

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICROSOFT CORPORATION,
Petitioner,

v.

DIALECT, LLC,
Patent Owner.

IPR2025-01352
Patent: 8,620,659
Issued: December 31, 2013
Application No. 13/022,370
Filed: February 7, 2011

Title: SYSTEM AND METHOD OF SUPPORTING ADAPTIVE
MISRECOGNITION IN CONVERSATIONAL SPEECH

DECLARATION OF PAUL JACOBS

TABLE OF CONTENTS

	Page
I. INTRODUCTION AND ENGAGEMENT	5
II. BACKGROUND AND QUALIFICATIONS	5
III. MATERIALS CONSIDERED AND INFORMATION RELIED UPON REGARDING '659 PATENT.....	10
IV. UNDERSTANDING OF PATENT LAW	12
V. GROUNDS	16
VI. THE '659 PATENT	17
A. Prosecution History Of The '659 Patent	20
VII. LEVEL OF SKILL IN THE ART	20
A. Person Of Ordinary Skill In The Art.....	20
VIII. CLAIM CONSTRUCTION	21
A. "Domain Agent"	21
IX. THE ANCESTOR APPLICATIONS DO NOT SUPPORT CLAIM 42'S STEP OF DETERMINING WHETHER A PERSONALIZED COGNITIVE MODEL ASSOCIATED WITH THE USER INCLUDES SUFFICIENT INFORMATION FOR PREDICTING ONE OR MORE ACTIONS ASSOCIATED WITH THE USER.....	22
1. Claim 42 Lacks Written Description Support.....	22
2. Claim 42 Is Not Enabled.....	25
X. GROUND 1: CLAIM 42 IS OBVIOUS OVER THE 2007 GRANDPARENT PUB. IN VIEW OF LEE.....	27
A. 2007 Grandparent Pub. (EX101005).....	27
B. Lee (EX1008).....	27
C. Combining The 2007 Grandparent Pub. And Lee Teachings.....	30

D.	Claim 42 Mapping.....	33
1.	Element [42.1]	34
2.	Element [42.2].....	34
3.	Element [42.3].....	35
4.	Element [42.4].....	35
5.	Element [42.5].....	36
6.	Element [42.6].....	36
7.	Element [42.7].....	39
XI.	GROUND 2: CLAIM 42 IS OBVIOUS OVER RAMASWAMY	40
A.	Ramaswamy (EX1009)	40
1.	Speech Recognition And Interpretation.....	42
2.	Command Generation and Execution	43
3.	Command Prediction Models	43
4.	Incorporated Ramaswamy NPL Provides Additional Details ..	47
B.	Claim 42 Mapping.....	53
1.	Element [42.1].....	54
2.	Element [42.2].....	56
3.	Element [42.3].....	57
4.	Element [42.4].....	60
5.	Element [42.5].....	62
6.	Element [42.6].....	68
7.	Element [42.7].....	77

XII. GROUND 3: CLAIM 42 IS OBVIOUS OVER RAMASWAMY IN VIEW OF RAMASWAMY NPL (EX1014).....	83
XIII. GROUND 4: CLAIM 42 IS OBVIOUS OVER RAMASWAMY IN VIEW OF HARTONO (EX1010)	87

I, Paul Jacobs, do hereby declare as follows:

I. INTRODUCTION AND ENGAGEMENT

1. I have been retained as an independent expert on behalf of Microsoft Corporation in connection with the above-captioned Petition for *Inter Partes* Review (“IPR”) to provide my analyses and opinions on certain technical issues related to U.S. Patent No. 8,620,659 (hereinafter “the ’659 patent”).

2. I am being compensated at my usual and customary rate for the time I spent in connection with this IPR. My compensation is not affected by the outcome of this IPR.

3. Specifically, I have been asked to provide my opinions regarding whether claim 42 (the “Challenged Claim”) of the ’659 patent would have been obvious to a person having ordinary skill in the art (“POSITA”) as of August 10, 2005. It is my opinion that the Challenged Claim would have been obvious to one of ordinary skill after reviewing the prior art discussed herein.

II. BACKGROUND AND QUALIFICATIONS

4. My name is Paul Jacobs, and I am over 21 years of age and otherwise competent to make this Declaration. I make this Declaration based on facts and matters within my own knowledge and on information provided to me by others, and, if called as a witness, I could and would competently testify to the matters set forth herein.

5. I have summarized in this section my educational background, career history, and other qualifications relevant to this matter. A more complete recitation of my professional experience including a list of my journal publications, patents, conference proceedings, book authorship, and committee memberships may be found in my curriculum vitae, which is attached hereto as Appendix A.

6. I am an expert in natural language processing and have a broad background in computer science that I have applied for decades as a scientist in the field of natural language processing, as a technology leader and executive, as a consultant, a professor, and as an advisor regarding software patents.

7. I received a Bachelor of Science in Applied Mathematics from Harvard University in 1981, a Master of Science in Applied Mathematics from Harvard in 1981, and a Ph.D. in Computer Science from the University of California at Berkeley in 1985.

8. I have authored or co-authored over 50 scientific and technical publications, primarily in the fields of artificial intelligence, natural language processing, and information retrieval. I am listed as an inventor on two U.S. patents directed to computational lexicons, and I have over 40 years of experience in the computer and information industry.

9. I have served in numerous professional and scientific capacities, including one year as a visiting professor of computer science at the University of

Pennsylvania and several years as a member of the executive committee of the Association for Computational Linguistics. Currently, I serve on the Technology Policy Committee of the Association for Computing Machinery (USTPC). As an adjunct lecturer, I taught classes in the College of Information Studies (The “iSchool”) at the University of Maryland in College Park from 2007. While my specialty has always been in artificial intelligence and natural language processing, teaching at the graduate level kept me current in a broad base of technologies pertaining to speech recognition and natural language processing systems.

10. Between 1985 and 1994, after completing my doctorate, I was employed as a computer scientist with General Electric (“GE”) Corporate Research and Development, where I worked on a broad range of speech recognition and natural language processing research and development, including text processing and written and spoken language interfaces. I also consulted for Infonautics, an early Internet information services and advanced search company. I was the editor of a book, entitled “Text-Based Intelligent Systems.” The book was a collection of papers based on a symposium I chaired in 1990, which brought together leaders of the field of Information Retrieval to address issues related to large-scale advanced text processing. My work foreshadowed subsequent advances in AI that captured the power of large quantities of real data to simulate human-like capabilities.

11. During my years at GE, I was principal investigator for the GE team in the Tipster program, sponsored by the Advanced Research Projects Agency (ARPA) of the United States Department of Defense and other government agencies. The technology developed in Tipster formed the foundation of the first web search engines, and my work as an ARPA principal investigator in natural language kept me in close contact with research in related fields, particularly spoken language recognition. For example, I served on the program committee for ARPA's Speech and Natural Language Conference.

12. I joined SRA International ("SRA") in the latter part of 1994 and became director of media information technologies. My responsibilities included new ventures and technology related to the Internet and the World Wide Web. From 1994 until 2002, as the Web came of age, I held a series of technology and business management jobs in organizations focused on networked information management applications. For example, from late 1999 through the end of 2000, I was president and then CTO of AnswerLogic, one of the first companies directed toward answering naturally-expressed user questions on the Web. During this time, we also experimentally developed spoken-language versions of our technology.

13. From my initial academic training in artificial intelligence, which was supervised by Marvin Minsky at MIT's Artificial Intelligence Laboratory while I was completing my Master's work at Harvard, through my career in commercial and

academic R&D and as a graduate instructor, I was closely involved in the advances in speech and language technology. I visited Verbex, the predecessor of Dragon Systems, while at Harvard, made multiple invited trips to Bell Laboratories and while at Berkeley regularly participated in seminars and collaboration with SRI, where I witnessed the birth of the company that later became Nuance, to which the '659 patent was assigned (the commercial heritage of Nuance is confusing, because the name belonged to another company called Scansoft that was in a different business, but Nuance's speech technology that was later used in Siri came from the SRI spinoff that was purchased by Nuance). As a DARPA principal investigator, I participated regularly in some of the first Speech and Language Workshops (sometimes known as Human Language Technology workshops) where we collaborated and tested against shared "benchmarks". As mentioned above, I later collaborated with Nuance and others in integrating question answering technology with speaker-independent spoken language recognition.

14. Based on my experiences described above, and as indicated in my Curriculum Vitae, I am qualified to provide the following opinions with respect to the patents in this case. While my extensive background in these areas was well beyond the capabilities of one of ordinary skill in August 2005 (and February 2011), I am qualified to opine on what one of ordinary skill would have known and

understood as of the alleged priority date of the '659 Patent (August 2005) and the actual filing date of the '659 Patent (February 2011).

15. A more complete recitation of my professional experience including a list of my journal publications, patents, conference proceedings, book authorship, and committee memberships may be found in my Curriculum Vitae, attached hereto as Appendix A.

III. MATERIALS CONSIDERED AND INFORMATION RELIED UPON REGARDING '659 PATENT

16. In preparing this declaration, I have reviewed the following materials bearing Exhibit Nos. that I understand are being referenced in the IPR to which my declaration accompanies:

LIST OF EXHIBITS

No.	Description
1001	U.S. Patent No. 8,620,659 (“ 659 patent ”)
1002	File History of U.S. Patent No. 8,620,659
1004	U.S. Patent Application No. 11/200,164 to Di Cristo et al. (“ Grandparent Application ”)
1005	U.S. Patent Application Pub. No. 2007/0038436 to Di Cristo et al. (“ 2007 Grandparent Pub. ”)
1006	U.S. Patent Application No. 12/571,795 to Di Cristo et al. (“ Parent Application ”)
1007	U.S. Patent Application Pub. No. 2010/0023320 to Di Cristo et al. (“ 2010 Parent Pub. ”)
1008	U.S. Patent Application Pub. No. 2005/0054381 to Lee et al. (“ Lee ”)

No.	Description
1009	U.S. Patent No. 6,622,119 to Ramaswamy et al. (“ Ramaswamy ”)
1010	PCT Patent Application No. WO 2000/011571 to Hartono et al. (“ Hartono ”)
1014	Ganesh Ramaswamy et al., <i>A Pervasive Conversational Interface for Information Interaction</i> , 6th Eur. Conf. on Speech Commc’n and Tech. (Sep. 1999)
1016	Julie A. Jacko and Andrew Sears, <i>The Human-Computer Interaction Handbook - Fundamentals, Evolving Technologies and Emerging Applications</i> , Lawrence Erlbaum Assocs. (2003)
1017	Mazin Rahim et al., <i>Robust Numeric Recognition in Spoken Language Dialogue</i> , Speech Comm., Vol. 34, 195-212 (2001)
1018	Ganesh N. Ramaswamy and Jan Kleindienst, <i>Hierarchical Feature-Based Translation for Scalable Natural Language Understanding</i> , 6 th Int’l Conf. on Spoken Language Processing (Oct. 2000)
1020	Craig A. Knoblock and Yigal Arens, <i>An Architecture for Information Retrieval Agents</i> , AAAI Tech. Rep. SS-94-03, 49-56 (1994)
1021	Lawrence R. Rabiner, <i>A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition</i> , IEEE, Vol. 77, No. 2, 257-286 (Feb. 1989)
1022	Jelinek, F. et al., <i>Design of a Linguistic Statistical Decoder for the Recognition of Continuous Speech</i> , IEEE Transactions of Info. Theory, Vol. IT-21, No. 3, 250-256 (May 1975)
1023	Sadaoki Furui, <i>Speaker-Independent Isolated Word Recognition Using Dynamic Features of Speech Spectrum</i> , IEEE Transactions on Acoustics Speech and Signal Processing, Vol. ASSP-34, No. 1, 52-59 (Feb. 1986)
1024	PCT Patent Application No. WO 2004/021706 to Goronzy et al.
1025	Mikael Sollenborn and Peter Funk, <i>Category-Based Filtering and User Stereotype Cases to Reduce the Latency Problem in</i>

No.	Description
	<i>Recommender Systems</i> , 6th Eur. Conf. on Case Based Reasoning, 395-405 (Sep. 2002)
1026	Diane Kelly and Nicholas J. Belkin, <i>A User Modeling System for Personalized Interaction and Tailored Retrieval in Interactive IR</i> , ASIST, 316-325 (2002)
1027	Dan Ariely, et al., <i>Learning by Collaborative and Individual-Based Recommendation Agents</i> , Journal of Consumer Psychology, 14(1&2), 81-95 (2004)

IV. UNDERSTANDING OF PATENT LAW

17. I am not an attorney. For the purposes of this declaration, I have been informed about certain aspects of the law that are relevant to my opinions. My understanding of the law was provided to me by the Petitioner's attorneys.

18. I understand that when considering the scope of the claims of a patent that the patent claim terms should generally be given the ordinary meaning that the terms would have to a person of ordinary skill in the art in question after reading the patent as of the earliest claimed priority date. As discussed below, it is my opinion that the challenged claims are not entitled to the earliest priority date asserted by the '659 patent. However, in my opinion, the meaning of the claim terms would not have materially differed to a person of ordinary skill in the art whether the claims were evaluated at the time of the earliest claimed priority date (August 10, 2005) or at the time the application that led to the '659 patent was filed (February 7, 2011).

19. I understand that the person of ordinary skill in the art is deemed to read the claim term not only in the context of the particular claim in which the term appears, but in the context of the entire patent, including the other claims and the specification of the application as filed. I further understand that the principal considerations regarding the scope and meaning of the claims are the plain language of the claim (including the surrounding claim language and context), the patent specification, and the prosecution history. I understand that while a claim is to be read in light of the specification, one must generally avoid importing limitations into the claim from the specification. I am also informed that the prosecution history can often inform the meaning of the claim by demonstrating how the inventor understood the invention and whether the inventor limited the invention in the course of prosecution, making the claim scope narrower than it would otherwise be. I applied these understandings when considering the scope and meaning of the claims of the '659 patent.

20. I understand that a claim is unpatentable if it would have been obvious. Obviousness of a claim requires that the claim would have been obvious from the perspective of one of ordinary skill at the time the alleged invention was made. I understand that a claim could have been obvious from a single prior art reference or from a combination of two or more prior art references. I understand that an obviousness analysis requires an understanding of the scope and content of the prior

art, any differences between the alleged invention and the prior art, and the level of ordinary skill in evaluating the pertinent art.

21. I further understand that a claim would have been obvious if it unites old elements with no change to their respective functions, or merely substitutes one element for another known in the field, and that combination yields predictable results. While it may be helpful to identify a reason for this combination, I understand that there is no strict requirement of finding an express teaching, suggestion, or motivation to combine within the references. When a product is available, design incentives and other market forces can prompt variations of it, either in the same field or different one. If one of ordinary skill can implement a predictable variation, obviousness likely bars its patentability. For the same reason, if a technique has been used to improve one device and a skilled artisan would have recognized that it would improve similar devices in the same way, using the technique would have been obvious. I understand that a claim would have been obvious if common sense directs one of ordinary skill, without hindsight bias caused by knowledge of the challenged claims, to combine multiple prior art references or add missing features to reproduce the alleged invention recited in the claims.

22. I further understand that certain factors may support or rebut the obviousness of a claim. I understand that such secondary considerations include, among other things, commercial success of the patented invention, skepticism of

those having ordinary skill in the art at the time of invention, unexpected results of the invention, any long-felt but unsolved need in the art that was satisfied by the alleged invention, the failure of others to make the alleged invention, praise of the alleged invention by those having ordinary skill or expertise in the art, and copying of the alleged invention by others in the field. I understand that there must be a nexus—a connection—between any such secondary considerations and the alleged invention. I also understand that contemporaneous and independent invention by others is a secondary consideration tending to show obviousness.

23. I understand that for a patent claim to be entitled to priority to the filing date of a previously filed application, the previously filed application must provide written description support for the claim such that the disclosure of the application relied upon reasonably conveys to those skilled in the art that the inventor had possession of the claimed subject matter as of the filing date. Similarly, I understand that disclosure of the earlier filed application must enable a skilled artisan to make and use the full scope of the claim without undue experimentation. I understand that relevant factors to consider in determining whether undue experimentation would be required include (1) the quantity of experimentation necessary, (2) the amount of direction or guidance presented, (3) the presence or absence of working examples, (4) the nature of the invention, (5) the state of the prior art, (6) the relative skill of

those in the art, (7) the predictability or unpredictability of the art, and (8) the breadth of the claims.

24. I am not aware of any allegations by the named inventors of the '659 patent or any assignee of the '659 patent that any secondary considerations tend to rebut the obviousness of any Challenged Claim of the '659 patent.

25. I understand that in considering obviousness, it is important not to determine obviousness using the benefit of hindsight derived from the descriptions found in the patent being considered.

26. I understand that Petitioner has the burden of proving unpatentability by a preponderance of evidence, which means that the claims are more likely than not unpatentable.

V. GROUNDS

27. I understand that Petitioner requests *inter partes* review of claim 42 (the “Challenged Claim”) of the '659 patent, on the following statutory grounds:

	Reference(s)	Basis	Claim
Ground 1	2007 Grandparent Pub. (EX1005) ¹ in view of Lee (EX1008) ²	35 U.S.C. § 103	42
Ground 2	Ramaswamy (EX1009) ³	35 U.S.C. § 103	42
Ground 3	Ramaswamy in view of Ramaswamy NPL (EX1014) ⁴	35 U.S.C. § 103	42
Ground 4	Ramaswamy in view of Hartono (EX1010) ⁵	35 U.S.C. § 103	42

VI. THE '659 PATENT

28. I understand that the '659 patent issued from U.S. Patent Application No. 13/022,370 (the "'370 application"), filed on February 7, 2011. The '370 application is a continuation of application No. 12/571,795 (the "Parent Application"), filed on October 1, 2009, and which published as U.S. Patent Publ. No. 2010/0023320 (the "2010 Parent Pub.", EX1007), which is a continuation of

¹ U.S. Patent App. Publ. No. 2007/0038436.

² U.S. Patent App. Publ. No. 2005/0054381.

³ U.S. Patent No. 6,622,119.

⁴ Ganesh Ramaswamy et al., "A Pervasive Conversational Interface for Information Interaction," 6th Eur. Conf. on Speech Commc'n and Tech. (Sep. 1999).

⁵ PCT Patent Application No. WO 2000/011571.

application No. 11/200,164 (“the Grandparent Application”), filed on August 10, 2005, and which published as U.S. Patent Publ. No. 2007/0038436 (the “2007 Grandparent Pub”, EX1005).

29. The ’659 patent’s specification describes a speech-based information query, processing and presentation system that includes one or more speech recognition engines that purportedly “may enhance responses to commands and questions by including a prediction of user behavior.” EX1001, 2:62-64, 13:52-53; *see also id.*, Abstract, 2:41-3:25, 13:6-26, 13:45-51, 14:4-30.

30. FIG. 1 of the patent shows a system that includes a speech recognition engine. *E.g., id.*, 12:28-53, 15:14-19, 22:62-23:20, FIG. 1. The speech recognition engine “recognizes words and phrases” (*id.*, 22:64) and transcribes verbal utterances to text. *Id.*, 2:46-49, 3:65-67, 12:62-65, 16:27-29. This recognized text then “passes [] to the parser [] for interpretation.” *Id.*, 22:64-66. The parser determines the correct domain agent to process the command or request (*e.g., id.*, 25:12-13) and “may form the question or command in a standard format or hierarchical data structure” for the domain agent. *Id.*, 24:62-63.

31. FIG. 8 shows one or more speech recognition engines included in a “conversational speech analyzer” that receives speech input from a user. *Id.*, 2:41-43, 12:54-57. The speech recognition engines “transcribe utterances to textual messages.” *Id.*, 12:63-65. “Based on information received from [a] general cognitive

model ... and/or [a] personalized cognitive model, the system may enhance responses to commands and questions by including a prediction of user behavior.”

Id., 2:60-64; *see also id.*, 13:49-53.

