Trials@uspto.gov
Tel: 571-272-7822

Paper 48
Entered: June 18, 2026

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICROSOFT CORPORATION,
Petitioner,

v.

X1 DISCOVERY, INC.,
Patent Owner.

IPR2025-00254
Patent 10,552,490 B2

Before WILLIAM V. SAINDON, SCOTT C. MOORE, and LISA A. MURRAY, *Administrative Patent Judges.*

MURRAY, *Administrative Patent Judge*, for the Board.
SAINDON, *Administrative Patent Judge*, concurring.

JUDGMENT
Final Written Decision
Determining That No Challenged Claims Are Unpatentable
35 U.S.C. § 318(a)

MURRAY, *Administrative Patent Judge*:

I. INTRODUCTION

We instituted *inter partes* review pursuant to 35 U.S.C. § 314 to review claims 1–20 of U.S. Patent 10,552,490 B2 ("the '490 patent") owned by X1 Discovery, Inc. ("Patent Owner"). We have jurisdiction under 35 U.S.C. § 6(c). This Final Written Decision is entered pursuant to 35 U.S.C. § 318(a) and 37 C.F.R. § 42.73. For the reasons discussed below, Petitioner Microsoft Corporation ("Petitioner") has not shown by a preponderance of the evidence that any of the challenged claims of the '490 patent are unpatentable.

II. BACKGROUND

Petitioner filed a Petition requesting *inter partes* review of claims 1–20 of the '490 patent. Paper 1 ("Pet."). Patent Owner filed a Request for Discretionary Denial (Paper 7) and a Preliminary Response (Paper 8). Petitioner filed an Opposition to the Request for Discretionary Denial, (Paper 9), Patent Owner filed a Reply in Support of the Request for Discretionary Denial (Paper 11), and Petitioner filed a Sur-Reply in Support of its Request for Discretionary Denial (Paper 12). The then-Acting Director denied the request for discretionary denial and referred the Petition to the Board to issue a decision on institution addressing the merits of the Petition. Paper 13.

After considering the merits of the Petition and the arguments presented against institution by Patent Owner, we instituted *inter partes* review of all the challenged claims of the '490 patent on all grounds asserted in the Petition. *See* 37 C.F.R. § 42.108(a); Paper 14 (Decision Granting Institution ("Dec.")).

During the trial phase, Patent Owner filed a Response to the Petition. Paper 21 ("PO Resp."). Petitioner filed a Reply ("Reply Br.") (Paper 28), and Patent Owner filed a Sur-Reply ("Sur-Reply Br.") (Paper 38). Oral argument was held on April 27, 2026, the transcript of which is filed in the record. Paper 47 ("Tr.").

A. Related Matters

The parties identify the following as matters that may affect, or be affected by, a decision in this proceeding:

> *X1 Discovery, Inc. v. Microsoft Corp.*, Civil Action No. 8:23-cv-02415 (C.D. Cal.) (filed Dec. 19, 2023).
>
> *X1 Discovery, Inc. v. Acer Inc.*, Civil Action No. 2:23-cv-10674 (C.D. Cal.) (filed Dec. 20, 2023).
>
> *X1 Discovery, Inc. v. ASUSTeK Computer Inc.*, Civil Action No. 2:23-cv 10672 (C.D. Cal.) (filed Dec. 20, 2023).
>
> *X1 Discovery, Inc. v. Dell Technologies Inc.*, Civil Action No. 2:24-cv 04109 (C.D. Cal.) (filed Dec. 20, 2023).
>
> *Certain Computing Devices Utilizing Indexed Search Systems and Components Thereof*, USITC Inv. No. 337-TA-1389 (filed Dec. 19, 2023).
>
> *Microsoft Corp. v. X1 Discovery, Inc.*, IPR2025-00253 (instituted July 22, 2025).
>
> *Microsoft Corp. v. X1 Discovery, Inc.*, IPR2025-00255 (instituted July 22, 2025).

Pet. xiii–xiv; Paper 3, 2–3 (Patent Owner's Mandatory Notice).

B. The Patent-at-Issue

1. Prosecution History

The '490 patent, titled "Methods and Systems for Search Indexing," issued on February 4, 2020, from an application filed on April 24, 2017, as one of a series of continuation applications dating back to September 3,

2003, and claiming the benefit of five provisional applications filed between September 23, 2002, and June 13, 2003. Ex. 1001, codes (22), (45), (54), (60), (63). The ’490 patent is a great-grandchild of U.S. Patent No. 7,370,035, which is being concurrently challenged in IPR2025-00253. *See* Ex. 1058 (’035 patent). It is also a child of U.S. Patent No. 9,633,139, concurrently challenged in IPR2025-00255. *See* Ex. 1075 (’139 patent). As issued on February 4, 2020, the ’490 patent contains twenty claims, of which only claim 1 is independent. Ex. 1001, 29:55–32:24.

2. Overview

The ’490 patent “relates generally to data processing, and in particular to systems and methods for locating data.” Ex. 1001, 1:38–39. The patent identifies various drawbacks of conventional search application programs. *Id.* at 1:43–2:16. According to the ’490 patent, such applications were “often . . . slow and cumbersome to use,” and many “require[d] the user to type in a search term, click on a search button, and review the results,” in some cases repeatedly. *Id.* at 1:43–50, 1:62–2:16. In addition, according to the ’490 patent, many search application programs only provided for limited search filtering, and some did not include search features directed to specific types of search targets, necessitating the use of separate tools to search for email, Web pages, and files. *Id.* at 1:50–61.

In view of such drawbacks, in one exemplary embodiment the ’490 patent “provides for incremental or reactive searching of a variety of search targets,” including “files, emails, email attachments, Web pages, specific databases, and/or the like.” *Id.* at 2:20–24. Because the search is performed incrementally, according to the ’490 patent, the search results are “provided or narrowed substantially immediately after each character in a search string

is entered" by the user, and, "[t]hus, the user is provided with substantially immediate feedback as the search string is being entered, and so can quickly decide on the desirability of entering additional search characters, entering a new search string, or deleting one or more search string characters." *Id.* at 2:24–32. The '490 patent also describes, for example, an embodiment that includes receiving at least two search strings as they "are being entered into the same search field," and "incrementally locating at least a first document that has a[t] least a first word that begins with the first search string and a second word that begins with the second search string." *Id.* at 2:41–47.

C. Illustrative Claim

Of the challenged claims of the '490 patent, only claim 1 is independent. We reproduce claim 1 below, with Petitioner's bracketed identifiers and formatting added (*see* Ex. 1040):

> [1.1] A computing system comprising:
>
> [1.2] one or more hardware computer processors;
>
> [1.3] a display device viewable by a user of the computing system;
>
> [1.4] a computer readable storage medium storing:
>
> > [1.5] an index data structure comprising associations between each of a plurality of character strings to associated files of a plurality of files in which the character strings are included; and
> >
> > [1.6] software instructions executable by the one or more hardware computer processors causing the computing system to:
> >
> > > [1.7] cause display of an interactive user interface on the display device, the interactive user interface including at least a first search field;
> > >
> > > [1.8] receive a first partial search string in the first search field;

[1.9] receive a second partial search string in the first search field;

[1.10] wherein the first and second partial search strings are separated by a string separator character in the first search field;

[1.11] searching the index data structure for a first one or more of the plurality of files each including at least one string that corresponds to the first partial search string;

[1.12] searching the index data structure for a second one or more of the plurality of files each including at least one string that corresponds to the second partial search string; and

[1.13] updating a set of matching files of the plurality of 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.

Ex. 1001, 29:56–30:28.

D. Asserted Grounds and Testimony of Record

Petitioner asserts the following prior art references:

Reference		**Effective Date**	**Exhibit**
Wu	US 5,991,756	Nov. 23, 1999	1019
True	US 6,112,172	Aug. 29, 2000	1009
Wilcox	Adam A. Wilcox, *PC Learning Labs Teaches Lotus Notes 3.0*	1993	1059
Londergan	Stephen Londergan & Pat Freeland, *Lotus Notes R5 for Dummies*	1999	1061

Reference		Effective Date	Exhibit
Baeza-Yates	Ricardo Baeza-Yates & Berthier Ribeiro-Neto, *Modern Information Retrieval*	1999	1023
Raskin	Jef Raskin, *The Humane Interface*	2000	1021
Schwartz	Steve Schwartz, *Visual QuickStart Guide: Entourage 2001 for Macintosh*	2001	1004
Sebastian	Bonita Sebastian et al., *Microsoft Office 2001: A Hands-on Guide for Macintosh*	2001	1005
Barber	Nan Barber & David Reynolds, *Office 2001 for Macintosh: The Missing Manual*	2001	1006

See Pet. 2. Petitioner refers to Wilcox and Londergan collectively as "the Lotus Notes references." *Id.* Petitioner refers to Schwartz, Sebastian, and Barber collectively as "the Entourage references." *Id.*

Petitioner asserts the following grounds of unpatentability:

Ground	Claims	35 U.S.C. §	References
1	1–20	103(a)[1]	Wilcox, Londergan, Raskin, Wu
2	1–20	103(a)	Schwartz, Sebastian, Barber, True, Baeza-Yates

Pet. 2.

As evidence, Petitioner relies on a Declaration of Dr. Loren Terveen, a Professor of Computer Science and Engineering at the University of Minnesota who is proffered as one "qualified in the technology fields pertaining to this proceeding . . . at least as a person of ordinary skill in the

[1] The relevant sections of the Leahy-Smith America Invents Act ("AIA"), Pub. L. No. 112-29, 125 Stat. 284 (2011), took effect on March 16, 2013. Because the effective filing date for the '490 patent precedes the effective date of the relevant amendment, the pre-AIA version of Section 103 applies.

art." Ex. 1003 ¶¶ 4, 10. Dr. Terveen also provided a Reply Declaration. Ex. 1100. Petitioner also relies on a Declaration of Delainey Tanner regarding a search of the Internet Archive (web.archive.org) (Ex. 1038), a Declaration of Sylvia Hall-Ellis regarding the authenticity and public availability of several documents (Ex. 1060), and a Declaration of Ingrid Hsieh-Yee regarding the authenticity and public availability of the Londergan reference (Ex. 1062).

Patent Owner relies on a Declaration of Mr. Martin Haeberli, who is proffered as an expert with more than fifty years of experience in the field of computer software, systems, hardware, networking, and communications. Ex. 2001 ¶ 9. Mr. Haeberli also provided a Supplemental Declaration. Ex. 2020.

