

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICROSOFT CORPORATION,
Petitioner,

v.

X1 DISCOVERY, INC.,
Patent Owner

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

PATENT OWNER'S SUR-REPLY

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

TABLE OF CONTENTS

	Page
I. Introduction.....	1
II. Claim Construction.....	2
A. “plurality”	3
B. “User Interface”	4
III. Motivation to Combine.....	6
A. Petitioner’s Shifting MTC Fails	6
B. There is No Evidence to Support Petitioner’s Argument That Inter- and Intra-File Search Are Comparable.....	8
C. Petitioner’s Own Prior Art Shows Why a POSITA Would Keep Filtering and Search Separate.....	10
D. Petitioner’s “Obvious to Try” Argument Ignores Deliberate Design Choices.....	12
E. Petitioner’s “Bodily Incorporation” Argument Lacks Merit	13
IV. Reasonable Expectation of Success.....	13
A. Petitioner’s New Theory Contradicts the Petition	14
B. Petitioner’s References Do Not Teach Incremental Multi-File Search	14
C. The Prior Art Index Structures Petitioner Relies Upon Demonstrate That Incremental Two-String Search Was Impractical.....	17
D. Petitioner’s “Lack of Implementation Detail” Argument Is Contradicted by the Intrinsic Record	20
E. Petitioner Failed to Rebut PO’s Evidence That Increased Index Size Increases Search Time.....	20
F. Typical Databases in 2002 Far Exceeded Petitioner’s Assumed Size	22
V. True is Not Analogous Art	24

VI. The Ground 1 References Do Not Teach Elements [1.5], [1.10], And [1.13].....	25
--	----

TABLE OF AUTHORITIES

	Page(s)
Cases	
<i>Axonics, Inc. v. Medtronic, Inc.</i> , 75 F.4th 1374 (Fed. Cir. 2023)	2
<i>Intelligent Bio-Sys., Inc. v. Illumina Cambridge Ltd.</i> , 821 F.3d 1359 (Fed. Cir. 2016)	2
<i>Juniper Networks, Inc. v. Correct Transmission, LLC</i> , 2024 WL 3517862 (Fed. Cir. July 24, 2024).....	14
<i>Phillips v. AWH Corp.</i> , 415 F.3d 1303 (2005).....	5
<i>In re Yamamoto</i> , 740 F.2d 1569 (Fed. Cir. 1984) (Miller, J., dissenting).....	13
Other Authorities	
37 C.F.R. 42.104(b)(4).....	26

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
2011	Information Disclosure Statement for US 12/328,410 dated Dec. 4, 2008 and filing receipt
2012	Information Disclosure Statement for US 11/108,335 dated June 26, 2008

Exhibit No.	Description of Document
2013	Information Disclosure Statement for US 15/199,928 dated June 30, 2016 (now U.S. Patent No. 10,897,443)
2014	Office Action for US 11/128,397 dated May 18, 2007
2015	Office Action for US 11/108,479 dated March 24, 2008
2016	Office Action for US 11/771,554 dated Dec. 23, 2009 (now U.S. Patent No. 8,533,176)
2017	Office Action for US 12/786,246 dated Jan. 10, 2012
2018	Office Action for US 13/918,306 dated June 24, 2015
2019	Information Disclosure Statement for US 15/012,684 dated Feb. 1, 2016 (now U.S. Patent No. 10,546,030) and filing receipt
2020	Supplemental Declaration of Martin P. Haeberli
2021	Transcript from the Deposition of Dr. Loren Terveen, dated October 21, 2025
2022	Dan Cosley et al., <i>SuggestBot: Using Intelligent Task Routing to Help People Find Work in Wikipedia</i> , Proceedings of the 12th International Conference on Intelligent User Interfaces (2007), marked as Exhibit 1 at the Deposition of Dr. Loren Terveen
2023	<i>Reserved</i>
2024	Windows XP Home Edition System Requirements (2001)
2025	Declaration of Natasha E. Daughtrey in Support of Motion for <i>Pro Hac Vice</i> Admission Under 37 C.F.R. § 42.10(c)

Exhibit No.	Description of Document
2026	Transcript from the Deposition of Dr. Loren Terveen, dated March 24, 2026
2027	The Entourage Help Page, Size Limits (2005), http://web.archive.org/web/20050407071330/http://www.entourage.mvps.org/database/size.html (Introduced as Exhibit 1 at Deposition of Dr. Terveen, March 24, 2026)

I. Introduction

Patent Owner X1 was founded in 2003 to develop, market, and sell software that makes it faster and easier for users to search their files. With approximately 20 employees, X1 has spent more than two decades building a successful business in a software industry dominated by large incumbents like Petitioner Microsoft. The '490 patent reflects core technology that is embedded in each of X1's products and which X1 has relied on to compete in the market; indeed, X1 asserts that it developed and distributed the first system that incrementally searches across multiple files. X1 is precisely the type of innovator the patent system is designed to protect.