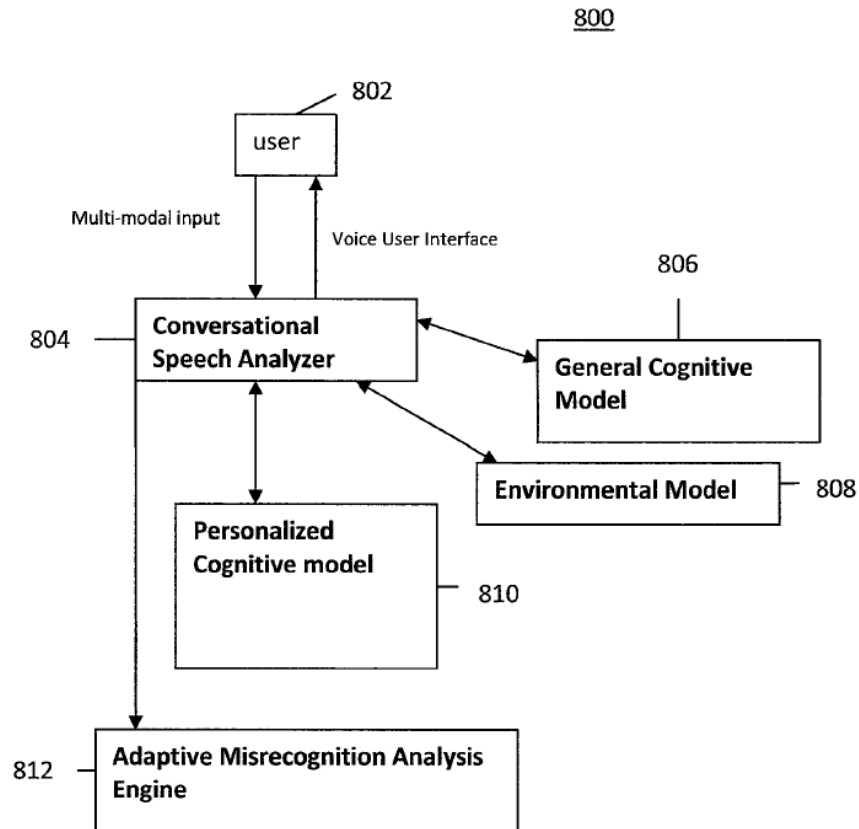


FIGURE 8

EX1001, FIG. 8.

32. The “personalized cognitive model [] is a model derived from a user’s interaction pattern with the system and may be used to predict what actions the user may take next in time, thus assisting with speech recognition and/or question or command recognition.” *Id.*, 13:6-11.

33. The general cognitive model, on the other hand, “is a statistical abstract that corresponds to interaction patterns with the system *for multiple users*.” *Id.*, 13:17-20 (emphases added). Like the personalized cognitive model, however, “[d]ata stored within general cognitive model [] may be used to predict a user's next action, thus assisting with speech recognition and/or question or command recognition.” *Id.*, 13:20-22. “[T]he personalized cognitive model may be consulted first” (*id.*, 13:13-14), while the generative cognitive model “may be used when the user interacts with the system in a way that is not handled in the personalized cognitive model” (*id.*, 13:24-26).

A. Prosecution History Of The '659 Patent

34. I have reviewed the file history of the '659 patent; however, at this time, the file history does not inform the meaning of the claims or prior art.

VII. LEVEL OF SKILL IN THE ART

A. Person Of Ordinary Skill In The Art

35. The person of ordinary skill in the art in August 2005 (“POSITA”) would have had a bachelor’s degree in electrical engineering, computer science, computer engineering, or equivalent field, and two years of work experience working with speech recognition and natural language processing systems. Additional work experience could make up for less education and vice versa. This definition would not differ meaningfully were the date in question February 2011.

In this Declaration, I use the terms “one of ordinary skill” and “skilled artisan” as synonymous with the “person of ordinary skill in the art.”.

VIII. CLAIM CONSTRUCTION

36. I have been instructed to apply the plain and ordinary meaning of all claim terms.

A. “Domain Agent”

37. I understand that in a case involving Patent Owner and third-party Amazon.com, the District Court for the Eastern District of Virginia construed the term “**domain agent**” (EX1012, 2) recited in the claims of other Dialect patents (U.S. Patent Nos. 8,195,468 and 9,263,039) as “**software with domain-specific behavior and information.**” *Id.*, 2 (emphasis added). I also understand that Patent Owner and another third party agreed upon that same construction in another case involving those and other Dialect patents (U.S. Patent Nos. 7,640,160 and 8,447,607). EX1008, 2. While I understand that Petitioner does not adopt this construction, the prior art discussed below renders claim 42 obvious including under this construction.

IX. THE ANCESTOR APPLICATIONS DO NOT SUPPORT CLAIM 42’S STEP OF DETERMINING WHETHER A PERSONALIZED COGNITIVE MODEL ASSOCIATED WITH THE USER INCLUDES SUFFICIENT INFORMATION FOR PREDICTING ONE OR MORE ACTIONS ASSOCIATED WITH THE USER

38. Claim 42 recites that its computer implemented method performs the step of “**determining whether a personalized cognitive model associated with the user includes sufficient information for predicting one or more subsequent actions associated with the user.**” EX1001, 36:59-62 (emphasis added). As explained below, the specification of the Ancestor Applications, which are substantively the same as the ’659 patent specification,⁶ do not enable or provide written description support for this limitation.

1. Claim 42 Lacks Written Description Support

39. It is my opinion that the written descriptions of the Ancestor Applications do not reasonably convey to one of ordinary skill that the inventors of the ’659 patent had possession of a method that includes the step of “determining whether a personalized cognitive model includes sufficient information for predicting one or more subsequent actions associated with the user.” EX1001, 36:59-

⁶ Each of the ’659 patent and its Ancestor Applications share the same Abstract, Figures, and Specification Description. They differ substantively only in their claims.

62. The ancestor specifications state only the following as to when a personalized cognitive model is used: “When the system is attempting to predict user behavior, the personalized cognitive model may be consulted first[,]” and that a general cognitive model “may be used when the user interacts with the system in a way that is not handled in the personalized cognitive model.” EX1005, [0101]-[0102]; EX1007, [0101]-[0102]; *see also* EX1001, 13:12-14, 13:23-26. The ancestor specifications do not describe (1) what information is included in the personalized cognitive model; (2) how to measure or otherwise quantify such information; and (3) how to determine whether the information is sufficient or insufficient (or what it means to be sufficient or insufficient). The specifications simply do not make any mention of a sufficiency determination for information in a personalized cognitive model.

40. Further, if the support for “determining ... sufficient” in claim 42 was deemed to rely on “the user interacts with the system in a way that is not handled,” (EX1001, 13:24-25) one of ordinary skill would have no way of knowing what sort of interaction would not be handled, other than to try the system and test whether a user action might be predicted. Such an approach would not, in my opinion, comprise a determination that the information in the personalized cognitive model is “sufficient”. Nor do the ancestor specifications give any hint that the personalized model’s handling of a particular user interaction depends on the sufficiency of the

model's information.⁷ For example, the specifications do not say: "use the personalized model only when it includes sufficient information." Thus, the Ancestor Applications do not provide written description support for claim 42

⁷ One of skill in the art would have been aware of various ways in which a personalized cognitive model would not have been able to handle a given user interaction that are unrelated to determining the sufficiency of information in the model. For example, a user interaction would not be handled in a personalized cognitive model in instances where a user is not specified or the system is unable to identify the given user. A skilled artisan understood that a system would be unable to utilize a model associated with a given user if the system did not know the identity of the user. As another example, a skilled artisan understood that a personalized model might not handle a given interaction based on a simple determination that an interaction has not been performed before. *See* EX1021, 319 (describing supplementing a personal behavior model with a general behavior model in instances of "previously unseen behavior"). A determination that a given interaction hasn't been performed before is distinct from determining whether a personal cognitive model includes sufficient information to predict an action. For example, even if a system were to determine that an interaction was previously determined, there may still be insufficient information in a personal model to predict a user action.

because they do not reasonably convey to a skilled artisan that the inventors had possession of a method that includes “determining whether a personalized cognitive model associated with the user includes sufficient information for predicting one or more subsequent actions associated with the user.” EX1001, 36:59-62.

2. Claim 42 Is Not Enabled

41. In addition to lacking written description support, it is my opinion that the Ancestor Applications do not enable one of ordinary skill in the art to use the method of claim 42 without undue experimentation. The '659 patent's specification lists 21 separate “example applications for the invention,” including finding “information for businesses and individuals,” “communications control functions,” “map” functions, “home entertainment ... information and device control,” “weather information,” “home organization,” and digital media management, among many others. EX1001, 10:1-65. Among those 21 examples the specification mentions one that explicitly relies on a user profile: “interactive educational programs using local and network material, with lesson material level set based on user's profile.” EX1001, 10:60-62. In the world of online learning, it would have been a research topic even to determine how to select appropriate educational material based on a user profile, much less to determine whether that user's interactions would have predicted their subsequent behavior.

42. One of ordinary skill would have recognized that each contemplated application would include different user interactions and that predicting subsequent user actions would require varying types and quantities of data depending on the user interaction to be predicted. The specifications provide no idea about how to handle individual use cases, much less of how to implement a general approach that might determine whether a user model was sufficient. Claim 42 encompasses any implementation that determines whether a user-specific model has “sufficient information” to predict user behavior, including for each application and associated user interaction contemplated in the specifications. The specifications’ bare disclosure of falling back to a general model for user interactions “not handled in the personalized cognitive model” (EX1001, 13:25-26) offers no guidance to a skilled artisan regarding how to make that determination. For example, the ancestor specifications provide no indication of what constitutes “sufficient information;” provides no objective metric, threshold, or test for determining sufficiency; and provides not a single example or embodiment that performs such a determination.

43. As I explained above, determining how to apply a user model in even one of the envisioned applications would have been a broad research topic. To come up with and implement a method, given the disclosure in the specification regarding whether a user interaction is “handled,” that “determines ... sufficient information” for a personalized user model for the full scope of user applications contemplated by

the '659 patent, from my experience, would have required substantial research. Because the ancestor applications do not provide the skilled artisan with a path to determine whether information in a personalized user model is sufficient to predict user action the actual invention is left to that person of ordinary skill. Given the absence of guidance in the ancestor applications, one of ordinary skill in the art would be left to develop a mechanism for evaluating sufficiency of information in a user model to predict user action, invent from scratch the relevant criteria of sufficiency to be used in the evaluation, and experiment extensively to determine for each type of user application contemplated in the '659 patent at what point the criteria meet the threshold for sufficiency.

**X. GROUND 1: CLAIM 42 IS OBVIOUS OVER
THE 2007 GRANDPARENT PUB. IN VIEW OF LEE**

A. 2007 Grandparent Pub. (EX101005)

44. The 2007 Grandparent Pub. is a published U.S. patent application that published in 2007. I understand that it qualifies as prior art under at least Section 102(b) (pre-AIA) because it published more than one year before the 2011 filing date of '659 patent's application. The specification of 2007 Grandparent Pub. is substantively identical to that of the '659 patent, aside from their claims.

B. Lee (EX1008)

45. Lee is a published U.S. patent application that published in 2005. I understand that it qualifies as prior art under at least Section 102(b) (pre-AIA)

because it published more than one year before the 2011 filing date of '659 patent's application.

46. Lee is analogous art to the '659 patent at least because it describes systems and methods for processing natural language utterances. *E.g.*, EX1008, [0007] (describing natural language processing), [0025] (describing bi-directional communication between a user and an intelligent agent), [0028], [0056], [0230], FIG. 12. Moreover, the '659 patent expressly describes that its methods can be used for telephone number lookup and "automatic telephone dialing," which relates to Lee's disclosures of suggesting next telephone calls to a user. EX1001, 10:1-8; EX1008, [0259]-[0305].

47. Lee teaches a computational device having a "proactive user interface [that] actively makes suggestions to the user, based upon prior experience with a particular user and/or various preprogrammed patterns from which the computational device could select, depending upon user behavior." EX1008, Abstract; *see also id.*, [0069]. One specific application of Lee is a "mobile information device" (*id.*, [137]) including user modeling functionality for predicting and suggesting a user's next telephone call (*id.*, [0273]-[0305]). Lee describes a function CallStateHandler that is "preferably a rule based algorithm that listens to call events" and generates suggested call actions. *Id.*, [0274]. "CallStateHandler [] also preferably maintains a rule base that is responsible for two major functions:

Machine Learning, which maintains the call associations knowledge base; and the AI probability based inference of whether to suggest a number for a telephone call to the user.” *Id.* Lee thereby “suggest[s] to the user a telephone number for placing a following call after a telephone call has ended, or for an alternative telephone call after a telephone call has failed.” *Id.*, [0276].

48. Lee discloses that the knowledge base maintained by the CallStateHandler function is “preferably designed as a history window in which associations are added as occurrences in the history of the subject.” *Id.*, [0289]. “The history window size is preferably defined by the algorithm as the number of associations (occurrences) which the algorithm is to manage (or remember).” *Id.*, [0292]. Based on the user’s previous calls in the history window, Lee predicts the probability of a user calling a particular contact following a call to a first contact according to a “probability based inference” (*id.*, [0293]) equation:

$$P(C_i) = \text{Max}(H_o(C_i)/H_{\text{size}})$$

(*id.*, [0297]), where $H_o(C_i)$ is the number of times contact C_i is found in the history window for a first contact C_0 (*id.*, [0294]-[0295]) and H_{size} is the history window size (*id.*, [0296]). Thus, following a call to a first contact, Lee infers the probability a user will make a next call for a second contact based on the frequency of the second contact appears in the first contact’s history window.

49. Lee discloses determining whether its inference process is successful based on the determined probability and the amount of information contained in the history window for a given contact. Lee's inference process is considered successful "only if it can infer a probability that is more than % 50 and preferably also if the history window is full." *Id.*, [0300]. Thus, Lee teaches determining whether its model for predicting a next user action (a telephone call) includes sufficient information in two ways: 1) based on the robustness of the call history (e.g., the history window being full) and 2) based on whether the history window for a contact contains sufficient information to predict a next contact to call with a probability greater than 50%.

C. Combining The 2007 Grandparent Pub. And Lee Teachings

50. One of ordinary skill would have been motivated with a reasonable expectation of success to combine the 2007 Grandparent Pub.'s disclosed natural language processing system and method with Lee's method of predicting a next user telephone call, including whether a user model includes sufficient information to make such prediction. Specifically, one of ordinary skill would have combined the 2007 Grandparent Pub.'s natural language processing system including a personalized cognitive model to predict user actions (EX1005, [0101]) with Lee's specific teachings of using a knowledge base of prior phone call interactions to

predict a next phone call and to determine whether such a prediction would be successful (EX1008, [0274]-[0300]).

51. The 2007 Grandparent Pub. discloses using cognitive models to predict a user's action based on a history of user interactions (EX1005, [0101]-[0102]), that its disclosed method is used on a mobile telephone device (*id.*, [0019], [0045], [0054]), and that example use cases include look up of telephone numbers and "automatic telephone dialing" (*id.*, [0061]). The 2007 Grandparent Pub., however, provides no specific teachings on how to predict user actions based on user interaction history in the context of telephone calls. One of ordinary skill would have therefore sought guidance for such a use case and would have come across Lee's teachings. Lee specifically discloses how a personalized cognitive model would utilize user interaction history to predict a user's next action (i.e., a telephone call), which would be directly applicable to the 2007 Grandparent Pub.'s intended applications of automatic telephone dialing and telephone number lookup.

52. The 2007 Grandparent Pub. also discloses that a personalized cognitive mode "may be consulted first" (*id.*, [0101]) and that the "general cognitive model ... may be used when the user interacts with the system in a way that is not handled in the personalized cognitive model" (*id.*, [0102]) but does not further explain when or how a personalized cognitive model would not handle a specific interaction. Thus, one of ordinary skill would have recognized that Lee provides guidance on when a

personalized cognitive model would not be able handle a given user interaction in the context of predicting a user's next telephone call, i.e., by determining whether a model's knowledge base contains sufficient historic interaction data to make a successful prediction. EX1008, ([0293]-[0300]). One of ordinary skill would have recognized the benefit of Lee's teachings of determining whether a prediction is successful and to not provide an unsuccessful prediction to a user, because providing predictions with low probability of being correct would adversely affect system usability, potentially waste a user's time, and reduce user confidence in the system. *See* EX1027, 93-94 (recognizing that users will abandon recommendation systems having low recommendation quality).

53. One of ordinary skill would have been further motivated to combine references because the 2007 Grandparent Pub. describes the benefit of predicting user actions in a natural language processing system, stating that its personalized cognitive model "may be used to predict what actions the user may take next in time, **thus assisting with speech recognition and/or question or command recognition.**" EX1005, [0101] (emphasis added). One of ordinary skill would have therefore been motivated to implement Lee's specific teachings concerning predicting user calls in order to assist speech recognition and question or command recognition. For example, the natural language processing system would be able to

more easily recognize a spoken number or name to call if the system already predicted that a user would make such a command.

54. The skilled artisan would have had a reasonable expectation of success of incorporating Lee's teachings into the 2007 Grandparent Pub.'s disclosed system and method. They would have understood the references to be compatible because both describe teachings for natural language processing systems on telephonic devices. Lee offers further guidance on how to implement the 2007 Grandparent's teachings by 1) providing further teachings concerning how to predict user actions related to telephone calls, and 2) providing further teachings on when a personalized cognitive model would not be able to handle such a prediction. One of ordinary skill would have therefore reasonably expected to successfully implement the relevant teachings of Lee concerning call prediction into the natural language processing system of the 2007 Grandparent Pub.

55. Thus, one of ordinary skill would have been motivated with a reasonable expectation of success to incorporate Lee's teachings on when and how to predict a user's next call based on a history of user interactions into the personalized cognitive model based methodology of the 2007 Grandparent Pub.

D. Claim 42 Mapping

56. As set forth below, the 2007 Grandparent Pub. in combination with Lee renders claim 42 obvious.

1. Element [42.1]⁸

[42.1] A method of processing natural language utterances, the method being implemented by a computer system that includes one or more processors executing one or more computer program instructions which, when executed, perform the method, the method comprising:

57. The 2007 Grandparent Pub. teaches this claim element. *E.g.*, EX1005, Abstract, [0007], [0009], [0020], [0095], [0123], [0231], claim 14, FIG. 1. For example, the 2007 grandparent Pub. discloses a “method of processing speech and non-speech communications.” *Id.*, claim 26. The method “may take the form of a computer program containing one or more sequences of machine-readable instructions” that are stored in memory. *Id.*, [0231]. One of ordinary skill knew that these machine-readable instructions are executed by one or more processors, since this is how machine-readable instructions are executed.

2. Element [42.2]

[42.2] receiving a first input of a user that comprises a natural language utterance;

58. The 2007 Grandparent Pub. teaches this claim element. *E.g.*, EX1005, Abstract, [0007], [0009], [0088], [0100], [0109], [0215], claim 26, FIGS. 1, 3, 8. For example, the 2007 Grandparent Pub. discloses “receiving speech and/or non-speech

⁸ Reference numbers in the format of [claim#.limitation#] are added throughout for ease of reference.

communications of natural language questions and/or commands.” *Id.*, Abstract.
 “[S]peech units [] may include one or more microphones, for example array microphone [], to receive the utterances from the user.” *Id.*, [0109].

3. **Element [42.3]**

[42.3] generating an interpretation of the natural language utterance based on one or more recognized words of the natural language utterance;

59. The 2007 Grandparent Pub. teaches this claim element. *E.g.*, EX1005, [0007], [0095], [0104], [0176], [0215]-[0216], [0218], [0223], [0227], FIGS. 4A-5. For example, the “speech recognition engine [] recognizes words” in the natural language utterance and then the parser generates an interpretation of the utterance based on these recognized words. *Id.*, [0176]; *see also, e.g., id.*, [0181]-[0185].

4. **Element [42.4]**

[42.4] generating a request based on the interpretation;

60. The 2007 Grandparent Pub. teaches this claim element. *E.g.*, EX1005, [0007], [0095], [0104], [0185]-[0186], [0215]-[0216], [0219]-[0220], [0223]-[0224], [0227]-[0229], FIGS. 3-4B. For example, the parser “form[s] the question or command in a standard format or hierarchical data structure used for processing by the agents.” *Id.*, [0186].

5. Element [42.5]

[42.5] transmitting the request to a domain agent for processing;

61. The 2007 Grandparent Pub. teaches this claim element. *E.g.*, EX1005, [0119]-[0120], [0122], [0181], [0186], [0191]-[0192], [0216], [0219]-[0220], [0224]. For example, “[c]ommands for external entities are generally processed by a domain agent.” *Id.*, [0192]. The parser sends the request to a domain agent, which formats and executes the request. *E.g.*, *id.*, [0186]-[0194], [0219]-[0220]. “[T]he parser [] evokes the correct agent to process the question or command,” “[t]he agents ... may receive a command ... once the parser [] has placed it in the required standard format” (*id.*, [0191]), and the domain agents “generate their own requests and/or commands ... [which] may be formatted according to the requirements of the target sources” (*id.*, [0220]). Once the agents have properly formatted the requests, they execute them (e.g., in an asynchronous manner). *Id.*, [0193]-[0201], [0220].

