

UNITED STATES DISTRICT COURT
NORTHERN DISTRICT OF CALIFORNIA

VIRTAMOVE, CORP.,
Plaintiff,

v.

GOOGLE LLC,
Defendant.

Case No. 25-cv-00860-NW
Case No. 26-cv-00704 (consol.)

CLAIM CONSTRUCTION ORDER

Re: ECF Nos. 63, 65, 185, 191

VIRTAMOVE, CORP.,
Plaintiff,

v.

AMAZON.COM, INC., et al.,
Defendants.

Case No. 25-cv-10579-NW

CLAIM CONSTRUCTION ORDER

Re: ECF Nos. 71, 77

In these related actions, Plaintiff VirtaMove, Corp. (“VirtaMove”) asserts that Defendants Google LLC (“Google”) and Amazon.com, Inc., Amazon.com Services LLC, and Amazon Web Services Inc. (collectively, “Amazon”) infringe certain claims of VirtaMove’s patent(s). In the first action, case no. 25-cv-00860-NW (the “*Google* Action”), VirtaMove asserts that Google infringes certain claims of two of VirtaMove’s patents, U.S. Patent No. 7,519,814 (the “’814 Patent”) and U.S. Patent No. 7,774,762 (the “’762 Patent”).¹ In the second action, case no. 25-cv-10579-NW (the “*Amazon* Action”), VirtaMove asserts that Amazon infringes certain claims of the ’814 Patent only. Both cases were transferred to this Court from the Western District of Texas. On October 1, 2026, the Court held a technology tutorial and *Markman* hearing in both cases.

¹ Pursuant to the parties’ stipulation, the *Google* Action also includes case no. 26-cv-00704-NW, which was consolidated with case no. 25-cv-00860-NW. *See* 26-cv-00704-NW, ECF No. 47.

1 *Google* Action, ECF No. 201; *Amazon* Action, ECF No. 215.

2 Having considered the parties' submissions and arguments, the Court resolves the disputed
3 terms that the parties agree are dispositive as to the '762 and '814 Patents and construes the terms
4 as set forth below.² *See* *Google* Action, ECF No. 165 at 2; *Amazon* Action, ECF No. 82 at 4.
5 Because the Court finds at least one term indefinite as to each of the asserted claims, it determines
6 that claims 17–20 of the '762 and claims 1–30 of the '814 Patents are invalid for indefiniteness.³

7 **I. BACKGROUND**

8 **A. The State of the Art as Identified in the '762 and '814 Patents**

9 The '762 and '814 Patents are related; the '762 Patent is a continuation-in-part of the
10 application that became the '814 Patent, and incorporates the '814 Patent by reference. '762
11 Patent at 1:18–24.⁴ Both patents relate to the field of “computer software, and in particular, . . . to
12 management and deployment of” applications. '762 Patent at 26–32; '814 Patent at 12–16.

13 It is relevant to both patents that computers have a limited amount of hardware resources to
14 be used by applications, such as “CPU time, memory, file activity, and network bandwidth.” *See*
15 *id.* at 1:36–37, 12:31–32. “It is traditionally the task of an operating system to provide a
16 mechanism to safely and effectively control access to shared resources required by application
17 programs.” *Id.* at 37–40. The core of an operating system is the “kernel” that serves as the bridge