Petitioner's Reply (Paper 28; "Reply") underscores its failure to articulate a consistent and legally sufficient theory of obviousness in the Petition. Faced with the deficiencies identified in the Patent Owner Response (Paper 21; "POR"), Petitioner shifts positions, raises new claim constructions for terms it failed to address in its Petition, and reframes the prior art in ways untethered to the Petition or the claim language.

Having failed to identify art that discloses all elements of the challenged claims, Petitioner resorts to using four and five reference combinations for the broadest independent claims, purporting to extract high-level "ideas" (e.g., incremental search of text within a single file) from references in entirely different

contexts, ignoring how those systems actually function and offering no coherent explanation for how they would be combined to perform the claimed invention. Petitioner also improperly resorts to knowledge of a POSITA to fill gaps in the prior art for several dependent claim limitations

These shifting and internally inconsistent positions are not harmless; they underscore that Petitioner did not in the Petition, and still cannot, articulate a consistent theory of how the prior art renders obvious incremental (as the user types), multi-string search of files. The Board should reject this moving-target approach and evaluate Petitioner's arguments for invalidity based on the Petition as filed, from the view point of a POSITA who would evaluate the prior art in context, which fails to meet Petitioner's burden.

II. Claim Construction

Petitioner advances for the first time in its Reply constructions for “plurality” and “user interface.” (Reply, 3-8, 24.) This is improper. “[A] reply or sur-reply that raises a new issue or belatedly presents evidence may not be considered.” USPTO Consolidated Trial Practice Guide, 73. *Intelligent Bio-Sys., Inc. v. Illumina Cambridge Ltd.*, 821 F.3d 1359, 1369-70 (Fed. Cir. 2016). Even Petitioner's case (*Axonics, Inc. v. Medtronic, Inc.*, 75 F.4th 1374, 1380 (Fed. Cir. 2023) (citations omitted)), explains that “the petitioner's contentions ... define the scope of the litigation all the way from institution through to conclusion” and that

“a petitioner may not submit new evidence or argument in reply that it could have presented earlier.” X1 respectfully requests that the Board give no weight to Petitioner’s new constructions or any arguments that rely on them, and instead evaluate the Petition’s arguments, which did not identify teachings in the prior art consistent with these constructions.

A. “plurality”

The Reply discloses an affirmative construction of “plurality,” asserting the claims “recite searching across ‘a plurality’ of files (i.e., two or more), not a large volume,” and “[t]he plain and ordinary meaning of a ‘plurality’ is two or more.” (Reply, 4.) The Board should reject this claim construction theory because it is untimely. Petitioner does not directly justify its belated claim construction, but does contend that its reliance on new prior art teachings is proper because it “directly rebut[s] an allegation in the POR that is based on the unforeseeable ‘large volume’ implicit claim construction argument.” (Reply, 5-6 n.2.)

There is nothing unforeseeable about PO’s arguments. (Reply, 5-6 n.2.) The POR establishes a POSITA would not have been motivated to combine Petitioner’s asserted references because doing so would have required implementing incremental search across large file sets explicitly disclosed in *Petitioner’s primary references*, Lotus Notes and Entourage. (See, e.g., POR, 19-28, 57-59, 62-63.) A POSITA in 2002 knew that consumer computers lacked the memory and

processing power to implement incremental search across any appreciable number of files, undermining Petitioner's only stated motivation (speed and efficiency). (*See, e.g.*, POR, 23-25.) This argument is a direct rebuttal to Petitioner's motivation and reasonable expectation of success arguments and does not rely on any "implicit" construction of "plurality."

Further, PO's arguments were entirely predictable given Petitioner chose to advance obviousness theories based on systems intended for large numbers of files. (*See, e.g.*, EX1059, 85 ("add a new discussion database to your workspace and it has 1,000 documents in it."); EX1006, 384 ("After a short time...you'll collect a lot of email messages, contacts, and other Entourage items. Trying to find a particular morsel of information just by browsing becomes impractical."), 385 ("multimegabyte database"), 387 ("hundreds or thousands of messages in your message list").) Petitioner chose to base its obviousness theories on searching across large volumes of files, not "two or more." Petitioner's new claim construction is not responsive to any "unforeseeable" argument and should be disregarded.

B. "User Interface"

The panel should also reject Petitioner's untimely and unsupported claim construction argument for "user interface." The Reply construes "user interface" to include multiple windows, citing testimony from Mr. Haeberli in response to a

generic question about user interfaces, outside the context of the '490 patent.

