UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICROSOFT CORPORATION,
Petitioner,

v.

X1 DISCOVERY, INC.,
Patent Owner.

IPR2025-00255
Patent 9,633,139

PETITIONER'S NOTICE OF APPEAL

NOTICE IS HEREBY GIVEN that, pursuant to 35 U.S.C. §§ 141(c) and 142 and 37 C.F.R. § 90.2, Petitioner Microsoft Corporation ("Microsoft") appeals to the United States Court of Appeals for the Federal Circuit from the Patent Trial and Appeal Board Final Written Decision entered June 18, 2026 (Paper 47, "Decision") in IPR2025-00255, and all underlying or interlocutory orders, decisions, rulings and opinions that are adverse to Petitioner. A copy of the Decision is attached hereto as Exhibit A.

In accordance with 37 C.F.R. § 90.2(a)(3)(ii), Petitioner states that the issues on appeal include, but are not limited to: (i) the Board's determination that Petitioner did not show by a preponderance of the evidence that claims 1-27 of U.S. Patent No. 9,633,139 are unpatentable as obvious; (ii) any finding or determination supporting or related to these issues, including, for example, the findings that Petitioner did not show that Raskin teaches the claimed incremental search or that one of ordinary skill in the art would have realized that this type of search could improve the Lotus Notes search functionalities, and that the Lotus Notes style of searching does not appear compatible with an incremental search; and (iii) all other issues decided adversely to Petitioner in any orders, decisions, rulings, and opinions entered in IPR2025-00255.

Pursuant to 37 C.F.R. § 90.3(a)(1), this Notice of Appeal is timely, having been duly filed within 63 days after the date of the Final Written Decision.

Pursuant to 35 U.S.C. § 142 and 37 C.F.R. § 90.2(a), this Notice is being filed with the Director of the United States Patent and Trademark Office, the Patent Trial and Appeal Board, and the Clerk's Office for the United States Court of Appeals for the Federal Circuit.

Respectfully submitted,

Dated: August 19, 2026

By: /Todd M. Siegel/

Todd M. Siegel, Reg. No. 73,232
todd.siegel@klarquist.com
KLARQUIST SPARKMAN, LLP
One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Tel: 503-595-5300
Fax: 503-595-5301

Counsel for Petitioner

CERTIFICATE OF SERVICE
IN COMPLIANCE WITH 37 C.F.R. § 42.6(e)(4)

The undersigned certifies that on August 19, 2026, a true and correct copy of **PETITIONER'S NOTICE OF APPEAL** was:

(1) Electronically filed through P-TACTS;

(2) Electronically filed with the Director of the United States Patent and Trademark Office by email to efileSO@uspto.gov, noting the document type in the email subject line;

(3) Filed in the U.S. Court of Appeals for the Federal Circuit using the Court's CM/ECF filing system and pay.gov to pay the filing fee; and

(4) Sent as a courtesy copy via electronic mail to the following attorneys of record for the Patent Owner listed below:

Lori Gordon GOODWIN PROCTER LLP 1900 N St. N.W. Washington, D.C. 20036 Gordon-ptab@goodwinlaw.com DG-X1-IPR@goodwinlaw.com	Christie L. Larochelle GOODWIN PROCTER LLP 100 Northern Avenue Boston, MA 02210 clarochelle@goodwinlaw.com
Emma L. Murray GOODWIN PROCTER LLP 100 Northern Avenue Boston, MA 02210 emurray@goodwinlaw.com	Natasha E. Daughtrey (pro hac vice) GOODWIN PROCTER LLP 601 S. Figueroa Street, Ste. 4100 Los Angeles, CA 90017 ndaughtrey@goodwinlaw.com

By: /Todd M. Siegel/
Todd M. Siegel, Reg. No. 73,232
todd.siegel@klarquist.com
KLARQUIST SPARKMAN, LLP

One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Tel: 503-595-5300
Fax: 503-595-5301

Counsel for Petitioner

EXHIBIT A

Trials@uspto.gov
571-272-7822

Paper 47
Entered: June 18, 2026

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICROSOFT CORP.,
Petitioner,

v.

X1 DISCOVERY, INC.,
Patent Owner.

IPR2025-00255
Patent 9,633,139 B2

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

SAINDON, *Administrative Patent Judge*.

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

I. INTRODUCTION

A. Background and Summary

Microsoft Corporation ("Petitioner") filed a Petition (Paper 1, "Pet.") requesting an *inter partes* review of claims 1–27 ("the challenged claims") of U.S. Patent No. 9,633,139 B2 (Ex. 1001, "the '139 patent"). The Board instituted an *inter partes* review under 35 U.S.C. § 314. Paper 14 ("Dec. on Inst."). Patent Owner filed a Response (Paper 21, "PO Resp."), Petitioner filed a Reply (Paper 27, "Pet. Reply"), and then Patent Owner filed a Sur-reply (Paper 37, "PO Sur-reply"). An oral hearing was held. Paper 46 ("Tr.").

We have authority to enter this final written decision ("Decision") under 35 U.S.C. § 318(a). The standard for review is set forth in 35 U.S.C. § 316(e), which provides that "the petitioner shall have the burden of proving a proposition of unpatentability by a preponderance of the evidence." For the reasons provided below, we determine that Petitioner has not met this burden of establishing unpatentability of any of the challenged claims.

B. Real Parties in Interest

Petitioner identifies Microsoft Corporation, Acer America Corporation, Acer Inc., ASUS Computer International, ASUSTeK Computer, Inc., Dell (Chengdu) Company Limited, Dell Products LP, and Dell Technologies Inc. as real parties in interest. Pet. xiv. Patent Owner identifies X1 Discovery, Inc. as the real party in interest. Paper 3, 2 (Patent Owner's Mandatory Notices).

C. Related Matters

The parties identify *X1 Discovery, Inc. v. Microsoft Corp.*, No. 8:23-cv-02415 (C.D. Cal.) as involving the ’139 patent and related U.S. Patent Nos. 7,370,035 (“the ’035 patent”) and 10,552,490 (“the ’490 patent”). Pet. xiv; Paper 3, 2; Paper 10, 2 (Patent Owner’s Updated Mandatory Notices). The parties indicate that Petitioner is concurrently challenging the ’035 and ’490 patents in IPR2025-00253 and IPR2025-00254, respectively. Pet. xv; Paper 3, 3. Patent Owner’s Updated Mandatory Notices further identify an additional patent, U.S. Patent No. 8,498,977 (“the ’977 patent”), as being asserted against Petitioner in *X1 v. Microsoft*. Paper 10, 2. Patent Owner reports that the ’977 patent and still another patent, U.S. Patent No. 8,856,093 (“the ’093 patent), also are asserted against Acer Inc. and Acer America Corporation; ASUSTeK Computer Inc. and ASUS Computer International; and Dell Technologies Inc., and Dell Products LP in *Certain Computing Devices Utilizing Indexed Search Systems & Components Thereof*, Inv. No. 337-TA-1389 (ITC), and in the following district court proceedings: *X1 Discovery, Inc. v. Acer Inc.*, No. 2:23-cv-10674 (C.D. Cal.); *X1 Discovery, Inc. v. ASUSTeK Computer Inc.*, No. 2:23-cv-10672 (C.D. Cal.); and *X1 Discovery, Inc. v. Dell Technologies Inc.*, No. 2:24-cv-04109 (C.D. Cal.). *Id.* at 2–3.

D. Overview of the ’139 Patent

The ’139 patent generally relates to incremental, as opposed to delimited, searching. Pet. 10–11 (citing Ex. 1001, 1:38–48, 2:15–27). In a delimited search, the user defines the search and then executes it by clicking a button or hitting the enter key. Ex. 1001, 1:38–67. In contrast, an

incremental search begins and continues as a user types the query. *Id.* at 2:15–27. For example:

> [I]f the user enter[s] "Jo D" documents with the strings ("Jon" and "David"), ("Joe" and "Daniel"), ("John" and "Doe") will be located. If user then adds an "h" to "Jo" so that the search strings are now "Joh D" then the list of search results will be substantially immediately narrowed to present only documents having at least one string beginning with "Joh" and one string beginning with "D", such as a document containing ("John" and "Doe").

Ex. 1001, 22:67–23:8. Using the above example, the '139 patent knows that the user is searching for two words, as opposed to a string of characters containing a space character, by the system having a designated string separator character. *Id.* at 2:53–57 (stating that the '139 patent takes a string and "treat[s] at least a portion of the punctuation marks as string separators"), 29:31–36 ("where punctuation marks are in a word or character string, the word and character string can be selectively broken into multiple words or strings"). In this way, a single text box can be used to enter multiple search strings. *See id.* at 29:44–51; Pet. 11 ("The claims specifically recite incrementally searching two strings entered into the same search field").

E. Challenged Claims

Claims 1–27 (all claims) are challenged. Claims 1, 23, and 27 are independent. Claim 1 is illustrative of the claimed subject matter and is reproduced below.[1]

[1.1] A method of performing a search comprising:

---

[1] Paragraph labels provided by Petitioner have been added in brackets. *See* Ex. 1040 (List of Challenged Claims).

[1.2] receiving a first partial search string in a first search field;

[1.3] receiving a second partial search string in the first search field;

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

[1.5] incrementally searching a set of documents in order to identify search results, wherein the incrementally searching comprises:

[1.6] searching for a first one or more documents each including at least one string that corresponds to the first partial search string;

[1.7] searching for a second one or more documents each including at least one string that corresponds to the second partial search string; and

[1.8] updating the search results 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 one or more documents included in both the first one or more documents and the second one or more documents, wherein the first partial search string and the second partial search string are non-adjacent in at least one of the identified documents.

Ex. 1001, 29:43–67.

F. Prior Art and Asserted Grounds

Petitioner's grounds rely on the following prior art references:

Name	Reference	Exhibit No.
Wu	US Pat. 5,991,756, iss. Nov. 23, 1999	1019
True	US Pat. 6,112,172, iss. Aug. 29, 2000	1009
Wilcox	Adam A. Wilcox, *PC Learning Labs Teaches Lotus Notes 3.0*, Ziff-Davis Press (1993)	1059

Name	Reference	Exhibit No.
Londergan	Stephen Londergan & Pat Freeland, *Lotus Notes R5 For Dummies Quick Reference*, IDG Books Worldwide, Inc. (1999)	1061
Baeza-Yates	Ricardo Baeza-Yates & Berthier Ribeiro-Neto, *Modern Information Retrieval*, ACM Press (1999)	1023
Raskin	Jef Raskin, *The Humane Interface: New Direction for Designing Interactive Systems*, ACM Press (2000)	1021
Sebastian	Bonita Sebastian et al., *Microsoft Office 2001 A Hands-On Guide for Macintosh*, Computer Literacy Press (excerpts) (2000)	1005
Schwartz	Steve Schwartz, *Visual QuickStart Guide Entourage 2001 for Macintosh*, Peachpit Press (2001)	1004
Barber	Nan Barber & David Reynolds, *Office 2001 for Macintosh: The Missing Manual*, Pogue Press/O'Reilly (1st ed.) (Apr. 2001)	1006

Petitioner asserts that the challenged claims would have been unpatentable on the following grounds:

Claim(s) Challenged	35 U.S.C. §	Reference(s)/Basis
1–27	103	Lotus Notes (Wilcox, Londergan), Raskin, Wu
1–27	103	Entourage (Schwartz, Sebastian, Barber), True, Baeza-Yates

Petitioner offers the testimony of Dr. Loren G. Terveen. Ex. 1003; Ex. 1100. Patent Owner offers the testimony of Martin P. Haeberli. Ex. 2001; Ex. 2020.