E. Level of Ordinary Skill in the Art

To determine whether an invention would have been obvious at the time it was made, we consider the level of ordinary skill in the pertinent art at the time of the invention. *Graham v. John Deere Co. of Kansas City*, 383 U.S. 1, 17 (1966). Various factors may be considered, including the "type of problems encountered in the art; prior art solutions to those problems; rapidity with which innovations are made; sophistication of the technology; and educational level of active workers in the field." *In re GPAC, Inc.*, 57 F.3d 1573, 1579 (Fed. Cir. 1995) (citing *Custom Accessories, Inc. v. Jeffrey-Allan Indus., Inc.*, 807 F.2d 955, 962–63 (Fed. Cir. 1986)). "[O]ne or more factors may predominate." *Id.*

Petitioner argues that one of ordinary skill in the art at the time the invention was made "would have had a bachelor's degree in computer science, or equivalent, with two or more years of experience developing and

implementing search engines and/or search interfaces. More education, e.g., a Master's or Ph.D., would compensate for less work experience and vice versa." Pet. 5; Ex. 1003 ¶ 30.

Patent Owner does not comment on Petitioner's formulation or offer any competing formulation in either its Preliminary Response or its Response to the Petition. *See generally* Prelim. Resp.; PO Resp. Patent Owner's expert Mr. Haeberli, however, testifies as follows:

> 29. In my opinion, the POSITA at the time of the alleged invention of the Asserted Patents would have had a (1) a Bachelor's degree in a pertinent discipline, such as computer science, electric engineering, computer engineering, or a related field and (2) have 2–4 years of experience in working with computing devices and have an understanding of indexed search systems utilized by such computing devices.
>
> 30. I am aware that Petitioner has provided its own definition . . . .
>
> 31. I believe Petitioner's definition is not meaningfully different from my definition of a POSITA. My opinions would not change regardless of which definition of POSITA is applied.

Ex. 2001 ¶¶ 29–31.

In our Decision on Institution, we adopted Mr. Haeberli's formulation, finding it to be consistent with the evidence then of record, including the asserted prior art and the '490 patent. Dec. 9. Following a review of the full record, we agree with Mr. Haeberli's statement that Petitioner's definition of one of ordinary skill in the art is "not meaningfully different" from Mr. Haeberli's for the purposes of this proceeding. We continue to view Mr. Haeberli's proposed definition as consistent with the '490 patent and the asserted prior art, and we apply it in our analysis below. However, our analysis of the issues below would have been the same had we instead adopted Petitioner's proposed definition.

III. CLAIM CONSTRUCTION

In an *inter partes* review proceeding based on a petition filed on or after November 13, 2018, such as this, we construe claims using the same claim construction standard that would be used in a civil action under 35 U.S.C. § 282(b), as articulated in *Phillips*, 415 F.3d at 1303, and its progeny. *See* 37 C.F.R. § 42.100(b). When applying that standard, we interpret the claim language as it would have been understood by one of ordinary skill in the art in light of the specification. *Wasica Fin. GmbH v. Cont'l Auto. Sys., Inc.*, 853 F.3d 1272, 1279–80 (Fed. Cir. 2017).

In our Decision on Institution, the only term that we found it necessary to construe was the term "string separator character" in claim 1. We preliminarily construed this term to mean "a single-character string separator." Dec. 11, 30–31. We invited the parties to address the appropriateness of this construction over the course of the trial. *Id.* at 11, 31. In its briefing, Patent Owner agrees with the Board's preliminary construction, arguing that it is "consistent with the intrinsic evidence." PO Resp. 8 (citing Ex. 1001, claims 8 and 20). Patent Owner further argues that a person of ordinary skill would have understood a multi-character Boolean operator such as "AND" to be "a search syntax instruction rather than a string separator." *Id.* at 9 (citing Ex. 1059, 162; Ex. 2001 ¶ 112). Petitioner's Reply Brief does not address the issue of claim construction. *See generally* Reply Br.; *see also* Pet. 11 ("Unless otherwise noted, Petitioner applies the plain and ordinary meaning of all claim terms below.").

Following our review of the full record, we determine that it is not necessary to construe this or any other claim term for the purposes of this

Decision.[2] *See Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co.*, 868 F.3d 1013, 1017 (Fed. Cir. 2017) (Only terms that are in controversy need to be construed, and then only to the extent necessary to resolve the controversy.). In our analysis below, we interpret the language of the claims according to its plain and ordinary meaning.

IV. OBVIOUSNESS

Petitioner alleges that claims 1–20 of the '490 patent are unpatentable for obviousness over the cited prior art. Pet. 1–2. For the reasons discussed below, we determine that neither of Petitioner's asserted grounds of unpatentability are persuasive. In Section IV.B below, we consider Petitioner's Ground 1, i.e., whether the combination of Wilcox, Londergan, Raskin, and Wu teaches each element of claims 1–20. In Section IV.C, we consider Petitioner's Ground 2, i.e., whether the combination of Schwartz, Sebastian, Barber, True, and Baeza-Yates teaches each element of the challenged claims.

A. Legal Framework

"In an IPR, the petitioner has the burden from the onset to show with particularity why the patent it challenges is unpatentable." *Harmonic Inc. v. Avid Tech., Inc.*, 815 F.3d 1356, 1363 (Fed. Cir. 2016) (citing 35 U.S.C. § 312(a)(3) (requiring *inter partes* review petitions to identify "with particularity . . . the evidence that supports the grounds for the challenge to

[2] Patent Owner argues that Petitioner's Reply Brief advances constructions for two additional terms, "plurality" and "user interface." Sur-Reply Br. 2–6. Patent Owner offers alternative constructions of these terms. *Id.* Because our Decision does not turn on this claim language, we determine that it is not necessary to construe either term.

each claim")); *Dynamic Drinkware, LLC v. Nat'l Graphics, Inc.*, 800 F.3d 1375, 1378 (Fed. Cir. 2015) (discussing the burden of proof in *inter partes* review).

A claim is unpatentable for obviousness if the differences between the claimed subject matter and the prior art are such that the subject matter, as a whole, would have been obvious to one of ordinary skill in the art to which said subject matter pertains. *KSR Int'l Co. v. Teleflex Inc.*, 550 U.S. 398, 406 (2007). "Once all relevant facts are found, the ultimate legal determination [of obviousness] involves the weighing of the fact findings to conclude whether the claimed combination would have been obvious to an ordinary artisan." *Arctic Cat Inc. v. Bombardier Recreational Prods. Inc.*, 876 F.3d 1350, 1361 (Fed. Cir. 2017). Relevant factors include: (1) the scope and content of the prior art; (2) any differences between the claimed subject matter and the prior art; (3) the level of ordinary skill in the art; and (4) objective evidence of nonobviousness, if present.[3] *See Graham*, 383 U.S. at 17–18.

To support this conclusion, however, it is not enough to show merely that the prior art includes separate references covering each separate limitation in a challenged claim. *Unigene Labs., Inc. v. Apotex, Inc.*, 655 F.3d 1352, 1360 (Fed. Cir. 2011). Rather, obviousness additionally requires that a person of ordinary skill at the time of the invention "would have selected and combined those prior art elements in the normal course of research and development to yield the claimed invention." *Id.*

---

[3] While the Petition briefly discusses the topic of objective evidence of nonobviousness (Pet. 91–92), neither party submitted evidence relevant to this topic over the remaining course of the trial. Consequently, we conclude that objective evidence of nonobviousness is not present here.

"Consideration of differences, like each of the findings set forth in *Graham*, is but an aid in reaching the ultimate determination of whether the claimed invention *as a whole* would have been obvious." *Stratoflex, Inc. v. Aeroquip Corp.*, 713 F.2d 1530, 1537 (Fed. Cir. 1983). Moreover, as a factfinder, we must remain aware "of the distortion caused by hindsight bias and must be cautious of arguments reliant upon *ex post* reasoning." *KSR*, 550 U.S. at 421.

B. Ground 1: The Lotus Notes References, Raskin, and Wu

1. The Asserted Prior Art

Petitioner's Ground 1 asserts that the combination of Wilcox (Ex. 1059) and Londergan (Ex. 1061) ("the Lotus Notes references") with Raskin (Ex. 1021) and Wu (Ex. 1019) renders each of the challenged claims unpatentable for obviousness. At the hearing, however, Petitioner confirmed that it is not relying on Londergan as teaching any limitation of independent claim 1. *See* Tr. 23:15–22. Londergan is used only to teach or suggest certain elements of the dependent claims. *See*, *e.g.*, *id.* at 54:18–55:6; Pet. 35 (citing to both Ex. 1059, 164 and Ex. 1061, 145–146 regarding dependent claims 10 and 11).

a) Wilcox and Londergan

Wilcox (Ex. 1059) and Londergan (Ex. 1061) are publications describing Versions 3.0 and R5, respectively, of the Lotus Notes software package. *See* Ex. 1059, (a), xvi; Ex. 1061, (i). It is undisputed that these publications qualify as prior art under 35 U.S.C. sections 102(a) and 102(b) because they were published in 1993 and 1999, respectively, and were publicly available more than one year before the earliest possible priority date of the challenged patent. *See* Pet. 16.

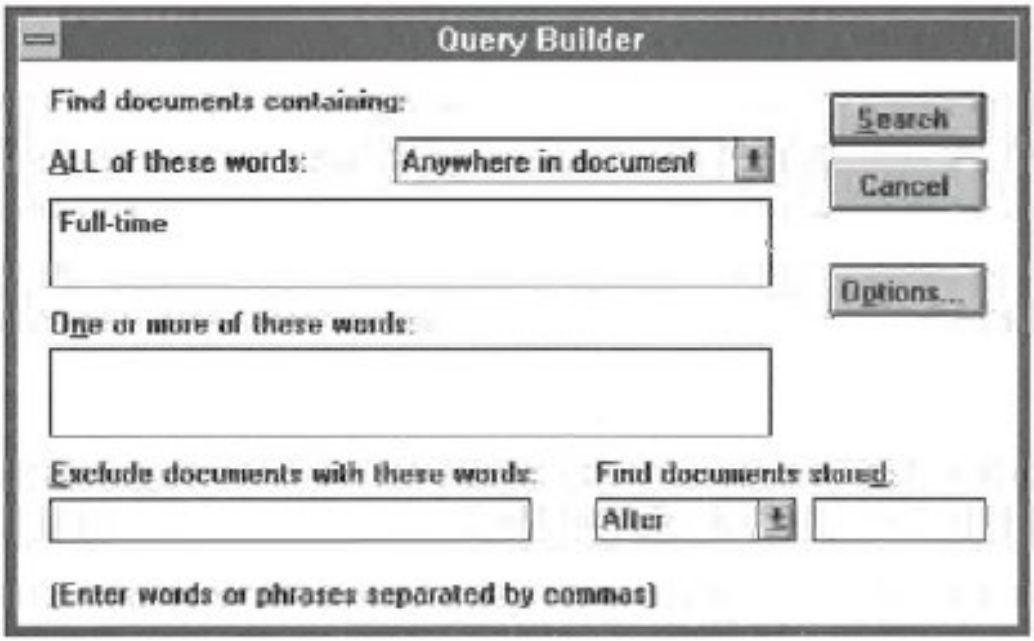
Lotus Notes is a "groupware" software product that is designed to allow groups of users to communicate and share information. Ex. 1059, 2; Ex. 1061, 6. Both versions of Lotus Notes store documents in a central database that resides on a network, rather than on the computers of individual users. Ex. 1059, 2–3; Ex. 1061, 12. Both versions of Lotus Notes contain electronic mail software. Ex. 1059, 5, Ex. 1061, 11. Electronic mail messages are also stored in a database on a central server. Ex. 1059, 5; Ex. 1061, 11–12. Lotus Notes indexes email messages and other types of documents, and allows users to conduct searches. *See* Ex. 1059, 160–65; Ex. 1061, 139–142.