(Reply, 24.) This construction is incorrect because it contradicts the specification, which “is the single best guide to the meaning of a disputed term,” and “[u]sually, it is dispositive.” *Phillips v. AWH Corp.*, 415 F.3d 1303, 1315 (2005) (internal quotation marks omitted).

For example, claim 10 requires a portion of the identified file be displayed “in the interactive user interface,” i.e., in the same window as the search field. As discussed in the POR, the '490 patent describes this interactive user interface as a unified user interface in which “the search results are displayed in the list pane or area, and the view area or pane displays the contents of the first item or a selected item from the list area,” such that a portion of the identified file is automatically displayed in the view pane upon selection, without any need to open the file. (POR, 32-35; EX1001, 7:1-11.) The specification here is dispositive. The term “user interface,” as used in the claims, does not encompass multiple windows.

The POR explains that the Lotus Notes references both require the user to open a file before any portion is displayed or highlighted. (POR, 34; EX1059, 164; EX1061, 145-46; EX2021, 66:4-17, 71:7-73:8.) Similarly, the Entourage references only highlight search strings during a search of an open file. The highlighting is not shown in a portion of the file displayed via the user interface. (POR, 66; EX1004, 11.) This is not displaying or highlighting in the “user

interface” as contemplated by the claims. Petitioner’s untimely and incorrect claim construction should be rejected.

III. Motivation to Combine

A. Petitioner’s Shifting MTC Fails

In the Reply, Petitioner improperly reframes its MTC in a way that is divorced from its Petition. The Petition’s entire obviousness argument has a POSITA apply incremental search to systems (Lotus Notes/Entourage) that have large numbers of files, arguing a POSITA would have been motivated to do so for reasons of “speed, effectiveness, and ease of use.” (*See* Petition, 19 (Lotus Notes); *see also id.*, 67 (Entourage).) Petitioner’s new framing of the claims as requiring “searching” across only “a plurality” of files (meaning two or more) ignores the Petition’s stated motivation, as there can be no motivation to solve a speed and responsiveness problem that does not exist for a database of just two files. (*See* EX1001, 1:32-2:5 (describing problem of “slow and cumbersome” delimited searches across multiple files); EX1021, 125 (“Incremental searching wastes less of a user’s time” than delimited search); EX1044, 3 (“the advantages are great: users can broaden, narrow, or refocus their search several times in as many

seconds”).¹ Petitioner argues “a POSITA would have reasonably expected a 2002-era computer to be able to perform an incremental search for two search strings across a plurality of files, especially when searching across a smaller number (but still a plurality) of files.” (Reply, 5-6.) But that cuts against motivation, not in favor of it. In short, a POSITA would have no reason to develop and implement an incremental search system if the search is only across two files. (EX2020, ¶¶51, 63-71.) Petitioner has not identified a problem that a two-file incremental search would solve, and without such a problem, there is no motivation to combine the references in either Ground.

The only context in which a motivation to combine could plausibly exist is searching across a larger corpus of files. Petitioner offers its two-file framing

¹ Petitioner also advances an entirely new MTC and reasonable expectation of success argument: that a POSITA would have been motivated to combine Lotus Notes with Raskin and Entourage with True “at least because 2002-era computers could provide the response times needed to perform an incremental search across a plurality of files.” (Reply, 5.) While Petitioner may respond to PO’s feasibility argument in the POR, Petitioner’s attempt to reframe its response as a new, independent motivation to combine with a reasonable expectation of success should be disregarded.

precisely to sidestep the feasibility concerns raised by Mr. Haeberli. (EX2020, ¶¶47-62, 84-88.) That retreat is telling. As explained in the POR, the unresolved technical challenges evident in the prior art, including index size, RAM constraints, and resulting performance degradation, would deprive a POSITA of any reasonable expectation of success in implementing incremental search across a non-trivial file set. Petitioner has not shown the proposed combinations would successfully address any of these challenges. (EX2020, ¶¶47-62.) Petitioner's argument fails from a contradiction of its own making: a POSITA would have no motivation to combine in the two-file scenario, but the large-corpus scenario (i.e., the only one that could supply motivation) lacks any reasonable expectation of success. (*See infra* §§IV.C, IV.F.)