II. ANALYSIS

A. Legal Standards Used in the Merits Analysis

"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 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 under 35 U.S.C. § 103 if "the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious . . . to a person having ordinary skill in the art to which said subject matter pertains." *KSR Int'l Co. v. Teleflex Inc.*, 550 U.S. 398, 406 (2007). The question of obviousness is resolved on the basis of underlying factual determinations, including: (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 skill in the art; and (4) when in evidence, objective indicia of non-obviousness. *Graham v. John Deere Co. of Kansas City*, 383 U.S. 1, 17–18 (1966). 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.*

Moreover, in determining the differences between the prior art and the claims, the question under 35 U.S.C. § 103 is not whether the differences themselves would have been obvious, but whether the claimed invention as a whole would have been obvious. *Litton Indus. Prods., Inc. v. Solid State Sys. Corp.*, 755 F.2d 158, 164 (Fed. Cir. 1985) ("It is elementary that the claimed invention must be considered as a whole in deciding the question of obviousness."); *see also Stratoflex, Inc. v. Aeroquip Corp.*, 713 F.2d 1530, 1537 (Fed. Cir. 1983) ("[T]he question under 35 U.S.C. § 103 is not whether the differences *themselves* would have been obvious. 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."). As a factfinder, we also must be 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. Level of Ordinary Skill in the Art

The level of skill in the art is "a prism or lens" through which we view the prior art and the claimed invention. *Okajima v. Bourdeau*, 261 F.3d 1350, 1355 (Fed. Cir. 2001). "This reference point prevents . . . factfinders from using their own insight or, worse yet, hindsight, to gauge obviousness." *Id.*

"The *Graham* analysis includes a factual determination of the level of ordinary skill in the art. Without that information, a . . . court cannot properly assess obviousness because the critical question is whether a claimed invention would have been obvious at the time it was made to one

with ordinary skill in the art." *Custom Accessories, Inc. v. Jeffrey-Allan Indus., Inc.*, 807 F.2d 955, 962 (Fed. Cir. 1986); *see also Ruiz v. A.B. Chance*, 234 F.3d 654, 666 (Fed. Cir. 2000) ("The determination of the level of skill in the art is an integral part of the *Graham* analysis.").

Factors pertinent to a determination of the level of ordinary skill in the art include: (1) educational level of the inventor; (2) type of problems encountered in the art; (3) prior art solutions to those problems; (4) rapidity with which innovations are made; (5) sophistication of the technology; and (6) educational level of workers active in the field. *Env't Designs, Ltd. v. Union Oil Co.*, 713 F.2d 693, 696–697 (Fed. Cir. 1983) (citing *Orthopedic Equip. Co. v. All Orthopedic* Appliances*, Inc.*, 707 F.2d 1376, 1381–82 (Fed. Cir. 1983)). Not all such factors may be present in every case, and one or more of these or other factors may predominate in a particular case. *Id.* Moreover, these factors are not exhaustive but are merely a guide to determining the level of ordinary skill in the art. *Daiichi Sankyo Co. v. Apotex, Inc.*, 501 F.3d 1254, 1256 (Fed. Cir. 2007). In determining a level of ordinary skill, we also may look to the prior art, which may reflect an appropriate skill level. *Okajima*, 261 F.3d at 1355.

Petitioner asserts that:

> A person of ordinary skill in the art for purposes of the '139 patent in September 2002 ("POSITA") 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. EX1003, ¶30.

Pet. 6.

Patent Owner does not comment on Petitioner's formulation of the level of ordinary skill, or offer its own formulation in its Response. Patent Owner's declarant, however, states:

> 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 Patent Owner's formulation, finding it to be consistent with the evidence of record at this stage of the proceeding, including the asserted prior art and the '139 patent. Dec. on Inst. 8. Following a review of the full record, we agree that Petitioner's definition of one of ordinary skill in the art is "not meaningfully different" from Patent Owner's for the purposes of this proceeding. We continue to view Patent Owner's proposed definition as consistent with the '139 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.

C. Claim Construction

As stated in 37 C.F.R. § 42.100(b),

> a claim of a patent . . . shall be construed using the same claim construction standard that would be used to construe the claim in a civil action under 35 U.S.C. 282(b), including construing the claim in accordance with the ordinary and customary meaning of such claim as understood by one of ordinary skill in the art and the prosecution history pertaining to the patent. Any prior claim construction determination concerning a term of the claim in a civil action, or a proceeding before the International Trade Commission, that is timely made of record in the *inter partes* review proceeding will be considered.

37 C.F.R. § 42.100(b). Thus, under 37 C.F.R. § 42.100(b), we apply the claim construction standard set forth in *Phillips v. AWH Corp.*, 415 F.3d 1303, 1312–13 (Fed. Cir. 2005) (en banc). As stated in *Phillips*, "the words of a claim 'are generally given their ordinary and customary meaning' . . . that the term would have to a person of ordinary skill in the art in question at the time of the invention." *Id.* at 1312–13.

Petitioner does not assert any explicit claim constructions. Pet. 12. In our Decision on Institution, the Board construed "string separator character" as "a single-character string separator." Dec. on Inst. 10, 28–29. Patent Owner explicitly agreed with this construction in its Response (Resp. 8–9), and Petitioner offered no further comment in its Reply. Although no claim construction issue is squarely before us now, we maintain our prior construction of "string separator character," and do not construe any further limitations. *See Realtime Data, LLC v. Iancu*, 912 F.3d 1368, 1375 (Fed. Cir. 2019) ("The Board is required to construe 'only those terms . . . that are in controversy, and only to the extent necessary to resolve the

controversy.'") (quoting *Vivid Techs., Inc. v. Am. Sci. & Eng'g, Inc.*, 200 F.3d 795, 803 (Fed. Cir. 1999)).

D. Overview of the Prior Art

1. Art of Ground 1

Petitioner asserts that claims 1–27 would have been obvious in view of Wilcox, Londergan, Raskin, and Wu. Pet. 12–45. Wilcox and Londergan discuss software called Lotus Notes; our discussion of "Lotus Notes" will be of features described in one or both of these references. We will describe the relevant disclosures of these references below.

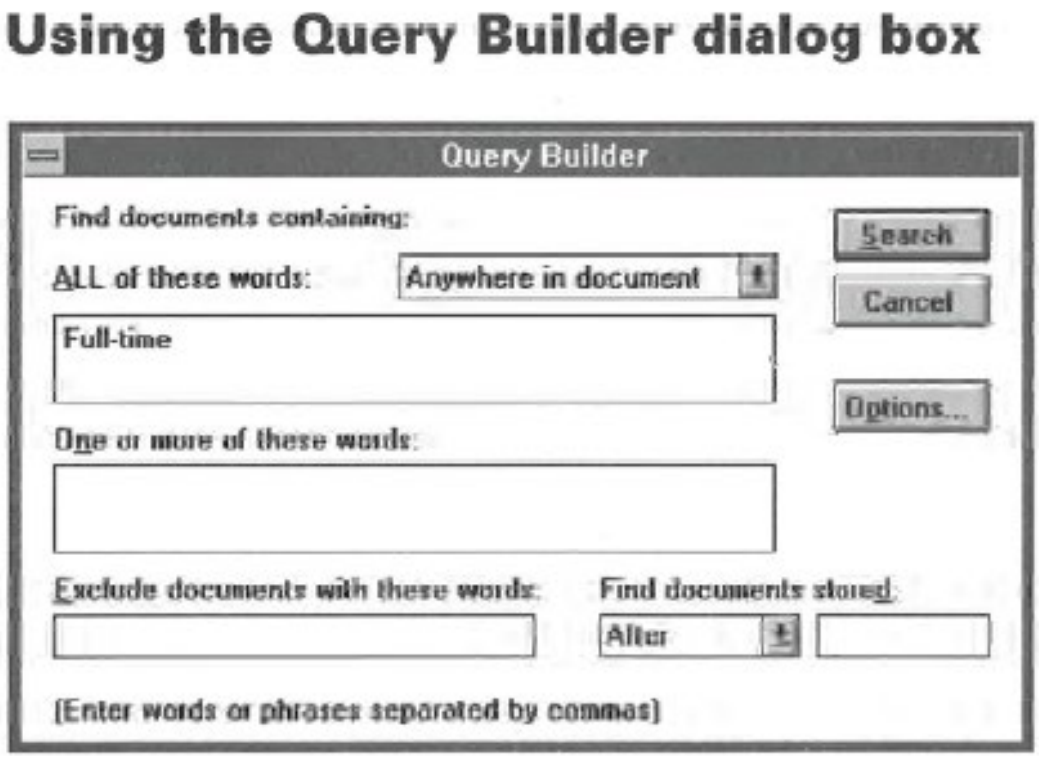
a) Lotus Notes References

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). 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 an electronic mail software. Ex. 1059, 5, Ex. 1061, 11. These 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 citing Wilcox primarily rely on the Query Builder, a dialog box that makes it easier for users to construct complex searches. *See* Pet. 26 (citing Ex. 1059, 163).

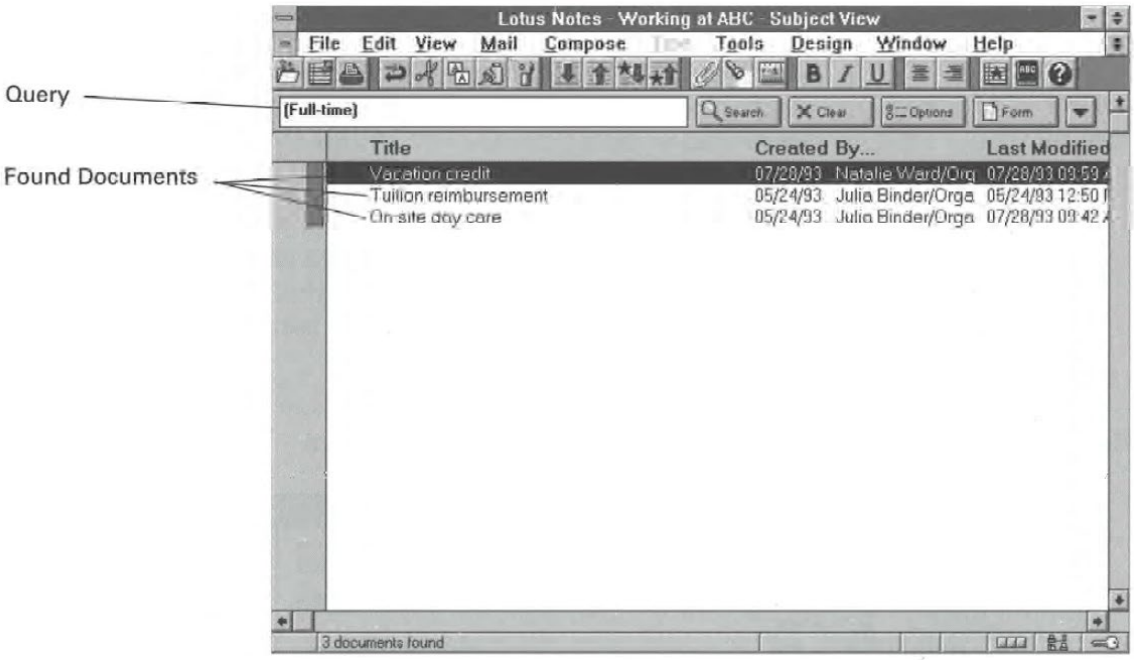
Wilcox Figure 5.16, reproduced below, depicts the Query Builder dialog box:



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

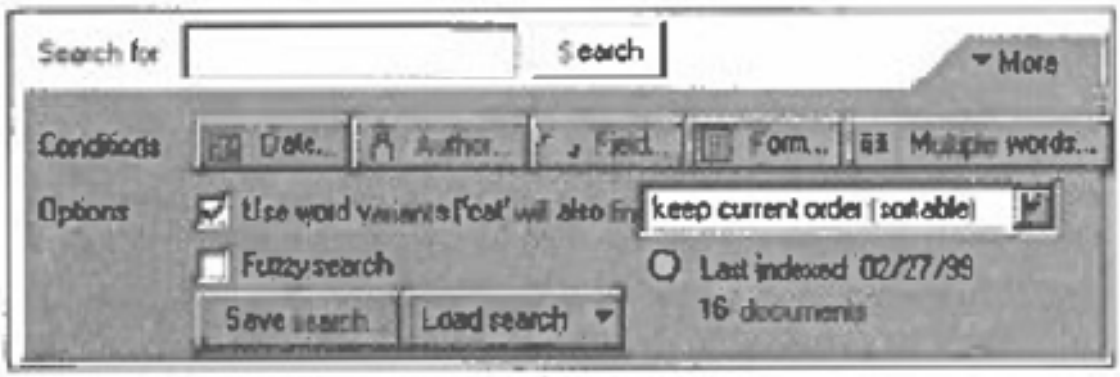


Ex. 1059, 164. Figure 5.17 depicts a search result window that displays the user's query in a search bar at the top of the window, with a list of found documents underneath. The list displays document titles, create dates, authors, and last modified dates, but does not show the documents' contents. *Id.*,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

(2) Londergan

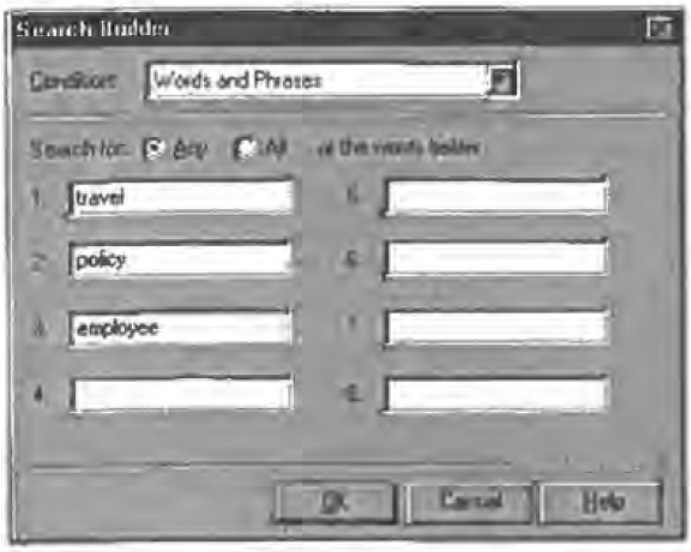
Londergan describes the search functionality of Version R5 of Lotus Notes. Ex. 1061, (i), 139–48. The 1999 version of Lotus Notes (R5) eliminated the Query Builder feature in favor of a different feature with different functionality called "Search Builder." Ex. 2001 ¶ 82. Search Builder provides multiple 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 (Ex. 1021) is a textbook describing user interface design principles for interactive systems. Ex. 1021, (i), (xi)–(xii). 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 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 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 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. Petitioner's contentions regarding Wu are uncontested, and this reference receives little attention from the parties. Thus, no further discussion of Wu is necessary.

2. Art of Ground 2

Petitioner asserts that claims 1–27 would have been obvious in view of Schwartz, Sebastian, Barber, True, and Baeza-Yates. Pet. 46–93. We briefly describe these references below.

a) Entourage References

Schwartz, Sebastian, and Barber discuss software called Entourage; our discussion of "Entourage" will be of features described in one or more of these references. Entourage for the Macintosh was generally similar to modern-day Microsoft Outlook. *See, e.g.*, Ex. 1004, 1 (comparing Entourage to Outlook Express, but with more features such as address book, calendar, tasks, etc., all present in modern-day Outlook). Petitioner

primarily relies on two independent features of Entourage—filter and Find. *See, e.g.*, *id.* at 8–12, 161–164. Find has a further sub-feature called Advanced Find. *See, e.g.*, *id.* at 11–12, 162–163. The filter and Find features are available in most Entourage components (e.g., email, address book). *Id.* at 8.

The Entourage references serve as the chief references in Petitioner's second ground. See Pet. 46–93. In particular, Petitioner cites to Entourage's filter to teach incremental search, and Find to teach searching within a document's text. *See, e.g.*, *id.* at 63, 77. We will next walk through these features.

(1) Entourage's "filter" Feature

Schwartz states that "[p]erhaps the easiest way to find a particular message, contact record, note, or any other Entourage item is to filter the item list. Ex. 1004, 8. The filter feature is made available "[a]bove the list in many Entourage windows" in the form of "one or two pop-up menus and a text box." *Id.* Figure 1.12 of Schwartz provides an example, and is reproduced below:

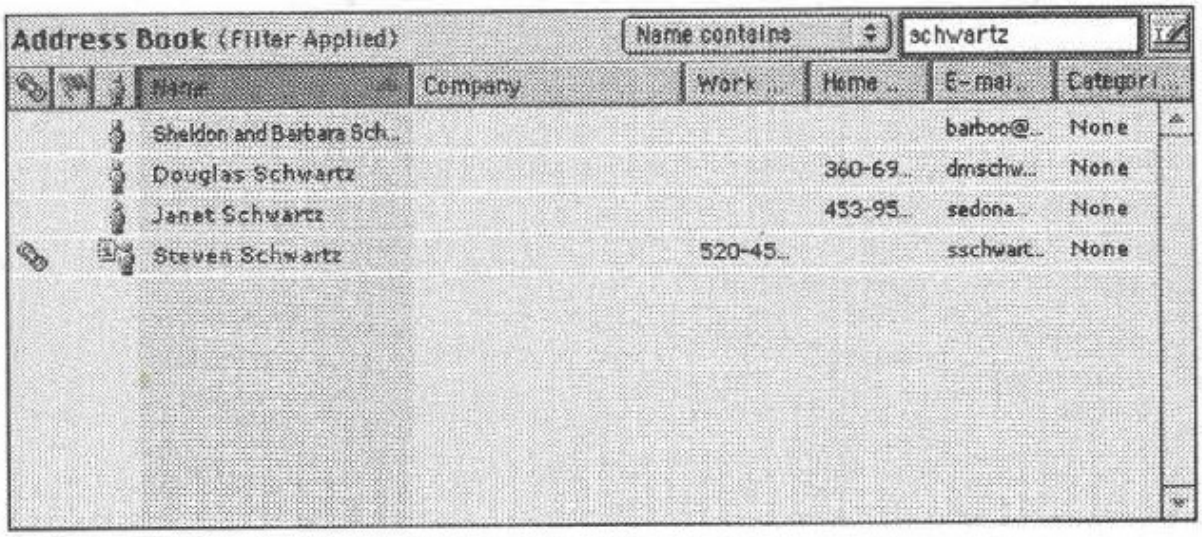


Figure 1.12 As you type the filter text, only matches are shown in the list.

Ex. 1004, 9. Figure 1.12 of Schwartz depicts the address book window with a filter applied. *Id.* A button called "Name contains" is selected above an address book window, with the string "schwartz" typed into an adjacent text box. *Id.* The entries displayed in the list all appear to include the string "schwartz" in the "Name" column. *Id.* Notably, when using a filter, "[a]s you type the filter text, only matches are shown in the list," indicating an incremental search capability. *Id.* (caption of Figure 1.12).

The filter feature applied to email includes selection criteria limited to fields like "Subject, From, To, or Category." Ex. 1004, 161. Figure 11.15 of Schwartz is reproduced below:

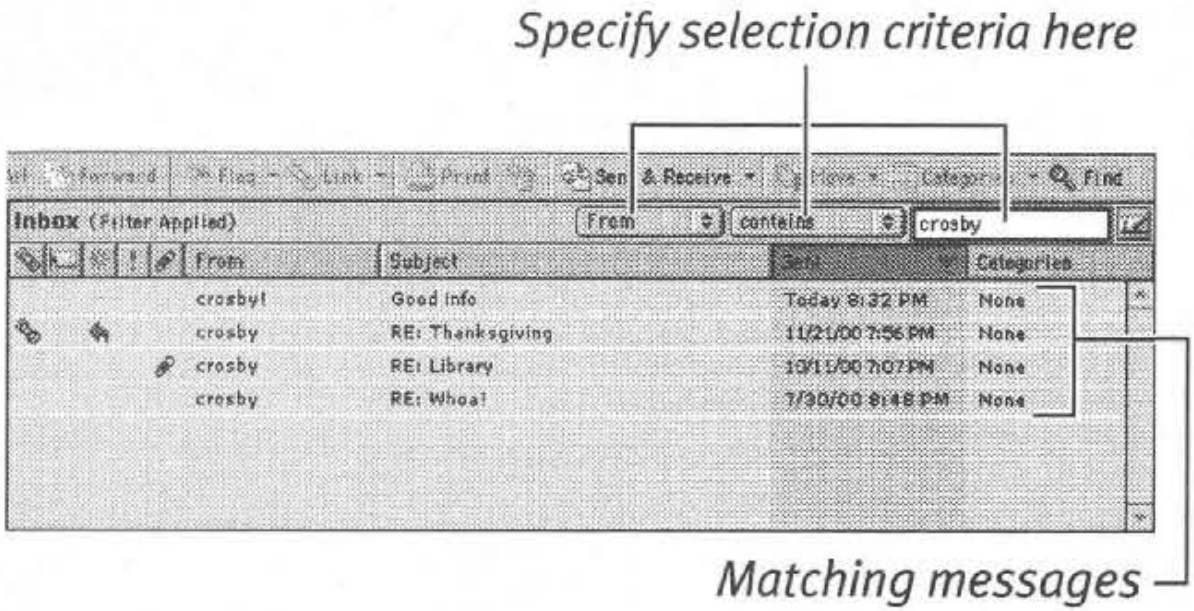


Figure 11.15 To locate messages quickly based on Subject, From, To, or Category information, you can enter selection criteria above the message list.

