

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICROSOFT CORPORATION.,
Petitioner,
v.

X1 Discovery, Inc.,
Patent Owner

U.S. Patent Nos.
9,633,139

Case No. IPR2025-00255

DECLARATION OF MARTIN P. HAEBERLI.

TABLE OF CONTENTS

I.	Introduction.....	1
II.	Summary of Opinions.....	2
III.	Materials Relied Upon.....	2
IV.	Background and Qualifications	3
V.	Legal Standards	7
A.	Obviousness.....	7
B.	Claim Construction.....	9
VI.	Person of Ordinary Skill in the Art (“POSITA”)	9
VII.	Overview of the ’139 Patent.....	11
VIII.	Claim Construction.....	13
IX.	Background of the Technology	14
X.	Overview of the Asserted Art.....	19
A.	Wilcox (EX1059)	19
B.	Londergan (EX1061).....	21
C.	Raskin (EX1021).....	23
D.	Wu (EX1019)	25
E.	Entourage (Schwartz (EX1004), Sebastian (EX1005), and Barber (EX1006)).....	27
F.	True.....	30
G.	Baeza-Yates	33
XI.	GROUND 1: The challenged claims are not obvious over wilcox (Ex1059), Londergan (EX1061), Raskin (ex1021), and wu (ex1019).....	34
A.	A POSITA Would Not Have Been Motivated To Combine Wilcox And Londergan Because Lotus Notes 3.0 And Lotus Notes R5 Have Distinct Features And Are Not The Same System.	34
1.	Query Builder (v. 3.0) vs. Search Builder (R5).	36
2.	Search Bar	39

3.	Wilcox (Lotus Notes v. 3.0, EX1059) and Londergan (Lotus Notes R5, EX1061) Are Improperly Combined.....	39
4.	Dr. Terveen’s Proposed Wilcox-Londergan Combination Represents Improper Hindsight Bias.	43
B.	A POSITA Would Not Have Been Motivated To Combine Wilcox-Londergan With Raskin	44
C.	A POSITA Would Not Have Been Motivated To Combine Wilcox-Londergan With Wu.....	50
D.	The Prior Art Does Not Disclose All Limitations Of The ’139 Patent’s Claims.....	55
1.	The Combination Of Wilcox, Londergan, Raskin, And Wu Does Not Teach The First And Second Partial Search Strings Being Separated By A String Separator Character In The First Search Field (Limitations [1.4], [23.5], and [27.4]).....	55
2.	The Combination Of Wilcox, Londergan, Raskin, And Wu Does Not Teach Incremental Searching A Set Of Documents Using Two Partial Strings (Limitations [1.5]-[1.8], [23.6]-[23.9], and [27.5]-[27.8])	60
XII.	GROUND 2: The challenged claims are not obvious IN VIEW of Schwartz (EX1004), Sebastian (ex1005), barber (ex1006), True (EX1009), and Baeza-yates (ex1023).....	63
A.	Dr. Terveen Mischaracterizes the Search in Schwartz, Sebastian, and Barber.	63
1.	The Entourage References Describe an Email System.....	64
2.	Entourage’s List Filtering Function.	64
3.	Entourage’s Find and Advanced Find Commands.	65
4.	Dr. Terveen Mischaracterizes Entourage’s Search Functionality.....	68
B.	A POSITA Would Not Have Been Motivated to Combine Entourage with True.....	71
1.	True Does Not Teach Incrementally Searching a Set of Documents for Multiple Search Strings.	72

2.	True Is Directed To A Wholly Different Purpose Than Entourage And Its “Interest Level” Method Is Inherently Imprecise.	76
3.	Dr. Terveen Fails To Explain How A POSITA Would Modify Any Particular Search Tool In Entourage To Include True’s Disclosure.	79
C.	A POSITA Would Not Have Looked to Baeza-Yates to Improve Entourage.	81
D.	The Prior Art Does Not Disclose All Limitations Of The ’139 Patent’s Claims.	84
1.	The Combination Of Entourage, True, And Baeza-Yates Does Not Teach Incrementally Searching A Set Of Documents (Limitations 1.5, 23.6, and 27.5).	85
2.	The Combination Of Entourage, True, And Baeza-Yates Does Not Disclose Searching On Multiple Partial Strings.	86
3.	The Ground 2 References in Combination Do Not Teach Two Partial Search Strings Separated By A String Separator Character In The Same Search Field (Limitations [1.4], [23.5], and [27.4])	91
4.	The Combination Of Entourage, True, And Baeza-Yates Does Not Teach Highlighting A Plurality Of Occurrences Of The First And Second Partial Search Strings In The Selected Document (Claims 5 and 26)	92
XIII.	Conclusion	93

I. INTRODUCTION

1. I have been retained by counsel for Patent Owner X1 Discovery, Inc. (“X1”) in the above-captioned proceedings. I understand that Petitioner Microsoft Corporation (“Petitioner” or “Microsoft”) seeks *Inter Partes* Review (“IPR”) of U.S. Patent No. 9,633,139 (the “’139 Patent”). I submit this declaration in support of Patent Owner’s Preliminary Response for the ’139 Patent and in response to the opinions of Dr. Loren G. Terveen provided in EX1003.

2. Microsoft’s Petition for the ’139 Patent challenges claims 1-27. As explained in detail below, it is my opinion that Dr. Terveen fails to show that any challenged claim of the ’139 Patent is invalid.

3. I note that, in connection with my prior art analysis, a failure to demonstrate that even one claim limitation is disclosed in the prior art has the result that every claim incorporating that limitation has not been shown to be invalid. For each challenged claim of the ’139 Patent, I have identified at least one limitation that is missing from the prior art cited in Microsoft’s Petition. I note, however, that my election not to address other limitations of the claims herein is not intended as an admission that those other limitations are found in the prior art.

4. In connection with providing my opinions, I have also been asked to discuss the technology addressed by the ’139 Patent and the relevant state of the technical art at the time of the claimed invention.

5. I am being compensated for my time at \$580 per hour.

My compensation has not, and will not, affect my opinions in this matter, and my compensation is not dependent on the content of my opinions. I have no financial interest in the outcome of this proceeding, and my compensation will be unaffected by the outcome of this proceeding.

II. SUMMARY OF OPINIONS

6. None of the '139 Patent's challenged claims would have been obvious in light of the prior art cited by Petitioner, or combination of prior art, or under the knowledge of a person of ordinary skill in the art because each asserted prior art reference lacks at least one element of the asserted claims and because a person of ordinary skill in the art ("POSITA") would not have been motivated to modify or combine the prior art at the time of the invention in the ways asserted by Petitioner.

III. MATERIALS RELIED UPON

7. In forming my opinions expressed in this Declaration, I relied upon my education and experience, which I describe below and in my curriculum vitae (EX2002). In addition to the materials listed in the Exhibit List attached to this document, I reviewed and considered the following:

- a. Petition for Inter Partes Review of U.S. Patent No. 9,633,139
- b. Declaration of Loren G., Terveen, Ph.D.

IV. BACKGROUND AND QUALIFICATIONS

8. My qualifications for forming the opinions in this report are briefly summarized here and explained in more detail in my *curriculum vitae*, which has been attached as EX2002.

9. I have more than 50 years of experience with computer software, systems, hardware, networking, and communications.

10. Starting in 1971, I studied computer science and electrical engineering at the University of Wisconsin, Madison. At the same time, I worked at the University as a systems programmer, first for the Nuclear Physics department, and later for the Psychology department. In that capacity, I developed and maintained operating system software, wrote software to allow computers to communicate with the campus mainframe, and developed software for scientific applications, such as acquisition and analysis of data from nuclear physics experiments.

11. In June 1974, I moved to Holland, where I worked for about two years as the chief software and hardware support person for Techmation N.V., a small Dutch distributor of Harris/Datacraft US-made 24-bit scientific computers. In that role, I developed and enhanced operating systems and communications software, and diagnosed, repaired, and maintained computers that had been sold to our customers.

12. In March 1976, I returned to the United States and worked for approximately three years for Bolt, Beranek, and Newman Inc., where I was in charge of software for an advanced multiprocessor packet switch and terminal handling system called the Pluribus Terminal Interface Processor (“PTIP”). I also participated in developing software for Pluribus Interface Message Processors (“IMPs”), which were early packet routers for the ARPAnet. I worked on Internet communication protocols, including the IP portion of the TCP/IP protocol, as early as 1977, when I wrote and implemented the only IP version 2, version 3, and version 4 combined router that ever existed, which required a packet-by-packet heuristic approach to determining what IP protocol version was being routed.

13. I began working at Xerox Advanced Systems Development in Palo Alto, California in December 1978. There, I developed systems and applications software related to early word processors, and also worked with early user trials to help facilitate adoption and to help Xerox learn from those trials. In January 1981, I transferred from Advanced Systems Development to the Xerox Palo Alto Research Center (“PARC”) to work on chip design and manufacture. At PARC, I also worked under the guidance of Richard Lyon on enhancing the original prototype single-chip PARC optical mouse design, originally invented by Mr. Lyon.

14. In March 1982, I was recruited by Steve Jobs to join the Macintosh team at Apple, where I worked on custom chip design for the Macintosh. I led efforts to design a single large-scale “glue” chip for the Macintosh, working with Brian Howard and with Burrell Smith, who headed the Macintosh digital design effort. The first silicon for this chip was received in August 1982, but the Macintosh project had by then switched to a discrete digital design. I continued to lead VLSI (very-large-scale integration) design for Apple while contributing to the Macintosh project in other ways. For example, I was responsible for the memory allocator within the Macintosh operating system, and I co-authored MacTerminal, the first commercial VT-100 terminal emulator for the Mac, with Mike Boich. I remained at Apple through 1995, working variously on technical product marketing aimed largely toward college and university customers, and in Apple’s Advanced Technology Group (“ATG”). My work with ATG related to the internet and education. I also conceived, developed, built, and operated Apple’s first e-commerce online software store.

15. At Apple Computer, one of my roles was “Internet Wizard,” where I helped evangelize the early web-based internet, and I also developed and operated Apple’s first internet commerce site, which was used for Apple’s first electronic sales of software, in 1994 and 1995.

16. In January 1996, I joined Netscape as Director of Technology, reporting to then-Chief Technology Officer Marc Andreessen. I later ran Netscape's education marketing efforts as Director of Education Marketing. At Netscape I worked on internet security, internet commerce, internet privacy, and other internet standards on behalf of Netscape. In that role I also further developed my relationship with the Bootstrap Institute, now known as the Douglas Engelbart Institute, a non-profit that supports the work of (the now late) Douglas Engelbart, a widely-acknowledged hypertext pioneer. In 1999, I co-founded and served as Chief Technology Officer of OverSea Systems, an internet-based startup aimed at dramatically reducing the paperwork and administrative costs associated with maintaining and repairing intermodal shipping containers.

17. I left OverSea Systems at the end of 1999 and began consulting in various areas, including performing technical due diligence for venture capital investors, serving as a testifying or consulting expert in patent litigation, and advising attorneys on technology issues relating to patent prosecution and trade secret matters.

18. I also studied Concepts of Relational Database Management Systems at UC Berkeley Extension in Berkeley, California. This was a survey course in relational database management systems and in SQL.

19. I have continued to work as a consulting and testifying expert through the present day. In relation to such work, I have conducted well more than two dozen source code reviews in a wide variety of cases, including patent, copyright, and trade secret cases, covering technologies including a broad area ranging across, among others, mobile wireless data synchronization, embedded systems in set-top boxes and cable modems, storage area networks, mobile phone technologies, wireless email, and web-based language translation systems. I have also conducted research into publications and products alleged to be prior art to various patents.

V. LEGAL STANDARDS

20. I am not an attorney and offer no opinions on the law. I am, however, informed on several legal principles, summarized below, which I have used in reaching my opinions.

A. Obviousness

21. I understand that a patent is invalid if it is established that the differences between the subject matter of the patent and the prior art are such that the subject matter, as a whole, would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. I also understand that the question of obviousness requires consideration of (a) the scope and content of the prior art; (b) the level of ordinary skill in the art;

(c) the differences between the claimed invention and the prior art; and (d) secondary considerations of non-obviousness. I also understand that the scope and content of the prior art must be viewed through the perspective of a person of ordinary skill in the art at the time of the invention.

22. Because the test for obviousness is what would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains, I understand that analyses that rely on hindsight to establish obviousness are improper. I understand that the best way to avoid hindsight reasoning is to demonstrate that a person having ordinary skill in the art would have been motivated to modify the prior art to meet the claimed invention.

23. I understand that in forming an opinion regarding obviousness, one must consider so-called “secondary considerations” or “objective indicia” of non-obviousness, which include, for example:

- a. Whether the invention was commercially successful as a result of the merits of the claimed invention (rather than the result of design needs or market-pressure advertising or similar activities);
- b. Whether the invention satisfied a long-felt need;
- c. Whether others had tried and failed to make the invention;
- d. Whether others invented the invention at roughly the same time;

- e. Whether others copied the invention;
- f. Whether there were changes or related technologies or market needs contemporaneous with the invention;
- g. Whether the invention achieved unexpected results;
- h. Whether others in the field praised the invention;
- i. Whether persons having ordinary skill in the art of the invention expressed surprise or disbelief regarding the invention;
- j. Whether others sought or obtained rights to the patent from the patent holder; and
- k. Whether the inventor proceeded contrary to accepted wisdom in the field.

24. I understand that any secondary considerations must have some relationship or nexus to the invention claimed in the patents at issue.

B. Claim Construction

25. I understand that claim terms are generally given their ordinary and customary meaning to a person of ordinary skill in the art (“POSITA”) at the time of the invention.

VI. PERSON OF ORDINARY SKILL IN THE ART (“POSITA”)

26. I understand that the ’139 Patent is to be viewed from the perspective of one of ordinary skill in the art. I understand that the perspective of one of

ordinary skill in the art means the level of skill of an ordinarily skilled person in the art in question at the time of the claimed invention.

27. I have been asked to assume that the relevant date for conducting my analysis is September 3, 2002.

28. I understand that factors such as the education level of those working in the field, the sophistication of the technology, the types of problems encountered in the art, the prior art solutions to those problems, and the speed at which innovations are made inform the relevant level of skill in the art for a POSITA.

29. In my opinion, the POSITA at the time of the alleged invention of the Asserted Patents would have had (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 Dr. Terveen has provided its own definition of a POSITA as a person that would have had the following qualifications: 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. Pet. at 6.

More education, e.g., a Master's or Ph.D., would compensate for less work experience and vice versa. *Id.*

31. I believe Dr. Terveen'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.

32. Based on my qualifications outlined above, I qualify as at least a POSITA under both parties' definitions, as of the effective filing dates for the Challenged Claims and today.

VII. OVERVIEW OF THE '139 PATENT

33. The '139 Patent, which is titled "Methods and systems for search indexing," issued from U.S. Application Ser. No. 14/507,519 filed on Oct. 6, 2014.

34. The '139 Patent generally relates to "data processing, and in particular to systems and methods for locating data." (Ex. 1001, 1:35-36.) As the '139 Patent explains, conventional searching methods of the time "require the user to type in a search term, click on a search button, and review the results." (Ex. 1001, 1:37-2:11.) If the search did not return the desired results, "the user needs to edit the search terms, click on an initiate search button, and again review the results"—making the process "slow and cumbersome." (Ex. 1001, 1:37-2:11.)

35. The '139 Patent resolved these inefficiencies by allowing users to incrementally search multiple strings using just a single search field. (Ex. 1001, 3:43-54.) As the '139 Patent explains, the results for incremental searches are “narrowed immediately after each character in a search string is entered by a user.” (Ex. 1001, 2:20-22.) Accordingly, unlike a conventional delimited search, which requires the user to execute a search by, for example, pressing a search button or hitting enter before displaying the results, the results of an incremental search display as the user types out the string. (Ex. 1001, 1:38-48, 2:15-27.) By searching incrementally, the “user is provided with substantially immediate feedback as the search string is being entered, and so [they] can quickly decide on the desirability of entering additional search characters, or of entering a new search string, or deleting one or more search string characters.” (Ex. 1001, 2:22-27.)

36. Further improving upon the efficiency of incremental searching, the primary embodiment of the '139 Patent allows the user to incrementally search multiple strings using a single search field. (Ex. 1001, 2:36-42.) As such, the user can enter two strings into a single “search field; and incrementally locat[e] at least a first document that has a least a first word that begins with the first string and a second word that begins with the second string.” (Ex. 1001, 2:36-42.) Thus, the search results immediately narrow for each portion of the string the user enters

into the searching field. (Ex. 1001, 2:36-42, 16:26-45.) Figure 3G of the '139 Patent, showing an embodiment of the disclosed searching method, is included below:



(Ex. 1001, Fig. 3G.)

VIII. CLAIM CONSTRUCTION

37. For all terms, I have applied the plain and ordinary meaning of the term as it would be commonly understood by a POSITA at the time of the invention.

IX. BACKGROUND OF THE TECHNOLOGY

38. Before the '139 Patent's invention, the general availability of information and documents had been steadily rising and the amount was expected to grow exponentially. (*See* EX1011, 1:23-26; EX1013, 1:14-20.) There was an ever growing need for efficient searching and accessing the unprecedented amount of stored information and documentation. (EX1023, 2-3.) Conventional search application programs at the time required users to type in a search term, click on a search button, and review the results, leading to slow and cumbersome searches. (EX1001, 1:38-41.) In instances where the user was not satisfied with the results, the user would need to edit the search terms and go through the process again. (EX1001, 1:41-45.) Users needed a unified interface and application that allowed effective searches on various types of data and allowed searching on multiple criteria.

39. In the lead up to the invention of the '139 Patent, a variety of methods and systems were proposed to fill this need. In his "State of the Art" section, Dr. Terveen describes such attempts, including attempts for implementing alleged incremental search. These attempts all fell short of the '139 Patent's method using a unified search that allowed for incremental searching a set of documents over multiple search terms. Dr. Terveen alleges this incremental searching claimed in the '139 Patent was "well known long before the patent's alleged 2002 priority

date” and provides a long list of citations in support. (EX1003, ¶¶33-34.) Yet, none of the prior art Dr. Terveen lists nor the asserted prior art disclose the ’139 Patent’s incremental searching a set of documents over two strings in the same search field. Dr. Terveen’s list and his discussion of incremental search in the prior art is a litany of mischaracterizations.