B. There is No Evidence to Support Petitioner's Argument That Inter- and Intra-File Search Are Comparable

Because Petitioner lacks any prior art teachings of incremental search across files², it relies on a single paragraph in Dr. Terveen's declaration to argue a POSITA would not have technical difficulty jumping from intra-file search to inter-file search. Dr. Terveen's claim that the only technical difference between these types of search is "keep[ing] track of the files that contain the search string"

² None of the references in Petitioner's string cite discloses incrementally searching across multiple documents. (Reply, 7; *see infra* §IV.B.)

grossly oversimplifies the technical realities of multi-file search and ignores the real implementation burdens that arise once a system must search for a group of files rather than text already present in a single, open file. (*See* EX2020, ¶¶15, 35-36, 38-62; EX1100, ¶11.)

As explained in the POR, searching across a set of files to find specific files requires the system to identify which files are in the set, access those files (which are often not loaded in active memory), search their contents, determine which files contain matches to search criteria, and then present results to the user; all tasks that do not exist in the same way (or at all) for a search confined to text in a single open file already loaded into memory. (*See* EX2020, ¶¶15, 35-36, 38-62.) PO previously explained the many challenges involved in implementing incremental search across a large corpus of files. (POR, 22-25, 62-63; *see also* EX2020, ¶¶15-16, 35-36, 38-49, 51-52.) These are not trivial engineering details; they represent materially different architectural and performance challenges. (*Id.*)

A POSITA would therefore understand that adapting a single-file incremental search to operate across a collection of files is not a trivial extension requiring only a bookkeeping adjustment. (*See id.*) Rather, it requires solving distinct and non-trivial engineering problems, including how to access file contents efficiently, how to manage memory and resource load, and how to return results

with acceptable speed, problems Dr. Terveen's characterization entirely ignores.
(EX2020, ¶¶47-52; EX1100, ¶11.)

C. Petitioner's Own Prior Art Shows Why a POSITA Would Keep Filtering and Search Separate

That the developers of Entourage kept filtering and search as distinct tools serving different purposes is real world evidence contradicting Petitioner's argument that a POSITA would have combined them. The Entourage references describe three distinct search functionalities Petitioner implicitly combines in its obviousness argument: (1) list filtering; (2) Find; and (3) Advanced Find. (EX1004, 8-12; EX1006, 384-87; POR, 38-49.) Schwartz explains that Entourage allows a user to "filter the message list" with the filtering functionality, or, separately, to "search for messages with the Find command." (EX1004, 161-62.) The Find command is a separate, non-incremental search. (*Id.*) These functionalities represent different tools designed for different use cases, and Petitioner fails to explain why a POSITA would choose any given feature from one of these three search tools and combine it with another to arrive at the claimed invention. (Reply, 18-19.)

The fact that the developers of Entourage chose to maintain incremental list filtering and delimited Find/Advanced Find as distinct tools would suggest to a POSITA that doing so was for good reason, such as technical constraints or

usability considerations that make combination undesirable or impractical. (POR, 45-49; EX2020, ¶74.) In Entourage, the separation appears driven by performance concerns: the Find and Advanced Find features are described as being slower, particularly when searching across broader file sets. (*See* EX1006, 384.)

Maintaining incremental list filtering as a separate, lightweight feature allowed users to quickly narrow down items within a narrow, structured list, such as names or email addresses in an address book, without triggering the more resource-intensive search operations required by Find and Advanced Find. (POR, 45-49; EX2020, ¶105.) Combining these approaches would have required either maintaining substantially larger indexes or performing real-time sequential scanning, either of which would exacerbate the existing performance problems and degrade the user experience. (*Id.*)

Petitioner has provided no explanation for why a POSITA would have overridden these deliberate design choices, arguing only that the references provide “side-by-side disclosures” of the same product’s features. (Reply, 19-20.) This misses the point. The fact that the Entourage references disclose these functionalities side-by-side only underscores that the developers of Entourage were aware of both incremental list filtering and delimited Find and Advanced Find tools and still deliberately chose to keep them separate. (EX1006, 384-87; EX2020, ¶¶74-75.) Petitioner does not explain why a POSITA would do the

opposite, particularly when the separation was likely driven by performance considerations. (EX2020, ¶¶74-75.)

D. Petitioner’s “Obvious to Try” Argument Ignores Deliberate Design Choices

Petitioner’s argument that combining the functionalities of Lotus Notes’ different search tools was obvious to try because they represented “a limited number of well-known implementation options, each with its own recognized advantages” suffers from the same deficiency as Petitioner’s combination of the Entourage search tools. (Reply, 17-18.) The developers of Lotus Notes were aware of these different search functionalities and deliberately chose to keep them separate. As explained in the POR, the 1993 Lotus Notes included the Query Builder tool, which Petitioner alleges allowed multi-string search in a single field, but the 1999 R5 version eliminated Query Builder and replaced it with the Search Builder, which accepts only a single search string per field. (POR, 10-11; EX1059, 162-64; EX1061, 141-42.) The developers did not simply maintain parallel tools; they affirmatively abandoned the multi-string, single-field approach in favor of a different design. Petitioner does not explain why a POSITA would combine functionalities that the developers who created them specifically chose not to combine, and indeed moved away from.

