.**” (Schmandt, ¶97.)

3. Claim 3: “The system according to claim 2, wherein the speech unit includes a speech coder that creates a digitized speech signal from the utterance.”

A POSITA would have understood and found it obvious for a “**speech unit**” in Coffman’s system to “**include[] a speech coder that creates a digitized speech signal from the utterance.**” (Schmandt, ¶¶98-99.) To start, Coffman expressly discloses that its devices include a “**speech coder.**” Referring to Figure 3 (shown below), which depicts a distributed system comprising a client device and other networked devices, Coffman describes using “speech coders” and “acoustic signal processing engines...and suitable audio hardware” to carry out distributed speech processing:

The speech coders 234, 234a provide perceptually acceptable or intelligible reconstruction of the compressed speech and optimized conversational performance (e.g., word error rate). The speech is captured (and transformed into features) on the respective networked devices using acoustic signal processing engines (audio subsystems) 232, 232a and suitable audio hardware 233, 233a.

(Coffman, 26:9-16; *see also id.*, 26:16-27.)

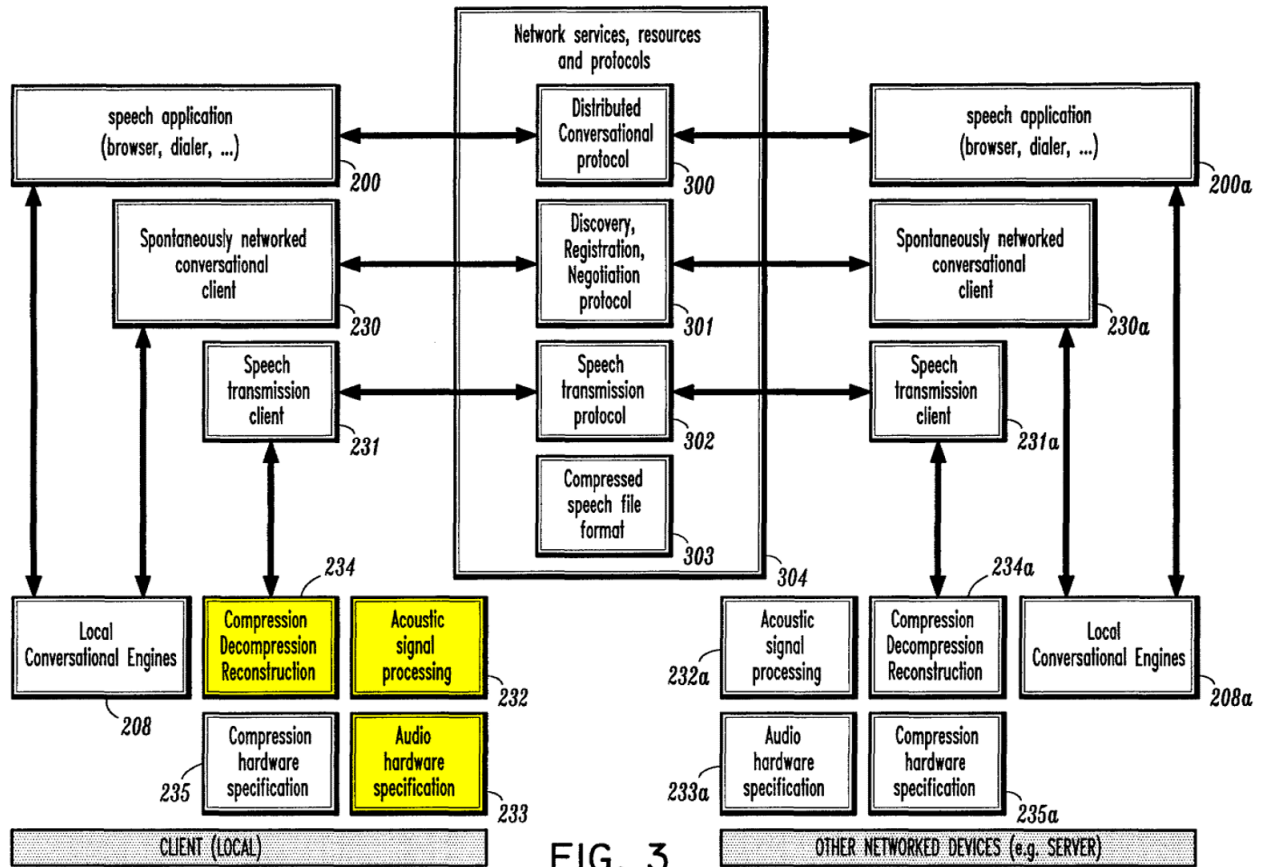


FIG. 3

(Coffman, Fig. 3 (highlighting added), 23:3-5; *see also id.*, Fig. 7 (illustrating distributed architecture).) A POSITA would have understood and found it obvious that a client device’s “speech coder” and/or “acoustic signal processing engines” (and “suitable audio hardware”) in Coffman’s system would correspond to a **“speech coder that creates a digitized speech signal from the utterance”** as claimed; and further, that as part of the functionality for processing incoming speech, it would comprise part of the device’s **“speech unit.”** (Schmandt, ¶199.)

Moreover, in a speech-processing system using conventional digital computing components, such as Coffman’s system, a POSITA would have

understood that, in order to process a user's utterance, the utterance would need to be converted into a digitized speech signal. (Schmandt, ¶98 (citing Coffman, 13:1-3; EX1009, pp.005 ("defining analog-to-digital converter"), 0013 (defining "digital signal"))).) A "**coder**" is simply functionality that can convert an analog audio or video signal to a digital form. (*Id.* (citing EX1009, p.0010 (defining "codec"))).) A POSITA would have therefore understood and found it obvious that a user device in Coffman's system, such as telephone 600, would include a "**speech unit**" with functionality to convert a user's speech into a digital signal—i.e., "**a speech coder**"—as claimed. (*Id.*)

4. Claim 4: "The system according to claim 1, wherein the utterance contains at least one request that includes at least one of a query or a command."

The '738 patent states that a "**question**" and a "**command**" are similar but that a question typically results in a response while a command results in an action. ('738, 5:53-56, 10:66-11:3.) The '738 patent further states that an utterance can include both aspects of a question and a command. ('738, 3:39-45.)

Coffman similarly explains that queries and commands based on a user's utterance "will result in certain results (e.g., outputs or actions)." (Coffman, 41:23-24; Schmandt, ¶100.) And as explained for the claim 1 preamble, Coffman discloses several examples of natural language speech utterances that contain a "**request that includes at least one of a query or a command**," such as "I would like to open

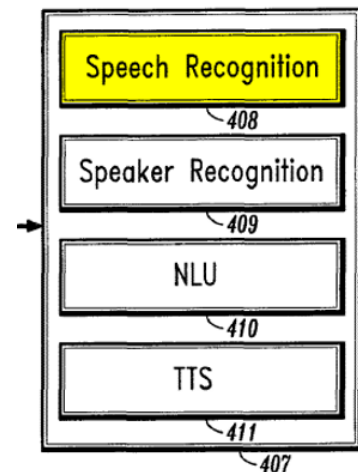
this,” “it should be synchronized with my PDA,” and “e-mail from my mother.” (Coffman, 6:32-7:3, 48:1-3, 56:30-57:1, 56:25-30.) More generally, Coffman describes transcribing user input into query and commands. (Schmandt, ¶100; Coffman, 38:26-27 (“[T]he input procedure provides a set of multi-mode input streams, each transcribed into an ASCII command, query, or list of attribute value n-uples.”), 51:12-14 (“Queries can be decoded, parsed and then translated into a logic combination of queries (symbolic query) using NLU technology.”).)