Ex. 1004, 161. Figure 11.15 of Schwartz depicts an inbox with various buttons and a text field above the inbox. *Id.* The filter interface is labeled with a note that says "Specify selection criteria here," and includes buttons that allow the user to select criteria for filtering. *Id.* Here, "From" and "contains" are chosen, with "crosby" typed into a text box. *Id.* Figure 11.15 contains a caption that states, "To locate messages quickly based on Subject, From, To, or Category information, you can enter selection criteria above the message list." *Id.* Figure 11.15 of Schwartz, depicting an email filter, is generally similar to Figure 1.12 of Schwartz, depicting an address book filter. *Compare id. with id.* at 9.

Petitioner also cites Sebastian for showing a filter (Pet. 48, citing Ex. 1005), but there is no further information made available at that cite; it

merely shows a figure depicting user's inbox that in general appears very similar to the filter function in Schwartz depicted in Figure 11.15, described above. Ex. 1004, 161; Ex. 1005, 173.

Barber briefly discusses filters. Ex. 1006, 384. In particular, Barber states that:

> You can *filter* messages, notes, tasks, and addresses easily enough (that is, hide all but a certain group) by typing into the little box above the message, task, notes, or address list; for example, you might configure the companion pop-up menus to say, for example, "Subject" and "Contains," and then type *humor* into the box.

Id. This disclosure is consistent with Schwartz, discussed above.

The most notable feature of the filter function is that results are displayed "[a]s you type the filter text," i.e., incremental searching. Ex. 1004, 9. According to Schwartz, "[a]lthough though it is not as sophisticated as the Find and Advanced Find commands discussed later in this section, filtering is the fastest way to perform a simple search." *Id.* Barber adds, "when the needle still eludes you in your rummage through the Entourage haystack, you can use Entourage's slower, more powerful search," referring to the Find command. Ex. 1006, 384.

(2) Entourage's "Find" Feature

Separate from the filter feature is the Find feature. *See, e.g.*, Ex. 1004, 161–163. According to Schwartz, "[t]he Find command is slightly more powerful than filtering. In addition to searching for matching text in an item title, it can search for matches within the body text of all messages or only the current message." *Id.* at 9. Thus, in contrast to the filter, Find allows a user to search for text within the body of a message. *Id.* Find does not use the same interface that the filter uses; instead, Schwartz describes "Find" as

appearing in a dialog box (a separate pop-up window). *Id.* A dialog box requires the user to fill out a form to select various options and then to click a button to begin the search and see the results. *Id.* at 9–10. Figure 1.13 of Schwartz, reproduced below, depicts a Find dialog box.

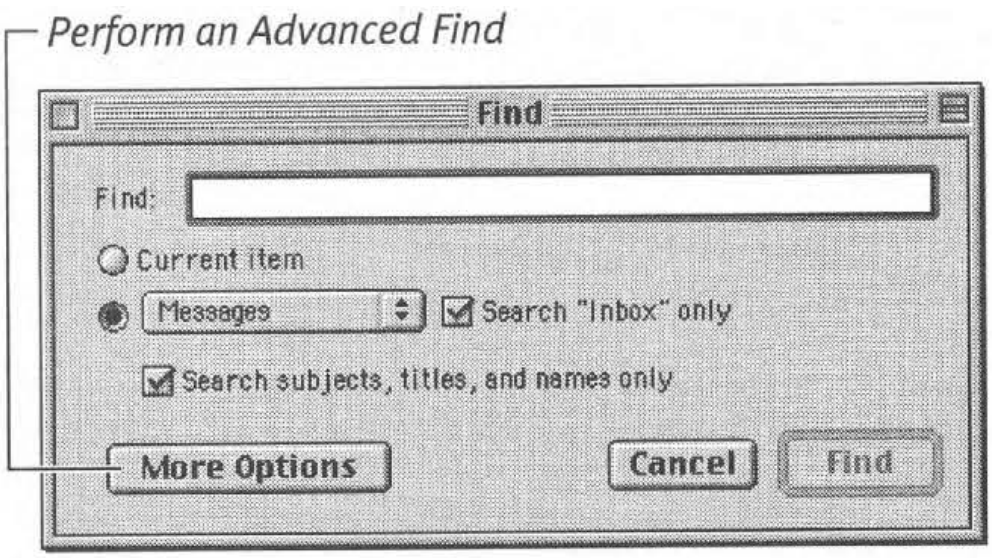


Figure 1.13 You can conduct simple searches by specifying criteria in the Find dialog box.

Ex. 1004, 9. Figure 1.13 of Schwartz depicts a "Find" dialog box, and the caption states that "You can conduct simple searches by specifying criteria in the Find dialog box." *Id.* Schwartz describes the various steps for using the Find dialog box, such as selecting what is being searched, defining text to find, and clicking on a button to execute the search. *Id.* at 9–10.

Once the Find search executes, "results are displayed in a separate window," as shown in Figure 1.14, reproduced below.

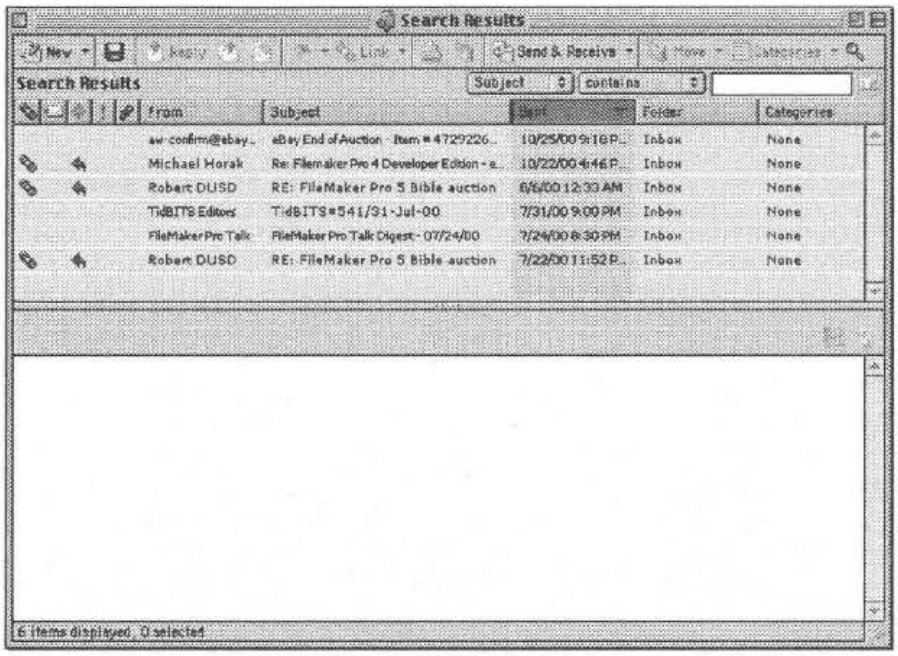


Figure 1.14 When you conduct a Find or Advanced Find, search results are displayed in a separate window. In this example, I searched for messages in the Inbox that contained the word *FileMaker*.

Ex. 1004, 10. Figure 1.14 of Schwartz displays a separate window with the title "Search Results." *Id.* The caption of Figure 1.14 states that "search results are displayed in a separate window" and that the search results displayed are "for messages in the Inbox that contained the word *FileMaker*." *Id.*; *see also id.* at 162 (Fig. 11.17) (stating in the caption that "[s]earch results are displayed in a new window").

(a) Entourage's Ability to Highlight Text Search Results

Schwartz states that once the search results are displayed in the new window, "[t]o examine any of the matches, click its header and read the message in the preview pane." *Id.* at 162 (Fig. 11.17). That is to say, the

search results provide a list of hits. To locate and display particular text within a document, a user has to perform a separate action. *Once a message is selected*, then a <u>*user*</u> can start a *separate search* for the text within that selected message. *Id.* at 10–11 (describing how "[t]o search the current item for a text string"), 163–164 (describing how "[t]o search the current message for a text string"). In particular, as depicted in the "Find" dialog box of Figure 1.13, reproduced earlier, if a user instead chooses to search within the "Current item," then the search is redirected to only look at the selected item, allowing the user "[t]o search the current item for a text string." *Id.* at 10–11. Figure 11.19, reproduced below, shows a user about to search within the body of a message:

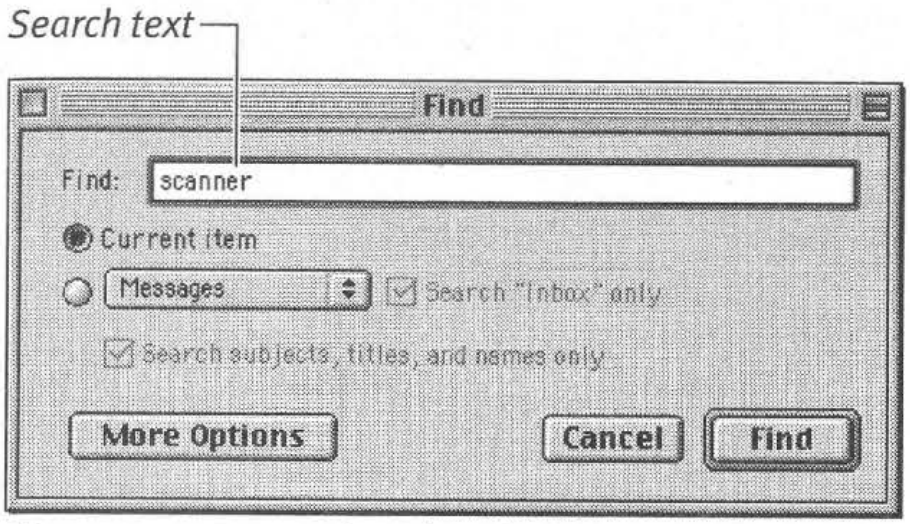


Figure 11.19 To search within the body of the current message for a particular text string, click the Current item radio button in the Find dialog box. All other options are disabled.

Ex. 1004, 163. The caption of Figure 11.19 states, "[t]o search within the body of the current message for a particular text string, click the Current item radio button in the Find dialog box. All other options are disabled." *Id.*

In the image of Figure 11.19, the "Current item" option is selected and "scanner" is typed into a text box labeled "Find." Other options normally available in the Find command are colored gray to indicate that they are disabled.

