

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICROSOFT CORPORATION,
Petitioner,

v.

X1 DISCOVERY, INC.,
Patent Owner

IPR2025-00254
U.S. Patent No. 10,552,490

PATENT OWNER'S RESPONSE

Mail Stop PATENT BOARD
Patent Trial and Appeal Board
U.S. Patent & Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450

TABLE OF CONTENTS

	Page
I. Introduction.....	1
II. Technology Background.....	2
III. The '490 Patent.....	5
IV. Claim Construction.....	8
V. GROUND 1: The Challenged Claims Are Patentable Over the Combination of Wilcox (EX1059), Londergan (EX1061), Raskin (EX1021), and Wu (EX1019).....	9
A. Petitioner Fails to Provide any Motivation to Combine the Disparate Teachings of Wilcox (EX1059), Londergan (EX1061), Raskin (EX1021), and Wu (EX1019).....	10
1. The Lotus Notes References Describe Four Distinct Search Tools.....	10
a. Query Builder.....	10
b. Search Builder.....	11
c. Search Bar.....	12
d. Searching by Forms	13
2. Petitioner Has Not Met Its Burden to Show Motivation to Combine the Different Search Tools Within the Lotus Notes References to Arrange Them According to the Claimed Invention.....	14
3. Petitioner Has Not Met Its Burden to Show Motivation to Combine the Lotus Notes References With Raskin and Wu.....	17
a. The Search Technologies Disclosed in the Lotus Notes References and Raskin Are Fundamentally Different and Incompatible	18
b. A POSITA Would Not Be Motivated to Combine Raskin's Sequential Scanning of a Single File with the Lotus Notes References' Indexed Search Tools.....	19

B.	Petitioner’s Reasonable Expectation of Success Arguments are Conclusory and Not Credible.....	26
C.	The Lotus Notes-Raskin-Wu Combination Does Not Render Obvious All Limitations of Independent Claim 1	28
1.	Element [1.5].....	28
2.	Element [1.10].....	29
3.	Element [1.13].....	30
D.	The Lotus Notes-Raskin-Wu Combination Does Not Render Obvious Dependent Claims 2-20	31
1.	Claims 2-3, 8-9, 20.....	31
2.	Claim 4.....	31
3.	Claim 5-7.....	31
4.	Claims 10-11	32
5.	Claims 14-18	36
6.	Claim 19	37
VI.	GROUND 2: Petitioner Fails to Show the Challenged Claims Are Obvious Over the Combination of Schwartz (EX1004), Sebastian (EX1005), Barber (EX1006), True (EX1009), and Baeza-Yates (EX1023).	37
A.	Petitioner Fails to Provide any Motivation to Combine the Disparate Teachings of Schwartz (EX1004), Sebastian (EX1005), and Barber (EX1006)	38
1.	The Entourage References Describe an Email System.....	39
a)	Entourage’s List Filtering Tool.....	40
b)	Entourage’s Find Tool	41
c)	Entourage’s Advanced Find Tool.	44
d)	Petitioner Mischaracterizes the Entourage References.....	45
2.	Petitioner Has Not Met Its Burden to Show Motivation to Combine the Different Search Tools Within the Entourage References to Arrange Them According to the Claimed Invention.....	49

B.	True is Non-Analogous Art.....	50
1.	True Relates to Graphical Representations and User Navigation of Audio Files.....	51
2.	True is Not in the Same Field of Endeavor	53
3.	True is not Reasonably Pertinent to the Problem to be Solved by the '490 Patent	55
C.	A POSITA Would Not Have Been Motivated to Combine the Entourage References with True	57
1.	True Does Not Teach Incrementally Searching a Set of Files for Multiple Search Strings.	59
2.	True is Directed to a Wholly Different Purpose than Entourage and Its “Interest Level” Method is Inherently Imprecise.....	60
3.	Petitioner Fails To Explain How A POSITA Would Modify Any Particular Search Tool in Entourage To Include True’s Disclosure.....	61
D.	A POSITA Would Not Have a Reasonable Expectation of Success in Combining the Ground 2 References	62
E.	The Entourage-True-Baeza Yates Combination Does Not Render Obvious All Limitations of Claim 1	63
1.	Elements [1.8-1.10].....	63
F.	The Entourage-True-Baeza Yates Combination Does Not Render Obvious the Dependent Claims	64
1.	Claims 2-20	64
2.	Claim 4.....	65
3.	Claim 5-7.....	65
4.	Claims 11	66
5.	Claims 14-18	66
6.	Claim 19.....	67
VII.	Conclusion	67

TABLE OF AUTHORITIES

	Page(s)
Cases	
<i>ActiveVideo Networks, Inc. v. Verizon Commc'ns, Inc.</i> , 694 F.3d 1312 (Fed. Cir. 2012)	15, 16
<i>Arctic Cat Inc. v. Bombardier Recreational Prods. Inc.</i> , 876 F.3d 1350 (Fed. Cir. 2017)	20
<i>Broadcom Corp. v. Emulex Corp.</i> , 732 F.3d 1325 (Fed. Cir. 2013)	15
<i>Cheese Sys., Inc. v. Tetra Pak Cheese & Powder Sys., Inc.</i> , 725 F.3d 1341 (Fed. Cir. 2013)	15
<i>Cir. Check Inc. v. QXQ Inc.</i> , 795 F.3d 1331 (Fed. Cir. 2015)	57
<i>In re Clay</i> , 966 F.2d 656 (Fed. Cir. 1992)	55, 56
<i>Cook Grp. Inc. v. Boston Sci.c Scimed, Inc.</i> , 809 Fed. Appx. 990 (Fed. Cir. 2020).....	23
<i>DSS Tech. Mgmt., Inc. v. Apple Inc.</i> , 885 F.3d 1367 (2018).....	68
<i>Innovention Toys, LLC v. MGA Ent., Inc.</i> , 637 F.3d 1314 (Fed. Cir. 2011)	57
<i>InTouch Techs., Inc. v. VGO Commc'ns, Inc.</i> , 751 F.3d 1327 (2014).....	16
<i>Juniper Networks, Inc. v. Correct Transmission, LLC</i> , 2024 WL 3517862 (Fed. Cir. July 24, 2024).....	26, 27, 64
<i>Kinetic Concepts, Inc. v. Smith & Nephew, Inc.</i> , 688 F.3d 1342 (Fed. Cir. 2012)	16, 39

<i>In re Klein</i> , 647 F.3d 1343 (Fed. Cir. 2011)	57, 61
<i>KSR Int'l v. Teleflex, Inc.</i> , 550 U.S. 398 (2007).....	16, 21, 25, 38
<i>In Re Magnum Oil</i> , 829 F.3d at 1380	63
<i>Metalcraft of Mayville, Inc. v. Toro Co.</i> , 848 F.3d 1358 (Fed. Cir. 2017)	10, 26
<i>Natera, Inc. v. NeoGenomics Lab'ys, Inc.</i> , 106 F.4th 1369 (Fed. Cir. 2024)	25
<i>In re Nuvasive, Inc.</i> , 842 F.3d 1376 (Fed. Cir. 2016)	17
<i>Polaris Indus., Inc. v. Arctic Cat, Inc.</i> , 882 F.3d 1056 (Fed. Cir. 2018)	16, 17
<i>In re Stepan Co.</i> , 868 F.3d 1342 (Fed. Cir. 2017)	14, 30, 40
<i>U.S. v. Adams</i> , 383 U.S. 39 (1966).....	24
<i>Virtek Vision Int'l ULC v. Assembly Guidance Sys., Inc.</i> , 97 F.4th 882 (Fed. Cir. 2024)	16, 38

EXHIBIT LIST

Exhibit No.	Description of Document
2001	Declaration of Mr. Martin Haeberli, dated April 28, 2025
2002	CV of Mr. Martin Haeberli
2003	Revised Stipulation to Stay, <i>XI Discovery, Inc. v. Microsoft Corp.</i> , No. 8:23-cv-02415 (C.D. Cal. Feb 29, 2024), ECF No. 52
2004	Order Granting Revised Stipulation to Stay, <i>XI Discovery, Inc. v. Microsoft Corp.</i> , No. 8:23-cv-02415 (C.D. Cal. Mar. 1, 2024), ECF No. 53
2005	Joint Status Report, <i>XI Discovery, Inc. v. Microsoft Corp.</i> , No. 8:23-cv-02415 (C.D. Cal. Mar. 12, 2025), ECF No. 57
2006	Minutes of Status Conference, <i>XI Discovery, Inc. v. Microsoft Corp.</i> , No. 8:23-cv-02415 (C.D. Cal. Mar. 12, 2025), ECF No. 59
2007	Excerpt of the Initial Determination Granting-In-Part Respondents' Motion for Summary Determination, <i>Certain Computing Devices Utilizing Indexed Search Systems and Components Thereof</i> , ITC-337-TA-1389, at 14 (Sept. 3, 2024)
2008	Excerpt of the Initial Determination of Violation of Section 337 and Recommended Determination on Remedy and Bond , <i>Certain Computing Devices Utilizing Indexed Search Systems and Components Thereof</i> , ITC-337-TA-1389, at 31-32 (Feb. 26, 2025)
2009	Email from counsel for Microsoft to counsel for X1 regarding <i>Sotera</i> stipulation, Apr. 18, 2025
2010	File History of U.S. Patent No. 7,370,035
2011	Information Disclosure Statement for US 12/328,410 dated Dec. 4, 2008 and filing receipt

Exhibit No.	Description of Document
2012	Information Disclosure Statement for US 11/108,335 dated June 26, 2008
2013	Information Disclosure Statement for US 15/199,928 dated June 30, 2016 (now U.S. Patent No. 10,897,443)
2014	Office Action for US 11/128,397 dated May 18, 2007
2015	Office Action for US 11/108,479 dated March 24, 2008
2016	Office Action for US 11/771,554 dated Dec. 23, 2009 (now U.S. Patent No. 8,533,176)
2017	Office Action for US 12/786,246 dated Jan. 10, 2012
2018	Office Action for US 13/918,306 dated June 24, 2015
2019	Information Disclosure Statement for US 15/012,684 dated Feb. 1, 2016 (now U.S. Patent No. 10,546,030) and filing receipt
2020	Supplemental Declaration of Martin P. Haeberli
2021	Transcript from the Deposition of Dr. Loren Terveen, dated October 21, 2025
2022	Dan Cosley et al., <i>SuggestBot: Using Intelligent Task Routing to Help People Find Work in Wikipedia</i> , Proceedings of the 12th International Conference on Intelligent User Interfaces (2007), marked as Exhibit 1 at the Deposition of Dr. Loren Terveen
2023	<i>Reserved</i>
2024	Windows XP Home Edition System Requirements (2001)

I. Introduction

The challenged claims of the '035 patent are patentable. Petitioner Microsoft Corporation (“Microsoft” or “Petitioner”) has not met its burden to demonstrate that either proposed combination renders the claims obvious. Specifically, the Petition lacks any rational motivation to combine Wilcox, Londergan, Raskin, and Wu, relying solely on impermissible hindsight and fails to show that a POSITA would not have a reasonable expectation of success. These failures alone are sufficient to find the claims patentable over the Ground 1 combination. Moreover, Petitioner’s surface treatment of dependent claims fails to show the combination discloses or renders obvious all limitations. Petitioner instead relies heavily on POSITA knowledge to fill gaps in its 4-reference combination without providing requisite support.

Petitioner’s combination of Schwartz, Sebastian, Barber, True, and Baeza-Yates fares no better. The combination fails as a threshold matter because True is not analogous art. But even if the combination is considered, this 5-reference combination fails to teach every limitation of the independent claims. Moreover, the Petition provides no motivation to combine the prior art teachings of these five references exists other than Petitioner’s purely hindsight reconstruction and fails to show a reasonable expectation of success in combining the teachings of the prior art. This failure is not surprising because, as X1’s expert establishes, a POSITA

would not have any expectation of success in the combination, let alone a reasonable one.

The Board should give little weight to the Petitioner's expert testimony, because Petitioner's expert admits to resorting to knowledge of a POSITA extensively for gap-filling, admits not considering whether the combinations he proposes could even be implemented, and supports several of PO's arguments.¹

II. Technology Background

While consumers today are accustomed to receiving and storing huge volumes of data in their cell phones and computers, this was a new phenomenon in 2002 at the time of the invention of the '490 patent. (EX2001, ¶38.) Before that time, the general availability of information and files had been steadily rising and the amount was expected to grow exponentially. (See EX1011, 1:23-26; EX1013, 1:14-20; EX2001, ¶38.) This increasing volume of data created many problems with the then-existing search functionalities. (EX2001, ¶38.) The functionalities of different search systems were generally grouped in different buckets, each of which had different coding, structure, and user experiences. (EX2020, ¶¶8-9.) A search could be conducted by sequentially scanning the words in one or more files to find characters that match a search string (a "brute force" search). (EX1023,

¹ The Board should not credit the testimony of Petitioner's expert for the additional reason that his declaration merely parrots the Petition, virtually word for word. (Compare, e.g., EX1003, ¶¶26-203 with Pet., 1-92.)

