

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

DATABRICKS, INC.,
Petitioner,

v.

BYTEWEAVR INC.,
Patent Owner.

Case No. IPR2025-00714
U.S. Patent No. 6,839,733

Petition for *Inter Partes* Review of U.S. Patent No. 6,839,733

TABLE OF CONTENTS

	<u>Page</u>
TABLE OF AUTHORITIES.....	v
PETITIONER’S EXHIBIT LIST	vii
MANDATORY NOTICES	1
A. Real Parties-in-Interest [37 C.F.R. § 42.8(b)(1)].....	1
B. Related Matters [37 C.F.R. § 42.8(b)(2)]	1
C. Lead and Back-Up Counsel [37 C.F.R. § 42.8(b)(3)].....	1
D. Service Information [37 C.F.R. § 42.8(b)(4)].....	2
I. INTRODUCTION	3
II. REQUIREMENTS OF <i>INTER PARTES</i> REVIEW	4
A. Standing	4
B. Identification of Challenge and Relief Requested	4
1. U.S. Patent No. 6,029,175 (“Chow”) (Ex. 1004)	5
2. U.S. Patent No. 6,263,358 (“Lee”) (Ex. 1006).....	5
3. U.S. Patent No. 5,603,031 (“White”) (Ex. 1005).....	5
C. How the Challenged Claims Are To Be Construed Under 37 C.F.R. § 42.104(b)(3).....	6
D. How the Challenged Claims Are Unpatentable Under 37 C.F.R. § 42.104(b)(4)	7
E. Supporting Evidence Under 37 C.F.R. § 42.104(b)(5).....	7
III. PATENT OVERVIEW	8
A. Alleged Invention of the ’733 Patent.....	8
B. Prosecution History of the ’733 Patent.....	13

TABLE OF CONTENTS *(continued)*

	<u>Page</u>
C. Person of Ordinary Skill in the Art.....	14
D. Prior Instituted IPR on the '752 Patent.....	15
IV. GROUNDS OF UNPATENTABILITY	16
A. Overview of White as Prior Art.....	16
B. Ground 1: Chow in View of White Renders Obvious Claims 37-43	18
1. Overview of Grounds and Prior Art	18
2. A POSITA Would Have Been Motivated to Combine Chow and White	23
3. Detailed Analysis of Ground 1	27
a. Element 37[pre] A method comprising:.....	27
b. Element 37.a[1] admitting a user to a network system ..	27
c. Element 37.a[2] wherein at least one agent is operable to consume a service resource while utilizing a service to perform a task for the user; and.....	29
d. Element 37.b allowing the user to create, modify, or delete the agent within the network system.	34
e. Claim 38 The method of claim 37 further comprising accepting an agent command from the user.....	36
f. Claim 39 The method of claim 38 wherein the agent command is one of a create agent command, an edit agent command, and a delete agent command.....	36
g. Claim 40 The method of claim 37 further comprising suspending execution of the agent.	40
h. Claim 41 The method of claim 37 further comprising accepting a template command from the user.....	42

TABLE OF CONTENTS (continued)

	<u>Page</u>
i. Claim 42 The method of claim 41 further comprising allowing the user to create, modify, or delete an agent template within the network system.....	43
j. Claim 43 The method of claim 37 further comprising admitting the user to an agent area of a user interface...44	44
C. Ground 2: Lee in View of White Renders Obvious Claims 37-43	44
1. Overview of Grounds and Prior Art	45
2. A POSITA Would Have Been Motivated to Combine Lee and White.....	51
3. Detailed Analysis of Ground 2	55
a. Element 37[pre] A method comprising:.....	55
b. Element 37.a[1] admitting a user to a network system ..	55
c. Element 37.a[2] wherein at least one agent is operable to consume a service resource while utilizing a service to perform a task for the user; and.....	57
d. Element 37.b allowing the user to create, modify, or delete the agent within the network system.	60
e. Claim 38 The method of claim 37 further comprising accepting an agent command from the user.....	61
f. Claim 39 The method of claim 38 wherein the agent command is one of a create agent command, an edit agent command, and a delete agent command.....	62
g. Claim 40 The method of claim 37 further comprising suspending execution of the agent.	63
h. Claim 41 The method of claim 37 further comprising accepting a template command from the user.....	63

TABLE OF CONTENTS (*continued*)

	<u>Page</u>
i. Claim 42 The method of claim 41 further comprising allowing the user to create, modify, or delete an agent template within the network system.....	68
j. Claim 43 The method of claim 37 further comprising admitting the user to an agent area of a user interface...	69
V. ENABLEMENT AND SECONDARY CONSIDERATIONS	71
VI. PTAB DISCRETION SHOULD NOT PRECLUDE INSTITUTION	72
A. <i>Fintiv</i> Analysis.....	72
VII. CONCLUSION	75
APPENDIX: CHALLENGED CLAIM LISTING	78

TABLE OF AUTHORITIES

Cases

<i>Apple Inc. v. Fintiv, Inc.</i> , IPR2020-00019, Paper 11 at 5–6 (P.T.A.B. Mar. 20, 2020)	74
<i>Byteweavr, LLC v. Cloudera, Inc.</i> , (Case No. 1:24-cv-00261 (W.D. Tex.)).....	1
<i>Byteweavr, LLC v. Databricks, Inc.</i> , No. 2:24-cv-00162-JRG-RSP (E.D. Tex.).....	1
<i>Google LLC v. Parus Holdings, Inc.</i> , IPR2020-00847, Paper 9 at 12 (P.T.A.B. Oct. 21, 2020)	75
<i>Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co.</i> , 868 F.3d 1013, 1017 (Fed. Cir. 2017).....	7
<i>Sand Revolution II, LLC v. Continental Intermodal Group-Trucking LLC</i> , IPR2019-01393, Paper 24 (P.T.A.B. June 16, 2020).....	76
<i>VMware, Inc. v. Intellectual Ventures I LLC</i> , IPR2020-00470	7, 27, 33
<i>Weatherford U.S. v. Enventure Global Tech., Inc.</i> , IPR2020-01684, Paper 16 at 11–13 (P.T.A.B. Apr. 14, 2021).....	77

Statutes

35 U.S.C. § 102(b).....	5
35 U.S.C. § 102(e).....	5
35 U.S.C. § 103	3, 4
35 U.S.C. § 103(a).....	13
35 U.S.C. § 311	3

Regulations

37 C.F.R. § 42.10(b).....	2
37 C.F.R. § 42.100(b).....	6
37 C.F.R. § 42.104(a).....	4

37 C.F.R. § 42.104(b).....	4
37 C.F.R. § 42.24(d).....	75
37 C.F.R. § 42.63(e)	8
37 C.F.R. § 42.8(b)(1)	1
37 C.F.R. § 42.8(b)(2)	1
37 C.F.R. §§ 42.6(e)	77
37 C.F.R. §§ 42.8(b)(3)	1

PETITIONER’S EXHIBIT LIST

Exhibit	Description
1001	U.S. Patent No. 6,839,733
1002	File History of U.S. Patent No. 6,839,733—Application No. 9/712,712
1003	Declaration of Darrell Long, Ph.D
1004	U.S. Patent No. 6,029,175 (“Chow”)
1005	U.S. Patent No. 5,603,031 (“White”)
1006	U.S. Patent No. 6,263,358, (“Lee”)
1007	Complaint from <i>Byteweavr, LLC v. Databricks, Inc.</i> , No. 2:24-cv-00162-JRG-RSP (E.D. Tex.)
1008	F. Cheong, <i>Internet Agents</i> (1996)
1009	D.B. Lange, M. Oshima, G. Karjoth, K. Kosaka, “Aglets: Programming mobile agents in Java” (March 1997)
1010	S. Fünfroeken, “How to Integrate Mobile Agents into Web Servers” (June 1997)
1011	P. Dömel, A. Lingnau, O. Drobniak, “Mobile Agent Interaction in Heterogeneous Environments” (April 1997)
1012	Guideware Agent System for Java - Customer Support, October 17, 1997 Web Archive
1013	Guideware Agent System for Java – Agent Facts, October 17, 1997 Web Archive
1014	Guideware Agent System for Java – Sample Agents, October 17, 1997 Web Archive
1015	Y. Aridor, M. Oshima, “Infrastructure for Mobile Agents: Requirements and Design” (September 1998)
1016	ACM Transactions on Computer Systems (TOCS) 2.1 (1984): 39-59
1017	IPR2020-00470, Paper 1 (Petition)
1018	IPR2020-00470, Paper 7 (Patent Owner Preliminary Response)
1019	IPR2020-00470, Paper 13 (Institution Decision)

Exhibit	Description
1020	IPR2020-00470, Paper 18 (Patent Owner Response)
1021	Claim Construction Order, <i>Intellectual Ventures I, LLC et al. v. VMware, Inc.</i> , No. 1:19-cv-01075 (W.D. Tex) June 10, 2020, Dkt. 92.
1022	Joint Claim Construction and Prehearing Statement and Exhibits thereto in <i>Byteweavr, LLC v. Databricks, Inc.</i> , No. 2:24-cv-00162-JRG-RSP (E.D. Tex.) (dkt. entry 55)
1023	U.S. Patent No. 7,949,752
1024	U.S. Patent No. 6,163,794
1025	As-Filed Application of U.S. Patent No. 6,163,794
1026	Report and Recommendation, <i>Intellectual Ventures I, LLC et al. v. HCC Insurance Holdings, Inc. et al.</i> , No. 6:15-CV-660 (E.D. Tex)
1027	Docket Control Order in <i>Byteweavr, LLC v. Databricks, Inc.</i> , No. 2:24-cv-00162-JRG-RSP (E.D. Tex.)
1028	Docket in <i>Byteweavr, LLC v. Databricks, Inc.</i> , No. 2:24-cv-00162-JRG-RSP (E.D. Tex.) (retrieved March 13, 2025)

MANDATORY NOTICES

A. Real Parties-in-Interest [37 C.F.R. § 42.8(b)(1)]

The real party-in-interest is Petitioner Databricks, Inc. (“Petitioner” or “Databricks”). No entity not named above is funding, controlling, or directing this Petition, or otherwise has had an opportunity to control or direct this Petition or Petitioner’s participation in any resulting IPR.

B. Related Matters [37 C.F.R. § 42.8(b)(2)]

The ’733 patent that is the subject of this Petition is currently the subject of a patent litigation suit brought by Patent Owner Byteweaver, LLC, *Byteweavr, LLC v. Databricks, Inc.*, No. 2:24-cv-00162-JRG-RSP (E.D. Tex.) (Ex. 1007). The ’733 patent was previously asserted by Patent Owner Byteweavr against Cloudera, Inc., in *Byteweavr, LLC v. Cloudera, Inc.*, (Case No. 1:24-cv-00261 (W.D. Tex.)).

C. Lead and Back-Up Counsel [37 C.F.R. § 42.8(b)(3)]

Pursuant to 37 C.F.R. §§ 42.8(b)(3), 42.8(b)(4), and 42.10(a), Petitioner provide the following designation of counsel:

Lead Counsel for Petitioner	Back-Up Counsel for Petitioner
Robin Snader (Reg. No. 66,085) Erise IP, P.A. 7015 College Blvd., Ste. 700 Overland Park, KS 66211 Phone: 913-777-5600 Fax: Fax: 913-777-5601 Email: robin.snader@eriseip.com	Jennifer Bailey (Reg. No. 52,583) Erise IP, P.A. 7015 College Blvd., Ste. 700 Overland Park, KS 66211 Phone: 913-777-5600 Fax: Fax: 913-777-5601 Email: jennifer.bailey@eriseip.com

Lead Counsel for Petitioner	Back-Up Counsel for Petitioner
	Eric Buresh (Reg No. 50,394) Erise IP, P.A. 7015 College Blvd., Ste. 700 Overland Park, KS 66211 Phone: 913-777-5600 Fax: 913-777-5601 Email: eric.buresh@eriseip.com Kathleen Fitterling (Reg. No 62,950) Erise IP, P.A. 7015 College Blvd., Ste. 700 Overland Park, KS 66211 Phone: 913-777-5600 Fax: 913-777-5601 Email: kathleen.fitterling@eriseip.com

A Power of Attorney accompanies this Petition in accordance with 37 C.F.R. § 42.10(b).

D. Service Information [37 C.F.R. § 42.8(b)(4)]

Please address all correspondence to the lead and back-up counsel as shown above. Petitioners consent to electronic service by e-mail at the e-mail addresses provided above.

I. INTRODUCTION

Pursuant to 35 U.S.C. § 311, Petitioner hereby petitions for IPR of U.S. Patent No. 6,839,733 (Ex. 1001) (the “’733 Patent”) and seeks cancellation of claims 37–43 as unpatentable under 35 U.S.C. § 103. Petitioner further shows that a person of ordinary skill in the art would have been motivated to combine references, resulting in the claimed invention.

The ’733 Patent claims a method of admitting a user to a network system where an agent consumes service resources while performing tasks for the user and allowing the user to create, modify, or delete agents within the network system. Additional claims require commands to create, edit, or delete agents; creation, modification, or deletion of an agent template; and an agent area of a user interface.

This alleged advance was not new. Presented here are two prior art references that each render obvious the alleged invention of the ’733 Patent. While not acknowledged in the specification for the ’733 Patent, the prior art makes clear that software agents themselves acting withing a network system on a user’s behalf were well established before the priority date. That includes numerous software frameworks for developing user agents which would have been well known to one skilled in the art.

Petitioner respectfully submits that it is appropriate for the Board to institute IPR to invalidate this patent that never should have issued.

II. REQUIREMENTS OF *INTER PARTES* REVIEW

A. Standing

Pursuant to 37 C.F.R. § 42.104(a), Petitioner certifies that the '733 Patent is available for IPR and that Petitioner is not barred or estopped from requesting an IPR on the grounds identified herein.

B. Identification of Challenge and Relief Requested

Pursuant to 37 C.F.R. § 42.104(b), Petitioner requests that the Board institute IPR of claims 37–43 of the '733 Patent under AIA 35 U.S.C. § 103.

The '733 Patent issued from an application filed November 14, 2000, which claims priority as a continuation of an application filed October 23, 1998. For purposes of this Petition, Petitioner does not challenge whether the claims are entitled to that priority date.

The precise relief requested by Petitioner is that claims 37–43 be canceled based on the grounds below:

Ground	Claims	Basis for Rejection¹
1	37–43	Obviousness over Chow and White
2	37–43	Obviousness over Lee and White

¹ All obviousness grounds include the knowledge of a POSITA.

The above challenges are made in view of the following prior art references:

1. U.S. Patent No. 6,029,175 (“Chow”) (Ex. 1004)

U.S. Patent No. 6,029,175, titled “Automatic Retrieval of Changed Files by a Network Software Agent” issued on February 22, 2000, from a provisional application filed on October 26, 1995. The non-provisional application was filed on June 7, 1996. Chow qualifies as prior art to the ’733 Patent under pre-AIA 35 U.S.C. § 102(e). Chow was not cited or considered during prosecution of the ’733 Patent.

2. U.S. Patent No. 6,263,358 (“Lee”) (Ex. 1006)

U.S. Patent No. 6,263,358, titled “Scheduler for a software system having means for allocating tasks” issued on July 17, 2001, from parent application PCT/GB98/02237, filed July 27, 1998. Lee qualifies as prior art to the ’733 Patent under pre-AIA 35 U.S.C. § 102(e). Lee was not cited or considered during prosecution of the ’733 Patent.

3. U.S. Patent No. 5,603,031 (“White”) (Ex. 1005)

U.S. Patent No. 5,603,031, titled “System and Method for Distributed Computation Based Upon the Movement, Execution and Interaction of processes in a Network,” issued on February 11, 1997, from a non-provisional application filed on July 8, 1993. White therefore qualifies as prior art to the ’733 patent under pre-AIA 35 U.S.C. § 102(b). White was not applied in any Office Actions during prosecution of the ’733 patent or discussed by the Examiner or Applicant.

C. How the Challenged Claims Are To Be Construed Under 37 C.F.R. § 42.104(b)(3)

A claim subject to IPR shall be construed using the same standard as in a civil action. 37 C.F.R. § 42.100(b). The Board need only construe terms “to the extent necessary to resolve the controversy.” *Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co.*, 868 F.3d 1013, 1017 (Fed. Cir. 2017). As explained below, the Board has previously construed a claim term in a related patent common to the claims of the ’733 Patent. Without taking a position on the merits of the prior construction, and without waiving its right to seek a different construction in the corresponding district court litigation, this Petition applies the constructions from the prior Institution Decision.

Specifically, in a previously instituted IPR in *VMware, Inc. v. Intellectual Ventures I LLC*, IPR2020-00470 (“VMWare IPR”), the Board construed “agent” as “a process that occupies a place and that is mobile, i.e., can move from a first place to a second place.” IPR2020-00470, Paper 13 (Ex. 1019) at 13. That instituted IPR was of U.S. Pat. No. 7,949,752 (“the ’752 patent”) which is a continuation of, and shares the same specification as, the ’733 patent. The Board explained that this construction was based on lexicography because the ’752 patent incorporates the text of White, as does the ’733 Patent (Ex. 1001, 5:27–31), and White includes a glossary specifically defining “agent” as “a process that occupies a place and that is mobile, i.e., can move from a first place to a second place” (Ex. 1005, 16:65–67).