5. **Claim 5:** “The system according to claim 1, further comprising a speech recognition engine that recognizes words and phrases contained in the received user utterance using information in one or more dictionary and phrase tables, wherein the parser examines the recognized words and phrases to identify the one or more keywords.”

Coffman discloses “a speech recognition engine.”

Coffman explains that its system includes various “conversational subsystems,” including a “speech recognition engine” (illustrated at right (Fig. 4 (excerpt))).

(Coffman, 10:28-30, 29:1-4, Fig. 4.)



Coffman further makes clear that its speech recognition engine “**recognizes words and phrases contained in the received user utterance,**” explaining that “[s]poken input from a speech client (e.g., telephone 600) is subject to a speech recognition process 604 and other input (e.g., keyboard, mouse clicks etc) are subject to NLU processing 605.” (Coffman, 37:20-22; *see also*

id., Fig. 6 (depicting input to and output from “speech reco” 604), 38:9-11 (“Each input stream is converted into its ASCII transcription and aligned with input time-marks via speech recognition processing 604.”); Schmandt, ¶103.) As noted above, Coffman describes various examples of utterances of words and phrases, including “I would like to open this,” “e-mail from my mother,” “grocery list,” and “it should be synchronized with my PDA.” (Coffman, 6:32-7:3, 48:1-3, 56:25-57:2.)

A POSITA would have also understood and found it obvious that the “**recogniz[ing]**” in Coffman’s system would be performed “**using information in one or more dictionary and phrase tables.**” (Schmandt, ¶104.) As explained below, the “**one or more dictionary and phrase tables**” correspond in Coffman to n-gram language models used for speech recognition. (*Id.*)

In particular, Coffman explains that each conversationally-aware application registers with a conversational virtual machine (“CVM”), and provides its language model(s) for controlling the application. (Coffman, e.g., 57:22-58:11, 7:12-14, 13:8-13.) In a speech recognition context, a “language model” generally refers to a formal representation of the structure of a natural language. (Schmandt, ¶105 (citing EX1008, p.0031 (definition of “language model”)).) “In most speech recognisers language models consist of an n-gram model, i.e. probabilities of the occurrence of words, words pairs, sequences of three words, etc.” (*Id.* (quoting EX1008, p.0031; citing EX1008, pp.0035 (defining “n-gram language model”), 0050 (defining

“stochastic language model”)).) A “trigram” language model, for example, therefore refers to an n-gram language model that uses the probabilities of the occurrence of sequences of three words (i.e., the “n” in “n-gram” is three). (*Id.* (citing EX1008, p.0053 (defining “trigram”)).)

The use of language models for speech recognition was well-known. (Schmandt, ¶106.) *See, e.g., Bayer Healthcare LLC v. Baxalta Inc.*, 989 F. 3d 964, 982 (Fed. Cir. 2021) (“[a] patent need not teach, and preferably omits, what is well known in the art”). And as noted, Coffman expressly incorporates Kanevsky (EX1004) (Coffman, 37:10-13), which further explains that a speech recognition engine may be implemented using a “trigram language model.” (Kanevsky, 13:50-56; *see also id.*, 13:48-61 (describing such a speech recognition engine as an “existing component”)).)

Under the proposed combination of Coffman and Kanevsky, therefore, the speech recognition engine’s process to identify one or more trigrams corresponding to the utterance discloses “**recogniz[ing] words and phrases contained in the received user utterance using information in one or more dictionary and phrase tables**” because the trigrams comprise words and sequences of words (i.e., phrases). (Schmandt, ¶107 (citing EX1008, p.0053 (defining “trigram”)).) And moreover, recognizing trigrams from the user’s spoken input (“**words and phrases contained in the received user utterance**”) under the proposed combination “**us[es]**

information in one or more dictionary and phrase tables.” This is because a POSITA would have understood and found it obvious to organize the n-gram language model(s) used by the Coffman’s conversational system as a **“table.”** (*Id.* (citing Sutcliffe (EX1016), p.0016 (with n-gram language model, “syntactic knowledge is captured in a large table of word bigrams (pairs), trigrams (triples),...N-grams”), Denenberg (EX1014), ¶0040 (“...N-gram language model [] works from tables of probabilities of sequences of words”))).) Thus, a trigram language model itself comprises **“one or more dictionary and phrase tables”** because it is organized as a table and includes a list of words and sequences of words to refer to during speech recognition. (*Id.* (citing EX1015, p.005 (defining “dictionary” as encompassing “a list (as of items of data or words) stored in a computer for reference (as for information retrieval or word processing)”))).)

Rationale and Motivation to Combine (Coffman and Kanevsky). The rationale and motivation to combine Coffman and Kanevsky provided in the analysis for 1[b] applies here as well. (Schmandt, ¶108.) Regarding claim 5, it would have further been obvious to a POSITA to combine Coffman and Kanevsky’s teachings regarding trigram language models for speech recognition, predictably resulting in the conversational system of Coffman in which language models used for speech recognition were n-gram language models, such as a trigram language model. (*Id.*) Combining the teachings of Coffman and Kanevsky in this manner would have

therefore resulted in “**a speech recognition engine that recognizes words and phrases contained in the received user utterance using information in one or more dictionary and phrase tables**” as claimed. (*Id.*)

A POSITA would have been motivated to combine. As discussed, Coffman expressly incorporates Kanevsky by reference. A POSITA, seeking further detail for implementing Coffman’s system would have naturally turned to Kanevsky for its teachings regarding speech recognition engines. (Schmandt, ¶109.) Moreover, as explained, Coffman already discloses that a conversationally-aware application provides its “language models” to the conversational system in connection with processing a user’s input. A POSITA would have understood that a language model in this context likely refers to an n-gram language model (*Id.* (citing EX1008, pp.0031 (“language model”), 0035 (“n-gram language model”))), and Kanevsky provides express confirmation, explaining that a speech recognition engine can be implemented using a trigram language model (*Id.* (citing Kanevsky, 13:50-56)). Additionally, n-gram language models were one of a number of finite ways of implementing speech recognition, and the use of an n-gram language model as taught by Kanevsky would have involved the application of a known technique to implement or improve Coffman’s speech recognition functionality in the same way. (*Id.*)

A POSITA would have had a reasonable expectation of success, as the combination would have involved applying well-known technologies and techniques without undue experimentation. (Schmandt, ¶110 (citing ' 738, 1:40-41; Kanevsky, 13:48-61).)