191; EX2020, ¶9.) Alternatively, a search could utilize an index to avoid having to look at every word of every file upon every search. (*See* EX1023, 191-92; EX2020, ¶9.) Search user experiences could be either delimited or incremental. (EX2001, ¶35.) Delimited search was most common and required a user to take an action such as hitting a search button before the search was initiated, whereas a search that was incremental would be conducted as the user typed their query. (EX2001, ¶35.) Searches could be characterized as occurring in a single open file, or across a large distributed collection of files. (EX2020, ¶9.) Searches could allow for single string (search term) queries where a string might be a single word or multiple words treated together as a single string, or multiple string queries where each string is treated as separate in the search. (EX1021, 125; EX1019, 6:54-62.) Finally, a search system might implement the ability to perform multiple string queries using multiple search field which each accept a single string, or by allowing for multiple strings in a single search fields separated by a string separator character or, alternatively, by a Boolean operator. (EX2020, ¶9.)

Conventional search applications that searched across multiple files or large volumes of data at the time of invention performed delimited searches, requiring users to type in a search term, click on a search button, often wait a very long time, and then review the results. (EX2001, ¶38; EX1001, 1:43-46.) In instances where the user was not satisfied with the results, the user would need to edit the search

terms and repeated this process. (EX2001, ¶38; EX1001, 1:46-50.) While easy to implement, delimited searches are slow and cumbersome for the user as they often require a long wait time before reviewing the results and then refining the search strings as necessary. (EX2001, ¶38; EX1001, 1:43-2:16.)

To reduce the amount of time it takes to search across a large number of files, search systems may index the file system by creating a data structure that associates each word found in the files with the documents where those words are found. (EX2020, ¶9; EX1023, 191-92.) However, the larger the file system or desired capabilities of the search, the more words there are to be indexed which results in a larger index. (See EX2020, ¶16; EX1023, 192-95.) As the size of the index grows, more computer resources are required to store the index, load the index into memory, and search the index, all of which reduce performance of the search system, leading to slower searches. (EX2020, ¶16.)

At the time of invention, incremental searching had been implemented in the context of searching for a single string within a single file or a list. (EX2001, ¶59.)² In these systems, the search was performed by sequentially scanning through all of the text in an open file. (EX2020, ¶¶10-25; EX1022, 55-57.) Implementing this

² Petitioner asserts (Pet., 30) that PO's expert in an ITC proceeding (on a different patent with different limitations at issue) admitted that "incremental searching" was well known but its cited evidence discusses incremental searching generally, not incremental search *across files*.

type of brute force incremental search across a large number of files would be time consuming and also consume significant computer resources. (EX2021, ¶18; EX2023, ¶148.) As of 2002, there were no systems that could search for and identify files on a file system incrementally. (EX2001, ¶39.)

In the years leading to the invention of the '490 patent, a variety of methods and systems were proposed to address some of the limitations of the prior art, but none of them (including those identified by Petitioner) incrementally searched across multiple documents as a query was entered. (EX2020, ¶¶8-25; EX2001, ¶¶38-47.)

III. The '490 Patent

The '490 patent, which is titled “Methods and systems for search indexing,” issued from U.S. Application Ser. No. 15/495,507 filed on April 24, 2017. The '490 patent generally relates to methods and systems for enhanced data searching and indexing across large volumes of files, and dynamically delivering files to a user in real time as they type their query so that they can actually see the files live during the search. (EX1001, 2:21-32.) As the '490 patent explains, the prior art delimited search tools and how they were “slow and cumbersome” and did not allow a user to update their searches in real time as they viewed the results. (EX1001, 1:43-2:16.)

To address these critical limitations described above when searching across multiple files, the '490 patent invented systems where users could incrementally search across large volumes of files using multiple search strings in a single search field and retrieve those files *as the search is conducted*. (EX1001, 2:21-32.) This allows the results for incremental searches to be “narrowed immediately after each character in a search string is entered by a user” so the user can adjust the search as needed based on the results. (EX1001, 2:21-32.) This was a major improvement over conventional delimited search, which requires the user to press a search button or hit enter before waiting (sometimes an extended period) before displaying the results. (EX1001, 2:21-47.)

The primary embodiment of the '490 patent allows the user to incrementally search multiple files via an index data structure, identify multiple groups of files each having one of the search terms, and retrieve results with a single search field that is capable of receiving multiple partial search strings. (EX1001, 2:21-32.) As such, the user can enter two strings into a single “search field; and incrementally locat[e] at least a first file that has a least a first word that begins with the first string and a second word that begins with the second string.” (EX1001, 2:41-47.) Figure 3G of the '490 patent shows an embodiment of the disclosed system:



FIG. 3G

(EX1001, Fig. 3G.).

Contrary to Petitioner's assertion, the claims of the '139 patent were allowed not just because the prior art did not disclose entering "two search strings entered in the same search field." (Pet., 3) Instead, the examiner found that the prior art, which had similar disclosures to the art raised by Petitioner, did not teach or suggest the "updating a set of matching files of the plurality or files as each successive character of the first partial search string and the second partial search string is received respectively in the first search field by identifying any of the plurality of files included in both the first one or more of the plurality of files and the second one or more of the plurality of files, wherein the first partial search string and the second partial search string are non-adjacent in at least one of the matching files." (EX1002, 129.).

IV. Claim Construction

Petitioner does not propose any terms for construction (Pet., 11) but improperly applies implicit constructions unsupported by any argument or intrinsic evidence, including at least for the terms “string separator character” (Pet., 23-24 n.7; EX1003, ¶¶88-89) and “document” (Pet., 64; EX1003, ¶160). Neither of Petitioner’s implicit constructions are supported by the intrinsic evidence and should be rejected.

In Ground 1, Petitioner equates a Boolean operator such as “AND” with the claimed “string separator character” required by all claims, without any supporting intrinsic evidence and contrary to the ALJ’s holding in the related ITC investigation. (EX2008, 31-32 (holding that Boolean operators are commands.) In the Institution Decision, the Board preliminarily construed “string separator character” to mean “a single-character string separator.” (Paper 14 (“ID”), 9-10.)

PO agrees with this construction as it is consistent with the intrinsic evidence. First, the ’490 patent distinguishes between string separator *characters* (such as space, comma, etc.) and “modifiers” (such as AND, OR, NOT, etc.). (*Compare* EX1001, cl. 8 with cl. 20.) Second, a POSITA would not understand Boolean operators such as “AND” to function as string separator *characters*. (*See, e.g.* EX1059, 162 (discussing the use of “logical operators AND, OR, NOT, and ACCRUE to find documents,” which a POSITA would understand to be referring

to a Boolean operator); EX2001, ¶112.) Indeed, a POSITA would not have understood Boolean operators like AND to be a string separator as used in the claims because when that word is used in a search, it would be read (and not ignored like the word “the” in a phrase) and the system would need to perform an action that modifies how the query is parsed. (EX2001, ¶112.) In other words, a POSITA would have known that the operator AND is a search syntax instruction rather than a string separator.

Similarly, in Ground 2 Petitioner equates a section of the transcript of an audio recording (the “transcript window” in True) with a file, without any citations to intrinsic evidence, and without explanation or justification. (Pet., 69.) The term “file” should take its plain and ordinary meaning and need not be construed. Indeed, Petitioner’s use of the term “file” is inconsistent with the intrinsic evidence and how a POSITA would understand the term as used in the ’139 Patent. (EX2020, ¶7; EX2001, ¶¶134-135.)

V. GROUND 1: The Challenged Claims Are Patentable Over the Combination of Wilcox (EX1059), Londergan (EX1061), Raskin (EX1021), and Wu (EX1019).

The challenged claims are patentable over the Lotus Notes-Raskin-Wu combination. Petitioner has failed to show unpatentability for Ground 1 for at least four reasons: (1) Petitioner has not established any motivation to combine the four different search tools disclosed in Wilcox and Londergan (the Lotus Notes

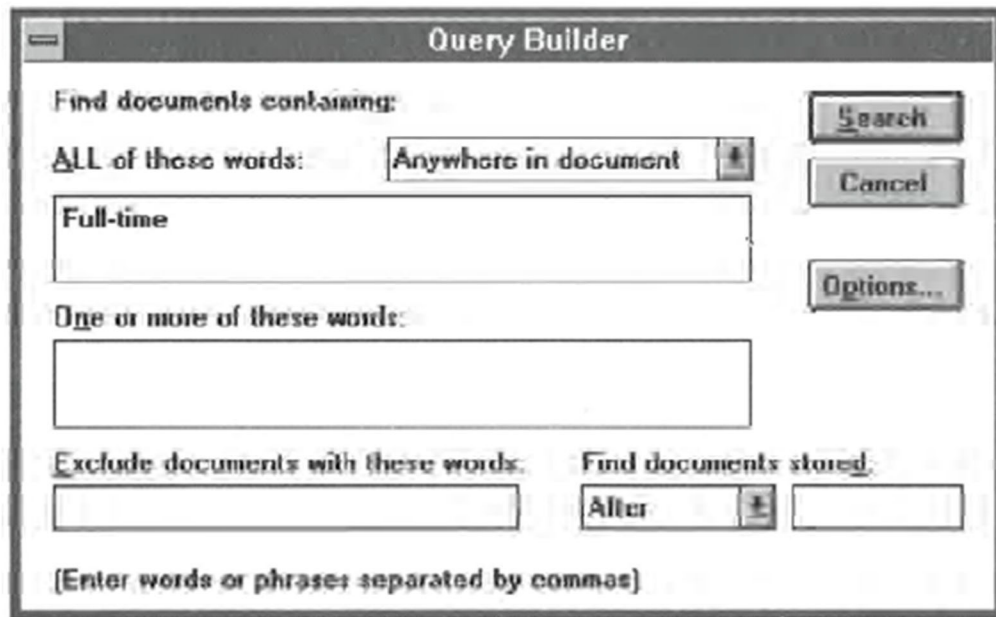
references); (2) Petitioner has not established motivation to combine the Lotus Notes references with Raskin; (3) Petitioner has not established a reasonable expectation of success; and (4) Petitioner improperly relies on the knowledge of a POSITA to supply missing claim limitations. Petitioner's Ground 1 combination amounts to selectively picking and choosing disclosures from the prior art, without regard for why a POSITA would do so, nor whether the disparate disclosures could be implemented together to achieve the stated goals, using "hindsight bias [as] the thread that stitches together prior art patches into something that is the claimed invention." *Metalcraft of Mayville, Inc. v. Toro Co.*, 848 F.3d 1358, 1367 (Fed. Cir. 2017).

A. Petitioner Fails to Provide any Motivation to Combine the Disparate Teachings of Wilcox (EX1059), Londergan (EX1061), Raskin (EX1021), and Wu (EX1019).

1. The Lotus Notes References Describe Four Distinct Search Tools

a. Query Builder

The 1993 version of Lotus Notes (v. 3.0) includes a feature called Query Builder. (EX1059, 162-64.) Query Builder, pictured below, provides multiple fields where, Petitioner alleges, a user can type multi-string search queries:



(EX1059, 163; *see also* Pet., 16-17; EX1003, ¶¶41, 90.) Petitioner relies on the Query Builder functionality described in Wilcox in its attempt to show the Wilcox-Londergan combination discloses searching on multiple strings entered into a single search field. (Pet., 23-24.)

b. Search Builder

The 1999 version of Lotus Notes (R5) eliminated the Query Builder feature, moving away from (as Petitioner alleges) multiple string search in a single field, and instead included Search Builder. (EX1061, 141-42.) Search Builder provides multiple fields, each of which can only be used for a single search string, as shown below:



(EX1061, 142; *see also* EX2001, ¶90.) Petitioner separately relies on the Search Builder functionality described in Londergan for allegedly disclosing features such as highlighting search strings in files as part of the search (as discussed below, it does no such thing) (Pet., 23-24) and searching for appointment information (Pet., 23-24.)

c. Search Bar

Both versions of Lotus Notes also include a separate search tool called the Search bar, which Petitioner relies on as allegedly disclosing searching on a first and second search string entered into a first search field, where the first and second strings are separated by a string separator character. (EX1059, 161; EX1061, 139-41; Pet., 13.) The Search bar is a search field that can perform multi-word searches

only under two conditions: first, for exact multiword phrases if the phrase is enclosed in quotation marks (EX1059, 162; EX1061, 140); second, when multiple search strings that are connected with a Boolean operator, such as “AND” or “OR” (EX1059, 162; EX1061, 140-41.) In neither circumstance can the Search Bar search multiple, separate strings with a string separator character as defined by the patent. (EX2001, ¶117.)

d. Searching by Forms

The Lotus Notes references disclose yet another search tool, searching by Forms. (EX1059, 167-68; EX1061, 143-44.) These forms are used to create files and are, in fact, “blank version[s]” of files. (EX1059, 58; EX1061, 100.) Forms are comprised of “fields,” which each “can contain a particular kind of data.” (EX1059, 58; EX1061, 100.) The searching by Forms functionality allows a user “to create queries that search for text in a particular field.” (EX1059, 167; EX1061, 100.) Thus, the searching by Forms functionality is limited to a search of files that were created using a specific form. (EX1059, 167; EX1061, 143.) There is no disclosure in either of the Lotus Notes references that searching by forms allows for: (1) incrementally searching for and identifying files; or (2) searching on two partial search strings entered into a single search field separated by a string separator character. (EX2001, ¶¶26-31.)