The Board also noted that “[t]his definition also is consistent with the embodiments described in the Specification of the ’752 patent.” IPR2020-00470, Paper 13 (Ex. 1019) at 13.

Petitioner therefore proposes that, for purposes of this IPR, “agent” be construed as “a process that occupies a place and that is mobile, i.e., can move from a first place to a second place.”

In the pending litigation against Petitioner, Patent Owner proposes that “agent” be construed as “a software application, program, or process responsible for performing a task on behalf of a user.” Ex. 1022. This Petition applies both constructions and the prior art invalidates under either construction.

D. How the Challenged Claims Are Unpatentable Under 37 C.F.R. § 42.104(b)(4)

An explanation of how the Challenged Claims are unpatentable under the statutory grounds identified above, including the identification of where each element of the claim is found in prior art patents or printed publications, is provided below.

E. Supporting Evidence Under 37 C.F.R. § 42.104(b)(5)

The exhibit numbers of the supporting evidence relied upon and the relevance of the evidence to the Challenged Claims, including an identification of specific portions of the evidence that support the challenge, are provided below. A list of

exhibits with the exhibit number and a brief description of each exhibit is also included in this Petition pursuant to 37 C.F.R. § 42.63(e). The technical information and grounds for rejection explained in this Petition are further supported by the Declaration of Darrell Long, Ph.D. (Ex. 1003).

III. PATENT OVERVIEW

The application which became the '733 Patent was filed Nov. 14, 2000, and claims priority to Oct. 23, 1998.

A. Alleged Invention of the '733 Patent

The '733 Patent described a problem with services such as email, voice mail, electronic organizers, and online databases. Ex. 1001, 1:48–52, 64–65. The problem was that the software applications supporting those services were “often developed with a broad spectrum of subscribers in mind.” *Id.*, 1:65–67. This meant the applications “address[ed] the generalized needs of many subscribers, but not the specialized needs of any one particular subscriber or group of subscribers.” *Id.*, 2:1–3. For a subscriber to obtain modified software to address the subscriber’s specialized needs, the subscriber would have to convince the service provider to bear the costs of developing, testing, and maintaining the modified software. *Id.*, 2:4–28.

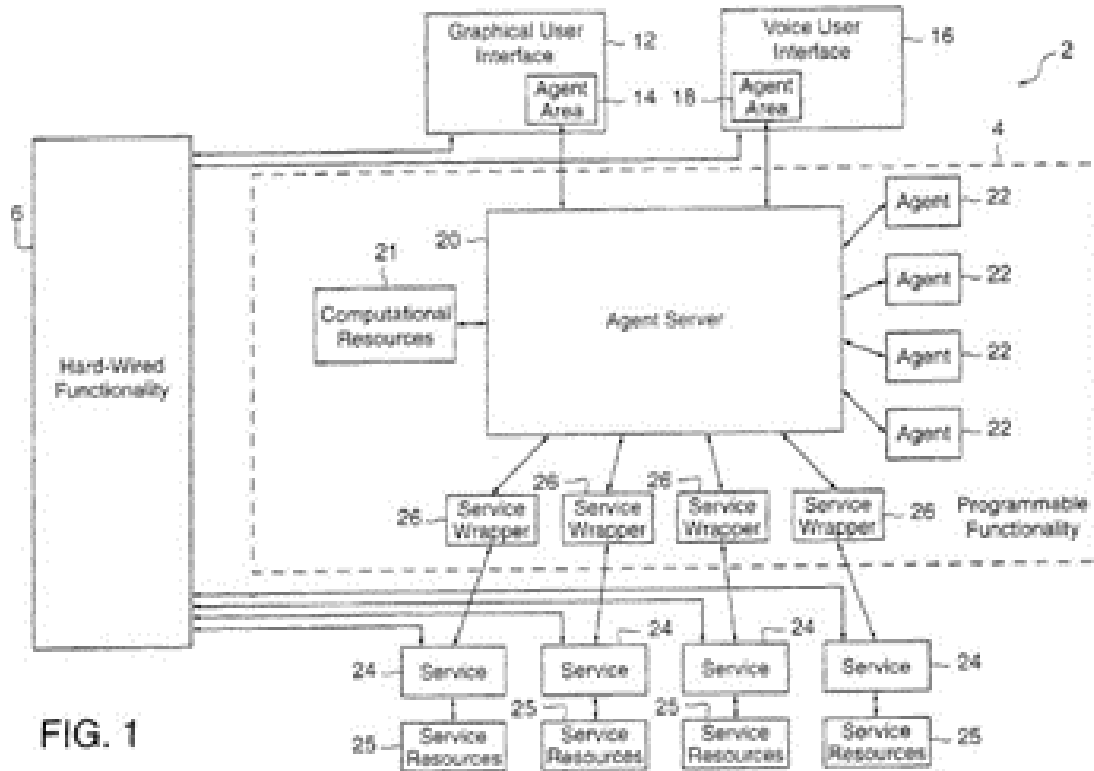
The solution discussed as the “present invention” in the summary section of the '733 Patent was to make the network system “extensible (e.g. programmable) by ‘end-users.’” Ex. 1001, 2:35–37. This was accomplished by augmenting the

network system with an agent system. *Id.*, 2:47–48. This augmentation was made possible by programmatically exposing capabilities of the network through services, service resources, and service wrappers, which software agents could use to modify the services. *Id.*, 2:49–58.

Reading these portions of the '733 Patent specification, one might get the impression that the applicants invented agent systems, or at the very least invented *extensible* agent systems. But, as discussed in the overview of Lee, this is not the case. See § IV.B.1. By October 1998, extensible network software agents were known in the art. See *id.* Numerous patents and publications addressed the functionality of agent systems prior to the filing of the '733 Patent. See *e.g.*, F. Cheong, *Internet Agents* (1996) (Ex. 1008); Long Decl. (Ex. 1003), ¶ 32-40. Lee reviews several pre-existing frameworks for agent development as background material. Lee (Ex. 1006), 1:34–46.

The disclosed embodiment in the '733 specification was implemented in the network system (2) of Figure 1 (reproduced below). Ex. 1001, 5:23–25. The network system had two main parts: “programmable functionality component” (4) and “hard-wired functionality component” (6). *Id.*, 5:56–57. The programmable functionality component comprised an agent system, including an agent server (20) and agents (22). *Id.*, 5:64–6:1, Figure 1. The network system also included graphical and voice user interfaces (12 and 16), through which users interacted with the

network system. *Id.*, 6:7–8, 18–21, 51–54. And, finally, the network system also included one or more services (24) with associated service wrappers (26) and service resources (25). *Id.*, 10:26–11:25, Figure 1.



The agents were “implemented as a software application, program, or process” which ran on behalf of an affiliated user or “principal.” Ex. 1001, 8:31–35, 37–40. Agents were “responsible for performing a particular task or set of tasks on behalf of the respective principal.” *Id.*, 8:62–64. Such tasks included using services (24). *Id.*, 10:26–28. Examples included an agent placing or receiving calls or voice mails with a service that supported call processing or an agent retrieving documents, such as electronic periodicals, from an on-line data retrieval service. *Id.*, 10:30–34, 44–49.

In performing tasks like these, agents consumed computational resources (21) and service resources (25). Ex. 1001, 9:37–45. Computational resources were “resources provided or supported by a computer-based system” and included “processing time, memory storage space, and the like.” *Id.*, 8:21–29. Service resources “enable[d] a service to be performed.” *Id.*, 11:2–3. For example, in an on-line data retrieval service, service resources could have “include[d] a telephony connection, a modem for on-line communication, an on-line database provider services, etc.” *Id.*, 11:9–12. Furthermore, some service resources might have “comprise[d] discrete units which [were] ‘consumed’ during utilization of the respective resource by an agent 22,” and which could be exhausted. *Id.*, 11:15–17, 25:57–60. For example, a service resource related to telephony services might have comprised units of long-distance calling time that were consumed and exhausted by placing a call. *Id.*, 11:17–21.

Agents executed on the agent server, which “control[led], coordinate[d], and otherwise manage[d] the overall operation of [the] programmable functionality component.” Ex. 1001, 7:55–57. The agent server could be used to “invoke, initiate, or execute various routines, processes, objects, and the like,” including agents, and also controlled the consumption of computational resources and service resources. *Id.*, 7:57–59, 7:66–8:3. The agent server also communicated (exchanged information) with the graphical and voice user interfaces. *Id.*, 7:51–55. Finally, the

agent server also interfaced with the services through their associated service wrappers, which “mediate[d] the interaction between a service ... and the remainder of [the] programmable functionality component.” *Id.*, 11:27–33.

Service wrappers “enable[d] communication between services 24 and agent server 20” by, for example, “converting between a computer language (or instruction set) used within agent server 20 and the computer language (or instruction set) of the respective service 24.” Ex. 1001, 11:47–51. Each service wrapper was “implemented, at least in part, with an application programming interface (API),” which programmatically exposed the capabilities of a service to an agent. *Id.*, 11:51–56.

“Agents 22 are created from agent templates.” Ex. 1001, 10:10. The ’733 specification explains the role of templates.

An agent template can be a “blueprint” for one or more agents 22. In object-oriented terminology, each agent 22 is an object and each template is an agent class from which agents 22 can be created by instantiating the class/template. In one embodiment, an agent template is essentially an agent object in its initial state. After the agent has been created, it may be executed. During execution, the agent object begins to receive events, handle these events, and so change its state.

In accordance with an embodiment of the present invention, network system 2 allows users (e.g., subscribers) to create, copy, modify, edit, or delete agents 22 and the associated templates as desired, thereby affording extensibility.

Id., 10:10–24. Templates can thus be either models, or “blueprints,” from which agents are created, or templates may be programmatic instantiations of the code defining the agent. Once an agent is created from a template and executing, its operation causes it to “change its state.” *Id.*, 10:17–18.

To manage agents, the “network system includes a user interface which allows a user to interact with the network system.” *Id.*, 3:1–3. An agent server with a user interface may “create or modify an agent in response to interaction by the user.” *Id.*, 3:5–8.

As explained below, however, the elements of this claim (as well as all others claimed) were previously disclosed, are not novel or innovative, and are obvious in view of each of Chow and Lee with White.

B. Prosecution History of the ’733 Patent

The application that issued as the ’733 Patent—Application No. 9/712,712 (“the ’712 application”)—purports to claim priority back to U.S. Pat. Appl. No. 9/178,366 (“the ’366 application”), filed Oct. 23, 1998, now U.S. Pat. No. 6,163,794, entitled “Network System Extensible By Users.” The ’712 application was filed on November 14, 2000.

The first office action on December 29, 2003, rejected some claims based on obviousness-type double patenting and separate claims as unpatentable under 35 U.S.C. § 103(a) over U.S. Pat. No. 6,657,990 (“Dilip”). Non-Final Office Action at

3 (Dec. 29, 2003) (Ex. 1002 at 433). According to the examiner, Dilip “taught the invention substantially as claimed including a system which provided network based interaction for allowing an agent application to consume a service resource while utilizing a service.” *Id.* In response, the applicant canceled all claims rejected based on Dilip and filed a terminal disclaimer to overcome the obviousness-type double patenting objection of the remaining claims. Applicant Arguments/Remarks Made in an Amendment at 10–11 (Apr. 28, 2004) (Ex. 1002 at 449–50). The examiner issued a notice of allowance on July 15, 2004.

C. Person of Ordinary Skill in the Art

The '733 Patent relates to the field of “computer software systems and, more particularly, to a network system extensible by users.” Ex. 1001, 1:33–35. A person of ordinary skill in the art (or “POSITA”) would have been someone with a bachelor’s degree in electrical engineering, computer science, or computer engineering (or a like discipline) and at least two years of experience in networked computer software systems. Long Decl. (Ex. 1003), ¶ 19-23. This level of skill in the art is reflected by the references cited in this Petition, the state of the art, and the experience of Professor Darrell Long as described in his declaration. In this Petition, references to a POSITA refers to a person with these or similar qualifications.

D. Prior Instituted IPR on the '752 Patent

The '752 patent was the subject of a prior instituted IPR that ultimately terminated prior to issuance of a final written decision. On February 10, 2020, Petitioner VMware, Inc. petitioned for IPR of claims 1–4, 6, 9–11, 13–14, and 22–26 of the '752 patent. At the time, the '752 patent was assigned to Intellectual Ventures I LLC (“IV”). VMware presented alternate grounds based on two different potential constructions of “agent”—one in which “agent” is not limited to mobile agents and another in which “agent” is construed as a mobile agent according to the lexicography in White. Ex. 1017, 16–17. The invalidity grounds under the “mobile agent” construction were based on Chow and White—a ground also presented in the present Petition. *Id.*, 56–60.

On May 20, 2023, IV filed a Preliminary Response, first arguing that “agent” must be construed according to the lexicography in White as a *mobile* agent, and thus the grounds based on the broader construction of agent were insufficient. Patent Owner Preliminary Response, IPR2020-00470 (Ex. 1018), 19. IV then argued, under the narrower interpretation of “agent,” that VMware did not show that a POSITA would have been motivated to combine Chow and White, or how it would have been appropriate to modify Chow to include the mobile agent of White. *Id.*, 20–25. Finally, IV also argued that VMware had failed to demonstrate how the art

teaches that “the service resource is exhausted upon being consumed by the network-based agent.” *Id.*, 25–26.

The Board rejected IV’s arguments on motivation to combine and “service resource,” and instituted IPR of the ’752 patent. Ex. 1019, 2. The IPR was terminated due to settlement prior to the issuance of a final written decision.

IV. GROUNDS OF UNPATENTABILITY

A. Overview of White as Prior Art

White is incorporated by reference into the ’733 patent as disclosing “[a]n exemplary construction for an agent system.” Ex. 1001, 5:23–31. White includes a glossary that defines “agent” as “a process which occupies a place and which is mobile, i.e. can move from a first place to a second place.” White (Ex. 1005), 16:65–67.

White discloses mobile agents as an alternative to the prior art. Specifically, White explains that, in the prior art, “[m]any computer communication systems implement what is called ‘remote procedure calling.’” *Id.*, 2:66–67. According to White, “[i]n remote procedure calling, a client process, which is executing on a first computer system and which seeks to manipulate data on a second computer system, requests action through a server process, which is executing on the second computer system and which manipulates data in response to requests from the client process.” *Id.*, 3:1–6. The drawback of remote procedure calling, however, is that the second

computer system can only “perform[] any of an inevitably finite list of operations on behalf of the client process,” and it may often be the case that the “precise function requested by the client process is not among the particular operations performed by the server processes.” *Id.*, 3:6–10.

White thus teaches an alternative to the prior art “remote procedure calling” called “remote programming.” *Id.*, 4:22–31. In remote programming, a first process on a client computer, sends a list of instructions to a second process on a server computer which then executes the list of instructions. *Id.* An advantage of remote programming is that it requires fewer uses of network communication media and also allows client processes to use server operations not explicitly provided for by the developers of the server processes. *Id.*, 5:17–25. Remote programming relies on the movement of agents “from a first place process in a first computer system to a second place process.” *Id.*, 9:26–29; *see also id.*, 23:50–52 (“Agents of the present invention therefore represent a significant improvement over the known prior art processes.”).

White also explains that its remote programming is an improvement over existing remote programming processes. *Id.*, 5:26–39. For example, White acknowledges two prior attempts to implement remote programming: Wolfson and Tsichritzis. *Id.*, 5:41–6:67. But White criticizes Wolfson and Tsichritzis for their inefficiencies and lack of sufficient security measures. *Id.*, 5:51–6:2, 6:41–7:11.

White's improved remote programming comprises "agent" and "place" classes. *Id.*, 8:27–29. The agent class could be used to form an "agent process" (or "agent") that could "(i) examine and modify itself, (ii) transport itself from a first place process ... in the network to a second place process in the network, and (iii) interact with other agents found at the second place process." *Id.*, 8:29–38. This allowed an agent to "travel from a first computer system to a second computer system." *Id.*, 8:40–41. Thus, White teaches a mobile agent system (remote programming) that solves the shortcomings of the prior art. *See generally id.*, 8:51–13:22.