Claim 5 further recites “**wherein the parser examines the recognized words and phrases to identify the one or more keywords,**” which was largely addressed in the analysis of 1[b]. As explained, Coffman discloses “**the parser**” in the form of the conversational system’s “natural language understanding and natural language parsing system” (Coffman, e.g., Fig. 4), which is used to determine a context for a user’s utterance. A POSITA, as discussed, would have understood and found it obvious (alone or in view of Kanevsky) for Coffman’s parser to be configured to determine a context by examining recognized words and phrases in the user’s utterance to identify keyword(s) associated with the context. (Schmandt, ¶111.) For example, Coffman describes recognizing the word “open” (“**keyword**”) in the user’s spoken utterance “I would like to open this” and mapping it to an “open” command in an application’s symbolic language associated with a context. (Coffman, 6:32-7:3; *see also id.*, 42:1-5 (“fit[ting] the utterance at the level of the NLU symbolic language”), 30:20-22 (using NLU “symbolic language (attribute value n-uple) [to map] the actual query”); Schmandt, ¶111.)

6. Claim 6: “The system according to claim 5, wherein the speech recognition engine further recognizes an identity of the user based on information contained in one or more user profiles.”

Claim 6 is disclosed and obvious in two separate but similar ways. First, Coffman describes using conversational biometrics (such as acoustic information for a user’s speech) to collect information about the user in order to recognize them and perform “more robust recognition.” (Coffman, 37:5-10.) Coffman points to Kanevsky for additional information (Coffman, 37:5-13), and Kanevsky further describes creating an “acoustic user model” against which samples from the user’s utterance are compared in order to identify the user. (Kanevsky, 6:66-7:8; *see also id.*, 5:39-46 (describing speaker identification using “acoustic user model”), 4:54-55 (identifying “speech biometric [that] contains acoustic information”).) A POSITA would have recognized that, by identifying the speaker, Coffman’s speech recognition engine could “perform more robust recognition.” (Schmandt, ¶112.) For example, as Coffman recognizes, collecting user-specific information helps inform the speech recognizer in determining the correct words and phrases included in a user’s utterance. (*Id.*) And as Coffman and Kanevsky explain, the speaker identification is performed using information collected about the user, such as the user’s acoustic user model—i.e., a “**user profile.**” (*Id.*)

As a second way that claim 6 is obvious, Coffman describes performing text-independent speaker identification (Coffman, 47:25-27, 8:16-18, 5:1-4), which

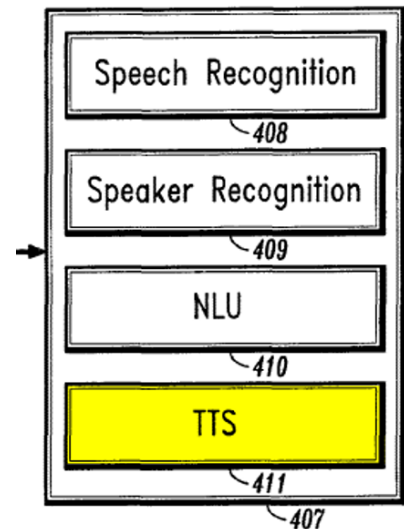
involves using speech signal features to determine the speaker's identity and does not require the utterance to comprise specific linguistic content. (Schmandt, ¶113 (citing EX1008, pp.0047 (defining "speaker recognition"), 0051 (defining "text-independent speaker recognition system")).) Kanevsky provides further detail regarding text-independent speaker identification, explaining that it involves the "speech recognizer (such as ASR **28** in FIG. 2) extract[ing] the acoustic features" associated with a spoken utterance and being fed into a "text-independent speaker recognition module (such as module **52** in FIG. 2) which provides the system with a list of potential users." (Kanevsky, 12:2-10.) As discussed above, the speaker identification uses information collected about the user, such as an acoustic user model—i.e., a "**user profile**." (Kanevsky, e.g., 6:66-7:9 (text-independent speaker module **52** "utilizes the user model previously built").) Therefore, because Kanevsky teaches that a "speech recognizer" (i.e., "**speech recognition engine**") and collected user information ("**user profile**") are used for text-independent speaker identification, the combination of Coffman and Kanevsky discloses and renders obvious the "**speech recognition engine further recognizes an identity of the user based on information contained in one or more user profiles.**" (Schmandt, ¶113.)

The rationale and motivation to combine Coffman and Kanevsky applies here as well. (Schmandt, ¶114.)

7. **Claim 7:** “The system according to claim 1, further comprising a text to speech engine that creates one or more utterances to be presented as output to the user.”

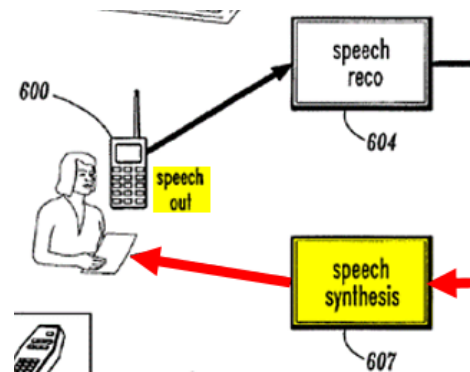
Coffman’s conversational system further includes “a text to speech engine that creates one or more utterances to be presented as output to the user.” Coffman explains that its

conversational subsystem (“**engines**”) includes a “**text-to-speech**” (TTS) engine, depicted as item 411 in Figure 4 (excerpted at right). (Coffman, 10:28-30, 29:1-4, Fig. 4.) Coffman further describes using TTS, i.e., speech



synthesis, to present a response to the user as a result of processing the user’s input.

(Coffman, Fig. 6 (excerpt at right, showing speech synthesis 607 and “speech out” at telephone 600), 13:8-13 & 58:9-11 (CVM converts application “output into speech...provided to the user”), 44:21-22 (“[W]ith one conversational request...the user



can listen to the output of a task”), 40:9-19, 13:8-13, 58:9-11, 39:3-5; Schmandt, ¶115.)

- 8. Claim 8: “The system according to claim 7, further comprising a personality module that includes one or more characteristics for formatting the utterances to be presented as the output to the user.”**

The '738 patent describes a “**personality module**” as a software module that “enables various personalities to be created and used.” ('738, 2:33-46, 10:22-25; Schmandt, ¶116.) The '738 patent states that a response to user’s query may be “formatted to reflect a particular personality or tone to the response in order to make the response more ‘human.’” ('738, 22:32-35.)

Coffman discloses that its system further includes a “**personality module**” as claimed, explaining that the system includes software functionality for a user to create and use different “personalities” corresponding to different types of synthesized voices. (Schmandt, ¶117.) For example, the system’s development tools allow “advanced conversational interfaces to be constructed with multiple personalities, such as Voice fonts, which allows the user to select the type of voice providing the output.” (Coffman, 8:1-3.) Coffman further explains that, “access to each task or resource can be individually customized to preferences of the user requester. For instance, the personality/behavior of the CVM (e.g. synthesized voice - Voice Fonts) can be automatically customized to an identified user’s preferences.” (Coffman, 47:8-13; *see also id.*, 39:30-32.) The different types of voices that Coffman’s system provides in connection with synthesizing voice output to the user

comprise “one or more characteristics for formatting the utterances to be presented as the output to the user.” (Schmandt, ¶117.)