(1) Wilcox

Wilcox explains that version 3.0 of Lotus Notes provides two tools for searching through documents in the indexed Lotus Notes database, a Search bar and a Query Builder, plus a Forms feature that allows users to search for text in a particular field in documents created using a specific form. Ex. 1059, 160–161, 167–168. Petitioner's obviousness arguments primarily rely on the Query Builder, a dialog box that makes it easier for users to construct complex searches.

Wilcox's Figure 5.16, reproduced below, depicts the Query Builder dialog box:

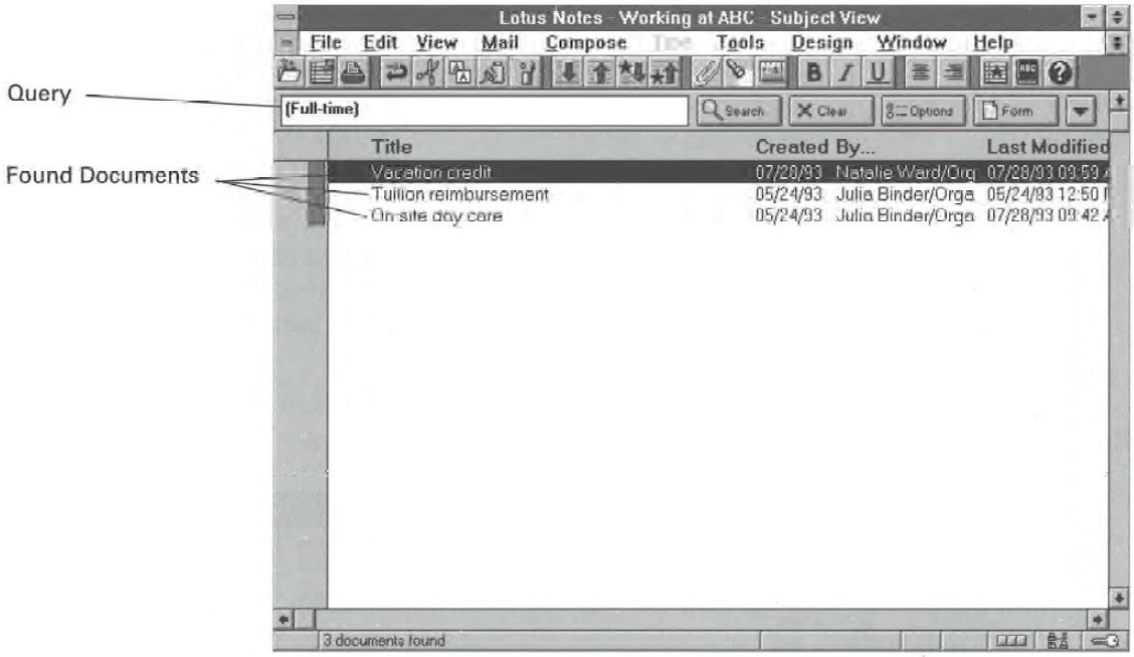
Using the Query Builder dialog box



Ex. 1059, 163. As shown above, the Query Builder includes a drop-down menu that allows a user to choose whether to conduct a search for the selected search terms "anywhere in document," or "near each other" (in which case search results are ranked by how close search terms appear to one another). *Id.* at 162–163. The Query Builder also contains an "All of these words" text box that can be used to search for documents that contain all of two or more user-specified words, and a "One or more of these words" box that can be used to search for documents that contain one or more of the specified words. *Id.* The Query Builder additionally contains a textbox that permits a user to exclude certain words from a search, and a drop-down menu and text box that a user can use to search for documents stored before or after a certain date. *Id.* The user is instructed that when entering multiple words in a textbox, the user should "[e]nter words or phrases separated by commas[.]" *Id.* Once a query has been entered, the user must click on the

"Search" button to view in a separate window a list of documents containing the search terms. *Id.*

Wilcox's Figure 5.17, reproduced below, depicts the results of a query constructed using Query Builder:



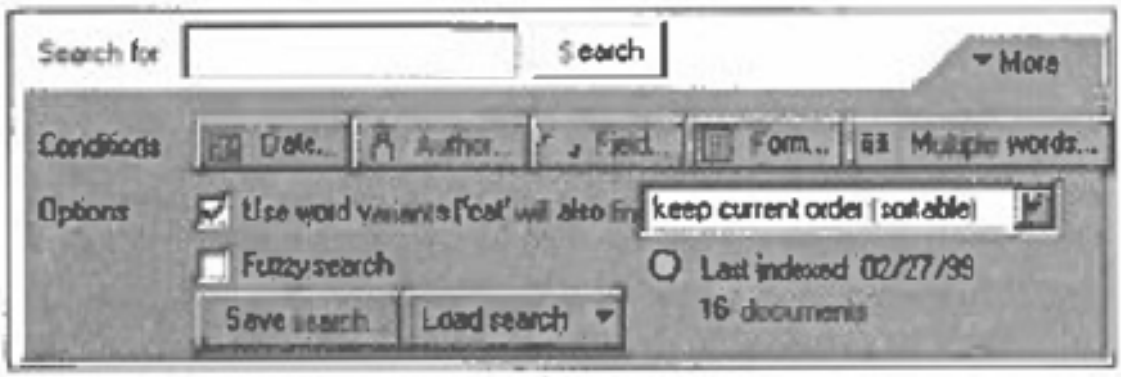
Ex. 1059, 164. Figure 5.17 depicts a search result window that displays the user's query in a search bar at the top of the window, with a list of found documents underneath. The list displays document titles, create dates, authors, and last modified dates, but does not show the documents' contents. *Id.* at Fig. 5.17. To view search results in a particular document, the user must manually open a document, at which point the text matching the search is highlighted by a red box. *Id.* at 164.

(2) Londergan

Londergan describes the search functionality of Version R5 of Lotus Notes. Ex. 1061, (i), 139–48. The 1999 version of Lotus Notes (R5) eliminated the Query Builder feature in favor of a different feature with different functionality, called "Search Builder." Ex. 2001 ¶ 82. Search

Builder provides multiple fields, each of which can only be used for a single search string. Like Version 3.0, Lotus Notes R5 includes a Search bar that can be used to search for words and phrases. *See id.* at 140. The Search bar in Version R5, however, includes a "More" drop-down button that displays the "Search Builder" dialog box. *See* Ex. 1061, 140–142.

A figure from Londergan depicting the Lotus Notes R5 Search Builder dialog box is reproduced below:



Ex. 1061, 141. As shown above, the Search Builder dialog box contains several buttons and fields that permit a user to specify various types of search conditions. *Id.* at 141–45. For example, the Search Builder includes an "Author" condition button that can be used to limit a search query to documents written by a particular author, and a "Date" condition button that can be used to limit a search to documents created before, after, or between certain dates. *Id.* at 141–42. There is also a "Field" condition button, which opens a separate dialog box that permits searches for text contained in particular fields of the Lotus Notes database. *Id.* at 143.

The Search Builder also includes a "Multiple words" condition button. *See id.* at 142. This button opens a separate dialog box, reproduced below, that permits searches for documents that contain "any" or "all" of two or more user-specified words. *Id.*

Ex. 1061, 142. Unlike in Query Builder, in Search Builder each search term is entered into a separate textbox. *Id.* As shown above, a user must click on an "OK" button to view the search results, which then appear in a separate window. *Id.*

b) Raskin

Raskin is a textbook describing user interface design principles for interactive systems. Ex. 1021, (i), (xi)–(xii). Raskin was published in 2000 and was publicly available more than one year before the earliest possible priority date of the challenged patent. Pet. 20–21. It therefore qualifies as prior art under 35 U.S.C. sections 102(a) and 102(b).

Portions of Raskin discuss interfaces for conducting string searches and "Find" searches. *See* Ex. 1021, 124–32. Raskin teaches that such search interfaces typically use either or both of two specific search strategies: delimited searches and incremental searches. *Id.* at 125.

In the first and more common type of search strategy, delimited searching, the user would typically enter text in a dialog box field. *Id.* The user would then initiate a search by entering a delimiting character (e.g., the Return key), or clicking a button (e.g., OK, Search, Find). Ex. 1021, 125. This would cause the system to search for instances of the relevant text string, and to designate each matching instance with a cursor. *Id.*

In the less common strategy described in Raskin, incremental searching, the user might summon a dialog box and enter a search term. Ex. 1021, 125. In an exemplary incremental search, the computer would immediately search for the first character entered by the user, and designate the first instance of this character in the searched document with a cursor. *Id.* After the second character is entered, the computer would begin searching for the first instance of those two characters together and then highlight the first such instance with a cursor. *Id.* The computer would then repeat this pattern each time an additional character was added to the search term. *Id.*

Raskin teaches that incremental searches have a number of advantages over delimited searching. Ex. 1021, 126. For example, incremental searches waste less of a user's time, make it easy for users to identify and correct search errors, and provide constant feedback to the user. *Id.* Notwithstanding these advantages, Raskin teaches that incremental searching is less common than delimited searching because interface-building tools (e.g., JavaScript and Visual BASIC) make it "difficult or impossible" for developers to implement incremental searches. *Id.*

c) Wu

Wu (Ex. 1019) is a U.S. Patent issued in 1999 and titled "Information Retrieval from Hierarchical Compound Documents." Ex. 1019, codes (45),

(54). Wu generally relates to an apparatus for searching for selected documents in a document repository that contains a large number of documents. Ex. 1019, 3:24–27. Wu discloses a search engine that uses a document repository and word index to perform an intersection search. *Id.* at 6:51–52. Wu describes such a search in response to a user-entered query for the search string "The game of Go." *Id.* at 6:52–54. Wu indicates that the search engine would disregard the common words "the" and "of," search through the document lists for the terms "game" and "go," and generate a match list for each term. *Id.* at 6:54–62. The search engine would then create a new match list containing all documents that appear on both of the previously-created match lists. *See id.* at 9:11–44. Thus, in this example, the final match list would identify documents containing both "game" and "go."

2. The Asserted Combination

Petitioner asserts that the Lotus Notes references teach each limitation of the challenged claims except for "(1) locating the files incrementally (i.e., character-by-character) and (2) doing so using an intersection approach (i.e., searching each string separately and then identifying the files that appear in both sets of search results)." Pet. 11. Petitioner further argues that "Raskin expressly discloses (1) and Wu expressly discloses (2)." Pet. 11.