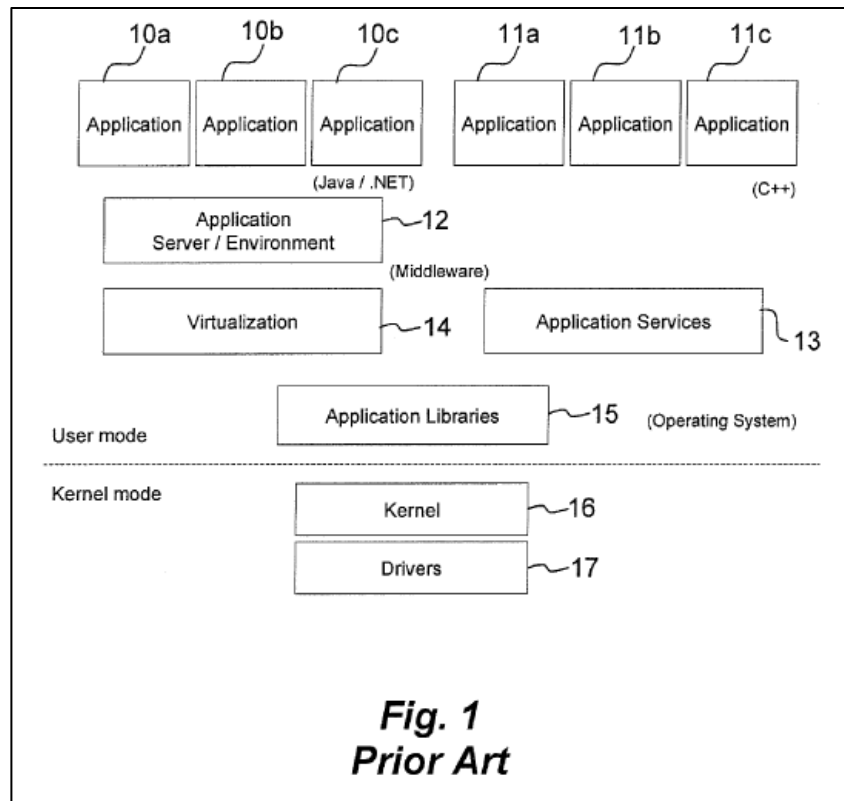
18
19 ² The Court previously determined that the *Google* and *Amazon* actions were related. *Google*
20 Action, ECF No. 167. Although some of the construed terms across the actions differ, several
21 terms are common (in particular, the disputed term “disparate computing environments,” used in
22 claim 1 of in the '814 Patent). “Given the importance of uniformity in the treatment of a given
23 patent,” even if constructions of a term in one case are not binding on constructions in another
24 cases, a court “would be remiss to overlook [the] construction of the same claim terms in the same
25 patent.” *Cf. Finisar Corp. v. DirecTV Grp., Inc.*, 523 F.3d 1323, 1329 (Fed. Cir. 2008) (in an
26 appeal of a claim construction entered by the District Court for the Eastern District of Texas,
27 “consult[ing] the claim analysis of different district courts on the identical terms in the context of
28 the same patent.”) (internal quotation marks and citations omitted). For this reason, while the
Court’s constructions as to the '762 Patent and most terms of the '814 Patent are only entered in
the *Google* case, the Court issues this joint claim construction order and enters constructions in
both the *Google* and *Amazon* cases as appropriate.

³ The Court therefore need not, and does not, construe most of the other terms disputed by the parties.

⁴ Unless otherwise noted, record citations are to the Electronic Case File (“ECF”) and pinpoint citations are to the ECF-generated page numbers at the top of documents. For the patents, citations are to the column/line number where available and otherwise to the parts of the patent.

between applications and the computer's shared resources. *See Google Action*, ECF No. 199-2 at 13 (VirtaMove's technology tutorial); October 1, 2026 Hrg. Tr. at 23:16–24:5, 32:24–33:13, 38:11–18 (neither Google nor Amazon disagreed with this aspect of VirtaMove's presentation).⁵

As illustrated in Figure 1, in the prior art, individual software application functions would rely on local file libraries of the operating system, and thereby make system calls to the kernel when access to the computer's shared resources were required:



'762 Patent, Fig. 1.; *see also Google Action*, ECF No. 199-2 at 14–15. As Google further explained at the hearing, based on the specification of the '814 Patent, an “application library” is a collection of code for performing common functions. *Google Action*, ECF No. 200-2 at 4 (citing U.S. Patent No. 7,784,058 (the “’058 Patent”) at 6:46–55).⁶ An application library can include a “static librar[y],” where the code for common functions is contained directly in the application,

⁵ Citations to the hearing are to the rough transcript, as the final transcript is not yet available.