2. Petitioner Has Not Met Its Burden to Show Motivation to Combine the Different Search Tools Within the Lotus Notes References to Arrange Them According to the Claimed Invention

Petitioner's Wilcox-Londergan combination fails because Petitioner has not shown any motivation to combine elements from the four different search tools disclosed in the two Lotus Notes References that are six years and two versions apart. Indeed, Petitioner mischaracterizes these disparate search tools as a single search tool. (Pet., 13-15; EX1003, ¶¶61-65.) Where a combination involves "multiple embodiments from a single reference" (or from multiple references describing the same or similar systems), Petitioner must articulate "a motivation to make the combination." *In re Stepan Co.*, 868 F.3d 1342, 1346 n.1 (Fed. Cir. 2017). Petitioner has failed to do so.

Petitioner's only stated motivation to combine the Lotus Notes references and their teachings is that a POSITA would want to "obtain a more complete and comprehensive understanding of the Lotus Notes system, its features, and its capabilities." (EX1003, ¶76; Pet., 18.) But that misses the critical point of why a POSITA would have been motivated to combine the *elements* from those teachings in the way that the inventors did. Nowhere does Petitioner provide any rationale for why a POSITA would choose to combine, *inter alia*: performing a delimited search by entering two search strings in the same field separated by a Boolean operator as in the Search Bar tool; performing a delimited search by entering

multiple search strings each in a separate field as in the Search Builder tool; performing a delimited search by entering multiple search strings in a single field separated by commas as in the Query Builder tool (which had been discontinued by the time the 1999 R5 version was released); performing a delimited search by using the “Form” button to search specific fields of documents created using that form; and highlighting search strings within a search result only *after* opening the file. Instead, Petitioner’s obviousness argument boils down to the proposition that “because all of the elements that comprise the invention were known in the prior art,” it is obvious. *Broadcom Corp. v. Emulex Corp.*, 732 F.3d 1325, 1335 (Fed. Cir. 2013) (rejecting conclusory obviousness arguments that fail to provide a “plausible rational [sic] as to why the prior art references would have worked together”).

While PO disagrees that all elements, as Petitioner has artificially divided them, were independently disclosed in the prior art, even if they were, the Federal Circuit has repeatedly rejected these exact types of obviousness arguments as they are clearly based on the “hindsight combination of components selectively culled from the prior art to fit the parameters of the patented invention.” *Cheese Sys., Inc. v. Tetra Pak Cheese & Powder Sys., Inc.*, 725 F.3d 1341, 1352 (Fed. Cir. 2013); *ActiveVideo Networks, Inc. v. Verizon Commc’ns, Inc.*, 694 F.3d 1312, 1327 (Fed. Cir. 2012) (affirming rejection of expert opinion that all components of invention

were known in the prior art and were “modular” such that they could be combined as conclusory and “fraught with hindsight bias”); *Kinetic Concepts, Inc. v. Smith & Nephew, Inc.*, 688 F.3d 1342, 1368 (Fed. Cir. 2012) (“We must still be careful not to allow hindsight reconstruction of references to reach the claimed invention without any explanation as to how or why the references would be combined to produce the claimed invention.”); *Polaris Indus., Inc. v. Arctic Cat, Inc.*, 882 F.3d 1056, 1068 (Fed. Cir. 2018) (“We have observed that ‘the prejudice of hindsight bias’ often overlooks that the ‘genius of invention is often a combination of known elements which in hindsight seems preordained.’”); *KSR Int’l v. Teleflex, Inc.*, 550 U.S. 398, 418 (2007) (holding “it can be important to identify a reason that would have prompted a person of ordinary skill in the relevant field to combine the *elements* in the way the claimed new invention does”) (emphasis added).

To the extent Petitioner argues improperly for the first time in reply (*Virtek Vision Int’l ULC v. Assembly Guidance Sys., Inc.*, 97 F.4th 882, 887 n.3 (Fed. Cir. 2024)) that the alleged “straightforward” or “interchangeable” nature of the claim elements would have motivated a POSITA to combine them in the way that is claimed, the Board should reject this argument like the Federal Circuit has repeatedly. *ActiveVideo*, 694 F.3d at 1327; *InTouch Techs., Inc. v. VGO Commc’ns, Inc.*, 751 F.3d 1327, 1351-52 (2014) As to the merits, as explained by Mr. Haeberli, the technology is anything but simple or straightforward. (EX2003,

¶64.) And even if it were (it is not), the Federal Circuit has cautioned that “when the art in question is relatively simple... the opportunity to judge by hindsight is particularly tempting” and should be rejected. *Polaris*, 882 F.3d at 1068 (internal quotations omitted).

The only stated motivation to combine the teachings of the Lotus Notes references generally, to better understand the Lotus Notes system, says nothing about why a POSITA would be motivated to combine the disparate elements of the four search tools in Lotus Notes in a combination that aligns with the claimed invention, it is conclusory and lacks the “thorough and searching” requirement for specificity when trying to prove that a claimed invention is obvious. *In re Nuvasive, Inc.*, 842 F.3d 1376, 1381-82 (Fed. Cir. 2016). Therefore, Ground 1 fails as to all challenged claims.

3. Petitioner Has Not Met Its Burden to Show Motivation to Combine the Lotus Notes References With Raskin and Wu

Petitioner’s stated motivation to combine the Lotus Notes references with Raskin’s disclosure of “incremental search” - “speed, effectiveness, and ease of use” - is not supported by the prior art, and is based on a fundamental mischaracterization of the claimed invention. The claimed invention is incrementally searching and identifying *files* across a large set of files utilizing multiple search strings so a user can view a subset of those files in real time as the

user types the query in a single search field. By utilizing separate search strings in the same field, each string incrementally searches multiple documents and locates an independent group of documents responsive to the corresponding string in order to incrementally identify one or more documents that are included in each of the first and second groups of documents. The Petition's argument that the Lotus Notes references render the two search terms in a single field claim element obvious would require a POSITA to equate having two strings in the same field for purposes of any search (including a delimited and/or sequential search) with the claimed invention, which is simply not the case. (EX2020, §§26-62.) As such, a POSITA would not be motivated to combine the Lotus Notes references with Raskin, which has nothing to do with incrementally searching and identifying groups of files using multiple search strings, simply because Raskin mentions the word "incremental" or the "idea" of incremental search, as Petitioner contends. Petitioner's hindsight attempt to show obviousness by pointing to the prior art's mention of "incremental search" without any consideration of the context of that disclosure is insufficient to show motivation to combine.

a. The Search Technologies Disclosed in the Lotus Notes References and Raskin Are Fundamentally Different and Incompatible

The Lotus Notes references disclose a database system which requires an index to enable full-text searching of the files in the database. (EX1059, 159;

EX1061, 144.) The Lotus Notes references and both sides' experts agree that the size of an index in such a system will necessarily depend on what search capabilities are desired, and the larger the index, the slower the search. (EX1059, 169; EX2021, 38:1-39:17; EX2020, ¶16.) Larger indexes are also disfavored because they require more processing power and available RAM. (EX2020, ¶¶32-33.). Thus, a smaller index is preferred if one wishes to have faster search, or if there is less processing power available. (EX2021, 38:1-39:17 (Petitioner's expert testifying that a larger index "could increase the time it took to search it" and "making the index smaller would be more efficient in general."))

All of the search tools in Lotus Notes references are delimited searches; requiring a user to click "Search" to perform the search. (Pet., 15.) The portion of Raskin Petitioner relies on to allegedly disclose incremental searching is a high-level discussion of the search functionality in the GNU EMACS text editor. (Pet., 19-20; EX1021, 125-27.) EMACS only allows search within a single, open text file. (EX1022, 55.) Once opened, EMACS performs a sequential or "brute force" search (*see supra* §II) from wherever the cursor is when the search begins by scanning forward through the text aka "buffer" in EMACS terminology for those characters. (EX1022, 55; EX2020, ¶¶35-37.)

**b. A POSITA Would Not Be Motivated to Combine
Raskin's Sequential Scanning of a Single File with the
Lotus Notes References' Indexed Search Tools**

Petitioner's stated motivation to combine Raskin with the Lotus Notes references, improved "speed, effectiveness, and ease of use," is conclusory and contradicted by the prior art. (Pet., 19; EX1003, ¶78.) First, Dr. Terveen testified that he was simply taking from Raskin the "*idea* of incremental search," without considering the context in which it was disclosed. (EX2021, 114:10-19, 116:20-118:6.) Dr. Terveen testified that he did not consider that Raskin's disclosure of incremental search was with a sequential scanning tool within a single open file, and that doing so was not required for his obviousness opinion. (EX2021, 116:10-117:17.) But a POSITA must consider all the teachings of a reference, and not pick out general "concepts" disclosed in the art without considering the context in which they are used. *Arctic Cat Inc. v. Bombardier Recreational Prods. Inc.*, 876 F.3d 1350, 1360 (Fed. Cir. 2017) ("it is error to fail to consider the entirety of the art."). And as discussed below, a POSITA considering the context of Raskin's disclosures would not be motivated to combine the primitive incremental searching of Raskin with the Lotus Notes references' search tools because doing so would be incompatible with Petitioner's stated motivation.

First, for Petitioner to prove motivation to combine, it must demonstrate that a POSITA would understand that the search functionalities in Raskin and the Lotus Notes references are "similar" or that the incremental search technique disclosed in Raskin would improve the search functionality in the Lotus Notes references "in

the same way.” *KSR*, 550 U.S. at 417-418 (proving obviousness when combining devices in different prior art references requires, for example, establishing that “if a technique has been used to improve one device... ***a person of ordinary skill in the art would recognize that it would improve similar devices in the same way...***”) (emphasis added). Petitioner fails to do this at all, incorrectly insisting that it need not and that identifying each element of the claimed invention was in the prior art is sufficient. (Pet., 18-20; EX1003, ¶¶78-81; EX2021, 137:3-138:20.)

As Mr. Haeberli explains, the search functionalities are meaningfully different and a POSITA would have recognized that implementing Raskin’s incremental search functionality in the Lotus Notes search system would result in slower searches for the user, undermining Petitioner’s stated motivation. (EX2020, ¶12.) Raskin discloses a brute force search that sequentially scans every character in the text of a single, open file for a single search string, not multiple strings (*see supra* §V.A.3.a), while the Lotus Notes references disclose an indexed database system, which accommodates delimited (not incremental) searches for multiple search strings. (*Supra* §V.A.1.) A POSITA would understand that these differences are fundamental and meaningful and would have differences in architectural approach and processing requirements. (EX2020, ¶¶33-37.)

As for actually combining the teachings of these two references, a POSITA would not be motivated to do so because he would know that doing so would not

result in improved “speed, effectiveness, and ease of use” as Petitioner alleges. (Pet., 19; EX1003, ¶78; EX2021, 106:7-109:6.) Dr. Terveen testified that increased speed means “the search being faster for the user” (EX2021, 106:7-107:14), but Petitioner fails to explain *how* Raskin’s sequential scanning incremental search would be implemented in the search system disclosed in the Lotus Notes references to increase speed, effectiveness, or ease of use, and, in fact, Dr. Terveen stated that he was not required to consider how incremental search would be implemented, but only that a POSITA would be motivated to use the “idea” of incremental search in Lotus Notes. (EX2021, 116:20-118:6.)

Even if it was proper for Petitioner to ignore Raskin’s entire teachings and rely solely on the “idea” of incremental search to be implemented into the Lotus Notes system, a POSITA would not be motivated to do so for two reasons: (1) implementing Raskin’s incremental search would slow down the search and (2) would not be attractive to a POSITA in 2002 because of technical limitations in the computing power of most consumers at that time. A POSITA’s view on implementation of the combination is critical as a POSITA is only motivated to combine teachings that would actually solve a problem or improve a system and is *not* motivated implement teachings that would undermine or reduce the functionality of a system. *Cook Grp. Inc. v. Boston Sci.c Scimed, Inc.*, 809 Fed. Appx. 990, 1000-01 (Fed. Cir. 2020) (“[T]he Board may consider whether the

modification [of a primary reference] renders the reference inoperable for its intended function in deciding whether a POSA would have a motivation to combine ...”) (citing *DePuy Spine, Inc. v. Medtronic Sofamor Danek, Inc.*, 567 F.3d 1314, 1326 (Fed. Cir. 2009)).