If there is a hit, "[t]he first instance of the text string . . . is highlighted." *Id.* at 11. Figure 1.16, reproduced below, shows the results of a search within a selected email item:

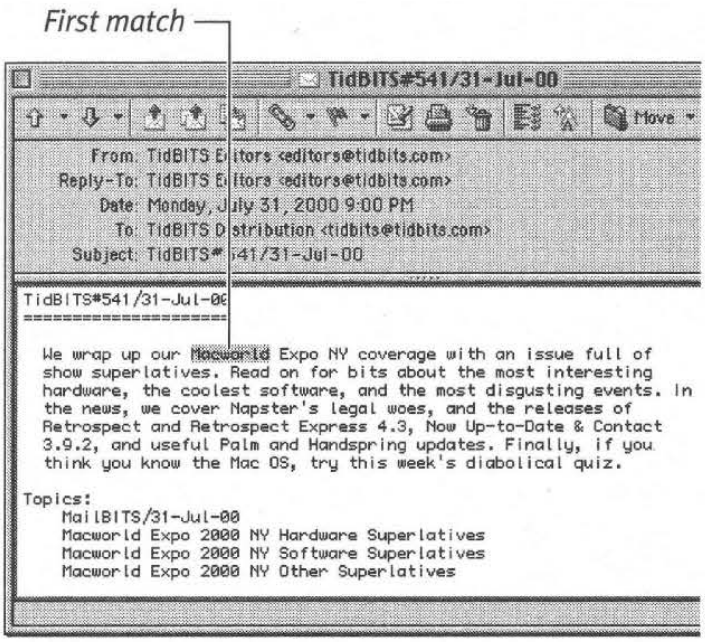


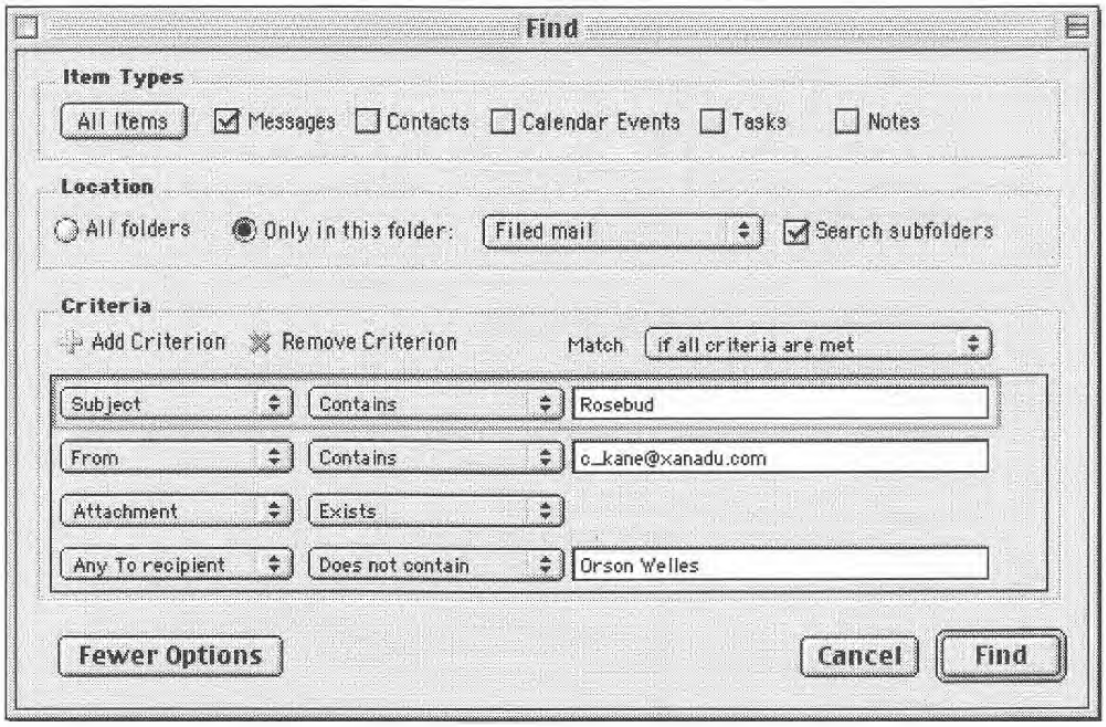
Figure 1.16 When you search for matching text within an item, Entourage highlights the first match, scrolling as necessary to display it.

Ex. 1004, 11; *see also id.* at 163 (Figure 11.20, showing the results of the search in Figure 11.19 for "scanner"). The caption of Figure 1.16 states that "[w]hen you search for matching text within an item, Entourage highlights the first match, scrolling as necessary to display it." The Figure shows an open window that contains an email message, and in the body of the message the word "Macworld" is highlighted. The highlighted word is annotated as the "First match." Accordingly, the Find feature provides two

modes, one to find an item (potentially containing certain text) and a second to find and display text within a selected item.

(b) Entourage's "Advanced Find" Feature

The Advanced Find feature is accessed through the Find pop-up by clicking "More Options" and provides ways for a user to "perform a more complex search (such as one that uses multiple criteria)." *See, e.g.*, *id.* at 162–163 (caption of Figure 11.18); *see also id.* at 11–12 (providing another overview of Advanced Find); Ex. 1006, 385–386 (providing yet another overview of Advanced Find). For example, the Advanced Find allows a user to look for items "If Any Criteria Are Met" or "If All Criteria Are Met." Ex. 1004, 163; *see also id.* at 12 (similar). Figure 11-7 of Barber, reproduced below, depicts a dialog box showing Advanced Find:



Ex. 1006, 387. Figure 11-7 depicts a search in "Messages," only in the folder "Filed mail" and its subfolders, where all of the following criteria are met: the subject field contains the word "Rosebud," the from field contains c_kane@xanadu.com, and any "To" recipient does not contain "Orson Welles."

(c) Summary of Entourage

In sum, the Find and Advanced Find features are separate from the filter feature, and require a user to open a dialog box to build up a search query by selecting multiple options and then executing the search that takes into account all of the options selected. *See, e.g.*, Ex. 1004, 9–12. Results of Find are provided in another separate window, and if the user wants to find and highlight particular words within the body of a message found in the results, the user has to click on the particular message and perform a separate search within that message. *See, e.g.*, *id.* at 9–11. The filter feature, on the other hand, allows for searching within various defined categories (essentially, metadata) though a list of items displayed. *See, e.g.*, *id.* at 8–9. Unlike the Find feature, which uses a pop-up window to build a search query, and then displays results in yet another window, the filter works in the current window, and updates matches as the user types. *See id.*

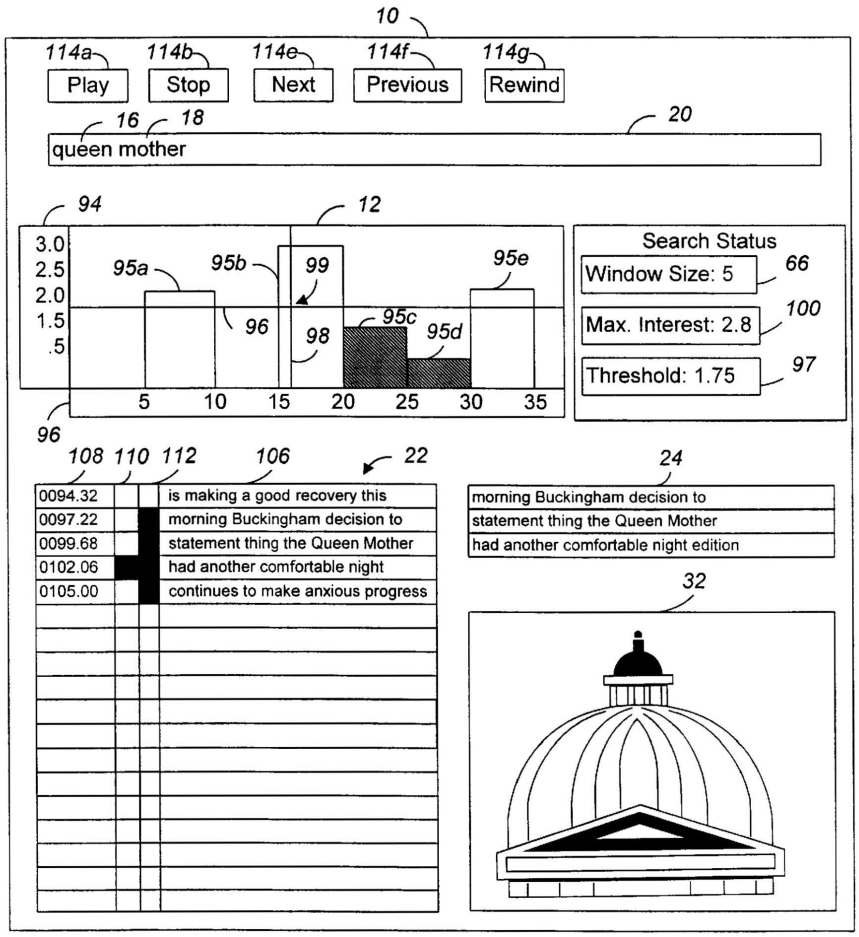
b) True

Petitioner cites to True for incrementally searching for multiple search strings entered in a single search field. Pet. 53. True provides a system for interactive searching wherein the searchable information "may be derived from an audio signal that is linked to a video signal from, e.g., a television broadcast feed." Ex. 1009, 1:31–33. Because the text is derived from audio

tracks, the underlying text data "may not always accurately represent the portions of the underlying information . . . that the user is trying to find." *Id.* at 1:19–22. In particular, the information "is a text file generated by a speech recognizer" and "the words produced by a speech recognizer are sometimes not the words that were spoken." *Id.* at 1:22–27. Thus, the recognized words are associated with a confidence interval that the spoken word (i.e., sound) is the recognized word (i.e., text). *Id.* at 4:21–37. If a user is looking for the combination of words "queen mother," then the system will look for "queen" and "mother" separately and combine their confidence intervals to arrive at a confidence in the results of having found "queen mother." *Id.* at 5:41–51; *see also id.* at 5:52–6:22, Fig. 2 (providing details and an implementation example).

True presents the search results graphically, indicating where in the audio track the particular words the user is looking for are likely to be, as shown in Figure 2, reproduced below:

FIG. 2



Ex. 1009, Fig 2. Figure 2 of True shows a user searching for the words "queen mother" using query window 20. *Id.*; *see also id.* at 3:20–33. Results are displayed in bar chart format in result window 12, with the transcript divided into time segments, and the heights of bars 95a, 95b, 95c, etc. representing True's estimate of the confidence that the searched-for words appear in that portion of the audio transcript. *Id.* at 5:29–39, 6:6–14. Estimates of where the phrase "queen mother" may appear are listed with the phrase in context in transcript window 22. *Id.* at 6:38–44. The user can play back the audio using navigation buttons at the top of the window (114a,

114b, etc.) and can watch the corresponding video feed 32. *Id.* at 6:45–61, 7:13–24.

True suggests an incremental search. *Id.* at 5:5–6 ("the user can view the results of a query as the query is being modified"). If a user specifically wants one of the words typed to appear in the results, they must place a plus sign before the query term. *Id.* at 5:26–28. The query is performed on the individual words, adding together the confidence level of the accuracy of the speech recognition for each of the words appearing in a window of time. *Id.* at 5:35–50. Thus, a high-scoring result could be from a single word appearing multiple times within a given window of time.

In sum, True takes a search string that has multiple words in a single field, and looks for those words separately using a confidence interval analysis to allow the user to see if the searched-for words actually appear in the transcript and were likely said.

c) Baeza-Yates

Petitioner cites to Baeza-Yates to teach independently searching for multiple strings and taking the intersection of those results. Pet. 70. Petitioner's contentions regarding Baeza-Yates are uncontested, and this reference receives little attention from the parties. Thus, no further discussion of Baeza-Yates is necessary.