6. Element [42.6]

[42.6] determining whether a personalized cognitive model associated with the user includes sufficient information for predicting one or more subsequent actions associated with the user, wherein the personalized cognitive model is generated based on a tracking of a pattern of interactions between the user and the system, and wherein the one or more subsequent actions include one or more actions predicted to occur after receiving the first input; and

62. The combination of the 2007 Grandparent Pub. and Lee teaches this claim element. The 2007 Grandparent Pub. discloses a personalized cognitive model associated with the user, generated based on a tracking of a pattern of interactions between the user and the system. *E.g.*, EX1005, Abstract, [0010]-[0013], [0100]-[0101], [0104], [0106]-[0107], [0218]-[0219], claim 28. For example, “personalized cognitive model [] is a model derived from a user's interaction pattern with the system and may be used to predict what actions the user may take next in time.” *Id.*, [0101]. Lee teaches details for using such a model in the specific use case of call prediction and teaches determining whether such a model includes sufficient information for predicting a subsequent user action, wherein the subsequent user action (i.e., a next phone call) is predicted to occur after receiving a first input (i.e., an initial phone call). EX1008, [0273]-[0300].

63. For example, the 2007 Grandparent Pub. discloses that the personalized cognitive model “is a model derived from a user’s interaction pattern with the system and may be used to predict what actions the user may take next in time.” *Id.*, [0101]. Though the 2007 Grandparent Pub. does not teach determining whether the personalized cognitive model includes sufficient information, Lee does.

64. Lee discloses a user model including a “knowledge base” (*id.*, [0274]) with the knowledge base preferably designed as a “history window” (*id.*, [0289]) tracking associations between contacts based on prior user interactions. Lee

discloses using its “probability based inference” (*id.*, [0293]) algorithm to predict a next contact for a user to call and that such prediction “is considered to be successful only if it can infer a probability that is more than % 50 and preferably also if the history window is full.” *Id.*, [0300]. As disclosed by Lee, the model predicts the probability a user will call a second contact after calling a first contact based on the frequency with which the second contact appears in the first contact’s history window. *Id.*, [0290]-[0299]; *see also supra* Section X.B. For example, if the user called a second contact six times in a first contact’s history window of ten calls, the probability of the user calling that second contact after calling the first contact is 0.6. *Id.* Thus, the method of Lee determines whether a prediction will be successful based on two separate information sources: first, there must be sufficient prior interaction data present to fill the history window; and second, the data present in the history window must be sufficient to provide a prediction probability greater than 50%. One of ordinary skill would have understood that by determining whether sufficient information is included in the knowledge base to successfully predict a next user phone call, Lee teaches determining whether a personalized cognitive model contains sufficient information for predicting one or more subsequent user actions.

65. When implementing the 2007 Grandparent Pub.’s teachings to consult the personalized cognitive model first (EX1005, [0101]) but use the generalized cognitive model “when the user interacts with the system in a way that is not handled

in the personalized cognitive model” (*id.*, [0102]), one of ordinary skill would have looked to the teachings from Lee on when and how to use a personalized cognitive model to predict a next call. Specifically, one of ordinary skill would have understood that where the personalized cognitive model cannot successfully predict a subsequent phone call, then this constitutes an interaction not handled by the personal cognitive model. Where the personalized cognitive model includes sufficient information (i.e. enough information to provide a sufficiently high-probability call prediction and full history window), then one of ordinary skill would have understood to use the personalized cognitive model. Conversely, when the personalized cognitive model does not include sufficient information, one of ordinary skill would have understood to use the generalized cognitive model.

7. **Element [42.7]**

[42.7] predicting the one or more subsequent actions based on a generalized cognitive model in response to a determination that the personalized cognitive model does not include the sufficient information, wherein the generalized cognitive model is generated based on a tracking of patterns of interactions between a plurality of users and the system.

66. The 2007 Grandparent Pub. teaches this claim element. *E.g.*, EX1005, Abstract, [0010]-[0011], [0100], [0102], [0104], [0218], claims 31-33. For example, the 2007 Grandparent Pub. discloses that the generalized cognitive model “is a statistical abstract that corresponds to interaction patterns with the system for multiple users” and “may be used to predict a user’s next action.” *Id.*, [0102].

Further, the generalized cognitive model “may be used when the user interacts with the system in a way that is not handled in the personalized cognitive model.” *Id.* Thus, when incorporating Lee’s teachings to determine whether the personalized cognitive model includes sufficient information, one of ordinary skill would have implemented the combined system to use the generalized cognitive model to predict a subsequent user action when the personalized cognitive model does not include sufficient information at least because one of ordinary skill would have recognized that this is a situation in which the generalized cognitive model would be more accurate, and thus better equipped, to provide a prediction.

XI. GROUND 2: CLAIM 42 IS OBVIOUS OVER RAMASWAMY

67. As explained below, claim 42 is obvious over Ramaswamy.

A. Ramaswamy (EX1009)

68. Ramaswamy is a U.S. patent that issued on September 16, 2003. I understand that it qualifies as prior art under at least Section 102(b) (pre-AIA) at

least because it published more than one year before the alleged priority date of the '659 patent.

69. Ramaswamy is analogous art to the '659 patent at least because it describes systems and methods for processing natural language utterances. *E.g.*, EX1009, Title, Abstract, 2:33-35, 3:39-50, 4:10-35.

70. Ramaswamy describes systems and methods for processing spoken (*id.*, 4:18-19) natural language utterances, e.g., “do I have any new messages?” (*id.*, 4:27). The system includes a user interface, a recognition system, an NLU engine, a dialog manager, a command executor, and a command predictor.

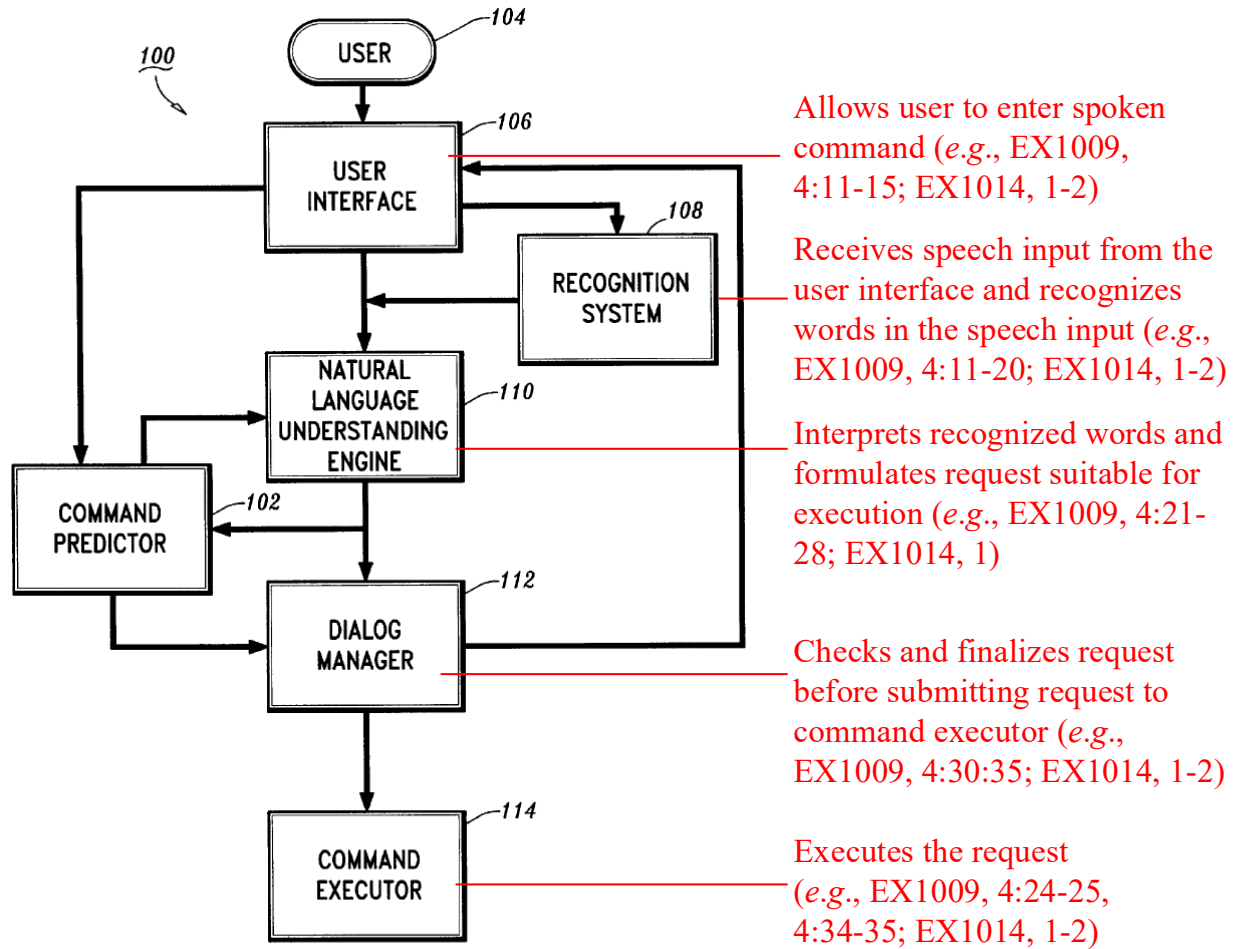


FIG. 1

EX1009, FIG. 1 (red annotations added).

1. Speech Recognition And Interpretation

71. The recognition engine recognizes words in a user's spoken command, and the natural language understanding (NLU) engine semantically parses/interprets the recognized words to determine what the user wants / is asking for and generates a corresponding command that can be executed by the computer. EX1009, 4:17-35. Specifically, the recognition engine "converts the input to text form" and "[t]he text

input is then submitted to the natural language understanding (NLU) engine [], which converts the input text to a formal command, suitable for execution by [the] command executor.” *Id.*, 4:20-25. For example, “if the user input is ‘do I have any new messages?’ the corresponding formal command may be: checkNewMail().” *Id.*, 4:26-27. Thus, the “recognition engine ... converts the [speech] input to text form” and then the NLU engine interprets the recognized words and “converts them to a formal command.” *Id.*, 4:19-24; *see also* EX1014, 1.

2. Command Generation and Execution

72. The NLU engine and dialog manager generate the computer-executable command, and the command executor executes the command. EX1009, 4:21-35, 4:51-55. Specifically, the NLU engine “converts the input text to a formal command” (*id.*, 4:23-24) and then transmits the formal command “to a dialog manager [], to be processed for execution The dialog manager [] resolves any ambiguities that may be present in the formal command, and submits the formal command to the command executor [] for execution of the command.” *Id.*, 4:29-35. Ramaswamy NPL (described below in Section XI.A.4) provides additional details on the NLU engine.

3. Command Prediction Models

73. Ramaswamy’s command predictor uses personalized and generalized prediction models to predict a user’s next command and provides this prediction to

the NLU engine, which uses it to more accurately interpret a user's natural language input. *E.g.*, EX1009, 3:19-20 ("The prediction of a next command may be used to improve the accuracy and speed of the natural understanding system."), 3:39-42 ("If the regularities exhibited by the user are captured using a statistical model, then the model may be used to improve the accuracy of the natural language understanding process used to interpret the user input."), 3:46-50 ("The system can predict the most probable choices ... and when the user subsequently presents one of these [] inputs, the input can be interpreted more accurately."), 4:29-49, 5:9-30, 6:1-24, 6:25:29 (By limiting the number of possible commands using a weighting scheme (e.g., based on a prediction of the next command), the "NLU engine [] translation will not only be more accurate, but will also result in lower computational complexity."), 8:3-9:30, FIG. 1.

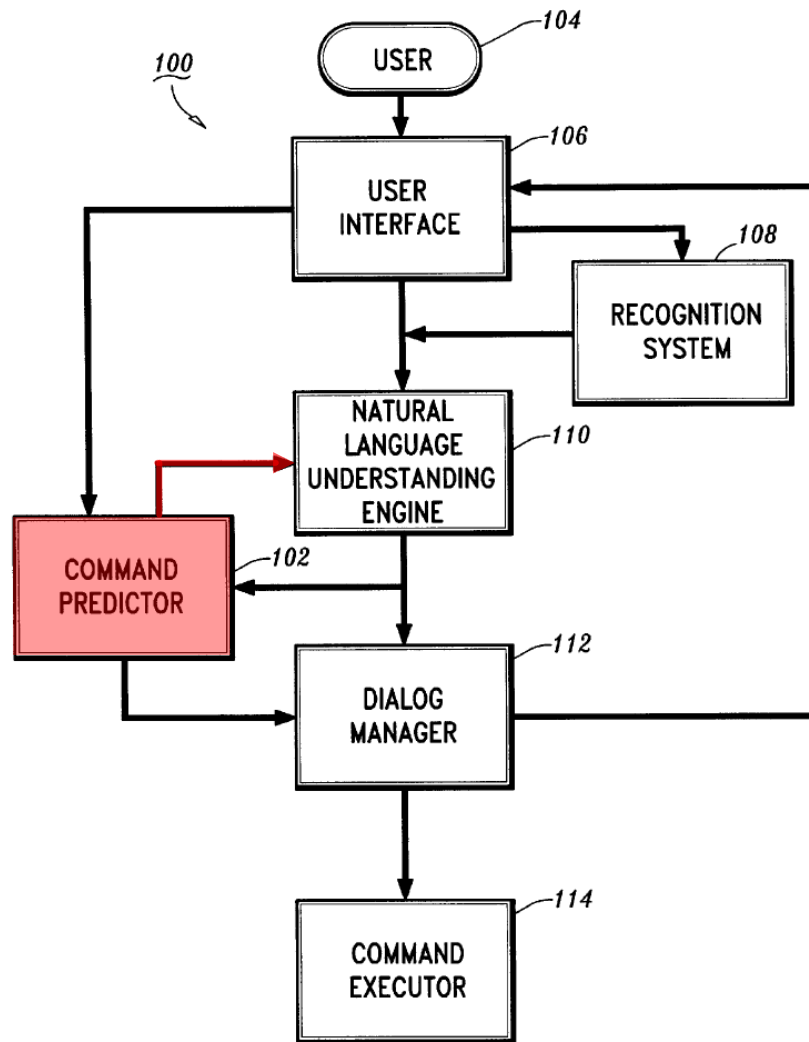


FIG. 1

EX1009, FIG. 1 (red annotation added).

74. For example, if the user typically saves or sends a new email draft, then the system may “give higher priority to the ‘send’ and ‘save’ commands” and when the user “create[s] a new message” and “subsequently presents one of these two inputs, the input can be interpreted more accurately.” EX1009, 3:44-50.

75. Ramaswamy's command predictor is shown in FIGS. 2 and 4 (*id.*, 4:59-61, 8:25-28) and includes a probability calculator (e.g., probability calculator 206) that predicts commands and their probabilities (e.g., *id.*, 5:9-12, FIGS. 2, 4) based on (1) a general prediction model (e.g., prediction model 204) (e.g., *id.*, 5:9-12, 6:30-8:19, 8:28-29, FIGS. 2-4) and/or (2) a user's personal prediction model (e.g., personal prediction model 404) generated based on the user's command history (e.g., *id.*, 1:55-2:54, 3:13-50, 4:36-51, 6:1-3, 8:19-24, 8:29-9:31, claim 1).

76. The command predictor uses only the general prediction model when the user first starts using the system, but then creates and also uses a personal prediction model once the user has interacted enough with the system to generate sufficient command history data. *Id.*, 8:38-9:30. The general prediction model "work[s] well for the average user" (*id.*, 8:19) because it is constructed from "a large number of formal commands from a large number of users" (*id.*, 6:54-55). *See also id.*, 6:48-55, 8:16-20. Thus, when "[a] new user starts using the system," the command predictor uses the general prediction model. *Id.*, 8:28-29.

77. "However, as a given user continues to use the system, additional data is generated, and additional data can be used to personalize the prediction model [], such that the system will predict commands with even greater accuracy for that particular user." *Id.*, 8:19-24. Specifically, "[e]very time the user issues a command, the resulting formal command ... [is] recorded in the command history ... [and]

written to a personal training corpus.” *Id.*, 8:29-33. The personal training corpus “continues to accumulate the data, and when there is **sufficient data** to construct a new model ... the model constructor ... construct[s] a personal prediction model.” *Id.*, 8:29-46 (emphasis added). “As the personal training corpus [] becomes larger, ... the prediction will become more personalized to the given user, and the accuracy of the command predictor will improve.” *Id.*, 9:17-20. Thus, “[t]he command predictor [] can adapt the set of formal commands to a given user, thereby providing higher accuracy with lower computations, which will result in greater satisfaction to the user.” *Id.*, 9:27-30.

78. Thus, the NLU engine interprets the recognized words based not only on the words recognized by the recognition system, but also based on a command predictor’s prediction of the user’s command, which is based on a general model that is representative of the average user and/or a personalized model that is tailored to a given user based on their command history.

4. Incorporated Ramaswamy NPL Provides Additional Details

79. Ramaswamy expressly incorporates by reference a 1999 Ramaswamy publication (EX1014, “Ramaswamy NPL”), and specifically incorporates Ramaswamy NPL’s natural language understanding (NLU) engine and dialog manager teachings. EX1009, 4:52-55 (“In one embodiment NLU engine [] and dialog manager [] are provided as described in [Ramaswamy NPL].”).

80. In my opinion, one of ordinary skill would have considered the Ramaswamy and Ramaswamy NPL references to describe the same or similar system architecture at least because they describe similar natural language processing systems that perform the same or similar functions. They also include analogous modules that perform similar functions/roles. For example, one of ordinary skill would have considered Ramaswamy NPL's speech recognition and Application Abstraction modules to be analogous to the recognition engine and command executor modules in Ramaswamy. They have similar names, the same or similar functionality, and/or are in the same or similar position in the system diagrams, indicating they have analogous roles in the system architecture.

81. For example, Ramaswamy's "recognition engine" (*e.g.*, EX1009, 4:19) and Ramaswamy NPL's "speech recognition component" (*e.g.*, EX1014, 1) have similar names and have similar roles/functionalities—both convert the user's speech to a string of text (*e.g.*, a string of words). EX1009, 4:17-21 ("Inputs ... such as spoken ... inputs, are submitted to a recognition engine [], which converts the input to text form."); EX10014, 1 ("The speech recognition component converts the user's utterance to a string of text").

82. Similarly, Ramaswamy's command executor and Ramaswamy NPL's Application Abstraction are similarly situated in the system (both interface with the dialog manager) and they perform the same role/function—they execute commands

received from the dialog manager, which processes user input from the NLU engine. EX1009, 4:24-25, 4:32-35 (“The dialog manager ... submits the formal command to the command executor [] for execution of the command.”); EX10014, 2 (The Application Abstraction includes the “information on how a specific command is to be accomplished” and “forward[s] commands to the appropriate application and accumulate[es] the responses.”).

83. Thus, Ramaswamy includes Ramaswamy NPL’s speech recognition, NLU engine, dialog manager, and Application Abstraction teachings.

84. Ramaswamy NPL provides additional details on at least the dialog manager and on how commands submitted by the dialog manager are executed.

85. In addition to what Ramaswamy discloses about the dialog manager (e.g., that it processes commands for execution, resolves ambiguities in the commands, and submits the formal commands to the command executor) (EX1009, 4:29-35), Ramaswamy NPL also discloses that the “dialog management component ... decides on the appropriate action to be taken and the response to be generated.” EX1014, 1. The “Dialog Manager consists of a Mediator ... [that] chooses the appropriate decision network to spawn” *Id.*, 1. This “decision network” is a “recipe for accomplishing a specific transaction” (*id.*, 1), such as a recipe for “[p]ut[ting] all the messages from Jeff about [a] proposal into a folder called proposal”) (*id.*, 3).

86. Though Ramaswamy itself provides few details on the command executor (just that it executes commands), Ramaswamy NPL provides more details on how commands are executed in its Application Abstraction. EX1014, 1-2. In my opinion, one of ordinary skill would have understood that Ramaswamy NPL's Application Abstraction serves a role similar to Ramaswamy's command executor because each interfaces with the dialog manager and performs similar functions related to carrying out commands provided by the dialog manager, as reflected in Fig. 1 of each reference.