Implementing incremental search in the Lotus Notes indexed system would necessarily require a larger index, which would slow down, rather than increase the search speed. (EX2020, ¶ 33; EX2021, 37:7-39:17; *supra* §V.A.3.a.) Alternatively, applying the non-indexed, brute force search of Raskin into a file storage system of Lotus Notes would be even slower as each file would have to be, sequentially scanned one at a time. (EX2020, ¶36; EX1003, ¶148 (Dr. Terveen explaining that “scanning every word of every file sequentially ... takes more time and consumes more computing resources” than searching using an index).) This would slow the search down so much that it may cease to be incremental in practice. (EX2020, ¶35.) This not only contravenes Petitioner’s cited motivation of increasing speed but, if the search was no longer incremental due to the slow speed, would not even fall within the scope of the claimed invention. (EX2020, ¶52; *see also* EX2021, 104:17-105:11.)

Furthermore, the average consumer’s computer in 2002 had far less memory and processing power than computers today. (EX2020, ¶¶47-48) Incrementally searching multiple files with multiple search terms in an indexed file system would

require a much larger index and would take up a large proportion of the average consumer computer's RAM and hard disk space in 2002. (EX2021, 37:7-39:17.) In one conservative estimate, Dr. Haeberli explains that an index for such a system could take up half of the available RAM for the computer. (EX2020, ¶¶47-48) This would likely lead to performance issues, and searches may lag to the point of no longer being incremental in practice. (EX2020, ¶¶49-51) A POSITA does not exist in a vacuum, and would not be motivated to combine prior art teachings if doing so would not be practical in the real world or would not achieve the stated motivation. *See U.S. v. Adams*, 383 U.S. 39, 51-52 (1966) (a POSITA is not motivated to combine the prior art when doing so would be "not practical.").

Dr. Terveen's own cited prior art, as well as post art, also notes these drawbacks of increased index size and computing limitations when implementing incremental search, suggesting that a POSITA would not be motivated to increase the index size. (EX2020, ¶¶54-62.) Thus, a POSITA would not have been motivated to implement Raskin's incremental search in the Lotus Notes search functionality because doing so would have required either a larger index that could have made searches *slower* rather than *faster*, or conducting a sequential/brute force search that would have the same result, defeating Petitioner's stated motivation to combine. (EX2020, ¶59.)

Raskin himself noted that notwithstanding any potential benefits of incremental searching, “almost all interface-building tools make it easy to implement delimited searches and *difficult or impossible* to implement incremental searches.” (EX1021, 126 (emphasis added).) A POSITA is not motivated to undertake a modification of the prior art with something that the art teaches would be difficult or impossible to do. *Natera, Inc. v. NeoGenomics Lab’ys, Inc.*, 106 F.4th 1369, 1377-78 (Fed. Cir. 2024) (affirming a finding that there was no motivation to modify an existing array system because the prior art “put forth significant evidence of obstacles” in modifying it).

A careful examination of the disadvantages and challenges for how a POSITA would potentially implement the disparate teachings of Raskin and the Lotus Notes references reveals that Petitioner’s motivation to combine is rife with hindsight bias. *KSR*, 550 U.S. at 421 (“A factfinder should be aware, of course, of the distortion caused by hindsight bias and must be cautious of arguments reliant upon ex post reasoning.”). Petitioner has not cited to a single reference that even suggests performing an incremental search across a database of files and indeed every cited reference for incremental search describes doing so in a single file via sequential scanning. (EX2020, ¶61.) And by Dr. Terveen’s own admission, his combination merely picks out the “idea” of incremental search from Raskin (while ignoring its teachings as to implementation and challenges) and applies it in some

unknown fashion to one or more of the tools in the Lotus Notes references or some hodge-podge system using the elements of those tools to meet the claim limitations. *Metalcraft*, 848 F.3d at 1367. Petitioner has not established motivation to combine.

B. Petitioner’s Reasonable Expectation of Success Arguments are Conclusory and Not Credible

Petitioner’s only argument that a POSITA would have a reasonable expectation of success in combining Raskin’s incremental sequential scanning of a single file with the Lotus Notes References’ indexed search system is that “it would have required only simple and straightforward software modifications.” (Pet., 28; EX1003, ¶96.) This conclusory and unsubstantiated statement is insufficient as a matter of law. *Juniper Networks, Inc. v. Correct Transmission, LLC*, 2024 WL 3517862, at *4 (Fed. Cir. July 24, 2024).

As discussed above, the search functionalities disclosed in the Lotus Notes References and Raskin are materially different in such a way as to render combining them anything but simple and straightforward. (*See supra* §V.A.) While Petitioner relies on Raskin’s discussion of the advantages of incremental search in the abstract, Petitioner provides no meaningful details about how incremental search works as implemented in EMACS, nor how a software designer would go about implementing incremental search in any other system. (*See* Pet., 15-17, 18-

20; EX1003, ¶¶67-69, 78.) Petitioner also ignores Raskin’s own admission that it is “*difficult or impossible* to implement incremental searches.” (EX1021, 126 (emphasis added).) Indeed, Dr. Terveen admitted that he did not consider, and did not need to consider, any aspect of implementation. (EX2021, 116:20-117:17.) But the fact that Dr. Terveen took the “idea of incremental search” from Raskin and failed to consider the known hurdles to implementation (which would also negatively impact end user’s experience), including those identified by Raskin himself, renders Dr. Terveen’s testimony that the combination would be straightforward conclusory and not credible. *See Juniper Networks*, 2024 WL 3517862, at *4.

Implementing the brute force, single document incremental search of Raskin with the delimited, index search of Lotus Notes would not be simple or straightforward. (EX2020, ¶64.) For example, to the extent the combination is accomplished using an index, this would require a significantly larger index, which would result in slower searches, and potentially even exceed a user’s computer processing capabilities. (EX2020, ¶67.) Alternatively, to the extent the combination is accomplished by sequential scanning of each file in the set of files, the search would be so slow as to be unusable. (EX1003, ¶148.) The problem of slow and cumbersome searches is the precise problem with the prior art search systems identified in the ’490 patent and is the very problem the inventors sought

to address . (See EX1001, 1:35-2:11.) Thus, implementing the incremental search of Raskin into the Lotus Notes system without actually causing the search to slow so much as to be inoperative for end users would not be simple or straightforward . (EX2020, ¶¶63-71.)

If it were as easy to code for any software functionality one desires, as Petitioner and Dr. Terveen opine, then using incremental search across groups of files would certainly have been implemented long before the priority date here. And if it were so simple, Raskin would not have characterized implementing incremental search as “*difficult or impossible*” when only discussing incremental search in the context of a single document, let alone across groups of documents. (EX2021, 126.) Even the fact that both of Petitioner’s Grounds rely on a combination of four and five references to achieve the functionality Petitioner claims was “obvious” and “straightforward” to implement undermines Petitioner’s argument.

C. The Lotus Notes-Raskin-Wu Combination Does Not Render Obvious All Limitations of Independent Claim 1

1. [1.5]

Petitioner has not established that its Lotus Notes-Raskin-Wu combination renders obvious this limitation. Petitioner relies solely on the Lotus Notes references for this limitation and acknowledges that the Lotus Notes references expressly disclose only a full text index. (Pet., 21; EX1003, ¶84.) Petitioner then

invokes knowledge of a POSITA for the specific index structure required by the claim, stating that “this is how indexes work.” (Pet., 21; EX1003, ¶84.) Relying on the knowledge of a POSITA to supply a missing limitation is improper gap-filling, *see* 37 C.F.R. 42.104(b)(4). Further, Petitioner fails to acknowledge that the “*character strings*” of claim 1 must be matched to “*partial search strings*” [1.8-1.13] and thus must themselves be partial strings. Petitioner does not even address this feature or provide any evidence that a POSITA would have known to index partial search strings at the time of invention. Further, Petitioner fails to acknowledge that there are many different possible index structures and points to no disclosures in the cited art of the structure disclosed in this claim limitation. (Pet., 23-26; EX2020, ¶72.)

2. [1.10]

Petitioner has not established that its Lotus Notes-Raskin-Wu combination renders obvious this limitation under the correct construction of “*string separator character*.” (EX2020, ¶73.) Petitioner first points to the Lotus Notes Search bar functionality, which requires multiple search strings to be separated by a Boolean AND operator, which, as the panel correctly found in the Institution Decision, is not a string separator character for the reasons explained above. (Pet., 23-24; EX1003, ¶¶88-91; *see supra* §IV.) Petitioner alternatively points to both the Query Builder and Search Builder functionalities, which Petitioner alleges receive two

strings separated by a comma. (Pet., 23-24; EX1003, ¶¶88-91.) But Search Builder only accepts a single search string in each field. (EX1061, 142; ID, 29 n.18.)

Petitioner's reliance on the Query Builder functionality is insufficient to render this limitation obvious for the reasons explained above, specifically because Petitioner has not adequately explained how the different search functionalities disclosed in the Lotus Notes references are combined to meet all limitations of the challenged claims. *In re Stepan Co.*, 868 F.3d 1342, 1346 n.1 (Fed. Cir. 2017).

3. [1.13]

Petitioner has not established that its Lotus Notes-Raskin-Wu combination renders obvious this limitation, which requires incrementally searching across “*a plurality of files*.” Petitioner acknowledges that the Lotus Notes references do not disclose incremental search. (Pet., 15; EX1003, ¶¶66, 92.) Instead, Petitioner relies on Raskin for incremental search. (Pet., 15, EX1003, ¶¶66, 92.) However, as explained above, the portion of Raskin Petitioner relies on (EX1021, 125-27) does not disclose searching “a plurality of files” to identify a “first partial search string” and a “second partial search string”; rather, Raskin discloses sequentially scanning the text *of a single open file* for the search string. (*See supra* §V.A.) As explained above, Petitioner does not explain why a POSITA would be motivated to combine, or how to implement, Raskin's teachings about sequential scanning of a single text file in an indexed search system designed to search for multiple files, nor does

Petitioner establish that a POSITA would have had an expectation of success in doing so. (*Supra* §V.B.) Therefore, Petitioner has failed to establish that the prior art renders obvious this claim limitation.

D. The Lotus Notes-Raskin-Wu Combination Does Not Render Obvious Dependent Claims 2-20

1. Claims 2-3, 8-9, 20

Petitioner's combination for these dependent claims fails for at least the reasons explained in Sections V.A-C. (*Supra* §§V.A-C.)

2. Claim 4

Petitioner relies solely on the knowledge of a POSITA to meet this limitation. (Pet., 31-32; EX1003, ¶¶104-106.) Relying on the knowledge of a POSITA to supply a missing limitation is improper gap-filling, *see* 37 C.F.R. 42.104(b)(4). Further, Petitioner does not provide any evidence that a POSITA would have known to include associations with attribute fields in an index, fails to acknowledge that there are many different possible index structures, and points to no disclosures in the cited art of the structure disclosed in this claim limitation. (Pet., 31-32; EX1003, ¶¶104-106.)

3. Claim 5-7

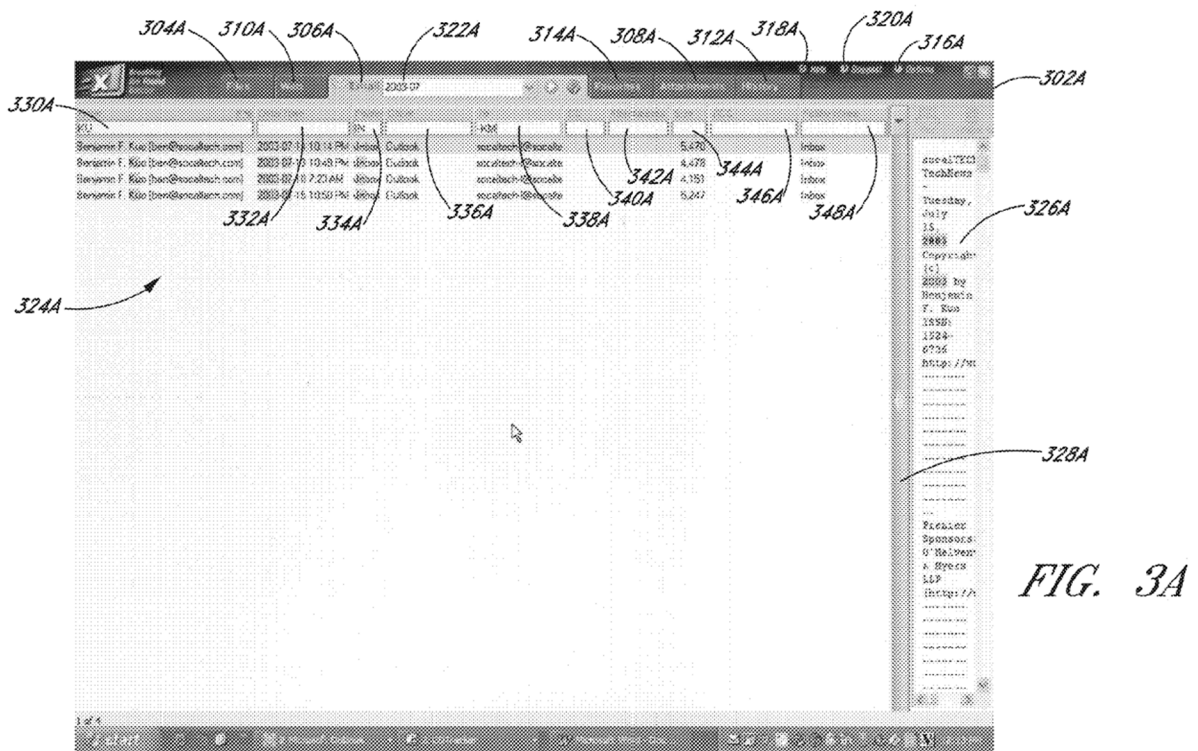
Petitioner relies solely on the knowledge of a POSITA to meet these limitation. (Pet., 32-33; EX1003, ¶¶107-08.) Relying on the knowledge of a POSITA to supply a missing limitation is improper gap-filling, *see* 37 C.F.R.