9. **Claim 9: “The system according to claim 1, wherein the parser determines the context for the one or more keywords by applying prior probabilities or fuzzy possibilities to one or more of keyword matching, a user profile, or a dialog history.”**

Claim 9 refers to “**prior probabilities.**” As discussed for claim 5, it would have been obvious based on the proposed combination for Coffman’s system to use n-gram language model(s) for speech recognition, such as a trigram language model. (Schmandt, ¶118.) A POSITA would have understood and found it obvious that the probabilities associated with the language model trigrams are “**prior probabilities**” because they each correspond to the probability of observing some phenomenon estimated from previously collected training data, independent of future observations. (*Id.* (citing EX1008, p.0043 (defining “prior probability”), EX1016, p.006 (explaining that a language model is “extracted or learnt from a corpus, a representative set of texts in the domain of the speech recognizer.”)).)

With that understanding, a POSITA would have understood and found it obvious for Coffman’s natural language parser (“**parser**”) to “**determine[] the context for the one or more keywords by applying prior probabilities or fuzzy possibilities to...keyword matching.**” (Schmandt, ¶119.) Coffman describes determining the “different statistical parsers that must operate on a query for the

dialog controller 404 to identify the active context and complete the query.” (Coffman, 40:12-15.) Coffman’s use of a statistical parser in this manner suggests that its natural language parsing can apply language model probabilities (“**prior probabilities**”) to keyword matching when determining the context for processing the user’s utterance. (Schmandt, ¶119.) For example, a given word, such as “call,” could be a noun or a verb depending on its use—“call Chris” (verb) or “I can’t take your call now” (noun). Language model n-gram probabilities can help determine the correct part of speech, thereby assisting keyword matching for determining the context. If “call <name>” is likely when “call” is used as a verb according to n-gram probabilities, recognizing “Chris” helps the parser understand that “call” is being used as a verb, and therefore helps match a keyword (e.g., “call”) to complete a query associated with a context. (*Id.*) As another example, speech recognizers are error-prone and as a result may provide not only the sequence of words that is most likely given the acoustical evidence and n-gram probabilities, but also the next few most likely word sequences (this is called an “N-best list”). (*Id.* (citing Maes (**EX1007**), 18:13-16 (describing estimating confidence levels for speech recognition using “fraction of the NBEST list with the same word as the top choice”).)⁵ In order to

⁵ As discussed below, Coffman incorporates Maes by reference and claims priority to the same provisional patent applications as Maes. (Coffman, 22:25-29, Coffman 1:2-3 & Maes, 1:4-5.)

choose among them, a natural language parser, assisted by n-gram probabilities, can choose the most likely word sequence as part of determining a meaning of the utterance and thus the appropriate context by matching keyword(s) in that word sequence—for example, determining the “call Chris” is more likely than “call risk,” and using “call Chris” in completing query associated with a context. (*Id.*)

Coffman and Kanevsky also disclose and render obvious claim 9 in another way. As discussed for claim 5, under the proposed combination, the n-gram language model probabilities (“**prior probabilities**”) are used for the initial step of recognizing speech in the user’s utterance. Because the results of speech recognition are, in turn, used for the subsequent step of natural language understanding to determine a context for the user’s utterance by keyword matching, a POSITA would have understood that the prior probabilities also played a role in carrying out this subsequent keyword matching step. (Schmandt, ¶120.)

10. Claim 10: “The system according to claim 1, wherein at least one of the plurality of domain agents creates a query to at least one local or external information source when processing a request that includes a question.”

A POSITA would have understood and found it obvious for a conversationally aware application (“**domain agent**”) in Coffman’s system to “**create[] a query to at least one local or external information source when processing a request that includes a question.**” (Schmandt, ¶121.) For example, Coffman describes a user asking the conversational system for turn-by-turn driving directions to a store.

(Coffman, 57:5-7, 56:24.) Consistent with the '738 patent's recognition that a user's request can include aspects of both a question and a command, a POSITA would have understood that the user's request for driving directions would include, explicitly or implicitly, a question where the store was located. (Schmandt, ¶121; '738, 3:39-45.) Coffman explains that, in order to provide the directions, the system resolves such a question by fetching the location of the store, "locally or from a server." (Coffman, 57:5-7.) A POSITA would have therefore understood and found it obvious that, for example, a conversationally aware navigation application ("**domain agent**") would create a query to a local or external information source when processing a user's request for driving directions. (Schmandt, ¶121.)

As another example, Coffman further explains that "[a]t any time, you can request navigation information about where you are, what to do next, how far to go or even request a different itinerary." (Coffman, 57:7-9.) A POSITA would have understood that these types of requests would include a question—e.g., "what should I do next?" or "how far until I reach the store?"—that would involve an application creating a query to a local or external information source to determine the answer. (Schmandt, ¶122.)

As another example, Coffman explains that conversational system can be used to "check the news or stock quotes." (Coffman, 57:17-18.) A POSITA would have found it obvious for a conversationally aware application to query an information

source as part of responding to a user's question about news or a stock quote—for example, “what is the current price of IBM?” (Schmandt, ¶123.)

Coffman's teachings are confirmed by Maes (EX1007), which Coffman incorporates by reference and which claims priority to the same provisional patent applications as Coffman. (Coffman, 22:25-29; Coffman 1:2-3 & Maes, 1:4-5; Schmandt, ¶124.) Coffman specifically points to Maes for aspects of its conversational computing system. (Coffman, 22:25-29, 26:21-25, 45:11-22, 61:5-6.) Maes provides further detail about the operation of an application that provides news and stock information, making clear that a user's question about news or stocks would involve a query to a local or external information source. (Schmandt, ¶125 (citing Maes, 21:20-22:2).)

11. Claim 11: “The system according to claim 1, wherein at least one of the plurality of domain agents directs a command to the system or an external entity when processing a request that includes a command.”

A POSITA would have understood and found it obvious for a conversationally aware application (“**domain agent**”) in Coffman's system to “**direct[] a command to the system or an external entity when processing a request that includes a command.**” (Schmandt, ¶126.) For example, Coffman explains that “a name dialer application can be added to a conversational smart phone.” (Coffman, 45:2-3.) A POSITA would have also understood that, in response to receiving a command to call a particular name, the name dialer application would direct a command to the

system to initiate a telephone connection. (Schmandt, ¶126.) In addition, Coffman makes clear that a command to call someone using the application can involve the application directing a command to an external entity, explaining that “unknown names or names that were never used before are sent to a more powerful networked machine for recognition and then download the updated information (phone number to dial etc.).” (*Id.* (quoting Coffman, 45:2-6).) Coffman’s teachings are again confirmed by the incorporated Maes application (EX1007), which further details the operation of a name dialer application, including automatically initiating a telephone call in response to a user’s command, as well as directing a command to a remote server for performing name recognition. (*Id.* (quoting Maes 21:6-15).)