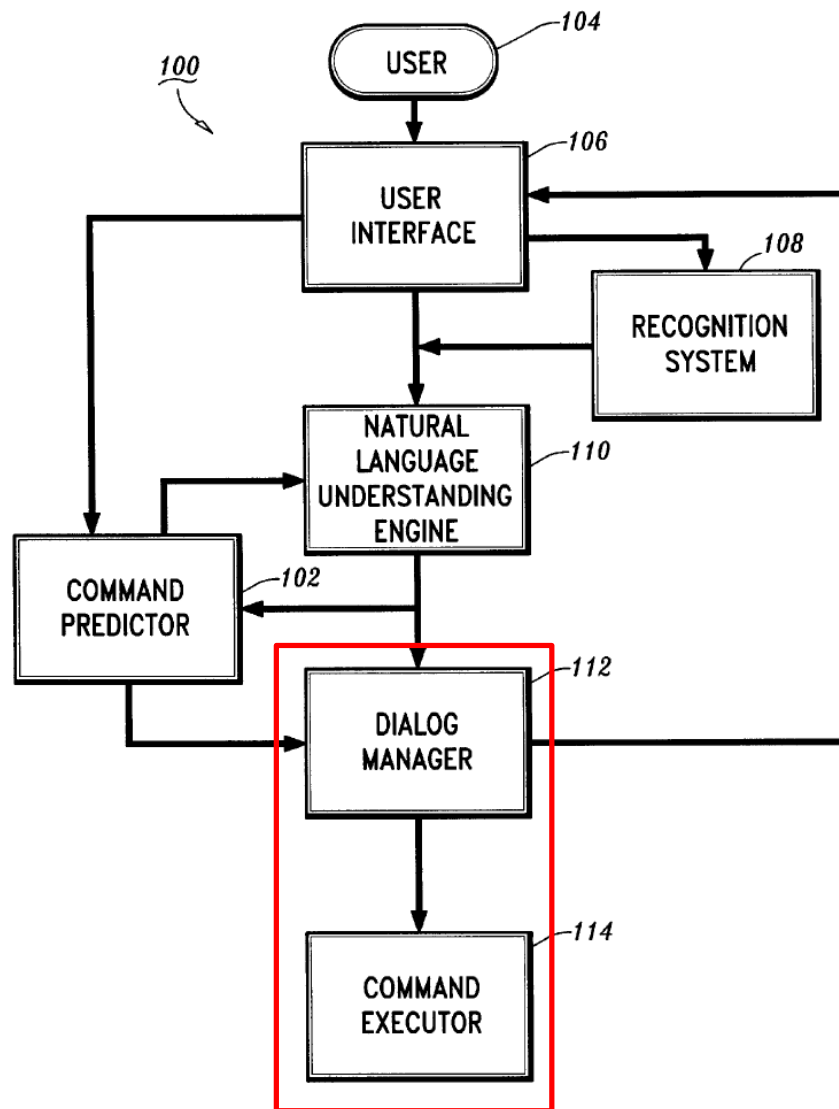


FIG. 1

EX1009, FIG. 1 (red annotation added).

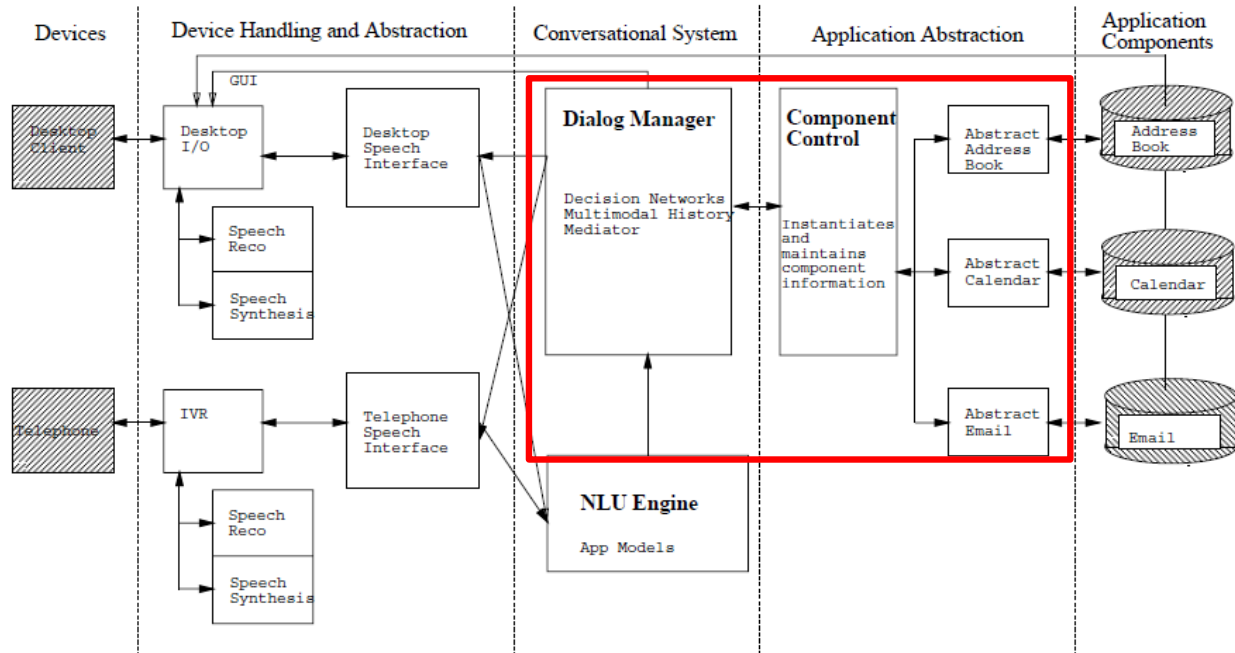


Figure 1: Architecture of the multi-client, multi-application, multi-modal system

EX1014, 2 (red annotation added); *see also* EX1009, 4:24-25, 4:34-35 (a “command executor [] for execution of the command”); EX1014, 1, 2 (The Application Abstraction includes the “information on how a specific command is to be accomplished” and “forward[s] commands to the appropriate application and accumulate[es] the responses.”).

87. Ramaswamy NPL’s Application Abstraction “consists of two pieces: a Component Control and individual abstraction components for each application.” EX1014, 2. The Component Control interfaces directly with the dialog manager and “functions as a switchyard for forwarding commands to the appropriate application and accumulating the responses.” *Id.*, 2. The abstraction components include the “information on how a specific command is to be accomplished,” thus “shield[ing]

the Dialog Manager from the details of the applications [and] allowing a very high level of communication.” *Id.*, 2. In this way, “[t]he applications may therefore be interchanged with no changes necessary to the Dialog Manager; only the abstraction component needs to be changed.” *Id.*, 2.

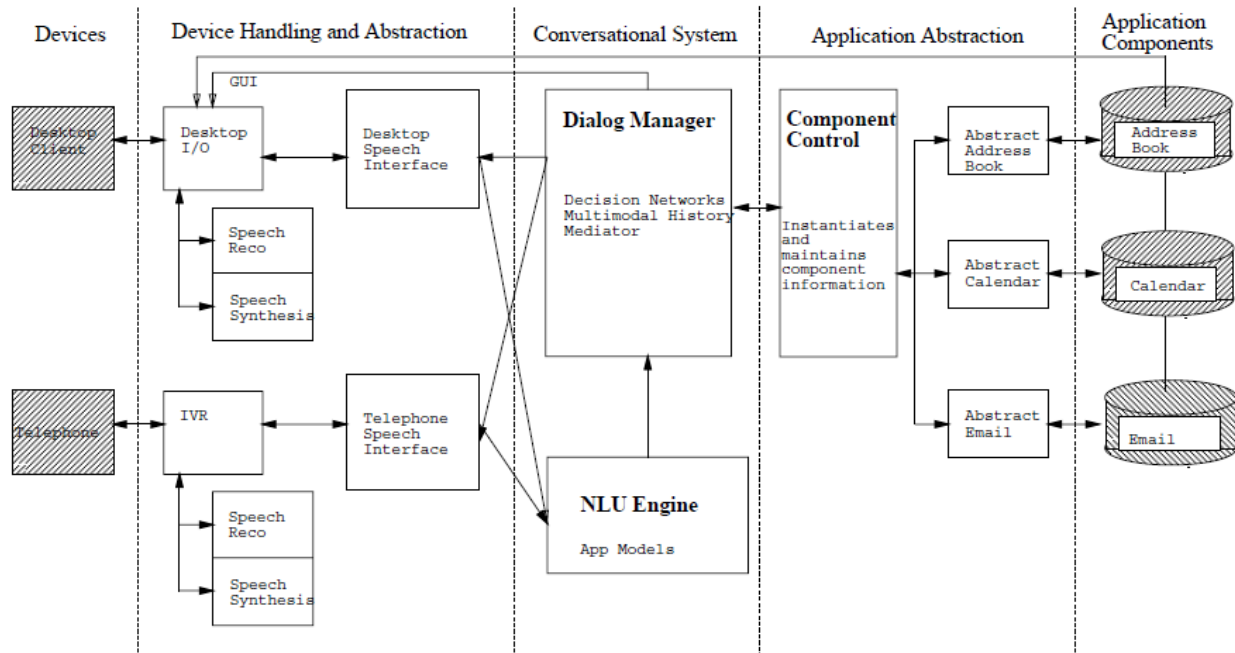


Figure 1: Architecture of the multi-client, multi-application, multi-modal system

EX1014, 2.

B. Claim 42 Mapping

88. As set forth below, Ramaswamy renders obvious claim 42.

1. **Element [42.1]**

[42.1] A method of processing natural language utterances, the method being implemented by a computer system that includes one or more processors executing one or more computer program instructions which, when executed, perform the method, the method comprising:

89. If the preamble is limiting, Ramaswamy teaches it at least because Ramaswamy teaches a “method” (*e.g.*, EX1009, Title, Abstract, 3:14, 10:47-63 claim 12) of processing natural language utterances including “natural language dialog,” “speech,” and “spoken input” (*e.g.*, *id.*, 4:9, 4:13, 4:64). Ramaswamy teaches the method being implemented by a computer system (*e.g.*, “general purpose digital computer[,]” *id.*, 4:5) that includes a “processor” (*e.g.*, *id.*, 4:6) executing one or more computer program instructions (*e.g.*, *id.*, 2:51-52 (“a program of instructions executable by [a] machine”), 4:3-5 (“software”)) which, when executed, perform the “method.”

90. For example, Ramaswamy discloses that “[t]he methods set forth may be implemented by employing a program storage device readable by machine, tangibly embodying a program of instructions executable by the machine to perform the method steps for predicting a next command in a natural language understanding system.” *Id.*, 2:49-44, *see also id.*, claim 17. Further, Ramaswamy discloses that the elements shown in its figures “are preferably implemented in software on one or

more appropriately programmed general purpose digital computers having a processor and memory and input-/output interfaces.” *Id.*, 4:3-7. *See also id.*, 4:1-2.

91. Further, Ramaswamy teaches processing natural language utterances at least because it teaches steps of: **(1)** recognizing speech and “convert[ing] the [speech] input to text form” (*id.*, 4:19; *see also id.*, 4:17-21), **(2)** “convert[ing] the input text to a formal command, suitable for execution by ... a computer” using a “natural language understanding (NLU) engine” (*id.*, 4:22-25), **(3)** “process[ing] [the command] for execution,” including “resolv[ing] any ambiguities that may be present in the formal command, and submit[ting] the formal command to the command executor [] for execution of the command” (*id.*, 4:30-35), and **(4)** “predicting the users next command” (*e.g.*, *id.*, 4:31-32). Ramaswamy’s methods are “used to improve the accuracy of the natural language understanding process” (*id.*, 3:40-41) by capturing “regularities exhibited by the user ... [in] a statistical model” (*id.*, 3:39-40) and “personaliz[ing] the prediction model” as additional user data is obtained to “predict commands with even greater accuracy for [a] particular user” (*id.*, 8:20-24). *See also, e.g., id.*, 1:55-65, 2:9-22, 2:33-54, 3:13-25, 5:9-9:31, claims 15-16, 12:17-27, claims 20-21.

2. Element [42.2]

[42.2] receiving a first input of a user that comprises a natural language utterance;

92. Ramaswamy teaches this claim element at least because Ramaswamy's recognition engine receives a first input of a user ("user input," *e.g.*, EX1009, 4:12) that comprises a natural language utterance ("spoken input," *e.g.*, *id.*, 4:13-14). *See also id.*, 3:26-38, 4:11-20. For example, Ramaswamy's recognition engine receives spoken input from the user interface and recognizes one or more words in the spoken utterance. EX1004, 4:11-21 ("A user [] submits a command input through a user interface []. The user input may be in one of many different modalities, such as spoken input, handwritten input, gestures, typed text, other keyboard and mouse actions. ... Inputs that need translation to text from, such as spoken or handwritten inputs, are submitted to a recognition engine [], which converts the input to text form."); *see also id.* FIG. 1.

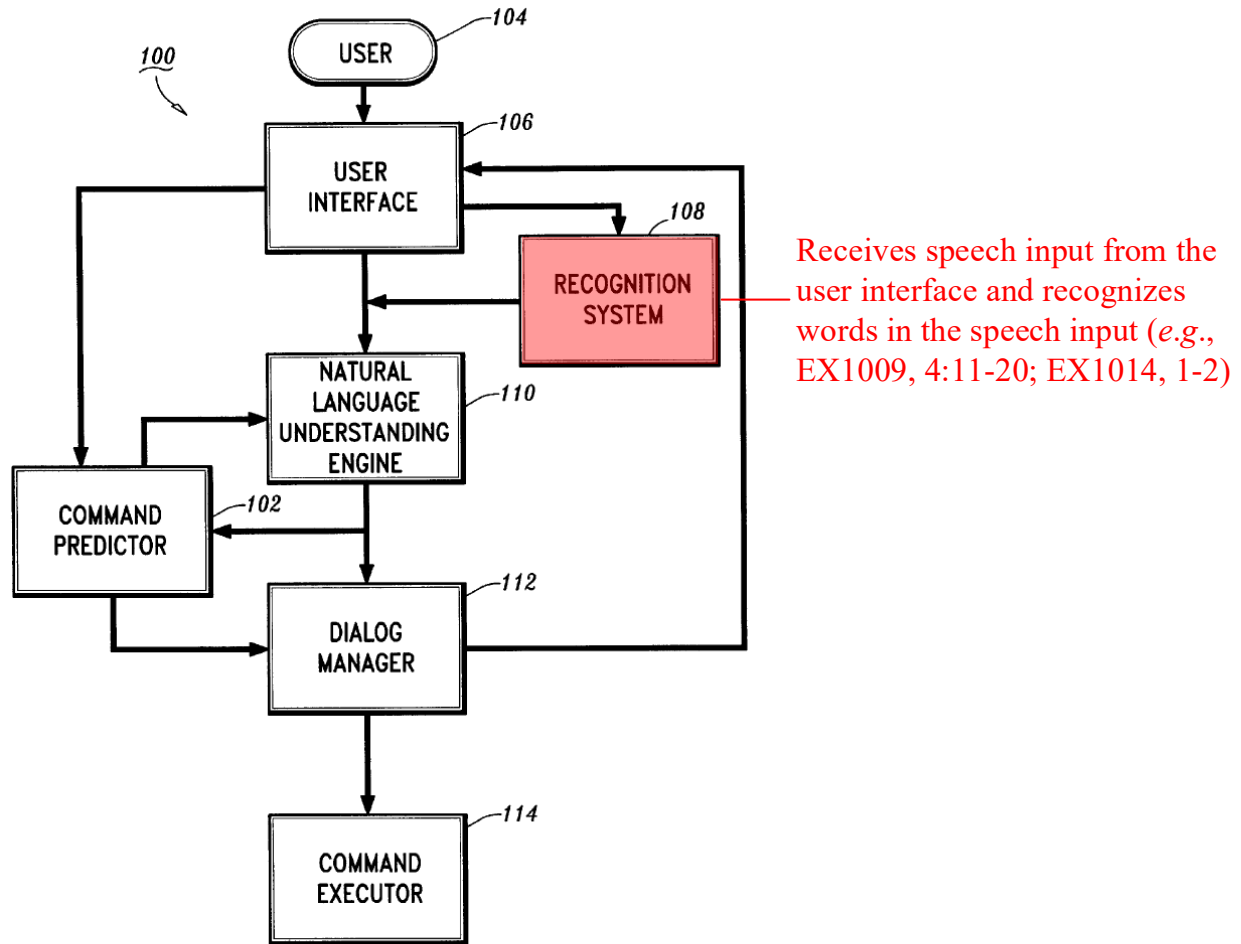


FIG. 1

EX1009, FIG. 1 (red annotations added).

3. Element [42.3]

[42.3] generating an interpretation of the natural language utterance based on one or more recognized words of the natural language utterance;

93. Ramaswamy teaches this claim element at least because Ramaswamy's natural language understanding (NLU) engine generates an interpretation of received natural language utterances (e.g., "spoken input," e.g., EX1009, 4:13-14)

based on words in the spoken input that are recognized by its recognition engine. *Id.*, 3:38-50, 4:17-32, 4:51-55, FIG. 1; EX1014, 1 (“[T]he NLU component extracts the meaning conveyed by the recognized text.”). Ramaswamy explains that its “recognition engine [] ... converts [spoken] input to text form” and that this text “is then submitted to the natural language understanding (NLU) engine [], which converts the input text to a formal command.” EX1009, 4:19-25.

94. One of ordinary skill in the art would have understood or at least found it obvious that the speech recognition engine’s conversion of speech to text involves recognizing words from a speech input at least because this is what then-existing speech recognition systems routinely did. EX1009, 4:20-21 (“Methods for speech recognition ... are known in the art.”); *id.*, 3:55-59 (describing system recognizing a user saying “yes” and explaining that words like “yes” or “no” are not only “easier to say” but also “easier for the system to understand”), 4:26-27 (describing recognition of “do I have any new messages?” from user input); EX1021, 275 (describing how conventional speech recognition systems in 1989 recognized words); EX1022, 254 (describing a 1975 approach for recognizing word sequences in speech); EX1023, 52 (describing a 1986 speaker-independent approach for recognizing words in speech).

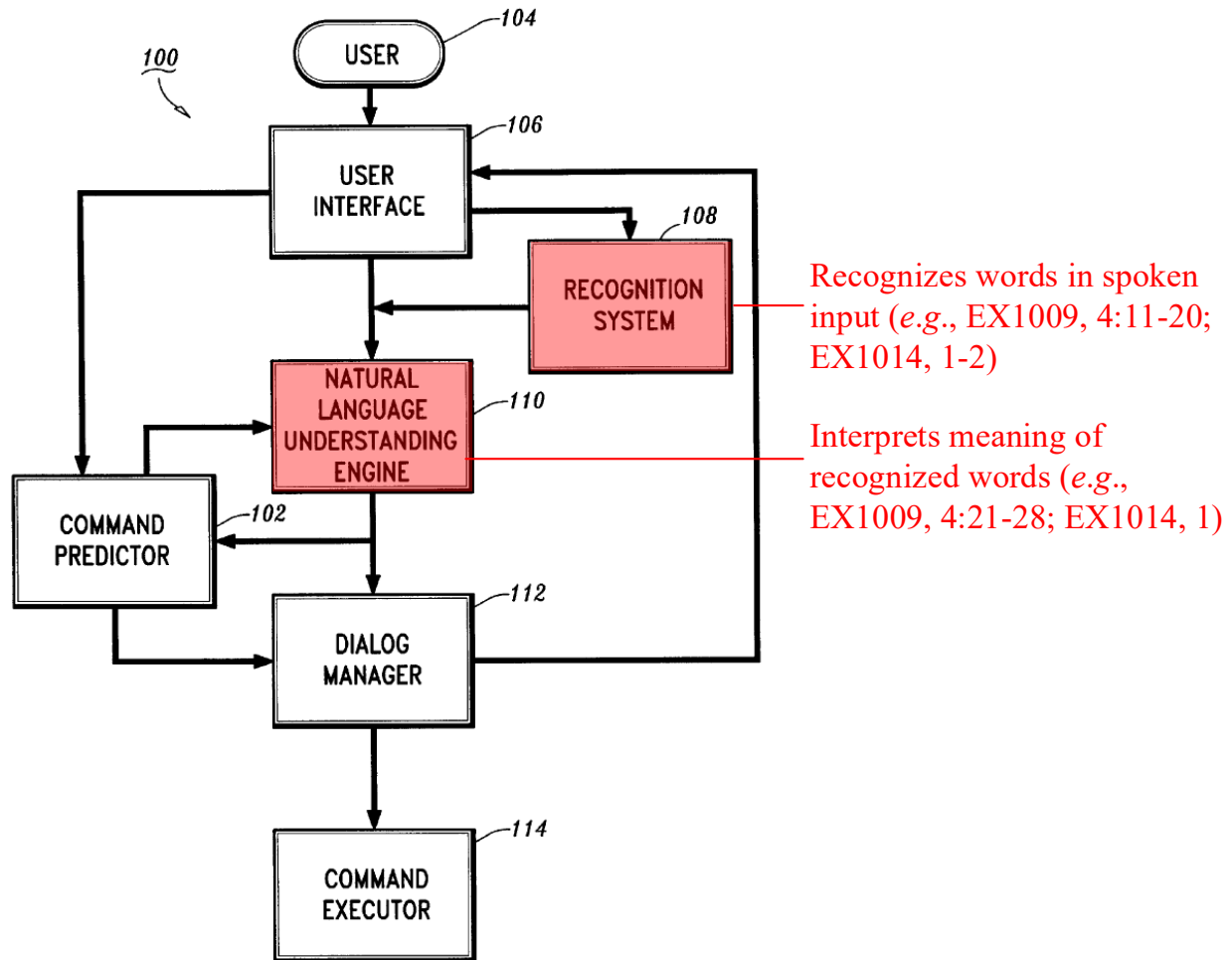


FIG. 1

EX1009, FIG. 1 (red annotations added).