B. Ground 1: Chow in View of White Renders Obvious Claims 37-43

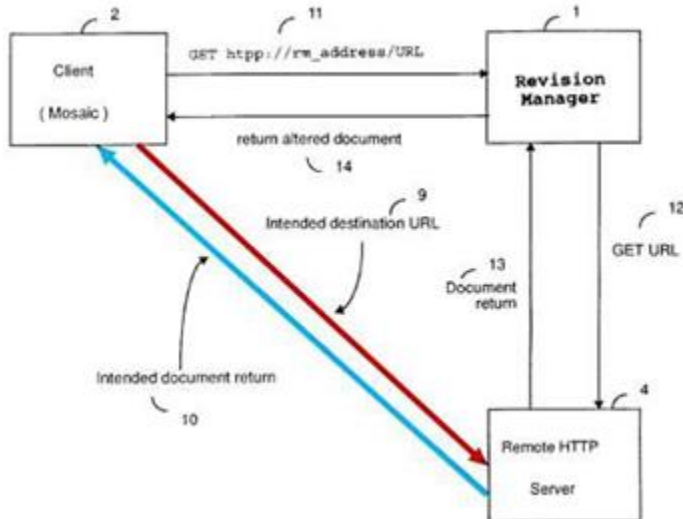
Chow in view of White discloses each and every element of claims 37–43. Long Decl. (Ex. 1003), ¶¶97-152. As such, Chow in view of White renders obvious those claims to a POSITA in view of the knowledge of such a person in this art. *Id.*

1. Overview of Grounds and Prior Art

Chow recognized a problem with the display of webpages. The problem was that after a client browser downloaded a webpage there was no way for a web server to inform the client browser when there was an updated version of the webpage available for download. Chow (Ex. 1004), 3:48–53. This was because web servers were "stateless and connectionless, so it [was] impossible for servers to maintain clients' states." *Id.*

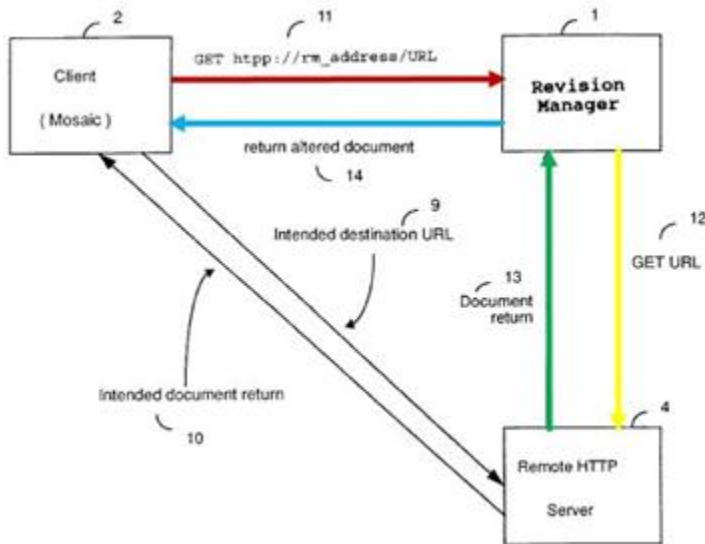
Chow's solution to this problem was "to provide a software agent for automatically retrieving changed documents previously accessed from network and internetwork servers." *Id.*, 3:60–63. Chow christened this intelligent network agent as the "Revision Manager." *Id.*, 3:63–64.

Figure 3 demonstrated how to obtain a webpage both with and without Chow's Revision Manager. Without the Revision Manager, it was a two-step process. Step one (rn. 9): A client sends a request to an HTTP server for a document identified by a URL. Step two (rn. 10): The HTTP server responds by sending the document. *Id.*, 9:52–58. In the annotated version of Figure 3 below, step one is shown in red and step two is shown in blue.



Id., FIG. 3 (annotated with colored lines). But, *with* the Revision Manager, it was a four-step process. Step one (rn. 11): The client sends the request to the Revision Manager. Step two (rn. 12): The Revision Manager forwards the request

to the HTTP server. Step three (rn. 13): The HTTP server provides the requested webpage to the Revision Manager. Step four (rn. 14): The Revision Manager returns an altered version of the requested webpage to the client. *Id.*, 9:59–10:30. In the annotated version of Figure 3 below, step one is shown in red, step two is shown in yellow, step three is shown in green, and step four is shown in blue.



Id., FIG. 3 (annotated with colored lines).

Upon returning the altered webpage, the Revision Manager would activate a “polling daemon” process which checked whether the target webpage was updated and whether the client was due for an update. See Chow (Ex. 1004), 17:44–20:64; Figs. 18–22. The Revision Manager acted as an agent for the user in monitoring and relaying changes to the web resources identified by the user. See Chow (Ex. 1004), 5:32–36. The user could both create new agents for a new “object of interest,” modify the polling frequency for an existing “object of interest,” or delete references

to an external “object of interest” once no longer needed. *See id.*; *see also id.*, 7:16–21, 11:3–6. Chow disclosed exemplary embodiments of user interfaces for a user to create or modify these agents and parameters as shown in figures 5 and 6.

301

Enter the URL of the documents you wish to check:

Enter the CCI port number you have selected on your browser:

FIG. 5

Chow (Ex. 1004), FIG 5. “FIG. 5 is a form that is filled in by a user to ***specify a document of interest to be monitored*** by the Revision Manager.” *Id.*, 7:16–18 (emphasis added).

302

☐ Alert me on source update for: <http://www.tekknowledge.com/>

Check for souce update every seconds

Enter the URL of the documents you wish to check:

Submit Info

Clear Form

FIG. 6

Chow (Ex. 1004), FIG 6. “FIG. 6 is a form that is attached to Revision Manager monitored documents and *filled in by the user to specify an updated interval.*” *Id.* 7:19–21 (emphasis added).

Chow’s Revision Manager also considered whether there were enough resources to access the requested webpage. For example, the Revision Manager could consider the monetary cost of retrieving the document “(such as communication charges, server charges, and copyright holder charges)” and only obtain the document “[i]f the cost would not exceed an acceptable level.” *Id.*, 25:65–26:5.

In addition to monetary considerations, performing tasks for the user consumed cache resources and computational resources. The Resource Manager provided “caching of objects of interest,” *id.*, 5:50–61, and each “Revision Manager

[could] serve a group of parties ... with a shared cache of pertinent objects,” *id.*, 4:44–59. A cache would consume disk space to store the “objects of interest.” Long Decl. (Ex. 1003), ¶ 108. Besides consuming cache resources, the executing software of the Revision Manager inherently consumed computational resources including processing time, memory storage space, and other computer resources. Long Decl. (Ex. 1003), ¶ 109.

Thus, Chow discloses the purportedly novel elements of the ’733 Patent.

2. A POSITA Would Have Been Motivated to Combine Chow and White

Prior to the filing of the ’733 patent, a POSITA would have recognized that Chow implements remote procedure calling, and a POSITA would have been motivated to modify Chow’s remote procedure calling with White’s improved remote programming. Long Decl. (Ex. 1003), ¶¶ 106-125. As discussed below, this modification would have resulted in significant benefits as discussed in White. *Id.*

Specifically, Chow implements remote procedure calling. For example, in Chow, the “polling daemon” process of the Revision Manager, which determines whether the web page hosted by a remote web server had been updated, is a “remote procedure calling” process. Long Decl. (Ex. 1003), ¶¶ 113, 119 (citing Chow (Ex. 1004), 17:44–20:64, Figs. 18–22). As a remote procedure calling process, the polling daemon is limited to performing a “finite list of operations.” White (Ex.

1005), 3:6–10; Long Decl. (Ex. 1003), ¶ 119. Moreover, remote procedure calling processes can require “substantial use of communications media which in turn involves considerable expense and time.” White (Ex. 1005), 3:6–15.

To overcome these drawbacks to remote procedure calling, a POSITA would have been motivated to modify Chow to implement the “remote programming” process using mobile agents as taught in White. *See id.*, 5:17–25. Indeed, given the teaching in White that its remote programming was preferable to remote procedure calling (as well as preferable to other remote programming implementations), a POSITA would have been motivated to modify the remote procedure calling agent process in Chow with the remote programming agent process in White. Long Decl. (Ex. 1003), ¶ 120. This combination would be the simple substitution of one known technique (remote programming) for another (remote procedure calling) and would have obtained predictable results. *Id.* ¶ 121. Because White solved some of the issues with prior art remote programming systems, as well as improved remote programming systems themselves, a POSITA would have preferred to use White’s remote programming implementation over Chow’s more limited remote procedure calling. *Id.* For example, White identifies several benefits of its remote programming, which could be achieved in the system of Chow:

Remote programming improves upon remote procedure calling by (i) enabling processes to interact without communicating across network communications media and (ii) improving the

performance of the interactions between processes by
eliminating communication across network communications
media which often has high-latency.

White (Ex. 1005), 9:19–25. In other words, the potential improvements described in White—remote programming using mobile agents instead of remote procedure calling using stationary agents—would have motivated a POSITA to combine the references. Long Decl. (Ex. 1003), ¶¶ 119-121.

For example, as explained above, Chow’s “polling daemon” is limited to performing a finite list of operations. Long Decl. (Ex. 1003), ¶ 119. A POSITA would have been motivated to expand the capabilities of the polling daemon by allowing remote programming of new capabilities, as taught in White. *Id.* ¶ 121. The expanded capabilities of the polling daemon would have included performing additional operations on documents of interest. *Id.*

In its Institution Decision in the *VMware* IPR, the Board rejected the then-patent-owner’s argument that “Petitioner fails to adequately support the combination of Chow and White.” Ex. 1019, 31–32. Petitioner here identifies the same motivations as included in that prior petition. Thus, as the Board explained in the Institution Decision, “the potential improvements stated by White reasonably would have motivated the combination of Chow and White.” *Id.*, 32.

Additionally, a POSITA would have understood that Chow expressly contemplates, or at the very least strongly suggests, use of mobile agent and/or

mobile code technology. Ex. 1003, ¶¶ 112, 122; Ex. 1004, 26:51-60 (“[A] client could have interest in a financial document, and could send to the Revision Manager a computer program that would monitor the price of certain stocks and bonds, and notify the client of the price trends for the last week when the price of any one of these stocks and bonds would change by more than five percent over a weekly interval.”), 22:13-24. Chow also suggests that it may be desirable to “further reduce latency and communication costs” and a POSITA would have recognized that implementing mobile agents in Chow’s revision management system would do just that. Ex. 1003, ¶ 124 (citing Ex. 1004, 22:13-24). A POSITA would have found it obvious and been motivated to implement mobile agent technology in Chow’s Revision Manager considering Chow’s disclosures. Ex. 1003, ¶¶ 122, 124.

A POSITA would also have been motivated to take advantage of the well-known benefits of mobile agents used in the same or similar contexts. Ex. 1003, ¶ 123; Ex. 1009, 11 ((1) “The client can search and monitor the Web even while it is disconnected, and monitoring of Web sites does not require a continuous network connection” and (2) “For Web sites, it means that interested readers are automatically and instantaneously notified of important updates.”); Ex. 1011, 5 (“The mobility of agents can help to do this [search the Internet, monitor changes of Web pages, etc.] more efficiently by avoiding network traffic whenever great amounts of data have to be processed.”). Indeed, use of mobile agent technology,

including for use with Web servers, for information retrieval, and even in the specific context of website monitoring, was well-known to a POSITA by October 1998. Ex. 1003, ¶¶ 123-125, 33-40 (citing, in part, Ex. 1009, Ex. 1011, Ex. 1010). And, a POSITA would have known that implementing mobile agent technology on a web server did not require the code of the Web servers to change. Ex. 1003, ¶¶ 123, 35 (citing Ex. 1010). Accordingly, a POSITA would have had a reasonable expectation of success in combining Chow and White. Ex. 1003, ¶ 123.

For these reasons, a POSITA would have been motivated to modify Chow to include mobile agents as taught in White. Long Decl. (Ex. 1003), ¶¶ 118-125.

3. Detailed Analysis of Ground 1

Chow in view of White discloses each and every element of claims 37–43. Long Decl. (Ex. 1003), ¶¶ 97-152.

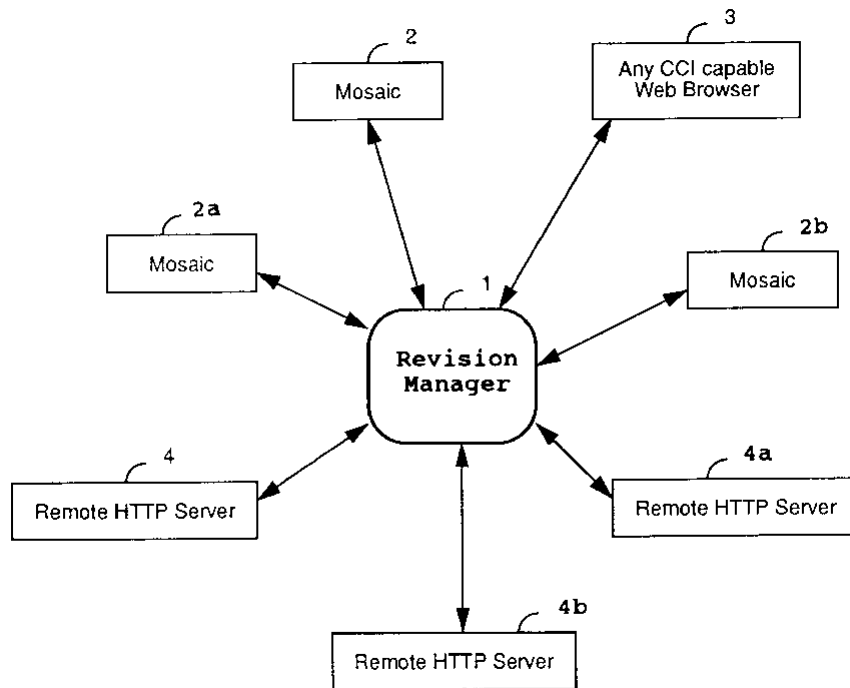
a. Element 37[pre] A method comprising:

To the extent the preamble is limiting, Chow discloses a method. Chow (Ex. 1004) at Abstract (“The invention provides a method”).

b. Element 37.a[1] admitting a user to a network system

Chow discloses this limitation. Long Decl. (Ex. 1003) ¶¶ 99-104. The invention disclosed in Chow provides “a software agent for automatically retrieving

changed documents previously accessed from network and internetwork servers. Such a software agent will be referred to below as a ‘Revision Manager.’” Chow (Ex. 1004), 3:60–64. This “Revision Manager” functions “as an intermediary between a number of Mosaic browsers 2, 2a, 2b, or any other CCI capable web browser 3, and a number of remote HTTP servers 4, 4a, 4b.” Chow (Ex. 1004), 9:32–36. The combination of web browsers and web servers are a network system. Long Decl. (Ex. 1003), ¶ 101.



Chow (Ex. 1004), FIG. 1. Figure 1 shows that the Revision Manager is the intermediary in this network system and the “Revision Manager is viewed by the client as a kind of server that sends these documents to the client’s browser for viewing by the user.” Chow (Ex. 1004), 4:3–5. The Revision Manager operates as

an agent for, and intermediates the user's access to, the network. *See id.* In order to do so (and in order to manage the user's consumption of resources, *see infra*), the Revision Manager maintains a directory of clients being serviced by the Revision Manager (i.e., clients of the network system managed by the revision manager). Ex. 1004, 26:26-29. When the user is added to this directory (*see, e.g., id* at 27:2-3), the client is admitted to the network managed (intermediated) by the Revision Manager. Ex. 1003, at ¶¶ 102-103.

Thus, Chow discloses “admitting a user to a network system” within the meaning of the '733 Patent.

**c. Element 37.a[2]
wherein at least one agent is operable to consume a
service resource while utilizing a service to perform a
task for the user; and**

Chow in view of White discloses this limitation. Long Decl. (Ex. 1003) ¶ 105-125. The “Revision Manager is viewed by the client as a kind of server that sends these documents to the client's browser for viewing by the user.” Chow (Ex. 1004), 4:3–5. Furthermore, “the Revision Manager is located close to multiple users so that their common information requirements can be served quickly from a shared local cache.” *Id.*, 4:26–28. Thus “the Revision Manager augments the typical WWW browser with capabilities for accessing a shared cache of automatically updated documents and for responding appropriately and automatically to the change in

information within a previously viewed document,” all of which are tasks done by the Revision Manager on behalf of the user. *See id.*, 4:34–39. Chow’s Revision Manager is a software application, program, or process responsible for automatically retrieving changed documents for a user and is therefore an “agent” under Patent Owner’s proposed co-pending litigation construction. *See, e.g.*, Ex. 1004, at Abstract, 3:60–64, 1:13-15; Long Decl. (Ex. 1003) ¶ 111.

Chow in view of White teaches that the Revision Manager program (the network-based agent under Patent Owner’s proposed co-pending litigation construction) retrieves documents and webpages from a source (an online service) and would only retrieve the documents or webpages if the monetary costs (consumed units of a service resource) associated with that retrieval did not exceed a “certain acceptable cost.” Chow (Ex. 1004), 25:65–26:5, 29:60–30:6. A POSITA would have understood that a cost (monetary or otherwise) constitutes a resource which enables a service to be performed and is a service resource within the meaning of the ’733 patent. Long Decl. (Ex. 1003) ¶¶ 59-62; Ex. 1001, 26:14-44; Ex. 1024, 26:11-43; Ex. 1025 at pg. 91 (claim 16). A POSITA would have thus understood that Chow teaches using a service (*e.g.*, an on-line database) and a service resource (*e.g.*, monetary costs) configured to be consumed by the network-based agent (Revision Manager). Long Decl. (Ex. 1003) ¶ 107.