E. Petitioner’s “Bodily Incorporation” Argument Lacks Merit

Petitioner’s invocation of the bodily incorporation doctrine is a red herring that mischaracterizes PO’s argument. PO never argued, as Petitioner contends, that bodily incorporation of Raskin’s entire alleged architecture into the Lotus Notes system is required for feasibility. (Reply, 8-9.) Rather, PO noted, as confirmed by Dr. Terveen’s own testimony, that Petitioner simply took from Raskin the abstract “idea of incremental search,” without considering the context of the disclosure. (POR, 19-26.) Raskin’s disclosure of incremental search implements a sequential scanning tool operating within a single open file while Lotus Notes uses an indexed search system to search over a large corpus of files. (POR, 20-22.) Suggesting that Petitioner explain why a POSITA would look to Raskin’s vastly different system to modify Lotus Notes is not a matter of “bodily incorporation”; rather, the issue here is “the operation of the [systems] is so different that one of ordinary skill in the art would not view this as a natural and logical combination.” *In re Yamamoto*, 740 F.2d 1569, 1573 (Fed. Cir. 1984) (Miller, J., dissenting) (citing *In re Walker*, 374 F.2d 908, 913 (1967)).

IV. Reasonable Expectation of Success

Faced with PO’s evidence that a POSITA would not have had a reasonable expectation of success (“RES”) in implementing the Petitioner’s combinations, Petitioner falls back to *KEYnetik* for the alleged proposition that if an expert

testifies the software modifications are simple and straightforward, that is sufficient to establish RES. (Reply, 12-13.) As explained in the POR (POR, 26), this conclusory statement is insufficient when rebutted with competent evidence to the contrary, as PO has done. *Juniper Networks, Inc. v. Correct Transmission, LLC*, 2024 WL 3517862, at *4 (Fed. Cir. July 24, 2024).

A. Petitioner’s New Theory Contradicts the Petition

In a footnote, Petitioner asserts “a system that practices the claims does not return results until two search strings are entered” and thus, “the computer’s response time is less important than in an incremental search where results are returned as a user enters the first string.” (Reply, 5 n.1.) This is a complete departure from the Petition, which consistently relies on the idea that the claims require character-by-character incremental search. (*See*, e.g., Petition, 9 (claims require “incremental (i.e., character-by-character) searching”), 67 (“POSITA would have been motivated to incorporate True’s teachings to incrementally search (i.e., search as the user types or modifies their search query) files”).) This new and plainly contradictory theory should be disregarded.

B. Petitioner’s References Do Not Teach Incremental Multi-File Search

Because none of Petitioner’s primary references discloses incremental search on multiple strings across a corpus of files, Petitioner instead argues that multi-file incremental search was well known in the art. (Petition, 6-7; Reply 4, 7-

8; EX1100, ¶20.) As explained in the POR, none of these references disclose incremental multi-file search. (POR, 40-49, 53-55, 59-60; EX2001, ¶39-47, 134-143; EX2020, ¶¶10-25.) At most, Petitioner's references describe incremental filtering or searching on a single string within a single file or a structured list of items.

Recognizing the shortcomings of its cited art, Petitioner, for the first time in its Reply, argues that Raskin teaches incremental search across multiple files. (Reply, 6.) This argument is untimely and should be rejected. Regardless, Petitioner mischaracterizes Raskin's disclosure, citing a portion of Raskin that describes an "expanding sweep" approach to incremental search in which, after a local file is fully searched, the search proceeds, file-by-file, through each of the following files in the same folder, and then continues to search file-by-file starting from the beginning of the first file in the folder. (EX1021, 132) After cycling through the folder, the next larger domain is searched, and so forth. (*Id.*) Raskin criticizes this method of searching across files, conceding its inefficiency: "[g]enerally, people will use more efficient strategies than simply relying on this expanding sweep of the hierarchy." (*Id.*) Instead, Raskin recommends manually identifying and selecting a specific file first. (*Id.*) In other words, Raskin recommends doing the *opposite* of what is claimed.

This is significant evidence of why the claimed inventions are not obvious.

Raskin does not present multi-file incremental search as a viable or preferred mechanism; it is presented as an approach that a POSITA would naturally avoid in favor of a more targeted multi-step approach premised on narrowing the search domain to a single file before incremental search begins. Far from teaching incremental search across a corpus of files, Raskin acknowledges that the multi-file expanding search is inefficient and that a POSITA would choose to constrain incremental search to a single, pre-selected file.³ (EX1021, 132.)