40. First, Dr. Terveen cites Entourage as disclosing incremental search. (EX1003, ¶126.) As I discuss in more detail below, Dr. Terveen relies on the filtering feature in Entourage as being incremental search ,but all the filtering examples in the Entourage prior art references Dr. Terveen cites primarily describe themselves as filtering lists like the address book and do not use two search strings. (*See, e.g.*, EX1004, 8-9.) While Entourage does have the Find and Advanced Find features that allow for more customization, as Dr. Terveen admitted, these tools only perform delimited searches which led to the slow, cumbersome searches the ’139 Patent looked to solve. (EX1003, ¶127.)

41. Dr. Terveen then cites True, a reference that was disclosed to the examiner during prosecution of the ’139 Patent, as an example of incremental search even though it also does not contain incremental search of documents over two strings in the same search field. Indeed, it is arguable whether True is performing a search at all, and True does not search a set of documents and return the documents as results. Instead, True searches a singular text file that is created

by a speech recognizer receiving an audio file. (See EX1009, 3:34-37, 4:5-20.) Additionally, True only provides results in the form of transcript windows that *may* contain the strings in the query and a “confidence level” that indicates “the likelihood that the recognized word is correct.” (See EX1009, 3:37-49, 4:34-38.) None of the disclosure in True enables incremental searching over two strings. Dr. Terveen also cites U.S. Patent No. 5,444,823 to Nguyen (EX1010) as evidence of incremental search even though this patent does not provide results as the user types but only in two circumstances: (1) when the search string includes a keyword detected by the search engine that matches a list of symptoms or if three characters are typed in so it can perform a “trigram” search. (EX1010, 7:34-53.)

42. Dr. Terveen’s laundry list of citations regarding incremental search includes U.S. Patent No. 7,039,635 to Morgan (EX1011). However, Dr. Terveen’s characterization of Morgan is misleading because Morgan only looks at whether a query aligns with pre-loaded “search strategies” that provide suggestions for potential searches and “quick matches” and only takes one string in its search field. (EX1011, Abstract, Fig. 7B, 8.) Dr. Terveen also suggests EX1012, U.S. Patent Pub. 2002/0055981 to Spaey, and a cite to EX1013 (U.S. Patent No. 5,845,278 to Kirsch) support his assertion that the ’139 Patent’s incremental search was well known. (EX1003, ¶33.) However, EX1012 and EX1013 were examined extensively in the prosecution of the ’139 Patent and the examiner

allowed the patent over the art and found that the challenged claims were non-obvious because the references only disclosed receiving search terms in separate field (similar to Lotus Notes R5's "Search Builder"). (*See* EX1002 at 195-96.)

43. Dr. Terveen's mischaracterizations of the state of the art at the time of the '139 Patent's invention continue with his discussion of U.S. Patent No. 4,255,796 to Gabbe (EX1014), which does not provide any actual incremental search results, and instead only discloses "the user is fed back information on the number of possible matches" as the user types. (EX1014, Abstract.) Again, there is no discussion of incremental searching on multiple strings. Similarly, Dr. Terveen discusses U.S. Patent No. 5,692,173 to Chew (EX1015) as describing "the incremental search method" even though that reference is directed specifically towards searching lists and not multiple documents. (EX1015, Abstract.)

44. Dr. Terveen's reliance on U.S. Patent No. 6,049,796 to Siitonen (EX1016), is similarly lacking when compared to the '139 Patent because it is merely a search method for searching/filtering a contact list on a PDA and does not actually relate to searching documents. (EX1016, Abstract.) Dr. Terveen then cites to U.S. Patent No. 5,303,361 to Colwell (EX1017) but this reference too only discloses incremental search on a single string and a secondary search after the user has opened a data file. (*See* EX1017, 10:1-22.) There is again no incremental search on two strings.

45. Dr. Terveen then cites to U.S. Patent No. 6,741,985 to Green (EX1018) as yet another example that incremental search was well known. (EX1003, ¶33.) This patent too does not disclose incremental searching over multiple strings and instead discloses a method of searching using a word look-up table that breaks words into prefixes. (EX1018, Abstract, Fig. 8.) Searching on these prefixes does not constitute incremental search, let alone incremental search over two strings. Dr. Terveen then alleges that Raskin (EX1021) and EMACS (EX1022) disclose incremental search as disclosed by the '139 Patent even though Raskin and EMACS only discuss incremental search using a single string through an individual unformatted text file. (EX1021, 125-26; EX1022, 55-57.)

46. Dr. Terveen's cite to U.S. Patent No. 6,952,805 to Tafoya (EX1029) is similarly unconvincing as it is only directed to automatically completing search entries when searching a list of contacts/email addresses and does not describe searching actual documents. (EX1029, Abstract.) Dr. Terveen also cites to EX1039, a book about Eudora 5, an email system. This reference too fails to show incremental search using two strings that returns documents and instead allows for filtering of emails, like Entourage. (EX1039, 103.)

47. Finally, Dr. Terveen cites to another Entourage book (EX1043) that runs into the same issues of not disclosing incremental search and instead disclosing filtering functionality, and EX1044, an article about text searches that

discuss dynamic queries. The dynamic query discussed in the article is not an incremental search on two strings and instead allows users to have a results screen that stays up as the “phases of the search are changed.” (EX1044, 3.) This is still a far cry from incremental searching documents using multiple strings in a single search field.

X. OVERVIEW OF THE ASSERTED ART

48. The Petition and the Terveen Declaration cite a number of references as alleged prior art to the Asserted Patents. I address each of those references below.

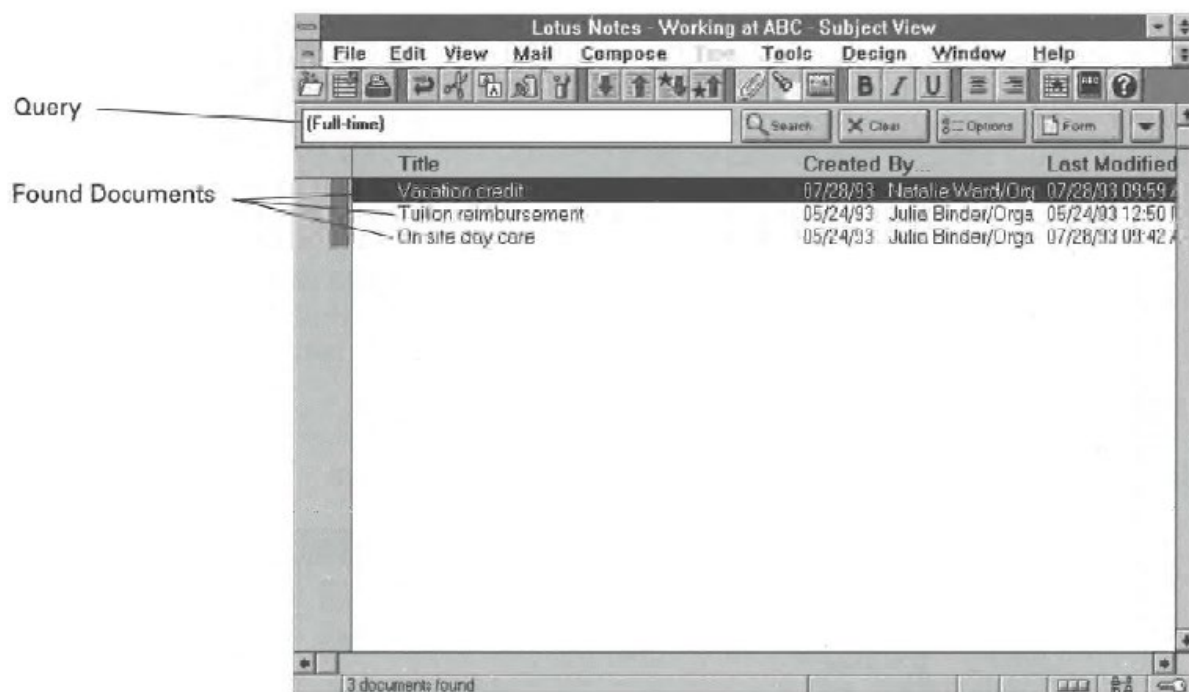
A. Wilcox (EX1059)

49. PC Learning Labs Teaches Lotus 3.0 (“Wilcox”) is an educational book, published in 1993, designed to teach readers how to use Lotus Notes 3.0.

50. Wilcox is designed to be a quick guide to users to attain a high level of fluency in Lotus Notes 3.0. (EX1059, xvi.) Wilcox describes Lotus Notes 3.0 as a “groupware program that manages information for many users on a computer network” and allows colleagues to communicate and share information using their personal computers. (*Id.*, 2.) Lotus Notes 3.0 is a system with a server and several workstations connected to it and all the connected workstations can share information and documents by storing them in a central location. (*Id.*, 3-4.) The

primary goal for Lotus Notes 3.0 is encouraging collaboration and cooperation between colleagues/group members. (*Id.*, 5.)

51. Wilcox discusses some of the search capabilities of Lotus Notes 3.0, including a Search Bar that can receive a string that is used to search for documents once the user clicks “Search:”

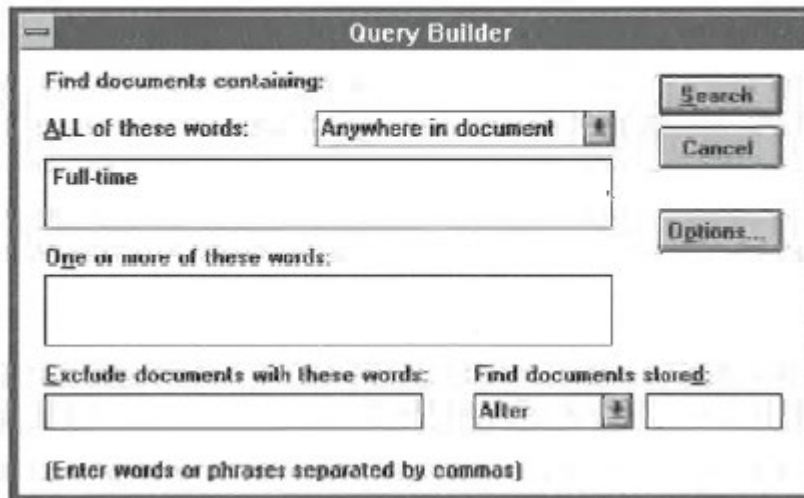


(EX1059, 164.) Users can open the document to see the text they were looking for. (*Id.*)

52. Another search capability in Lotus 3.0 is the Query Builder. (*Id.*, 160-65.) This tool is a separate window with an interface that has multiple text boxes for users to enter words as text. (*Id.*, 163.) The Query Builder allows for Boolean

searches using operators like AND, OR, NOT, and ACCRUE and provides options to tailor the search. (*Id.*, 162-67.)

Using the Query Builder dialog box



(*Id.*, 163.) Any searches are initiated once the user clicks the “Search” button.

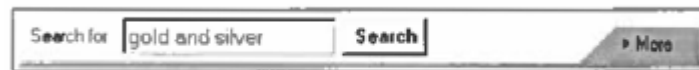
B. Londergan (EX1061)

53. Lotus Notes R5 for Dummies (“Londergan”) is a quick reference book published in 1999. This book was intended to be a quick reference book that helps users to perform actions in Lotus Notes R5 (a newer version of Lotus Notes than Lotus Notes 3.0 discussed by Wilcox) that they cannot remember how to do or are doing for the first time. (EX1061, 1.)

54. Londergan describes Lotus Notes R5 as a “computer program that enables you to communicate and interact electronically with other people.” (*Id.*, 5.) It is a groupware program that helps users: send e-mail documents, create databases and put information into them that the user and others can read and edit,

enable collaboration, and ensure information management. (*Id.*, 6.) The entire program is organized around databases as exemplified by a “discussion database” which allows for users to compose and store a main document that other users can read and compose their own document to reply to this main document, creating a discussion. (*Id.*, 7.)

55. Londergan discusses some search functionality provided by Lotus Notes R5, including the Search Bar, which can take text and perform a full-text word search using the entered text as the query:



(EX1061, 140.) This Search Bar will only search documents in the current view being used on Lotus Notes R5 and can take inputs such as a single word, a phrase in quotes, and words connected by Boolean operators like AND. (*Id.*, 134, 140.)

56. Lotus Notes R5 also has a more complex search option with its Search Builder functionality. Search Builder is a dialogue box that presents users with various options to refine their searches including enabling conditions and providing multiple, individual text boxes to run searches on different single strings. (*Id.*, 141-42.)



These multiple text boxes allow users to enter one word or phrase in each box and then click search to return results. The use of words and phrases here suggests that this search builder does not perform incremental search as a button press is required.

C. Raskin (EX1021)

57. The Humane Interface: New Directions for Designing Interactive Systems (“Raskin”) is a textbook that was published in 2000.

58. Raskin generally relates to creating the ideal user interface for an operating system. (EX1021, xi-xii.) As explained by Raskin, EMACS is a text editor (a program for editing unformatted text data), and any such incremental searching performed in EMACS relates only to searching a single document, and not an entire file system or even a collection of multiple documents. When explaining the aspects of the ideal searching system in such a user interface, Raskin discusses the incremental searching system implemented in EMACS—a

text editor. (EX1021, 125-27.) As Raskin describes, “in most implementations of incremental searches, as with the delimited search, the user first summons a dialog box that contains a field in which the user can enter the pattern.” (EX1021, 125.) In EMACs, this is done using a keyboard shortcut (e.g., ctrl+s or ctrl+r). (*Id.*) If 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.” (EX1021, 125.) If the user enters another character before the first instance is found, “the search continues, now looking for an instance of the now extended [string].” (EX1021, 125.) This method is repeated “as each character is added to the [string].” (EX1021, 125.) A POSITA would understand the EMACS searching tool as a method of searching for a term *within* a document—not a method of searching among documents for a specific document containing the term. (*See* EX1021, 124-30.) A POSITA would have known that searching a file system poses different constraints and problems that search in a single document does not have to consider. As an example of differences between searching a single document and searching a collection of documents in a file system, it is important to remember that performance and responsiveness to user interactions govern what is practical. For single documents, it is usually reasonable to store the whole document in memory, providing for very rapid responses to search requests. For

collections of documents in a file system, it becomes important to build indexes and other mechanisms to speed up the identification of potential candidate matching documents, which will ultimately result in documents matching the search criteria being speedily identified, and search matches then quickly being displayed.

59. In another section of the book, Raskin discusses a document storage system that is devoid of filenames, folders, or any other sort of “hierarchical file structure.” (EX1021, 117-21.) Raskin argues that the use of filenames requires a user to “remember [a] file by that one tiny portion and nothing else.” (EX1021, 118.) Without filenames, Raskin envisions a “whole text search,” where users can find documents using “any word or phrase” from the document, as opposed to just its name. (EX1021, 119-20.) Following the same logic, Raskin argues for the removal of folders and any other “hierarchical file structure,” leaving the user interface completely unstructured. (EX1021, 120-21.) Raskin does not discuss any method of incrementally searching for documents. (*See generally* EX1021.)

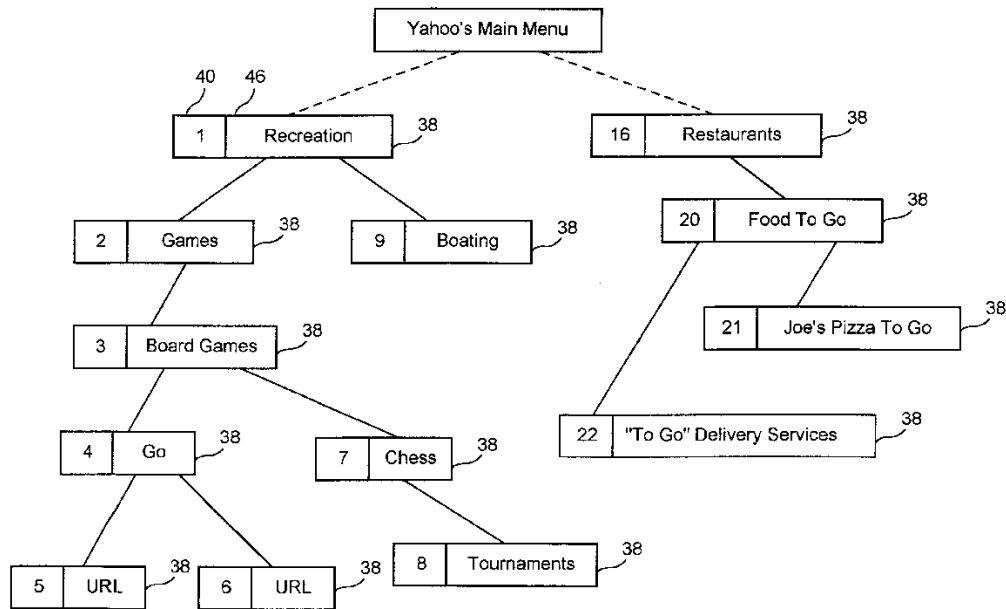
D. Wu (EX1019)

60. U.S. Patent No. 5,991,756 (“Wu”) was filed on November 3, 1997, and issued on November 23, 1999.

61. Wu generally relates to “the field of electronic document storage and management.” (EX1019, 1:15-16.) Wu describes applying search strings to

websites “organized into a hierarchy.” (EX1019, abstract.) The hierarchy defined by Wu is made up of category nodes, site nodes and sub-category nodes. (EX1019, 3:50-65.) The site nodes and sub-category nodes are given an “importance weighting value” which indicates how valuable and relevant the website is in relation to other websites based on external events or the significance of the website as described by the node. (EX1019, 6:30-40.)

62. The primary embodiment disclosed by Wu relates to searching for websites on the internet using the “Yahoo! Search database.” (EX1019, 3:50-54.) In this embodiment, websites “are placed in appropriate locations in the hierarchy by an editorial staff using the experience and suggestions from site submitters.” (EX1019, 3:54-57.) Figure 2, showing an embodiment of the “Yahoo! Search” website repository, is included below:



(EX1019, Fig. 2.)