In the *VMware* IPR, over the prior patent owner's objections, the Board found that "Chow . . . describes the exhaustion of resources upon the consumption of those resources when a cost is paid to obtain an object, e.g., a document." Ex. 1019, 35. For example, the Board credited the prior petitioner's argument that "Chow describes a 'cost associated with retrieving the document from the source (such as communication charges, server charges, and copyright holder charges),' " and further explained that "Chow subsequently describes 'obtain[ing] the object,' i.e., document, 'from [a] source having the minimum value of [a] cost-benefit function." *Id.* Because the Board concluded that Chow teaches "exhaustion of resources upon the consumption of those resources," Chow similarly must teach that the resources are "consumed." *Id.*

Patent Owner's infringement theories in the related litigation assert that computational resources qualify as the claimed service resources. Ex. 1007 at 7 (describing service resources as including "CPU and memory"). To the extent that Patent Owner contends that "service resources," as used in the challenged claims, can include *computational* resources such as CPU and memory, Petitioner reserves the right to argue that the computational resources disclosed in Chow would disclose the consumption of a service resource in addition to the additionally disclosed service resources consumed. Long Decl. (Ex. 1003) ¶ 109-110.

In particular, in the course of providing services on behalf of the user, the Revision Manager manages and consumes other resources than those already described. One described “embodiment of the Revision Manager” provided “caching of objects of interest” and “redirection of [] request[s] to the best supplier of the requested resource, if a resource of sufficient quality is not already present in the cache.” Chow (Ex. 1004), 5:50–61. Each “Revision Manager can serve a group of parties ... with a shared cache of pertinent objects.” *Id.*, 4:44–59. A POSITA would understand that a cache is a limited resource the Revision Manager agent would consume in the course of providing services to clients. Long Decl. (Ex. 1003), ¶ 108.

Still further, in addition to the cache, the Revision Manager is agent software running on a computer server and consuming resources. Chow (Ex. 1004), 3:60–64 (the “Revision Manager” is “a software agent for automatically retrieving changed documents previously accessed from network and internetwork servers”), 9:40–44 (“the program for the Revision Manager could be executed by ... the same digital computer that executes the program for a server.”). Chow goes on to describe how users can upload programs to monitor information on their behalf:

For example, a client could have interest in a financial document, and could send to the Revision Manager a computer program that would monitor the price of certain stocks and bonds, and notify the client of the price trends for the last week when the price of any one of these stocks and bonds would change by more than five percent over a weekly interval.

Chow (Ex. 1004), 26:54-60. A POSITA would understand that, in the Chow-White combination, this program would be the mobile agent of White, as discussed below. Ex. 1003, ¶¶ 112, 122. As software running on a computer, the agent would inherently consume computational resources including processing time, memory storage space, and other computer resources. Long Decl. (Ex. 1003), ¶ 109; *see also* Ex. 1001, 8:25–27 (“Computational resources 21 include processing time, memory storage space, and the like.”).

Finally, in prior district court litigation for a related patent, “consume” was construed to mean “used,” and “service resource” was construed to mean “a resource that enables a service to be performed.” Ex. 1026 29-31; 26-28. A POSITA would understand that Chow teaches that the agent is “operable to consume a service resource” under this construction as well, at least since the agent (i.e. the Revisions Manager) uses resources (such as the cache) that enable a service (e.g., the network-based functionality) to be performed. Ex. 1003, ¶¶ 107-108.

As explained above, White discloses mobile agents. White (Ex. 1005), 16:65–67. Indeed, White (which is incorporated by reference in the ’733 patent) is the source of Petitioner’s construction of “agent.” Accordingly, in the Chow-White system, White’s agents would consume these resources in a similar or identical fashion to that described above with respect to Chow’s system. Therefore, a POSITA would have understood that Chow in view of White discloses “at least one agent is

operable to consume a service resource while utilizing a service to perform a task for the user” within the meaning of the ’733 Patent. Long Decl. (Ex. 1003), ¶ 105-125. For the reasons discussed in Section IV.B.2, a POSITA would have found it obvious and been motivated to modify Chow to include mobile agents as taught by White.

**d. Element 37.b
allowing the user to create, modify, or delete the agent
within the network system.**

Chow in view of White discloses this limitation. Long Decl. (Ex. 1003) ¶¶ 126-130. “The Revision Manager accepts a user input indicating that the user has some interest in monitoring a particular document.” Chow (Ex. 1004), 5:32–34. In addition, Chow discloses a form “filled in by a user to specify a document of interest to be monitored by the Revision Manager.” *Id.*, 7:16–18, FIG. 5. Other forms disclosed allow “the user to specify an updated interval.” *Id.*, 7:19–21, FIG. 6. The user designating a document for monitoring by a Revision Manager or updating the update interval for an existing designated document would be examples of, respectively, creating and modifying the Revision Manager Agent. Long Decl. (Ex. 1003), ¶ 128. Chow also discloses that when “a client is deleted from the client directory, then the other data structures associated with the client are also deleted or de-allocated from memory.” Chow (Ex. 1004), 27:3–6. Such a deletion occurs when the “client browser of an interested party has exited.” *Id.*, 11:3–6. As described

above, Chow also describes how “a client ... could send to the Revision Manager a computer program that would monitor the price of certain stocks and bonds, and notify the client of the price trends.” *Id.* at 26:54-60. Chow expressly gives this program (which would, as discussed above, be implemented as White’s mobile agent in the Chow-White combination) as an example of information stored in the user’s data structures. *Id.* at 26:49-54. The deletion of the cache and other data structures associated with a user in the Revision Manager would thus include the deletion of the user’s agent within the network system. Long Decl. (Ex. 1003), ¶ 128. Chow explicitly discusses the execution flow of the Revision Manager when “an object directory update is requested by the client” which results in “execution branches ... to add or delete the object to or from the object directory.” *See* Chow (Ex. 1004), 26:61–27:34.

In addition, a POSITA would have found it obvious and been motivated to implement the combined Chow-White system to allow users to create, modify, or delete remote agents as was well-known at the time. Long Decl. (Ex. 1003), ¶¶ 129, 33-34, 36 (citing Ex. 1009 and 1011).

Thus, Chow in view of White discloses “allowing the user to create, modify, or delete the agent within the network system” within the meaning of the ’733 Patent.

e. Claim 38
The method of claim 37 further comprising accepting an agent command from the user.

Chow in view of White discloses this limitation as more specifically required by Claim 39, discussed below. Thus, Chow discloses “accepting an agent command from the user” (e.g., via standard web protocol commands such as GET and POST) within the meaning of the ’733 Patent. Chow (Ex. 1004), 9:59–65.

f. Claim 39
The method of claim 38 wherein the agent command is one of a create agent command, an edit agent command, and a delete agent command.

Chow in view of White discloses this limitation. Long Decl. (Ex. 1003) ¶ #. “The Revision Manager accepts a user input indicating that the user has some interest in monitoring a particular document.” Chow (Ex. 1004), 5:32–34. Chow’s embodiments use standard web server commands from clients, including the “GET command” and the “POST command.” Chow (Ex. 1004), 9:59–65.

301

Enter the URL of the documents you wish to check:

Enter the CCI port number you have selected on your browser:

Submit Info Clear Form

FIG. 5

Chow (Ex. 1004), FIG 5. For example, the user can fill out and submit a form (using the HTTP POST command) to monitor a URL: “FIG. 5 is a form that is filled in by a user to specify a document of interest to be monitored by the Revision Manager.” *Id.*, 7:16–18. “To automatically be provided with updates to a document of interest ... the browser client sends a GET command 11 to request a HTML document ‘rm.html’ from a Revision Manager 1. The user enters a URL by filling in a form ‘rm.html’ (301 shown in FIG. 5). When the user submits the form, the client 2 sends a POST command to the Daemon (6 in FIG. 4) of the Revision Manager 1.” *Id.*, 9:59–65. When the Revision Manager receives the POST command, one of the steps taken is to set the “update_interval” variable which defines “how often the user wishes to have the document checked for an update” by the Revision Manager acting

as the user's agent. Chow (Ex. 1004), 11:55–12:18. As described above, Chow additionally describes how, as a part of this process, the user can upload their own computer program for the Revision Manager to run (which, in the Chow-White combination, would be creating a new agent to perform this monitoring on the user's behalf; see Ex. 1003, ¶ 134). The POST command supplied with such a computer program is the claimed "create agent command." *Id.*

Next, Chow describes how a user can edit an agent previously created by adding additional URLs to be monitored by that agent:

The Revision Manager responds with a confirmation message "Current document now registered for update: <http://www.teknowledge.com/HIBURST/>." The Revision Manager also keeps displaying "Enter the URL of the document desired:" followed by a blank space so that the user may retrieve new documents as well.

Ex. 1004, 22:1-6; *see also* FIG. 29 *infra* (showing the confirmation message for creating the agent (green box) and the interface for submitting an update command (blue box):

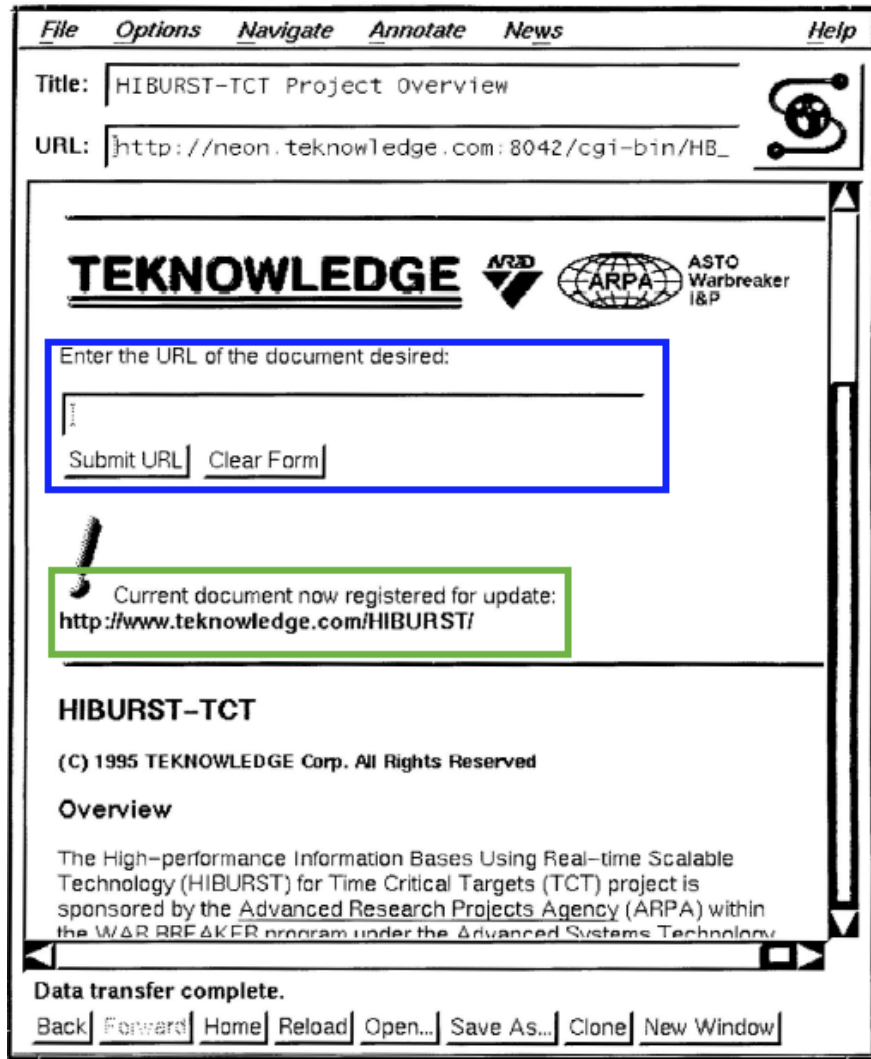


FIG. 29

Finally, in the Chow-White combination, a user canceling registration causes deletion of the agent. Long Decl. (Ex. 1003), ¶ 136. As described above, when a user cancels their registration, the data structures associated with that user are deleted, and these data structure include the program (mobile agents in the Chow-White system) uploaded by the user. Ex. 1004, 27:3–6; 26:49-54. In Chow’s embodiments, “there are two ways to cancel a registration: one is through the short form provided

on the returned document, and the other is when the user terminates the browser. Other policies and mechanisms could also be easily supported.” *Id.*, 6:29-32. In fact, Chow specifically claimed a “method of operating a server in a data network to service client requests” “wherein the server receives a request from said at least one of the clients to cancel registration for updates.” Chow (Ex. 1004), Claims 94, 96.

Thus, Chow discloses “wherein the agent command is one of a create agent command, an edit agent command, and a delete agent command” within the meaning of the ’733 Patent.

g. Claim 40
The method of claim 37 further comprising
suspending execution of the agent.

Chow in view of White discloses this limitation. Long Decl. (Ex. 1003) ¶¶ 138-140. Chow’s flowchart “of a client notification method” includes a check “whether a prior notification to the client for a change in the same object under consideration has been suspended.” Chow (Ex. 1004), 33:65–34:1. One step of Chow’s embodiment checks parameters which, when met, “execution branches to step 556 to suspend the client notification task until” other parameters are met. Chow (Ex. 1004), 34:32–35.

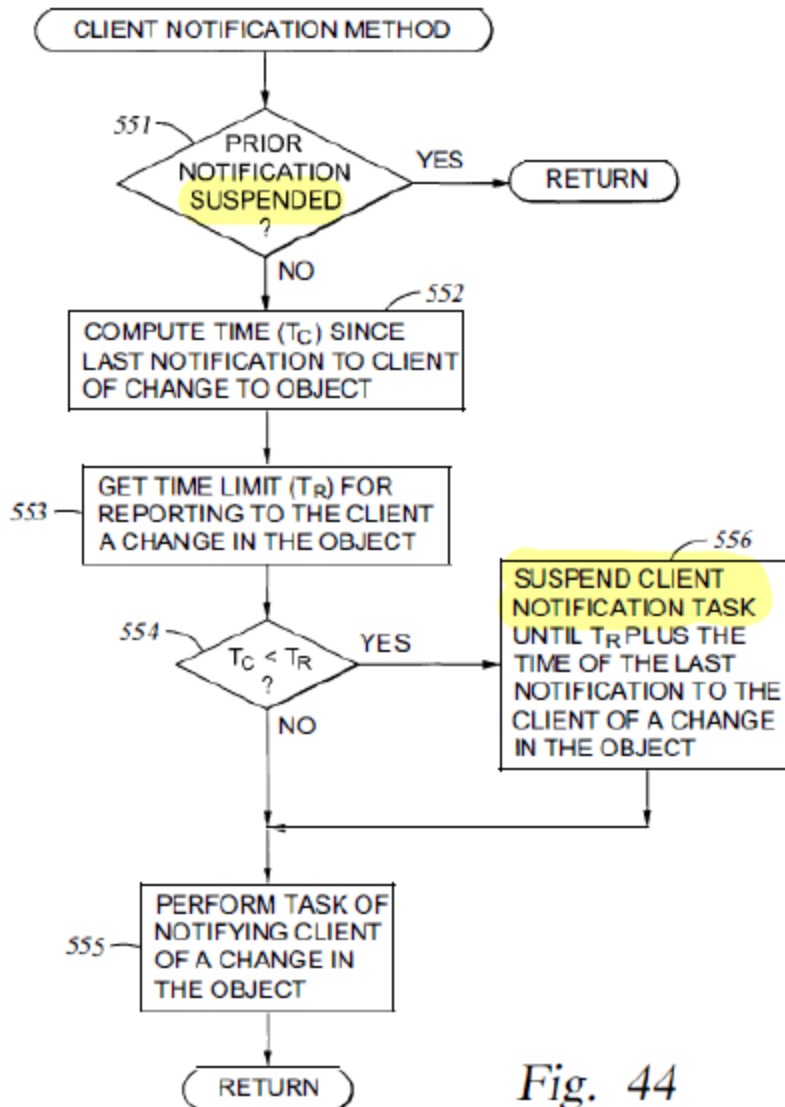


Fig. 44

Chow (Ex. 1004), FIG. 44 (emphasis added). Based on the description and the figures in Chow, a POSITA would find suspending execution of the agent disclosed. Long Decl. (Ex. 1003), ¶ 139.

Thus, Chow discloses “suspending execution of the agent” within the meaning of the ’733 Patent.

h. Claim 41
**The method of claim 37 further comprising accepting
a template command from the user.**

Chow in view of White discloses this limitation. Long Decl. (Ex. 1003) ¶ 141-146. Chow discloses that the “Revision Manager may accept specifications for an HTML-formatted document.” Chow (Ex. 1004), 37:27–28. In the context of these HTML-formatted templates, Chow offers that “[i]n addition to using the links identifying related objects of interest, links could be used to propagate changes to related objects. For example, summary documents could be linked to supporting documents so that the summary information is updated automatically in response to changes in the supporting documents.” *Id.*, 37:42–47. The submission of a specification for “an HTML-formatted document” would be understood by a POSITA as the Revision Manager accepting a template command that could include details of the user customized source documents the user wanted to consolidate according to the provided template. Long Decl. (Ex. 1003), ¶ 142-143. Chow gives the example of “summary documents [that] could be linked to supporting documents so that the summary information is updated automatically in response to changes in the supporting documents. A specific example would be a financial statement that would change the total value of an investment portfolio in response to changes in the value of stocks and bonds in the portfolio.” Chow (Ex. 1004), 37:47–50.