³ Raskin also describes a preferred version of incremental search implemented using LEAP, which was not considered by Dr. Terveen in his analysis and is not cited by Petitioner. (EX2026, 218:9-14.) LEAP employs cursor-based sequential scanning similar to that employed by EMACS. (EX1021, 135-139.) Notably, Raskin criticizes the use of dialog boxes (like those in Petitioner's key prior art) to initiate searches, observing that conventional searches are "initiated modally, by means of a dialog box," and contrasting that approach with LEAP, which "is modeless." (EX1021, 129.)

C. The Prior Art Index Structures Petitioner Relies Upon Demonstrate That Incremental Two-String Search Was Impractical

Petitioner argues that a POSITA would have had a reasonable expectation of success when searching across a larger set of files using known index structures. (Reply, 5-6; EX1100, ¶¶15-17.) But Petitioner’s cited references undermine rather than support Petitioner’s argument.

For example, Petitioner alleges Green teaches an index structure having a “search algorithm associated with a binary tree [that] very quickly reduces the number of lexical nodes that must be compared with a search string.” (EX1100, ¶17 (citing EX1018, 1:40-47).) But Green’s own description of the same index structure acknowledges that it “severely impairs retrieval time,” indicating it would not be able to be done as fast as the user types. (EX1018, 1:47-49.) Petitioner further argues Baeza-Yates discloses an index structure built on 250 MB of text that can search a simple word in 80 milliseconds. (EX1100, ¶21.) But Baeza-Yates also notes that a search of two to five words would take 250 to 350 milliseconds. (EX1023, 196.) Therefore, at best, the index structure provides for a *delimited* (not incremental) two-word search that would hit, if not exceed, the theoretical 250-millisecond performance ceiling that Petitioner and Dr. Terveen adopt from Raskin for incremental search. (Reply, 5-6; EX1100, ¶21.)

Dr. Terveen's suggestion that Baeza-Yates teaches a whole-word index structure that could be used for incremental search without increasing index size is also misplaced. (EX2026, 210:10-17.) Baeza-Yates describes searching for prefixes (the beginnings of words) using a whole-word index. (EX1023, 195; EX1003, ¶111.) A prefix query against a whole-word index is not the same as an incremental search and does not resolve to a single index entry; rather, each matching vocabulary term carries its own separate occurrence list, all of which must be retrieved and merged before results are returned. (EX1023, 195.) Baeza-Yates notes "the most time-demanding operation on inverted indices is the merging or intersection of the lists of occurrences," (EX1023, 196), an overhead that scales with the number of terms matching the prefix and would need to be repeated with each character the user types for an incremental search. Thus, simply being able to perform a prefix search against a whole-word index does not establish that such a search could be performed at the speeds required for incremental search.

The weight of the evidence, including testimony from both parties' experts, is that an incremental search would increase index size beyond that required for a standard delimited search index, making it wholly implausible that an incremental search across two strings could be performed in under 250 milliseconds when a delimited two-word search on a smaller index already cannot. (Reply, 7; EX1100, ¶¶19, 51-52.) Petitioner's art fails to support its RES arguments.

Lotus Notes, Petitioner's primary reference for Ground 1, further rebuts any expectation of success arguments based on use of an index. Lotus Notes indicates that its index for *delimited* search constitutes at least 50% of the size of the database. (EX1061, 136.) As such, a 250MB database, using the example from Baeza-Yates, would require approximately 125MB of index storage alone, consuming virtually the entire 128MB of RAM capacity available on a 2002-era computer. As Mr. Haeberli explained, general performance degradation across the entire computer system occurs once approximately 50% or more of available RAM is consumed. (EX2020, ¶¶48-49.) An index consuming 125MB of 128MB of available RAM, approximately 98%, far exceeds this threshold. Moreover, this 125MB figure reflects only the index required for delimited search; as the weight of the evidence shows, incremental searching could increase index size. (Reply, 7; EX1100, ¶¶19, 51-52; EX2020, ¶¶32-33, 53-54, 87.) A POSITA would thus have understood that if a delimited search index alone consumes virtually all available RAM on a 2002-era computer, degrading system performance beyond acceptable levels, a larger index required for incremental search would only compound that degradation.

Petitioner's references collectively demonstrate that performing an incremental two-string search within acceptable response times on 2002-era

hardware using known index structures was not simple and straightforward; it was computationally impractical or impossible.