63. In one embodiment, Wu discloses a process of locating websites stored in the searching repository using at least two search strings. (EX1019, 3:33-35.) In this process, each string is separately searched through the repository, and any websites containing the individual string are collected in a “match list.” (EX1019, 6:54-62.) Websites present in each of the match lists, corresponding to the individual strings, are then collected and returned to the user as the result of the search. (EX1019, 8:66-11:32.)

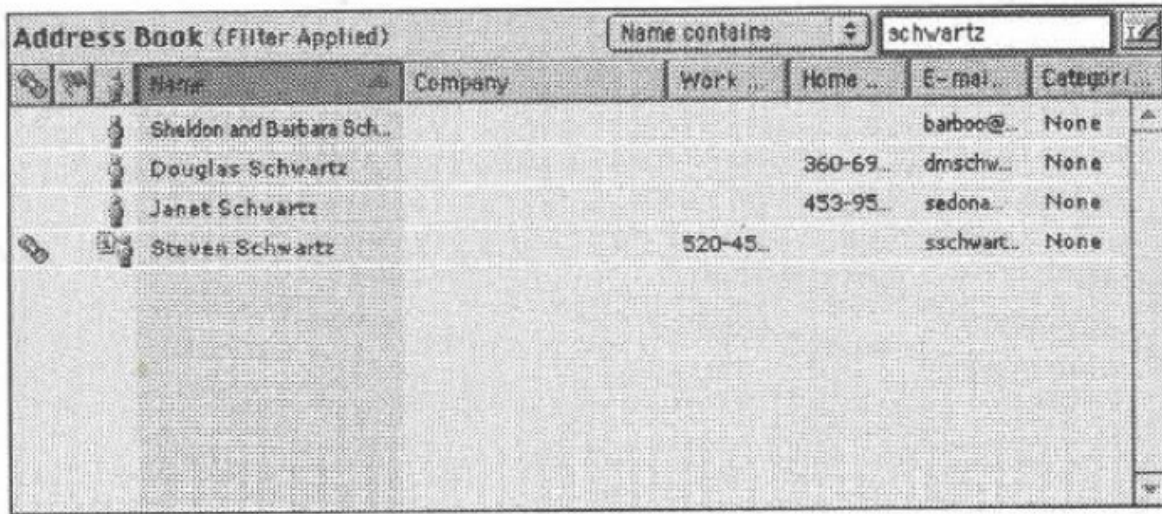
E. Entourage (Schwartz (EX1004), Sebastian (EX1005), and Barber (EX1006))

64. Running Microsoft Office 2001 for Mac (“Schwartz”), Microsoft Office 2001 A Hands-On Guide for Macintosh, (“Sebastian”), and Visual

QuickStart Guide Entourage 2001 for Macintosh (“Barber”) are all books that were published in 2001.

65. Schwartz, Sebastian, and Barber (collectively, “Entourage”) instruct users on operating Entourage 2001 for Macintosh—an email system designed by Microsoft. (EX1004, (i); EX1005, (a)-(c), 171-88; EX1006, (i), 1-3, 5, 9, 283-392.) The portions of Entourage relied upon by Dr. Terveen generally relate to searching for emails. (*See generally* Ex. 1004, 2-12, 161-64; Ex. 1005, 173; Ex. 1006, 283-85, 384-87.)

66. Entourage discloses a filtering method that identifies emails with headings (i.e., To, From, Subject, Category) containing the string entered by the user. (EX1004, 161; EX1005, 173; EX1006, 358, 384.) The results from this method are displayed “[a]s [the user] type[s] the filter text.” (EX1004, 9.) Accordingly, the filtering method concurrently displays emails with headings that include the currently entered string, as the user types out the search. (EX1004, 161; EX1005, 173; EX1006, 384-86.) Figure 1.12 from Entourage, displaying the email filtering window, is displayed below:



(EX1004, 9, Fig. 1.12.)

67. Entourage also discloses the Find and Advanced Find delimited searching methods for finding specific emails on the platform. (EX1004, 9-12.) These searching methods are delimited—meaning the user will not see the results until they execute the search. (EX1004, 162-63; EX1006, 384-87.) In the Find method, a user can search for specific terms in the headers or the body portion of an email. (EX1004, 9-10, 161–62; EX1006, 384-85.) Using the Advanced Find method, a user can “search multiple item types (messages *and* events, for example); and search any folder regardless of the one that is currently selected.” (EX1004, 11.) Figures 1.13 and 1.17 from Entourage, displaying the Find and Advanced searching windows, respectively, are displayed below.

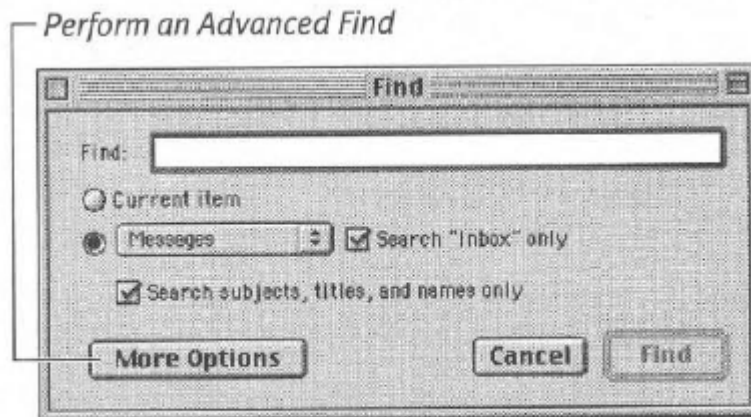


Figure 1.13 You can conduct simple searches by specifying criteria in the Find dialog box.

(EX1004, 9, Fig. 1.13.)

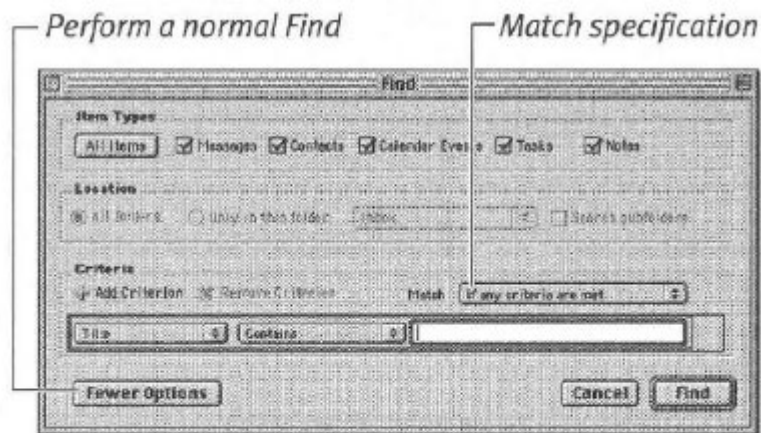


Figure 1.17 The Advanced Find dialog box enables you to search multiple item types simultaneously, specify a location, and use multiple criteria that aren't available to you when you use the Find command.

(EX1004, 11, Fig. 1.17.)

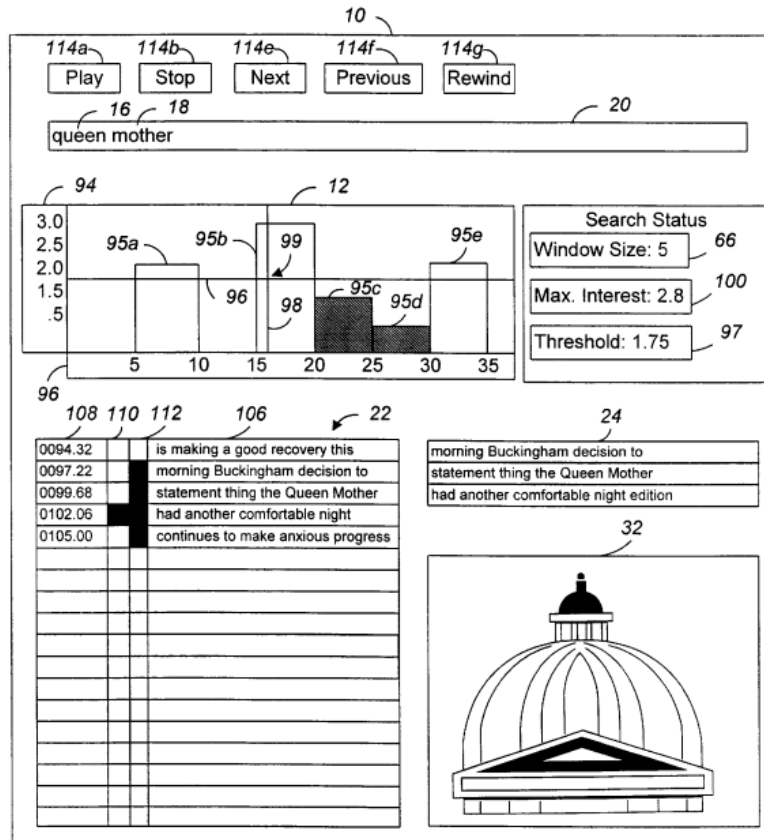
F. True

68. U.S. Patent No. 6,112,172 (“True”) was filed on March 31, 1998, and issued on August 29, 2000.

69. True purports to be generally relates to the field of “interactive searching.” (EX1009, 1:4.) True says that it discloses a method to “control a search of a file that represents underlying raw information, and to find parts of the file that represent material of interest in the raw information.” (EX1009, 1:38-42.)

70. In the primary embodiment disclosed by True, the “underlying raw information” is a transcript of “text produced by a speech recognizer.” (EX1009, 1:55-57.) In this embodiment, a “user types a query, consisting of a combination of words of interest 16, 18, into a query window 20” and a “result window 12 indicates the degree to which portions of the transcript 14 satisfy the query.” (EX1009, 3:35-40, Fig. 2.) Figure 2 of True, showing a disclosed graphical user interface for a search engine, is shown below:

FIG. 2



(EX1009, Fig. 2.)

71. True is specifically directed to searching for words within a computer transcription of an audio file, taking into account the confidence level of the accuracy of the speech-to-text transcription for each word in the transcript.

(EX1009, 1:55-60, 5:40-64.)

72. The results of True's search method are not a list of traditional search results, but rather a "result window 12 [which] indicates the degree to which portions of the transcript 14 match the query particularly well." (EX1009, 3:37-40.) Each recognized word in the transcript includes a confidence level, "a number

between zero and one indicating an estimated likelihood that the recognized word is correct.” (EX1009, 4:21-33.) To perform a search, the user specifies a window size in seconds to define “transcript windows” which “correspond[] to a segment in the digitized audio” and then enters a query. (EX1009, 4:39-65.) True’s query results consist of a graphical display showing the “interest level” of transcript windows, calculated “as the sum of the confidences of each word in the transcript window matching a word in the query.” (EX1009, 5:29-50.) There is no disclosure in True as to how to compute confidence levels for partial-word matches or incremental matches and thus nowhere does True enable or teach such.

G. Baeza-Yates

73. Modern Information Retrieval (“Baeza-Yates”) is a textbook published in 1999.

74. Baeza-Yates generally relates to Information Retrieval (“IR”) in the context of searching the internet—the introduction of which the textbook credits with increasing the importance of IR. (EX1023, 1-3).

75. Baeza-Yates defines IR in contrast to data retrieval. (EX1023, 1-2). Data retrieval, as Baeza-Yates explains, “consists mainly of determining which documents of a collection contain the keywords in the user query.” (EX1023, 1.) Conversely, Baez-Yates outlines that IR relies upon “natural language text” to “retrieve all the documents which are relevant to a user query while retrieving as

few non-relevant documents as possible.” (EX1023, 2.) Given the difficulties of finding “useful information on the Web,” Baeza-Yates stresses the usefulness of IR when searching for information on the internet. (EX1023, 3.)

XI. GROUND 1: THE CHALLENGED CLAIMS ARE NOT OBVIOUS OVER WILCOX (EX1059), LONDERGAN (EX1061), RASKIN (EX1021), AND WU (EX1019)

76. It is my opinion that the challenged claims are not obvious over Dr. Terveen’s proposed combination of Wilcox, Londergan, Raskin, and Wu. Wilcox, Londergan, Raskin, and Wu fail to disclose critical elements of the claims, and Dr. Terveen has not articulated a reason why a POSITA would have been motivated to either modify Wilcox (which discusses the functionality of Lotus Notes v. 3.0) to combine it with Londergan (which discusses a different version of Lotus Notes, R5), or why a POSITA would have been motivated to further combine the Wilcox-Londergan combination with both Raskin and Wu. Finally, Dr. Terveen fails to explain with particularity how exactly a POSITA would implement the disparate teachings of the Ground 1 references.

A. A POSITA Would Not Have Been Motivated To Combine Wilcox And Londergan Because Lotus Notes 3.0 And Lotus Notes R5 Have Distinct Features And Are Not The Same System.

77. It is my opinion that a POSITA would not have found the teachings of Wilcox (EX1059) and Londergan (EX1061) to be complementary and would not have combined the references and treat them as a single reference (“the Lotus

Notes references”). Dr. Terveen’s proposed motivation to combine Wilcox and Londergan has minimal explanation and hinges on his assumption that the two references allegedly describe the same system:

A POSITA would have been motivated to combine the complementary teachings on Lotus Notes from Londergan (EX1061) and Wilcox (EX1059) to obtain a more complete and comprehensive understanding of the Lotus Notes system, its features, and its capabilities. For example, Wilcox provides additional details on the core functionality of Lotus Notes’ user interface and search functionality (EX1059, 84-85, 159-168) while Londergan provides details on new features of Lotus Notes R5, including its calendaring and auto-complete features (EX1061, 45-74, 146-147). A POSITA would have looked to Wilcox to gain a solid understanding of Lotus Notes’ core functionality, and would have looked to Londergan for details on the new features in Lotus Notes 5 that were added to this core functionality.

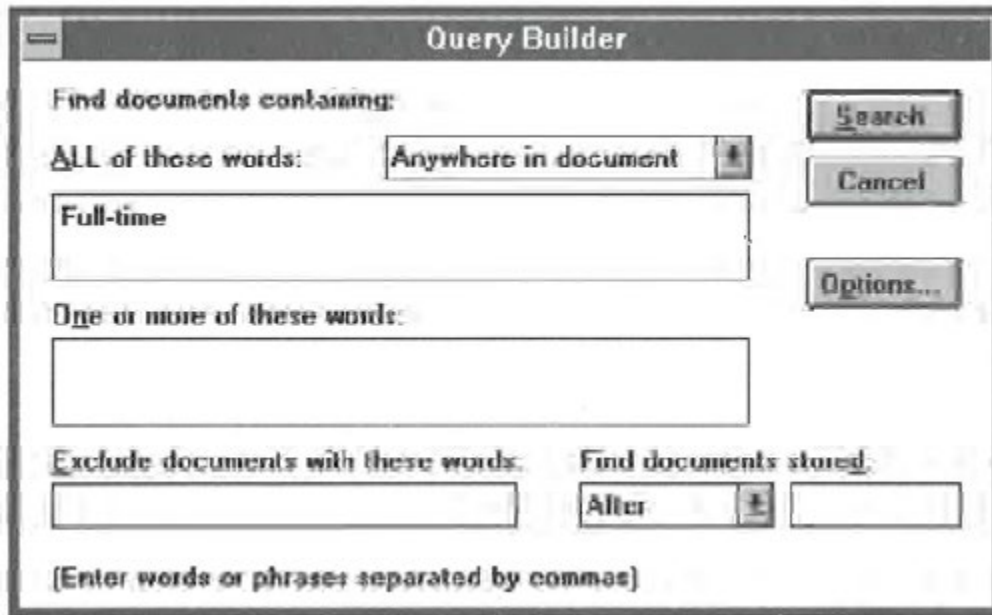
(EX1003, ¶73.) Dr. Terveen implies that the two references describe the same system. I disagree. Each reference discusses different versions of the Lotus Notes system. (See EX1003 ¶57.) Wilcox discusses Lotus Notes 3.0, which was released in 1993. *Id.* Londergan discusses Lotus Notes R5, released in 1999. (*Id.*) Not only were these two versions under the Lotus Notes brand released six-years apart,

there was at least one other major release (v. 4.0) of Lotus Notes between 1993 and 1999. (EX1059, (h); EX1061, (iii), 114.)

78. While Dr. Terveen opines that a POSITA would look to Londergan (R5, EX1061) to understand “some of the new features of Lotus Notes 5” and to Wilcox (v. 3.0, EX1059) for “additional details on Lotus Notes’ core user interface and search functionality,” as discussed below, the relevant functionality in Lotus Notes 3.0, as disclosed in Wilcox, and Lotus Notes R5, as disclosed in Londergan, are starkly different, particularly with respect to the functionality Dr. Terveen relies on for his opinions that Lotus Notes allegedly renders the claims obvious. Dr. Terveen never explains what his proposed Wilcox-Londergan combination would look like or which features of Londergan would be implemented in Wilcox’s system. Accordingly, in my opinion a POSITA would not be motivated to combine Wilcox and Londergan.

1. Query Builder (v. 3.0) vs. Search Builder (R5).

79. The 1993 version of Lotus Notes (v. 3.0) included a feature called Query Builder. Query Builder, pictured below, provides multiple fields where, Dr. Terveen alleges, a user can type multi-string search queries:



(EX1059, 163; *see also* EX1003, ¶61.)

80. The 1999 version of Lotus Notes (R5) eliminated the Query Builder feature and instead included a different feature with different functionality called Search Builder. (EX1061, 141-42.) Search Builder provides multiple fields, each of which can only be used for a single search string, as shown below:



(EX1061, 142.) Dr. Terveen’s analysis ignores this critical difference in functionality between the two versions of Lotus Notes and haphazardly picks and chooses features between the two. Dr. Terveen relies on the Query Builder functionality described in Wilcox in an attempt to show that the Wilcox-Londergan combination discloses searching on multiple strings entered into a single search field. (EX1003, ¶61.) He then separately relies on the Search Builder functionality described in Londergan for allegedly disclosing other features, such as highlighting search strings in documents (once a user opens the document from the search result window and does another search) (EX1003, ¶97) and searching for appointment information (EX1003, ¶99).