According to Petitioner, one of ordinary skill in the art "would have been motivated to perform Lotus Notes' multi-string search incrementally for the reasons stated by Raskin: speed, effectiveness, and ease of use." Pet. 19 (citing Ex. 1021, 125–127). Petitioner argues that when implementing Lotus Notes' multi-string search, one of ordinary skill would have been motivated to perform an intersection search as taught by Wu, "at least because it was one of a limited number of well-known, predictable

approaches, was simple and easy to implement, and is especially well suited for incremental searches." *Id.* (citing Ex. 1019, 6:52–62, 9:11–12); Ex. 1003 ¶ 79. Petitioner concludes that a person of ordinary skill in the art would have reasonably expected to succeed in combining the teachings of Raskin and Wu with the Lotus Notes computing system because "it would have required only simple and straightforward software modifications." Pet. 19–20; *see* Ex. 1003 ¶¶ 80–81; Ex. 1100 ¶ 40.

3. Analysis

For the reasons explained below, we determine that Petitioner has not met its burden of showing unpatentability of any claim in light of the proposed combination of Wilcox, Londergan, Raskin, and Wu.

a) Independent Claim 1

The claimed invention, as recited in claim 1, has the following relevant characteristics: (1) it performs an incremental search; (2) of an index data structure; (3) associated with a plurality of files; (4) using multiple partial search strings entered into a single search field and separated by a string separator character. *See* Ex. 1001, 29:56–30:28. (While claim 1 includes other limitations, these are the aspects of the claim that are contested.) Petitioner asserts that the first characteristic is disclosed in Raskin, and that Wilcox—specifically its description of the Query Builder function in Lotus Notes Version 3.0—teaches or suggests the remaining three characteristics. Pet. 11, 13, 21, 25–26. Wu teaches that a search can be an intersection search, an uncontested issue. *See id.* at 25–26. Londergan is not relied upon for any aspect of claim 1. *See* Tr. 23:15–22.

(1) The Parties' Arguments

Patent Owner argues that "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[.]"[4] PO Resp. 18. Patent Owner's expert, Mr. Haeberli, testifies that one of ordinary skill in the art would not have been motivated to combine one of the Lotus Notes search tools, such as the Query Builder in Wilcox, with Raskin's incremental search, for three reasons. First, Raskin's entire philosophy of user interfaces was incompatible with Lotus Notes. Second, Raskin was directed to searching within a single document at a time. Third, at the time, using incremental searches presented "extreme technical difficulties." Ex. 2001 ¶ 101.

Petitioner's expert, Dr. Terveen, disagrees. In his opinion, one of ordinary skill in the art would have been motivated to perform the Lotus Notes multi-string search incrementally for the purposes of "speed, effectiveness, and ease of use[,]" using "only simple and straightforward software modifications." Ex. 1003 ¶¶ 73, 80. According to Dr. Terveen, such a person "would have reasonably expected success implementing Lotus Notes' user interface (UI) to support incremental searching using an appropriate programming language such as C or C++, or whichever programming language Lotus Notes was built on at the time (likely C or C++)." Ex. 1100 ¶ 42; *see id.* ¶ 44 ("Implementing such functionality would

[4] Patent Owner also argues that a POSITA would not be motivated to combine Wilcox with Londergan in the first place. Appeal Br. 10–17. We do not address this issue here because Petitioner represents that Londergan is only relevant to an analysis of the dependent claims. *See* Tr. 26:16–24.

have required little effort from a programmer."). Dr. Terveen notes that "[m]oreover, the '490 patent does not provide any details on how to build a user interface that supports an incremental search, further confirming that a POSITA would have had the knowledge to do so and would have reasonably expected success in building such an interface." *Id.* ¶¶ 45, 50–52.

Having reviewed the evidence of record, we find Mr. Haeberli's testimony to be more persuasive and better supported by the evidence. According to Dr. Terveen, "[r]obust programming languages such as C and C++, together with well-developed operating system and UI frameworks, . . . allowed an ordinary programmer to build a UI that supported incremental searching, as evidenced by the numerous prior art systems that implemented such functionality, including Entourage, Eudora, Emacs, Canon Cat, SwyftWare, and others. Ex. 1100 ¶ 43 (citing Ex. 1004, 8–9, 161; Ex. 1021, 125, 127, 129; Ex. 1039, 102–103; Ex. 1043, 385–389). The relevant question, though, is not whether it would have been possible to build a user interface that supported incremental search, but whether it would have been "simple and straightforward" to execute an incremental search of multiple partial search strings on large, indexed multi-file databases such as those described in the Lotus Notes references, as Dr. Terveen alleges. None of the prior art discussed in the "State of the Art" section of Dr. Terveen's initial declaration, taken individually or together, discloses incrementally searching a plurality of files using two partial search strings. *See* Ex. 1003 ¶¶ 32–33, 37, 40; Ex. 2001 ¶ 100. Thus, creating a user interface might be "simple," but the evidence suggests that the back-end execution of an incremental search is not. *See, e.g.*, Ex. 1021, 126 (Raskin, concluding that "almost all interface-building tools make it easy to implement delimited searches and difficult or impossible to implement incremental searches").

(2) Lack of Motivation to Combine or Reasonable Expectation of Success

Below, we discuss Mr. Haeberli's reasons for concluding that one of ordinary skill in the art would not have been motivated to modify Wilcox's Lotus Notes database by adding the incremental search capability discussed in Raskin. For the reasons discussed below, we adopt Mr. Haeberli's reasoning as our own except where indicated.

First, Mr. Haeberli testifies that "Raskin's entire philosophy to user interfaces was incompatible with Lotus Notes." Ex. 2001 ¶ 101; *see id.* ¶¶ 96–99. He testifies that "Lotus Notes is focused primarily on centralized collaboration around documents, files, and file structures." *Id.* ¶ 99. Wilcox describes "a system that is built around a database architecture that requires an index to enable full-text searching of documents stored in the database." Ex. 2020 ¶ 32; Ex. 1059, 160, 169 ("You can perform simple searches on databases that are not indexed, but to use the full-text search features on a database it must be indexed."). The Query Builder function described in Wilcox only becomes available if a database has been indexed, and every time the database is altered, it must be re-indexed before full-text searching will work properly. Ex. 1059, 171. Users interact with the indexed database in order to "share work from a user's PC with fellow collaborators electronically[,]" which is possible because searchable information is stored in a centralized location that multiple users can access. Ex. 2001 ¶ 97.

According to Mr. Haeberli, "Raskin goes in a completely different direction, espousing the elimination of filenames and 'hierarchical file structures' including directories and folders[,] which effectively renders Lotus Notes' functionality of hierarchical viewing of and responding to documents unusable." Ex. 2001 ¶ 99; *see* Ex. 1021, 119. 121. In

Mr. Haeberli's opinion, "[a] POSITA would understand that the lack of a server-wide file management hierarchy or file management systems taught by Raskin would severely hamper the discussions and responses envisioned by Lotus Notes and thus . . . a POSITA would not look to Raskin to improve Lotus Notes." Ex. 2001 ¶ 99.

Second, Mr. Haeberli testifies that unlike the indexed approach in Lotus Notes, "Raskin was directed to searching a single document." Ex. 2001 ¶ 101; *see id.* ¶¶ 93–95. An index provides the locations of all instances of an indexed term across all indexed files. *See, e.g.*, Ex. 2020 ¶ 15 ("A simple index is a data structure that associates the words in the indexed documents with the occurrences of the words in each document."). Raskin's approach to incremental search is different. Ex. 2020 ¶¶ 36–37. The portion of Raskin that Dr. Terveen relies upon discusses the search functionality implemented in the 1980s in the GNU EMACS text editor. Ex. 1003 ¶¶ 35, 68–70. Rather than scanning through an index of multiple documents, EMACS performs a "sequential search," which involves scanning through the text character-by-character within an open document to find matches for those characters. Ex. 1022, 55. In EMACS, when the user enters a character in the summoned dialog box, the EMACS searching tool "immediately begins to search for the first instance of that character" and places the cursor at the end of the first term containing the entered character that the search encounters. Ex. 1021, 125; Ex. 2001 ¶ 59. Mr. Haeberli characterizes this as "a brute force approach—Raskin's disclosure concerns non-indexed, sequential scanning within a single document (in EMACS terminology, the 'buffer') using a single search term." Ex. 2020 ¶ 36. He continues:

> The technical differences between these two search methodologies are substantial and represent fundamentally

> different approaches to the problem of finding information. The Lotus Notes system uses pre-computed indexes to enable rapid searching across large document collections, while the EMACS system described by Raskin uses real-time sequential scanning of text that is already loaded into the computer's active memory.

Ex. 2020 ¶ 37.

Based on Mr. Haeberli's testimony, Patent Owner argues that "[t]he search capability in EMACS operates within the confines of a single text document that is currently open in the editor (i.e., loaded into the active buffer)—it is not a system for searching across multiple documents or for locating documents within a file system." PO Resp. 18; *see* Ex. 1022, 125.

Here, we disagree somewhat with Patent Owner's summary of the evidence. As Petitioner argues, Raskin also teaches that the search domain can expand beyond a single file. Reply Br. 9 (citing Ex. 1021, 129, 132). Specifically, Raskin teaches:

> *In larger texts, a search can proceed circularly not only through the local document but also afterward in automatically expanding domains, up to and including the entire Internet.* After the local document has been completely searched, the search then proceeds through the following documents until the end of the folder is reached, when the search continues from the beginning of the first document in the folder until the current, already searched document is reached. After cycling through the folder, the next larger domain is treated similarly, and so forth.

Ex. 1021, 132. Thus, Raskin discloses that by conducting a "brute force search" one document at a time through all of the documents in a folder or domain, it is possible to engage in non-indexed, sequential scanning of a series of multiple documents. (Raskin also discloses that "[g]enerally, people will use more efficient strategies than simply relying on this expanding sweep of the hierarchy." *Id.*)

We agree with Patent Owner, however, that this is a fundamentally different approach from that taken in Lotus Notes. The difference between Raskin's sequential scanning method and the index-based system described in Wilcox is that sequential scanning by definition can only scan one document at a time as each document is loaded into a "buffer." *See, e.g.*, Ex. 2020 ¶ 35. Lotus Notes, however, only needs to load the index into memory for a user to be able to locate the single index entry identifying all instances of an indexed term across all indexed files. In fact, Raskin acknowledges that indexing may be necessary to maintain an acceptable response time. Ex. 1021, 132 n.10; *see also id.* at 129 (indexing speeds search times).

Mr. Haeberli's third and final reason for concluding that one of ordinary skill would not have been motivated to combine Raskin's incremental search with Lotus Notes is that at the time using incremental searches would have presented "extreme technical difficulties." Ex. 2001 ¶ 101. Here again we find Mr. Haeberli's explanation more persuasive than Dr. Terveen's opinion that combining the teachings from Raskin with Lotus Notes' teachings "would have required only simple and straightforward software modifications." Ex. 1003 ¶ 80. According to Mr. Haeberli, "there are serious technical challenges with implementing the complex search strategies of the challenged claims in the systems on which Dr. Terveen relies." *Id.* ¶ 96. As Patent Owner argues:

> searching across a set of files to find specific files requires the system to identify which files are in the set, access those files (which are often not loaded in active memory), search their contents, determine which files contain matches to search criteria, and then present results to the user; all tasks that do not exist in the same way (or at all) for a search confined to text in a single open file already loaded into memory.