D. Petitioner’s “Lack of Implementation Detail” Argument Is Contradicted by the Intrinsic Record

Petitioner incorrectly argues that the ’490 patent’s alleged lack of detailed explanation for implementing the claimed search functionality itself confirms that such implementation was within the skill of a POSITA. (Reply, 13-14.) Dr. Terveen’s bare assertion that he “see[s] nothing in the challenged patent that describes how to scale up its incremental search” does not make it so, and Petitioner offers no independent analysis beyond that conclusory statement. (EX1100, ¶51.) The ’490 patent addresses implementation, providing structural and operational detail regarding how the claimed incremental search is performed, including the architecture of the words file and the manner in which it is searched. (*E.g.*, EX1001, 5:37-62, 7:1-8:40.) Petitioner is simply wrong that the patent is silent on implementation.

E. Petitioner Failed to Rebut PO’s Evidence That Increased Index Size Increases Search Time

Petitioner’s argument that PO’s concern about a larger index “misses the point” because “[e]ven if the index’s size increases, an index generally still increases the speed of the search because it contains less data than the underlying files” sidesteps the actual issue. (Reply, 7.) The concern is not whether an index is

faster than searching raw files, but whether an index of the size required for incremental search across a corpus of files to find groups of files would be fast enough to meet the response time demands of the claimed incremental search. Even if Petitioner could prove it would be obvious to use an index that improves speed over brute search of the full files, but still returns results in seconds rather than milliseconds, this still does not render obvious the claimed invention. Petitioner's argument that an index "generally" increases speed leaves unrebutted PO's evidence that the increased index size required for incremental search as disclosed in Petitioner's prior art renders the system too slow to meet the claim limitations, not merely slower than it otherwise could be.

Dr. Terveen contradicts this very argument: "This would result in an even larger index ... and there certainly is no explanation as to how that index would provide results quick enough to perform an incremental search when searching across a large volume of files." (EX1100, ¶52.) Dr. Terveen thus acknowledges that a larger index does not automatically translate to results that are "quick enough" for incremental search at scale. This is the precise issue that Petitioner has failed to address in the Petition or Reply. Petitioner cannot simultaneously argue that index size is irrelevant to incremental search speed while its own expert concedes that a larger index will slow processing such that any search may not be "incremental" at all.

F. Typical Databases in 2002 Far Exceeded Petitioner's Assumed Size

Petitioner's contention that Mr. Haeberli's analysis regarding the RAM demands of an incremental search index "assumes an unrealistically large database of one gigabyte" (Reply, 7) is contradicted by both the hardware capabilities of 2002-era computers and the embodiments described in the '490 patent. (EX2020, ¶48, EX1001, 5:37-52.) Mr. Haeberli's analysis assumes a 1 GB database based on the performance capabilities of a 2002-era computer. (EX2020, ¶48.) A 1 GB database thus occupies only 67% of the available file storage space, well within the system's capacity. Applying the 5% index-to-database ratio from Baeza-Yates, as adopted by both Mr. Haeberli and Dr. Terveen, the corresponding index for a 1 GB database would be 50 MB, consuming nearly half of the system's total RAM (EX2020, ¶48; EX1100, ¶14.) As Mr. Haeberli explains, this level of RAM consumption would cause general performance degradation across the entire computer system. (EX2020, ¶49.) The '490 patent independently confirms the reasonableness of Mr. Haeberli's 1 GB assumption. The patent describes one embodiment in which an index is constructed in 4-16 MB bulks, each corresponding to one of five different target types. (EX1001, 5:37-52.) If the index is constructed using the smaller 4 MB bulks, the resulting index would be 20 MB, corresponding to a 400 MB database under the Baeza-Yates ratio. If the larger 16 MB bulks are used, the resulting index would be 80 MB, corresponding to a 1.6

GB database. The midpoint of this database range is 1 GB, precisely the database size Mr. Haeberli assumed. Far from being “unrealistically large,” Mr. Haeberli’s 1 GB assumption reflects the database scale that the ’490 patent itself contemplates when addressing the limitations of 2002-era computers.

Dr. Terveen’s reliance on EX1061, EX1095, and EX1096 to establish a 2-10 MB benchmark for “then-existing databases” is misplaced. (EX1100, ¶13.) Each of those exhibits is concerned exclusively with online email storage, rather than the size of an email database stored locally on one’s computer, which Petitioner’s own references confirm and Dr. Terveen admitted could be substantially larger than Petitioner’s assumption. (EX1004, 120 (explaining POP accounts use server storage only as a temporary repository and messages are deleted from server once downloaded); EX2027, 1 (Entourage supports databases of up to 2 GB); EX2026, 201:6-22, 202:1-16.) Petitioner’s reliance on these web-mail exhibits to establish a 2-10 MB benchmark for locally stored databases is therefore misplaced. Further, none of the references speaks to the size or performance characteristics of databases spanning multiple file types.