95. Further, Ramaswamy's natural language understanding engine generates an interpretation of the natural language utterance based on the recognized words, hence its name, an "understanding engine." *E.g.*, EX1009, 3:40-50, 4:21-29. For example, Ramaswamy's NLU engine interprets the recognized words "do I have any new messages?" as a request to check new email messages and issues the formal command "checkNewMail()" to the dialog manager. *Id.*, 4:26-28 ("In the example

of an electronic mail application described above, if the user input is ‘do I have any new messages?’ the corresponding formal command may be: checkNewMail().”). Indeed, one of Ramaswamy’s stated purposes is to “improve the accuracy of the natural language understanding process used to interpret the user input.” *Id.*, 3:40-42; *see also id.*, 3:19-20, 3:44-50, 6:25:29.

96. Thus, Ramaswamy interprets the natural language utterance based on one or more recognized words of the natural language utterance.

4. Element [42.4]

[42.4] generating a request based on the interpretation;

97. Ramaswamy teaches this claim element at least because Ramaswamy’s NLU engine and/or its dialog manager generate a request (“command”) based on the interpretation of the natural language utterance. *E.g.*, EX1009, 4:21-35; EX1014, 1-2.

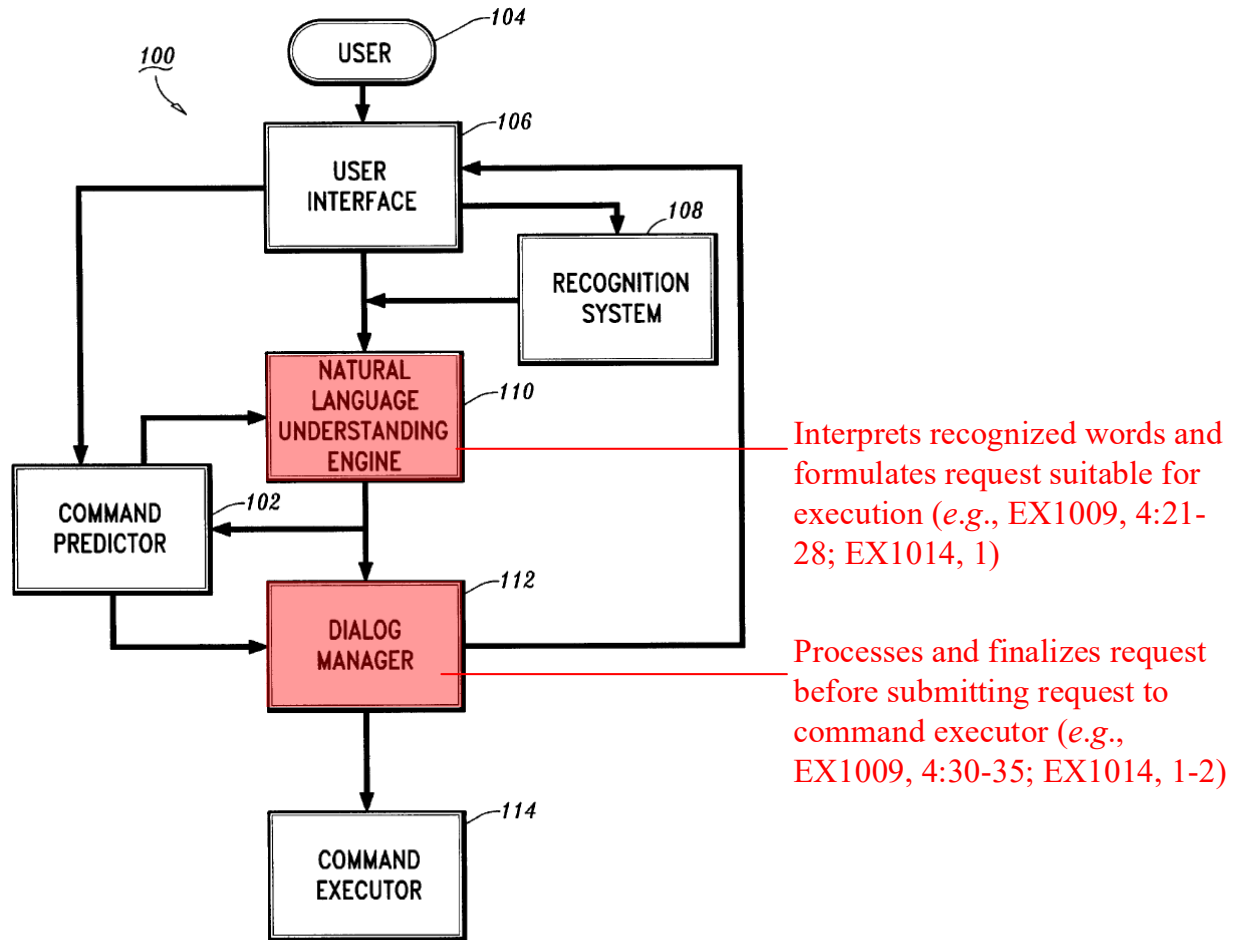


FIG. 1

EX1009, FIG. 1 (red annotations added).

98. Specifically, “the natural language understanding (NLU) engine ... converts the input text to a formal command, suitable for execution by a command executor [], for example, a computer.” EX1009, 4:22-25. For example, “if the user input is ‘do I have any new messages?’ the corresponding formal command may be: checkNewMail(). The formal command from NLU engine [] is submitted to a dialog manager [], to be processed for execution The dialog manager [] resolves any

ambiguities that may be present in the formal command, and submits the formal command to the command executor [] for execution of the command.” *Id.*, 4:27-35.

99. Thus, Ramaswamy’s NLU engine generates a request (a “command” such as “checkNewMail()”) based on the interpretation of the natural language utterance (*e.g.*, “do I have any new messages?” *id.*, 4:27) and the dialog manager resolves any ambiguities in the command and submits the command to the command executor. *Id.*, 4:21-35. As such, Ramaswamy’s NLU engine and/or its dialog manager generate a request based on the interpretation of the natural language utterance.

5. Element [42.5]

[42.5] transmitting the request to a domain agent for processing;

100. Ramaswamy teaches this claim element at least because Ramaswamy’s NLU engine and/or its dialog manager transmit (“submit”) the request (“command”) to a domain agent (“command executor”) for processing (*e.g.*, finalization and execution). *E.g.*, EX1009, 4:21-35; EX1014, 1-2.

101. The command executor processes the request at least because it executes the request. EX1009, 4:21-25, 4:32-35 (“The dialog manager ... submits the formal command to the command executor [] for execution of the command.”); *see also* EX1014, 2. Though Ramaswamy itself provides few details on the command executor (just that it executes commands), Ramaswamy NPL provides

more details on how commands are executed in its Application Abstraction. *Id.*, 1-2. Specifically, Ramaswamy NPL discloses that its Application Abstraction, which consists of a Component Control and individual abstraction components, executes the command by routing the commands to the appropriate application(s) and collecting responses from the applications. *Id.*, 1-2. The Component Control “functions as a switchyard for forwarding commands to the appropriate application and accumulating the responses” and the abstraction components include the “information on how a specific command is to be accomplished.” *Id.*, 2.

102. As explained above in Section XI.A.4, it is my opinion that one of ordinary skill would have understood that Ramaswamy NPL’s Application Abstraction teachings are a part of Ramaswamy, specifically Ramaswamy’s command executor, at least because Ramaswamy incorporates Ramaswamy NPL by reference (EX1009, 4:51-55) and because Ramaswamy NPL’s Application Abstraction is analogous to the command executor. *Id.*, 4:21-35, FIG. 1; EX1014, 1-2. They have the same or similar functionality (command execution) and are located in the same position in the system (next to the dialog manager). *Id.*

103. Specifically, the command executor “execut[es] [] the command” (EX1009, 4:34-35) and the Application Abstraction includes the “information on how a specific command is to be accomplished” and “forward[s] commands to the appropriate application and accumulate[es] the responses.” (EX1014, 2). *See also*

EX1009, 4:24-25, 4:32-35; EX1014, 1. Both interface with the dialog manager and receive commands from the dialog manager. Compare EX1009, FIG. 1 with EX1014, 2 (Figure 1).

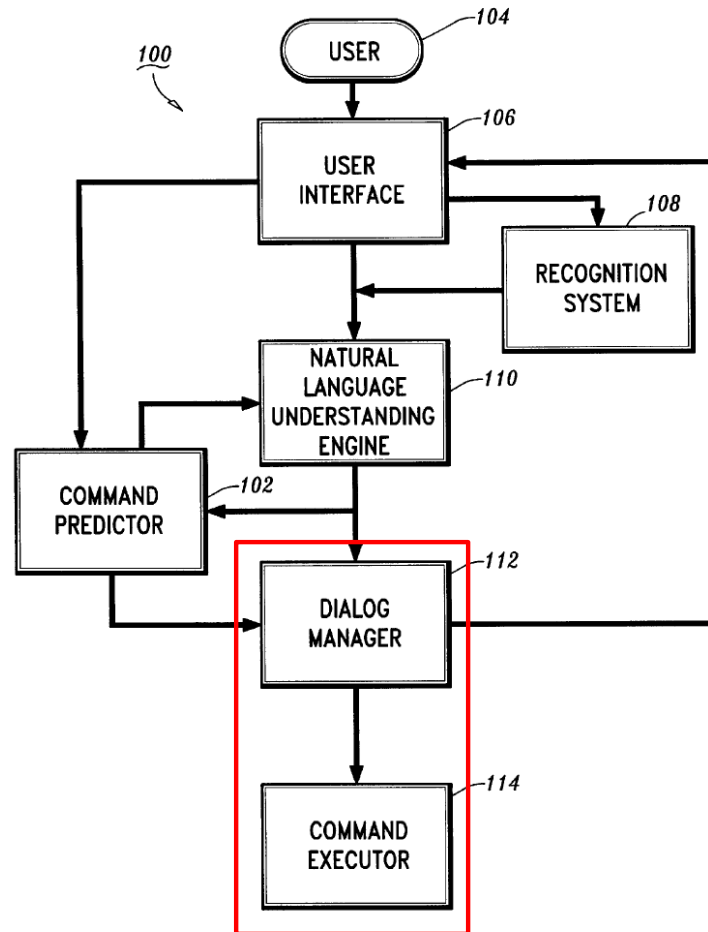


FIG. 1

EX1009, FIG. 1 (red annotation added).

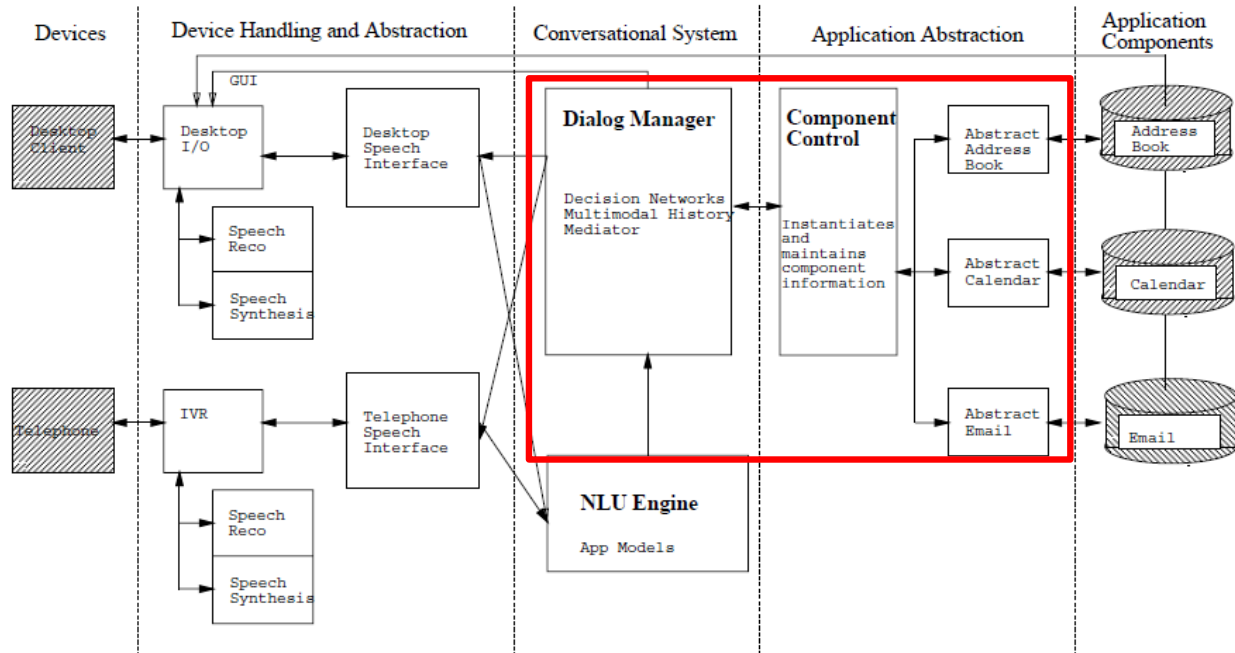


Figure 1: Architecture of the multi-client, multi-application, multi-modal system

EX1014, 2 (red annotation added).

104. Ramaswamy's command executor / Application Abstraction (as a whole, and including the Application Abstraction's component control and abstraction components individually) is a domain agent, including if construed as software with domain-specific behavior and information. *See supra* Section VIII.A. The component control is software with domain-specific behavior and information at least because it interfaces with only a specific set of applications (email, calendar, and address book) and "functions as a switchyard for forwarding commands to the appropriate application and accumulating the responses." EX1014, 2. The abstraction components are software with domain-specific behavior and information at least because they include the information "on how a specific command is to be

accomplished” by a particular application. *Id.*, 2. That is, they act as an intermediary between the component control and each individual application for executing the commands and accumulating the responses. *Id.*, 2. Thus, the Application Abstraction is a domain agent at least because it is a specialized software module that handles specific types of commands involving specific types of applications and specific types of information (i.e., it is tailored to a single application domain).

105. Moreover, it would have been obvious to one of ordinary skill when implementing Ramaswamy’s command executor to include multiple domain agents, each responsible for specific types of information and/or requests, at least because these agents and their benefits were well known in the art, dating back to at least 1994. EX1010, 7:1-3, 7:6-9, 17:8-14; EX1020, 49 (Agents with a “domain of expertise” are “modular, extensible, flexible and efficient. ... Each information agent is specialized to a single ‘application domain’ and provides access to the available information sources within that domain.”), 50-56. Specifically, because domain agents are modular (they contain expertise and information specific to a particular domain), they can be easily added to a system to scale up the system (i.e., they are extensible). *E.g.*, EX1010, 7:6-9 (“[W]ith a modular distributed agent architecture, the system and method of the present invention is extendable to multiple applications and is scalable to large sets of networked computer systems.”); EX1020, 54-55. Further, domain agents can execute multiple tasks in parallel,

increasing the efficiency of the command execution process. *E.g., Id.*, 55 (“[T]he generation of parallel access plans, the caching of retrieved data, and the ability to learn about information sources provide efficient access to large amounts of distributed information.”).

106. Thus, persons of ordinary skill in the art would have known that domain agents have a number of advantages: “(1) modularity ... (2) extensibility ... (3) flexibility ... and (4) efficiency” (EX1020, 55-56), and would have found it obvious to implement Ramaswamy’s command executor with at least one domain agent, especially given Ramaswamy NPL’s teachings to include an Application Abstraction that is specialized to a single application domain. EX1014, 2.

107. Thus, Ramaswamy teaches this claim element at least because the request (“command”) generated by the NLU engine and/or the dialog manager is submitted to a domain agent (“command executor”) for processing (e.g., finalizing and executing the command).

108. To the extent Patent Owner argues that Ramaswamy and/or Ramaswamy NPL do not teach a domain agent (claim element [42.5])—which they do—and/or that Ramaswamy does not incorporate Ramaswamy’s Application Abstraction teachings—which it does—the claims would have been obvious to one of ordinary skill over Ramaswamy in combination with Ramaswamy NPL (Ground 3) or Hartono (Ground 4).

6. **Element [42.6]**

[42.6] determining whether a personalized cognitive model associated with the user includes sufficient information for predicting one or more subsequent actions associated with the user, wherein the personalized cognitive model is generated based on a tracking of a pattern of interactions between the user and the system, and wherein the one or more subsequent actions include one or more actions predicted to occur after receiving the first input; and

109. Ramaswamy teaches determining whether a personalized cognitive model (“personal prediction model”) associated with the user includes sufficient information for predicting one or more subsequent actions associated with the user (e.g., “subsequent commands,” “next command”), wherein the personalized cognitive model is generated based on a tracking of a pattern of interactions between the user and the system (e.g., “command history”, “history of interaction between the user and the system,” “personal training corpus”), and wherein the one or more subsequent actions include one or more actions (e.g., “next command”) predicted to occur after receiving the first input (first command). EX1009, 3:27-28, 3:51, 8:29-9:33.

a) Ramaswamy’s personal prediction model is a personalized cognitive model generated based on a tracking of a pattern of interactions between the user and the system.

110. Ramaswamy’s personal prediction model is generated based on tracking a pattern of interactions (e.g., “command history”) between the user and the system. EX1009, 2:9-22, 8:29-46. Specifically, the user’s command history is used

to generate the personal prediction model. *E.g., Id.*, 2:9-22, 8:29-46 (“Every time the user issues a command, the resulting formal command ... [is] written to a personal training corpus []. ... [W]hen there is sufficient data to construct a new model, the data is presented to a personal corpus processor []. The personal corpus processor ... creates a personal retained corpus [] and a personal held-out corpus []. The personal retained corpus [] and held-out corpus [] are used by a personal model constructor ... to construct a personal prediction model [.]”), 10:42-46, (claim 11) (“the prediction model includes a personal prediction model for a given user, the personal prediction model being constructed from a personal training corpus and based on a command history”).

111. Ramaswamy’s personal prediction model is a personalized cognitive model at least because it is a model derived from a user’s interaction pattern with the system and is used to predict what action(s) the user will take next. *Id.*, 1:23-28, 3:13-50, 8:19-9:30, 10:30-46, (claims 9-11). Specifically, Ramaswamy discloses storing the user’s interaction patterns (command history) with the system (*e.g., id.*, 8:28-46) and providing a personal prediction model that is used to calculate the conditional probability of a given next command according to equation 7 (reproduced below):

$$P^p(F_t|F_{t-1}, F_{t-2}, M_t) = \lambda_3^p g^p(F_t|F_{t-1}, F_{t-2}, M_t) + \lambda_2^p g^p(F_t|F_{t-1}, M_t) + \lambda_1^p g^p(F_t|M_t) \quad (7)$$

where F_{t-n} is a user's prior command n commands before the current command (t) (e.g., *id.*, 5:17-22) and P^p is the conditional probability of a given action for that user (e.g., *id.*, 5:13-22, 8:8-12). *Id.*, 8:50-51 (equation 7). Thus, this personal prediction model is a "statistical model" that captures "the regularities exhibited by the user," and is used to predict the user's next command, thus "improv[ing] the accuracy of the natural language understanding process used to interpret the user input." *Id.*, 3:29-42.

112. Ramaswamy's personal prediction model is specific to a given user's interaction pattern with the system. *Id.*, 8:19-9:30. "[T]he personal prediction model [] is used to calculate the conditional probability $P^p()$ given the history ... for a **particular** user." *Id.*, 8:46-49 (emphasis added); *see also id.*, 2:19-22, 2:45-49 ("constructing the personal prediction model for a given user based on previously issued commands") 8:19-24, 9:17-30, 10:42-46 (claim 11). "As the user continues to use the system, the personal training corpus will become larger ... and the prediction will become more personalized to the given user" *Id.*, 9:14-19; *see also id.*, 2:10-20, 8:19-24 ("[A]s a given user continues to use the system, additional data is generated, and additional data can be used to personalize the prediction model [], such that the system will predict commands with even greater accuracy for that particular user."), 8:25-9:13, 9:20-30, 10:30-46 (claims 9-11).

113. Thus, Ramaswamy's personal prediction model is a personalized cognitive model generated based on a tracking of a pattern of interactions between the user and the system.

b) Ramaswamy teaches determining whether the personalized cognitive model includes sufficient information for predicting a subsequent action associated with the user.