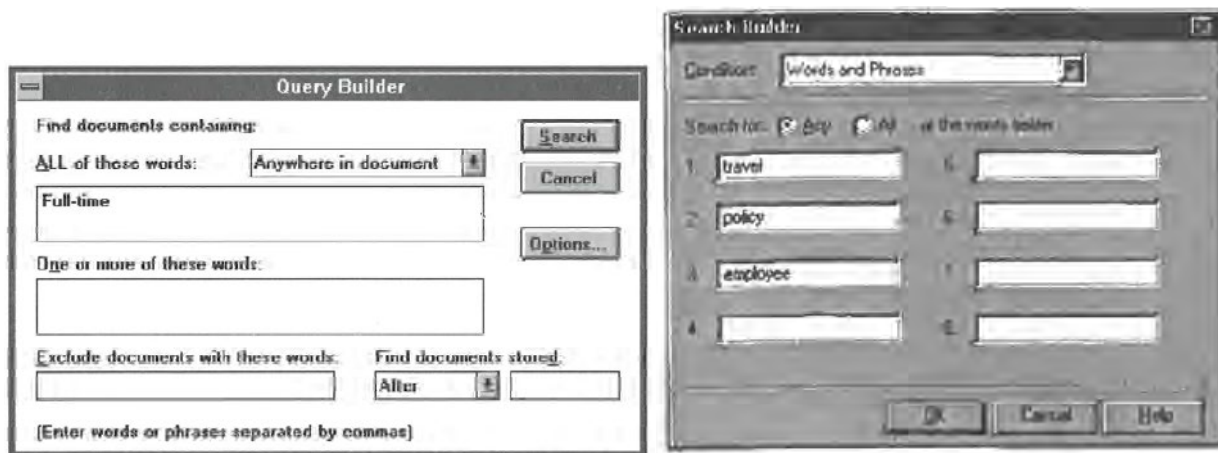
2. Search Bar

81. Both versions of Lotus Notes also include a separate search feature called the Search Bar. (EX1059, 43-44, 161; EX1061, 139-41.) Dr. Terveen relies on the Search Bar to disclose searching on a first and second search string entered into a first search field, where the first and second strings are separated by a string separator character. (EX1059, 161; EX1061, 139-41; EX1003, ¶81.) However, Dr. Terveen ignores the fact that the Search Bar can perform multi-word searches only under two conditions: first, the Search bar can perform searches for exact multiword phrases if the phrase is enclosed in quotation marks (EX1059, 162; EX1061, 140); second, the Search bar can perform searches on multiple search strings that are connected with a Boolean operator, such as “AND” or “OR” (EX1059, 162; EX1061, 140-41). The Search Bar functionality as discussed here and as disclosed in Wilcox and Londergan appears to be the same for both versions of Lotus Notes, version 3.0 and R5. In neither circumstance can the Search Bar search multiple, separate strings using a string separator character.

3. Wilcox (Lotus Notes v. 3.0, EX1059) and Londergan (Lotus Notes R5, EX1061) Are Improperly Combined.

82. Dr. Terveen suggests that the Wilcox and Londergan references are complementary and can be combined in part because both references disclose “entering the search strings in a single search field” of a simple Search bar or a Query Builder/Search Builder effectively treating the Query Builder in Wilcox

(Lotus Notes 3.0) and Search Builder in Londergan (Lotus Notes R5) as the same functionality. (EX1003, ¶¶59-61.) Dr. Terveen is wrong. As I explained above, Wilcox and Londergan each have differences and one of those differences is the advanced search dialogue boxes (Query Builder and Search Builder). The more modern version of Lotus Notes decided to move away from a single search field and instead provided users with multiple fields to break out individual terms. The visual comparison below illustrates the change in functionality:



Query Builder/Lotus Notes 3.0 (left), Search Builder/Lotus Notes R5 (right)
(EX1059, 163; EX1061, 142.)

83. Additionally, a POSITA would have seen Wilcox (Lotus Notes 3.0) to be a more rudimentary system as shown by the other changes Londergan (Lotus Notes R5) made to search functionality, including providing more search variants. For example, Londergan allows for “fuzzy search” which is an option that “tells Notes to use not only the word you typed, but similar phrases and corrected

spellings if you misspelled your criterion.” (EX1061, 144.) Londergan also searches over an increased corpus of data when compared to Wilcox, including searching for people. (*See* EX1061, 147.)

84. Thus, in my opinion, a POSITA would not view Wilcox as simply providing “additional details” on Lotus Notes’ “user interface and functionality” as Dr. Terveen suggests. (EX1003, ¶57.) Instead, a POSITA would view Wilcox as being directed to an earlier, phased out version of the program when it comes to how searches could be done.

85. In my opinion, Dr. Terveen does not provide a concrete example of a combined version of Lotus Notes using the Wilcox and Londergan references. However, to the extent ¶73 of Dr. Terveen’s Declaration (EX1003) suggests combining certain features, I disagree with Dr. Terveen’s cherry-picked features to suggest a combination of the two different versions. Dr. Terveen’s cherry-picking of certain aspects of both references seems to be primarily motivated by impermissible hindsight. Dr. Terveen first suggests that Wilcox provides “additional details on core functionality of Lotus Notes’ user interface and search functionality. (EX1003, ¶73.) He then suggests a POSITA would include Londergan’s calendaring and auto-complete features. (EX1003, ¶73.) Based on his analysis of the specific claims, it appears Dr. Terveen’s proposed combination would keep Wilcox’s “Query Builder” despite the existence of a more modern

version. (*See* EX1003, ¶82.) In other words, Dr. Terveen asserts that a POSITA would take the older version of Lotus Notes and incorporate some, but not all, of the new features in a version from six years later (and with two versions being released in between). Dr. Terveen does not explain why a POSITA would engage in such cherry-picking to arrive at Dr. Terveen's proposed combination.

86. Dr. Terveen's characterization of these two different versions of Lotus Notes as a single prior art reference is infected by hindsight. Dr. Terveen's expectation of success arguments illustrate this hindsight. In asserting that a POSITA would have had a reasonable expectation of success in combining the two Lotus Notes references, Dr. Terveen only references the Search bar that is common between Wilcox and Londergan. (*See* EX1003 at 74 ("the references describe using the same search tool (e.g., Lotus Notes' Search Bar) ...").) In doing so, Dr. Terveen ignores the fact that Londergan (Lotus Notes R5), the most current version of Lotus Notes at the time of the '139 Patent's priority date, had abandoned using a single search field in its "Search Builder" tool. (EX1061, 142 ("Instead of entering in the text box words or phrases that you want to find, as we describe in the previous section, click Multiple Words and enter one word or phrase in the numbered boxes in the Search Builder dialog box.")) In my opinion, a POSITA would not have chosen to work with an older, outdated version of a

product as the basis for a combination when a more recent and potentially improved version of the product existed.

4. Dr. Terveen's Proposed Wilcox-Londergan Combination Represents Improper Hindsight Bias.

87. Additionally, in my opinion, Dr. Terveen is trying to identify aspects of the claimed invention in the prior art using hindsight. As I described above, over the years Lotus Notes had moved away from a unified search field. The most modern version at the time, Lotus Notes R5 used individual boxes for individual search terms when formatting more complicated queries. (EX1061, 142.) The '139 Patent taught "a first instruction configured to read at least a first search string and a second search string entered into a *same first search field*." (EX1001, 2:43-49 (emphases added).) Thus, a POSITA would have understood Lotus Notes, as a whole, to be moving away from the search technology claimed in the '139 Patent, yet Dr. Terveen cherry-picks certain functionality from both versions.

88. As I described above, the newer version of Lotus Notes (R5) as discussed in Londergan is more advanced in a number of other ways that would lead to a POSITA to prefer the newer version when building a search user interface. A POSITA reading Londergan would not have thought to halt or reverse the progress of the developers of Lotus Notes R5 by reverting some of the changes that had been made to Londergan's (Lotus Notes R5) "Search Builder" to make it more similar to Wilcox's (Lotus Notes 3.0) "Query Builder." Dr. Terveen fails to

provide any reason why a POSITA would have considered reverting the changes to Lotus Notes R5's "Search Builder" other than hindsight bias that the "Query Builder" is a better fit for Dr. Terveen's argument. *Id.*

89. Accordingly, a POSITA would not have been motivated to combine Wilcox and Londergan as suggested by Dr. Terveen.

B. A POSITA Would Not Have Been Motivated To Combine Wilcox-Londergan With Raskin

90. I disagree with Dr. Terveen that a POSITA would have been motivated to combine Wilcox-Londergan with Raskin. Dr. Terveen admits that the improper Wilcox-Londergan combination does not disclose incremental searching and turns to Raskin to remedy this issue. (EX1003, ¶84.) A POSITA would not have been motivated to combine these references.

91. Dr. Terveen is wrong that a POSITA would look to Raskin to implement incremental search into the claimed invention, which is which is directed to receiving multiple, partial search strings in a single search field to search across "a set of documents." '139 Patent cl. 1. This is because Raskin only discusses problems with, and solutions for, searching in single file and not across multiple documents. Raskin is a textbook spanning a broad list of topics related to philosophies and approaches to designing an optimal user interface, or a humane interface. (EX1021, xi-xii, 6-7.)

92. The pages that Dr. Terveen cites to in Raskin to support a motivation to combine it with Lotus Notes search functionalities are in a chapter titled “String Searches and Find Mechanisms.” (EX1021, 124-27.) This entire section of Raskin discusses ways to use string searches *in a single document*. For example, Raskin discusses searching for the phrase “little Tatsu” in a “long letter” and demonstrates how the “Find and Replace” function works when a single document is being searched. (*Id.*, 124-25 (discussing that “searches usually start from the current cursor location [in the document] and continue forward through the text [of the document]”).) Indeed, the only specific search system that Raskin mentions in the pages cited by Dr. Terveen is EMACS, which Raskin describes as “an editor used with the UNIX operating systems.” (*Id.*, 125.) A POSITA would understand that EMACS was a text editor that only allowed for search within a single document, and not across multiple documents. Indeed, Raskin’s description of a search being conducted in EMACS where the system searches “for the first instance of that character in the chosen search direction” further confirms that such a search is being done in a single document. (EX1021, 125.)

93. Even Raskin’s discussion of incremental search makes clear that he is discussing it only in the context of a single document. Raskin discusses that incremental search is a “less common strategy” and is implemented in EMACS, which only searches single document. When discussing how to use incremental

search, Raskin states that the “found instance” (search hit) “should appear in the middle of a display area, not at the top or the bottom. Thus, material both before and after the instance is visible; that is, the found instance is displayed in context.” (EX1021, 127.) A POSITA reading Raskin’s discussion of implementing incremental search would understand he was referring to here, as throughout the cited pages, a search of a single document where the system can move to where there is a first hit and show the user the context around that hit. Raskin never discusses using incremental search across multiple documents.

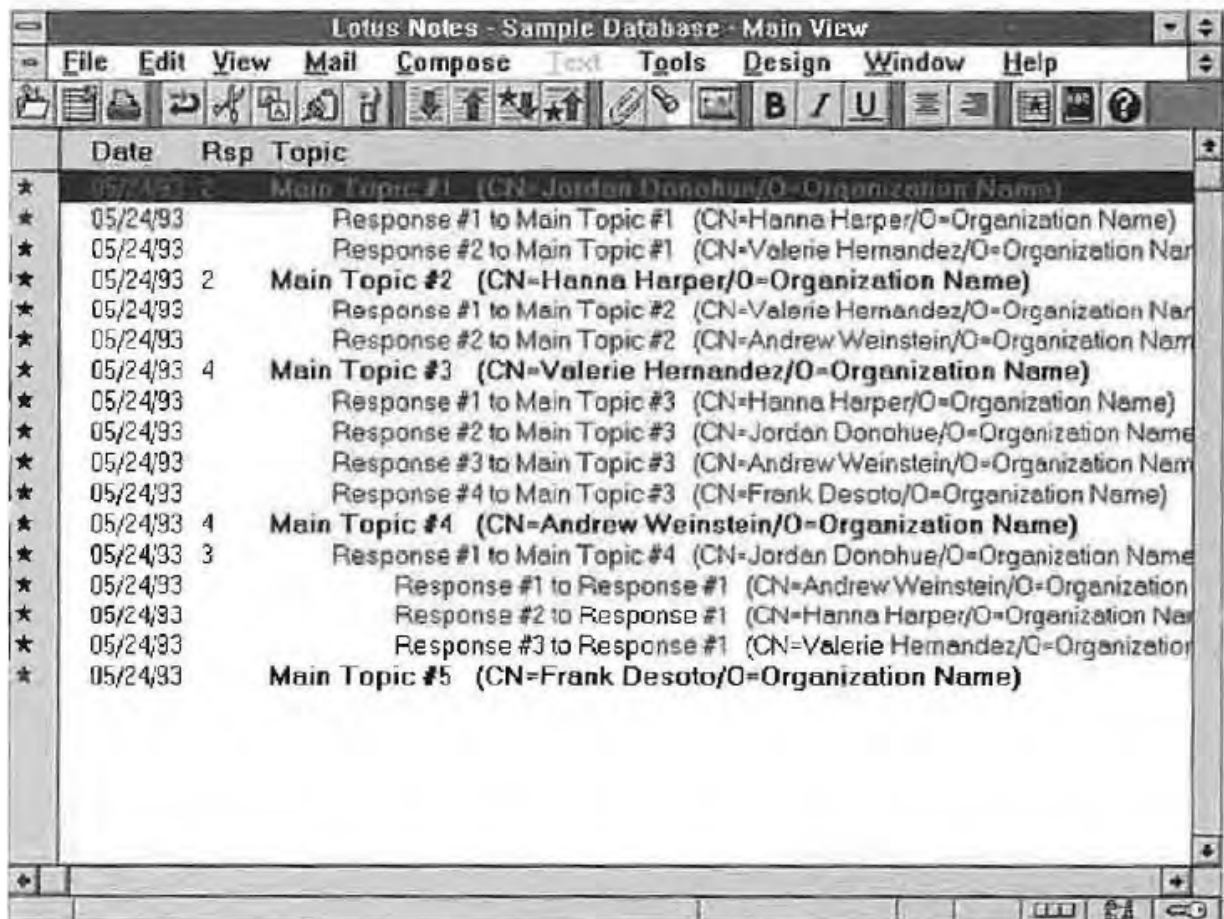
94. If a POSITA would have understood Raskin to apply to a search of multiple documents (which it does not), a POSITA would not have been motivated to implement incremental search because there are serious technical challenges with implementing the complex search strategies of the challenged claims in the systems on which Dr. Terveen relies. Dr. Terveen does not consider or provide an explanation on how to circumvent these challenges. Some of these challenges are, in fact, described in the same prior art Dr. Terveen uses as support for his opinions. Raskin himself 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.” *Id.* at 126 (emphasis added).

95. In my opinion, a POSITA would not have been motivated to use the teachings of Raskin with the various search systems disclosed in Lotus Notes because Raskin's proposals are incompatible with Lotus Notes generally. Dr. Terveen's does not explain what would motivate a POSITA use Raskin in the face of these incompatibilities. Lotus Notes, as I described above and as generally described by Wilcox and Londergan, is a groupware product that allows users to share work from a user's PC with fellow collaborators electronically and provides features such as sending emails, storing searchable information in a centralized location that multiple users can access, and provide a central location to look for people's daily calendars. (See EX1059, 1-3; EX1061, 6-9.) Raskin proposes designing user interfaces to take into account limitations of humans, like having one locus of attention (EX1021, 9), and proposing solutions to a variety of issues like user interfaces requiring users to switch contexts (EX1021, 31) and reducing complexity of commanding a computer to perform certain tasks or actions by making the interface monotonous (EX1021, 66-68) and modeless (EX1021, 37).

96. Wilcox and Londergan teach that in Lotus Notes, a user can create a document, save it to a central database, have another user/employee read the original document and then respond to that document. (EX1059, 2-3.) A POSITA would have understood that Lotus Notes is designed around databases and views to facilitate users finding and viewing "main topic" documents and creating

“response” documents through views, as shown below. (EX1059, 24-25; *see also* EX1061, 8 (“In Notes, databases contain views and folders. Both contain documents”), 36 (“you use views and folders to sort through and organize your incoming and outgoing e-mail messages. Your e-mail database comes ready-made with eight views and folders.”).) An example of the specific hierarchy used by Lotus Notes is illustrated below:

The Main View of the Sample Database



	Date	Rsp	Topic
★	05/24/93	2	Main Topic #1 (CN=Jordan Donohue/O=Organization Name)
★	05/24/93		Response #1 to Main Topic #1 (CN=Hanna Harper/O=Organization Name)
★	05/24/93		Response #2 to Main Topic #1 (CN=Valerie Hernandez/O=Organization Name)
★	05/24/93	2	Main Topic #2 (CN=Hanna Harper/O=Organization Name)
★	05/24/93		Response #1 to Main Topic #2 (CN=Valerie Hernandez/O=Organization Name)
★	05/24/93		Response #2 to Main Topic #2 (CN=Andrew Weinstein/O=Organization Name)
★	05/24/93	4	Main Topic #3 (CN=Valerie Hernandez/O=Organization Name)
★	05/24/93		Response #1 to Main Topic #3 (CN=Hanna Harper/O=Organization Name)
★	05/24/93		Response #2 to Main Topic #3 (CN=Jordan Donohue/O=Organization Name)
★	05/24/93		Response #3 to Main Topic #3 (CN=Andrew Weinstein/O=Organization Name)
★	05/24/93		Response #4 to Main Topic #3 (CN=Frank Desoto/O=Organization Name)
★	05/24/93	4	Main Topic #4 (CN=Andrew Weinstein/O=Organization Name)
★	05/24/93	3	Response #1 to Main Topic #4 (CN=Jordan Donohue/O=Organization Name)
★	05/24/93		Response #1 to Response #1 (CN=Andrew Weinstein/O=Organization Name)
★	05/24/93		Response #2 to Response #1 (CN=Hanna Harper/O=Organization Name)
★	05/24/93		Response #3 to Response #1 (CN=Valerie Hernandez/O=Organization Name)
★	05/24/93		Main Topic #5 (CN=Frank Desoto/O=Organization Name)

EX1059 at 26.