White also teaches that its mobile agent “is capable of creating one or more clone processes of the agent and transporting each clone process to a respective place.” Ex. 1005, 25:30–35; *see also id.* 11:10–28. In the combined Chow-White system, a POSITA would have found it obvious and been motivated to accept a command from a user to clone an agent (i.e., a template command) to enable sending clone agents to several places simultaneously as stated in White, such as for monitoring changes in supporting/linked documents. *Id.*; Long Decl. (Ex. 1003), ¶ 145.

Thus, Chow discloses “accepting a template command from the user” within the meaning of the ’733 Patent.

i. Claim 42
The method of claim 41 further comprising allowing the user to create, modify, or delete an agent template within the network system.

Chow in view of White discloses this limitation. Long Decl. (Ex. 1003) ¶ 147–149. As discussed above for Claim 41, Chow discloses “accepting a template command from the user.” The user submission of “specifications for an HTML-formatted document” would create the agent template. Chow (Ex. 1004), 37:27–28; Long Decl. (Ex. 1003), ¶ 148. In the case of cloning, the agent template (parent agent to the clone(s)) would be created, modified, or deleted as any other agent in the system. *See* Element 37.b.

Thus, Chow discloses “allowing the user to create, modify, or delete an agent template within the network system” within the meaning of the ’733 Patent.

j. Claim 43
The method of claim 37 further comprising admitting the user to an agent area of a user interface.

Chow in view of White discloses this limitation. Long Decl. (Ex. 1003) ¶ 150-152. See the figures and discussion for Claim 39 above describing and reproducing the figures disclosed in Chow related to the agent area of a user interface. These include FIG. 5, “a form that is filled in by a user to specify a document of interest to be monitored by the Revision Manager,” Chow (Ex. 1004), 7:16–18, and FIG. 6, “a form that is attached to Revision Manager monitored documents and filled in by the user to specify an updated interval,” Chow (Ex. 1004), 7:19–21.

Thus, Chow discloses “admitting the user to an agent area of a user interface” within the meaning of the ’733 Patent.

C. Ground 2: Lee in View of White Renders Obvious Claims 37-43

Lee discloses each and every element of claims 37–43. Long Decl. (Ex. 1003), ¶¶ 153-199. To the extent Lee does not disclose the mobile nature of agents required by the ’733 patent’s incorporation by reference of White, Lee in view of White renders obvious those claims to a POSITA in view of the knowledge of such a person in this art. *Id.*

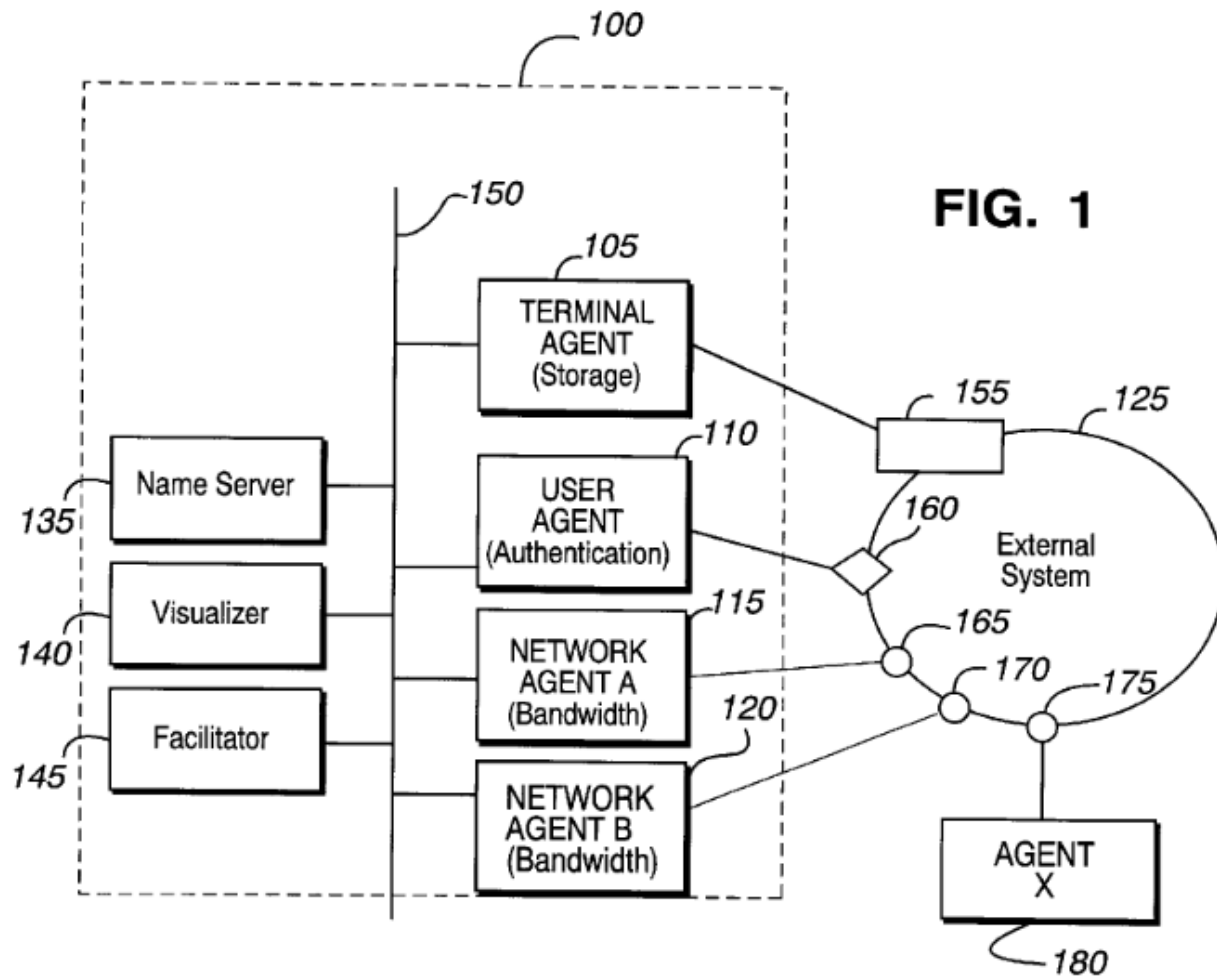
1. Overview of Grounds and Prior Art

Lee discloses details of an “agent building system tool-kit known as the Collaborative Agent Building System (‘CABS’).” Lee (Ex. 1006), 4:36–40. CABS employs computer program software agents which act as agents for a user, a piece of equipment, or a business. *Id.*, 1:8–10. Even predating the CABS system, Lee identified other software frameworks for building agents, including the following. *Id.*, 1:34–46.

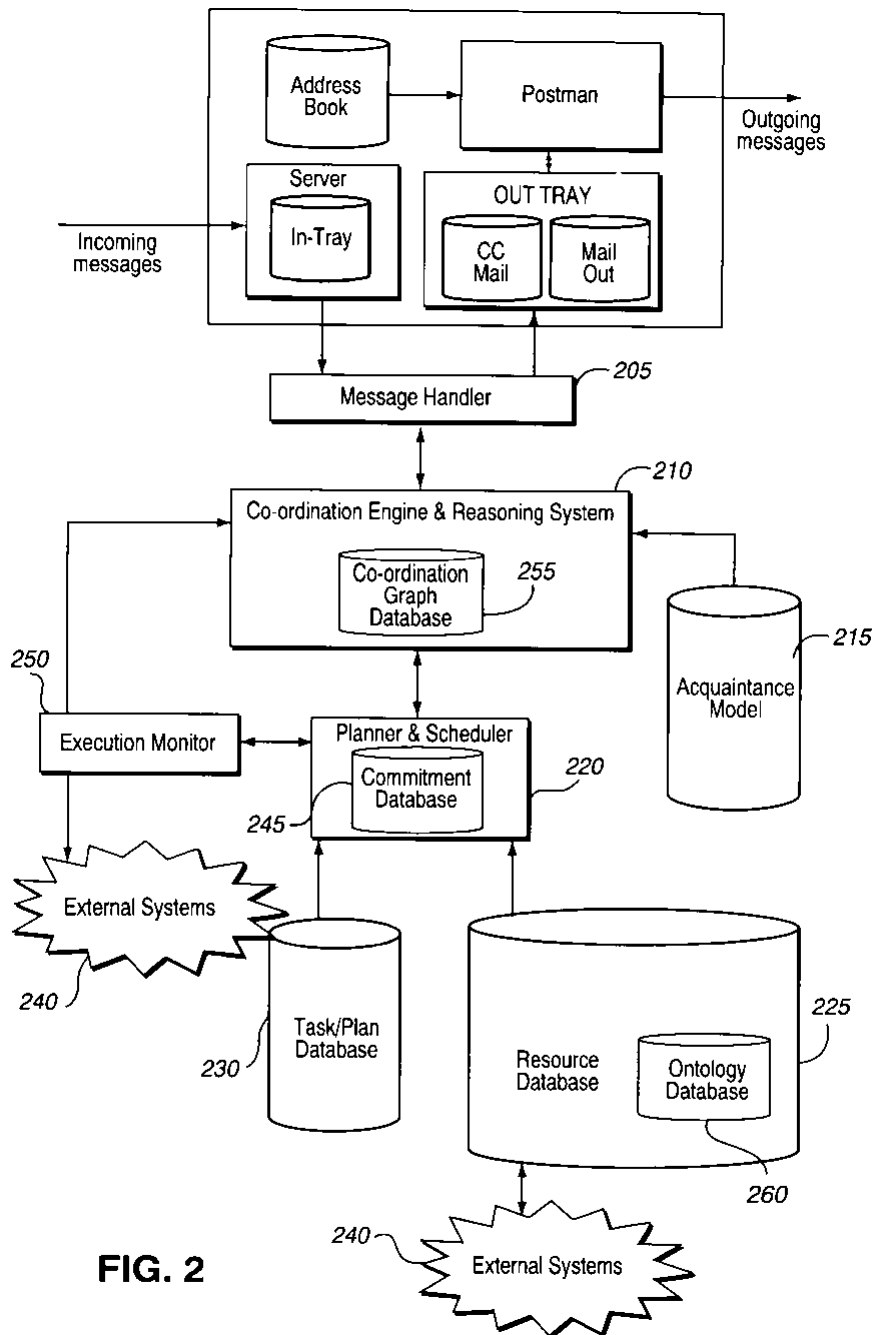
- “IBM’s Agent Building Environment (ABE) which is essentially a C++ class library []. ABE is a tool-kit that facilitates the construction of agent-based applications or helps add an agent to existing applications. This tool-kit applies to relatively trivial ‘interface’ agents, or agents that work alone.” *Id.*, 1:47–58.
- “MIT’s SODABOT [], General Magic’s Telescript and Odyssey [], and IBM’s Aglets []. These all provide other environments which facilitate the construction of ‘mobile’ agents-based applications. However, they ... do not provide specific advanced agent-building arrangements.” *Id.*, 1:59–67.

CABS provided “a software building environment, for building a software system for use in control, monitoring and/or management of a process or apparatus, ... and means for generating the software system” *Id.*, 2:35–42.

Agents built by the CABS system authenticated access to network resources and agents could “collaborate with each other” and with other systems. *Id.*, 6:34–44; FIG. 1.



Id., FIG. 1.



Id., FIG. 2. “FIG. 2 shows a schematic architecture for a software module constituting an agent for distributed control, monitoring and management of a system.” *Id.*, 4:49–50.

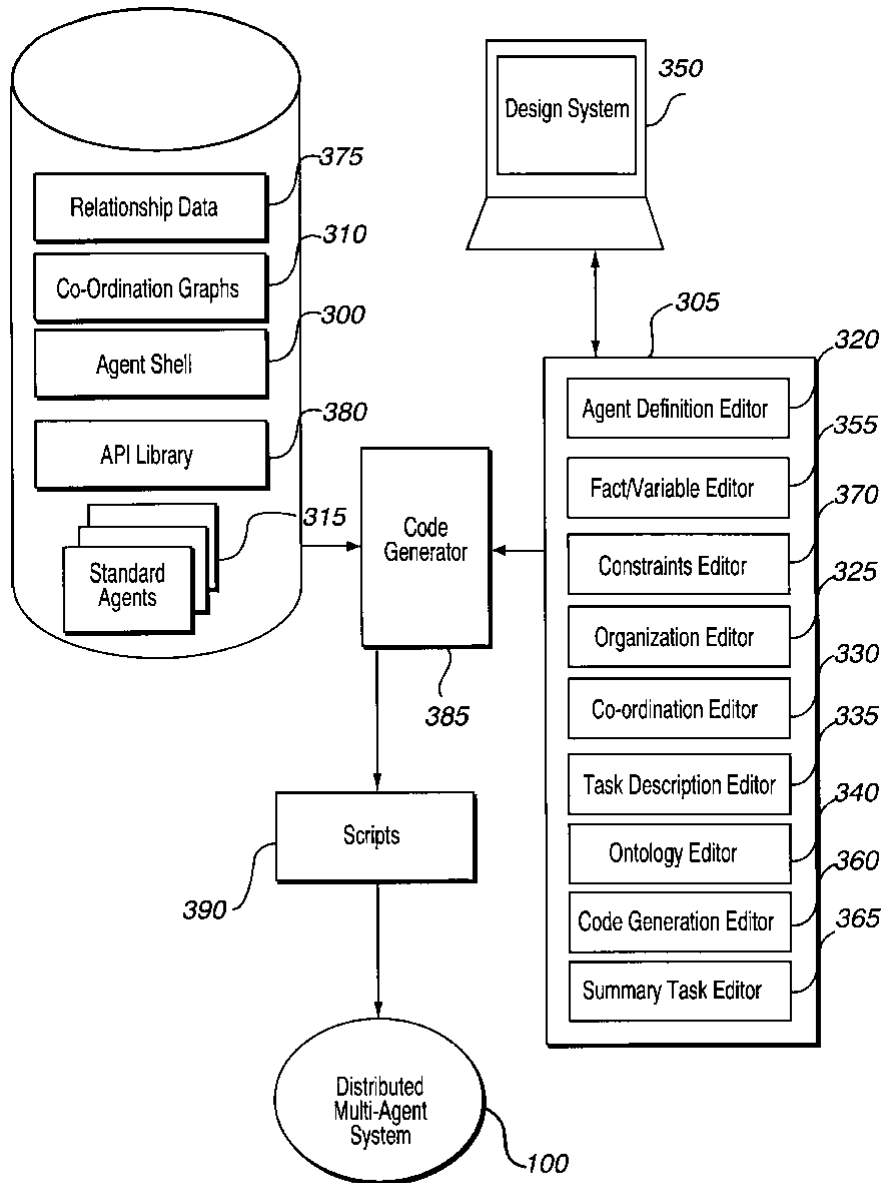


FIG. 3

Id., FIG. 3. “FIG. 3 shows a CABS platform for building an agent as shown in FIG. 2.” *Id.*, 4:47–48. “The agent shell 300 is simply the framework to support an agent structure as shown in FIG. 2.” *Id.*, 23:50–51. “Referring to FIG. 3, the means for ... automatically building a software module according to said architecture, is provided

by the CABS platform. This provides an agent template 300 which dictates the agent structure shown in FIG. 2.” *Id.*, 23:39–45.

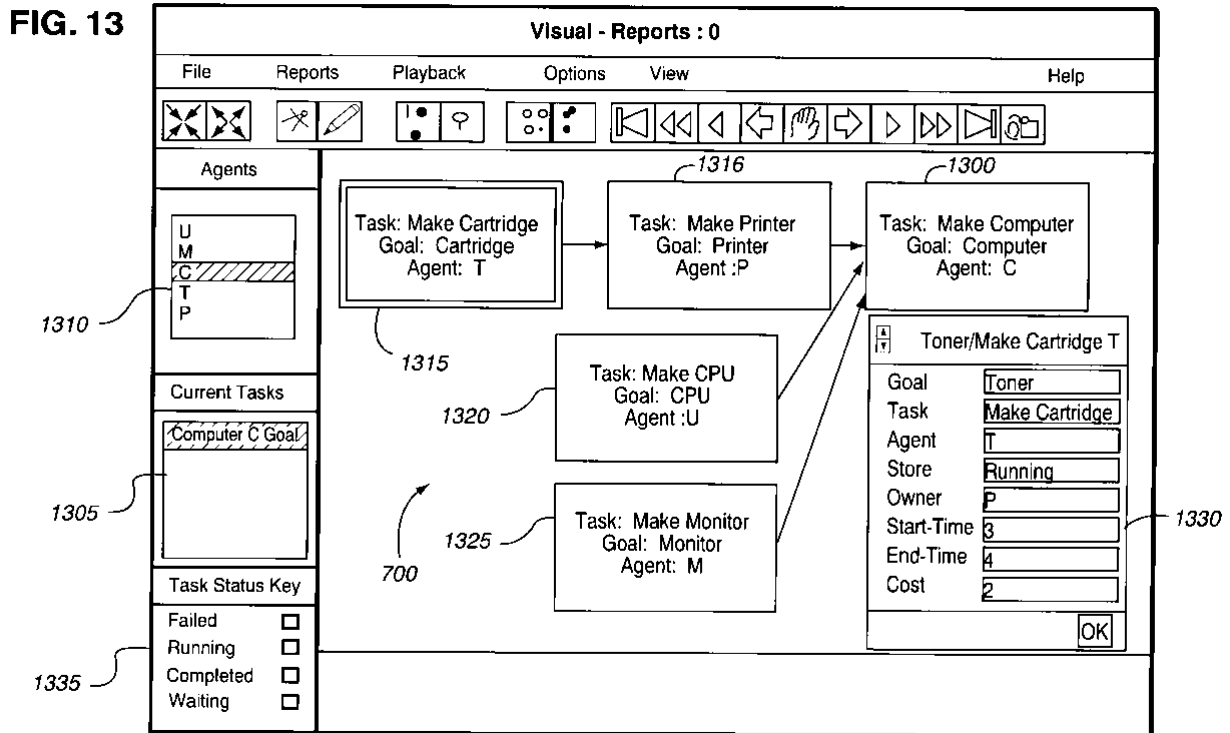
The CABS system provided “an agent definition editor” and other editors to define and manage agent functionality. *Id.*, 23:60–24:34, 2:50–67. The agents included functionality for managing “resources [] available to the agent” and a “mechanism for over-booking tasks.” *Id.*, 7:36–45, Abstract. The agent’s “[t]asks consume[d] certain resources” during execution and could adapt based on “constraints between and within the consumed and produced items” including defining “constraint[s] on the resource for [a] particular task, such as processing capacity or equipment type.” *Id.*, 29:56–30:4.