114. "A new user starts using the system" with a generalized prediction model ("prediction model 204"). EX1009, 8:28-29. Once the user has interacted with the system enough, the system builds the personalized cognitive model and starts using it (along with the generalized prediction model) to predict the next user's action, as indicated by equation 9. *Id.*, 8:19-9:24, 8:29-9:30. At first, however, the system gives the personalized cognitive model little to no weight because it is still not accurate enough to be meaningfully relied on. *Id.*, 9:9-13. As the user continues to use the system, however, the personalized cognitive model accumulates more information, it becomes more accurate, and the system increases the weight given to its probability estimates. *Id.*, 9:13-30.

115. Specifically, to build the personalized cognitive model, the system records the user's commands as the user uses the system and accumulates the command history data in a personal training corpus. *Id.*, 8:29-34. "[W]hen there is *sufficient* data to construct a new model," (emphasis added) the system "construct[s] a personal prediction model." *Id.*, 8:34-35, 8:46; *see also id.*, 8:36-45. The personal

prediction model is then used by the probability calculator to predict the next command of the user after receiving a first command from the user. *E.g., Id.*, 1:66-2:5 (“A command prediction system includes a natural language understanding engine coupled to a user interface for receiving commands from a user and converting the commands to formal language commands. A command predictor is included for receiving the formal language commands from the natural language engine and for predicting at least one next command which is likely to be presented by the user.”) 2:10-20 (“The command predictor may include a prediction model ... used to predict the at least one next command. ... The prediction model may include a personal prediction model for a given user.”), 4:11-41 (“A user [] submits a command input through a user interface []. ... The command predictor [] uses [] the knowledge of previously issued commands ... to predict the users next command.”), 4:59-9:30, 10:29-33 (claim 9) (“the command predictor includes a prediction model ... used to predict the at least one next command”), 10:34-41 (claim 10), 10:42-46 (claim 11) (“the prediction model includes a personal prediction model for a given user, the personal prediction model being constructed from a personal training corpus [] based on a command history of the given user”).

116. Once constructed, the “personal prediction model [] is used to calculate the conditional probability” of a given command, according to equation 7 (described above). *Id.*, 8:46-52. This probability and the probability given by the generalized

cognitive model (equation 6) are each assigned weights (coefficients β_p and β_a , respectively) between zero and one based on how much information the personalized model includes, and these probabilities are combined to determine the probability of the user's next command. *Id.*, 8:46-9:30. This is shown in equation 9:

$$P(F_t|F_{t-1}, F_{t-2}, M_t) = \beta_a P^a(F_t|F_{t-1}, F_{t-2}, M_t) + \beta_p P^p(F_t|F_{t-1}, F_{t-2}, M_t) \quad (9)$$

where P^a is the average conditional probability calculated from the general prediction model (equation 6) (*id.*, 8:58-9:1), P^p is the personal conditional probability calculated from the personal prediction model (equation 7) (*id.*, 9:1-2), and β_a and β_p are the coefficients given to the personal and general prediction models' probably estimates (*id.*, 9:8-22). *Id.*, 9:6-7 (equation 9); *see also id.*, 8:46-9:30.

117. When the personalized cognitive model is first constructed, however, Ramaswamy gives it little to no weight by setting its coefficient β_p to a very small value (e.g., 0.01). *Id.*, 9:9-14 ("Initially, when there is only a small amount of data in the personal training corpus [], the personal prediction model [] will not be as reliable as when there is a large amount of data, and therefore β_p should be set to a small value, for example 0.01."). Thus, at least when the personalized cognitive model is first constructed, Ramaswamy effectively only uses only the generalized cognitive model to predict subsequent user actions by weighting it very heavily (e.g., giving it a coefficient of 0.99). *Id.*, 9:8-13 (explaining that the coefficients β_p and β_a

add together to equal one and that initially, the personalized cognitive model's coefficient, β_p , is set to 0.01.).

118. Ramaswamy teaches determining whether the personalized cognitive model includes sufficient information for several reasons.

119. **First**, Ramaswamy assigns the model little to no weight (e.g., a coefficient of 0.01) when it first constructs the model but assigns the model increasingly greater weight as it gains more information, indicating to one of ordinary skill that Ramaswamy's system determines that the model includes insufficient information when it is first constructed but includes sufficient information after a certain period of use. *Id.*, 9:9-22 (explaining that Ramaswamy assigns the model a weight between zero and one based on the sufficiency of the model's information). That is, one of ordinary skill would have understood or found it obvious that Ramaswamy's system determines that the information is insufficient when it gives the model little to no weight and determines that the information is sufficient when it gives the model a greater (e.g., non-insignificant) weight. Thus, Ramaswamy teaches determining whether the model's information is sufficient at least because Ramaswamy assigns the model a weight based on how sufficient its information is.

120. **Second**, to the extent the claim requires making a binary determination (sufficient or insufficient) based on whether the model's information is above or

below a specific threshold, such a binary approach would have been an obvious design choice. Specifically, instead of determining how sufficient the information is and then assigning the personalized model a corresponding weight between 0 and 1 (as expressly disclosed by Ramaswamy), it would have been an obvious design choice to instead determine whether the information is above or below a certain threshold and assign the model a corresponding weight based on that determination (e.g., a 0 if the information is insufficient (below the threshold), or a 1 if the information is sufficient (above the threshold)). This was one of a limited number of known ways of implementing Ramaswamy's teaching to give the personalized cognitive model little to no weight when it has insufficient information and to give it a greater weight when it has sufficient information. *E.g.*, EX1024, FIG. 4A (showing giving no weight (a 0) to a user-specific model when there is insufficient information); EX1025, 398 (describing using a general model ("stereotype") when the information in a user-specific model is insufficient). Thus, it would have been obvious to implement Ramaswamy's sliding scale approach (assign a weight between 0 and 1 based on how sufficient the information is) as a binary approach (i.e., using the generalized model alone when the assigned weight of the personalized model would have been zero).

121. ***Third***, one of ordinary skill would have found it obvious to implement Ramaswamy's system such that it determines that the personalized cognitive model

includes insufficient information when the user's command falls outside the domain of the model's training data (command history). For example, if the personalized cognitive model was trained on only email-specific commands (e.g., "send email to John") and the user's command relates to a navigational command (e.g., "get me directions to the nearest restaurant"), one of ordinary skill would have found it obvious that the personalized cognitive model includes insufficient information and would rely on only the generalized cognitive model because the personalized cognitive model has not been trained on this kind of command and thus is likely to be less reliable/accurate. EX1025, 398; EX1026, 319, 321-322.

122. ***Fourth***, Ramaswamy's express disclosure to make a data sufficiency determination renders the claims' sufficiency determination obvious. To the extent Patent Owner argues that claim 42 requires the sufficiency determination be made after the personalized cognitive model is constructed (as opposed to Ramaswamy's pre-construction determination), a skilled artisan would have found this to be an obvious design choice variation of Ramaswamy's approach. One of ordinary skill in the art was aware of the importance of not using a user model until such model contained sufficient information to provide an accurate prediction and a skilled artisan was aware that there were two ways in which to prevent use of such a model until sufficient information to make a prediction was available. That is, either a system collects sufficient data and then builds the user model once it is determined

that sufficient data is available (as disclosed in Ramaswamy, EX1009, 8:33-36), or the system constructs the user model and begins using the model once sufficient is received (*see e.g.*, EX1025, 398 (describing using a general model when the information in a user-specific model is insufficient)). One of ordinary skill in the art would have understood the latter approach to be fully consistent and easily implemented using Ramaswamy's conditional probability calculation, discussed above, so as to give no weight to the constructed personal prediction model until it contains sufficient information to make a prediction. This simple design choice would have been obvious to a skilled artisan.

7. **Element [42.7]**

[42.7] predicting the one or more subsequent actions based on a generalized cognitive model in response to a determination that the personalized cognitive model does not include the sufficient information, wherein the generalized cognitive model is generated based on a tracking of patterns of interactions between a plurality of users and the system.

123. Ramaswamy teaches predicting the one or more subsequent actions (“predicting [a / the] at least one next command”) (*e.g.*, EX1009, 1:59, 2:4-5, 10:22) based on a generalized cognitive model (*e.g.*, “prediction model 204”) (*id.*, *passim*) in response to a determination that the personalized cognitive model does not include the sufficient information, wherein the generalized cognitive model is generated based on a tracking of patterns of interactions between a plurality of users and the

system (e.g., the “command history” (*id.*, *passim*) of a “large number of users” (e.g., *id.*, 6:56, 8:18)). *Id.*, 4:59-9:22.

- a) **Ramaswamy teaches a generalized cognitive model that is generated based on a tracking of patterns of interactions between a plurality of users and the system.**

124. Ramaswamy’s prediction model 204 is a generalized cognitive model at least because it is a statistical abstract that corresponds to interaction patterns with the system for multiple users, and it is generated based on tracking patterns of interactions between a plurality of users and the system. EX1009, 5:9-8:19, FIGS. 2-3. Specifically, Ramaswamy teaches “building the prediction model 204” from a “training corpus 302,” which “includes a large number of formal commands from a large number of users.” *Id.*, 6:31-32, 6:54-56; *see also id.*, 5:19-8:19, FIGS. 2-3. The prediction model 204 is a statistical abstract of the interaction history (command history) of a plurality of users and is used to calculate a probability of a user’s next command according to equation 6 (reproduced below):

$$P(F_t|F_{t-1}, F_{t-2}, M_t) = \lambda_3 g(F_t|F_{t-1}, F_{t-2}, M_t) + \lambda_2 g(F_t|F_{t-1}, M_t) + \lambda_1 g(F_t|M_t) \quad (6)$$

where F is a prior command (e.g., *id.*, 5:17-22), g() is the relative frequency function (*id.*, 7:6), and P is the conditional probability of a given action for the average user (e.g., *id.*, 5:13-22, 8:16-19). *Id.*, 7:21-22 (equation 6); *see also id.*, 5:9-8:19. “Therefore, building the prediction model may be done simply by counting the relative frequencies of all observed sequences of command in the training corpus

302” (*id.*, 7:8-10) and applying weights (the lambdas (λ) in equation 6) to smooth the probability distribution (*id.*, 7:11-8:2). Because the “prediction model 204 ... is constructed using the training corpus 302 from a large number of users,” it “therefore will work well for the average user.” *Id.*, 8:16-19.

125. Thus, Ramaswamy teaches a generalized cognitive model that is generated based on a tracking of patterns of interactions between a plurality of users and the system.

b) Ramaswamy teaches predicting the one or more subsequent actions based on a generalized cognitive model in response to a determination that the personalized cognitive model does not include the sufficient information.

126. Ramaswamy teaches predicting the one or more subsequent actions based on the generalized cognitive model (“prediction model 204”) in response to a determination that the personalized cognitive model (e.g., “personal prediction model 404”) does not include the sufficient information at least because Ramaswamy always uses the generalized cognitive model to predict subsequent actions—both before and after construction of the personalized cognitive model. EX1009, 8:16-9:30, FIGS. 2, 4.

127. For example, as explained above for element [42.6] (*supra* Section XI.B.6), before the personalized model is constructed, “[a] new user starts using the system [] with the prediction model 204.” Thus, when Ramaswamy’s system

determines that there is insufficient information for the personalized model (such that the personalized model should not be constructed and used yet), Ramaswamy's system predicts the one or more subsequent user action based on only the generalized cognitive model. *E.g., Id.*, 8:28-29.

128. Only when Ramaswamy determines that there is "sufficient data" (*id.*, 8:34-35), does Ramaswamy teach also using the personalized cognitive model. *Id.*, 8:29-9:30. And even in this case (when using the personalized cognitive model), Ramaswamy still teaches using the generalized cognitive model. *Id.*, 8:58-9:2, 9:3-4 ("The average and personal conditional probabilities are then mixed to create the resulting final conditional probability."), 9:5-19, FIG. 4.

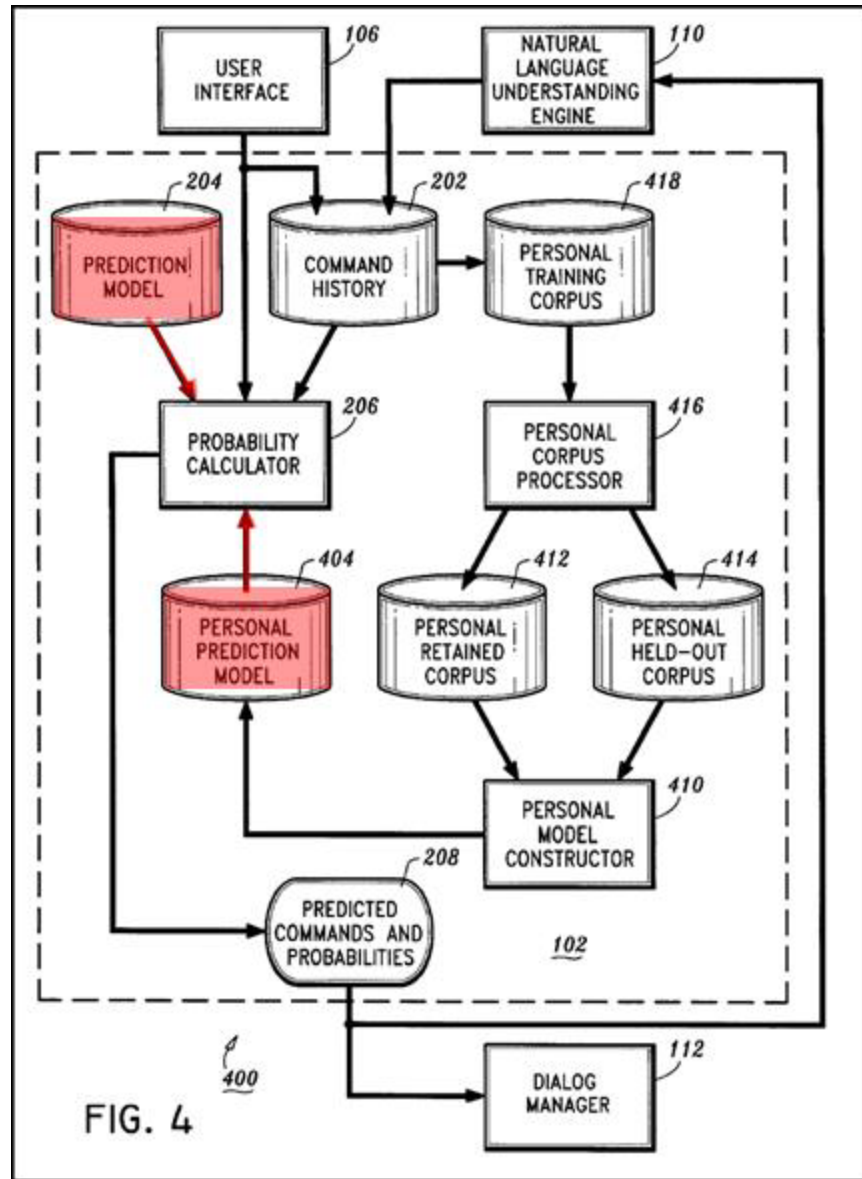


FIG. 4

EX1009, FIG. 4 (red annotations added). Specifically, Ramaswamy teaches using both the generalized cognitive model and the personal prediction model in equation 9:

$$P(F_t|F_{t-1}, F_{t-2}, M_t) = \beta_a^{Pa}(F_t|F_{t-1}, F_{t-2}, M_t) + \beta_p^{Pp}(F_t|F_{t-1}, F_{t-2}, M_t) \quad (9)$$

where P^a is the average conditional probability calculated from the general prediction model (equation 6) (EX1009, 8:58-9:1), P^p is the personal conditional

probability calculated from the personal prediction model (equation 7) (*id.*, 9:1-2), and β_a and β_p are coefficients that determine the weight given to the personal prediction model and the general prediction model (*id.*, 9:8-22). *Id.*, 9:6-7 (equation 9).

129. And, as explained above for element [42.6] (*supra* Section XI.B.6), even when employing both the general and personal prediction models, Ramaswamy teaches effectively using only the generalized model if the personalized cognitive model contains a small amount of data. *Id.*, 9:9-22. Ramaswamy does this by assigning the personalized cognitive model little to no weight. *Id.*, 9:9-13 (“when there is only a small amount of data in the personal training corpus [], the personal prediction model [] will not be as reliable as when there is a large amount of data, and therefore” the personalized prediction model’s coefficient “ β_p should be set to a small value, for example 0.01.”). Thus, Ramaswamy teaches predicting one or more subsequent user actions based on the generalized model responsive to a determination that the personalized model includes insufficient information because Ramaswamy assigns the generalized model a large coefficient (e.g., 0.99) responsive to a determination that the personalized model includes insufficient information. *Id.*, 9:9-13.

130. As also explained above for element [42.6], it would have been obvious to one of ordinary skill to employ a binary approach whereby the system uses only

the generalized cognitive model—not the personalized cognitive model—when the system determines that the personalized cognitive model includes insufficient information. *Supra* Section XI.B.6.

**XII. GROUND 3: CLAIM 42 IS OBVIOUS
OVER RAMASWAMY IN VIEW OF RAMASWAMY NPL (EX1014)**

131. To the extent Patent Owner argues that Ramaswamy alone does not teach element [42.5] (the domain agent limitation) and that Ramaswamy NPL’s Application Abstraction teachings are not a part of Ramaswamy (which is wrong), claim 42 is still obvious over Ramaswamy and Ramaswamy NPL because one of ordinary skill would have found it obvious to combine Ramaswamy NPL’s Application Abstraction teachings with Ramaswamy’s system, specifically its command executor. Thus, the Ground 3 combination combines Ramasamy’s system with Ramaswamy NPL’s Application Abstraction teachings, and, more specifically, incorporates Ramaswamy NPL’s Application Abstraction teachings into the command executor of Ramasamy’s system.

132. Ramaswamy NPL is analogous art to the ’659 patent at least because it describes systems and methods for processing natural language utterances. *E.g.*, EX1014, 1-2.

133. One of ordinary skill in the art would have been motivated with a reasonable expectation of success to combine Ramaswamy NPL’s Application Abstraction teachings with Ramaswamy’s command executor teachings at least

because the command executor and Application Abstraction are analogous/equivalent software modules. The command executor and Application Abstraction sit in the same position in Ramaswamy's system and perform the same function (command execution). EX1009, 4:29-35, FIG. 1; EX1014, 1-2.

134. Specifically, just like the command executor in Ramaswamy, the Application Abstraction in Ramaswamy NPL interfaces directly with the dialog manager (*compare* EX1009, FIG. 1, *with* EX1014, 2 (Figure 1)) and executes commands. EX1009, 4:24-25, 4:34-35 (a "command executor [] for execution of the command"); EX1014, 1, 2 (The Application Abstraction includes the "information on how a specific command is to be accomplished" and "forward[s] commands to the appropriate application and accumulate[es] the responses.").

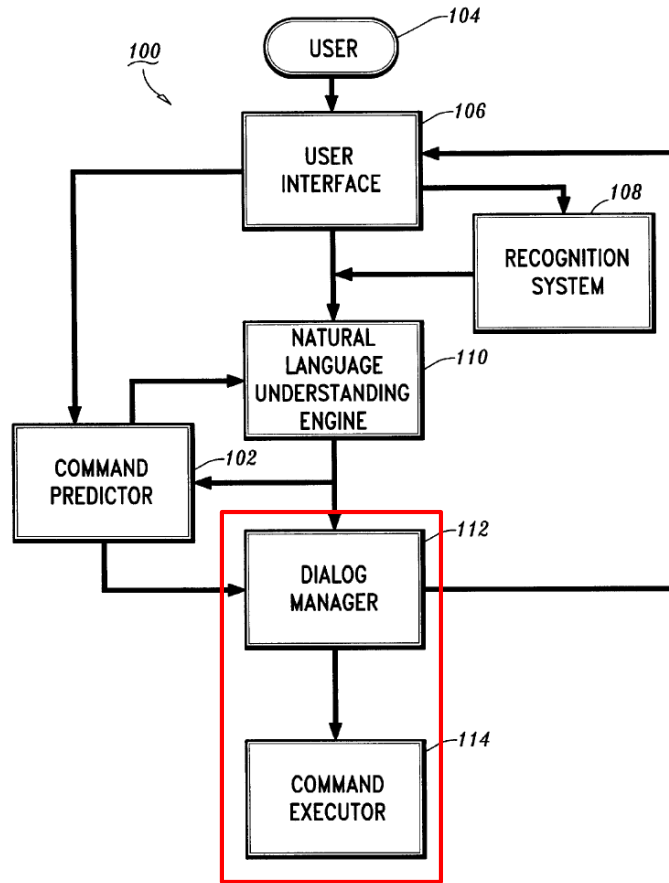


FIG. 1

EX1009, FIG. 1 (red annotation added).