97. A POSITA would have understood that Lotus Notes is focused primarily on centralized collaboration around documents, files, and file structures. 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. (*See, e.g.*, EX1021, 119 (“With the removal of file names go all the mechanisms for dealing with file names, such as directories of file names, rules for editing file names, ...”), 121 (“Eliminating hierarchical file structures does not mean that you have to give up structuring your stored information.”).) 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, in my opinion a POSITA would not look to Raskin to improve Lotus Notes.

98. To the extent Dr. Terveen asserts that a POSITA would be motivated to apply the incremental search of Raskin to the search tools discussed in Wilcox-Londergan on the simple basis that incremental search was known, I disagree. Dr. Terveen attempts to support his motivation to combine argument by repeatedly suggesting incremental searching was well known and that a POSITA would have known “fundamentally how incremental searching works.” (*See, e.g.*, EX1003, ¶¶ 84, 87.) Instead of suggesting why a POSITA would have looked to the single

document incremental search disclosed in Raskin, Dr. Terveen relies heavily on his “State of the Art” section in his declaration to suggest a motivation to combine. *Id.* However, Dr. Terveen’s discussion of the background references allegedly demonstrating the state of the art in incremental searching only involve many long string cites with few parentheticals and direct quotes. As I explained above in the Technology Background, Dr. Terveen mischaracterizes the prior art he cites and none of the references, taken individually or together, disclose incrementally searching a set of documents using two strings or two partial strings.

99. Accordingly, a POSITA would not have been motivated to combine one of the search tools in Wilcox-Londergan with Raskin’s incremental search because Raskin was directed to searching a single document, Raskin’s entire philosophy of user interfaces was incompatible with Lotus Notes, and Raskin noted the extreme technical difficulties with even using incremental searches.

C. A POSITA Would Not Have Been Motivated To Combine Wilcox-Londergan With Wu.

100. Dr. Terveen admits that neither the Wilcox-Londergan combination nor Raskin disclose *identifying one or more documents included in both the first one or more documents and the second one or more documents* and relies on Wu to supply this limitation. (EX1003, ¶¶76, 89-90.) I disagree with Dr. Terveen’s opinion that a POSITA would have been motivated to combine Wilcox-Londergan and Raskin with Wu and it is my opinion that Dr. Terveen failed to adequately

articulate any such motivation. As I described above, Lotus Notes is a groupware system with functionality intended to promote collaboration. Lotus Notes has search functionality through its search box and, depending on which version, its “Query Builder” or “Search Builder” tools. (EX1059, 163; EX1061, 142.)

101. Wu is titled “Information retrieval from hierarchical compound documents” and is directed to a system for searching large document repositories, such as the internet. (EX1019, Abstract, 3:24-36, 4:50-53; 12:54-58.) Indeed, Wu is specifically focused on improving searching the Internet. *See, e.g.*, EX1019 at 3:53-57 (referring to the “Yahoo! Search database” embodiment”), 4:15-21 (“System 10 is shown comprising an HTTP client 12 connected to a search server 14 via Internet 16.”), Figs. 1-2. Wu accomplishes improved search results by returning (1) documents that directly match one of the search terms; and (2) documents that indirectly match a search term by virtue of being a “child document in the hierarchy from a parent document matching all or part of the query term.” (*Id.*, 2:53-69, 3:40-43; 7:24-36.) In other words, a query according to Wu’s method will return results that may not include *any* of the terms making up the query.

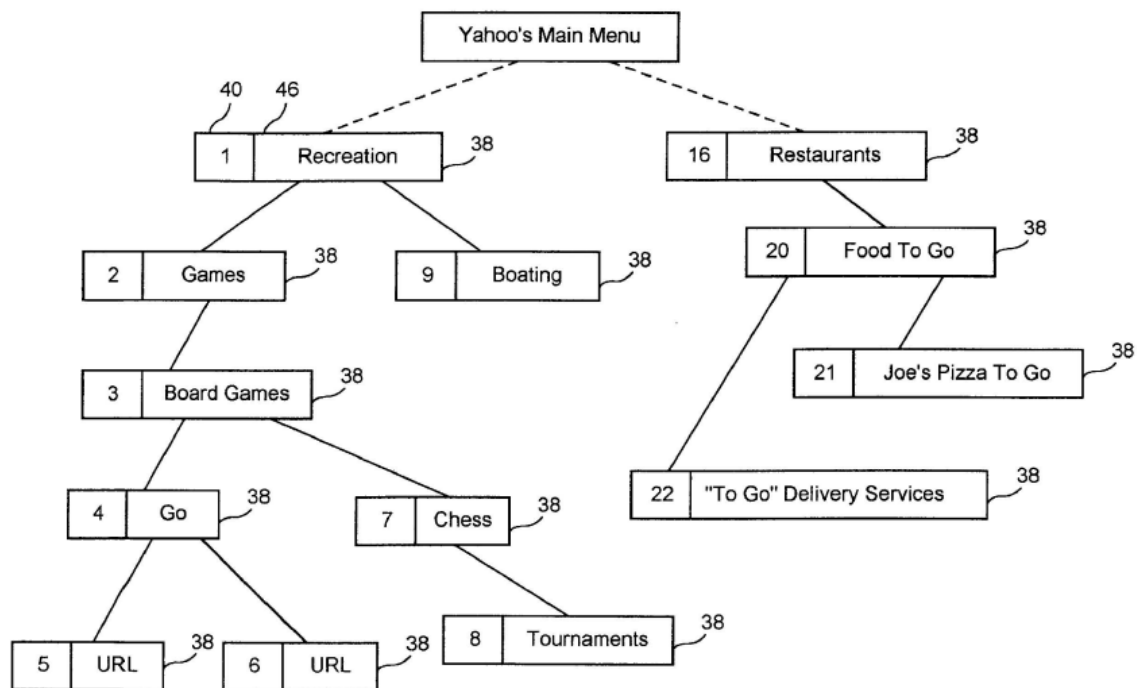
102. Dr. Terveen argues a POSITA would have been motivated to combine the alleged “intersection approach” from Wu’s hierarchical search method with the Wilcox-Londergan-Raskin combination because Wu’s method

“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.” (EX1003, ¶¶76, 89.) Dr. Terveen, however, fails to explain why a POSITA would have picked Wu and combined it with Wilcox, Londergan, and Raskin to arrive at the ’139 Patent’s claimed invention.

103. First, Dr. Terveen failed to articulate what search tool in which version of Lotus Notes in Wilcox or Londergan a POSITA would modify to implement the teachings of Wu. Based on this deficit alone, Dr. Terveen’s combination fails. To the extent one could surmise which search tool in Wilcox-Londergan that Dr. Terveen is referring to, a POSITA would not have looked to Wu to improve any search feature of the Wilcox-Londergan combination (or Lotus Notes generally) or to implement the alleged “intersection approach” described by Dr. Terveen. A POSITA would have known that searching the Internet, as contemplated by Wu, is materially different than searching a user’s computer or an organization’s central document database as taught by Wilcox-Londergan. As one of the asserted references describes, “searching the internet presents problems with the data itself, including: “[d]istributed data,” “high percentage of volatile data,” “[l]arge volume,” and “unstructured and redundant data.” (See EX1023, 368.) A POSITA reading Wilcox-Londergan would not need to consider such problems as the Lotus Notes program specifically contemplates searching on a much smaller

scale as it provides for a “full-text index” or a “list of text and documents that the search uses to find documents containing the text you want.” (EX1061, 135.) In fact, Lotus Notes does provide the option of web searching but, instead, explicitly states a user should specify a preferred search engine and Lotus Notes opens the specified search engine and allows that engine to conduct the search. (*See* EX1061, 135.) Clearly, a POSITA would not have been motivated to incorporate features from a web search system like Wu when Lotus Notes explicitly refers users to an outside search engine designed specifically for Internet searches, which are materially different than searching for large volumes of documents on a computer or computer network.

104. Additionally, Wu’s search method is, in part, predicated on using a specific file structure/hierarchy as it describes searching for specific nodes of the structure. *See* EX1019 at 4:50-5:6; Fig. 2 (below).



Such a complex file structure is not contemplated by Lotus Notes. Additionally, as I described above with Raskin, a POSITA would have understood that applying Raskin’s teaching of eliminating hierarchical file structure would not work with Wu. Thus, a POSITA would have no reason to look to Wu to improve searching in Lotus Notes generally or the Wilcox-Londergan-Raskin combination.

105. Dr. Terveen fails to describe with particularity how a POSITA would modify any of the search systems in Wilcox-Londergan to incorporate Wu’s intersectional search teachings. Dr. Terveen recites only says that it “would have required only simple and straightforward software modifications” but never describes what modifications are made and how those modifications would work

within any particular search system in Wilcox-Londergan. *See* EX1003 at ¶¶ 77, 91. Accordingly, Dr. Terveen failed to provide a motivation to combine Wilcox-Londergan with Wu's disclosure of intersectional search.

D. The Prior Art Does Not Disclose All Limitations Of The '139 Patent's Claims

106. Dr. Terveen does not assert that any claims of the '139 Patent are anticipated by any single prior art reference and relies on the combination of Wilcox-Londergan, Raskin, and Wu to show that each of the challenged claims is obvious. However, the combination of these four prior art references fails to disclose certain claim limitations altogether. Because Dr. Terveen has not shown that all claim limitations are taught by the Ground 1 combinations, Ground 1 fails.

1. The Combination Of Wilcox, Londergan, Raskin, And Wu Does Not Teach The First And Second Partial Search Strings Being Separated By A String Separator Character In The First Search Field (Limitations [1.4], [23.5], and [27.4])

107. Dr. Terveen alleges the combination of Wilcox and Londergan on its own disclose receiving two search strings in a single search field that are separated by a string separator character. (EX1003, ¶¶80, 114, 119.) I disagree. Dr. Terveen's proposed combination fails to disclose (1) two *partial* search strings in the same search field; and (2) two partial search strings separated by a string separator character.

108. Specifically, Dr. Terveen relies on, and groups together, the Lotus Notes Search Bar functionality (described above in §XI.A.2) and the “Query Builder/Search Builder” functionality (described above in §XI.A.1). (EX1003, ¶¶59, 61, 80 (equating the “single search field” with “a blank box of a Search bar and/or blank box of a Query Builder/Search Builder dialog box”). As I explained above, the Query Builder and Search Builder functionalities are meaningfully different and Search Builder (found in Lotus Notes R5) does not allow for multiple strings to be entered into a single search field. (*Supra* §XI.A.1) Because Dr. Terveen improperly groups together the disparate search systems and relies on them collectively (as if they are one system), when at least some of the underlying systems do not even disclose this claim limitation, Dr. Terveen’s argument that this limitation is disclosed by Wilcox-Londergan fails.

109. Furthermore, Dr. Terveen fails to specify any disclosure of a *partial* search string in Wilcox-Londergan; rather, Dr. Terveen, acknowledging that there is no such disclosure, vaguely references Raskin and states that “a POSITA would have understood or at least found it obvious that the received search strings are partial search strings” in the Wilcox-Londergan-Raskin combination because Raskin allegedly teaches incremental search. (EX1003, ¶83.) But, as explained above, Raskin only teaches searching over a single string and adding characters to that one string to narrow search results in a single document. (*See, e.g.*, EX1021,

124, 126.) Dr. Terveen has not cited to any suggestion or examples in Raskin of searching over two separate strings, let alone two separate partial strings.

110. For the string separator character portion of these limitations, Dr. Terveen points to an example query disclosed in Londergan, which allegedly teaches that two search strings may be entered into the Search Bar if they are separated by a Boolean operator such as “AND.” (EX1003, ¶81.; EX1061, 140 (disclosing search example “gold and silver”).) Dr. Terveen suggests, without citing to any intrinsic evidence for support, that in this example “AND” is a string separator and “gold” and “silver” are the two partial strings. (EX1003, ¶81.) As an initial matter, it appears Petitioner did not request construction of the term “string separator character” and Dr. Terveen does not explicitly state he is relying on a specific construction of the term. (*See* Pet., 12 (“Unless otherwise noted, Petitioner applies the plain and ordinary meaning of all claim terms below.”; EX1003, ¶52.) Furthermore, the intrinsic evidence does not support Dr. Terveen’s implicit claim construction. First, the ’139 Patent distinguishes between string separator *characters* (such as space, comma, etc.) and “modifiers” (such as AND, OR, NOT, etc.). (*Compare* EX1001, cl. 6 with cl. 22.) Second, a POSITA would not understand Boolean operators such as “AND” to function as string separator *characters*. (*See, e.g.* EX1059, 162 (discussing the use of “logical operators AND, OR, NOT, and ACCRUE to find documents,” which a POSITA would understand

to be referring to a Boolean operator).) Indeed, a POSITA would not have understood Boolean operators like AND to be a string separator as used in the claims because when that word is used in a search, it would be read (and not ignored like the word “the” in a phrase) and the system would need to perform an action that modifies how the query is parsed. In other words, a POSITA would have known that the operator AND is a search syntax instruction rather than a string separator.¹

111. Conceding that a Boolean operator won’t satisfy the claim elements, Dr. Terveen asserts that it would have been obvious to a POSITA “when implementing Raskin’s teachings to perform the search incrementally to use a single-character string operator” and uses the plus sign and a comma as an example for separating two strings. (EX1003, ¶ 81.) But Dr. Terveen has not identified any mention in Raskin of searching multiple strings; therefore it would not have been obvious in view of Raskin to use a character like a plus sign or a comma and perform a search on two, separate, partial strings. (*See, e.g.*, EX1021, 124-27.) But here, Dr. Terveen has no citation in Raskin which teaches anything like this. According to Raskin, a “string is a sequence of characters; ordinary English words and sentences are examples of strings” and “[s]tring searches look

¹ This understanding is consistent with the ALJ’s holding in the related ITC Investigation, 337-TA-1389, that Boolean operators such as “NOT” are commands (or, instructions for computer action). (EX2008, 31-32.)

through a (usually lengthy) string, called the text, for an instance of a (usually brief) string that the user specifies, called the pattern.” (EX1021, 124.) Dr. Terveen has not cited any examples in Raskin of a “pattern” being two separate strings. Instead, Raskin suggests that if a user continues adding characters to a single string or pattern they will get more accurate results. (*See* EX1021, 124 (searching for the single string “little Tatsu,” the “briefer string *Tatsu*,” and the word “pattern.”); *see also* EX1021, 126 (“The user can then add to the pattern, and the search continues without his losing the search already accomplished on the partial pattern.”).) Thus, Raskin does render obvious to a POSITA using two partial search strings and separating them with a single character.

112. Dr. Terveen alternatively relies on the Lotus Notes Query Builder/Search Builder to meet this claim limitation. Specifically, Dr. Terveen argues that Wilcox-Londergan disclose this limitation by separating “two search strings” with commas in the “Query Builder/Search Builder.” (EX1003, ¶82.) First, as discussed above, Dr. Terveen’s proposed combination of Wilcox and Londergan is improper because these two references are directed to different versions of the software (one with Query Builder and the other with Search Builder) with each have materially different search capabilities. (*See supra* §XI.A.1; *compare* EX1059 at 163 *with* EX1061 at 142.) To the extent Dr. Terveen relies on Search Builder as disclosed in Londergan, this does not disclose this

limitation as Londergan's Search Builder has multiple fields, not a single field, and only a single word or phrase may be entered in each box:



(EX1061, 142 (users should “enter one word or phrase in the numbered boxes in the Search Builder dialog box.”).)

113. Because none of Dr. Terveen's cited evidence discloses the claim limitations receiving a first/second partial search string in the first search field wherein the first and second partial search strings are separated by a string separator character, Dr. Terveen's Ground 1 fails.

2. The Combination Of Wilcox, Londergan, Raskin, And Wu Does Not Teach Incremental Searching A Set Of Documents Using Two Partial Strings (Limitations [1.5]-[1.8], [23.6]-[23.9], and [27.5]-[27.8])

114. As Dr. Terveen treats these elements together in his declaration, I will do the same. *See* EX1003 at ¶¶ 84-93, 114, 119. In my opinion, Dr. Terveen's proposed combination does not disclose these limitations. As an initial matter, as I

discussed above, Dr. Terveen's proposed combination fails to disclose receiving two separate partial strings wherein the two partial strings are separated by a string separator. This alone makes Dr. Terveen's opinions on elements 1.5 to 1.8 wrong.

115. Dr. Terveen is also wrong because the proposed combination of Ground 1 does not disclose incrementally searching a set of documents using two partial search strings. Neither Wilcox nor Londergan disclose receiving two partial search strings in a single search field. Dr. Terveen incorrectly suggests the Lotus Notes Search Bar receives and searches two partial search strings in a single search field. *See* EX1003 at ¶ 81. However, according to Dr. Terveen, the Lotus Notes Search Bar requires the use of the Boolean operator AND to receive two partial search strings, and the operator AND is not a string separator as I described above. Similarly, Dr. Terveen's suggestion that Lotus Notes' Query Bulder and Search Builder discloses searching on two partial strings is incorrect. The Search Builder from Londergan provides users with multiple search fields, each one for a separate word or phrase (in other words, a single string). (*See* EX1061, 142.) This leaves the Lotus Notes Query Builder from Wilcox as the only potential disclosure for Dr. Terveen to use in his combination.

116. To the extent Dr. Terveen is relying on the Lotus Notes Query Builder from Wilcox in his proposed combination, his allegations that it would have been obvious to a POSITA "when implementing Raskin's teachings to

perform the search incrementally to use a single-character string operator” like “a comma, as described by the Lotus Notes references” is wrong. As I described above, a POSITA reading Raskin would not have found it obvious to use a character like a comma and perform a search on two, separate, partial strings because Raskin only teaches searching with a single string. *See, e.g.*, EX1021 at 124-27. The disclosure from Raskin that Dr. Terveen relies on only discloses “incremental search” in the context of the search functionality found in EMACS, a text editor often used with the UNIX operating system. EX1021 at 125. Raskin only discloses searching for a one string “pattern” in the chosen search direction through a single document. *Id.* Given users of EMACS would only be looking to add characters to a single search string to refine the results found in their open document, using a comma as anything other than an additional character of the string, would not be obvious.