Sur-Reply 9; *see* Ex. 2020 ¶¶ 15, 35–36. There would have been other challenges as well. For example, "[t]he type of sequential scanning typically employed to search a single open document is very time and resource intensive if employed to search a large set of documents." Ex. 2020 ¶ 15. Also, "[i]mplementing incremental search in the Lotus Notes indexed system would necessarily require a larger index, which would slow down, rather than increase the search speed." PO Resp. 23; *see* Ex. 2020 ¶ 33. Finally, Raskin itself concludes that "even if incremental searching is advantageous, 'almost all interface-building tools make it easy to implement delimited searches and *difficult or impossible* to implement incremental searches.'" Ex. 2001 ¶ 96 (emphasis added by Mr. Haeberli) (quoting Ex. 1021, 126).

In short, Mr. Haeberli presents three significant reasons why there would have been no motivation or expectation of success in combining the teachings of Wilcox with those of Raskin. Dr. Terveen's opinion to the contrary, on the other hand, is not supported by evidence. Dr. Terveen testifies that "[a] POSITA would have been motivated to perform Lotus Notes' multi-string search incrementally for the reasons stated by Raskin: speed, effectiveness, and ease of use." Ex. 1003 ¶ 78 (citing Ex. 1021 ¶¶ 125–127). The evidence, however, not only fails to support Dr. Terveen's testimony, but also demonstrates that one attempting to make the proposed combination would not have reasonably expected the proposed combination to result in greater speed or effectiveness.

Wilcox describes a system built around a database architecture that requires an index to enable full-text searching of documents stored in the database. Ex. 1059, 159; Ex. 2020 ¶ 32. Mr. Haeberli testifies:

> implementing incremental search in a multi-document search system would require either an increased index size, which would slow down searching or exceed the available computing power of a typical consumer computer at the time of the invention, or would require sequential search across all documents, which would be so slow as to no longer function as incremental search at all.

Ex. 2020 ¶ 87; *see id.* ¶¶ 47–55. It appears to be undisputed that sequential search would be impractical. *See* Ex. 1003 ¶ 148 (sequential scanning takes more time and resources than searching using an index). Thus, as Dr. Terveen testified at his deposition, modifying Lotus Notes to add an incremental search feature would require adding partial search strings to an existing full-text index. Ex. 2021, 104:17–105:11. This would necessarily increase the size of the index.

"From a technical standpoint, the size of the index has a direct and significant impact on search performance—specifically, larger indexes result in slower search operations." Ex. 2020 ¶ 33. Dr. Terveen himself acknowledged this principle during his deposition, testifying that a larger index "could increase the time it took to search it" and that "making the index smaller would be more efficient in general." Ex. 2021, 38:1–39:17. Thus, combining Raskin's incremental search with the indexed database in Wilcox would almost certainly degrade the performance of Wilcox's system. *See id.* No matter how advanced the computer, there is no evidence that a system's search times would ever *speed up* as a result of increasing index size to allow an incremental search. Consequently, a POSITA attempting to increase search speed and effectiveness would not have been motivated to modify Wilcox with Raskin's incremental search.

None of the examples of prior art incremental search systems that Dr. Terveen cites disclose incrementally searching a plurality of files using

two partial search strings, *see* Ex. 1003 ¶¶ 32–33, 37, 40; Ex. 2001 ¶ 100, and Petitioner's proposed combination would, if anything, undermine the goals that Dr. Terveen contends would have motivated one of ordinary skill in the art. On balance, we are persuaded by Patent Owner's arguments and evidence, and accordingly we adopt as our own Mr. Haeberli's conclusion that one of ordinary skill in the art would not have been motivated to insert Raskin's sequential scanning approach into an existing index-based system such as Lotus Notes.

We further conclude that one of ordinary skill in the art would not have had a reasonable expectation of success in making the proposed combination. In its Reply Brief, Petitioner suggests that the problems associated with searching across a large volume of documents are not relevant because the challenged claims encompass searching across as few as two files. Reply Br. 3–8 (recited "plurality" means "two or more"). Petitioner argues that a POSITA would have been motivated with a reasonable expectation of success to combine Lotus Notes with Raskin "at least because 2002-era computers could provide the response times needed to perform an incremental search across a plurality of files, especially while using known index data structures, such as Lotus Notes' full-text index, when searching across larger volumes of files." Reply Br. 5; Ex. 1100 ¶¶ 13–32.

As Patent Owner argues, this argument, presented for the first time in the Reply Brief, improperly frames Petitioner's alleged motivation to combine theory "in a way that is divorced from its Petition." Sur-Reply 6. The argument is inconsistent with the very nature of the Lotus Notes database that Petitioner alleges a skilled artisan would have combined with Raskin and Wu. The Lotus Notes references describe a large, enterprise-

level database, not a small database containing only two items. Moreover, "Petitioner's new framing of the claims as requiring 'searching' across only 'a plurality' of files (meaning two or more) ignores the Petition's stated motivation [of 'speed, effectiveness, and ease of use,'] as there can be no motivation to solve a speed and responsiveness problem that does not exist for a database of just two files." *Id.* If the claimed search were only across two files, there would be no need to develop and implement a complex incremental search across an index of partial strings, and therefore there would be no motivation to modify the Lotus Notes references with Raskin. If, on the other hand, the number of files involved were large, one of ordinary skill in the art would not have a reasonable expectation of success, given the effect of increased index size on search speed, as discussed above. *See* Sur-Reply 6–8. Thus, even if this argument were timely, it would not persuasively support Petitioner's motivation to combine arguments or contradict Mr. Haeberli's opinions.

Based on our review of the evidence, we conclude that Petitioner has not established that one of ordinary skill in the art would have been motivated, with a reasonable expectation of success, to attempt to combine Raskin (and Wu) with the Lotus Notes references, particularly Wilcox, in a manner that would render independent claim 1 of the '490 patent unpatentable for obviousness.

b) Dependent claims 2–20

Petitioner has not shown that claim 1 is unpatentable for obviousness in light of Wilcox, Raskin, and Wu. Because the independent claim has not been shown to be obvious, its dependent claims also have not been shown to be unpatentable for obviousness. *In re Fritch*, 972 F.2d 1260, 1266 (Fed.

Cir. 1992) ("[D]ependent claims are nonobvious if the independent claims from which they depend are nonobvious.").

4. Summary

We have concluded that one of ordinary skill in the art at the relevant time would not have been motivated to combine the Lotus Notes references with Raskin and would not have had a reasonable expectation of success in doing so. Accordingly, we need not further consider whether the hypothetical prior art combination would teach each limitation of the challenged claims. Weighing all of the evidence and arguments offered by both sides, we conclude that Petitioner has not met its burden to show by a preponderance of the evidence that claims 1–20 are unpatentable in light of the combination of Wilcox, Londergan, Raskin, and Wu.

C. Ground 2: The Entourage References, True, and Baeza-Yates

1. The Asserted Prior Art

Petitioner's Ground 2 asserts that the combination of Schwartz (Ex. 1004), Sebastian (Ex. 1005), and Barber (Ex. 1006) ("the Entourage references") with True (Ex. 1009) and Baeza-Yates (Ex. 1023) renders claims 1–20 unpatentable for obviousness.

a) Schwartz, Sebastian, and Barber

Schwartz (Ex. 1004), Sebastian (Ex. 1005), and Barber (Ex. 1006) are publications that describe the same version of a Microsoft email system, Entourage 2001 for Macintosh. Ex. 1004, (i); Ex. 1005, (a)–(c), 171–188; Ex. 1006, (i), 1–3, 5, 9, 283–392. In 2001, Entourage had six main components: email, newsgroup reader, address book, calendar, tasks, and notes. Ex. 1003 ¶ 127. The Entourage references explain how each

component works and how to use the software. *Id.* Barber describes the various search options available in Entourage 2001. Ex. 1006, 384–87. Sebastian provides additional details on Entourage's main email interface, while Schwartz discusses the use of text highlighting within search results. *See* Pet. 49.

Entourage provides three ways for users to search for specific text: (1) list filtering, (2) Find, and (3) Advanced Find. Ex. 1003 ¶ 127. Schwartz describes the first option, list filtering, as follows:

> Perhaps the easiest way to find a particular message, contact record, note, or any other Entourage item is to *filter* the item list. Above the list in many Entourage windows are one or two pop-up menus and a text box. By choosing options from the pop-up menus and/or typing text in the box (Figure 1.11), you can filter the list below so that only matching items are displayed.

Ex. 1004, 8.

Schwartz's Figures 1.11 and 1.12 are reproduced below:

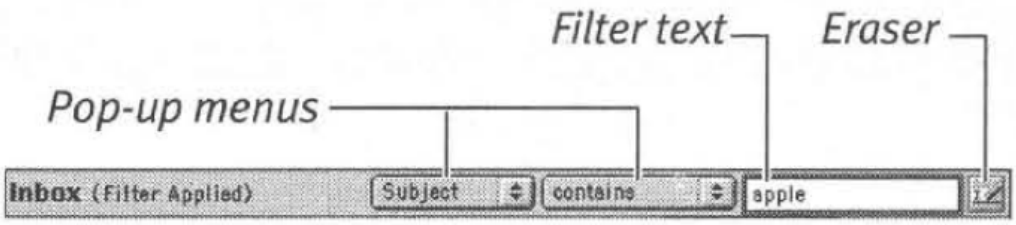


Figure 1.11 Entourage lists can be filtered to match simple criteria.