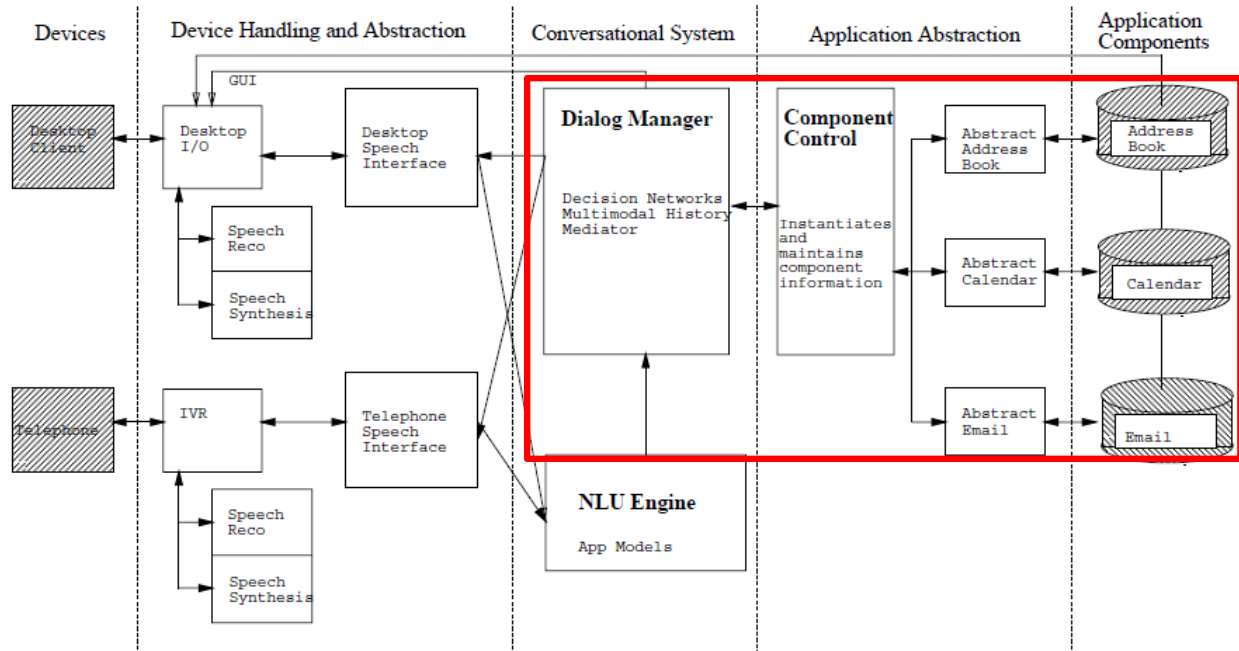


Figure 1: Architecture of the multi-client, multi-application, multi-modal system

EX1014, 2 (red annotation added).

135. Further, Ramaswamy directs one of ordinary skill to Ramaswamy NPL via its incorporation by reference. EX1009, 4:49-55. Although Ramaswamy does not expressly refer to Ramaswamy NPL's Application Abstraction, one of ordinary skill would have immediately recognized the high degree of similarity between the command executor and the Application Abstraction at least because of their identical placement in the diagrams shown above and analogous functionality. *Id.*, 4:29-35, FIG. 1; EX1014, 1-2. Because Ramaswamy provides few details on the command executor (EX1009, 4:24-25, 4:34-35), one of ordinary skill would have been motivated with a reasonable expectation of success to combine Ramaswamy NPL's Application Abstraction teachings with Ramaswamy's teachings on its command

executor to gain a fuller, more robust understanding of the command executor and how it works. Specifically, one of ordinary skill would have gone looking for additional details on the command executor, and, having already been directed to Ramaswamy NPL by Ramaswamy (via its incorporation by reference) (*id.*, 4:49-55.), would have come upon Ramaswamy NPL and immediately recognized the high degree of similarity between the command executor and the Application Abstraction and been motivated to supplement Ramaswamy's command executor teachings with Ramaswamy NPL's Application Abstraction teachings.

136. Thus, to the extent Ramaswamy NPL's Application Abstraction teachings are not a part of Ramaswamy, one of ordinary skill would have found it obvious to incorporate Ramaswamy NPL's Application Abstraction teachings in Ramaswamy's system, specifically in Ramaswamy's command executor. These Application Abstraction teachings from Ramaswamy NPL teach claim element [42.5], as explained above in Section XI.B.5. *Supra* ¶¶ 101-104; *Id.*, 4:21-35; EX1014, 1-2. And, as explained in Section XI.B., Ramaswamy teaches the other elements of claim 42. Thus, Ramaswamy in combination with Ramaswamy NPL renders claim 42 obvious.

XIII. GROUND 4: CLAIM 42 IS OBVIOUS OVER RAMASWAMY IN VIEW OF HARTONO (EX1010)

137. To the extent Patent Owner argues that Ramaswamy and/or Ramaswamy NPL do not teach element [42.5] (the domain agent limitation)—which

is wrong—claim 42 is obvious over Ramaswamy and Hartono because Hartono teaches this claim element, and, as explained below, one of ordinary skill would have been motivated with a reasonable expectation of success to combine these domain agent teachings from Hartono with Ramaswamy’s command executor teachings. Specifically, the Ground 4 combination combines Ramasamy’s system with Hartono’s task/class agent teachings, and, more specifically, incorporates Hartono’s task/class agent teachings in the command executor of Ramasamy’s system.

138. Hartono is analogous art to the ’659 patent because it describes systems and methods for processing natural language utterances. *E.g.*, EX1010, Abstract, 1:19-7:13, 9:3-14, FIG. 1.

139. Hartono teaches claim element [42.5] (transmitting the request to a domain agent for processing) at least because it discloses that “the request is routed to [a] specialized task agent” (EX1010, 15:14-15), which Hartono also refers to as a “application class agent” (*id.*, *passim*). *See also, e.g., id.*, 16:25-17:4. These task/class agents are domain agents at least because they include software with domain-specific behavior and information. *Supra* Section VIII.A; *Id.*, 4:13, 5:4-8, 7:1-9, 7:22-24, 8:10-13, 9:5-16, 15:13-15, 17:8-14, 23:15-30:20, FIGS. 1, 4.

140. Specifically, each task/class agent “preferably works with a single class of applications [] that have similar conceptual operations.” *Id.*, 23:19-20; *see also, e.g., id.*, 5:4-8, 10:14-17, 15:13-15, 23:15-32:12. “For example, different email

application generally perform the same conceptual actions of sending and receiving mail.” *Id.*, 23:20-22.

141. Exemplary task/class agents include a “meeting agent,” a “personal information manager agent,” an “email agent,” and a “voice training agent.” *Id.*, 9:11-13, FIG. 1.

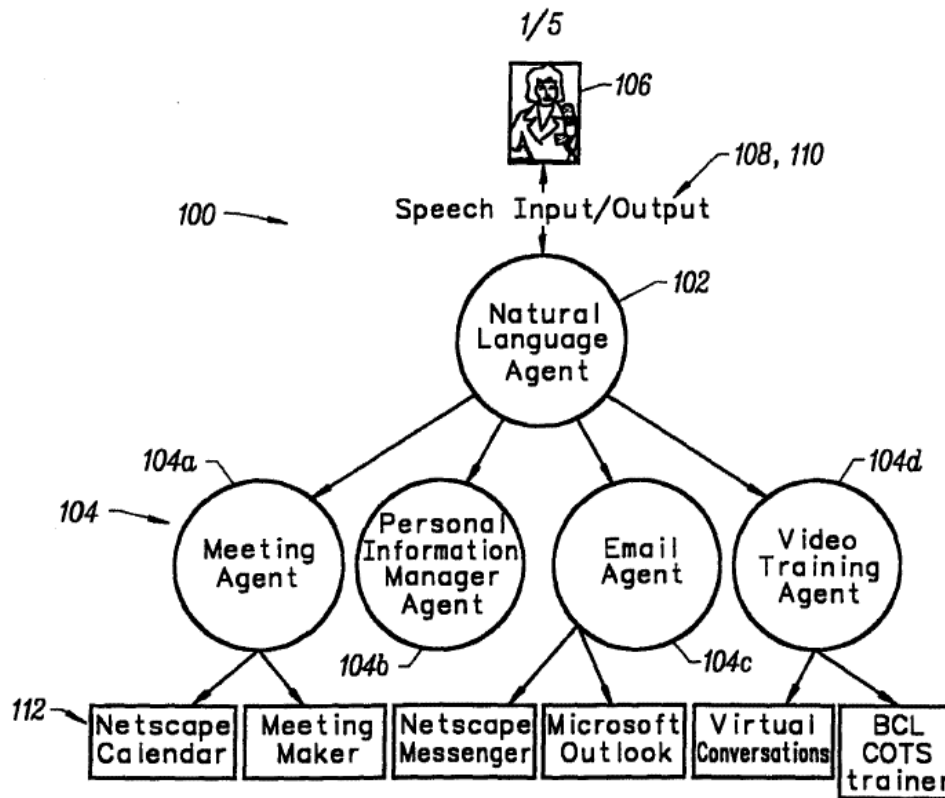


FIG. 1

EX1010, FIG. 1.

142. Table IV of Hartono includes a more exhaustive list of task/class agents.

TABLE IV	
TASK AGENT	SAMPLE TASKS
Mail	Sends, receives, composes and views email
Fax	Sends, receives, composes and views faxes
Letter	Composes, writes, and send letters
File	Manages files
OS	Manages OS, configuration, performance
Address	Manages Address books
Games	Plays and operates specific games
Flight Simulators	Runs flight simulators
Vehicle Simulators	Operates Land Vehicle Simulators (cars, bikes, tanks, etc.)
Naval Simulators	Operates water-based simulators (ships, subs etc.)
Sports Simulators	Operates Baseball, Soccer, Football etc. simulators
War Game & Strategy Simulators	Operates turn and real time-based war games, single and multi-player
Role Play Simulators	Operates avatar-based role playing games such as DOOM, TOMB RAIDER, ADVENTURE, ZORK
Action Simulators	Operates action games
PIM (personal information manager)	General interface to task, calendar, address book, and notebook manager
Printer	Selects and configures printers, prints documents
Calendar	Manages calendar, sets meetings and ticklers
Terminal	Connects to remote systems, logs on, logs off
Travel	Arranges travel
Encyclopedia	Searches for and displays information from encyclopedic literature
Image Viewer	Views and manipulates images
C++	Assists in managing and writing C/C++ Programs
Basic	Assists in managing and writing Basic Programs
GUI	Manipulates, configures and arranges Graphical User Interface
Presentation	Draws, arranges and manipulates slide presentations
Charting	Charts sets of numbers in various graphs and charts

TABLE IV	
TASK AGENT	SAMPLE TASKS
Meeting	Arranges and schedules meetings
Scheduler	Schedules tasks on the computer
Telephone	Dials out and receives calls; integrates with address book
Voice Mail	Sends, receives, plays, manages voicemail
Word Processor	Writes, prints, manipulates, formats documents
Spreadsheet	Writes, prints, manipulates, formats numerical data
Drawing	Draws, manipulates, formats diagrams, merges predrawn images
Web	Connects, navigates, searches the internet/world-wide web
Network	Connects networks, manages connections
Mathematical	Manages mathematical and scientific manipulation of numeric and formulaic data
Directory Assistance	Locates telephone numbers and addresses over the internet
Internet Retail Sales	Describes objects and sells to customers over the internet
Common Household Utility Agent (e.g., VCR, Toaster, HVAC)	Controls household devices
K-12 Education on, e.g. Physics, Chemistry, Math	Teaches, runs games, quizzes, etc. for mathematics and science subjects
General Education on History, Economics, Philosophy	Teaches specific liberal arts and humanities courses
Hands-on training for job-based tasks	Trains users to operate equipment
Internet Event lookup	Locates events, such as conferences, meetings, concerts and festivals, through the internet
Internet Product-Information lookup	Finds products and prices through the internet
Internet-based Meeting scheduler	Schedules meetings over the internet
Hardware manager	Manager Computer Hardware (screen, disk, etc.)

EX1010, 28-29 (Table IV).

143. These task agents are embodied in software at least because Hartono describes how its system, including the task agents, are “preferably embodied in a

computer program product in the form of computer coded instructions executable by a computer processor and stored in a computer readable medium.” EX1010, 33:17-20; *see also id.*, 33:21-35:3, FIGS. 7-8.

144. Thus, Hartono’s task/class agents are software with domain-specific behavior and information at least because they are software modules that include information and behavior that is specific to a given class of applications (e.g., an email agent that includes information and behavior for executing requests/commands within a set of email applications).

145. These domain agents process the user requests and return a response. *Id.*, 9:13-14, 10:17-19, 24:21-25:3, 26:14-21, 27:9-30:14. Specifically, the task/class agent’s interpretation engine (e.g., task interpretation engine 104B) receives and processes (e.g., interprets) the request from the natural language agent (e.g., natural language agent 102) and “invokes a routine that sends a set of requests to the task application wrapper” (*id.*, 25:2-3), which interfaces with the different applications in the class to perform the request. *Id.*, 9:13-14, 10:14-20, 13:12-14:7, 24:4-25:3, FIGS. 1, 4.

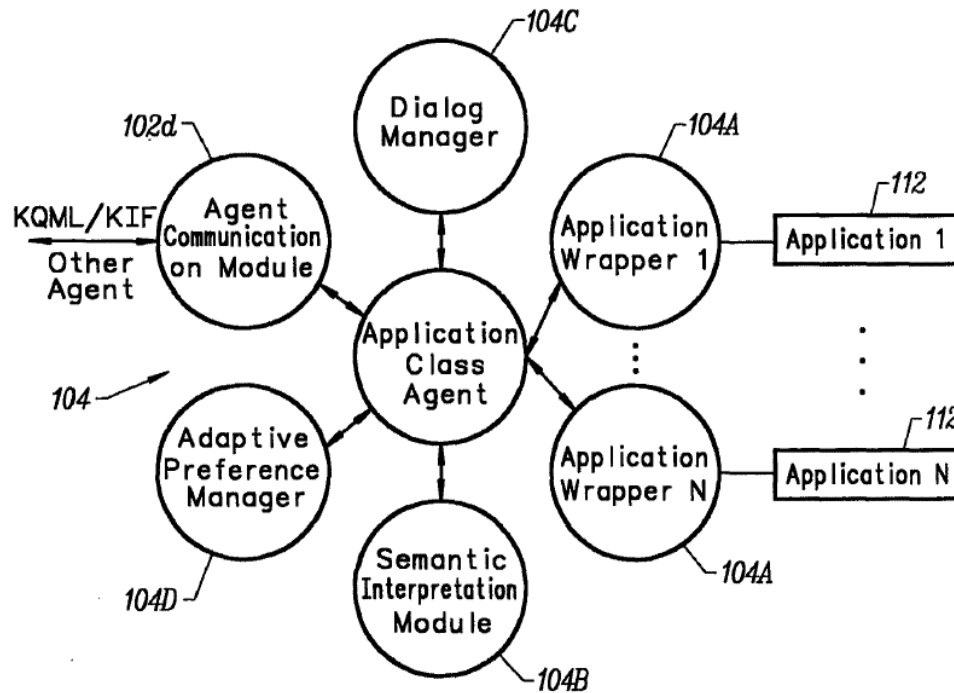


FIG. 4

EX1010, FIG. 4

146. The task agent then returns a response to the “natural language agent [] which then delivers the natural language output [] to the user.” EX1010, 9:14-15; *see also id.*, 10:14-20. For example, an address book agent displays all or part of an address. *Id.*, 26:16-17.

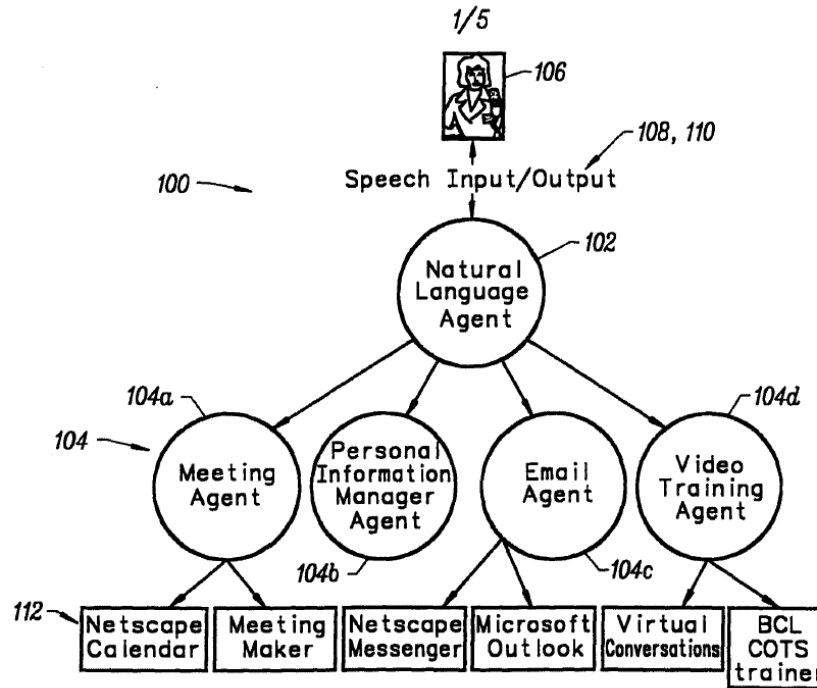


FIG. 1

EX1010, FIG. 1.

147. One of ordinary skill in the art would have been motivated with a reasonable expectation of success to incorporate Hartono's domain agent teachings in Ramaswamy's command executor to more efficiently handle a broad range of user requests, as taught by Hartono. EX1010, 7:1-9, 17:8-14. Specifically, as explained above in Ground 2, Ramaswamy teaches a command executor (EX1009, 4:24-25, 4:34-35, FIG. 1), including an email agent (EX1014, 2), and Ramaswamy NPL teaches employing a domain agent for handling requests involving address book, calendar, and email applications that includes specific agents for each of those applications (*id.*, 1-2).

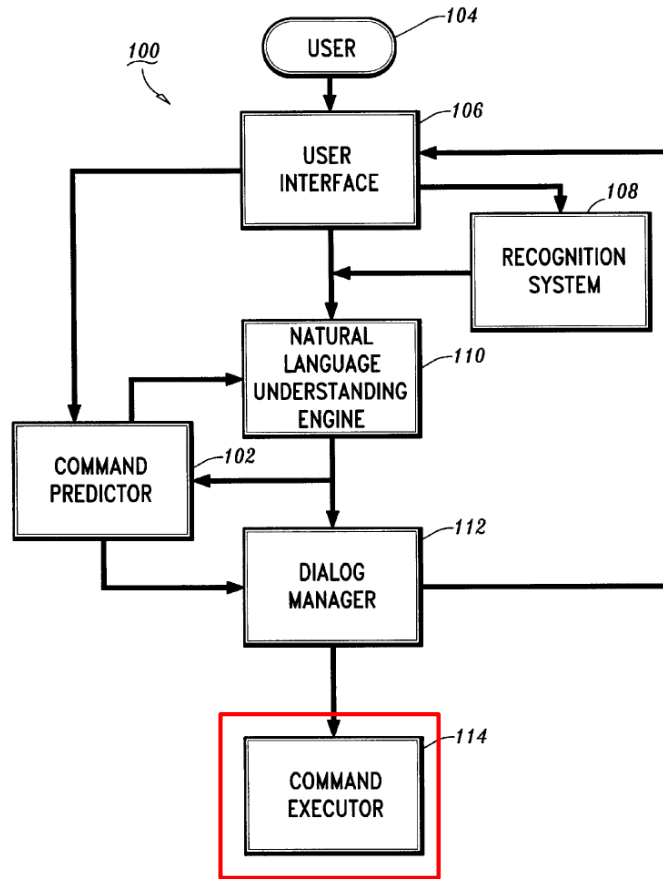


FIG. 1

EX1009, FIG. 1 (red annotation added).