E. Obviousness Analysis of Claims 1–27 in view of Wilcox, Londergan, Raskin, and Wu

Petitioner asserts that claims 1–27 would have been obvious in view of Wilcox, Londergan, Raskin, and Wu. Pet. 12–45. Wilcox and Londergan are the "Lotus Notes" references. Petitioner asserts that the Lotus Notes references teach every element of the claims except incremental search,

which Petitioner asserts is taught in Raskin, and an intersection approach, which Petitioner asserts is taught in Wu. *Id.* at 12–13. Patent Owner argues, *inter alia*, that Petitioner has not shown a reason to combine Lotus Notes with Raskin with a reasonable expectation of success. PO Response 17–28. In particular, Patent Owner argues that, as a technical matter, Raskin's incremental search is a different kind of search compared to the searches in Lotus Notes, and that Petitioner has not sufficiently addressed these differences. *See, e.g.*, *id.* at 26–27. Petitioner replies that it is only taking the idea of incremental searching from Raskin, and that Patent Owner is requiring a bodily incorporation of Raskin's search into Lotus Notes' search. Pet. Reply 8–9, 15–16. Patent Owner then reiterates its points about the differences between the two types of searching in Lotus Notes and Raskin. PO Sur-reply 9–10. Patent Owner also disputes Petitioner's "bodily incorporation" characterization, pointing out that Patent Owner is not discussing bodily incorporation, but rather an incompatibility between two different types of systems. *Id.* at 13–14. We have read and considered the arguments in the briefs, and the evidence cited therein. We focus our analysis below on independent claim 1, which is illustrative of the dispositive issues in this case.

The query builder and search builder in Lotus Notes conduct searches by *building* the search/query *and then* executing it, and the purpose of the search is to find *documents*. *See, e.g.*, Ex. 1059, 163 (noting Figure 5.16, depicting a dialog box stating "Find documents containing" various text), 164 (noting Figure 5.17, showing the results of the search as a list of documents); Ex. 1061, 140 (stating that, "when you ask Notes to search for a word, Notes finds all the documents for you"); Ex. 2001 ¶ 52. For example,

in Wilcox, the query builder has a number of text boxes that a user fills out in order to build the search, then executes it with all of the text boxes considered at once. Ex. 1059, 162–164; PO Resp. 19 (arguing that "[a]ll of the search tools in Lotus Notes references are delimited searches; requiring a user to click 'Search' to perform the search"). The query builder is gathering information to "translate[] your request into Notes query language." Ex. 1059, 162; *see also id.* at 167 (by using the Form mode in the search bar, the user can also create queries on text in a particular field). That is, Lotus Notes takes all of the information entered by the user to build up, and then submit, a query for the user. *Id.*; *see also* Ex. 2020 ¶ 34 (Patent Owner's declarant stating that, "[i]n a delimited search interface, the user must complete their query entry and then explicitly click a 'Search' button of some kind to initiate the search operation and receive results"). Thus, in a delimited search, all of the information needs to be gathered before the query can be built and submitted. The newer version of Lotus Notes discussed in Londergan uses a search bar, but it has largely the same configuration – searching over multiple fields in a single search. Ex. 1061, 140–142; *see also id.* at 143–144 (noting the query form, like in Wilcox, exists behind the search bar).

The type of delimited searching performed in the query and search forms of Lotus Notes is not the same as searching within an open document to find and move to a word in that document. Ex. 2020 ¶ 15 (Patent Owner's declarant contrasting scanning of a single document versus search for multiple documents), ¶¶ 34–37 (contrasting delimited and incremental searches). For example, Londergan says that "you may simply want to find some text in the document you're reading," and in such case, *instead* of

using the query builder, Londergan instructs the user to use the simple "Find/Replace" function. Ex. 1061, 148. In other words, Lotus Notes recognizes the difference between building a query to find documents and simply looking for a word or words in a document, and provides separate functionality to do so. *See* Ex. 2020 ¶ 31 (Patent Owner's declarant stating that the various search features in Lotus Notes are intentionally different because "each reflect different assumptions about how a user selects search terms, how results are filtered, what subset of documents is subject to the search, and how precisely a delimited search is initiated").

We agree with Patent Owner that a problem with Petitioner's combination is that it proposes to take the *idea* of incremental searching from Raskin and divorce it from Raskin's teaching of incremental searching. *See* PO Resp. 20 (arguing, "a [person of ordinary skill in the art] 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"); *see also id.* at 17–28 (setting out the whole argument); Ex. 2020 ¶¶ 39–62 (Patent Owner's declarant explaining why Petitioner's approach is "technically problematic"). Raskin's incremental search is tied to finding and moving to character strings within a document. Ex. 1021, 125. For example, if a user were to type a string that was found in the text, "the instance [of that string] is selected and the cursor placed immediately after the end of the selection." *Id.* The cursor can move about the text as characters of the search string are added or deleted. *Id.* at 125–126. Thus, Raskin discloses in-text search, even when searching across documents. *Id.* at 132 ("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."). Raskin even distinguishes its text-finding search from Lotus Notes' document-finding search, stating that "[i]t is usually easy to tell in what domain you are searching, because you see the results in context; *what you see is not just a list of file names, as with many present search systems*." *Id.* (emphasis added); *see also* PO Sur-reply 16 (citing the same page and arguing that "Raskin recommends doing the *opposite* of what is claimed").

This context gives credence to Patent Owner's argument that "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." PO Resp. 26–27. Indeed, Petitioner has given little-to-no explanation of how one takes the "idea of incremental search" and applies it to Lotus Notes. Ex. 2021, 116:20–117:17 (Petitioner's declarant testifying that "Raskin is not explicitly talking about searching across a set of documents [at once], but the teaching that I relied on in general from Raskin is the idea of incremental searching," and "I did not have to talk about implementation level details or incorporating a particular implementation. That wasn't necessary for my opinion."). Raskin does not teach a mode of incremental searching that applies to the query-style, document-finding searching in Lotus Notes. *See generally* Ex. 1021; *see also* Ex. 2020 ¶¶ 34–35 (Patent Owner's declarant contrasting Lotus Notes and Raskin). Raskin even states that it is "difficult or impossible to implement incremental searches" using standard interface-building tools. Ex. 1021, 126; Ex. 2020 ¶ 60 (Patent Owner's declarant stating that this

passage "directly undermines [Petitioner's declarant's] suggestion that a [person of ordinary skill in the art] would have been motivated to implement Raskin's incremental search approach in the Lotus Notes system").

Although Petitioner calls Raskin's statement a "red herring," we do not agree. *See* Pet. Reply 15. Even if a person of ordinary skill in the art *could* implement an incremental search using C or C++ instead of the interface-building tools discussed in Raskin, the scope of our inquiry is not mere technical capability. *See* Pet. Reply 15–16 (asserting that C or C++ "allowed an ordinary programmer to build a [user interface] that supported incremental searching"); Ex. 1098, 24:29–35:6 (Patent Owner's declarant agreeing that a person of ordinary skill in the art could have made a "dynamic text box" using C++); *see also KSR*, 550 U.S. at 418 ("a patent composed of several elements is not proved obvious merely by demonstrating that each of its elements was, independently, known in the prior art"). Petitioner does not offer evidence of incremental searching in a context similar to Lotus Notes, or otherwise explain how a person of ordinary skill in the art would take the apples of Raskin's within-a-document text search and apply it to the oranges of Lotus Notes' query builder search performed across a large set of documents. Ex. 2020 ¶ 65 (Patent Owner's declarant testifying that Petitioner's declarant offers an opinion "without grappling with the implementation hurdles that Raskin itself identified"); *see also id.* ¶ 17 ("I am not aware of any prior art before 2002 which discloses incrementally searching for and locating documents. None of [Petitioner's declarant's] cited references disclose such a search system."), ¶¶ 18–25 (persuasively explaining how the cited background references do not teach incremental document search). Petitioner then takes the disembodied idea of

a database-wide incremental search, not disclosed or taught in any reference, and then proposes to add it to Lotus Notes (itself not one but two separate and distinct references), waving away any potential issue of how one would do so in a manner that would yield the claimed invention. *See* PO Resp. 17 (arguing that the combination is faulty because "the only stated motivation to combine the teachings of the Lotus Notes references 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"); Ex. 2020 ¶ 31 (Patent Owner's declarant stating that the various features in Lotus Notes are separate because "each reflect different assumptions about how a user selects search terms, how results are filtered, what subset of documents is subject to the search, and how precisely a delimited search is initiated").

Petitioner has not persuasively shown that Raskin teaches an incremental search of the type required by the claims, much less that one of ordinary skill in the art would have realized that this type of search could improve Lotus Notes' query or search builder functions. Raskin is talking about a different kind of incremental search. Even assuming, *arguendo*, that we were persuaded that one of ordinary skill would have had reason to take the disembodied idea of Raskin's incremental search and combine it with Lotus Notes, we would still be left with the fundamental question of how Lotus Notes would need to be modified to implement an incremental search, given that the query builder (or search builder) is premised on *building* a search, which includes the ability to complete several fields before executing the built search. *See, e.g.*, Ex. 1059, 163 (Fig. 5.16); Ex. 1061, 141. We agree with Patent Owner that the Lotus Notes style of searching does not

appear compatible with an incremental search. PO Sur-reply 9–10; Ex. 2020 ¶¶ 79–80 (contrasting the two types of search), 82–83 (highlighting the open questions of what Petitioner's combination actually would be); *see also id.* at 63–71 (detailed explanation of the problems with the combination). We determine that Petitioner has not shown, by a preponderance of the evidence, that a person of ordinary skill in the art would have considered it obvious to add (or would have succeeded in adding) incremental search to the Lotus Notes query builder or search builder functionalities relied on by Petitioner. Petitioner's position for claims 2–27 repeat or rely on this position and are likewise found unpersuasive. *See* Pet. 32–45.

F. Obviousness Analysis of Claims 1–27 in view of Schwartz, Sebastian, Barber, True, and Baeza-Yates

Petitioner asserts that claims 1–27 would have been obvious in view of Schwartz, Sebastian, Barber, True, and Baeza-Yates. Pet. 46–93. Schwartz, Sebastian, and Barber discuss software called Entourage; our discussion of "Entourage" will be of features described in one or more of these references. Patent Owner's Response chiefly argues that Petitioner is mixing and matching disparate features in Entourage (PO Resp. 38–49), that there is no reason to combine the teachings of the references (*id.* at 56–62), that Entourage does not teach entering two search strings in the same field (*id.* at 63), and that True does not teach searching for documents (*id.*). Petitioner's Reply seeks to clarify its position on True (Pet. Reply 9–10), Entourage (*id.* at 19–20), and searching across documents (*id.* at 22). Patent Owner's Sur-Reply identifies Petitioner's allegedly "shifting and internally inconsistent positions" (PO Sur-reply 2) and discusses True further (*id.* at 24–25). We have read and considered the arguments in the briefs, and the

evidence cited therein. We focus our analysis below on independent claim 1, which is illustrative of the dispositive issues in this case.