42.104(b)(4). Petitioner provides no motivation for a POSITA to implement these features and invokes only the generic “straightforward software modifications” in support of reasonable expectation of success. (Pet., 33; EX1003, ¶108.) This is insufficient to meet Petitioner’s burden to establish obviousness for the same reasons explained above. (*Supra* §§V.A-B.)

4. Claims 10-11

Petitioner misleadingly characterizes the Lotus Notes references to obscure a fatal deficiency in Lotus Notes—to display a file, the user must manually open the file. (EX1059, 160-61; EX1061, 144-45.) That is, Lotus Notes does not display any portion of the selected identified file “*in the interactive user interface.*” The Petition contends that “the Lotus Notes references disclose displaying a portion of a selected file and highlighting occurrences of only the searched string,” citing Wilcox's statement that “[t]he found text [] is indicated by a red box” and Londergan's explanation that Lotus Notes “shows you the document and highlights the text you wanted to find.” (Pet., 35-36.) Dr. Terveen's declaration echoes this, opining that “[t]he Lotus Notes+Raskin+Wu combination renders these claims obvious at least because the Lotus Notes references disclose displaying a portion of a selected file.” (EX1003, ¶112.) However, both references expressly require the user to *open the file* before any portion can be displayed or highlighted.

Claim 10 requires that a portion of the identified file be displayed “*in the interactive user interface.*” The ’490 patent discloses that the user interface includes “a list area or pane 324A, which includes a list of the search results” and “a view area or pane 326A is provided that displays the content of the first item or user-selected item from the list pane or area 324A.” (EX1001, 14:33-42.) This is illustrated in Figure 3A:



(EX1001, Fig. 3A.) While clicking on the list pane entry causes the content of that file to be displayed in the view pane, the file itself can be opened “in its native application” by double clicking on it. (EX1001, 14:49-60.)

The disclosures of the Lotus Notes references contradict Petitioner's implied characterization and demonstrate that neither reference discloses or renders obvious displaying a portion of the identified file "*in the interactive user interface*." Wilcox instructs the user to "[o]pen the first document listed in the view." (EX1059, 164.) Londergan explains that "[y]ou open a document uncovered by a search the same way you would open any other document. Double-click it or select it with your mouse and press Enter. Notes then shows you the document and highlights the text that you wanted to find." (EX1061, 145-46.) Dr. Terveen agreed, testifying that "the express disclosure in Wilcox" is that "in order to view the document and see the highlighting or the red box you have to actually open the document." (EX2021, 66:4-17.) He admitted the same for Londergan, confirming "that is the direct or express disclosure in Londergan." (EX2021, 71:7-73:8.)

Opening a file in a *separate window* is not displaying a portion of the file "in the interactive user interface" as required by claims 10-11. The '490 patent describes a unified interface where "the search results are displayed in the list pane or area, and the view area or pane displays the contents of the first item or a selected item from the list area." (EX1001, 7:1-11.) The user does not open the file; the file portion is automatically displayed in the view pane when selected. Lotus Notes operates differently; the user interface displays only search results, and the user must open files in separate windows by double-clicking or pressing

Enter. Indeed, it would defeat the purpose of the invention if the user was required to click and open each file to even view it while at the same time having results of the search updated as the user types. (EX2001, ¶82.) By displaying a portion of the file in the user interface itself as the search results are dynamically updated, the system provides real time feedback. (EX2020, ¶105.) A burdensome process that requires the manual opening of files to view hit highlighting or the broader text itself is inconsistent with the problem that the '490 invention solves.

Petitioner's argument for the highlighting limitation of claim 11 fails for the same reasons. These claims require highlighting a plurality of occurrences of both partial strings while not highlighting intervening text. Because it depends from claim 10, the highlighting must occur in the portion displayed "*in the interactive user interface.*" But as discussed above, Lotus Notes requires opening files to view highlighting, it is not displayed "*in the interactive user interface.*"

Acknowledging that the Lotus Notes-Raskin-Wu combination does not render these claims obvious, Petitioner relies on the knowledge of a POSITA to fill in the missing limitations (Pet., 35-36; EX1003, ¶112; *see also* EX2020, 149:11-19.) Relying on the knowledge of a POSITA to supply a missing limitation is improper gap-filling, *see* 37 C.F.R. 42.104(b)(4).

5. Claims 14-18

Petitioner relies solely on the Lotus Notes references for these claims, but fails to identify any disclosures in these references that render these claims obvious. Petitioner points only to the searching by “Form” functionality as allowing searching within specific fields of a file. (Pet., 37-41; EX1003, ¶¶115-19.) The searching by “Form” functionality is wholly separate from the various search functionalities Petitioner relies on in its claim 1 analysis, the Search bar, Search Builder, and Query Builder functionalities, and is limited to searching for files created using a specific form. (EX1059, 160-61; EX 1061, 144-45.) Further, there is no disclosure of any ability to search on multiple strings entered into a single search field or of incrementally searching for files using the searching by “form” functionality. As Petitioner did not rely on this functionality in its claim 1 analysis, Petitioner never explains how this functionality would be incorporated into the Wilcox-Londergan combination or in the full Lotus Notes-Raskin-Wu combination. To the extent Petitioner impliedly created a combination that includes the searching by “Form” functionality, Petitioner has failed to adequately provide a motivation to combine and cannot show reasonable expectation of success for the same reasons discussed above. (*Supra* §V.B.)

6. Claim 19

Petitioner has not established that the Lotus Notes-Raskin-Wu combination discloses or renders obvious this claim.

As with claims 14-18, Petitioner relies on the searching by “forms” functionality in the Lotus Notes references as allegedly rendering this claim obvious. This argument fails for the same reasons explained above. (*Supra* §V.D.5.) Furthermore, Petitioner does not identify any disclosures in the Lotus Notes references that teach this limitation. Petitioner points first to the fact that the Lotus Notes references default is a full-text search and then separately points to the searching by “form” functionality as allowing a search across a single field. (Pet., 42; EX1003, ¶120.) Petitioner then creates a hypothetical search string that would allegedly meet this limitation, but fails to point to any disclosures showing that this search string could be implemented in the Lotus Notes system. (Pet., 42; EX1003, ¶120.) Indeed, the claimed limitation is functionally different from what is disclosed in the Lotus Notes searching by “form” functionality. (EX2020, ¶27.)

VI. GROUND 2: Petitioner Fails to Show the Challenged Claims Are Obvious Over the Combination of Schwartz (EX1004), Sebastian (EX1005), Barber (EX1006), True (EX1009), and Baeza-Yates (EX1023).

The challenged claims are patentable over Petitioner’s Ground 2 challenge for at least four reasons. First, Petitioner’s Ground 2 combination alleges (incorrectly) that each element of the claimed invention existed in various

embodiments disclosed among the five references in Ground 2, and fails to provide any reason why a POSITA would be motivated to combine *each* of those elements to achieve the claimed invention beyond mere hindsight. *KSR*, 550 U.S. at 418 (insufficient to show each element “was, independently, known in the prior art.”); *Virtek*, 97 F.4th at 888 (holding that it “does not suffice to simply be known” and a “reason for combining must exist.”); *Kinetic Concepts*, 688 F.3d at 1368 (warning against “hindsight reconstruction” for obviousness).

Second, Ground 2 also fails because it relies on non-analogous art (True) from a different field of invention whose teachings are not reasonably pertinent to the particular problem facing the ’490 patent inventors. Third, the Petitioner’s evidence to support its reasonable expectation of success argument is conclusory and, as demonstrated by Mr. Haeberli’s declaration and the prior art, not credible. Fourth, the True reference, which Petitioner relies on to satisfy several claim elements, does not disclose the claim element that Petitioner alleges it does.

A. Petitioner Fails to Provide any Motivation to Combine the Disparate Teachings of Schwartz (EX1004), Sebastian (EX1005), and Barber (EX1006)

Ground 2 fails because Petitioner neglects to provide *any* motivation for why a POSITA would take each of the alleged elements from three different embodiments in its lead references describing the Entourage (Schwartz, Sebastian, and Barber) program to arrive at the claimed invention. Similar to Ground 1 (*supra*

V.A.2), Petitioner’s only identified motivation relates to why a POSITA would combine the references generally: that a POSITA would do so “to obtain a more complete and comprehensive understanding of the Entourage system, its features, and its capabilities.” (Pet., 48.) But there is no single search system in Entourage that discloses all of the features on which Petitioner relies, and instead Petitioner’s obviousness combination improperly requires that a POSITA would pick and choose among distinct features of *three different search systems* described in the three Entourage references without any explanation for why a POSITA would do so. *In re Stepan Co.*, 868 F.3d at 1346 n.1. Specifically, as best can be garnered from the Petition, Petitioner relies on, and implicitly combines, three distinct search tools from the Entourage references: (1) list filtering; (2) Find and (3) Advanced Find. As discussed below, Petitioner misleadingly describes functionalities in the separate search tools it relies on for its obviousness argument as if they occur in a single search tool and never articulates any reason why a POSITA would choose any given feature from one of these three search tools in Entourage and combine it to arrive at the claimed invention.

1. The Entourage References Describe an Email System

The Entourage References (Schwartz (EX1004), Sebastian (EX1005), and Barber (EX1006)) describe the Entourage for Macintosh 2001 email software that includes six components: email client, newsgroup reader, address book, calendar,

tasks, and notes. (EX1004, (i), 1-2; EX1005, (a)-(c), 171; EX1006, (ii), 281; EX2021, 124:8-11.) Because the Entourage system is not indexed (EX2021, 125:13-126:5) all of its search tools necessarily use a sequential scanning or “brute force” search that will “read every word of every message, address, and set of driving directions” when a search is conducted. (EX1006, 385.)

a) Entourage’s List Filtering Tool.

The “List Filtering” tool allows the user to “filter most lists to show items that match simple search criteria (such as the presence of a given text string within an item title).” (EX1004, 8; EX1005, 183; EX1006, 358, 384.) List filtering is only available in Entourage windows displaying lists of Entourage items, such as email messages and address book entries. For example, a user can open an Address Book window, choose an option from the pop-up menu (“Name contains” in the example below), and then start typing a name in the text box:

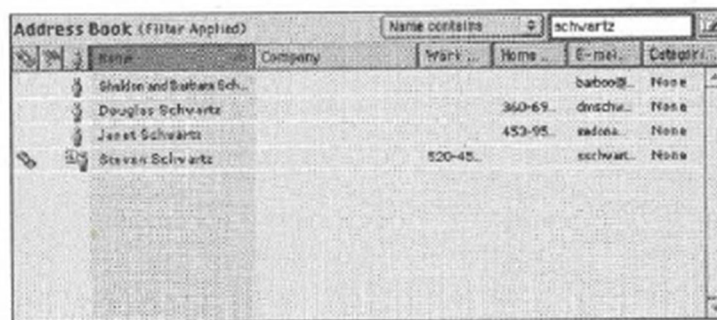


Figure 1.12 As you type the filter text, only matches are shown in the list.

Schwartz Fig. 1.12 (EX1004, 9)

The list is filtered to include, for example, only Address Book entries or message lists that include the characters entered in the text box as the user types. (EX1004, 9, 161.) Petitioner relies on this functionality to disclose searching for files, but fails to explain how filtering a list (which is how Entourage describes this tool) is equivalent to performing a search for files. (EX2020, ¶¶74-76.) Further, in the Entourage references that discuss list filtering, only a single string is entered into the filter text box. (*See, e.g.*, EX1004, 9 (filtering Address Book entries), 161 (filtering a message list); EX1005; EX1006, 384.). The Entourage references do not disclose that List Filter can accommodate multiple partial strings.

b) Entourage's Find Tool

Entourage also includes a “slower” search tool called the Find command. (EX1006, 384-86; EX1004, 9) When a user opens the “Find dialog box,” there is a text entry field and a pull-down menu with options for selecting which types of Entourage items to search for such as “messages.” (EX1004, 9; EX1006, 384-85.) The reason for having a search be limited to specific types of items is to prevent the search from slowing to a crawl. (EX1006, 385.) Indeed, in Entourage if you search “All Items,” you should “be prepared to rent a video during the time Entourage will take the search its entire multimegabyte database.” (EX1006, 385.) Find is a delimited search (EX2021, 129:1-18.), not incremental, and only allows for a single term to be entered.