With respect to EX1061, the small database sizes it describes are a direct consequence of the very index-size problem discussed above: because the Lotus Notes index structure consumes 50% or more of the size of the database (EX1061,

136), databases of any meaningful scale would generate indexes that overwhelm the available RAM on a 2002-era computer.

Even adopting Petitioner's 2-10 MB database size and its conclusion that the resulting index would occupy only 0.16% of a 2002-era computer's RAM (Reply, 7; EX1100, ¶¶13-14), Grounds 1 and 2 still fail for reasons independent of index size. No index-size calculation, however favorable to Petitioner, can cure the underlying disclosure deficiencies in the prior art or supply the missing motivation to combine. (POR, 19-31, 57-64.)

V. True is Not Analogous Art

Petitioner's argument that True is analogous art (Reply, 10-12) ignores the manner in which True actually performs its search, which is precisely the kind of slow, serial approach that the '490 patent sought to overcome. (POR, 55-56.) As Dr. Terveen acknowledges, True breaks the transcript into different transcript windows and returns results for each transcript window separately. (EX1100, ¶32.) Despite Dr. Terveen's contention that "intra-file searches are not meaningfully different than inter-file searches... and modifying an intra-file search to search across files would be trivial" (EX1100, ¶37), extending this architecture to a multi-file search would simply replicate True's serial, window-by-window scanning logic across multiple files, requiring the system to scan one entire file (or transcript window) at a time in a sequential manner. (EX2020, ¶¶84-85.) Far from solving

the problem of slow, cumbersome searches, Terveen's proposed adaptation of True exacerbates it.

VI. The Ground 1 References Do Not Teach Elements [1.5], [1.10], And [1.13]

For limitation [1.5], Petitioner contends that the claims do not require indexing partial strings and that it properly relied on POSITA knowledge to supply the index structure the claims require. (Reply, 20-22.) Petitioner misses the point.

Whether or not the claims explicitly require indexing prefixes, they unquestionably require searching partial strings. Elements [1.8] and [1.13] require matching indexed character strings against partial search strings entered by the user. The index data structure in Petitioner's proposed combination must therefore support partial string searching. Petitioner has identified no prior art disclosing such a structure, whether through indexing all partial strings or otherwise. Lotus Notes discloses only a full-text whole-word index, and Petitioner's background references establish nothing more than the generic proposition that indexes associate strings with files. (*See* EX1003, ¶¶41-48.)

Petitioner's argument that a system indexing whole words inherently indexes prefixes—because common words like “a” happen to be prefixes of longer words—is not credible. (Reply, 21-22.) The vast majority of partial strings entered during a search are not whole words that would appear in a full-text index. Dr.

Terveen conceded as much during his deposition: for the word “declaration,” not one of its partial strings “d,” “de,” “dec,” “decl,” etc, is a whole word that would be indexed. (EX2026, 213:16-214:1.) A few incidental overlaps between common words and prefixes do not transform the Lotus Notes whole-word full-text index into one capable of supporting the partial string searching required by the claims.

Likewise, Petitioner’s background references do not support that indexes enabling partial string search were within the knowledge of a POSITA. As discussed in the POR, this is improper gap filling. (POR, 28-29 (citing 37 C.F.R. 42.104(b)(4)).)

For limitations [1.10] and [1.13], Petitioner merely refers to its flawed motivation to combine arguments for the distinct Lotus Notes functionalities and multi-file search. For the reasons discussed above and in the POR, a POSITA would not be motivated with a reasonable expectation of success to do either. (*Supra* §§III.B, III.D; POR, 10-17.) The combination does not disclose or teach these limitations.

Date: April 1, 2026

Respectfully submitted,

/Lori A. Gordon/

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

Lead Counsel for Patent Owner

CERTIFICATE OF WORD COUNT

Pursuant to 37 C.F.R. § 42.24(d), the undersigned certifies that the foregoing **PATENT OWNER'S SUR-REPLY** complies with the complies with the type-volume limitation of 37 C.F.R. § 42.24(c)(4) because it contains 5,583 words according to the word-processing system used to prepare this Sur-Reply, excluding the words in the Table of Contents, Table of Authorities, List of Exhibits, Certification Under §42.24(d), and Certificate of Service, as set forth in 37 C.F.R. § 42.24(c).

Date: April 1, 2026

/Lori A. Gordon/

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

Lead Counsel for Patent Owner

CERTIFICATE OF SERVICE

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

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

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

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

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

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

Msft-X1-IPRs@klarquist.com

Attorneys for Microsoft Corporation

Date: April 1, 2026

/Lori A. Gordon/

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

Lead Counsel for Patent Owner