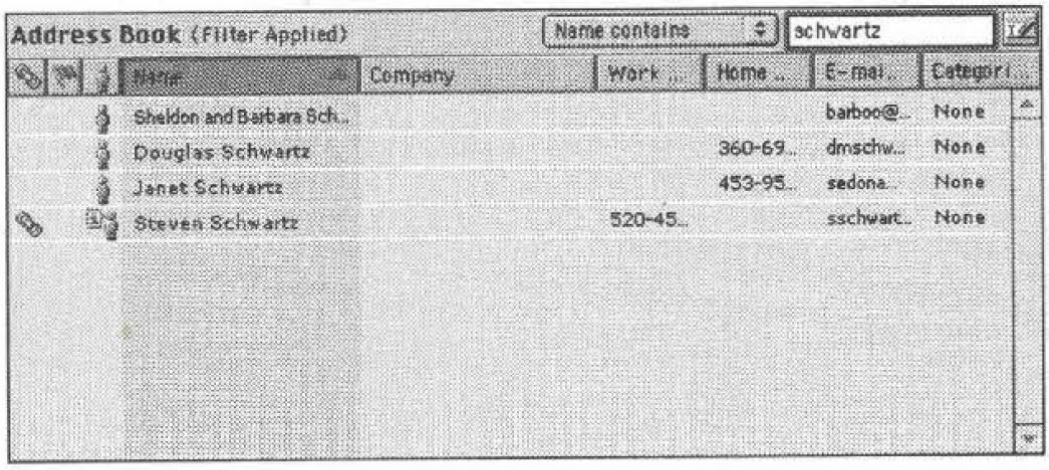


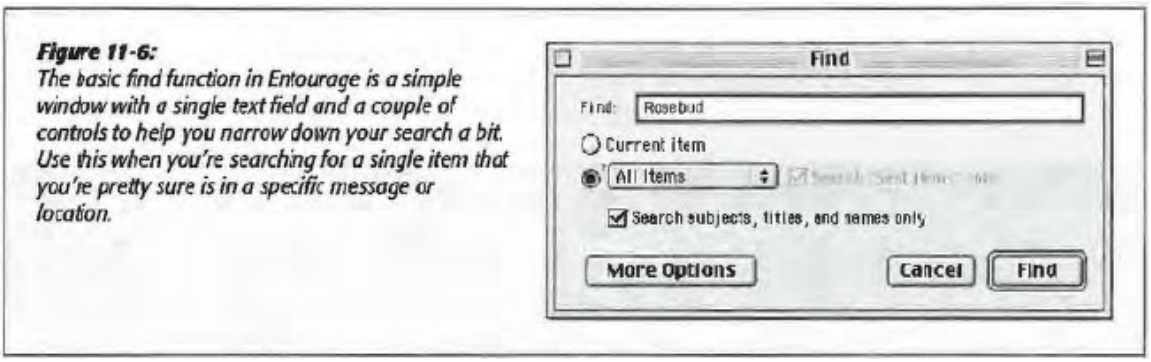
Figure 1.12 As you type the filter text, only matches are shown in the list.

Ex. 1004, 8–9. Figure 1.11 depicts the top portion of a typical Entourage window, displaying pop-up menus and a text box. Figure 1.12 depicts the complete Address Book window in Entourage. Below the pop-up menu and the text box is a list of Entourage "items," in this case Address Book entries. Field names such as "Name," "Company," and "E-mail" appear at the top of the list, and the appropriate data for each list entry appears beneath each field name.

Schwartz explains that to filter a list, a user would: (1) "[o]pen the window or folder that you want to filter, such as a message folder or the Tasks, Notes, or Address Book window"; (2) "[c]lick the pop-up menu(s) above the list, and choose search criteria"; and then (3) view the matching items displayed in the list. *Id.* at 8–9; *see also* Ex. 1006, 384. List filtering uses an incremental search; it allows a user to incrementally filter lists of Entourage items (emails, contacts, etc.) to only show items that match the search criteria. Ex. 1003 ¶ 128; Ex. 2001 ¶ 67 ("[T]he filtering method concurrently displays emails with headings that include the currently entered string, as the user types out the search."). List filtering is limited to searching

text that appears in the headings shown in the list window; it does not search within the body of the item (such as within an email) for text that matches the search criteria. *See* Ex. 1004, 8–9, Fig. 1.12; Ex. 2001 ¶ 67.

The second searching option in Entourage is the Find feature. In addition to searching for matching text in an item title, the Find command can search for matches within the body text of one or more Entourage items. Ex. 1004, 9. The Entourage Find dialog box is reproduced below:

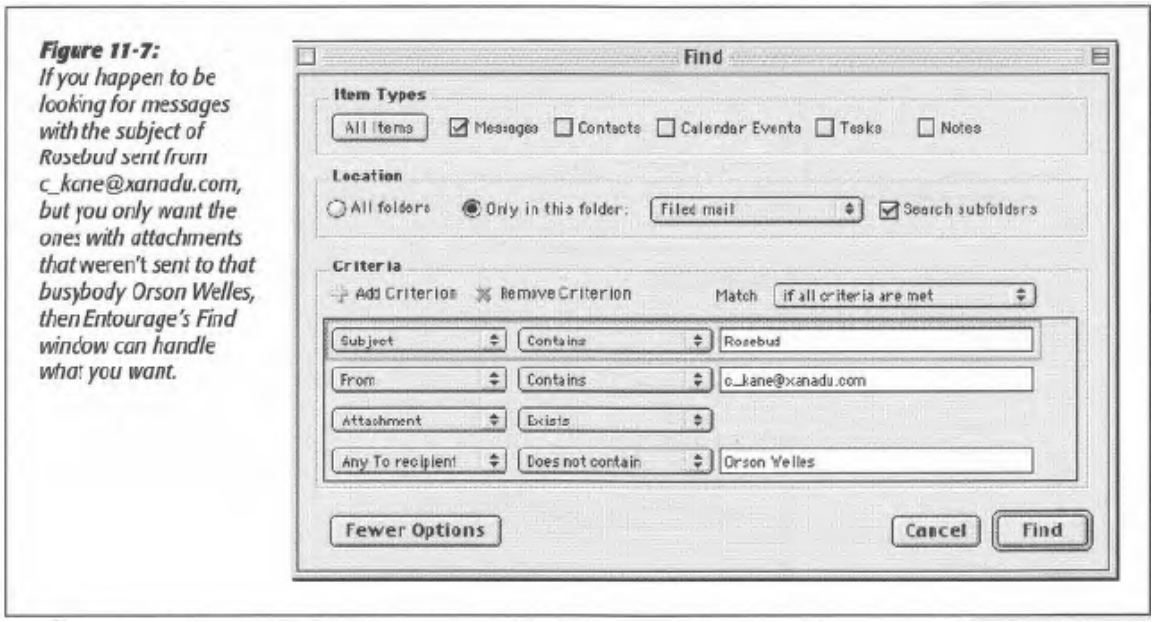


Ex. 1006, 385. Figure 11-6 of Barber, shown above, depicts the Find dialog box, which includes "a single text field and a couple of controls to help you narrow down your search a bit." *Id.* The controls allow a user to search within the current item, within all items, or within a subset of Entourage item types (email, contacts, calendar events, etc.). *Id.* at 384–385. The default is to search all text within each item, but the search can also be limited to subjects, title, and names only, which "*greatly* speeds up the search"). *Id.* at 385. Barber warns that if searching "All Items," one should "be prepared to rent a video during the time Entourage will take to search its entire multimegabyte database." *Id.*

Entourage's Find feature performs a delimited search. Ex. 1006, 385 ("After specifying how and where you want to search, click Find or press Return. Entourage begins its quest."). Once the search is complete, all of the results appear in a separate Search Results window, with a list view on the

top and a preview pane below. *Id.*; Ex. 1004, 9–11. If searching only the "Current Item," the first instance of the text string (if there is one) within the item is highlighted in the preview pane. Ex. 1004, 11.

The third searching option in Entourage is the Advanced Find feature, which is accessed by clicking on a "More Options" button in the Find dialog box. *See* Ex. 1006, 385 (Fig. 11-6). This opens a new dialog box as shown in Barber Figure 11-7, reproduced below:



Ex. 1006, 387. As shown above, the Advanced Find dialog box includes checkboxes for selecting any combination of item types, including "All Items"; radio buttons for choosing which locations to search; pop-up menus for defining multiple search criteria; a "Match" pop-up menu for selecting whether to find items with any, all, or none of the search criteria; and a Find button for initiating the search once all search criteria have been entered. *Id.*; Ex. 1004, 11 ("Advanced Find allows you 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."). While multiple search terms can be combined in Advanced Find, each search term is entered

in a separate text field. *See* Ex. 1006, 386 (add additional criteria "by clicking the Add Criterion button. Each time you do so, you get a new row in the dialog box."). Like Find, the Advanced Find feature performs a delimited search and lists the search results in a separate window. Ex. 1004, 12 ("As is the case with the Find command, the headers or titles of all matching items are displayed in the Search Results window[.]").

b) True

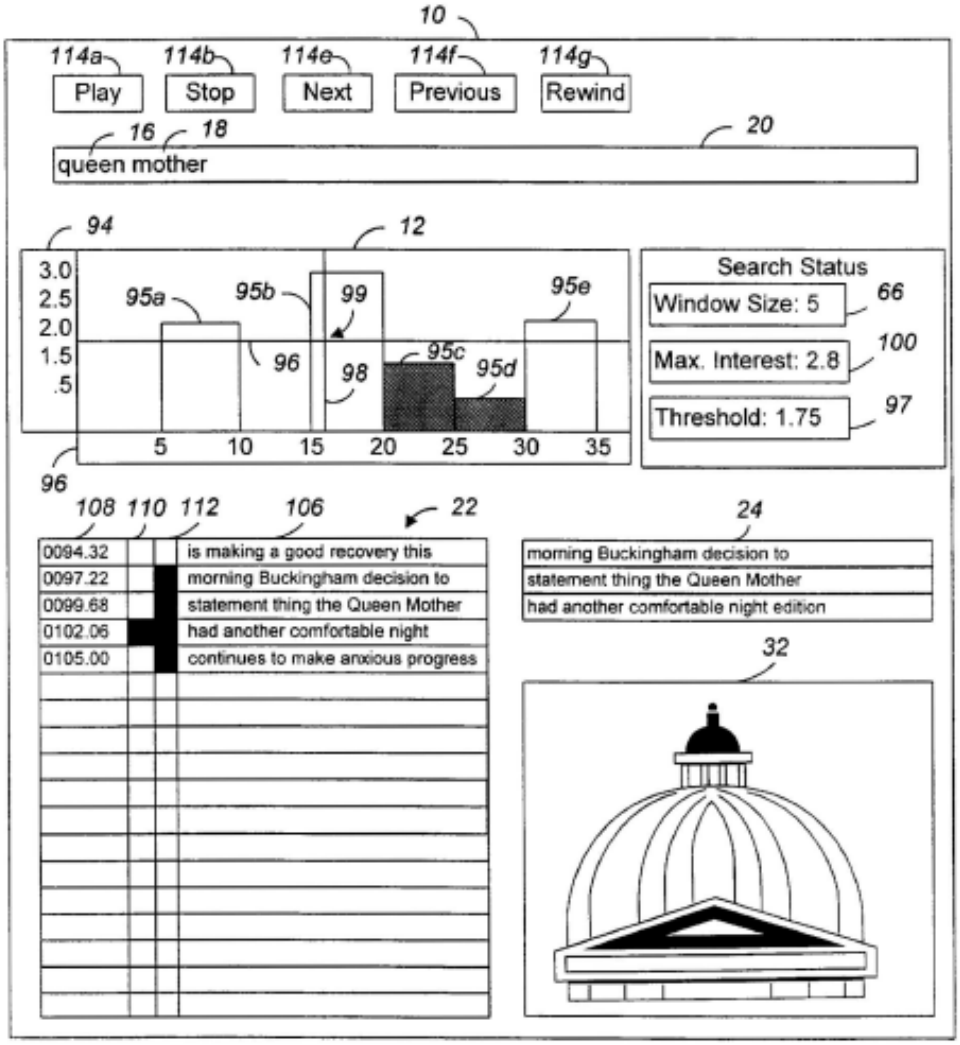
True is a U.S. patent entitled "Interactive Searching" that discloses searching text information underlying audio transcripts, using an interactive user interface that indicates which parts of the searchable information represent information in the audio transcript that is of interest to the user. Ex. 1009 (Abstr.). The search is done sequentially by scanning each character in the unindexed transcript text for the query words. Ex. 2020 ¶ 81 (citing Ex. 1009, 4:21–49). In particular, True discloses an audio transcript subdivided into equally-sized sections, referred to as "windows," with the search query performed on each window separately. Ex. 1009, 4:41–49.