In addition to defining agents and their functionality, agents in the CABS system included “a planner and scheduler” which “plans and schedules the tasks the agent is controlling” and “an execution monitor 235 which starts, stops, and monitors external systems tasks scheduled for execution or termination by the planner and scheduler.” *Id.*, 7:29–54.

CABS also included a Control Monitor Tool, useful with debugging, with the functions of suspending, resuming, or killing agents “done by sending request messages to these agents.” *Id.*, 41:26–33. The Control Monitor Tool “also provided for the user to tune various parameters of an agent,” which would edit the agent. *Id.*, 41:42–44. Another debugging tool besides the Control Monitor Tool was the

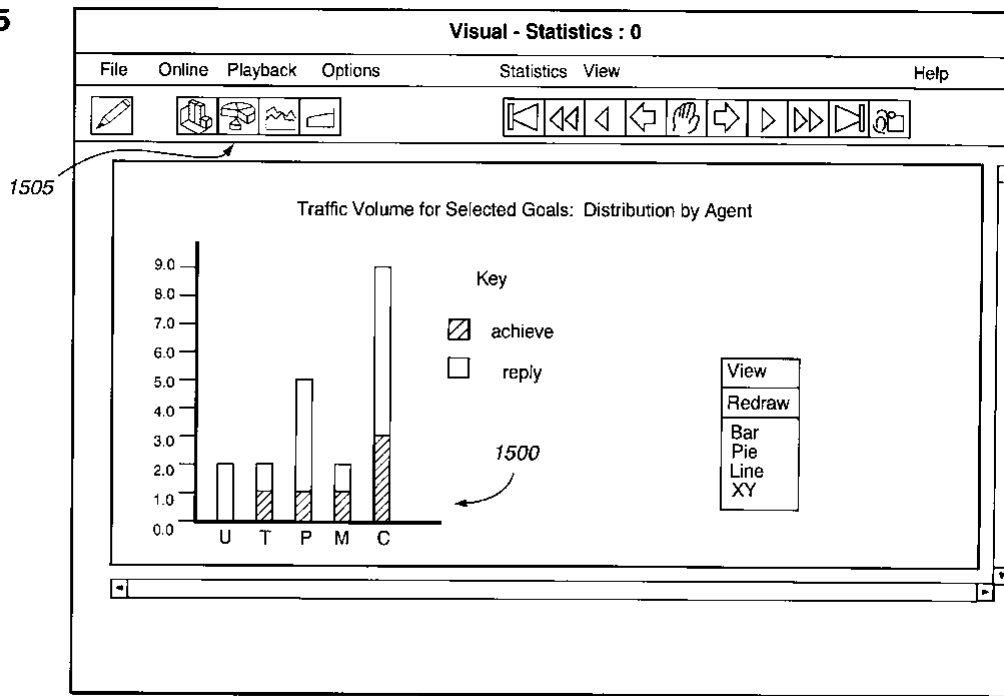
visualiser. *Id.*, 34:38–43. The visualiser could “modify an agent’s behaviour.” *Id.*, 35:9–11.

“The visualiser provides a generic, customisable and scaleable visualisation system for use with a domain-independent toolkit for constructing multi-agent applications.” *Id.*, 33:23–26. Figures 13 and 15 show exemplary agent areas of user interfaces disclosed in Lee.



Lee (Ex. 1006), FIG 13 (showing Agents U, M, C, T, and P). “FIG. 13 shows schematically a GANTT chart as an example of the screen-based output of a ‘reports tool’ of the visualisation system in use.” *Id.*, 5:1–3.

FIG. 15



Id., FIG. 15 (showing statistics for “Distribution by Agent”). “FIG. 15 shows schematically an example of the screen-based output of a ‘statistics tool’ of the visualisation system in use.” *Id.*, 5:7–8.

Thus, Lee discloses the purportedly novel elements of the ’733 Patent.

2. A POSITA Would Have Been Motivated to Combine Lee and White

Prior to the filing of the ’733 patent, a POSITA would have recognized that Lee discloses an agent-building environment where developers could “imbue [agents] with co-ordination abilities suitable for the problems the agents are being designated to tackle.” Lee (Ex. 1006), 2:50–59. Lee’s “CABS platform could [] be used for building almost any agent-based system where software agents need [] to

collaborate with other agents.” *Id.*, 5:18–22. Lee offered a system with the “functionality required for a collaborative agent” for “communications, reasoning, inter-agent co-operation and the ability to participate in goal-driven interactions between agents” that included an “asynchronous socket-based message passing system.” *Id.*, 5:42–51. Lee recognized that the advantages of mobile agent systems were desirable, but criticized popular mobile agent systems as being “not much more than languages.” Ex. 1006, 1:59-65.

White’s advanced mobile agent technology provides a robust architecture for the creation of mobile agents, and a POSITA would have recognized that White’s mobile agents overcome the disadvantages of existing mobile agent systems and would provide the benefits of mobility when combined with Lee’s agent-building environment. According to White, in “most network architectures, all processes are stationary and communicate by exchanging messages.” White (Ex. 1005), 127:43–45. In contrast, White taught that “some processes are stationary, but others are mobile, the former communicating by means of the latter. This is a paradigm shift.” *Id.*, 127:45–48. White thus expanded “the extensibility, generality and functionality of the set of computer instructions . . . which comprise an agent, how and to what destination the agent, and the information contained therein, travels.” *Id.*, 8:44–50.

Lee discusses the known coordination problems in multi-agent systems:

A particular problem arises with “collaborative” agents. Collaborative agents are a group of agents which co-operate with one another to co-

ordinate their activities in performing a particular task. Such co-operation may be necessary because know-how, resources or processing power may be distributed across the environment or across the agents in the system. The problem with collaborative agent systems is the need to co-ordinate their activities in a problem- and context-dependent manner.

Lee (Ex. 1006), 2:22–30. By “configur[ing] a number of different agents of varying functionality and behaviour,” Lee gave developers the means to organize agents in whatever manner they chose and “supply each agent with the necessary domain-specific problem-solving code.” *Id.*, 24:38–48.

The CABS framework developed by Lee, “allow[ed] the system developer to concentrate on domain-specific analysis without having to expend effort on agent-related issues.” *Id.*, 24:53–55. White offered functionality benefits, including reduced network communication and also allowing client processes to use server operations not explicitly provided for by the developers of the server processes. White (Ex. 1005), 5:17–25. White also offers more flexibility to the “problem- and context-dependent” dependencies of inter-agent communications Lee recognized by “allow[ing] client processes to use server operations not explicitly provided for by the developers of the server processes.” White thus enhances the flexibility of Lee’s framework. Long Decl. (Ex. 1003), ¶ 171.

While discussing the prior art environments of SODABOT, General Magic’s Telescript and Odyssey, and IBM’s Aglets, Lee disparaged them because “they are

also not much more than languages, comparable to the “Java” language developed by Sun Microsystems Inc., and do not provide specific advanced agent-building arrangements.” Lee (Ex. 1006), 1:64–67. Lee did acknowledge, however, that they “facilitate the construction of ‘mobile’ agents-based applications.” *Id.*, 1:62–64.

A POSITA would have found it obvious to improve the communication and interaction between agents in Lee by employing the improved paradigm disclosed in White. Long Decl. (Ex. 1003), ¶ 170, 173-174. Lee proposed embodiments “implemented in the Java programming language, allowing it to run on a variety of hardware platforms. Java is a good choice of language for developing multi-agent applications because it is object-oriented and multi-threaded, and each agent may consist of many objects and several threads. It also has the advantage of being portable across operating systems, as well as providing a rich set of class libraries that include excellent network communication facilities.” Lee (Ex. 1006), 5:27–36. White was directed to “an improved distributed computing environment wherein processes are created in an object oriented environment and direct their own movement throughout a computer network.” White (Ex. 1005), 1:39–43. Because Lee sought to give developers a “collaborative agent building environment[]” where “system developers can define a set of agents to work together in a system [and] organise them in relation to one another in whatever manner they choose,” a POSITA would be motivated to combine the benefits of mobile agents disclosed in

White within the agent building framework disclosed by Lee. Long Decl. (Ex. 1003), ¶ 173.

For these reasons, a POSITA would have been motivated to modify Lee to include mobile agents as taught in White. *Id.*, ¶ 168-174.

3. Detailed Analysis of Ground 2

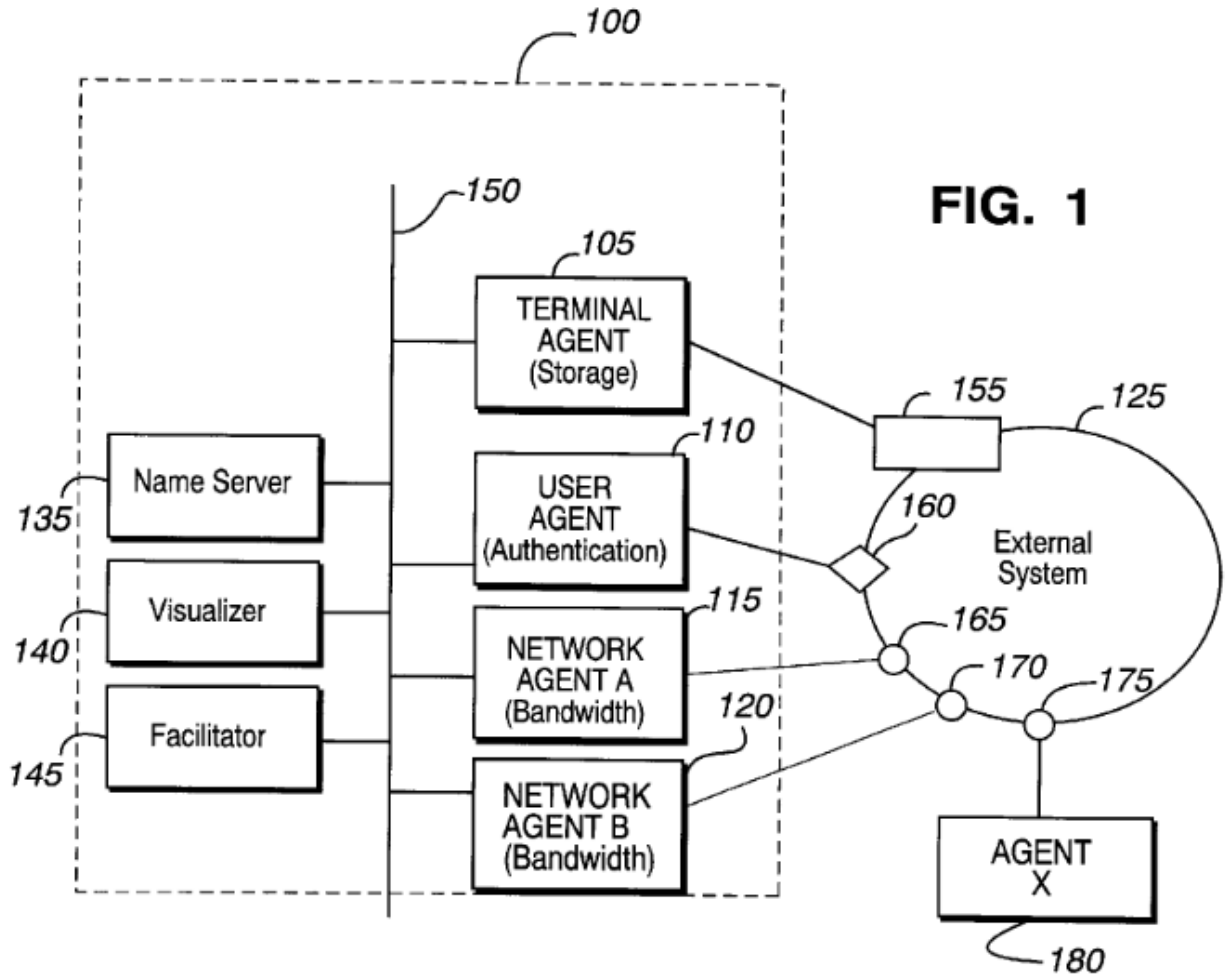
Lee in view of White discloses each and every element of claims 37–43. Long Decl. (Ex. 1003), ¶¶153-199. Because, as discussed above, a POSITA would be motivated to combine Lee and White, Lee in view of White renders obvious those claims to a POSITA in view of the knowledge of such a person in this art. *Id.*

a. Element 37[pre] A method comprising:

To the extent the preamble is limiting, Lee discloses a method. Lee (Ex. 1006) at 24:39 (“CABS embodies a system of methods....”). Lee discloses an “agent building system tool-kit known as the Collaborative Agent Building System (‘CABS’)” as an exemplary embodiment of Lee’s invention. *Id.*, 4:36–37.

b. Element 37.a[1] admitting a user to a network system

Lee discloses this limitation. Long Decl. (Ex. 1003), ¶ 155-160. “The system shown in FIG. 1 [of Lee] is an example of an agent-based system for use in communications.” Lee (Ex. 1006) 5:17–18.



Id., FIG. 1.

This exemplary system in Lee discloses “providing an authentication resource” to work with “network agents” that “collaborate with each other ... to provide the network bandwidth.” *Id.*, 6:34–44. “The user agent 110 must launch the authentication application 160” as part of “the agent system 100” communicating with the components of a communications system 125. *Id.*, 6:21–22, 34–40. For example, various agents of Lee’s network control access to a network link. *See, e.g., id.*, 6:60–61 (describing “separate agent 180, controlling a network link 175 of the

external system 125”). This, responsive to communication with the user agent 110, the network agent 180 admits the user (and the agents operating on the user’s behalf) to external system 125. Similarly, the user agent 110 communicates with network agents 115 and 120 to admit the user and the user’s agents to local area network 150.

Thus, Lee discloses “admitting a user to a network system” within the meaning of the ’733 Patent.

**c. Element 37.a[2]
wherein at least one agent is operable to consume a
service resource while utilizing a service to perform a
task for the user; and**

Lee in view of White discloses this limitation. Long Decl. (Ex. 1003) ¶ 161-174. Lee’s abstract describes “collaborative software agents work[ing] together ... [including a] scheduler [] built into each collaborative agent ... [with] a mechanism for over-booking tasks for any one agent.” Lee (Ex. 1006), Abstract. The collaborative agents described by Lee “need both to collaborate together and to carry out tasks. The tasks will include those directly involved in providing services to a user but may also include auxiliary tasks.” *Id.*, 6:29–33.