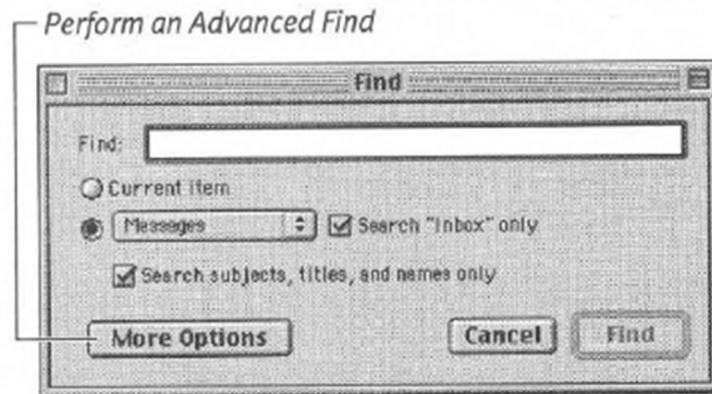


Figure 1.13 You can conduct simple searches by specifying criteria in the Find dialog box.

Schwartz Fig. 1.13 (EX1004, 9)

After clicking the Find button to initiate the search, “[t]he headers of all matching items are displayed in the Search Results window,” as shown below. (EX1004, 10.) The Search Results window is a separate window that pops up to display search results, which do not include any highlighting or file previews. (EX1004, 10.)

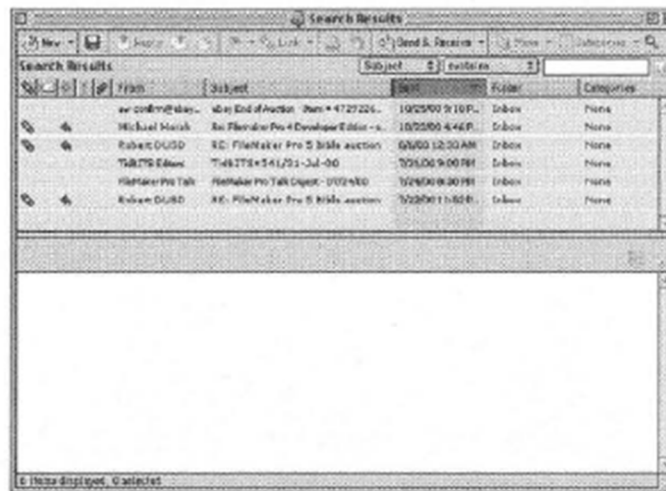


Figure 1.14 When you conduct a Find or Advanced Find, search results are displayed in a separate window. In this example, I searched for messages in the Inbox that contained the word *FileMaker*.

Schwartz Fig. 1.14 (EX1004, 10.)

The Find command can also separately be used to search for text *within a single item* or file. (EX1004, 10.) To do so, when a user is in an individual item such as an email, she can enter Apple-F or Edit-Find to bring up the Find dialog box and then a user can type a string into the text box and click Find. (EX1004, 10.) In this type of search, instead of a Search Results window opening, the window displaying the item scrolls to and highlights the first match, as shown below. (EX1004, 11.)

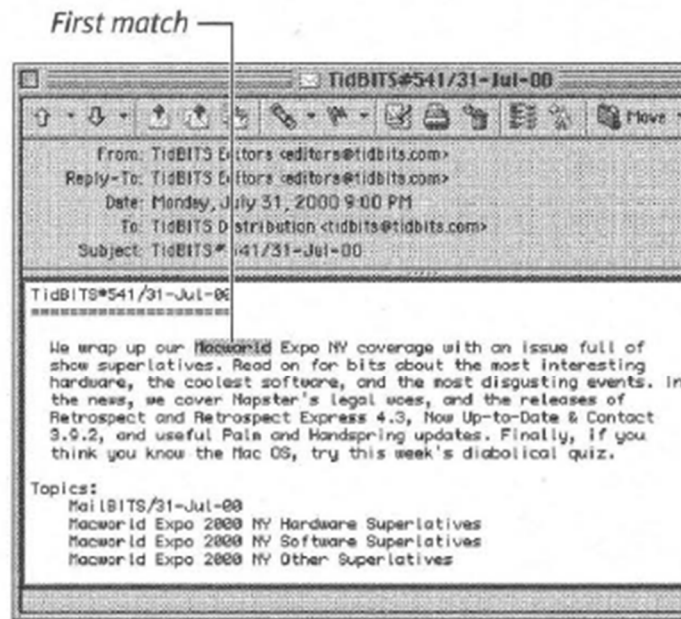


Figure 1.16 When you search for matching text within an item, Entourage highlights the first match, scrolling as necessary to display it.

Schwartz Fig. 1.16 (EX1004, 11)

The scrolling and highlighting features are only present when searching for text *within a single open Entourage item*, such as an email or newsgroup message. (EX2001, ¶124.)

c) Entourage's Advanced Find Tool.

When a user is in Find Functionality, she can press the "More Options" button to open another tool called Advanced Find command, which "allows [a user] to specify multiple search criteria; search multiple item types (messages *and* events, for example); and search any folder, regardless of the one that is currently selected." (EX1004, 11; *see also* EX1004, 385-86.) The Advanced Find, like Find,

is a delimited search that requires a user to click the “Find” button after entering a search string. (EX1004, 10-11; EX1006, 385; EX2001, ¶123.)

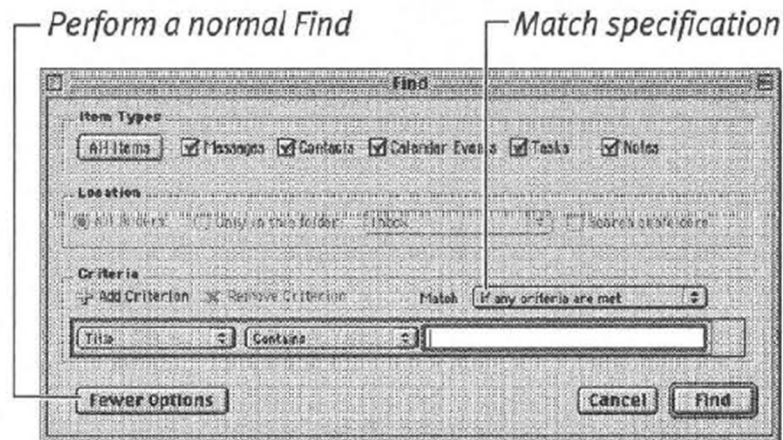


Figure 1.17 The Advanced Find dialog box enables you to search multiple item types simultaneously, specify a location, and use multiple criteria that aren't available to you when you use the Find command.

(EX1004, 11.)

Like the Find command, after hitting the Find button a separate search results window appears. (EX1004, 10.)

d) Petitioner Mischaracterizes the Entourage References

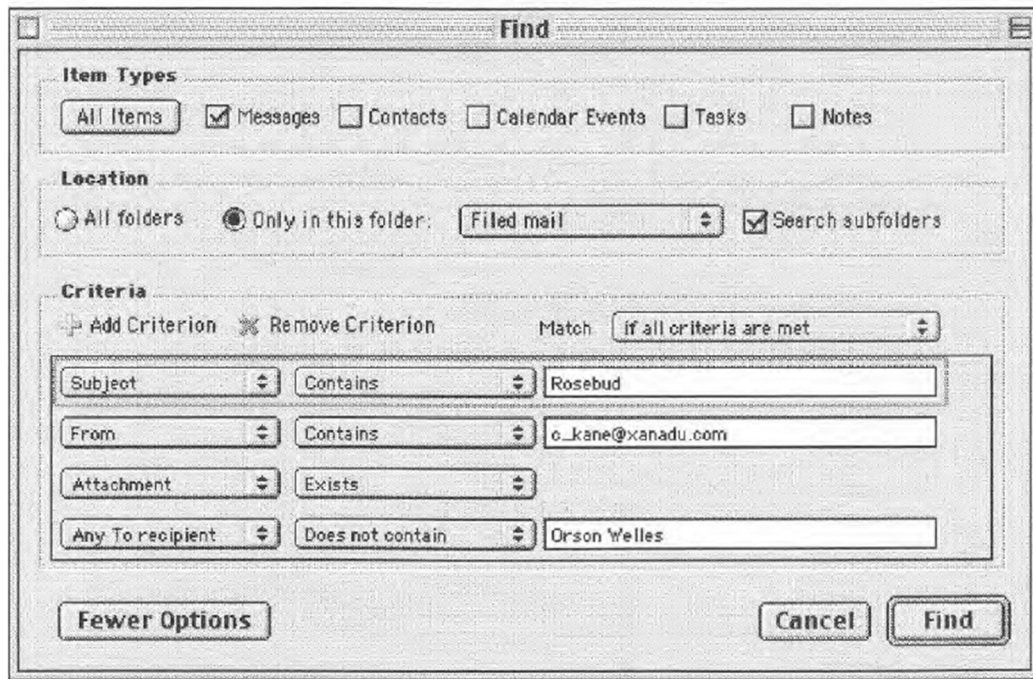
Petitioner mischaracterizes the Entourage references to suggest that certain functionalities from List Filtering, Find and Advanced Find within a group of files, and Find within a single file can all occur within a single search when they cannot. (EX2001, ¶125.) For example, Petitioner mentions the List Filtering tool as purporting to incrementally search files then misleadingly cites to portions of the Entourage references describing the Find command functionality to state that

“[a]fter Entourage finds a document (e.g., email) containing the search string(s), Entourage scrolls to the portion of the document that contains the search string(s) and highlights the search string(s)” as if these functionalities occur in a single search tool. (Pet., 47; EX1003, ¶131.) But Petitioner paints over the differences between Filtering and the Find command, implying that the allegedly incremental Filtering function and the delimited Find Command functionality are somehow the same. (EX2001, ¶127.) According to Schwartz, Entourage allows you to “filter the message list” with the filtering functionality described above, or, separately, to “search for messages with the Find command.” (EX2001, ¶127; EX1004, 161.) These are under separate headings and described separately in Schwartz, and Petitioner fails to explain how a POSITA would understand “filtering” to be “search” at all. (EX2001, ¶127.) The fact that the developers of Entourage chose to keep these functionalities separate would suggest to a POSITA that doing so was for a good reason, such as user experience or performance limitations. (EX2020, ¶74.)

In addition, the scrolling and highlighting features that Petitioner relies on are not associated with the List Filtering tool at all, but only with the Find command. (*Supra* §V.A.3.) And Entourage only scrolls to and highlights the first hit *when using the Find command to search for text within a single Entourage item*, such as an email or newsgroup message; Entourage does not scroll to or

highlight search results when using List Filtering or the Find command to search across multiple items. (*Supra* §V.A.3; EX1004, 11, 163; EX2001, ¶126.)

Furthermore, none of the three Entourage references include any disclosure that a user could enter multiple search strings separated by a string separator (i.e. two words to be searched separately); every example in each of the three references refers to entering a single filter or search term in the text box for the List Filtering, Find, and Advanced Find tools. (*See, e.g.*, EX1004, 8 (“apple”), 9 (“schwartz”), 10 (“macworld”); EX1005, 173; EX1006, 384 (“humor”), 385 (“Rosebud”), 387 (Fig. 11-7 showing individual strings) Petitioner argues that one reference (Barber) “at least suggest[]” incrementally searching on multiple strings by using the Find command to search on the string “Orson Welles.” (Pet., 59; EX2001, ¶153.)



Barber Figure 11-7

(EX1006, 387.) But Petitioner again collapses the distinct tools as if they were one because the Find command only allows for delimited (not incremental) searches. (*Supra* §V.A.3). Also, there is no teaching or suggestion that when a user types “Orson Welles” into the Find command that a search is performed on “Orson” and “Welles” separately as opposed to searching on the full text string “Orson Welles.” (EX2001, ¶153.) Indeed, a POSITA would understand from the fact that Find has multiple boxes, each for separate search terms, that multiple search strings cannot be entered in a single field. (EX2020, ¶156.)

Petitioner inaccurately paints Entourage as allowing a user to do all of these things in a single search. However, a user could incrementally filter a list of

contacts in the List Filter tool. (EX2001, ¶155.) Or, in Find and Advanced Find, a user can type a single string, hit enter, and wait to receive a list of search results, but without any highlighting to the search results. If a user goes into yet a different window by opening a single item and then utilizing the Find toolbar again, the user can enter a single search string, hit Find, and the system will then highlight the first match within the open item. (EX2001, ¶¶124.) Accordingly, Petitioner's assertion that the claimed invention is "nothing more than the obvious inclusion of well-known features in a then-existing incremental search tool" (EX1003, ¶¶125-130) is incorrect. Petitioner has not identified any "then existing incremental search tool" that contained all of the limitations of the claimed inventions that are allegedly disclosed in the Entourage References, which solely comprise the problematic sequential scanning and/or delimited search methods that the '490 invention solves. (*See* EX2001, ¶130.)