12. Claim 18: “The system according to claim 1, wherein the agent architecture further includes a system agent that provides default functionality and basic services, wherein each of the plurality of domain agents can use the functionality and services of the system agent.”

As explained for 1[a], Coffman discloses “**an agent architecture**” that includes a conversational virtual machine (“CVM”) and conversational subsystems, and a plurality of conversationally aware applications. The “**system agent**” in Coffman’s “**agent architecture**” corresponds to conversational virtual machine software including the “CVM kernel.” (Schmandt, ¶128.) The CVM kernel (also referred to as the CVM controller) “keeps track of [conversational] resources, transmit[s] input to the appropriate conversational subsystems and arbitrate[s]

between devices, state and applications. The CVM kernel layer also coordinates the output generation and prioritization according to the active conversation and conversation history, delayed returns, delegation across network resources and task delegation and memorization.” (Coffman, 5:12-24; *see also id.*, 5:12 (“CVM kernel layer (or CVM controller)”)).) This CVM software is a “**system agent**” because, as shown below, it provides basic functionality and services of the conversational system to perform operations on behalf of both a user as well as conversationally aware applications. (Schmandt, ¶128.)

Coffman’s CVM software (“**system agent**”) discloses the claimed “**default functionality**” in at least three different ways.

The first “**default functionality**” that the CVM software provides is handling input from/output to the user, such as by providing text or speech output from a conversationally aware application to the user. (Schmandt, ¶130.) Coffman explains, for example, that I/O resources (such as a touch screen and audio subsystem) will register with the CVM kernel, and that the CVM “automatically handles all input/output issues.” (Coffman, 19:20-25, 13:5-8, 58:8-9.) Coffman makes clear that this normal input/output functionality is “**default functionality**,” explaining that the normal input/output functionality can be changed by, for example, “redirecting” the results provided by an application:

[C]omplete and non-ambiguous commands will result in certain results

(e.g., outputs or actions). These results are similarly returned to the appropriate conversational engine 407 with a priority level and update the context, unless if re-directed by the user as in conventional systems.

(Coffman, 41:23-26; *see also id.*, 43:6-7, 44:13-16; Schmandt, ¶130.) Coffman describes examples of how the user can redirect system input/output, such as redirecting application output to use as input. (Coffman, 35:2-6.)

The second “**default functionality**” that the CVM software provides is mediating interaction between conversationally aware applications and CVM subsystems, such as a speech recognition engine, as discussed for 1[c]. (Schmandt, ¶131 (citing Coffman, e.g., 10:28-30).) By default, the subsystem is “hidden to the application through the conversational application APIs.” (Coffman, 5:1-6.) But Coffman explains that this default functionality can be changed, such that an application can directly interface with the subsystem. (Coffman, 29:1-13 (“...all conversational subsystem calls are hidden to the application (although the engine APIs are always available to the application if the developer wants a [sic] to implement a specific behavior of the engines 407).”); Schmandt, ¶131.)

The third “**default functionality**” that the CVM software provides is acting as a “master” CVM that handles all CVM-related processing for the system. (Schmandt, ¶132.) But Coffman explains that, in a distributed system involving

multiple CVMs, one of the CVMs will become the “master” while the others act as “slaves.” (Coffman, 25:25-26:1; *see also id.*, 58:3-6.)

Coffman further discloses that this CVM software provides “**basic services.**” Coffman explains that “[t]he core CVM kernel layer 202 comprises programming layers such as a conversational application & behavior/service manager layer 215,” which Coffman describes as providing “core functions” of the CVM kernel layer. (Coffman, 19:26-29.) Coffman explains that the conversational application & behavior/service manager layer 215 comprises functions for managing conversationally aware applications, such as “keeping track which applications are registered,” as well as “all the behaviors and features of a conversational UI that the user may expect to find in the applications and interactions” and “the features that an application developer would expect to be able to access via APIs.” (Coffman, 19:26-20:27; *see also id.*, 20:8-14 (listing examples of “conversational services and behavior provided by the CVM kernel”); Schmandt, ¶133.)

As shown above, the conversationally aware applications (“**domain agents**”) “**can use the functionality and services**” of the CVM software (“**the system agent**”). Conversationally aware applications use the CVM software’s input/output functionality, its functionality to mediate communication with conversational subsystems, and in accordance with its functionality to act as either a “master” or a “slave.” (Schmandt, ¶134.) Conversationally aware applications use the services

provided by the CVM software’s “conversational application & behavior/service manager layer 215,” such as for registration and for providing “behaviors and features” that applications can use without having to implement them in the applications themselves. (*Id.*; Coffman, 20:4-7 (“...without having to implement with the development of the application”).)

13. Claim 20: “The system according to claim 18, wherein the agent architecture further includes an agent library that contains one or more utilities for commonly used functions, wherein each of the system agent and the plurality of domain agents can use the utilities contained in the agent library.”

In Coffman’s “**agent architecture**,” the “**agent library**” corresponds to its conversational subsystems. (Schmandt, ¶135.) Coffman’s conversational subsystems “**contain[] one or more utilities for commonly used functions**,” including speech recognition and text-to-speech utilities. (Coffman, 5:1-4.)

Both the “**system agent**” (CVM software) and “**plurality of domain agents**” (conversationally aware applications) “**can use the utilities contained in the agent library**.” Coffman discloses that the CVM software uses the utilities, explaining, for example, that “[t]he CVM 401 will locate the requested conversational subsystem 407 (via the conversational subsystem services manager 406), drive its execution and return appropriately the results.” (Coffman, 29:7-9; Schmandt, ¶136.) Conversationally aware applications, as explained, use the utilities either via CVM-mediated APIs or directly. (Coffman, 29:9-13; Schmandt, ¶136.)

- 14. Claim 21: “The system according to claim 20, wherein the utilities contained in the agent library include at least one of text and string handling, network communications, database lookup and management, fuzzy and probabilistic evaluation, or text to speech formats.”**

The utilities in Coffman’s conversational subsystems (“**agent library**”) include at least “**text and string handling,**” “**fuzzy and probabilistic evaluation**” and “**text to speech formats.**”

With respect to “**text and string handling,**” a POSITA would have understood that Coffman’s conversational subsystems perform text and string handling in numerous ways. (Schmandt, ¶138.) For example, Coffman explains that “using the conversational engines 208...each input is converted into a binary or ASCII input, which can be directly processed by the programming language as built-in objects.”⁶ (Coffman, 18:27-30; *see also id.*, 6:26-28, 52:24-27.) Coffman further explains that “[e]ach input stream is converted into its ASCII transcription and aligned with input time-marks via speech recognition processing 604.” (Coffman, 38:9-11.) Coffman also explains that “Queries can be decoded, parsed and then translated into a logic combination of queries (symbolic query) using NLU technology.” (Coffman, 51:12-14.) A POSITA would have understood that these

⁶ “ASCII input” refers to text strings formatted using the American Standard Code for Information Interchange (“ASCII”). (Schmandt, ¶138 n.7.)

transcription, parsing and translating operations would involve “text and string handling” functions. (Schmandt, ¶138.)