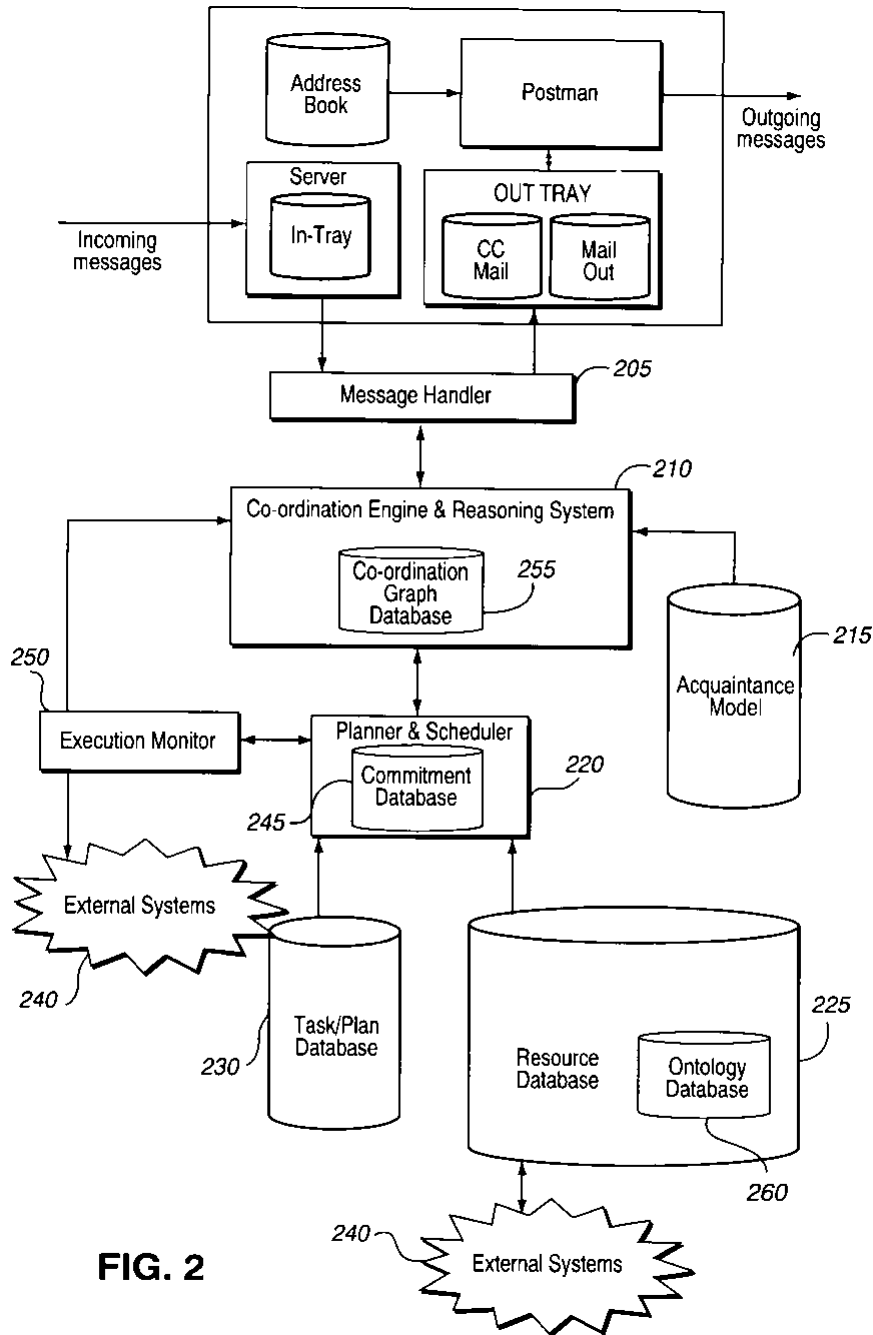


FIG. 2

Id., FIG. 2. “FIG. 2 shows a schematic architecture for a software module constituting an agent for distributed control, monitoring and management of a system.” *Id.*, 4:49–50.

Within the agent depicted in FIG. 2 is “a resource database 225 containing logical descriptions of the resources currently available to the agent; and providing an interface between the database and external systems such that the database can query external systems about the availability of resources and inform external systems when resources are no longer needed by the agent, and external systems can on their own initiative add, delete or modify resource items in the database, thus initiating changes in the agent’s behaviour.” Furthermore, according to Lee, “[t]asks consume certain resources to produce their products.” *Id.*, 29:56–57. For example, Lee gives the example of an agent that requires a particular resource to provide a video link between London and Ipswich. *Id.*, 20:67-21:3. Lee goes on to identify the required resources to provide this video link: a “fibre-optic” connection with speed of “1000.” *Id.*, 21:13-14. Such an “Internet connection” is expressly given by the ‘733 Patent as an example of a “service resource.” Ex. 1001, 11:12-14; Long Decl. (Ex. 1003), ¶ 164.

Patent Owner’s infringement theories in the related litigation assert that computational resources qualify as the claimed service resources. Ex. 1007 at 7 (describing service resources as including “CPU and memory”). To the extent that Patent Owner contends that “service resources,” as used in the challenged claims, can include *computational* resources such as CPU and memory, Petitioner reserves the right to argue that the computational resources disclosed in Lee would disclose

the consumption of a service resource in addition to the additionally disclosed service resources consumed. Long Decl. (Ex. 1003) ¶ 165.

First, Lee expressly contemplates computational resources as being managed similarly to other resources like the internet connection described above: “a task may call on a particular resource but there may be a constraint on the resource for that particular task, such as **processing capacity** or equipment type.” Ex. 1004, 29:63-66 (emphasis added). Second, a POSITA would recognize that Lee’s agents, as software running on a computer, would inherently consume computational resources including processing time, memory storage space, and other computer resources. Long Decl. (Ex. 1003), ¶ 165; *see also* 29:31-55; 30:50-53 (resource cost used by the agent to perform a task is part of the definition of an agent).

Thus, Lee discloses “at least one agent is operable to consume a service resource while utilizing a service to perform a task for the user” within the meaning of the ’733 Patent.

**d. Element 37.b
allowing the user to create, modify, or delete the agent
within the network system.**

Lee in view of White discloses this limitation. Long Decl. (Ex. 1003) ¶ 175-177. Lee discloses a “Control Monitor Tool” that can explicitly modify or delete agents within the system. Lee (Ex. 1006), 41:18–51.

The control monitor tool allows a user to select an agent of interest and browse-through, add, modify or delete its current jobs, resources, tasks, organisational relations and knowledge, and coordination strategies. Further, the user can suspend or resume jobs, and also suspend, resume or even kill agents. A facility is also provided for the user to tune various parameters of an agent

Lee (Ex. 1006), 41:37–46.

Thus, Lee discloses “allowing the user to create, modify, or delete the agent within the network system” within the meaning of the ’733 Patent.

e. Claim 38
The method of claim 37 further comprising accepting an agent command from the user.

Lee in view of White discloses this limitation. Long Decl. (Ex. 1003) ¶ 178-180. In the context of the Control Monitor Tool, the functions of suspending, resuming, or killing agents “are done by sending request messages to these agents with the contents being the above (e.g. a suspension notice).” *Id.*, 41:26–33. Sending request messages to agents would inherently require the agents accept these commands to perform the required function. Long Decl. (Ex. 1003), ¶ 178. Lee explicitly discloses that “the user can suspend or resume jobs, and also suspend, resume or even kill agents,” all of which would inherently require the user issue a command to perform the requested function. *See* Lee (Ex. 1006), 41:40-42; Long Decl. (Ex. 1003), ¶ 178. Indeed, Lee expressly provides a list of functions of the command tool, including “Browse-through,” “Add,” “Modify,” “Delete,”

“Suspend,” “[R]esume,” and “Kill.” *Id.*, 41:20-30. A POSITA would understand that these agent commands from the user must be accepted by the system in order to provide the described functions. Long Decl. (Ex. 1003), ¶ 178.

Thus, Lee discloses “accepting an agent command from the user” within the meaning of the ’733 Patent.

f. Claim 39

The method of claim 38 wherein the agent command is one of a create agent command, an edit agent command, and a delete agent command.

Lee in view of White discloses this limitation. Long Decl. (Ex. 1003) ¶ 181-182. As mentioned for Claim 38, the Control Monitor Tool has functions, including killing agents, “done by sending request messages to these agents.” Lee (Ex. 1006), 41:26–33. “The control monitor tool allows a user to select an agent of interest and browse-through, add, modify or delete its current jobs, resources, tasks, organizational relations and knowledge, and coordination strategies. Further, the user can suspend or resume jobs, and also suspend, resume or even kill agents.” *Id.*, 41:37–42. As described above, Lee expressly states that the Control Monitor tool provides for “Add,” “Modify,” and “Delete” agent functions (corresponding to the claimed “create agent,” “edit agent,” and “delete agent” commands, respectively). *Id.*, 41:20-24; *see also* Long Decl. (Ex. 1003), ¶ 181.

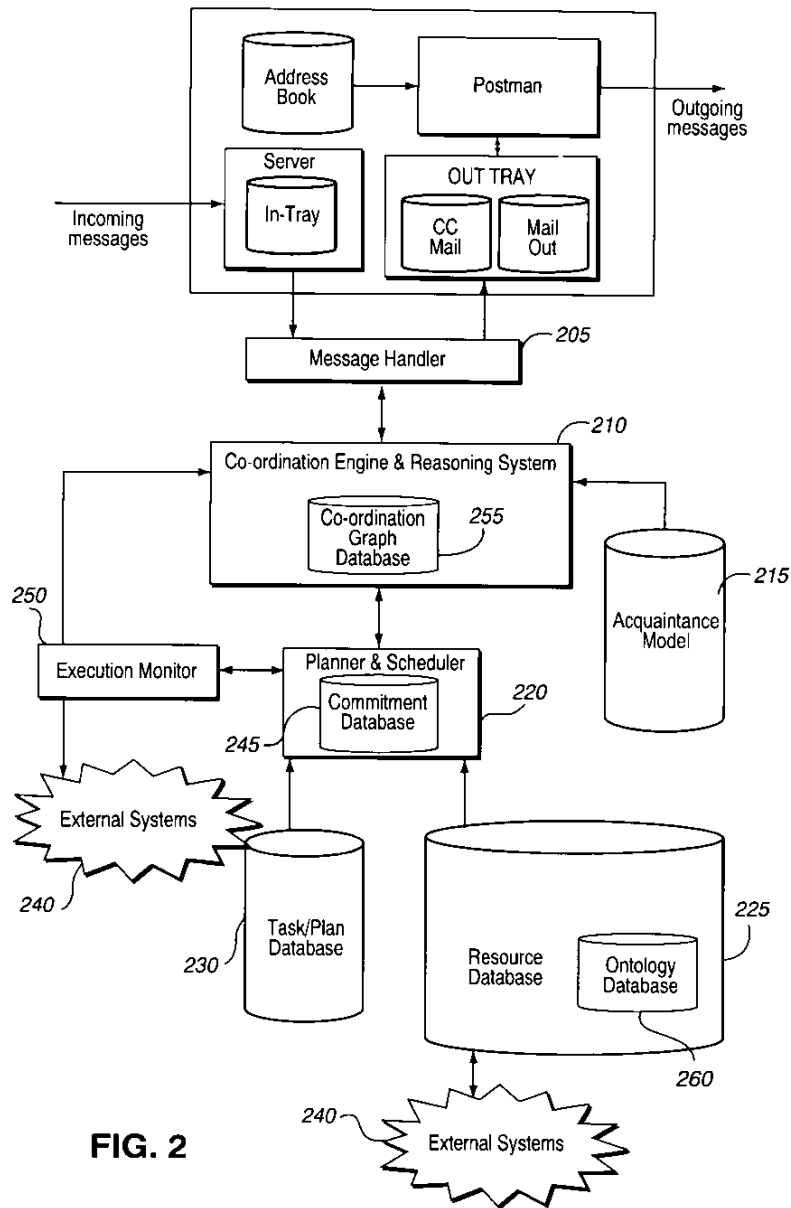
Thus, Lee discloses “wherein the agent command is one of a create agent command, an edit agent command, and a delete agent command” within the meaning of the ’733 Patent.

g. Claim 40
The method of claim 37 further comprising
suspending execution of the agent.

Lee in view of White discloses this limitation. Long Decl. (Ex. 1003) ¶ 183-184. According to Lee’s disclosure, “the user can suspend or resume jobs, and also suspend, resume or even kill agents.” Lee (Ex. 1006), 41:37–42. As described above, Lee expressly states that the Control Monitor tool provides for “Suspend,” (corresponding to the claims “suspending execution” functionality) and “[Resume]” agent functions. *Id.*, 41:20-24; *see also* Long Decl. (Ex. 1003), ¶ 183. Thus, Lee discloses “suspending execution of the agent” within the meaning of the ’733 Patent.

h. Claim 41
The method of claim 37 further comprising accepting
a template command from the user.

Lee in view of White discloses this limitation. Long Decl. (Ex. 1003) ¶ 185-190. “FIG. 2 shows a schematic architecture for a software module constituting an agent for distributed control, monitoring and management of a system.” *Id.*, 4:49–50.



Id., FIG. 2. “FIG. 3 shows a CABS platform for building an agent as shown in FIG. 2.” *Id.*, 4:47–48.

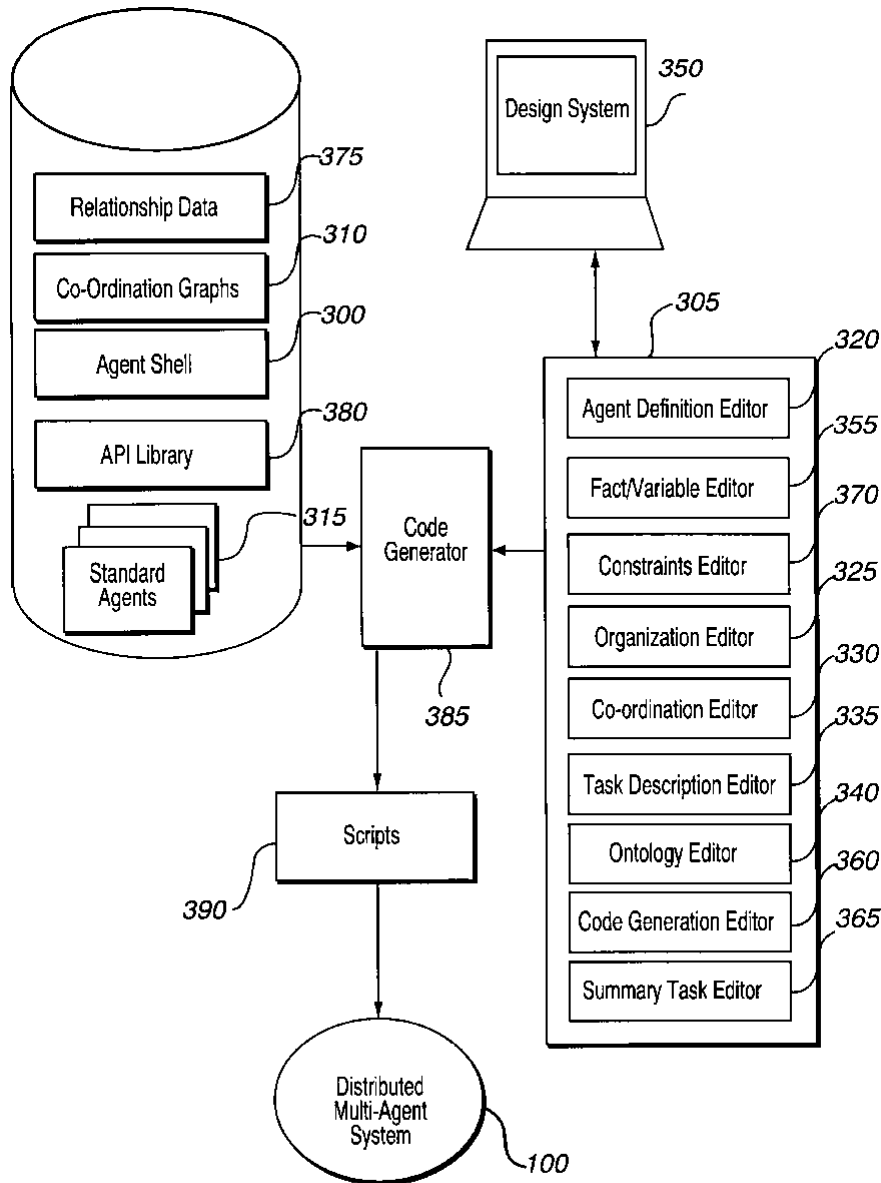


FIG. 3

Id., FIG. 3. “The agent shell 300 is simply the framework to support an agent structure as shown in FIG. 2.” *Id.*, 23:50–51. “Referring to FIG. 3, the means for ... automatically building a software module according to said architecture, is provided by the CABS platform. This provides an **agent template** 300 which dictates the agent structure shown in FIG. 2.” *Id.*, 23:39–45 (**emphasis** added). The CABS

system provided “an agent definition editor 320 ... for providing a logical description of the agent, its abilities and initial resources etc.” *Id.*, 23:60–67. Other editors, in addition to the “agent definition editor 320,” included: “an organization editor 325,” “a co-ordination editor 330,” “a task description editor 335,” “an ontology editor 340,” “a fact/variable editor 355,” “a code generation editor 365,” and “a summary task editor 370.” *Id.*, 24:1–34. The “set of editors” include functionality for “selecting agent functionality and inputting task and domain-related data.” *Id.*, 23:43–24:49. The ontology editor provides a more particular example.

The ontology editor 340 simply provides templates for describing these objects, that is (i) the different object types, (ii) the list of attributes of each of these object types, (iii) the type of values of that each attribute takes, and (iv) the default values (if any) for each attribute. This editor provides a template or form so that you can describe the vocabulary of the domain which the agents will be controlling.

The ontology editor 340 allows the user to identify an ENTITY, its ATTRIBUTES, the VALUES the attributes take, and typical DEFAULT values.

Id., 26:11–21. The use of an “editor” inherently involves a user providing commands to make the requested edits. Long Decl. (Ex. 1003), ¶ 189. The first stage in the CABS methodology is “Domain study and agent identification 515.” Lee (Ex. 1006), 25:26–30. “The user input at this stage will simply be a list of agent identifiers” *Id.*, 25:59–65. Inputs from a user directing how the CABS system will structure the

functionality of agents would be understood as commands from the view of a POSITA. Long Decl. (Ex. 1003), ¶ 189.