2. Petitioner Has Not Met Its Burden to Show Motivation to Combine the Different Search Tools Within the Entourage References to Arrange Them According to the Claimed Invention

Petitioner's only stated motivation to combine the Entourage references and their teachings is the same as what it proposed for the Lotus Notes references: a POSITA would want "to obtain a more complete and comprehensive understanding of the Entourage system, its features, and its capabilities." (Pet., 48; EX1003, ¶132.) And this motivation is conclusory and deficient for the same

reasons as those in Ground 1 for the Lotus Notes references; it misses the critical point of why a POSA would have been motivated to combine the *elements* from those teachings in the way that the inventors did. Petitioner fails to disclose why a POSITA would choose to combine: incremental filtering; delimited search of Find and Advanced Find tools, which are characterized as “slow,” that only allow for a single search term in each text box; and highlighting of search term hits that occurs in the Find tool when conducting a delimited search within a single item (and not across files). For the same reasons discussed in Ground 1 (*supra* §V.A.), merely pointing out the existence of elements of the invention in the prior art is insufficient to demonstrate obviousness.

B. True is Non-Analogous Art

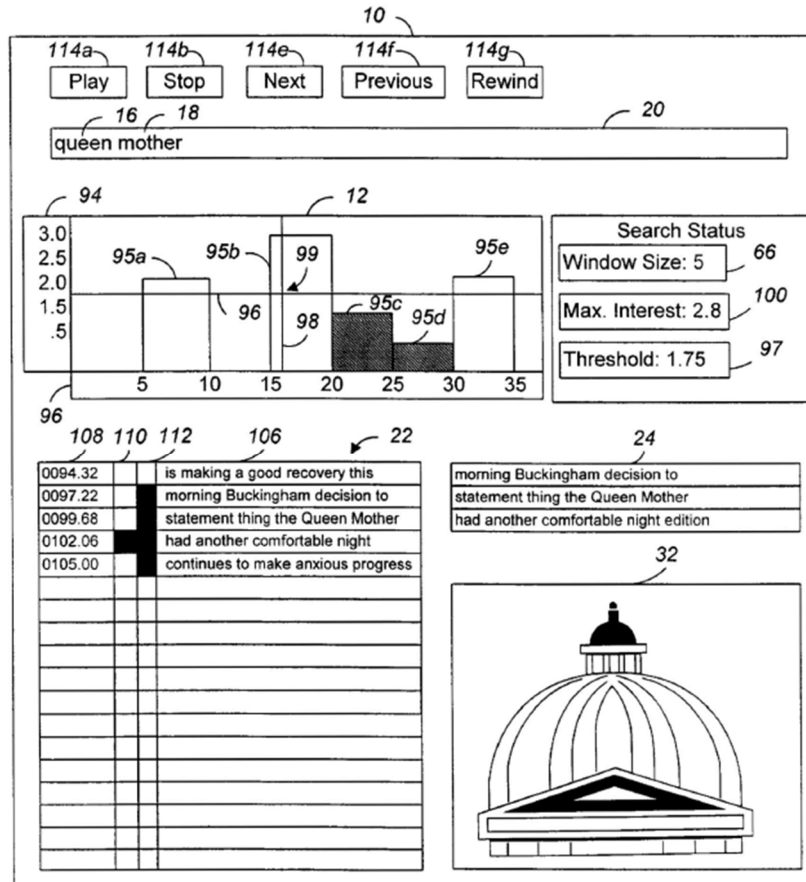
Ground 2 also fails because one of the primary references, True, is not analogous art. Petitioner relies on True to satisfy the missing element in the Entourage references of “receiving two partial search strings in a single search field.” (Pet., 43.) Because True is not in the same field of endeavor as the ’490 patent, and its disclosed invention is not reasonably pertinent to the particular problem to be solved by the claimed invention, it is not analogous art. (EX2020, ¶¶77-89.)

1. True Relates to Graphical Representations and User Navigation of Audio Files

True is a system for visualizing the contents of a single underlying file, such as an audio file, using a text transcription of that file that is inherently imprecise due to transcription errors. (*See* EX1009, 1:18-27, 1:38-45, 1:61-63, 6:38-44; EX2001, ¶¶134-38; EX2020, ¶¶78-81.) In the system, a user can navigate throughout the audio file that corresponds to a given part of the transcribed text, edit the transcription when errors are detecting, and locate “material of interest” in the text file and therefore the underlying audio file. (EX1009, 1:38-45.)

While True refers to “searching” the transcript file, the purpose of True is not to definitively locate where a search string actually appears in the text file. Instead, it uses “confidence levels” to show a user how likely it is that the underlying audio file actually contains a given word in the text file (instead of being an incorrectly recognized word). (EX1009, 4:34-38.) Each recognized word in the transcript includes a confidence level “indicating an estimated likelihood that the recognized word is correct.” (EX1009, 4:21-33.) Based on those confidence levels, True is able to show a user “which segments of the transcript relate to topics of interest” by summing the confidence levels of each word in a given time period of the audio file. (EX1009, 2:10-16, 3:40-43, 4:55-65.)) True’s query results

consist of a graphical display showing the “interest level” of “transcript windows,” as shown below:



True Figure 2 (EX1009)

Because of how True calculates the results of its search, it is possible for the results of a query (the “transcript window”) to not actually include any instance of the words in the query. (EX2020, ¶79.) True’s result window, therefore, provides a visual indication of which segments of the transcript (transcript windows) *might* contain *at least one* of the words in the search query. True does not provide a list of search results (let alone files) pointing to where all the instances of the words in

the search query appear. (EX2001, ¶134.) Thus, True is not searching and identifying files or groups of files that correspond to two partial search strings, as required by the claims. True's search is also done sequentially by scanning each character in the text of the transcript for the query words. (*See* EX1009, 4:21-49; EX2020, ¶79.)

2. True is Not in the Same Field of Endeavor

The field of endeavor for the '490 patent is systems and methods for incrementally searching for and identifying indexed files. (EX1001, Title, 1:38-39; 2:20-32; 3:46-58; 15:26-61; 25:21-38; 26:24-36; 29:31-49.) True does not relate to searching for files at all, does not relate to indexed systems, and does not relate to searching files for any given search strings. Instead, True provides a method for executing a search query on a single text file where the text does not necessarily correspond to the underlying raw data, and where the results of the query only indicate how likely it is that the text file accurately captures the underlying raw data. (EX2020, ¶81.) In other words, a query "hit" in True will not necessarily even identify a part of the transcript where the query word was actually said. (EX2020, ¶83.) Thus, True's teachings are not relevant to the '490 patent's invention of accurately searching and identifying groups of indexed files that actually correspond to the search queries. (EX2020, ¶81.)

Petitioner asserts that True is analogous art because it is in the same “field of endeavor”: “providing a user interface through which a user can enter search strings, and locating files **and/or portions of files** containing the search strings.” (EX1003, ¶134³ (emphasis added).) But the field of endeavor for the claimed invention is not methods and systems for entering search strings in search fields, it is directed to *searching* and *identifying indexed files* that include the partial search strings entered in the search field. (Pet., 49-50; EX2020, ¶¶82-83.) All claims of the patent require that a “first set of documents” and a “second set of documents” corresponding to the search strings are identified. The fact that True and the ’490 patent both relate to search, at the highest level, is not sufficient to be considered the same field of endeavor. *Wang Lab ’ys*, 993 F.2d at 864 (finding art non-analogous where both related to computer memories, but the prior art “involves memory circuits in which modules of varying sizes may be added or replaced; in contrast, the subject patents teach compact modular memories”). Nor is the fact that they both mention queries or search strings sufficient. *In re Clay*, 966 F.2d 656, 659 (Fed. Cir. 1992) (even though both related to petroleum, invention relating to the extraction of crude petroleum was not in the same field of endeavor

³ The bolded language does not appear in Petitioner’s other descriptions of the field of endeavor (*See* Pet., 12, 16-17, 44), demonstrating that Petitioner is trying to shoehorn True into a different, expanded, field of endeavor to even argue it is analogous.

as an invention relating to the storage of refined petroleum). True does not search files, groups of files, does not search indexed groups of files, and in fact does not even necessarily return results that would have the query word(s) in them.

(EX2020, ¶¶82-83.) As such, True is not in the same field of endeavor as the '490 patent. (EX2020, ¶¶82-83.)

3. True is not Reasonably Pertinent to the Problem to be Solved by the '490 Patent

True is not reasonably pertinent to the problem to be solved by the inventors because it would not “logically [] have commended itself to an inventor's attention in considering his problem.” *In re Clay*, 966 F.2d at 659.

Here, the problem solved by the '490 patent was slow and inefficient searching. (EX1001, 1:43-2:16.) The search functionality disclosed in True suffers from the exact same problems that the '490 patent seeks to solve. (EX2020, ¶¶84-85.) As discussed above, True is a sequential scan of a single file (not across a group of files), and its “search results” are portions of the file that may not even contain one of the search strings. (*Supra* §VI.B.1.) This sequential or brute force scanning known to be slow when searching across multiple files. (EX2020, ¶85.)

To the extent Petitioner asserts that True is analogous because it relates to solving the issue of searching generally, the Federal Circuit explained in *Circuit Check* that just because the prior art relates in a general sense to the problem

sought to be solved by the claimed invention, that does not mean it is reasonably pertinent. *Cir. Check Inc. v. QXQ Inc.*, 795 F.3d 1331, 1335-37 (Fed. Cir. 2015) (prior art related to rock carvings, engraved signage, and a machining technique for marking not pertinent to patent seeking improvements on markings in circuit board tester interface plates); *see also In re Klein*, 647 F.3d 1343, 1352 (Fed. Cir. 2011) (holding prior art references related to receptacles and removable dividers not analogous to partitioned container as “neither of the references shows a movable divider or the ability to prepare different ratios”). Instead, the focus is on whether the inventor would have consulted that prior art to solve the problem he was facing. Here, the ’490 inventors were seeking to solve the problem of the slow, linear searching of the prior art when searching large file sets and they would not consult True, which does not search files, groups of files, and only discloses the linear, line by line search that was already known in the prior art. *Innovation Toys, LLC v. MGA Ent., Inc.*, 637 F.3d 1314, 1321-23 (Fed. Cir. 2011) (holding prior art non-analogous when it pertained to methods for separating the solid object contents of containers, whereas claimed invention related to compartments for liquids).

C. A POSITA Would Not Have Been Motivated to Combine the Entourage References with True

Even if True were analogous prior art, which it is not, a POSITA would not have been motivated to combine the Entourage references with True.

Petitioner admits the Entourage references fail to disclose at least (1) the receiving a first partial search string in a first search field; (2) receiving a second partial search string in the first search field; and (3) wherein the first and second partial search strings are separated by a string separator character in the first search field limitations. (Pet., 43; EX2001, ¶¶86-87.) Petitioner relies on True to fill that gap. (Pet., 43.) Petitioner may also relies on True as allegedly teaching a POSITA to “incrementally locate (i.e., locate as the user types or modifies their search query) files.” (Pet., 67.) Petitioner’s alleged motivation that a POSITA would have to combine Entourage references with True is that multi-string search would return more relevant results than single-string query (Pet., 53).

Petitioner highlights a statement in one Entourage reference to support its argument, but that actually shows why a POSITA would *not* look to True if they were starting with the Entourage references. “Searching for two or more keywords - both of which *must* appear in the results—is an excellent strategy for doing precise searches and greatly reducing the number of search results you have to consider.” (Pet., 61 (emphasis in original); EX1057, 27) A POSITA would not be

motivated to look to True as it is not directed to finding precise matches to queries, and is designed to return results even where both search strings do not appear in the result. (EX2001, ¶140.) And as discussed above, True is not even analogous art to the '035 patent, suggesting that a POSITA would not have looked to True to modify the Entourage references. (*Supra* §VI.B.)

Similar to Dr. Terveen's hindsight based opinion in Ground 1 (*supra* §V.A), he did not consider how True was implemented, or the downsides and limitations of True. (EX2021, 148:3-149:12.) Instead, he only took from True that it disclosed "incremental search" and "multiple search strings in a single search field." (EX2021, 149:13-150:13 ("Anything else of True or this aspect of True the fact that it had these confidence factors was irrelevant for me.")) Even if it was proper for Petitioner to ignore all the other teachings in True and rely solely on the idea of incremental search to be implemented into some undisclosed search system from Entourage, a POSITA would not be motivated to do so for the same reasons that a POSITA would not be motivated to combine the teachings of Raskin with the Lotus Notes system: doing so would require an increased index which would slow down searching or exceed a consumer's computing power, or would require linear search which would be so slow as to no longer be incremental search at all. (*See supra* §V.A.3.b; EX2020, ¶¶86-87.) A POSITA is not motivated to combine prior art that would not be practical to end users, or would be expected to reduce

functionality of the system. (EX2020, ¶¶86-87.)

1. True Does Not Teach Incrementally Searching a Set of Files for Multiple Search Strings.

Petitioner asserts that a POSITA would look to True’s “teachings to [incrementally locate files] for multiple search strings entered in the search field.” (Pet., 61) But True does no such thing. As discussed above, True does not search files at all, and its “search” is designed to return results that do not necessarily even contain the words in the query. (EX2001, ¶140.) The output of True’s “search” are “transcript windows” which are merely portions of the file that are being searched that might have one or more of the query words present (EX1009, 3:37-40.) Thus, contrary to the Petitioner’s argument (Pet., 53-54), True does not disclose searching for the presence of multiple search strings in files and identifying files that contain multiple search strings, as required by the claims. (EX2001, ¶140.)