1. Discussion of Petitioner's Element 1.2–1.4 Analysis

Claim 1 requires receiving first and second partial search strings in a first search field, wherein the strings are separated by a string separator character. Rather than mapping the references to the claim element-by-element, Petitioner asserts that the "Entourage references in combination with True teach these claim elements, especially in view of the knowledge of a [person of ordinary skill in the art]." Pet. 58. Petitioner begins by pointing out that Schwartz teaches receiving one search string (*id.*, citing Ex. 1004, 9, 38) "and at least suggest[s] receiving two search strings" (*id.*, citing Ex. 1006, 387 (Figure 11-7, providing the name "Orson Welles" in a search field (reproduced *supra*)). Petitioner also asserts that "[t]he Entourage references also expressly disclose 'searching for text strings' (plural), at least suggesting to a POSITA to receive multiple search strings in the single search field of Entourage's incremental Quick Find search." *Id.* at 58–59 (citing Ex. 1004, 269). Lastly, Petitioner asserts that "[a person of ordinary skill in the art] would have been motivated to incorporate True's express teachings to receive two search strings in the search field." *Id.* at 60 (citing Ex. 1009, 2:26–31, 2:39–48, 4:38–43, 4:65–5:6, 5:41–50, Fig. 2). Petitioner has therefore set out three positions for how Entourage or True disclose the claimed first and second partial search strings.

Petitioner's first position, that Entourage "at least suggest[s] receiving two search strings," relies on an example screenshot showing a user's search, which includes the name "Orson Welles." Pet. 58 (citing Ex. 1006, 387). The problem with this position is that the string "Orson Welles" is not

two strings. Semantically, in a human mind, "Orson Welles" contains two words separated by a space. But there is no indication that "Orson" or "Welles" are searched separately by Entourage as required by the claim; indeed, they are treated as a unitary string (a name) for a reason. Ex. 1006, 387 (explaining how to remove hits to "Orson Welles" from search results in order to ignore emails that were sent to a person with that exact name); Ex. 2001 ¶ 151 (Patent Owner's declarant stating that "there is no evidence that when a user types 'Orson Welles' into this Find command text box that a search is performed on 'Orson' and 'Welles' separately as opposed to searching on the full text string 'Orson Welles.'").

Petitioner's second position, that the Entourage references expressly disclose searching for "text strings" (plural) relies on a citation to a page that is not of record. *Id.* at 58–59 (citing Ex. 1004, 269). Our copy of Exhibit 1004 does not have 269 pages, and if it did it would be part of the index (according to the index). Ex. 1004, vii; *but see* Ex. 2001 ¶ 152 (Patent Owner's declarant stating that the cite is to an index directing the reader to pages 10–11, which merely discuss the Find command and not some new feature we have not already reviewed). We cannot review Petitioner's cite and we do not credit it. That said, even if one of the references said that the purpose of the filter or Find feature were to search for "text strings," in our view the natural reading of such a passage merely means that the purpose of the form is to find strings—i.e., to be used when desired to find a string.

Petitioner's third position, that it would have been obvious to incorporate True's teachings regarding two search strings, has two subparts. *See* Pet. 60. The first subpart is what True teaches, and the second subpart is

whether it would have been obvious to a person of ordinary skill in the art to modify Entourage, in some fashion, with True's teaching.

As to the first subpart, True teaches breaking up a single search string containing two words separated by a space, e.g., "queen mother," and searching for the separate words "queen" and "mother." Ex. 1009, 5:40–50 (describing "matching a word in the query"). Thus, True teaches a search where a string is separated into two strings by a string separator character. *See* Ex. 1098, 39:10–12 (Patent Owner's expert testifying that "there must, of necessity, be a way of distinguishing the two search strings from each other" when using a search field). However, True is not doing a search like Entourage does a search. In Entourage, the user is looking for documents or records that contain certain text or whose metadata contains certain text. *See, e.g.*, Ex. 1004, 8 ("Entourage has features that make it easy to find items that meet your criteria."); Ex. 1006, 384 ("After a short time—shorter than you might think—you'll collect a lot of email messages, contacts, and other Entourage items. Trying to find a particular morsel of information just by browsing becomes impractical."). In contrast, True is calculating an "interest level" which "represents the degree to which text in the window represent segments of interest in the digitized audio." Ex. 1009, 5:29–50. The reason True breaks up the string into separate words and searches for them independently to generate a score is because True does not trust the quality of the underlying text. *Id.* at 1:18–30 (stating, "the searchable information . . . may not always accurately represent the portions of the underlying information . . . that the user is trying to find"); *see also* Ex. 2020 ¶ 114. Thus, True is breaking up the words so that if one of the two words is garbled or uncertain, but the other is clear, there is an increased chance of

finding the phrase. *See* Ex. 1009, 5:41–51 (calculating the total score based on the sums of the individual words appearing in a given timeframe).

Although True is searching text, it is really searching for sounds, with text as the proxy for sounds. *Id.* at 1:19–22. In other words, True does not use its two search strings to locate documents containing text, but rather to locate portions of a document that *might* contain at least one of the spoken words in the search query. *See* PO Resp. 51–52 (citing Ex. 2001 ¶ 132; Ex. 2020 ¶¶ 115–116); Ex. 2020 ¶ 116 (Patent Owner's declarant stating that "[i]t is possible for a transcript window [in True] to be highlighted as relevant even though the visible transcript text does not show the query word"). As Patent Owner points out, several of the results of the "queen mother" query include no hits whatsoever. *Id.* (discussing Ex. 1009, Fig. 2 (reproduced *supra*), pointing to search result list 22). Thus, although True discloses a search that can be characterized as searching for two strings in some sense, True's search is very different from Entourage's search.

As to the second subpart, despite the marked difference between Entourage and what True is doing and how it is doing it, Petitioner asserts that it would have been obvious to use True's two search strings "to provide more targeted searching, as was common sense and well known in the art." Pet. 60 (citing Pet. 8–9; Ex. 1057, 27); *see also id.* at 56, 66 (similar). But we are not persuaded by a preponderance of the evidence that True provides more targeted searching or better search results when finding documents. True itself appears silent on the issue. Petitioner cites Exhibit 1057, but all that stands for is the general premise that doing a Boolean AND search in a Web search engine reduces the number of search results. Ex. 1057, 27. Petitioner does not allege that True uses a Boolean AND search of the kind

discussed in Exhibit 1057. Thus, the statement in Exhibit 1057 regarding reducing the number of search results is tied to a manner of performing Boolean searches, something Petitioner does not use to modify Entourage. In other words, Exhibit 1057 is of little relevance to the Entourage-True combination, and does not serve as a factual underpinning in support of Petitioner's reasoning. Petitioner also cites to several other references that discuss Boolean-style searches, but these generic background disclosures do not explain why a person of ordinary skill in the art would have considered it obvious to use one or more features of the probabilistic-based interest level search from True in a document-finding application like Entourage. *See* Pet. 8–9 (string citing over a dozen references with no pinpoint citations). Accordingly, there appears to be little reason, absent hindsight, to incorporate some aspect of a text-as-proxy-for-sounds system like True into Entourage's filter or Find feature to provide more targeted searching. *Accord* Ex. 2020 ¶ 120 ("Patent Owner's declarant stating that "a [person of ordinary skill in the art] working on the problem of searching large document collections would not look to [True,] a system designed for navigating imprecise transcriptions of single audio files, as the technical approaches and constraints are entirely different").

Patent Owner argues, and we agree, that another problem with Petitioner's combination is that True does not appear to actually provide more targeted or better searching. PO Resp. 59–60 ("True's search capabilities are inherently imprecise") (citing Ex. 1009, 1:18–27; Ex. 2001 ¶ 137); *see also* Ex. 1009, 4:31–37 (explaining that True uses a speech recognition system that calculates confidence levels based on a "% likelihood that [a word] corresponds to the portion of the digitized audio [at

a given time]"); Ex. 2020 ¶ 114 (Patent Owner's declarant stating that "the purpose of True is not to definitively locate where a search string actually appears"). Petitioner does not explain what about True would make some aspect of Entourage's filter or Find features more targeted. Even if it was known to use Boolean AND searches, for example, Petitioner is not citing True to modify Entourage to have a Boolean search. True uses an "interest level" search instead, which is more inclusive than a Boolean AND search (it adds together hits for "queen" OR "mother," and may return hits containing neither). Ex. 1009, 5:29–50, Fig. 2 (showing results list 22 having several results which do not indicate any of the searched-for words); *see* Ex. 2020 ¶ 116 (Patent Owner's declarant analyzing True's interest level search would also include "text [that] does not show the query word"). Petitioner does not appear to be proposing to incorporate an interest level search or a Boolean-style search into the filter or Find features, and we are not persuaded that True in fact provides more targeted searching. Even if the quality of the transcripts in True happened to be perfect (*see* Pet. Reply 10, arguing that "True describes searching a text file that has 100% accuracy"), the fact remains that True attempts to find text in particular time windows using an interest level search based on the probability that a word or words were spoken in a given time window, which is completely different from what Entourage is doing with its search (*see* Ex. 2020 ¶¶ 114–116, 121).

In its Reply, Petitioner states that True teaches "an incremental search for multiple search strings entered in a search field." Pet. Reply. 10. This is a broad generalization of what True does, made with the benefit of hindsight, but as we have just explained, True's interest level search is different from the kinds of searches done in Entourage or even the Boolean-style web

searches performed in the background references. *See, e.g.*, Ex. 1057, 27. This means it is not a matter of taking one known teaching and simply applying it to another system. True is doing something different, and it's not clear how True's search would actually provide "precise, targeted search results" (Pet. Reply 3) given that True appears to suggest a less-targeted search than claimed (*see, e.g.*, Ex. 1001, 29:62–63, claiming "identifying one or more documents included in both the first one or more documents and the second one or more documents"). *See also* Ex. 2001 ¶¶ 135 (Patent Owner's declarant stating that "Entourage and True are directed to completely different purposes"), 136–138 (further analysis); Ex. 2020 ¶ 114 (stating that "the purpose of True is not to definitively locate where a search string actually appears"), ¶ 116 (stating that True returns text that does not even have the search word). True does not teach, suggest, or support Petitioner's general premise that there is a benefit for an incremental search for multiple search strings entered in a search field or that such searches would provide precise, targeted search results.

The problem with Petitioner's analysis becomes more apparent when the Petition turns to the "string separator character" of claim 1. Notably, it is the claimed string separator character that converts a field with a single search string into a field with two search strings. *See, e.g.*, Ex. 1098, 37:12–41–4 (deposition of Patent Owner's declarant) (discussing that a string separator character is what instructs the computer to break up one string of characters into separate strings of characters). Thus, one cannot discuss the string separator character apart from the limitations requiring first and second search strings in a single search field. Nevertheless, Petitioner treats the "string separator character" in isolation. *See* Pet. 61–62. Petitioner's

reason for modifying one or both of Entourage's filter and/or Find feature(s) to have a string separator character is because "it was common to [use string separator characters]." *Id.* at 61. However, even if something was known, it does not mean it is obvious to incorporate it into one or both of these features in Entourage. *KSR*, 550 U.S. at 418 ("a patent composed of several elements is not proved obvious merely by demonstrating that each of its elements was, independently, known in the prior art").