117. Additionally, Dr. Terveen’s analysis does not explain how his proposed combination discloses incrementally searching on a two partial strings. Dr. Terveen alleges a POSITA would have understood that implementing Raskin’s incremental search would have involved “treating the search strings as partial strings” because a POSITA would have known that “this is fundamentally how incremental searching works.” EX1003 at ¶87. To the extent a POSITA would have applied this knowledge of how incremental search works, Dr. Terveen’s

analysis fails to indicate that a POSITA would have applied the knowledge to a second partial search string in the same search field. In fact, Dr. Terveen's explanation only describes obtaining results starting with a single character, and "then as the user types additional characters, the search tool narrows the results to only documents containing words that start with those two characters, and so on." *Id.* There is no disclosure directed at using a second partial search string to search incrementally. Dr. Terveen seems to suggest that this would have been obvious to a POSITA but given the Raskin teaching's focus on single string searching, I disagree. Because none of Dr. Terveen's cited evidence discloses the claim limitations *updating the search results as each successive character of the first partial search string and the second partial string is received*, Ground 1 fails.

XII. GROUND 2: THE CHALLENGED CLAIMS ARE NOT OBVIOUS IN VIEW OF SCHWARTZ (EX1004), SEBASTIAN (EX1005), BARBER (EX1006), TRUE (EX1009), AND BAEZA-YATES (EX1023).

A. Dr. Terveen Mischaracterizes the Search in Schwartz, Sebastian, and Barber.

118. Dr. Terveen's Ground 2 obviousness combination is premised on cherry-picking distinct features of different types of unrelated search functionality described in the three Entourage references (Schwartz, Sebastian, and Barber). Dr. Terveen implicitly combines these features into a different, non-existent search system but refers to this system as Entourage. Specifically, Dr. Terveen, without explanation, relies on and implicitly combines two distinct search functionalities

described in the three Entourage references: (1) list filtering; and (2) the Find and Advanced Find Commands.

1. The Entourage References Describe an Email System.

119. The three Entourage References (Schwartz (EX1004), Sebastian (EX1005, and Barber (EX1006)) describe the Entourage for Macintosh 2001 email software. (EX1004, (i), 1; EX1005, (a)-(c), 171; EX1006, (ii), 281.) As Schwartz explains, “Entourage has six tightly integrated components,” an email client, newsgroup reader, address book, calendar, tasks, and notes. (EX1004, 2.) The Entourage references do not describe searching generally for documents on a user’s computer; rather, the search functionality within Entourage allows a user to filter lists (e.g., emails within folders or Address Book contacts) or do more targeted searching for Entourage items (e.g., emails, contacts, notes, etc.). (EX1004, 8; EX1005, 183; EX1006, 384.)

2. Entourage’s List Filtering Function.

120. Schwartz explains that Entourage includes a feature that allows the user to “filter most lists to show items that match simple search criteria (such as the presence of a given text string within an item title).” (EX1004, 8; *see also* EX1005, 183; EX1006, 358, 384.) Filtering is available in Entourage windows displaying lists of Entourage items, such as email messages and address book entries. For example, as shown below, a user can open an Address Book window,

choose an option from the pop-up menu (“Name contains” in the example below), and then start typing a name in the text box:

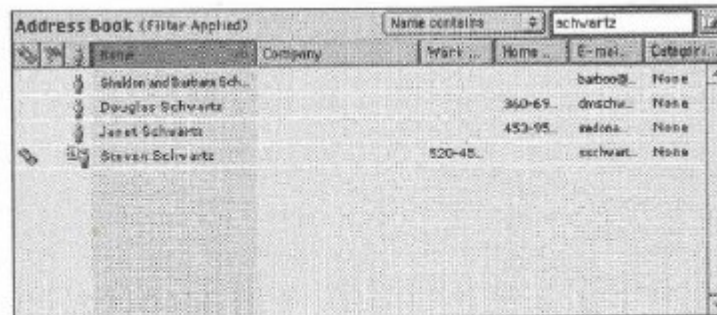


Figure 1.12 As you type the filter text, only matches are shown in the list.

Schwartz Fig. 1.12 (EX1004, 9)

The list is filtered to include only Address Book entries that include the name entered in the text box as the user types. (EX1004, 9.) None of the three Entourage references disclose filtering on more than one string; in all filtering examples in each of the Entourage references, only a single string is entered into the filter text box. (See, e.g., EX1004, 9 (filtering Address Book entries), 99 (filtering the Notes list) 161 (filtering a message list); EX1005, 183 (filtering Address Book entries); EX1006, 384 (filtering Calendar Events).)

3. Entourage’s Find and Advanced Find Commands.

121. Entourage also includes a “slightly more powerful” search function called the Find command. (EX1004, 9; *see also* EX1006, 384-86.) To access the Find command, a user must press Apple-F or select “Find” from the Edit menu in Entourage. (EX1004, 9; EX1006, 384.) This opens a “Find dialog box” with a text

entry field and a pull-down menu with options for selecting which types of Entourage items to search for (e.g., “Messages” in the example shown below). (EX1004, 9; *see also* EX1006, 385.)

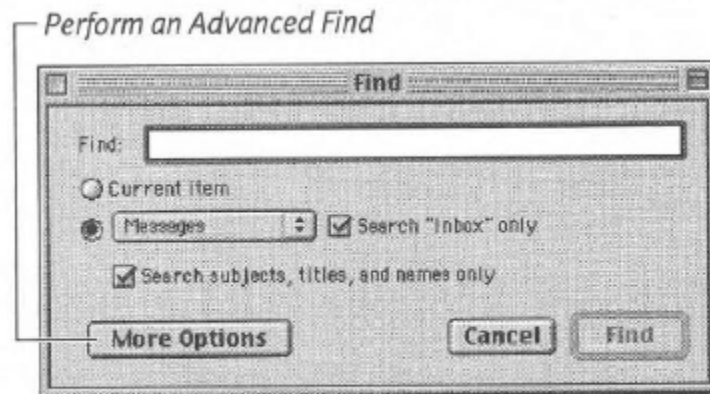


Figure 1.13 You can conduct simple searches by specifying criteria in the Find dialog box.

Schwartz Fig. 1.13 (EX1004, 9)

By pressing “More Options,” a user can access the Advanced Find command, which “allows [a user] to specify multiple search criteria; search multiple item types (messages *and* events, for example); and search any folder, regardless of the one that is currently selected.” (EX1004, 11; *see also* EX1006, 385-86.) The Find and Advanced Find commands require a user to click the “Find” button after entering a search string. (EX1004, 10-11; EX1006, 385.) In other words, the Find and Advanced Find commands perform delimited searches and not incremental searches.

122. After clicking the Find button, “[t]he headers of all matching items are displayed in the Search Results window,” as shown below. (EX1004, 10.) The

Search Results window is a separate window that pops up to display search results.
(EX1004, 10.)

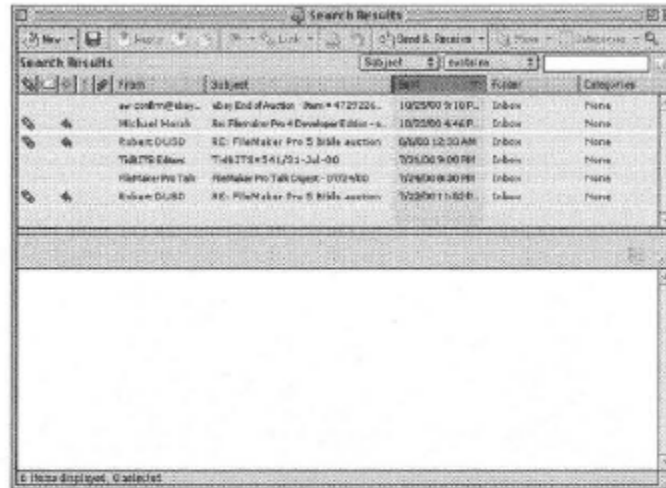


Figure 1.14 When you conduct a Find or Advanced Find, search results are displayed in a separate window. In this example, I searched for messages in the Inbox that contained the word *FileMaker*.

Schwartz Fig. 1.14 (EX1004, 10.)

The Find command can also be used to search for text within a single item in a list (e.g., an email message). (EX1004, 10.) To do so, either “[s]elect the item in the list (in the case of an email or newsgroup message) or open the item in its own window (in the case of a message, task, note, or Calendar event).” (EX1004, 10.) Similar to searching for items within a list or folder, Apple-F or Edit-Find is used to bring up the Find dialog box and then a user can type a string into the text box and click Find. (EX1004, 10.) In this type of search, instead of a Search Results window opening, the window displaying the item (i.e, the newsgroup message in

the example shown below) scrolls to and highlights the first match, if there is one, as shown in the figure below. (EX1004, 11.)

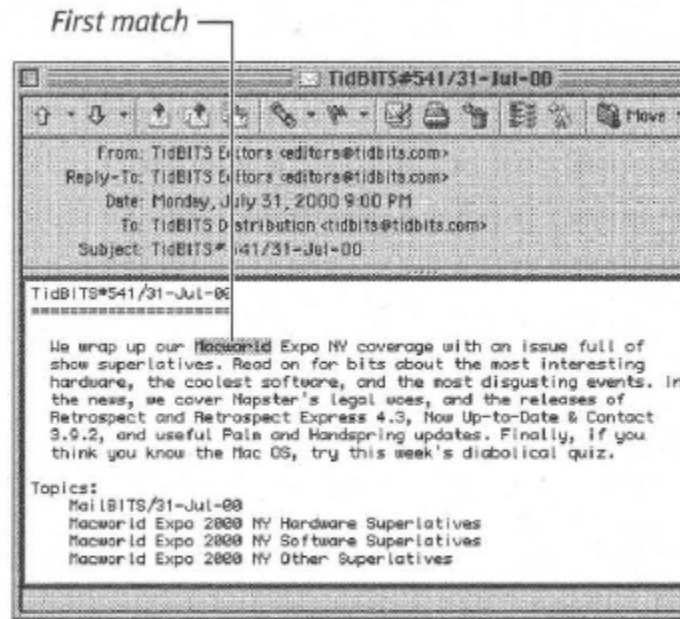


Figure 1.16 When you search for matching text within an item, Entourage highlights the first match, scrolling as necessary to display it.

Schwartz Fig. 1.16 (EX1004, 11)

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

4. Dr. Terveen Mischaracterizes Entourage's Search Functionality.

123. Dr. Terveen mischaracterizes the Entourage references to suggest that the functionalities discussed above from filtering a list, Find and Advanced Find within a group of documents, and Find within a single document can all occur in a

single search conducted in Entourage. The Declaration purports to describe how a user would conduct a search in Entourage:

“Users can quickly locate documents (e.g., emails, news articles, tasks, contacts) containing specific text using an incremental, quick-find search, accessible at the top of a main window.... Results are displayed ‘[a]s you type the filter text....’

(EX1003, ¶126 (showing two screenshots of Entourage’s filter a list tool).)

After Entourage finds a document (e.g., email) containing the search string(s), Entourage scrolls to the portion of the document that contains the search string(s) and highlights the search string(s).”

(EX1003, ¶129 (showing a screen shot of highlighting in a single document from Entourage’s Find command within a single document).)

124. But the capabilities Dr. Terveen cites are from different search tools in Entourage. Dr. Terveen alleges that Entourage has an “incremental, quick find *search*” in its main window. (EX1003, ¶126.) Dr. Terveen’s cited support for this assertion is the Filtering function described above. (EX1003, ¶126; EX1004, 8-9, 39, 89, 161, 204-205, 209; EX1005, 173; EX1006, 358, 384.) Dr. Terveen then provides misleading cites to portions of the Entourage references describing the Find command functionality to argue that “[a]fter Entourage finds a document (e.g. email) containing the search string(s), Entourage scrolls to the portion of the document that contains the search string(s) and highlights the search string(s).”

(EX1003, ¶129.) But the Entourage references cited by Dr. Terveen show no such thing. Indeed, as explained above, this is not how the Entourage references describe the filtering and Find command functionalities.

125. First, Dr. Terveen omits the differences between Filtering and the Find command and implies that the allegedly incremental Filtering function and the delimited Find Command functionality are somehow the same. (EX1003, ¶129.) A POSITA would not have understood these functionalities to be the same. As described above, these are completely different functionalities; specifically, filtering only allows a user to filter things like the Address Book or emails in a single folder on a single text string. (*Supra* §§XII.A.2, XII.A.3.) According to Schwartz, Entourage allows you to “filter the message list” with the filtering functionality described above, or, separately, to “search for messages with the Find command.” (EX1004, 161.) These functions are under separate headings and described separately in Schwartz, and Dr. Terveen fails to explain how a POSITA would understand “filtering” to be “search.”

126. Second, the scrolling and highlighting features that Dr. Terveen relies on are not associated with the incremental Filtering functionality, but only with the Find command. Worse, Entourage only scrolls to and highlights the first search result *when using the Find command to search for text within a single Entourage item*, such as an email or newsgroup message; Entourage does not

scroll to or highlight search results when using the Find command to search for items within a list. (*Supra* §XII.A.3, EX1004, 11, 163.)

127. Furthermore, none of the three Entourage references include any disclosures that would even imply that one could use multiple search strings in either the Filtering function or when using the Find command; every example in each of the three references refers to entering a single string in the text box. (*See, e.g.*, EX1004, 8 (typing in just “apple”), 9 (typing in just “schwartz”), 10 (typing in “macworld”); EX1005, 173; EX1006, 384 (describing filtering messages by typing in “humor”), 385 (typing “Rosebud” into the Find tool), 387 (Fig. 11-7 showing individual strings being typed into the “Advanced Find” tool).)

128. Thus, while Dr. Terveen argues that the Entourage references disclose a search system that performs incremental searches on multiple search strings, and after the system locates a match it will scroll and highlight the search strings, Entourage does no such thing. As Ground 2 is founded on Entourage as the starting system that a POSITA would modify, and Dr. Terveen’s description of its searching capabilities is wholly divorced from what the Entourage references teach, Ground 2 should fail.

B. A POSITA Would Not Have Been Motivated to Combine Entourage with True.

129. Dr. Terveen acknowledges that the Entourage references fail to disclose the *receiving a first partial search string in a first search field; receiving*

a second partial search string in the first search field; and wherein the first and second partial search strings are separated by a string separator character in the first search field limitations and relies on True (EX1009) to fill that gap. Dr.

Terveen also relies on True as allegedly teaching a POSITA to “incrementally locate (i.e., locate as the user types or modifies their search query) documents.”

(EX1003, But Dr. Terveen fails to demonstrate why a POSITA would be motivated to look to True because (1) True fails to search for the presence of any search string in any “documents” as required by the claims, let alone incremental search of documents; and (2) True is directed to a wholly different purpose than Entourage and is inherently imprecise. In addition, Dr. Terveen fails to prove motivation to combine True with the Entourage references because it fails to identify any specific search tool within Entourage that would be the starting point for modification to incorporate the teachings of True.

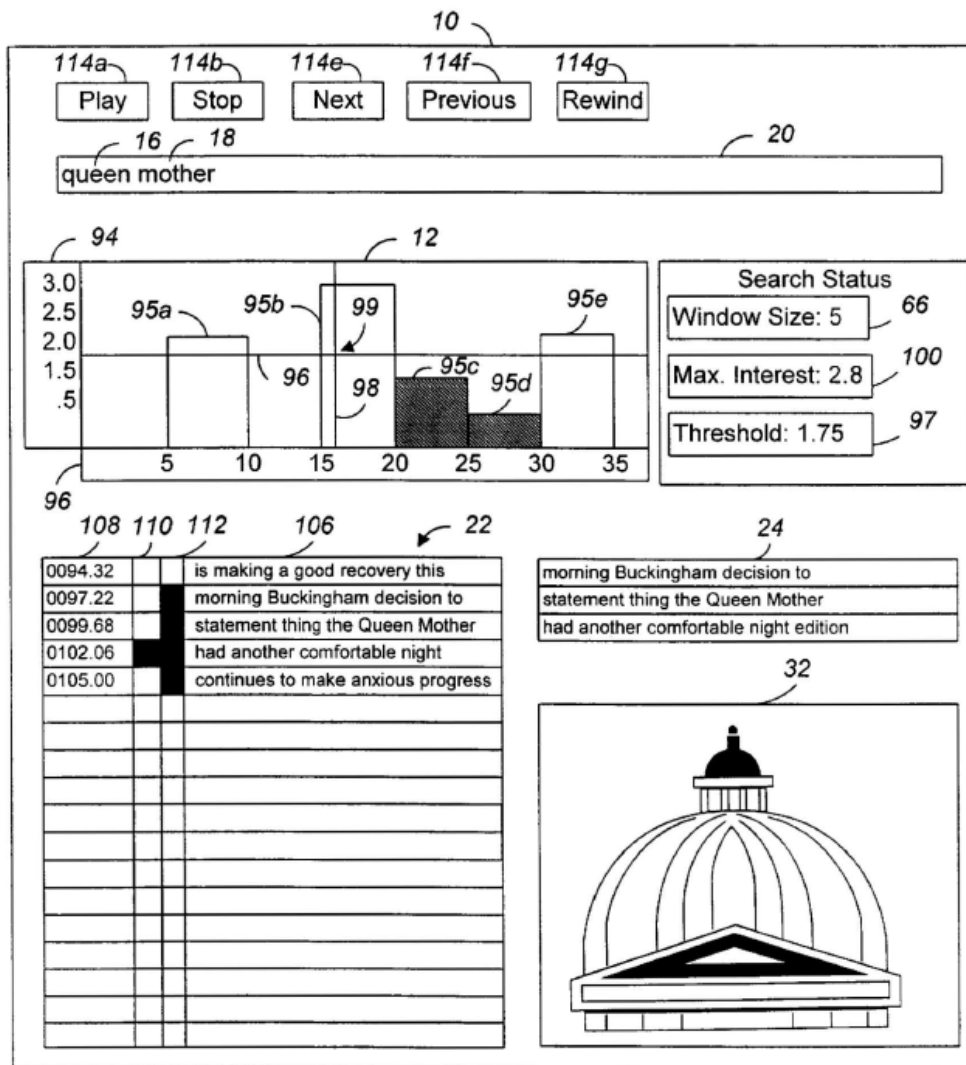
1. True Does Not Teach Incrementally Searching a Set of Documents for Multiple Search Strings.