> For each window in transcript 14, the process 78 calculates an "interest level" (step 94). An interest level represents the degree to which text in the window represents segments of interest in the digitized audio 26 and digitized video 34. An interest level is a function of (1) the extent to which the text in the transcript window accurately represents the words spoken in the digitized audio 26, and (2) the extent to which the text in the transcript window matches the query.
>
> A window's interest level may be calculated, for example, as the sum of the confidences of each word in the transcript window matching a word in the query. For example, if the query is "queen mother," and a transcript window contains an occurrence of the word "queen" with a confidence level of 0.27, another occurrence of the word "queen" with a confidence level of 0.75, and an occurrence of the word

> "mother" with a confidence level of 0.82, then the window's interest level is 1.84 (0.27+0.75+0.82). Thus, multiple occurrences of a search word are all given weight in the interest level computation.

Id. at 5:31–51. In other words, True shows which segments of the audio transcript likely relate to topics of interest by summing the confidence level that each search word is present in a given segment of the file. Because even low-confidence "hits" are counted towards a window's interest level, this approach may return results that do not actually contain the search terms. Ex. 2001 ¶ 134; *see* Ex. 2020 ¶¶ 79–81.

True's Figure 2 is reproduced below:



Ex. 1009, Fig. 2. As shown in Figure 2 above, True's result window provides a bar chart as a visual indication of which segments of the transcript might contain at least one of the words in the search query. Ex. 1009, 6:6–14, Fig. 2. It does not, however, provide a list of search results pointing to where all the instances of the words in the search query appear. *See* Ex. 2001 ¶¶ 132–134; Ex. 2020 ¶ 81.

Petitioner's expert, Dr. Terveen, describes True as disclosing "incrementally searching for, and locating, portions of an audio transcript containing multiple search strings entered in a single search field." Ex. 1003 ¶ 135. For example, referring to Figure 2, the user may enter the query "queen mother" in query window 20. This query contains two search terms in a single search field, separated by a space. *See* Ex. 1009, Fig. 2. As the user types, the search engine performs the query and a search engine UI process graphically displays the query results in the result window. *Id.* at 4:66–5:2. The user may modify the query, which "can include adding a single character to or deleting a single character from the query. As a result, the user can view the results of a query as the query is being modified." *Id.* at 5:2–6; *see also id.* at 2:26–31 ("[T]he graphical display of query results is constantly updated as the user modifies the query."), 2:39–48 ("Because query results are displayed as the query is being input, the user need not wait until the full query is entered before viewing the results.").

c) Baeza-Yates

Baeza-Yates is an undergraduate-level textbook published in 1999. Ex. 1023, (v). Among the information retrieval topics covered in Baeza-Yates is the concept of an intersection search, i.e., using indexes to locate files containing two search strings by locating a first set of files containing

"at least one word that begins with the first string and locating a second set of files containing at least one word that begins with the second string and then combining the results[.]" Pet. 52 (citing Ex. 1023, 101–103, 195–196, 207–208). Like the teachings of Wu discussed above, the teachings of Baeza-Yates regarding intersection searches are uncontested.

2. The Asserted Combination

Petitioner asserts that the combination of the Entourage references with True and Baeza-Yates "combines the Entourage references' teachings to incrementally locate files, including emails, containing at least one search string entered in a search field with True's teachings to do so for multiple search strings entered in the search field and Baeza-Yates teachings to do so [using an intersection search]." Pet. 53; Ex. 1003 ¶ 137.

Petitioner argues that one of ordinary skill in the art would have been motivated to combine the teachings of the three Entourage references, and would have reasonably expected to succeed in combining them, "at least because they describe the same version of Entourage (Entourage 2001) for the same computing system (Macintosh)." Pet. 48–49; Ex. 1003 ¶ 133. One of ordinary skill "would have been motivated to look to the teachings of all three references to obtain a more complete and comprehensive understanding of the Entourage system, its features, and its capabilities[.]" Pet. 48.

According to Petitioner, the motivation to combine True's multiple search strings with the teachings of the Entourage references would have been the knowledge that "such a multi-string search would return more relevant results than a single-string query." Pet. 53; Ex. 1003 ¶ 138. The motivation to incorporate the intersection search taught in Baeza-Yates

would have been that "this was one of only a limited number (a few at most) of known ways to locate files containing multiple search strings, was simple and easy to implement, and was especially well suited for an incremental search." Pet. 53–54; Ex. 1003 ¶ 139. Petitioner asserts that there would have been a reasonable expectation of success in combining the five references "because it would have required only simple and straightforward software modifications." Pet. 54; Ex. 1003 ¶ 141. Petitioner also asserts that one of ordinary skill in the art "would have been motivated with a reasonable expectation of success to use an index data structure to speed up the search process, as disclosed by Baeza-Yates . . . and well known in the art[.]" Pet. 54; Ex. 1003 ¶ 141.

3. Analysis

For the reasons explained below, we determine that Petitioner has not met its burden of showing unpatentability of any claim of the '490 patent in light of the proposed combination of Schwartz, Sebastian, Barber, True, and Baeza-Yates.

a) Independent Claim 1

Following our review of the record, we determine that Petitioner has not established that a person of ordinary skill in the art at the relevant time would have been motivated to combine the different search tools described in the Entourage references with one another or with the True reference. We further determine that such a person would not have had a reasonable expectation of success in attempting the proposed combination. Accordingly, Petitioner has not shown by a preponderance of the evidence that independent claim 1 is unpatentable for obviousness.

(1) Lack of Motivation to Combine the Various Search Functions Within Entourage

The Entourage references each describe three distinct search functions: (1) list filtering; (2) Find and (3) Advanced Find. Patent Owner argues that Petitioner relies on and implicitly combines all three, but "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." PO Resp. 39. Patent Owner has identified a fundamental problem with Petitioner's theory of unpatentability: Petitioner does not address why a person of ordinary skill in the art would have wanted to combine different elements from Entourage together, focusing only on the reason why such a person would have consulted Schwartz, Sebastian, and Barber in the first place: to learn about the Entourage system.

Petitioner's only stated motivation to combine the Entourage references and their teachings is that one of ordinary skill in the art "would have been motivated to look to the teachings of all three references to obtain a more complete and comprehensive understanding of the Entourage system, its features, and its capabilities[.]" Pet. 48. This may explain why one of ordinary skill would have consulted all three Entourage references, but it says nothing about how such a person would have used the knowledge gained as a result. In particular, it does not suggest that such a person would have been motivated to attempt to combine Entourage's list filtering feature ("Quick Find") with its Find and Advanced Find features. As Patent Owner argues, Petitioner's explanation "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." PO Resp. 50.

Petitioner's obviousness combination requires a POSITA to "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." PO Resp. 39. For example, Petitioner's Reply Brief argues that Schwartz, Sebastian, and Barber "teach performing an incremental search (e.g., in the Quick Find tool), searching for multiple search strings entered in a single search field (e.g., in the Advanced Find tool), and highlighting search strings in the results (e.g., in the Find tool)." Reply Br. 18. The evidence, however, demonstrates that the search tools are not interchangeable. Entourage's list filter function uses incremental search, but incremental searching is not available in the Find or Advanced Find functions. *Compare* Ex. 1003 ¶ 128; Ex. 2001 ¶ 67, *with* Ex. 1006, 385, 387. Conversely, Entourage's Find and Advanced Find functions can search within a selected document, but the list filter function cannot. *Compare* Ex. 1004, 9–11, *with* Ex. 1004, 8–9, Fig. 1.12; Ex. 2001 ¶ 67.

Petitioner's expert, Dr. Terveen, disagrees that combining these features in a single tool would have been difficult, testifying that "it would be relatively trivial to mix and match the functionalities of Entourage's different search tools." Ex. 1100 ¶¶ 59–61. While this opinion may be relevant to whether there would have been a reasonable expectation of success in attempting the proposed combination, Dr. Terveen does not identify any *motivation* for mixing and matching different search tools designed for different purposes.

In its Reply Brief, Petitioner argues that the rationale to combine these disclosures is "apparent" because the references present them "side by side," making the combination a "predictable variation" that does not "require a leap of inventiveness." Reply Br. 20 (citing *Paice LLC v. Ford Motor Co.*,

681 F. App'x 904, 917 (Fed. Cir. 2017); *KSR,* 550 U.S. at 417). The fact that list filtering, Find, and Advanced Find are presented side-by-side does not mean that they are compatible; if anything, it highlights the differences among them. The designers of the Entourage system knew how to implement an incremental search, and they knew how to implement a text search within the body of a document. They made a deliberate choice to separate these features in different search tools rather than building a single, all-purpose search engine.

Petitioner does not explain why combining elements of the three search features in Entourage would be a "predictable variation" on the actual Entourage software, arguing only that Patent Owner "fails to explain how these functionalities are incompatible." Reply Br. 20. In an *inter partes* review, however, that is not the Patent Owner's burden. Instead, "the petitioner has the burden from the onset to show with particularity why the patent it challenges is unpatentable." *Harmonic*, 815 F.3d at 1363. Because Petitioner has not identified any persuasive reason why a person of ordinary skill in the art would have been motivated to combine the incremental search of Entourage's list filter with the search capabilities of its Find and Advanced Find tools, Petitioner has not carried its burden to demonstrate that a motivation to combine the Entourage search tools together would have existed absent the benefit of hindsight.

(2) Lack of Motivation to Combine Entourage's Search Tools with True

Petitioner argues that "[t]he Entourage+True+Baeza-Yates combination combines the Entourage references' teachings to incrementally locate files, including emails, containing at least one search string entered in a search field with True's teachings to do so for multiple search strings

entered in the search field[.]" Pet. 53. Based on our review of the record, we find that even if one of ordinary skill in the art were motivated to combine elements of the various Entourage search tools, Petitioner does not sufficiently address why such a person would have been motivated to combine the result with True's teaching of permitting multiple search strings within a single search field.[5]

---

[5] We note Patent Owner's argument that True is not analogous art. PO Resp. 50, 53–55; Sur-Reply Br. 24–25. "In order to rely on a reference as a basis for rejection of the applicant's invention, the reference must either be in the field of the applicant's endeavor or, if not, then be reasonably pertinent to the particular problem with which the inventor was concerned." *In re Oetiker*, 977 F.2d 1443, 1447 (Fed. Cir. 1992). "A reference is reasonably pertinent if, even though it may be in a different field from that of the inventor's endeavor, it is one which, because of the matter with which it deals, logically would have commended itself to an inventor's attention in considering his problem." *In re Clay*, 966 F.2d 656, 659 (Fed. Cir. 1992).