With respect to “**fuzzy and probabilistic evaluation**,” a POSITA would have understood that Coffman’s speech recognition process (performed by the speech recognition engine) would involve a fuzzy and probabilistic evaluation of the spoken input using a language model to identify the most likely words comprising the input. (Schmandt, ¶139.) As explained for claim 1, under the proposed combination with Kanevsky, Coffman’s speech recognition would have involved using trigram language model probabilities to determine the most likely sequence of words in a user’s utterance. The fuzzy/probabilistic evaluation is reflected in the fact that the determination does not reflect certainty but rather a probability of being correct. (*Id.*)

With respect to “**text to speech formats**,” Coffman, as noted, identifies text-to-speech synthesis as one of its conversational subsystems. (Coffman, 29:1-4.) Coffman makes clear that text-to-speech synthesis involves using “voice fonts” that allow the speech to be formatted in a particular type of voice. (Coffman, 47:8-13, 8:1-3, 39:30-32; Schmandt, ¶140.)

15. **Claim 22: “The system according to claim 20, wherein the update manager uses a network interface to download, distribute, and install updates relating to one or more of the system agent, the plurality of domain agents, the agent library, one or more databases available to the agent architecture, or entries in the dictionary and phrase tables.”**

Claim 22 was largely addressed in the analysis of 1[d] and Ronning. As explained, it would have been obvious to combine Ronning’s teachings regarding a software agent to manage distribution, download and installation of application updates with Coffman, particularly since Coffman discloses that its conversational computing system includes functionality to install, update and re-install conversationally aware applications. (Coffman, 9:4-7, 52:7-11; Schmandt, ¶141.) The combination would have resulted in an update manager that used a “**network interface to download, distribute, and install updates relating to**” conversationally aware applications (“**the plurality of domain agents**”). (Schmandt, ¶141.) Ronning’s agent “manages distribution and download of files including digital products to a user’s machine, as well as locating and downloading updates to files contained on the user’s machine.” (Ronning, 9:8-11.) Ronning further explains that, for downloading, “[t]he transmission may occur through any wireline or wireless network,” making clear that the agent would use a “**network interface**” in connection with its operations. (Ronning, 11:63-64; *see generally id.*, 13:31-64; *see also* Schmandt, ¶141 (citing EX1009, p.0026 (defining “network

interface card”)).) Ronning also discloses that “[i]f the file has been successfully downloaded, the agent installs the file (step 590).” (Ronning, 12:55-56.)

- 16. Claim 23: “The system according to claim 22, wherein the updates include one or more of a new domain agent, additional domain knowledge for a domain agent, new keywords for a domain agent, preferred information sources for a domain agent, updated domain information for a domain agent, or updated content for a domain agent.”**

Claim 23 does not add anything that was not addressed in the discussions of 1[d] and claim 22. It would have been obvious for the “**updates**” to include a new conversationally aware application (“**new domain agent**”). (Schmandt, ¶142.)

17. Claim 26

Claim 26 is identical to claim 5, except that it depends from dependent claim 18 instead of directly from independent claim 1. For the reasons discussed for claims 5 and 18, claim 26 is obvious. (Schmandt, ¶143.)

18. Claim 31

Claim 31 is a duplicate of claim 5 and therefore obvious. (Schmandt, ¶144.)

- 19. Claim 32: “The system according to claim 1, further comprising an extension and modification facility that allows users and content developers to configure behavior of the agent architecture.”**

Coffman explains that its conversational system includes “**an extension and modification facility**” in the form of a development environment (called “Spoken Age”) that “allows a developer to build, simulate and debug conversational aware

application [sic] for CVM.” (Coffman, 18:16-18; *see also id.*, 58:12-20 (“Using developmental tools, the developer can easily build his application around the conversational API and CVM....This means that toolkits and the algorithms that it provides allows the user to re-build, adapt or extend the data files for a given task.”); Schmandt, ¶145.)

Coffman’s development environment “**allows users and content developers to configure behavior of the agent architecture**” by providing the ability for a user or developer to build new conversationally aware applications as well as to modify data files that are used for interacting with and controlling an application—such as the application’s vocabulary, language model(s), baseforms (pronunciation rules), and voice fonts. (Schmandt, ¶146; Coffman, 58:16-20 (“It also provides the environment for modifying the data files (conversational arguments) of the engines for a given application....This means that toolkits and the algorithms that it provides allows the user to re-build, adapt or extend the data files for a given task.”), 29:20-27, 60:31-61:2.) A POSITA would have understood that, building and incorporating a new conversationally aware application into Coffman’s conversational system would “**configure the behavior of the agent architecture**” by, for example, introducing a new domain of functionality with which the user could interface via speech. (Schmandt, ¶146.) A POSITA would also have understood that, for example, modifying an application’s vocabulary or language model could adjust the

words and commands that the application recognized and responded to. (*Id.*)

Similarly, different application voice fonts configure the behavior of the application by providing voice output in different types of voice. (*Id.*; Coffman, 8:1-3.)

Configuring the behavior of a conversationally aware application (“**domain agent**”) would thus serve to configure the behavior of the “**agent architecture.**” (Schmandt, ¶147.)

20. Claim 33: “The system according to claim 32, wherein the extension and modification facility can be used to extend or modify behavior of the plurality of domain agents.”

As explained for claim 32, Coffman’s development environment “**can be used to extend or modify the behavior of the plurality of domain agents**” by providing the ability for a user or developer to modify data files that are used for interacting with and controlling applications as well as to build new conversationally aware applications based on existing applications. (Schmandt, ¶148; Coffman, 58:16-20, 29:20-27, 60:31-61:2.)

21. Claim 34: “The system according to claim 33, wherein the users and content developers can add one or more of new commands, keywords, information, or information sources to the agent architecture to extend or modify the behavior of the plurality of domain agents.”

Claim 34 was largely addressed in the analysis of claim 32. As explained, “**user and content developers**” of Coffman’s development environment can add at least “**new commands, keywords, [and] information to the agent architecture to**

extend or modify the behavior of the plurality of domain agents” by providing the ability for a user or developer to build new conversationally aware applications as well as to modify data files that are used for interacting with and controlling applications. (Schmandt, ¶149; Coffman, 58:16-20, 29:20-27, 60:31-61:2.) This is because developing new applications or modifying data files would involve extending or modifying the commands, keywords and information that the system’s applications used and were responsive to. (Schmandt, ¶149.) Moreover, a POSITA would have understood and found it obvious that developing a new application or modifying an application’s data files could add a new **“information source”**—for example, developing a conversationally aware email application would enable the conversational system to interact with an associated email database (an **“information source”**). (*Id.*)

22. Claim 35: “The system according to claim 32, wherein the extension and modification facility provides a generic agent that can be used to create a new domain agent.”

As discussed for claim 32, Coffman’s development environment of its conversational system can be used to build a conversationally aware application. (Coffman, 18:16-18, 58:12-13.) Coffman further discloses two separate ways in which the development environment **“provides a generic agent that can be used to create a new domain agent.”**

First, Coffman explains that its development environment enables a conversationally aware application to act as a **“generic agent”** that can be **“used to create a new domain agent”** by porting the application to a new platform. (Schmandt, ¶151.) For example, a desktop-based spreadsheet application can be used to create a phone-based version of the application, thereby extending the application to a new domain (the phone). (*Id.* (citing Coffman, 58:27-32).)