130. Dr. Terveen’s suggested motivation to combine Entourage with True fails to explain why a POSITA would look to True to improve Entourage. In my opinion, a POSITA would have understood Entourage and True are directed to completely different purposes and a POSITA would not have considered True when seeking to modify Entourage. As I explained above, Entourage is an email client that has multiple search functionalities including a filtering feature and

delimited search features in its Find and Advanced Find features. True is for “interactive searching” of a file “that represents underlying raw information” and finding “parts of the file that represent material of interest in the raw information.” (EX1009, 1:38-42.) True is specifically directed to searching for words within a computer transcription of an audio file, taking into account the confidence level of the accuracy of the speech-to-text transcription for each word in the transcript. (EX1009, 1:55-60, 5:40-64.)

131. The output of True’s “search” method are not a list of traditional search results, but rather a “result window 12 [which] indicates the degree to which portions of the transcript 14 match the query particularly well.” (EX1009, 3:37-40.) Each recognized word in the transcript includes a confidence level, “a number between zero and one indicating an estimated likelihood that the recognized word is correct.” (EX1009, 4:21-33.) To perform a search, the user specifies a window size in seconds to define “transcript windows” which “correspond[] to a segment in the digitized audio” and then enters a query. (EX1009, 4:39-65.) True’s query results consist of a graphical display showing the “interest level” of transcript windows, calculated “as the sum of confidences of each word in the transcript window matching a word in the query.” (EX1009, 5:29-50.) True displays a “bar chart graph of the query results in result window 12,” as shown in True’s Figure 2 below. (EX1009, 5:52-54.) True also lacks any

teaching or disclosure as to how to compute interest levels in light of partial or incremental word matches, an inherently difficult problem.



True Figure 2 (EX1009)

132. Because True’s method calculates the interest level as the sum of confidences for each word, it is possible for transcript windows to meet the threshold for “interest” even if those windows do not include any instance of one of the words in the search query. For example, True explains that “multiple

occurrences of a search word are all given weight in the interest level computation.” (EX1009, 5:40-50.) True’s bar chart in the result window, therefore, provides a visual indication of which transcript windows *might* contain *at least one* of the words of interest entered into the search query but does not provide a list of search results pointing to portions of the transcript that necessarily include all words in the search query.

133. Further, to the extent True describes searching at all, it is searching within a single document, not a set of documents. Dr. Terveen, without any support, contends that True discloses searching for documents because True allegedly teaches that searching within a transcript (i.e., a single document) of an audio file returns as results *portions of the transcript*, referred to as “transcript windows.” (EX1003, ¶138.) In fact, Dr. Terveen equates “documents” with “transcript windows” without providing any citations to intrinsic evidence, and without explanation or justification for his implicit construction. (EX1003, ¶138.) Nor does Dr. Terveen even explain what a “transcript window” is. As explained in above, True’s transcript windows are “portions of the transcript” defined by a window size in seconds (e.g., five seconds). (EX1009, 3:40-43, 4:55-65.) In other words, these transcript windows are not themselves documents, but rather portions of a document. Dr. Terveen does not even attempt to justify his implicit construction of “document” to correspond to only a *portion* of a document.

2. True Is Directed To A Wholly Different Purpose Than Entourage And Its “Interest Level” Method Is Inherently Imprecise.

134. Dr. Terveen’s fails to explain with particularity why a POSITA would be motivated to combine Entourage with True and any such argument is contradicted by the teachings of True:

A POSITA would have been motivated to combine the Entourage references’ teachings to incrementally search at least one search string entered in a single search field ... with True’s express teachings to do so for multiple search strings entered in the search field ... at least because a POSITA knew that such a multi-string search would return more relevant results than a single-string query.”

(EX1003, ¶136 (internal citations omitted).)

135. Dr. Terveen states that the motivation for combining the three Entourage references with True is that “a multi-string search would return more relevant results than a single-string query,” but fails to explain why a POSITA would look to True specifically to improve Entourage in this way. (EX1003, ¶136.) Entourage and True are directed to completely different purposes and POSITA would not have considered True when seeking to modify Entourage as stated.

136. As explained above, Entourage, as described by the three Entourage references (Schwartz (EX1004), Sebastian (EX1005), and Barber (EX1006)), is an

email system that includes an email client, newsgroup reader, address book, calendar, task tracker, and notes. (EX1004, 2; EX1006, 281.) Entourage also includes a Filtering function and a Find command. (*See supra* §§XII.A.2, XII.A.3; *see also* EX1004, 8-14; EX1005, 173; EX1006, 384-387.) True's interest level method has a completely different purpose than the Entourage references.

137. Rather than filtering or searching for email messages, address book contacts, or text within a single message, True provides a “graphical display of query results” that allows a user to have a “visual overview of the extent of which segments of [a] transcript relate to topics of interest.” (EX 1009, 2:10-13; 1:38-42 (stating invention is directed to “interactive searching” of a file “that represents underlying raw information” and finding “parts of the file that represent material of interest in the raw information.”).) In other words, True is directed to searching a single text file, which itself is a computer transcription of an audio transcript produced by a speech recognizer, and then using the search results to produce a graph showing which “parts of the file are of particular interest.” (EX1009, 1:55-60; 3:20-33.) Using this information the user can then read that portion of the transcript or play the underlying audio. (*Id.*, 3:43-46.) True never describes searching multiple files, or any type of file other than a transcript from an audio file. Furthermore, True is directed to solving the problem of how to locate information you are seeking when the underlying data that you are searching (i.e. a

transcript of an audio file created by a speech recognizer) does not accurately represent what was actually said in the audio file. (Id., 1:18-27 (“the searchable information 1 may not always accurately represent portions of the underlying information 5 that the user is trying to find. This is the case, for example, if the searchable information 1 is a text file generated by a speech recognizer from underlying information 5 representing speech, because the words produced by a speech recognizer are sometimes not the words that were spoken.”).) It would not make sense for a POSITA to look to True to modify Entourage simply given the respective corpus of data on which searches are performed.

138. To the extent Dr. Terveen is suggesting that a POSITA would look to True’s interest level method to provide more “precise searches” or to “return more relevant results than a single-string query,” Dr. Terveen is wrong. (See EX1003, ¶136.) A POSITA would have understood that True, by the nature of its invention, is imprecise. As discussed above, True is designed to search a text file that is only a representation of a separate underlying file (e.g. an audio file) and this representation has potentially dubious accuracy. (EX1009, 1:18-27.) The search performed by True does not provide definitive results of individual hits like Entourage and the emails it returns. Instead True provides potential hits with “levels of confidence that the hits are representative of underlying information that is of interest to the user.” (EX1009, 3:19-27.) A POSITA would have understood

the confidence levels to refer to the confidence that the word the user searched in the text file was actually captured correctly by the speech recognizer and actually corresponds to that word in the audio file. (EX1009 at 1:61-63 (“Confidence levels may be descriptive of degrees to which parts of the file correspond to parts of the raw information.”).) Searching on this kind of data is inherently imprecise, especially in comparison to Entourage which searches written text in actual emails and returns results where the query text directly matches the email text. If a POSITA wanted to make the search functionality in Entourage more precise, it would not have made any sense to turn to True’s inherently imprecise interest level method.

3. Dr. Terveen Fails To Explain How A POSITA Would Modify Any Particular Search Tool In Entourage To Include True’s Disclosure.

139. In my opinion, Dr. Terveen does not adequately describe how a POSITA would modify Entourage to incorporate True’s disclosure of allegedly receiving two search strings in its search field. Dr. Terveen again recites that “[a] POSITA would have reasonably expected to succeed in incorporating this functionality from True in the Entourage search tool at least because the Entourage search tool already includes the same or substantially similar functionality and would have required only minor software modifications.” (EX1003, ¶143.) However, Dr. Terveen never describes what specific modifications are made other

than saying Entourage's search field would (hypothetically) receive two partial search strings, leaving room for uncertainty as to how the final result will function.

Id. Further, Dr. Terveen does not account for or otherwise acknowledge the obstacles that need to be overcome for the combination to properly function.

140. To illustrate, Dr. Terveen describes 3 different search approaches for Entourage including the quick-find, Find, and Advanced Find features, with quick-find allegedly being "incremental" and Find and Advanced Find being delimited searches. (EX1003, ¶¶126-27.) However, because quick-find, based on all the examples provided by Dr. Terveen and the examples of quick-find in the Entourage references, only performs filtering on fields like subject lines and names, and not searching on the text of the email, Dr. Terveen's combination must require the use of the Find or Advanced Find features. (*See, e.g.*, EX1004, 8, 9 (results are shown as "you type filter text"), EX1005, 183 (figure showing quick-find as "Address Book (filter applied)"); EX1006, 383-87.) In Entourage's delimited search tools Find and Advanced Find, the results are only displayed after a user selects a "Find" button and "search results are displayed in a separate window." (*See* EX1003, ¶127; EX1004, 10, Fig. 1.14.) Given 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.

141. Further, a POSITA would not even think about using True's incremental search that requires performing a search every time a character is entered given the fact that searching through all files and text in Entourage is known to take a long time even with the Find and Advanced Find tools. (EX1006, 385 ("you can choose All Items—but be prepared to rent a video during the time Entourage will take to search its entire multimegabyte database.") Dr. Terveen's analysis did not include any suggestions why a POSITA would consider True's incremental search in such conditions or any explanation as to how a POSITA would combine the references to avoid such major obstacle.

C. A POSITA Would Not Have Looked to Baeza-Yates to Improve Entourage.

142. Acknowledging that neither the Entourage references nor True disclose *identifying one or more documents included in both the first one or more documents and the second one or more documents*, which Dr. Terveen refers to as an "intersection approach," Dr. Terveen relies on Baeza-Yates to supply this missing limitation. (EX1003, ¶¶138, 155-56.)

143. Dr. Terveen failed to adequately articulate a motivation to combine Entourage with Baeza-Yates and did not offer a specific evidence or rationale to show why a POSITA would have picked the algorithm-based teachings of Baeza-Yates to modify Entourage. As I described above, Entourage is an email system with some calendar, scheduling, and search features. Entourage's search

functionality is provided through the search tool found at the top of the email client. EX1005 at 173. Baeza-Yates is a textbook intended for information retrieval students with an emphasis on retrieving information from the Internet. EX1023 at v (“Information retrieval (IR) has changed considerably in recent years with the expansion of the World Wide Web ...”), 2 (“IR was seen as a narrow area of interest mainly to librarians and information experts ... In the beginning of the 1990s, a single fact changed once and for all these perceptions – the introduction of the World Wide Web.”). According to Baeza-Yates, information retrieval is using “natural language text” to “retrieve all the documents which are relevant to a user query while retrieving as few non-relevant documents as possible.” (EX1023, 2.) Baeza-Yates distinguishes IR from data retrieval, which is “determining which documents of a collection contain the keywords in the user query.” (EX1023, 1.) Thus, IR is directed to finding *relevant* information about a *topic* whereas data retrieval, like that used in the Entourage References search tools, is directed to finding specific instances of a search term in documents like emails, address books, and notes.

144. Dr. Terveen fails to demonstrate why a POSITA seeking to improve a search tool in Entourage would look to Baeza-Yates for how to implement multiple partial string searches incrementally using the “intersection approach.” Dr. Terveen’s only argument of why a POSITA would have been motivated to

combine Baeza-Yates is that the intersection approach allegedly disclosed in Baeza-Yates is “one of only a limited number (a few at most) of known ways to locate documents containing multiple search strings, was simple and easy to implement, and was especially well suited for an incremental search.” (EX1003, ¶138.) But a POSITA would not have looked to Baeza-Yates to improve Entourage in the first place. Baeza-Yates itself makes the distinction between the two references. Entourage engages in “data retrieval” or “determining which documents of a collection contain the keywords in the user query.” EX1023 at 1. A POSITA would understand that data retrieval does not solve the problem of retrieving information about a subject or topic like information retrieval is designed to do. EX1023 at 2. A POSITA seeking to modify or improve a data retrieval system like one of the search tools in Entourage would not be motivated to look to an information retrieval textbook for teachings or suggestions. As Dr. Terveen’s own cited art describes, searching the Internet and performing information retrieval is on a completely different scale and comes with its own challenges when compared to searching a local computer or an email client as taught by Entourage. EX1023 at 373 (“in the Web, all queries must be answered without accessing the text (that is, only indices are available). Otherwise, that would require either storing locally a copy of the Web pages (too expensive) or accessing remote pages through the network at query time (too slow).” *See*

EX1023 at 373. Given the difference in scope and purpose of data retrieval versus information retrieval, it is clear a POSITA would not have been motivated to incorporate features from an Internet-focused information retrieval textbook like Baeza-Yates when looking to modify a data retrieval system like Entourage.

145. Dr. Terveen also does not describe with exactly which Entourage search tool would be the starting point for any modifications. Dr. Terveen suggests, without further explanation, that “[a] POSITA would have reasonably expected to succeed in incorporating this functionality from Baeza-Yates in the Entourage search tool at least because it would have required only simple and straightforward software modifications.” EX1003 at ¶139. However, Dr. Terveen never describes what specific modifications are made or even which “search tool” in Entourage he is referring to. Dr. Terveen only states that Entourage will locate documents “by searching each string separately and then intersecting the results.” (EX1003, ¶138.) Dr. Terveen did not provide any implementation details or suggest which functionalities would be changed in the combination of Entourage and Baeza-Yates.

D. The Prior Art Does Not Disclose All Limitations Of The '139 Patent's Claims

146. Dr. Terveen does not assert that any claims of the '139 Patent are anticipated by any single prior art reference and relies on the combination of the three Entourage References, True, and Baeza-Yates to show that each of the

challenged claims is obvious. However, the combination of these five prior art references fails to disclose certain claim limitations altogether. In particular, Dr. Terveen fails to establish that the combination discloses *incrementally search[ing] a set of documents* [1.5], [23.6], [27.5] and *receiv[e/ing] a second partial search string in the first search field* [1.3], [23.4], [27.3]. Because Dr. Terveen has not shown that all claim limitations are taught by the combination of the Ground 2 references, Ground 2 necessarily fails.

1. The Combination Of Entourage, True, And Baeza-Yates Does Not Teach Incrementally Searching A Set Of Documents (Limitations 1.5, 23.6, and 27.5).

147. Dr. Terveen opines that these limitations are disclosed by his proposed combination of Entourage, True, and Baeza-Yates. (EX1003, ¶¶141-46.) I disagree.

148. Neither the Entourage references, True, nor the proposed combination discloses performing incremental searching on a set of documents. Dr. Terveen appears to rely solely on the Entourage references for this claim limitation. Under the heading “Updating Search Results as Each Character Is Received (Performing An Incremental Search)”, Dr. Terveen states that “The Entourage references teach what was generally well known in the art: incrementally locating and displaying documents (e.g., contacts, emails, news articles, tasks, etc.) containing partial search string(s) as each successive character of the partial search strings is received.” (EX1003, ¶148.) The Entourage references disclose two tools to find

information: (1) filtering lists; and (2) searching for Entourage items or for words within a single Entourage item using the Find or Advance Find commands. (*Supra* §XII.A.3.) As discussed above, the latter functions of Entourage do not search incrementally and instead a user must hit a “search” button to initiate a search. Therefore, the Find and Advanced Find functions of Entourage do not supply these claim elements.

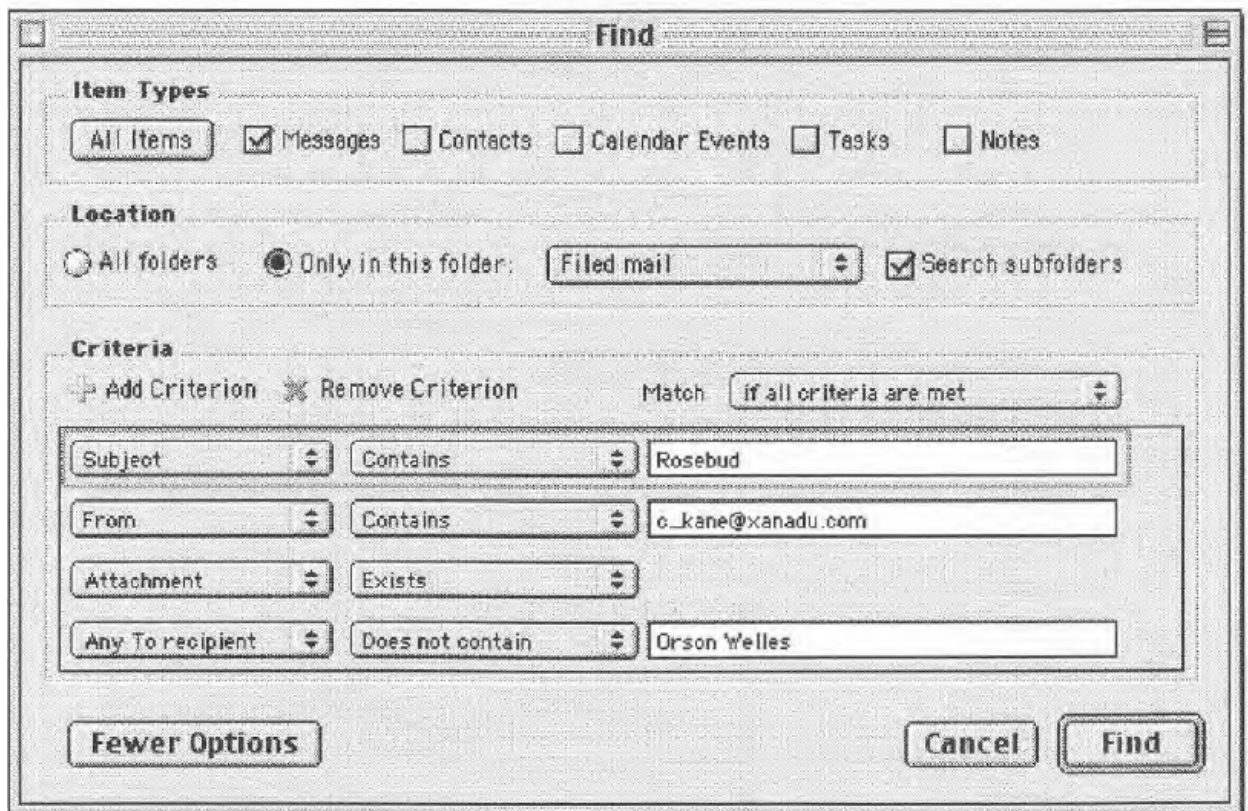
149. While Dr. Terveen does not identify what the primary or “lead” search tool he is using for Ground 2, to the extent Dr. Terveen is implying (without expressly stating) that the filtering function performs incremental *searches* for documents, this is inconsistent with what a POSITA would have understood from the teachings of Entourage. Entourage specifically distinguishes filtering from searching. (*Supra*, §§XII.A.2, XII.A.3.) Because Entourage’s filter a list function is not “searching” as described in the Entourage references, it is not *incrementally searching a set of documents* as required by the claims.