Here, the '490 patent states that "[t]he present invention relates generally to data processing, and in particular to systems and methods for locating data." Ex. 1001, 1:38–39. More specifically, the present invention advantageously provides for incremental or reactive searching of a variety of search targets[.]" *Id.* at 2:20–32. Petitioner argues that "True's title similarly discloses 'interactive searching,' its drawings depict UIs similar to the patent's, and it describes a similar incremental search process where the user is provided with 'immediate feedback in response to actions such as modification of the query.'" Reply Br. 10; *see* Ex. 1009 (code 54), Figs. 1A, 1B, 2, 1:5 ("This invention relates to interactive searching."), 2:26–48. Petitioner also notes that during prosecution of the '490 patent, Patent Owner submitted True in an IDS, suggesting that even Patent Owner considered True to be pertinent to the subject matter of the '490 patent. Reply Br. 11.

We are persuaded by Petitioner's evidence that the '490 patent and True are in the same field of endeavor. Accordingly, we consider True to be analogous art for the purposes of our obviousness analysis.

Petitioner's theory is that there would have been a motivation to combine the Entourage references with True because "a POSITA knew that such a multi-string search would return more relevant results than a single-string query" Pet. 53. The evidence, however, suggests the opposite. Entourage's Advanced Find feature already permits multi-string searches, albeit one in which each string is entered in a separate search field. There is no persuasive evidence that incorporating True's multi-string approach, rather than the one already built into Advanced Find, would either: (1) result in more search results overall; or (2) generate results that are more relevant than those obtained through a single-string query. (It is unclear which meaning of the phrase "more relevant results" Petitioner intends to argue.)

Regarding the sheer number of results found, a multi-string approach would only lead to more results if treated as a Boolean OR query. If handled as a Boolean AND query, there would be the same number or fewer results than a single-string query, not more. Petitioner does not address which type of Boolean search the proposed combination would adopt. In either case, there is no evidence that searching for multiple strings within a single search field, as in True, would result in more results than searching for multiple strings entered in multiple search fields and then combined together, as in Entourage's Advanced Find feature.

As for whether True's multi-string query would result in more *relevant* results, the evidence shows that incorporating True's teachings into the proposed combination could actually generate results that are *less* relevant. As Patent Owner argues, True "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." PO Resp. 57–58. In fact, as described above, because True's method includes even low-confidence "hits" when

determining a window's interest level, True's approach may return results that only contain near misses and do not actually include *any* search terms. Ex. 2001 ¶ 134; *see* Ex. 2020 ¶¶ 79–81. Petitioner does not explain why one of ordinary skill in the art would have looked to True, which is designed to return results even where the search strings do not appear in the results, to design a search function that would "return more relevant results than a single-string query." *See* Pet. 53. Nor does Petitioner offer any other explanation why such a person would have been motivated to combine the teachings of True with any of the search tools described in the Entourage references. Accordingly, we conclude that Petitioner has not made a persuasive case that there would have been a motivation to attempt the proposed combination.

(3) No Reasonable Expectation of Success

We further conclude that even if a person of ordinary skill in the art had attempted the proposed combination of the Entourage references, True, and Baeza-Yates, Petitioner has not shown that that person would have had a reasonable expectation of success. As Patent Owner argues, "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." PO Resp. 63. Petitioner fails to describe how one of ordinary skill would modify any specific search feature in Entourage to incorporate True's disclosure of receiving two search strings in a single field. *See id.* at 61. It is unclear whether Petitioner proposes to integrate a multi-string search query within one of the fields shown in the Find or Advanced Find user interface and then add incremental search capabilities, or whether the proposal is to incorporate

True's multi-string searching into Entourage's list filter, and then add the ability to search within documents that in Entourage is only available in the Find and Advanced Find tools.

A Petitioner must identify "with particularity . . . the evidence that supports the grounds for the challenge to each claim." 35 U.S.C. § 312(a)(3); *see Harmonic*, 815 F.3d at 1363. Faced with similar difficulties in combining incompatible search features, the designers of Entourage opted to keep incremental searching and the ability to perform a full-text search separate rather than attempting to combine them in a single search tool. Petitioner has not shown with particularity that one of ordinary skill in the relevant art would have had a reasonable expectation of success in attempting to do that which the Entourage designers could not accomplish. Consequently, after weighing all evidence and arguments offered by both sides, we conclude that Petitioner has not met its burden of showing by a preponderance of the evidence that claim 1 is unpatentable for obviousness.

b) Dependent claims 2–20

Petitioner has not shown that claim 1 is unpatentable for obviousness in light of the Entourage references, True, and Baeza-Yates. Because independent claim 1 has not been shown to be obvious, its dependent claims also have not been shown to be unpatentable for obviousness. *In re Fritch*, 972 F.2d at 1266.

4. Summary

Because we have concluded that one of ordinary skill in the art at the relevant time would not have been motivated to combine the Entourage references with True (and Baeza-Yates) and would not have had a reasonable expectation of success in doing so, we need not further consider

whether the combination would teach each limitation of the challenged claims. Petitioner has not shown that claims 1–20 are unpatentable in light of the combination of Schwartz, Sebastian, Barber, True, and Baeza-Yates.

V. CONCLUSION

Having reviewed the arguments and supporting evidence, and after a full and fair hearing, we hold that Petitioner has not shown by a preponderance of the evidence that claims 1–20 of the ’490 patent are unpatentable under 35 U.S.C. § 103(a) over either of the asserted prior art combinations of record.

The following is a summary of the outcome of this proceeding:

Claims	**35 U.S.C. §**	**Reference(s)/Basis**	**Claims Shown Unpatentable**	**Claims Not Shown Unpatentable**
1–20	103(a)	Wilcox, Londergan, Raskin, Wu		1–20
1–20	103(a)	Schwartz, Sebastian, Barber, True, Baeza-Yates		1–20
Overall Outcome				1–20

VI. ORDER

In consideration of the foregoing, it is hereby

ORDERED that the record show that Petitioner has not established that any of claims 1–20 of the ’490 patent are unpatentable; and

FURTHER ORDERED that because this is a Final Written Decision, parties to the proceeding seeking judicial review of the decision must comply with the notice and service requirements of 37 C.F.R. § 90.2.

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICROSOFT CORPORATION,
Petitioner,

v.

X1 DISCOVERY, INC.,
Patent Owner.

IPR2025-00254
Patent 10,552,490 B2

Before WILLIAM V. SAINDON, SCOTT C. MOORE, and
LISA A. MURRAY, *Administrative Patent Judges.*

MURRAY, *Administrative Patent Judge*, for the Board.
SAINDON, *Administrative Patent Judge*, concurring.

CONCURRENCE

SAINDON, *Administrative Patent Judge*:

I respectfully concur. On the issue of reason to combine in the Lotus Notes ground, I agree with the majority that Petitioner has not established a reason to combine with a reasonable expectation of success. The majority sets forth several reasons why Petitioner has not met its burden, and I agree with those reasons but one. Respectfully, I do not agree that the evidence demonstrates "that one attempting to make the proposed combination would not have reasonably expected the proposed combination to result in greater speed or effectiveness." Maj. 28.

In particular, the majority focuses on testimony that index size increases search time. *Id.* at 28–31. All that this establishes is that some aspect of the computer's behind-the-scenes search process could take longer. As I understand it, Petitioner's position is that incremental search will increase search speed *for the user*, not that it will decrease the time the computer spends searching. Pet. 26–27 (bolstering its position by pointing out the practical benefits *for the user* in search speed, as opposed to machine speed, stating that incremental searches provide "continu[ous] update[s]" and "constant feedback," and that they "allow[] 'the user to improve [their] search strategies,'" and "to obtain the desired results more quickly . . . [because] the user does not have to type the full query"); Ex. 2021, 106:10–107:14 (Petitioner's declarant Dr. Terveen was asked, "what are you referring to when you say 'speed'" and he answered, "the search being faster for the user"); Ex. 2020 ¶ 44 (Patent Owner's declarant acknowledging Dr. Terveen's testimony as framing the speed issue as user-focused). Thus, in my view, the majority faults Petitioner for a position it did not take.

The inquiry of whether incremental search will be faster is not limited to the computer's search algorithm. The computing time increase, if any,

from a larger index is not necessarily sufficient to slow down the user's perception of search time. The search speed we are interested in, in my opinion, is the entire activity of the user looking for and finding something, not just the part where the computer is running an algorithm that is looking for hits. The added speed benefits of incremental searching itself are not considered by the majority, but incremental searching, in and of itself, provides a faster user experience. *See, e.g.*, Ex. 1021, 126–127 ("Building a pattern incrementally allows the user to adjust the pattern interactively during the search, which leads the user to improve his search strategies through the feedback received."). For example, incremental searching: allows a user to find results faster due to the fewer keystrokes, gives the computer the lead time by searching while a user is typing, and also allows the user to refine the search in real time based on the results. *See, e.g.*, Ex. 1021, 126 ("Incremental searching wastes less of a user's time: The search begins when the first character of the pattern is typed; the system does not wait until the pattern is complete." "[A]s the user enters the pattern, it provides constant feedback as to the results of the search."); Ex. 1100 ¶¶ 93–94 (Petitioner's declarant, Dr. Terveen, discussing how incremental searching provides practical speed gains to the user). All of these factors should be considered before concluding that search speed would be problematic. It is of course possible that implementing incremental searches in some situations so completely hamstrings the computer that no number of other benefits from incremental searching leads to faster search speeds. However, the majority focuses on a mere increase in a computer's search time as dispositive of the issue of search speed, and I respectfully disagree with that framing of the issue.

In sum, my view is that Petitioner's position is based on search speed, meaning user's perceived search speed, not just the computation time of the search. Even if an incremental search requires a relatively larger index than a non-incremental search, that fact is not dispositive as to search speed. The record establishes that search speed has a number of meaningful components beyond computer algorithm run time. Although I find fault with Petitioner's ground, on this point I do not find fault. I fully join the remainder of the opinion.

For PETITIONER:

Todd Siegel
Amy Haspel
Andrew Mason
Samuel Thacker
KLARQUIST SPARKMAN, LLP
todd.siegel@klarquist.com
amy.haspel@klarquist.com
andrew.mason@klarquist.com
samuel.thacker@klarquist.com

For PATENT OWNER:

Lori Gordon
Christie Larochelle
Sanjeet Dutta
GOODWIN PROCTER LLP
gordon-ptab@goodwinlaw.com
clarochelle@goodwinlaw.com
sdutta@goodwinlaw.com