Second, Coffman explains that the development environment includes “conversation foundation classes” that enable a developer to convert a conventional application into one that is “conversationally aware.” (Coffman, 13:19-32.) The “conversation foundation classes” thus serve as a **“generic agent”** because they provide the base functionality for creating a new conversationally aware application (**“domain agent”**). (Schmandt, ¶152.)

23. Claim 36: “The system according to claim 32, wherein the users and content developers can configure the behavior of the agent architecture using one or more of a scripting language, a graphical user interface, or natural language utterances.”

As explained for claim 32, Coffman teaches that a user or content developer can configure the behavior of the system’s agent architecture by developing new conversationally aware applications as well as by modifying data files for the applications. Coffman further teaches that users of its development environment can develop new applications and modify data files (**“configur[ing] the behavior of the**

agent architecture” “using...a scripting language.” (Schmandt, ¶153.) With respect to developing new applications, Coffman explains that “[t]he code implementing the applications, API calls and protocol calls includes interpreted and compiled scripts and programs, with library links, conversational logic (as described below) engine call and conversational foundation classes.” (Coffman, 13:26-28; *see also id.*, 13:21-23.) For modifying data files, Coffman explains that “toolkits and the algorithms that it provides allows the user to re-build, adapt or extend the data files for a given task. This involves collecting data for the application following data collection rules and running the appropriate scripts to generate the data file and test the performances.” (Coffman, 58:19-22.)

24. Claim 37: “The system according to claim 32, wherein the users and content developers can create one or more of special reports, packages of queries, alerts, or output formats using the extension and modification facility.”

Coffman’s conversational system allows “**users and content developers**” to create “**packages of queries**” and “**output formats**” using the development environment (“**the extension and modification facility**”). (Schmandt, ¶154.) As discussed for claim 32, with Coffman’s development environment, users and developers can modify data files that are used for interacting with and controlling an application. (Coffman, 58:16-20.) Those data files include “possible queries for a speech input” (“**packages of queries**”) as well as “voice fonts” that are used to format output in a particular type of voice (“**output formats**”). (Coffman, 29:20-27

(listing application data files including “possible queries for a speech input”), 60:31-61:2 (listing application data files, including “prompts/voice fonts for the dialog”), 39:30-32, 8:1-3 (describing voice fonts “allow[ing] the user to select the type of voice providing the output”).)

25. Claim 39

Claim 39 is identical to claim 5, except that it depends from dependent claim 32 instead of directly from independent claim 1. For the reasons discussed above for claims 5 and 32, claim 39 is obvious. (Schmandt, ¶155.)

C. Ground 2: Claims 16, 17, 25, 30 and 38 Are Obvious Over Ground 1 Prior Art in Further View of Zadrozny

1. Claim 16: “The system according to claim 1, wherein the parser formulates at least one request in accordance with a grammar that the selected domain agent uses to process requests associated with the context for the one or more keywords contained in the utterance.”

As explained, Coffman describes using natural language understanding (“**parser**”) to translate a recognized utterance into a symbolic mapping and query/call to be carried out by an application. (Coffman, e.g., 13:8-13, 30:8-10.) For example, Coffman teaches that a user’s utterance “I would like to open this” would be processed into “a set of attributes value pairs: Command: Open, Object: Window” and result in an “open” command being performed. (Coffman, 6:32-7:3.) And Coffman, as noted, explains that an application (when registering with a conversational system), provides its active vocabulary, grammar, and language

models to the CVM for controlling the application. (Coffman, e.g., 24:31-25:9, 57:22-27, 36:31-32 (“each application will provide the CVM 401 its own parsing and translation arguments”).)

A POSITA would have understood and found it obvious that, in a system such as Coffman’s involving speech recognition using language model(s) and natural language understanding, a grammar associated with an application would be used to formulate the request as a symbolic mapping (such as a set of attribute-value pairs such as “Command: Open, Object: Window”) representing the system’s understanding and to, in turn, formulate an application query/call in accordance with that understanding. (Schmandt, ¶¶157-158 (citing Coffman, 53:25-26, 29:1-4, 6:32-7:3; EX1008, pp.0035 (defining “Natural Language Processing”), 0040 (defining “parsing”)).) Both the symbolic mapping and the application query/call resulting from parsing would separately correspond to the **“parser formulat[ing] at least one request in accordance with a grammar that the selected domain agent uses to process requests associated with the context for the one or more keywords contained in the utterance.”** (*Id.*)

Zadrozny (EX1006) confirms that it would have been obvious for Coffman’s natural language parser to formulate a request **“in accordance with a grammar that”** the selected conversationally-aware application (**“the selected domain agent”**) **“uses to process requests associated with the context for the one or more**

keywords contained in the utterance.” (Schmandt, ¶159.) **Zadrozny** is referenced in Kanevsky as an example of a semantic analyzer. (Kanevsky, 14:22-25.) Zadrozny describes a “grammar of constructions” associated with an application (such as a calendar application) for performing natural language understanding to formulate a recognized utterance as an intended action and parameters and then translate the action and parameters into “a form that can be executed by the application.” (Zadrozny, pp.0019-0020.) Zadrozny illustrates the technique with the example utterance “I want you to arrange a conference in my office at 5.” (Zadrozny, p.0021.) Using a grammar of constructions, the utterance is “analyzed as *sent(assert, svoc)*, an assertive sentence consisting of a subject, verb, an object and a complement” and formulated as an action (“schedule”) and series of parameters:

```
[ [ action_name schedule]
  [ event_name
    [ a conference]
  [ event_time
    [ [ minute 0]
      [ hour
        [ 5 am_or_pm]
      ]
    ]
  [ event_place
    [ my office]
```

(Zadrozny, pp.0021-0022.) As Zadrozny explains, this action and parameters are then translated into a form for the application to execute (i.e., an application

Petition for *Inter Partes* Review of
U.S. Patent No. 7,502,738 B2
call/query).

Zadrozny's process is just like Coffman's descriptions of formulating a user's utterance as a set of attribute-value pairs (e.g., "Command: Open, Object: Window") and then converting them into an application call/query. (Schmandt, ¶160 (citing Coffman, 30:8-10, 13:8-13.)

Rationale and Motivation to Combine (Coffman, Kanevsky and Ronning with Zadrozny). It would have been obvious to combine Zadrozny's teachings regarding processing speech using a grammar of constructions with the conversational computing system taught by the combination of Coffman, Kanevsky and Ronning. (Schmandt, ¶161.) The combination would have predictably resulted in the conversational system in which user requests were formulated by Coffman's natural language parser in accordance with a grammar used by the conversationally-aware application that was associated with the context determined using keyword(s) matched from the user's utterance. (*Id.*) The combination would have therefore disclosed **"the parser formulates at least one request in accordance with a grammar that the selected domain agent uses to process requests associated with the context for the one or more keywords contained in the utterance"** as claimed. (*Id.*)

Zadrozny is an analogous reference in the same field as the '738 patent and would have been reasonably pertinent to a number of problems facing the patent

inventors, including techniques for understanding natural language speech input and how to determine and formulate actions for an application to undertake in response to the input. (Schmandt, ¶162; '738, 1:13-15; Zadrozny, e.g., p.0019 (“We present an approach to natural language understanding based on a *computable grammar of constructions*.”) (emphasis in original).)