Adding a string separator character to divide a single string into two separately searched strings changes how the software would function. *See* Ex. 1098, 39:22–40:8 (Patent Owner's declarant testifying that software has to look for a string separator character in order to know multiple search strings were entered in the field). The space character in "Orson Welles," used in the Advanced Find example in Barber, for example, is not a string separator character because one looking to exclude hits with "Orson Welles" is searching for the string "Orson Welles"—the space is another character in the name string whose inclusion serves a specific semantic purpose (delineating between a first and last name). *See* Ex. 2001 ¶ 151 ("there is no evidence that when a user types 'Orson Welles' into this Find command text box that a search is performed on 'Orson' and 'Welles' separately as opposed to searching on the full text string 'Orson Welles.'"); Ex. 1006, 387 (discussing how exclude the specific person "Orson Welles" from the search, not to find someone having the first name "Orson" or the last name "Welles"); *see also* Ex. 1003 ¶ 144 (Petitioner's declarant implicitly acknowledging that the space in "Orson Welles" is not a string separator character). Thus, the space character in "Orson Welles" does not signal to the computer to break the string up into two parts at the space and perform

separate searches for the strings on either side of the string separator and then join those search results together. Petitioner has not given a persuasive reason to modify Entourage's filter or Find function to have a string separator character to enable the first search field to have first and second strings that are separately searched. Petitioner proposes to modify a sophisticated piece of software to meaningfully change its functionality simply because it was allegedly known to use string separator characters in other contexts. This meager rationale is not commensurate with the differences between Entourage and the claimed subject matter. Accordingly, for this additional reason, we find that Petitioner's reason to combine Schwartz, Sebastian, Barber, and True does not have sufficient factual underpinnings to meet a preponderance of the evidence standard.

For the reasons set out above, Petitioner's obviousness ground based on Entourage fails.

2. Petitioner's Analysis of Elements 1.5–1.8

Claim 1 elements 1.5 to 1.8 recite the steps of performing an incremental search, wherein each of the two search strings are identified in their own set of documents, and then identifying documents that appear in both sets. Given that these steps are the implementation of a search using the search strings gathered in prior steps 1.2–1.4, Petitioner's analysis is largely similar to what we have already discussed and found lacking. *See* Pet. 63–72. Indeed, Petitioner's discussion of why it would have been obvious to modify Entourage in its analysis of elements 1.5–1.8 is "to provide more targeted searching," which is the same insufficient reason as discussed above. *Id.* at 66. Petitioner adds that it would have been obvious "to incrementally locate documents containing two strings (as opposed to

just one string) to provide the user with *more relevant results*" (*id.* at 69 (emphasis added)), but this presumes that it would have been obvious to have two search strings separated by a string separator character as claimed, which we have already determined Petitioner has not shown.

Even taking Petitioner's "more relevant results" rationale as explaining why a person of ordinary skill in the art would use a string separator character to separate a string in a single search field into two strings, and then to separately search those two strings, and also to do all of that incrementally, we still find Petitioner's explanation lacking. Even if we assumed that each of these elements were taught or suggested in isolation, it would remain unclear how the prior art is being applied to Entourage and what would be the justification for those modifications. Petitioner states that "incrementally locat[ing] documents containing two strings" "is suggested by the Entourage references," yet Petitioner does not cite to an example of Entourage incrementally locating documents containing two search strings. Pet. 69. Petitioner also alleges that "incrementally locat[ing] documents containing two strings" "was known to a [person of ordinary skill in the art]," yet the citations provided do not appear to teach incrementally locating documents containing two search strings. *Id.* at 69 (citing "Section IV.B.2" [Pet. 8–9]; Ex. 1057, 27; Ex. 1021, 125–127; Ex. 1003 ¶ 154). Exhibit 1057 is a book discussing web search engines, and page 27 does not talk about incrementally locating documents containing two search strings. Exhibit 1021 (Raskin) talks about incremental searches (e.g., Ex. 1021, 125) and even incremental Boolean searches (*id.* at 127), but it does not talk about incremental searches using two search strings in a search field separated by a

string separator character.[2] Even if it did, that would merely establish knowledge; it would not provide a reason to combine. Exhibit 1003 is Petitioner's expert declaration, which restates the language of the Petition. Ex. 1003 ¶ 154. Having reviewed the evidence cited by Petitioner, we agree with Patent Owner's declarant that Petitioner has not shown the type of incremental searching *claimed* was well known at the time of the invention. Ex. 2020 ¶¶ 18–25; Ex. 2001 ¶¶ 39–47.

As explained above, Entourage discloses incremental searching via the filter function but not the Find function, and can search within a selected document using the Find function but not the filter function. Patent Owner's complaint that Petitioner fails to explain which aspects of True are being used to modify one or both of Entourage's filter and/or Find functions is not requiring "bodily incorporation," as Petitioner frames it. PO Resp. 60–61; Pet. Reply 8–10. Part of an obviousness analysis is establishing the "differences between the prior art and the claims at issue." *Graham*, 383 U.S. at 18. From there, we can evaluate whether Petitioner has articulated "some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness." *In re Kahn*, 441 F.3d 977, 988 (Fed. Cir. 2006). Here, the problem with Petitioner's analysis is that it treats each of the differences between the prior art and the claims independently, such that Petitioner never cogently accounts for the differences between the prior art and the claims *as a whole*. *Stratoflex*, 713 F.2d at 1537.

---

[2] Raskin is discussed in more depth above in our analysis of ground 1 involving Lotus Notes.

For example, let us first assume Petitioner is proposing to modify the filter in Entourage to allow searching for two search terms, separated by a string separator character, resulting in two sets of documents that are then reviewed to identify and display documents appearing in both sets. Even assuming that True teaches a two-search-string incremental search using a string separator character, Petitioner still must establish a reason, with factual underpinnings in support, to modify Entourage's filter. Petitioner states the modification would "provide more targeted searching." Pet. 60, 66; *see also id.* at 56 (Petitioner using similar "more relevant results" language). The evidence Petitioner cites, however, is unavailing. Exhibit 1057 and its declarant merely parrots the same language as the Petition. Ex. 1003 ¶ 143. True does not itself suggest any particular "more targeted searching" benefit; True suggests the opposite. Ex. 2020 ¶ 119 (Patent Owner's declarant stating that "when [True] returns a search result, that does not necessarily even mean the term is in the search result"); Ex. 2001 ¶ 138 (stating that "True, by the nature of its invention, is imprecise"). Thus, there is an insufficient factual basis for Petitioner's rationale.

As an additional and separate reason, we do not find True's search to be more targeted because it is looking within a single document, not across a database of documents like in Entourage, and because True's search is looking for windows of time where a calculated interest level is met, not looking for the presence of first and second strings. Ex. 1009, 5:29–50; *see also* Ex. 2020 ¶ 116 (Patent Owner's declarant analyzing True's interest level search would also include "text [that] does not show the query word"); Ex. 2001 ¶ 138 (stating, "it would not have made any sense to turn to True's

inherently imprecise interest level method"). Put another way, Petitioner's reason for combination is contrary to what True actually teaches.

Now let us assume that Petitioner is instead proposing to modify the Find feature in Entourage to allow incremental searching for two search terms, separated by a string separator character, resulting in two sets of documents that are then reviewed to identify and display documents appearing in both sets. In addition to the problems we identified with respect to the filter function, the Find function has additional challenges. An incremental search displays results as they are typed, but the Find feature builds a query in a pop up window and then displays the results in a separate window. How would the results be displayed? Would the separate window pop up as soon as the user starts typing into the Find pop up window? Where would it go? It would seemingly obstruct the Find window. *See* Ex. 2001 ¶¶ 139–140 (Patent Owner's declarant stating that Petitioner's position "does not account for or otherwise acknowledge the obstacles that need to be overcome for the combination to properly function" and that "[g]iven this separate window and the use of a delimited search that requires an explicit press of a button to initiate the search, a POSITA would not contemplate making this search incremental"). This can be neatly contrasted with Entourage's filter function, which does allow incremental searching and where there are no windows involved. Ex. 1004, 9 (noting Figure 1.12 – the filter components are in-line with the address book and the results); Ex. 2020 ¶ 109 (Patent Owner's declarant opining that, "when a product maintains two separate search mechanisms rather than combining them, it typically reflects technical constraints or usability considerations that make combination undesirable or impractical"). Petitioner's proposed

combination meaningfully changes the way a user would interact with the Find feature in Entourage, yet Petitioner provides no discussion of how the Entourage features would be modified to accommodate the features from True, much less why one of ordinary skill in the art would make these specific modifications. Without knowing the particular modification, we cannot determine that it would have been obvious to so modify the teachings or determine whether there would have been a reasonable expectation of success. *See* PO Resp. 62 (alleging Petitioner's proposed combination would not have a reasonable expectation of success); Ex. 2001 ¶ 139 (Patent Owner's declarant opining that Petitioner's declarant "does not account for or otherwise acknowledge the obstacles that need to be overcome for the combination to properly function"); Ex. 2020 ¶ 123 (asserting that modifying the Find function to have incremental searching would impose significant technical hurdles).

To be clear, we are not trying to discern the specific coding modifications required to modify the Find function to be incremental. *Cf. In re Mouttet*, 686 F.3d 1322, 1332 (Fed. Cir. 2012) ("It is well-established that a determination of obviousness based on teachings from multiple references does not require an actual, physical substitution of elements."). Even at a conceptual level, using Entourage's specific implementations as an example: because the prior art shows different ways to perform searches (e.g., filter vs. Find), it shows that due consideration has to be given to the way the user interacts with the relevant functionality and the technical implications of incremental search. *See* Ex. 2001 ¶¶ 140–141 (Patent Owner's declarant stating that the way the Find function is presented in Entourage would not be amenable to incremental search because of the interface as well as the time it

takes to do a Find); Ex. 2020 ¶ 123 (Patent Owner's declarant stating that a person of ordinary skill "would not expect incremental search in the Find or Advanced Find features of the Entourage references to yield a successful, workable search system due to speed and efficiency concerns"). Petitioner has not done so.

In sum, Petitioner has not identified sufficient factual underpinnings in support of a proposed modification to Entourage to include incremental search features and two-search-string search in the same field, delimited by a string separator character. Thus, we are not persuaded by a preponderance of the evidence that Petitioner has shown claim 1 to have been obvious in view of Schwartz, Sebastian, Barber, True, and Baeza-Yates.

Petitioner's arguments regarding claims 2–27 repeat or rely on its analysis of claim 1, and are rejected for the same reasons. *See* Pet. 73–93.

III. CONCLUSION

For the reasons given above, we determine that Petitioner has not established by a preponderance of the evidence that any claim of the '139 patent would have been obvious in view of the prior art cited.

The following table summarizes the outcome of this proceeding:

Claim(s)	35 U.S.C. §	Reference(s)/Basis	Claim(s) Shown Unpatentable	Claim(s) Not Shown Unpatentable
1–27	103	Lotus Notes (Wilcox, Londergan), Raskin, Wu		1–27
1–27	103	Entourage (Schwartz, Sebastian, Barber), True, Baeza-Yates		1–27

IV. ORDER

In consideration of the foregoing, it is hereby

ORDERED that the record show that Petitioner has not established that any of claims 1–27 of the ’139 patent are unpatentable;

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.

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