The CABS platform is based on a methodology for designing collaborative software agents which views the agents as having five primary sets of characteristics, or “layers”. ... [T]his methodology requires developers to provide inputs in respect of three of these, a definition layer 400, an organisation layer 405 and a coordination layer 410, as follows.

In an “agent definition” step 500, relevant to the definition layer 400, user inputs determine the agent in terms of its reasoning (and learning) abilities, goals, resources etc.

In an “agent organisation” step 505, relevant to the organisation layer 405, user inputs determine the agent in terms of its relationships with other agents, e.g. what agencies it belongs to, what roles it plays in these agencies, what other agents it is aware of, what abilities it knows those other agents possess, etc.

Lee (Ex. 1006), 24:58–25:6 (emphasis added). These “user inputs” further define the template that defines the agent functionality in the CABS system. Long Decl. (Ex. 1003), ¶ 187-189.

Thus, Lee discloses “accepting a template command from the user” within the meaning of the ’733 Patent.

i. Claim 42

The method of claim 41 further comprising allowing the user to create, modify, or delete an agent template within the network system.

Lee in view of White discloses this limitation. Long Decl. (Ex. 1003) ¶ 191-193. As described above, with respect to claim 41, the CABS platform is an architecture “for automatically building a software module” which “provides an agent template 300 which dictates the agent structure shown in FIG. 2” and tools for “selecting agent functionality and inputting task and domain-related data.” Lee (Ex. 1006), 23:38–49. The tools of the CABS system, which include numerous editors, allow the user to create and modify the agent template. Long Decl. (Ex. 1003), ¶ 192, 187-189. Among these tools is the Design System 350, which in turn includes user interface 305. Ex. 1006, 23:39-49; *see also id.* at FIG. 3. Lee describes how user interface allows the user to create and edit new templates: “In more detail [the editors of user interface] comprise: ... an ontology editor 340 for describing a suitable ontology for the domain [and] a fact/variable editor 355 for describing specific instances of facts and variables, using the templates provided by the ontology editor 340.” *Id.* 23:60-24:20 (internal parentheticals omitted). Thus, the ontology editor allows the user to create templates, and the fact/variable editor allows the user to edit them to describing specific instances of facts and variables. Long Decl. (Ex. 1003), ¶ 192. A POSITA would further understand that the user interface for creating and editing these templates would additionally provide functionality to delete unneeded

templates as a part of the iterative design process. *Id.*, ¶ 192; *see also* Ex. 1006, 25:66-26:4.

Lee considered the disclosed debugging and visualization tools “particularly important ... and novel.” Lee (Ex. 1006), 23:54–56. This was in part because “it is important to analyse what is going wrong so that it can be corrected.” Lee (Ex. 1006), 32:27–28.

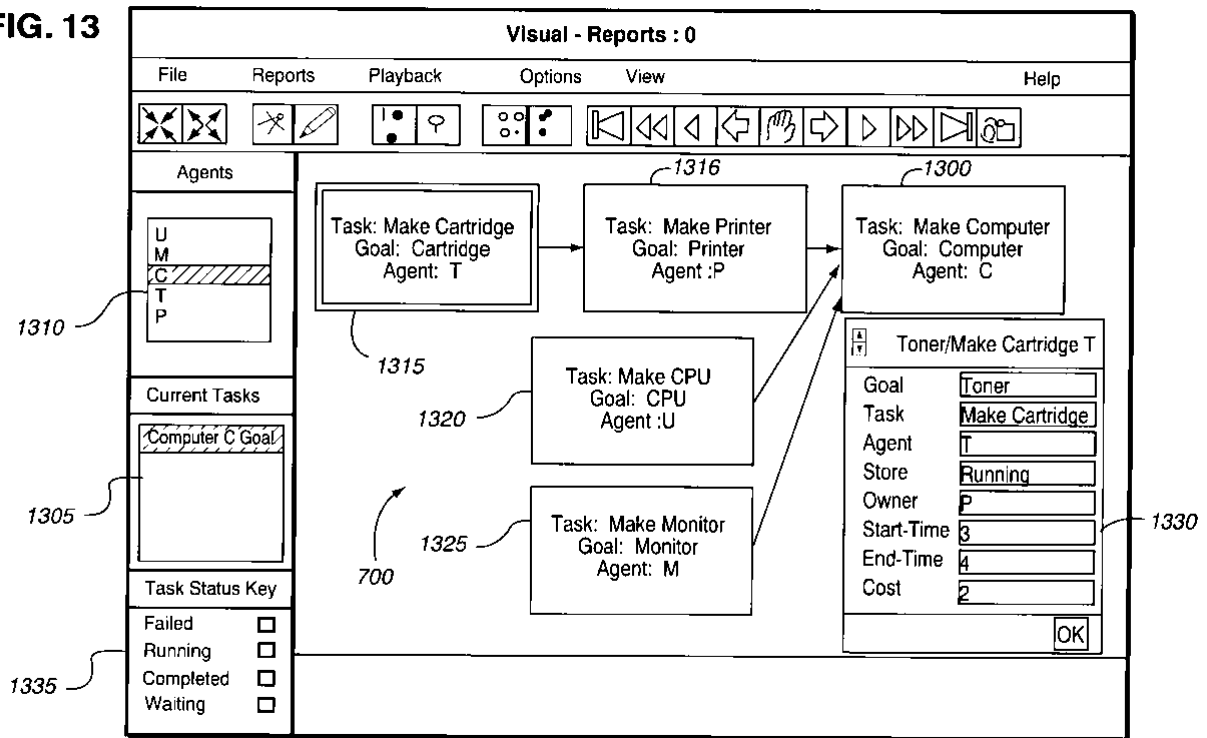
Thus, Lee discloses “allowing the user to create, modify, or delete an agent template within the network system” within the meaning of the ’733 Patent.

j. Claim 43
The method of claim 37 further comprising admitting the user to an agent area of a user interface.

Lee in view of White discloses this limitation. Long Decl. (Ex. 1003) ¶ 194-198. The discussion of Claim 41 included details of many tools for managing agents and defining their characteristics and functions. “The tool launcher 1015 launches (on demand by the user) instances of the different tools in the tool set, namely instances of the Society, Statistics, Micro, Reports, and Control Tools.” Lee (Ex. 1006), 34:53–56. “The visualiser provides a generic, customisable and scaleable visualisation system for use with a domain-independent toolkit for constructing multi-agent applications.” *Id.*, 33:23–26.

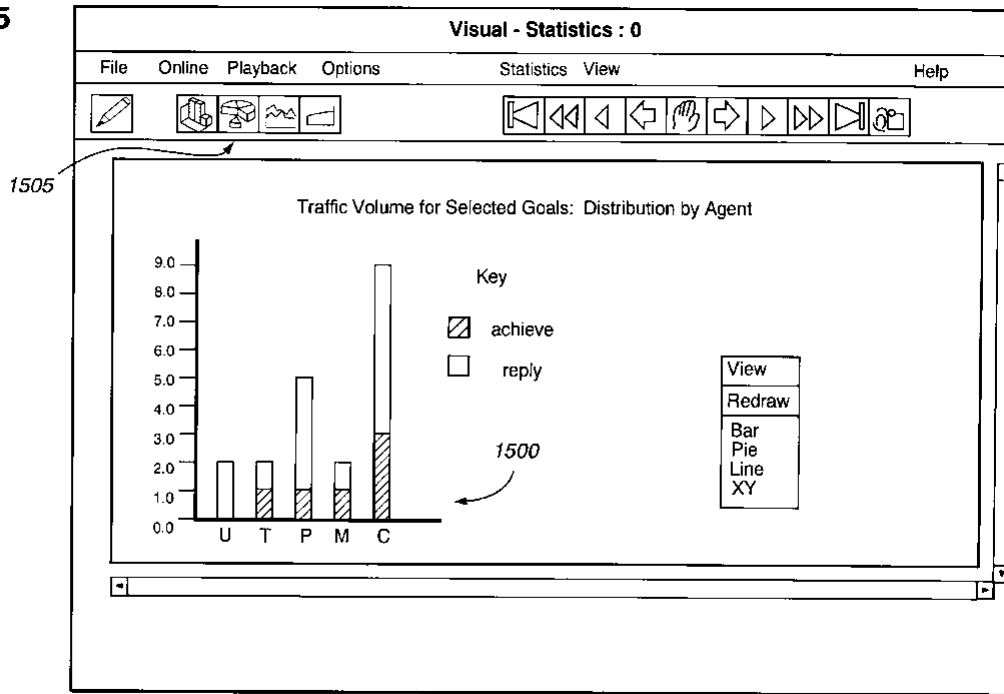
Figures 13 and 15 show exemplary agent areas of user interfaces disclosed in Lee.

FIG. 13



Lee (Ex. 1006), FIG 13 (showing Agents U, M, C, T, and P). “FIG. 13 shows schematically a GANTT chart as an example of the screen-based output of a ‘reports tool’ of the visualisation system in use.” *Id.*, 5:1–3.

FIG. 15



Id., FIG. 15 (showing statistics for “Distribution by Agent”). “FIG. 15 shows schematically an example of the screen-based output of a ‘statistics tool’ of the visualisation system in use.” *Id.*, 5:7–8.

Thus, Lee discloses “admitting the user to an agent area of a user interface” within the meaning of the ’733 Patent.

V. ENABLEMENT AND SECONDARY CONSIDERATIONS

The prior art references relied upon in this Petition are enabled. Long Decl. (Ex. 1003), ¶¶63-95.

Petitioner is not aware of any secondary considerations of non-obviousness tied to the Challenged Claims. Long Decl. (Ex. 1003), ¶199. To the extent Patent

Owner alleges that any secondary considerations of non-obviousness are tied to the Challenged Claims, Petitioner may request leave to file a reply to respond to any such allegations. In any event, any attempt by Patent Owner to rely on alleged secondary considerations of non-obviousness cannot overcome the strong showing of obviousness detailed above. Where, as here, there is a strong showing of obviousness, the Federal Circuit has held that even relevant secondary considerations supported by substantial evidence do not dislodge the primary conclusion of obviousness. *Ohio Willow*, 735 F.3d at 1344 (“[W]here a claimed invention represents no more than the predictable use of prior art elements according to established functions, as here, evidence of secondary indicia are frequently deemed inadequate to establish non-obviousness.”).

VI. PTAB DISCRETION SHOULD NOT PRECLUDE INSTITUTION

A. *Fintiv* Analysis

Discretionary denial under *Fintiv* is not warranted. Although there is currently a district court case in which the ’752 patent is asserted, the six-factor test set forth in *Apple Inc. v. Fintiv, Inc.*, IPR2020-00019, Paper 11 at 5–6 (P.T.A.B. Mar. 20, 2020) (precedential), favors institution. Ultimately, given the evidence of unpatentability described above, the *Fintiv* factors should not lead the Board to discretionarily deny institution.

Factor 1. Petitioner has filed a Motion to Transfer Venue to the Northern District of California. Ex. 1028. “Given NDCA’s high stay rates, NDCA may well stay [Petitioner’s] case ... pending IPR.” *Google LLC v. Parus Holdings, Inc.*, IPR2020-00847, Paper 9 at 12 (P.T.A.B. Oct. 21, 2020) (holding Factor 1 neutral where a similar motion was pending). Regardless of the venue, Petitioner intends to file a motion to stay the Current Litigation pending resolution of this IPR should this Petition be instituted. As such, this factor is neutral.

Factor 2. Trial in the Current Litigation is currently scheduled to begin on November 17, 2025. Ex. 1027, dkt. entry 28 (setting Jury Selection for 11/17/2025). However, as described above, Petitioner have moved to transfer the venue to the Northern District of California, which would, in all likelihood, result in trial being delayed past the date of any Final Written Decision in an IPR instituted based on the current Petition. Accordingly, any currently scheduled trial date is highly speculative. Thus, this factor remains neutral.

Factor 3. The Current Litigation is still in its early stages: the parties only recently filed their joint claim construction statement, only limited discovery requests have been served, and expert discovery remains far on the horizon. Ex. 1027, 3. Although the parties have exchanged initial infringement and invalidity contentions, “this investment is not significant.” *Vudu*, IPR2020-01688, Paper 16 at

12. This factor thus weighs against the Board exercising its discretion to deny the Petition.

Factor 4. There is minimal risk of overlap with arguments made in the district court because Petitioner hereby stipulates that, if IPR is instituted, Petitioner will not pursue in the district court litigation any ground for invalidity that was raised in the instituted IPR. PTAB case law holds that such a stipulation “mitigates to some degree the concerns of duplicative efforts between the district court and the Board, as well as concerns of potentially conflicting decisions.” *Sand Revolution II, LLC v. Continental Intermodal Group-Trucking LLC*, IPR2019-01393, Paper 24 (P.T.A.B. June 16, 2020) (informative). This factor thus weighs against denying the Petition.

Factor 5. This factor has no weight because the Petitioner and Patent Owner are the same parties as in the district court. *Weatherford U.S. v. Enventure Global Tech., Inc.*, IPR2020-01684, Paper 16 at 11–13 (P.T.A.B. Apr. 14, 2021) (holding only that “this factor does not weigh against exercising discretion to deny institution” where the Petitioner and Patent Owner are, respectively, the defendant and complainant in the parallel district court proceeding).

Factor 6. The prior art presented in this Petition renders the Challenged Claims unpatentable as obvious. The merits of Petitioner’s arguments are compelling because they lead to a strong conclusion that one or more claims are unpatentable by a preponderance of the evidence. In the previous IPR challenging

the claims of a related patent on the basis of the same prior art, the Board held that there was “a strong case for the obviousness of the challenged claims” and that “the prior art references cited in the Petition teach or suggest all limitations of the challenged claims.” This factor therefore weighs strongly against discretionary denial. In totality, the *Fintiv* factors weigh strongly in favor of institution.

VII. CONCLUSION

Petitioner have established a reasonable likelihood that the Challenged Claims are unpatentable. Petitioner therefore respectfully request that IPR be instituted and that all Challenged Claims be found unpatentable.

DATED: March 13, 2025

Respectfully Submitted,

/Robin A. Snader/

Robin Snader (Reg. No 66, 085)
Erise IP, P.A.
7015 College Blvd, Ste 700
Overland Park, KS 66214
robin.snader@eriseip.com
Attorney for Petitioner Databricks, Inc.

CERTIFICATION UNDER 37 C.F.R. § 42.24(d)

Under the provisions of 37 C.F.R. § 42.24(d), the undersigned attorney hereby certifies that the word count for Sections I–VII of the foregoing Petition for *Inter Partes* Review is 13,506, according to the word count tool in Microsoft Word.

DATED: March 13, 2025

By: /Robin A. Snader/

Robin Snader (Reg. No 66, 085)
Erise IP, P.A.
7015 College Blvd, Ste 700
Overland Park, KS 66214
robin.snader@eriseip.com
Attorney for Petitioner Databricks, Inc.

CERTIFICATE OF SERVICE

The undersigned certifies service pursuant to 37 C.F.R. §§ 42.6(e) and 42.105(a), (b) on the Patent Owner via FedEx overnight mail of a copy of this Petition for *Inter Partes* Review and supporting materials on the Patent Owner at the correspondence address of record for the '733 Patent:

STERNE, KESSLER, GOLDSTEIN & FOX P.L.L.C.
1101 K Street, NW
10th Floor
WASHINGTON, DC 20005
UNITED STATES

Further, a courtesy copy of this Petition for *Inter Partes* Review was sent via email to Patent Owner's litigation counsel:

Brandon V. Zuniga (bzuniga@bosfirm.com)
Marcus Benavides (mbenavides@bosfirm.com)
Mark Douglass (mdouglass@bosfirm.com)
Terry Afif Saad (tsaad@bosfirm.com)
Vivek Ramachandran (vramachandran@bosfirm.com)
Jeffrey Ray Bragalone (jbragalone@bosfirm.com)

DATED: March 13, 2025

By: /Robin A. Snader/

Robin Snader (Reg. No 66, 085)
Erise IP, P.A.
7015 College Blvd, Ste 700
Overland Park, KS 66214
robin.snader@eriseip.com
Attorney for Petitioner Databricks, Inc.

APPENDIX: CHALLENGED CLAIM LISTING

No.	Limitation
37[pre]	A method comprising:
37.a[1]	admitting a user to a network system
37.a[2]	wherein at least one agent is operable to consume a service resource while utilizing a service to perform a task for the user; and
37.b	allowing the user to create, modify, or delete the agent within the network system.
38	The method of claim 37 further comprising accepting an agent command from the user.
39	The method of claim 38 wherein the agent command is one of a create agent command, an edit agent command, and a delete agent command.
40	The method of claim 37 further comprising suspending execution of the agent.
41	The method of claim 37 further comprising accepting a template command from the user.
42	The method of claim 41 further comprising allowing the user to create, modify, or delete an agent template within the network system.
43	The method of claim 37 further comprising admitting the user to an agent area of a user interface.