A POSITA would have been motivated to combine. Kanevsky expressly cites Zadrozny as an example of a “semantic analyzer.” (Kanevsky, 14:22-25.) And authors of Zadrozny are identified as being associated with IBM, the assignee/applicant for both Coffman and Kanevsky. (Zadrozny, p.0019 (“IBM Research Division”).) A POSITA therefore would have turned to Zadrozny for detail about how to implement the systems and techniques described in Coffman and Kanevsky. (Schmandt, ¶163.)

A POSITA would have appreciated that the disclosures of Coffman and Zadrozny confirm their fit. (Schmandt, ¶164.) As noted, Coffman does not describe the natural language understanding process in detail, mostly describing at a high level: applications provide grammars, vocabularies, language models and symbolic mappings, a context is matched by “fit[ting] the utterance at the level of the NLU symbolic language” and “CVM 14 [] transform[s] queries into application calls.” Because Kanevsky expressly cites Zadrozny as an example of a semantic analyzer, a POSITA would have turned to it for its teachings regarding the use of grammars

in natural language understanding to translate a recognized utterance into an application call. (*Id.*) Moreover, as noted, Zadrozny’s process is just like Coffman’s descriptions of formulating a user’s utterance as a set of attribute-value pairs and then converting them into an application call/query. (*Id.*)

A POSITA would have had a reasonable expectation of success. (Schmandt, ¶165 (describing Zadrozny’s working implementation).)

2. **Claim 17: “The system according to claim 16, wherein the parser provides the formulated request to the event manager to invoke the selected domain agent, and the event manager sends and receives one or more events to the agent architecture to process the formulated request, the event manager thereby coordinating interaction between the parser and the agent architecture.”**

As discussed above for 1[c], a POSITA would have understood and found it obvious that Coffman’s system includes an “**event manager**” in the form of software functionality that mediates actions between its NLU system (“**parser**”) and conversationally-aware applications (“**the agent architecture**”). (Schmandt, ¶166.) For example, Coffman describes how interaction between NLU 410 and conversationally aware applications takes place via an application program interface (“API”) mediated by the CVM. (Coffman, 29:6-9, 13:26-28, 12:28-13:13.) As explained in the analysis of claim 16, a POSITA would have understood and found it obvious for Coffman’s NLU system (“**parser**”) to formulate a request based on the user’s utterance, the “**formulated request**” corresponding to either a symbolic

mapping representing the system's understanding or as an application query/call formulated in accordance with that mapping. (Schmandt, ¶166.)

With respect to claim 17, a POSITA would have further understood and found it obvious that Coffman's NLU system ("**parser**") would provide either the symbolic mapping or application query/call ("**formulated request**") to CVM functionality mediating interaction between applications and the NLU system (i.e., "**event manager**") in order to invoke the selected application. (Schmandt, ¶167.) Coffman explains that its conversational subsystems (such as its NLU system) "are responsible for converting voice requests into queries," but that it is the CVM "which decides which application is active" and that the conversational API "conveys all the information for the CVM 14 to transform queries in application calls." (Coffman, 13:8-13; *see also id.*, 36:32-37:3.) Moreover, as discussed for 1[c], Coffman depicts the CVM as mediating interactions between the conversational subsystems (including NLU) and its agent architecture. (Coffman, e.g., Figs. 4, 5 & 7.) A POSITA would have appreciated that implementing that functionality using routine and conventional programming techniques would involve the CVM invoking the selected application and sending and receiving events to process the request. (Schmandt, ¶167.)

A POSITA would have therefore understood and found it obvious that CVM "**event manager**" functionality, as part of "**coordinating interaction between the**

parser and the agent architecture,” would activate the selected conversationally aware application (“**invoke the selected domain agent**”), send an “**event**” to the application comprising an application call/query, and receive an “**event**” from the application comprising the application’s response after processing the call/query (i.e., “**the event manager sends and receives one or more events to the agent architecture to process the formulated request**”). (Schmandt, ¶168; Coffman, 13:8-13, 41:23-24.”.)

3. Claim 25

Claim 25 is identical to claim 16, except that it depends from dependent claim 18 instead of directly from independent claim 1. For the reasons discussed for claims 16 and 18, claim 25 is obvious. (Schmandt, ¶168.)

4. Claim 30

Claim 30 is a duplicate of claim 16 and therefore obvious. (Schmandt, ¶170.)

5. Claim 38

Claim 38 is identical to claim 16, except that it depends from dependent claim 32 instead of directly from independent claim 1. For the reasons discussed for claims 16 and 32, claim 38 is obvious. (Schmandt, ¶171.)

VII. CONCLUSION

Petitioner respectfully requests IPR institution.

Petition for *Inter Partes* Review of
U.S. Patent No. 7,502,738 B2

Dated: April 5, 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 COMPLIANCE WITH WORD COUNT

Pursuant to 37 C.F.R. § 42.24(d), I certify that this petition complies with the type-volume limits of 37 C.F.R. § 42.24(a)(1)(i) because it contains 13,856 words, according to the word-processing system used to prepare this petition, excluding the parts of this petition that are exempted by 37 C.F.R. § 42.24(a) (including the table of contents, a table of authorities, mandatory notices, a certificate of service or this certificate word count, appendix of exhibits, and claim listings).

DATED: April 5, 2024

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

/ Heidi L. Keefe /
Heidi L. Keefe
Reg. No. 40,673

CERTIFICATE OF SERVICE

I hereby certify, pursuant to 37 C.F.R. Sections 42.6 and 42.105, that a complete copy of the attached **PETITION FOR INTER PARTES REVIEW OF U.S. PATENT NO. 7,502,738 B2**, including all exhibits (**Nos. 1001-1004, 1006-1012, 1014-1016, 1020-1023**) and related documents, are being served via Federal Express on the 5th day of April, 2024, the same day as the filing of the above-identified document in the United States Patent and Trademark Office/Patent Trial and Appeal Board, upon Patent Owner by serving the correspondence address of record with the USPTO as follows:

174899 - The Law Offices of David A. Gerasimow, P.C.
211 W. Wacker Dr.
Ste. 1717
Chicago, IL 60606

And, via Federal Express upon counsel of record for Patent Owner in the litigation pending before the U.S. District Court for the District of Delaware entitled *Dialect, LLC v. Google LLC*, Case No. 1:23-cv-00378 (D. Del.) as follows:

Garland Stephens
BLUE PEAK LAW GROUP LLP
3139 W Holcombe Blvd, PMB 8160
Houston, TX 77025
Telephone: 281-972-3036
garland@bluepeak.law

DATED: April 5, 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