2. The Combination Of Entourage, True, And Baeza-Yates Does Not Disclose Searching On Multiple Partial Strings.

150. Dr. Terveen’s opinion that Entourage, True, or the proposed combination disclose searching multiple strings is wrong. Contrary to how Dr. Terveen portrayed it, Entourage never refers to or describes the use of multiple search strings, only entering a single search string. (*See, e.g.*, EX1004, 8 (only typing in the string “apple”), 9 (only typing in the string “schwartz”), 10 (only

typing in the string “macworld”); EX1005, 173; EX1006, 384 (describing filtering messages by only typing in the string “humor”), 385 (only typing the string “Rosebud” into the Find tool), 387 (Fig. 11-7 showing individual strings being typed into the “Advanced Find” tool).)

151. The only example Dr. Terveen points to as support for its argument that the Entourage references “at least suggest[.]” incrementally searching on multiple strings is an example using the Find command to search on the string “Orson Welles.”



Barber Figure 11-7

(EX1006, 387.) But, as I explained above, the Find command performs only delimited searches, not incremental searches as required by the claims. (*Supra* §XII.A.3; *see* EX1003, ¶142 (citing EX1006, 387 as “at least suggest[ing] incrementally searching for files containing two partial search strings entered in a single search field”).) Further, 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.” Therefore, Dr. Terveen fails to establish that the Entourage references alone disclose this limitation.

152. Dr. Terveen then argues that the references “expressly disclose ‘searching for text strings’” thereby “at least suggesting to a POSITA to receive multiple search strings in the single search field of Entourage’s incremental Quick Find search.” (EX1003, ¶142.) The alleged disclosure of “text strings” that Dr. Terveen cites to is an entry in the index of Schwartz that refers the reader back to pages 10 and 11 of Schwartz, which discusses the filtering a list and Find commands. (EX1004, 267 (under “Find Command”, “searching for text strings”), 269 (under “items”, “searching for text strings in”).) On pages 10 and 11, there are multiple discussions of searching a single text string, but no disclosure of searching multiple text strings. (*Id.*, 10-11 (“To search the current item for a text string... The first instances of the text string (if there is one) is highlighted.”).) A

POSITA would understand that the index in Schwartz was not disclosing searching for multiple separate partial strings and instead was referring to the fact that the referenced pages discussed multiple instances of searching a single search string.

153. To the extent Dr. Terveen suggests it would have been obvious to search for multiple partial strings, I disagree. A POSITA would not have found it obvious to add multiple strings to the filter text in Entourage's filtering method. A POSITA would understand the Entourage filtering method is based on using a single criteria, like different email fields, to sort and display results and using multiple strings would be unnecessary to further narrow the results. Indeed, when some of the filtering criteria, like the "To:" or "From:" fields, only have email addresses that are individual strings, allowing for filtering on multiple strings is unnecessary.

154. To the extent that Dr. Terveen suggests the combination of the Entourage references with True discloses these limitations, I disagree. (*See* EX1003, ¶¶150, 152-53.) Dr. Terveen alleges that a POSITA would have been motivated to incorporate True's "teachings to incrementally locate ... documents containing multiple partial search strings ..." (EX1003, ¶150.) However, True does not disclose searching for documents, nor does it provide "search results." True provides nothing to suggest it operates on multiple documents and provides no

reasons as to why it should. In fact, True explicitly teaches searching a single document as it “features a computer-implemented method that enables a user to control a search of *a file* that represents underlying ray information.” (EX1009, 1:37-43 (emphasis added); *see also* 3:20-27 (“to make it easier for a user to navigate through the results of a search made on a large file ...”); 8:36-12:5 (claims discussing search of a single file).) Searching a single document is not sufficient to meet the ’139 Patent’s claims.

155. True’s output is not a list of documents that contain the partial strings in the inputted query; rather, True provides results in the form of transcript windows that may or may not contain all strings in the query and an “interest level” that indicates the sum of the “confidence levels” for each word in the query. (See EX1009, 3:37-49, 4:34-38.) Therefore, even to the extent True could be understood to describe performing an incremental search for documents (which I do not concede), it does not provide search results that necessarily include all search strings in the query. Thus, the alleged combination of the Entourage references and True, to the extent such a combination is even described in the Petition, does not disclose *receiv[e/ing] a second partial search string in the first search field*.

3. The Ground 2 References in Combination Do Not Teach Two Partial Search Strings Separated By A String Separator Character In The Same Search Field (Limitations [1.4], [23.5], and [27.4])

156. Dr. Terveen has not established that the Ground 2 combination discloses searching two *partial* search strings separated by a string separator character in the same search field.

157. Dr. Terveen relies on Entourage References for the “two partial strings” claim limitation. (EX1003, ¶142.) Specifically, Dr. Terveen asserts that the combination of (1) the list filtering functionality being incremental and therefore inherently partial and (2) the alleged disclosures in Entourage of a plurality of search strings would render this claim element obvious. But filtering a list is distinguished from searching in Entourage, and Entourage does not disclose searching multiple partial search strings in one field. (*Supra*, §§XII.A.2, XII.A.3.) Then, acknowledging that there is no disclosure of a *partial* search string in Entourage, Dr. Terveen vaguely references that this claim element “would have been obvious” to a POSITA “at least because it was within a POSITA’s knowledge to do so.” *Id.* Dr. Terveen cannot rely on the general state of the art without further explanation to satisfy this missing claim element.

4. The Combination Of Entourage, True, And Baeza-Yates Does Not Teach Highlighting A Plurality Of Occurrences Of The First And Second Partial Search Strings In The Selected Document (Claims 5 and 26)

158. Dr. Terveen opines that this limitation is disclosed by his proposed combination of Entourage, True, and Baeza-Yates. (EX1003, ¶¶169-72.) I disagree.

159. As I discussed above, Dr. Terveen's proposed combination does not teach incrementally searching on two partial search strings. (*Supra*, §XII.D.3). Dr. Terveen solely relies on one of the Entourage references for these claims but that reference does not disclose this limitation. (EX1003, ¶171; *see also* EX1004, 10-11 (describing searching the "current item" using the "Find" feature), 162-63 (describing searching the "current message" using the Advanced Find feature).) As I explained above, and as Dr. Terveen admits, the "Find" and "Advanced Find" features of Entourage only practice delimited searches and not incremental searches as required by this claim. (*See* EX1003, ¶127.) And the filtering functionality described in the Entourage references does not highlight the occurrences of the first and second search strings.

160. Given Dr. Terveen did not suggest any other reference in the combination discloses this limitation, it is my opinion that Dr. Terveen's combination does not disclose highlighting a plurality of occurrences of the first and second partial strings in the selected document.

I declare under penalty of perjury under the laws of the United States of America that the foregoing is true and correct.

Executed in Redwood City, CA

Dated: April, 28, 2025

A handwritten signature in black ink, appearing to read "Martin P. Haeberli", written over a horizontal line.

Martin P. Haeberli

PETITIONER’S EXHIBIT LIST

Exhibit No.	Description of Document
1001	U.S. Patent No. 9,633,139 (“139 patent”)
1002	File History of U.S. Patent No. 9,633,139
1003	Declaration of Dr. Loren Terveen, dated December 19, 2024
1004	Steve Schwartz, Visual QuickStart Guide Entourage 2001 for Macintosh, Peachpit Press (2001) (“Entourage-1”)
1005	Bonita Sebastian et al., Microsoft Office 2001 A Hands-On Guide for Macintosh, Computer Literacy Press (excerpts) (2000) (“Entourage-2”)
1006	Nan Barber & David Reynolds, Office 2001 for Macintosh: The Missing Manual, Pogue Press/O’Reilly (1st ed.) (Apr. 2001) (“Entourage-3”)
1007	Eudora Email User Manual for Windows Version 5.0, QUALCOMM (Aug. 2000) (“Eudora”)
1008	Alan R. Neibauer, Running Microsoft Outlook 2000, Microsoft Press (excerpts) (1999) (“Outlook”)
1009	U.S. Patent No. 6,112,172 to True et al. (“True”)
1010	U.S. Patent No. 5,444,823 to Trung D. Nguyen (“Nguyen”)
1011	U.S. Patent No. 7,039,635 to Morgan et al. (“Morgan”)
1012	U.S. Patent Application Publication No. 2002/0055981 to Spaey et al. (“Spaey”)

Exhibit No.	Description of Document
1013	U.S. Patent No. 5,845,278 to Kirsch et al. (“Kirsch”)
1014	U.S. Patent No. 4,255,796 to Gabbe et al. (“Gabbe”)
1015	U.S. Patent No. 5,692,173 to Chee Heng Chew (“Chew”)
1016	U.S. Patent No. 6,049,796 to Siitonen et al. (“Siitonen”)
1017	U.S. Patent No. 5,303,361 to Colwell et al. (“Colwell”)
1018	U.S. Patent No. 6,741,985 to Robin A. R. Green (“Green”)
1019	U.S. Patent No. 5,991,756 to Jiong Wu (“Wu”)
1020	Internation Patent Application Publication No. WO 00/26762 to Mark Richanbach (“Richanbach”)
1021	Jef Raskin, The Humane Interface: New Direction for Designing Interactive Systems, ACM Press (2000) (“Raskin”)
1022	Richard Stallman, GNU Emacs Manual (1st ed.) (Jun. 1985) (“GNU Emacs”)
1023	Ricardo Baeza-Yates & Berthier Ribeiro-Neto, Modern Information Retrieval, ACM Press (1999) (“Baeza-Yates”)

Exhibit No.	Description of Document
1024	Sunny Lam, The Overview of Web Search Engines (Feb. 9, 2001) (“Lam”)
1025	Oscar Vallarino, On The Use Of Bit Maps For Multiple Key Retrieval, Proceedings of the 1976 Conference on Data: Abstraction, Definition And Structure, 108-144 (Mar. 1976) (“Vallarino”)
1026	Steve Lawrence & C. Lee Giles, Context And Page Analysis For Improved Web Search, IEEE Internet Computing, 38-46 (Jul.-Aug. 1998) (“Lawrence”)
1027	Rawley D. Johnson et al., Technical Reference Document Database System (TDOCS) Requirements Definition (Aug. 1993) (“Johnson”)
1028	Lev Finkelstein et al., Placing Search in Context: The Concept Revisited, ACM Transactions on Information Sys., Vol. 20, No. 1, 116-131 (Jan. 2002) (“Finkelstein”)
1029	U.S. Patent No. 6,952,805 to Tafoya et al. (“Tafoya”)
1030	U.S. Patent No. 6,009,459 to Belfiore et al. (“Belfiore”)
1031	U.S. Patent No. 5,220,625 to Hatakeyama et al. (“Hatakeyama”)
1032	U.S. Patent No. 6,185,551 to Birrell et al. (“Birrell”)
1033	U.S. Patent No. 6,839,702 to Patel et al. (“Patel”)
1034	U.S. Patent No. 6,973,450 to Silverbrook et al. (“Silverbrook”)
1035	U.S. Patent No. 6,834,276 to Jensen et al. (“Jensen”)

Exhibit No.	Description of Document
1036	U.S. Patent No. 5,946,678 to Ijsbrand Jan Aalbersberg (“Aalbersberg”)
1037	European Patent Application Publication No. EP 1 209 866 to Baldonado et al. (“Baldonado”)
1038	Declaration of Delainey Tanner, dated November 18, 2024
1039	Kaitlin Duck Sherwood, Overcome Email Overload With Eudora 5, World Wide Webfoot Press (excerpts) (1st ed.) (2001)
1040	Claims Listing
1041	Donald Knuth, The Art Of Computer Programming Vol. 3 Sorting And Searching, Addison Wesley Longman (excerpts) (2nd Ed.) (May 1998)
1042	United States District Courts Judicial Caseload Profile (excerpt)
1043	Steven Schwartz & Robert Correll, Running Microsoft Office 2001 for Mac, Microsoft Press (excerpts) (2001)
1044	Ben Shneiderman, Clarifying Search: A User-Interface Framework For Text Searches, D-Lib Magazine (Jan. 1997)
1045	N/A
1046	N/A
1047	N/A

Exhibit No.	Description of Document
1048	N/A
1049	N/A
1050	N/A
1051	N/A
1052	Ben Shneiderman, Designing The User Interface: Strategies For Effective Human-Computer Interaction, Addison Wesley Longman (excerpts) (3rd ed.) (1998)
1053	Claudine Badue et al., Distributed Query Processing Using Partitioned Inverted Files, Proc. Eighth Symp. on String Processing and Info. Retrieval, IEEE, 10-20 (2001)
1054	U.S. Patent No. 5,685,003 to Peltonen et al. ("Peltonen")
1055	Dick Pountain, The New Penguin Dictionary of Computing, Penguin Group (excerpts) (1st ed.) (2001)
1056	File History of U.S. Patent No. 7,370,035
1057	Alfred & Emily Glossbrenner, Search Engines For The World Wide Web, PeachPit Press (3rd ed.) (2001)
1058	U.S. Patent No. 7,370,035 to Gross et al. ("the '035 patent")
1059	Adam A. Wilcox, PC Learning Labs Teaches Lotus Notes 3.0, Ziff-Davis Press (1993)

Exhibit No.	Description of Document
1060	Declaration of Dr. Sylvia Hall-Ellis Dated December 16, 2024
1061	Stephen Londergan & Pat Freeland, Lotus Notes R5 For Dummies Quick Reference, IDG Books Worldwide, Inc. (1999)
1062	Declaration of Dr. Ingrid Hsieh-Yee Dated December 16, 2024
1063	Exhibit CX-0196 to Complainant's Pre-Trial Brief in the Matter of Certain Computing Devices Utilizing Indexed Search Systems and Components Thereof, ITC-337-TA-1389 (August 29, 2024)
1064	Exhibit CX-0198 to Complainant's Pre-Trial Brief in the Matter of Certain Computing Devices Utilizing Indexed Search Systems and Components Thereof, ITC-337-TA-1389 (August 29, 2024)
1065	Exhibit CX-0199 to Complainant's Pre-Trial Brief in the Matter of Certain Computing Devices Utilizing Indexed Search Systems and Components Thereof, ITC-337-TA-1389 (August 29, 2024)
1066	Exhibit CX-0203 to Complainant's Pre-Trial Brief in the Matter of Certain Computing Devices Utilizing Indexed Search Systems and Components Thereof, ITC-337-TA-1389 (August 29, 2024)
1067	Exhibit CX-0204 to Complainant's Pre-Trial Brief in the Matter of Certain Computing Devices Utilizing Indexed Search Systems and Components Thereof, ITC-337-TA-1389 (August 29, 2024)
1068	Exhibit CX-0206 to Complainant's Pre-Trial Brief in the Matter of Certain Computing Devices Utilizing Indexed Search Systems and Components Thereof, ITC-337-TA-1389 (August 29, 2024)
1069	Exhibit CX-0208 to Complainant's Pre-Trial Brief in the Matter of Certain Computing Devices Utilizing Indexed Search Systems and Components Thereof, ITC-337-TA-1389 (August 29, 2024)
1070	Exhibit CX-0209 to Complainant's Pre-Trial Brief in the Matter of Certain Computing Devices Utilizing Indexed Search Systems and Components Thereof, ITC-337-TA-1389 (August 29, 2024)
1071	Exhibit CX-0211 to Complainant's Pre-Trial Brief in the Matter of Certain Computing Devices Utilizing Indexed Search Systems and Components Thereof, ITC-337-TA-1389 (August 29, 2024)

Exhibit No.	Description of Document
1072	Exhibit CX-0213 to Complainant's Pre-Trial Brief in the Matter of Certain Computing Devices Utilizing Indexed Search Systems and Components Thereof, ITC-337-TA-1389 (August 29, 2024)
1073	Exhibit CX-0214 to Complainant's Pre-Trial Brief in the Matter of Certain Computing Devices Utilizing Indexed Search Systems and Components Thereof, ITC-337-TA-1389 (August 29, 2024)
1074	Exhibit CX-0215 to Complainant's Pre-Trial Brief in the Matter of Certain Computing Devices Utilizing Indexed Search Systems and Components Thereof, ITC-337-TA-1389 (August 29, 2024)
1075	N/A
1076	N/A
1077	U.S. Patent No. 10,552,490 to Gross et al. ("490 patent")
1078	File History of U.S. Patent No. 10,552,490
1079	Transcript, Evidentiary Hearing (Vol. IV) in the Matter of Certain Computing Devices Utilizing Indexed Search Systems and Components Thereof, ITC-337-TA-1389 (excerpts) (October 9, 2024)

PATENT OWNER'S EXHIBIT LIST

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