Petitioner, relying solely on its expert’s conclusory testimony, contends that True discloses searching for files because True allegedly teaches that searching within a transcript (i.e., a single file) of an audio file returns as results *portions of the transcript*, referred to as “transcript windows.” (Pet., 69; EX1003, ¶165.) Dr. Terveen equates “files” with “transcript windows” without providing any citations to intrinsic evidence, and without explanation or justification for his implicit construction. (EX1003, ¶165.) True’s transcript windows are “portions of the

transcript” defined by a window size in seconds (e.g., five seconds). (EX1009, 3:40-43, 4:55-65.) and a POSITA would understand that they are not “files” as described in the ’490 patent. (*Supra* §IV; EX2001, ¶37.)

2. True is Directed to a Wholly Different Purpose than Entourage and Its “Interest Level” Method is Inherently Imprecise.

Petitioner’s argument that multi-string search would return more relevant results is conclusory and is contradicted by the teachings of True. (Pet., 56; EX2001, ¶141.).

First, that “a multi-string search would return more relevant results than a single-string query,” fails to explain why a POSITA would look to True specifically to improve Entourage in this way. (Pet., 53; EX2001, ¶138.) Indeed, as discussed above (*supra* §VI.B), Entourage and True are directed to completely different purposes and POSITA would not have considered True when seeking to modify Entourage as stated. *See In re Klein*, 647 F.3d 1343, 1350-51 (Fed. Cir. 2011) (“If [a reference] is directed to a different purpose, the inventor would accordingly have had less motivation or occasion to consider it.”). Rather than filtering or searching for email messages, address book contacts, or text within a single message like in Entourage, True’s search capabilities are inherently imprecise and its invention is directed to visualizing portions of a single text file based on an underlying audio file that may be of interest to a user, including so the

user can correct errors in the transcript. (EX1009, 1:55-60; 3:20-33, 3:43-46; EX2001, ¶131.) True portends to solve the problem of how to identify information you are seeking when the underlying data that you are searching (i.e. a transcript of an audio file created by a speech recognizer) does not accurately represent what was actually said in the audio file. (EX1009, 1:18-27; EX2001, ¶140.) There is no suggestion that Entourage had the problem True seeks to solve (how to locate information when the underlying data you are searching is inaccurate). (EX1009, 1:18-27.) Finally, a POSITA would not look to True given the different respective corpus of data on which searches are performed in each system. (EX2001, ¶139.) If a POSITA wanted to make the search functionality in Entourage more precise (as Petitioner alleges), it would not have made any sense to turn to True's inherently imprecise interest level method. (EX2001, ¶139.)

3. Petitioner Fails To Explain How A POSITA Would Modify Any Particular Search Tool in Entourage To Include True's Disclosure.

Similar to the deficiencies in Ground 1, in Ground 2 Petitioner fails to describe with particularity how a POSITA would modify any specific search feature in Entourage to incorporate True's disclosure of allegedly receiving two "search strings" in a single field, or with True's alleged disclosure of incremental search, to the extent Petitioner relies on True for this limitation. (EX2001, ¶141.) Specifically, Petitioner never describes what specific modifications a POSITA

would make other than saying Entourage's search field would receive two partial search strings, or would implement incremental search, thereby leaving unexplained how the combination would be implemented. (EX2001, ¶141.)

As discuss above (*supra* §VI.A), Petitioner fails to identify why a POSITA would be motivated to take any particular element from the various search tools in Entourage and combine them in the way the inventors did. Petitioner fails to explain why a POSITA would consider using True's alleged incremental search with Entourage's Find and Advanced Find features, given that searching through files and text in Entourage was already known to be "slower" than List Filtering, and take up to hours when using the Find and Advanced Find tools. (EX2001, ¶143; EX1006, 385. Petitioner's analysis does not include any explanation as to why a POSITA would consider implementing True's confidence/interest level search method in any of the Entourage search tools or how a POSITA would combine the references to avoid further slowing the search down. This is insufficient to satisfy Petitioner's burden. *In Re Magnum Oil*, 829 F.3d at 1380.

D. A POSITA Would Not Have a Reasonable Expectation of Success in Combining the Ground 2 References

Similar to Ground 1 (*supra* §V.B), Petitioner makes no meaningful attempt to explain why a POSITA would have a reasonable expectation of success in combining the various elements allegedly disclosed in the Ground 2 references,

stating only that “it would have required only simple and straightforward software modifications.” (Pet., 54; EX1003, ¶¶140; EX2020, ¶88.) This conclusory and unsubstantiated statement is insufficient as a matter of law. *See Juniper*, 2024 WL 3517862, at *4.

As discussed above, the specific incremental list filtering from Entourage and the incremental “search” in True are fundamentally different from, and incompatible with, the delimited searches that can be done in Find and Advanced Find. As discussed above (*supra* §VI.A.1), Entourage repeatedly refers to its delimited search tools (Find and Advanced Find) as being slower and, if the search is not limited to a narrower set of files, can take up to hours to receive results. (EX1006, 385.) As Mr. Haeberli and Dr. Terveen both agree, implementing incremental search requires a much larger index which can slow down the search process, even to the point where the search may no longer be responsively ‘incremental.’ (EX2020, ¶88; EX2021, 49:13-52:19.). Petitioner has not shown that a POSITA would have a reasonable expectation of success.

E. The Entourage-True-Baeza Yates Combination Does Not Render Obvious All Limitations of Claim 1

1. [1.8-1.10]

Petitioner has not established that its Entourage-True-Baeza-Yates combination renders obvious these limitations. Petitioner argues that the Entourage references “at least *suggest*[] incrementally searching for files containing two

partial search strings entered in a single search field,” citing the references’ statements about “searching for text strings.” (Pet., 66.) Nothing in the cited portions on the references hints at entering two search strings in the single search field. (EX2020, ¶89.) Petitioner relies on True for incrementally searching on two strings in a single search field. (Pet., 67, EX1003, ¶¶155-157.) However, True does not disclose searching “a set of documents” to identify a “first set of documents” and a “second set of documents.” (*Supra* §VI.B.1) Petitioner does not explain why a POSITA would be motivated to combine, or how to implement, True’s teachings about sequential scanning of a transcription of an audio file to identify transcript windows that may or may not contain one or more of the search strings in an indexed search system designed to search for multiple files; nor does Petitioner establish that a POSITA would have had an expectation of success in doing so. (*Supra* §VI.D.)

Finally, Petitioner engages in improper gap filling by relying on the knowledge of a POSITA to fill in this missing limitation. (Pet., 62-63; EX1003, ¶157; EX2021, 160:1-161:9; 37 C.F.R. 42.104(b)(4).)

F. The Entourage-True-Baeza Yates Combination Does Not Render Obvious the Dependent Claims

1. Claims 2-20

Petitioner’s combination for these dependent claims fails for at least the reasons explained in Sections VI.C-E. (*Supra* §§VI.C-E.)

2. Claim 4

Petitioner relies solely on the knowledge of a POSITA to meet this limitation. (Pet., 31-32; EX1003, ¶¶180-181.) Relying on the knowledge of a POSITA to supply a missing limitation is improper gap-filling, *see* 37 C.F.R. 42.104(b)(4). Petitioner does not provide any evidence that a POSITA would have known to include associations with attribute fields in an index, fails to acknowledge there are many different possible index structures, and points to no disclosures in the cited art of the structure disclosed in this claim limitation. (Pet., 74-76; EX1003, ¶¶180-181; EX2020, ¶90.)

3. Claim 5-7

Petitioner relies solely on the knowledge of a POSITA to meet these limitation. (Pet., 76-78; EX1003, ¶¶182-185.) This is improper gap-filling, *see* 37 C.F.R. 42.104(b)(4). Petitioner provides no motivation for a POSITA to implement these features and provides only a conclusory statement that the necessary “software modifications” would have been “trivial” in support of reasonable expectation of success. (Pet., 77-78; EX1003, ¶185.) This is insufficient to meet Petitioner’s burden to establish obviousness for the same reasons explained above. (*Supra* §§VI.A-C; EX2020, ¶91.)

4. Claims 11

Petitioner relies solely on the Entourage references to disclose the highlighting limitation of claim 11 (Pet., 80-84), citing only to using the Find and Advanced Find command functionality to search for a single string within a single Entourage item. (Pet., 80-83; EX1003, ¶¶189-90; *see also* EX1004, 10-11, 162-63.) But any highlighting in Entourage occurs only in an open file during a search of that open file. The highlighting is not shown in a portion of the file displayed via the user interface. (EX1004, 11.)

Petitioner admits that the Entourage-True-Baeza-Yates combination does not render obvious highlighting “*a plurality of occurrences*” of the search strings, and instead relies on the knowledge of a POSITA to fill in the missing limitation (Pet., 84, 93; EX2020, ¶92; EX1003, ¶¶192-94; EX2021, 160:161:9.) Relying on the knowledge of a POSITA to supply a missing limitation is improper gap-filling, *see* 37 C.F.R. 42.104(b)(4).

5. Claims 14-18

In addition to Petitioner’s mischaracterization of the Find and Advanced Find features (*supra* §VI.A.1.d), Petitioner uses the knowledge of a POSITA “as a wholesale substitute for reasoned analysis and evidentiary support,” which the Federal Circuit has cautioned against. *DSS Tech. Mgmt., Inc. v. Apple Inc.*, 885 F.3d 1367, 1374 (2018).

The Petition states this “was trivial and well within the ability of a POSITA and would have been obvious to try.” (Pet., 87-88.) The Petition’s only citation for this conclusory statement is an identical, unsupported statement in Dr. Terveen’s declaration. (Pet., 88 (citing EX1003, ¶198).)

Petitioner offers no explanation as to why a POSITA would have been motivated to modify Entourage, *an email client*, to search *non-email* files, and instead relies on the knowledge of a POSITA to supply the missing limitation. (Pet., 87-88); EX1003, ¶198.) This is improper gap-filling, *see* 37 C.F.R. 42.104(b)(4).

6. Claim 19

Petitioner admits “the Entourage references do not expressly disclose searching all textual content of the files for the first search string and only a single field of the files for the second search string within the same search field.” (Pet., 90.) Acknowledging this gap, Petitioner improperly relies on the knowledge of a POSITA to fill in the missing limitation. (Pet., 90; EX1003, ¶201; *see also* EX2021, 149:11-19; 37 C.F.R. 42.104(b)(4)).

VII. Conclusion

For the foregoing reasons, Patent Owner respectfully requests that the Board reject Petitioner’s arguments.

Date: November 3, 2025

Respectfully submitted,

/Lori A. Gordon/

Lori A. Gordon (Reg. No. 50,633)
Goodwin Procter LLP
1900 N Street, N.W.
Washington, D.C. 20036
Phone: (202) 346 4435
Gordon-ptab@goodwinlaw.com

Lead Counsel for Patent Owner

CERTIFICATE OF WORD COUNT

Pursuant to 37 C.F.R. § 42.24(d), the undersigned certifies that the foregoing **PATENT OWNER'S RESPONSE** complies with the complies with the type-volume limitation of 37 C.F.R. §§ 42.24(a)(1)(i) and 42.24(b)(2) because it contains 13,998 words according to the word-processing system used to prepare this Response, excluding the words in the Table of Contents, Table of Authorities, List of Exhibits, Certification Under §42.24(d), and Certificate of Service, as set forth in 37 C.F.R. § 42.24(a)(1).

Date: November 3, 2025

/DRAFT/

Lori A. Gordon (Reg. No. 50,633)
Goodwin Procter LLP
1900 N Street, N.W.
Washington, D.C. 20036
Phone: (202) 346 4435
Gordon-ptab@goodwinlaw.com

Lead Counsel for Patent Owner

CERTIFICATE OF SERVICE

A copy of this Patent Owner's Response has been served on Petitioner's attorneys of record as follows via electronic mail:

Todd M. Siegel, Reg. No. 73,232
todd.siegel@klarquist.com

Amy C. Haspel, Reg. No. 78,385
amy.haspel@klarquist.com

Andrew M. Mason, Reg. No. 64,034
andrew.mason@klarquist.com

Samuel B. Thacker, Reg No. 78,633
samuel.thacker@klarquist.com

KLARQUIST SPARKMAN, LLP
121 SW Salmon Street, Suite 1600
Portland, Oregon, 97204
503-595-5300 (phone)
503-595-5301 (fax)

Msft-X1-IPRs@klarquist.com

Attorneys for Microsoft Corporation

Date: November 3, 2025

/DRAFT/

Lori A. Gordon (Reg. No. 50,633)
Goodwin Procter LLP
1900 N Street, N.W.
Washington, D.C. 20036
Phone: (202) 346 4435
Gordon-ptab@goodwinlaw.com

Lead Counsel for Patent Owner