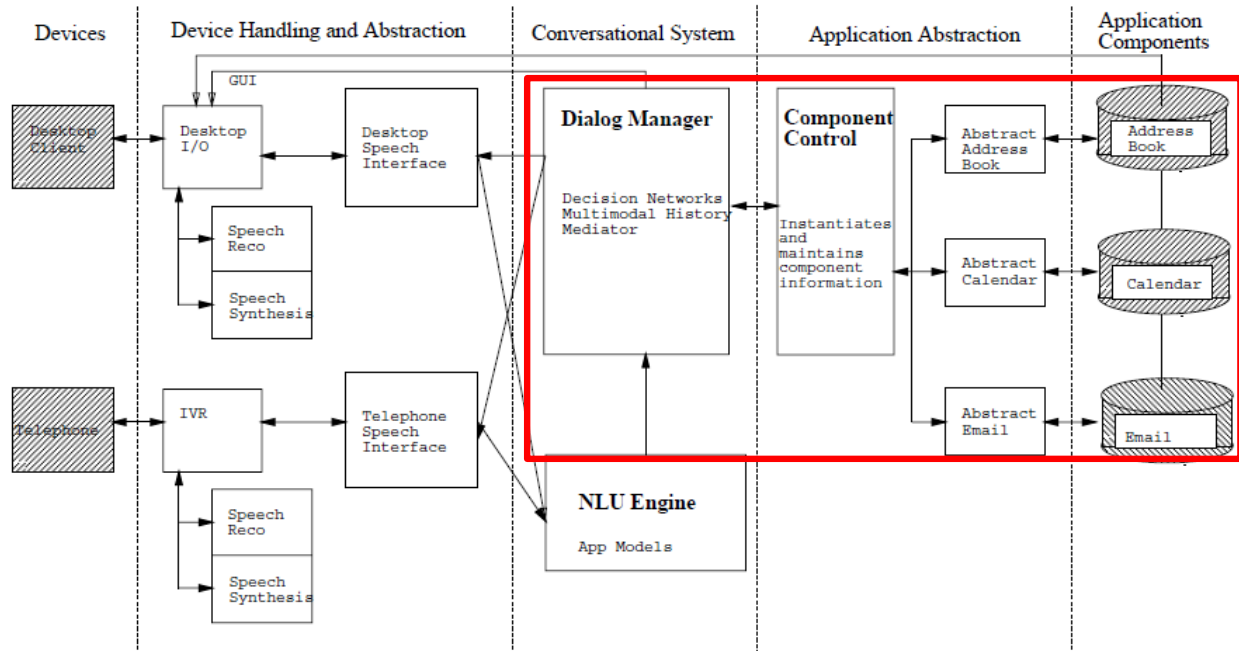


Figure 1: Architecture of the multi-client, multi-application, multi-modal system

EX1014, 2 (Figure 1) (red annotation added).

148. Hartono provides additional details on domain agents and explains how to modularize the domain agents so that they are “extendable to multiple applications and [] scalable to large sets of networked computer systems.” EX1010, 7:8-9, *see also id.*, 17:8-14. Specifically, by providing task/class agents for each specific class of applications, task/class agents can be created for any class of application, allowing the system to easily handle requests involving a broad range of applications. *Id.*, 7:1-3, 7:4-9 (“Each agent works with a class of applications, such as electronic mail, and communicates with specific applications through application wrappers. Thus, with a modular distributed agent architecture, the system and method of the present

invention is extendable to multiple applications and is scalable to large sets of networked computer systems.”), 17:8-14.

149. Further, one of ordinary skill would have been motivated to implement Ramaswamy’s command executor to include different domain agents for different types of commands and/or different classes of applications to efficiently handle a broad range of requests, as taught by Hartono. *Id.*, 7:1-9, 17:8-14, 23:15-32:12.

150. One of ordinary skill in the art would have reasonably expected to succeed in implementing Hartono’s domain agent teachings in Ramaswamy command executor at least because they would have required minor software modifications that were well within the skill of one of ordinary skill. Ramaswamy’s command executor software module already executes the user’s requests. EX1009, 3:34-35, 4:24-25. Thus, to implement Hartono’s teachings in Ramaswamy, all the skilled artisan would need to do is include Hartono’s individual domain agent software modules within Ramaswamy’s command executor software module (i.e., include specific sub-modules for each class of application and/or type of request), which was well within the programming skill of one of ordinary skill.

I, Paul Jacobs, do hereby declare and state, that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

Dated: Aug. 4, 2025


Signature

Paul S. Jacobs

Paul@jaketech.com
mobile: (703) 307-0467

4105 Faith Ct.
Alexandria VA 22311

Education

Ph.D. in Computer Science
S.M. in Applied Mathematics
A.B. in Applied Mathematics

University of California at Berkeley (1985)
Harvard University (1981)
Harvard University (1981)

Work History

Founder and President, Jake Technologies, Inc. (7/02 -)

- Founded technology services company focused on strategic technology development, product evaluation, and advising corporate counsel and law firms on issues related to intellectual property. Serve as expert consultant in patent litigation, research, and related matters.
- Appointed by the office of the Secretary of Commerce to the Patent Public Advisory Committee, advising the Under Secretary and Director of the U.S.. Patent and Trademark Office from 2012 – 2015.

Chief Technology Officer, Primus Knowledge Solutions, Inc. (6/01 – 7/02)

- Played senior executive role in integration of AnswerLogic into Primus, a publicly-held CRM software company focusing on knowledge capture and delivery in support center and web self-service applications.
- As CTO, responsible for technology “evangelism” both inside of and outside of Primus, including articulating company strategy and vision to analysts, customers, and prospects and working with product and product management teams.

President, Chief Operating Officer, and Chief Technology Officer, AnswerLogic, Inc. (10/99 – 5/01)

- Joined start-up at close of Series A financing, led through Series B financing and initial product launch and made sale of company to Primus Knowledge Solutions.
- Led technology and general management of over 90 employees, including over 50 linguists and over 30 engineers. Only senior executive retained after Primus acquired AnswerLogic.

Managing Vice President, Electronic Commerce, SRA International (9/98 – 10/99)

- Managed NetOwl product line and related IsoQuest technologies for parent company; SRA derived over \$10 million in product licenses and millions more in associated services from NetOwl and related products, with over 100 customers. SRA, which soon went public, was ranked in Business Week’s Top 20 Private IT firms in 1999, cited for e-mail software based on NetOwl.

President and Chief Executive Officer, IsoQuest, Inc. (7/96 – 9/98)

- Founded and led advanced Internet-focused knowledge management software company, backed by SRA International. IsoQuest’s main product, NetOwl, became the leading engine for automatic information extraction as well as forming the basis for the parent company’s product offering. In 1998, with IsoQuest at break-even, SRA re-integrated the subsidiary for strategic reasons.
- Responsible for all major financial, strategy, staffing and management decisions
- Conceived product strategy, launched and sold over \$2 million of new product (NetOwl) to 20 clients in first two years; brought company to break-even.

APPENDIX A

- Personally guided marketing, sales, and product management until executives in each area were hired.
- Landed major accounts, including Thomson Corporation, Knight Ridder, Bloomberg, Lexis/Nexis, Disclosure, NewsEdge, and Infoseek.

Director of Product Marketing, SRA International (7/95 – 6/96)

- Led launch of (\$200M) consulting firm's first software product (NameTag)
- Developed strategic marketing, branding, product management, and marketing communications programs, hired and managed agencies and contractors and led product marketing initiatives
- Successfully presented business plan and laid groundwork for formation of new company (IsoQuest)

Director of Media Information Technologies, SRA International (8/94 – 6/95)

- Developed and implemented long-range technology transfer plan
- Supported SRA business units (media, intelligence, others) in new initiatives and major sales opportunities
- Led search engine re-design and deployment for new venture (Picture Network International)

Computer Scientist, GE Corporate Research and Development (8/85-8/94)

- Led GE's initiatives in natural language text processing. Built world's leading team in automatic information extraction from free text. Received numerous management awards. Whitney Gallery of Technical Achievers
- Principal investigator for four years in U.S. government's Tipster program (Tipster was cited with the "Hammer" award as a national reinvention laboratory)
- Proposed, won, and deployed advanced text management projects in GE Information Services, Aircraft Engines, Aerospace, Capital Services, and external contracts with ARPA and other agencies.
- Completed GE Technical Management Course
- Spent one year as Visiting Associate Professor at University of Pennsylvania (Computer and Information Sciences, Institute for Research in Cognitive Science)
- Consulted with Infonautics, Inc. (start-up in Wayne, PA) – helped map out initial business plan and technical approach to Homework Helper, their first product.

Professional Activities

Author of over 50 published journal articles and technical papers. Holder of two U.S. Patents.

Author of *Text-Based Intelligent Systems* (Lawrence Erlbaum Associates, 1992).

Adjunct Professor, College of Information Studies (iSchool), University of Maryland. Currently teach Information Architecture. Delivered 2008 commencement address.

Member of Patent Public Advisory Committee (PPAC), US Dept. of Commerce, 2012-2015

Member of USACM (Public Policy Committee of Major Computer Trade Association)

Member of Intellectual Property Committee of the USACM

Executive Committee, Association for Computational Linguistics, 1997-9

Other Activities

Accomplished skier, distance runner. Iron Man Triathlete. Winner, Golden Gate Marathon, 1983, 1984. Enjoy exotic travel, adventure sports, hiking with my family.

APPENDIX A

PAUL S. JACOBS

PUBLICATIONS, PROFESSIONAL AND SCHOLARLY ACTIVITIES

Refereed Journal Articles

P. Jacobs. PHRED: A generator for natural language interfaces. *Computational Linguistics* 11 (4), pp. 219-242, 1985. Also appearing in L. Bolc and D. McDonald (eds.), *Natural Language Generation Systems*, Springer Verlag, 1988

P. Jacobs. Knowledge intensive natural language generation. *Artificial Intelligence* 33 (3), pp. 325-378, 1987.

L. Rau and P. Jacobs. $NL \cap IR$: Natural language for information retrieval. *International Journal of Intelligent Systems* 4 (3), pp. 319-343, 1989.

L. Rau, P. Jacobs, and U. Zernik. Information extraction and text summarization using linguistic knowledge acquisition. *Information Processing and Management* 25 (4), pp. 419-428, 1989.

P. Jacobs and L. Rau. SCISOR: A system for extracting information from on-line news. In *Communications of the Association for Computing Machinery* 33 (11), pp. 88-97, November 1990.

P. Jacobs. TRUMP: A transportable language understanding program. *International Journal of Intelligent Systems* 7 (3), pp. 245-276, 1992.

P. Jacobs and L. Rau. Innovations in text interpretation. *Artificial Intelligence* 63 (1-2), pp. 143-191, 1993.

P. Jacobs. Using statistical methods to improve knowledge-based news categorization. *IEEE Expert*, 8(2), pp. 13-23, Apr. 1993.

Refereed Conference Proceedings

P. Jacobs. Generation in a natural language interface. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, Karlsruhe, Germany, 1983.

P. Jacobs and L. Rau. Ace: Associating language with meaning. In *Proceedings of the Sixth European Conference on Artificial Intelligence*, Pisa, Italy, 1984. Reprinted in T. O'Shea (ed.), *Advances in Artificial Intelligence*, North Holland, 1986.

P. Jacobs. The KING natural language generator. In *Proceedings of the Seventh European Conference on Artificial Intelligence*, Brighton, England, 1986. Reprinted in B. du Boulay, D. Hogg, and L. Steels (eds.), *Advances in Artificial Intelligence II*, North Holland, 1987.

P. Jacobs. Knowledge structures for natural language generation. In *Proceedings of the 11th International Conference on Computational Linguistics*, pp. 554-559, Bonn, 1986.

P. Jacobs. Language analysis in not-so-limited domains. In *Proceedings of the Fall Joint Computer Conference*, Dallas, 1986.

APPENDIX A

P. Jacobs. A knowledge framework for natural language analysis. In *Proceedings of the 10th International Joint Conference on Artificial Intelligence*, Milan, 1987.

D. Besemer and P. Jacobs. FLUSH: A flexible lexicon design. In *Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics*, pp. 186-193, Palo Alto, 1987.

P. Jacobs. Achieving bidirectionality. In *Proceedings of the 12th International Conference on Computational Linguistics*, pp. 267-274, Budapest, 1988.

P. Jacobs. Concretion: Assumption-based understanding. In *Proceedings of the 12th International Conference on Computational Linguistics*, Budapest, 1988.

P. Jacobs. Why text planning isn't planning. In *Proceedings of the AAAI Workshop on Text Planning and Generation*, St. Paul, Minnesota, 1988.

P. Jacobs and U. Zernik. Acquiring lexical knowledge from text: A case study. In *Proceedings of the National Conference on Artificial Intelligence*, St. Paul, Minnesota, 1988.

P. Jacobs and L. Rau. Natural language techniques for intelligent information retrieval. In *Proceedings of the 11th International Conference on Research and Development in Information Retrieval*, Grenoble, France, 1988.

P. Jacobs. Extensible natural language interfaces. In *Proceedings of the 17th ASIS Mid-Year Meeting*, Ann Arbor, 1988.

P. Jacobs and L. Rau. A friendly merger of conceptual expectations and linguistic analysis in a text processing system. In *Proceedings of the Fourth IEEE Conference on Artificial Intelligence Applications*, pp. 351-356, San Diego, 1988.

L. Rau and P. Jacobs. Integrating top-down and bottom-up strategies in a text processing system. In *Proceedings of the Second Conference on Applied Natural Language Processing (ANLP)*, pp. 129-135, Austin, 1988.

P. Jacobs and L. Rau. The GE NLToolset: A software foundation for intelligent text processing. In *Proceedings of the 13th International Conference on Computational Linguistics*, Helsinki, 1990.

U. Zernik and P. Jacobs. Tagging for learning: Collecting thematic relations from Corpus. In *Proceedings of the 13th International Conference on Computational Linguistics*, pp. 34-39, Helsinki, 1990.

P. Jacobs, G. Krupka, L. Rau, N. Sondheimer, and U. Zernik. Generic text processing. In *Proceedings of the Third DARPA Speech and Natural Language Workshop*, 359-364, Somerset, PA, 1990.

P. Jacobs. To parse or not to parse: Relation-driven text skimming. In *Proceedings of the 13th International Conference on Computational Linguistics*, pp. 194-198, Helsinki, 1990.

APPENDIX A

L. Rau and P. Jacobs. Creating segmented databases from free text for text retrieval. In *Proceedings of the 14th International Conference on Research and Development in Information Retrieval*, pp. 337-346, Chicago, 1991.

P. Jacobs, G. Krupka, and L. Rau. Lexico-semantic pattern matching as a companion to parsing in text understanding. In *Proceedings of the Fourth DARPA Speech and Natural Language Workshop*, Asilomar, CA, 1991.

P. Jacobs. Parsing run amok: Relation-driven control for text analysis. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pp. 315-321, San Jose, 1992.

P. Jacobs. Joining statistics with NLP for text categorization. In *Proceedings of the Third Conference on Applied Natural Language Processing*, pp. 178-185, Trento, Italy, 1992.

P. Jacobs, G. Krupka, and L. Rau. A Boolean approximation method for query construction and topic assignment in TREC. In *Proceedings of the Second Annual Symposium on Document Analysis and Retrieval*, Las Vegas, 1993.

P. Jacobs. Word sense acquisition for multilingual text interpretation. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING 94)*, Vol. II, pp. 665-671, 1994.

P. Jacobs. Browsing vs. Surfing: The Next Generation of Search and Retrieval. In *Proceedings of the National On-Line Conference*. New York, 1998.

Books and Book Chapters

P. Jacobs. KING: A knowledge-intensive natural language generator. In G. Kempen (ed.), *Natural Language Generation: Recent Advances in Artificial Intelligence, Psychology, and Linguistics*, Kluwer Academic Publishers, 1987.

P. Jacobs. Two hurdles for natural language systems. In R. Freedle, (ed.), *Artificial Intelligence and the Future of Testing*, pp. 257-277, Lawrence Erlbaum Associates, Hillsdale, NJ, 1990.

P. Jacobs. Making sense of lexical acquisition. In U. Zernik, (ed.), *Lexical Acquisition: Exploiting On-line Resources to Build a Lexicon*, pp. 29-44, Lawrence Erlbaum Associates, Hillsdale, NJ, 1991.

P. Jacobs. Integrating language and meaning in structured inheritance networks. In J. Sowa, (ed.), *Principles of Semantic Networks: Explorations in the Representation of Knowledge*, pp. 527-542, Morgan Kaufman Publishers, San Mateo, CA, 1991.

P. Jacobs. Text power and intelligent systems. In *Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval*, pp. 1-8, Lawrence Erlbaum Associates, Hillsdale, NJ, 1992.

P. Jacobs (ed). *Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval*. Lawrence Erlbaum Associates, Hillsdale, NJ, 1992.

APPENDIX A

P. Jacobs. Why Text Planning (Still) Isn't Planning. In A. Ortony, J. Slack, and O. Slack (eds.), *Communication from an Artificial Intelligence Perspective*, Springer Verlag, Heidelberg, 1992.

P. Jacobs. Information extraction. In R. Cole, J. Mariani, H. Uszkoreit, A. Zaenen, and V. Zue (eds.), *Survey of the State of the Art in Human Language Technology*, pp. 230-232, Cambridge University Press, 1997.

Other Published Articles

P. Jacobs. Generation in the UNIX Consultant System, in D. Roesner (ed.), *International Workshop on Language Generation*, Institut fuer Informatik, Universitaet Stuttgart, 1983.

P. Jacobs. Teaching language to computers. *IEEE Potentials*, October, 1986.

P. Jacobs and L. Rau. Software for intelligent text processing. In *Proceedings of the 13th International Conference on Computational Linguistics (project notes)*, Helsinki, Vol. 3, 1990, pp 373-375.

G. Krupka, L. Iwanska, P. Jacobs, and L. Rau. GE NLToolset test results and analysis. In *Proceedings of the Third Message Understanding Conference (MUC-3)*, San Diego, 1991.

G. Krupka, P. Jacobs, L. Rau and L. Iwanska. Description of the GE NLToolset system as used in MUC-3. In *Proceedings of the Third Message Understanding Conference (MUC-3)*, San Diego, 1991.

P. Jacobs, G. Krupka, and L. Rau. The right tools for the job: Multiple language analysis strategies in the GE NLToolset. In *Proceedings of the Workshop on Fully Implemented Natural Language Systems*, Trento, Italy, March, 1992.

P. Jacobs, G. Krupka, L. Rau, T. Kaufmann and M. Mauldin. GE-CMU: Description of the TIPSTER/SHOGUN system as used for MUC-4. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, McLean, VA, 1992.

L. Rau, G. Krupka, P. Jacobs, I. Sider and L. Childs. GE NLToolset test results and analysis. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, McLean, VA, 1992.

P. Jacobs, G. Krupka, L. Rau, M. Mauldin, T. Mitamura, T. Kitani, I. Sider, and L. Childs. GE-CMU: description of the SHOGUN system used for MUC-5. In *Proceedings of the Fifth Message Understanding Conference (MUC-5)*, pp. 109-120, 1993.

P. Jacobs, G. Krupka, L. Rau, M. Mauldin, T. Mitamura, T. Kitani, I. Sider, and L. Childs. The TIPSTER/SHOGUN Project. In *Proceedings of the TIPSTER Text Phase I Final Meeting*, pp. 209-221, Morgan Kaufmann, 1993.

P. Jacobs. GE in TREC-2: Results of a Boolean approximation method for routing and retrieval. In *Proceedings of the Second Text Retrieval Conference (TREC-2)*, D. Harman (ed.), pp. 191-199, NIST Special Publication 500-215, 1994.

APPENDIX A

P. Jacobs. Natural language processing: A brief history for skeptics. *ServerWorld*, February 2001.

Technical Reports Not Published Elsewhere

P. Jacobs. *A Knowledge-Based Approach to Language Production*. Ph. D. Thesis, University of California, Berkeley. Available as Computer Science Division Technical Report 86/254, 1985.

P. Jacobs. A cooperative natural language directory system. General Electric Corporate Research and Development, Report 88CRD025, 1988.

Research Grants

Principal investigator of Tipster Phase I (GE, 1991, \$2.8 million) and Phase II (GE, 1993) \$10s of millions in commercial financing for advanced research and development from 1994 through 2000.

Professional Societies and Committees

Member of IEEE, ACM, AAAI, and ACL for many years.

Program committees:

- AAAI National Conference on Artificial Intelligence

- ACL Annual Conference

- ACL Conference on Applied Natural Language Processing

- ACM SIGIR Conference on Research and Development in Information Retrieval

- ARPA Speech and Natural Language Conference

- Others

Program chair, AAAI Symposium on Text-Based Intelligent Systems, 1990 (led to the publication of the book, *Text-Based Intelligent Systems* and to an increase in cross-disciplinary work in computational linguistics and information retrieval)

Program chair, ACL Conference on Applied National Language Processing, Stuttgart, Germany, 1994.

Executive Committee of the ACL, 1995-1998

USACM Public Policy Committee, Intellectual Property Subcommittee, 2005 -

U.S. Patents

5,243,520 Sense Discrimination System and Method, Sept. 7, 1993

5,251,129 Method for Automated Morphological Analysis of Word Structure, Oct. 5, 1993