⁶ The '058 Patent (formerly Pat. App. No. 10/946,536) is another related patent; the '762 Patent is also a continuation-in-part of that application. '762 Patent at 1:9–18. The '058 Patent was previously asserted in the *Google Action*, (*see Google Action*, ECF No. 134), and, like the '814 Patent, the '058 Patent is incorporated into the '762 Patent.

(*see* '058 Patent at 6:54–55), and a “shared librar[y],” where the “library code space shared among all user mode applications” in a space “that is accessible to multiple applications.” *Id.* at 6:49–53. By way of example, the '762 and '814 Patents provide a list of shared libraries included in the Linux Apache operating system distribution. '762 Patent at 5:13–37; '814 Patent at 2:55–3:13.

There are two problems in the prior art that the '762 and '814 Patent sought to solve: incompatibility between applications and operating systems, and conflicts between applications.

The first problem is that “[s]oftware applications are generally multi-layered and there are multiple facets to dependencies therebetween.” *See* '762 Patent at 5:44–52. For example, because a later operating system like Solaris 10 might have different (or differently-named) libraries from an earlier operating system like Solaris 8, “applications [that were] designed to run on a specific computer platform” typically “will not run on one it is not designed for.” *Id.* at 1:43–51; *see also id.* at 11:4–36. So, if programmers wanted their applications to be able to run on different or newer operating systems, they had “to design specific versions of their software that are compatible with each specific computer platform and operating system.” *Id.* at 1:43–51.

Second, even for applications running on their own, native computer platforms, if multiple applications were installed on a single computer, they could conflict over resources (*e.g.*, if multiple applications tried to use the same network port number). '814 Patent at 1:27–36. Or, even if not conflicting over resources, there might be security and data integrity conflicts and risks (*e.g.*, if applications tried to write to the same memory); this often meant that data created by applications had to be kept separate between distinct users or applications. *See id.*; *see also Google* Action, ECF No. 199-2 at 5 (explaining *id.*); October 1, 2026 Hrg. Tr. at 23:16–24:5, 32:24–33:13, 38:11–18 (neither Google nor Amazon disagreed with this aspect of VirtaMove’s presentation). In the prior art, according to the '814 Patent, preventing such conflicts required “each application [to be] installed on an individual computer system,” either physically or through a virtual machine. *See* '814 Patent at 1:27–2:3.

B. The '762 Patent

The '762 Patent, entitled “System Including Run-Time Software to Enable a Software Application to Execute on an Incompatible Computer Platform” was issued on Aug. 10, 2010.

1 '762 Patent at (45) (Date of Patent). It sought to solve the incompatibility problem and is directed
 2 to "a system for enabling a first software application, which is designed for execution on a first
 3 computer platform, to be executed on an incompatible computer platform in which the software
 4 application is not intended to execute." *Id.* at (57) (Abstract). The '762 Patent explains how to
 5 accomplish this result, such as by the method of claim 17, which provides:

6 In a system having a first software application, which is designed for
 7 execution on a first computer platform, a method of executing said
 8 first software application on an incompatible computer platform in
 9 which the first software application is not intended to execute, said
 10 first software application including an object executable by a local
 11 operating system of the incompatible computer platform for
 12 performing a task related to a service, the incompatible computer
 platform including a processor, an operating system including a
 local kernel residing on the incompatible computer platform, and a
 set of associated local system files compatible with the local kernel
 and the processor of the incompatible computer platform, the
 method comprising:

- 13 a) providing a computer readable set of first files,
 14 comprising a first group of files that include the first
 15 software application, designed to execute on the first
 16 computer platform and a second group of files that are
 17 distributed with or are specific to an operating system
 18 required to execute the first software application;
- 19 b) in the execution of the first software application on the
 20 incompatible computer platform using said second group
 21 of files in place of the associated local system files
 22 normally used to perform a same task, which are resident
 23 on the incompatible computer platform as part of the
 24 operating system, and,
- 25 c) executing on the second incompatible computer platform
 26 runtime software for managing a dynamic state and file
 27 location of the first software application, the runtime
 28 software including a kernel module resident in kernel
 mode and at least one application filter library resident in
 user mode and filtering one or more system service
 requests made by the first software application and
 providing values from the at least one application filter
 library which is provided in addition to the local
 operating system libraries to modify values that
 otherwise would have been returned by the local
 operating system in the absence of said filtering one or
 more system service requests from the first software
 application, and executing the first software application
 by accessing files from first set of files in place of
 operating system files and wherein predetermined values
 are provided instead of values related to an identity of the
 incompatible computer platform.

'762 Patent at 17:25–18:28. In essence, rather than relying on the “associated local system files”⁷ of the “incompatible computer platform” to run a software application, the '762 Patent discloses how to “provid[e] . . . a second group of files that are distributed with or are specific to” the operating system the software application requires, and then “us[e] said second group of files in place of the associated local system files normally used” to run the software. *Id.* The method of the claim 17 accomplishes this by using runtime software including an “application filter library” to “filter[]” the “system service requests made by the” software application to cause it to take the modified filtered values “instead of [the] values related to . . . the incompatible computer platform.” *Id.* This is shown in Figure 1a, with the filters 112 applying to applications 102a–c.

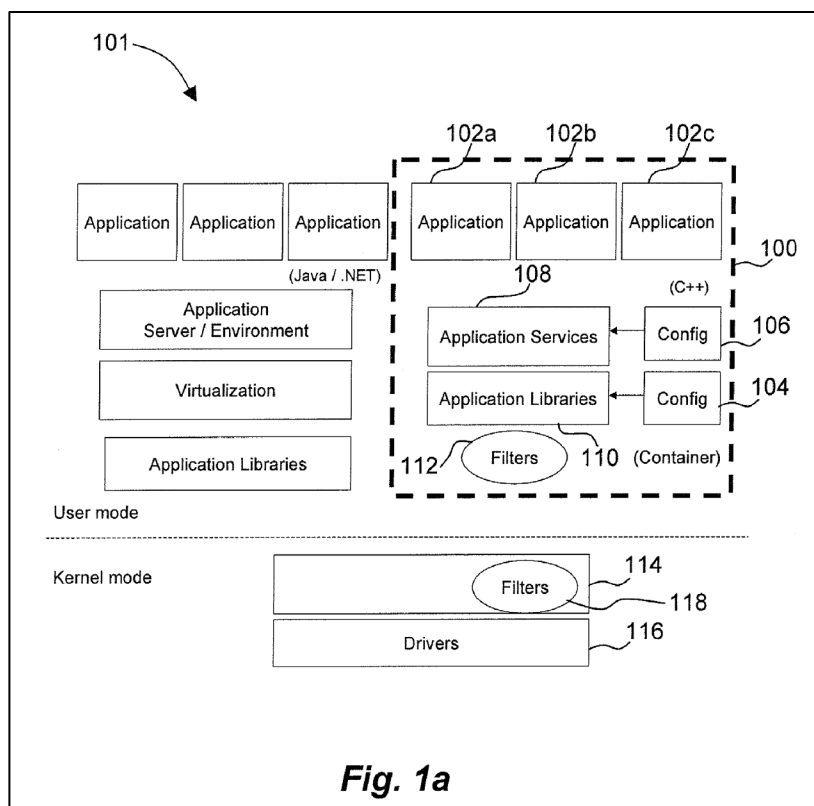


Fig. 1a

'762 Patent, Fig. 1a.

C. The '814 Patent

The '814 Patent, entitled “System for Containerization of Application Sets” was issued on April 14, 2009. '814 Patent at (45) (Date of Patent). The patent sought to solve the resource-and-

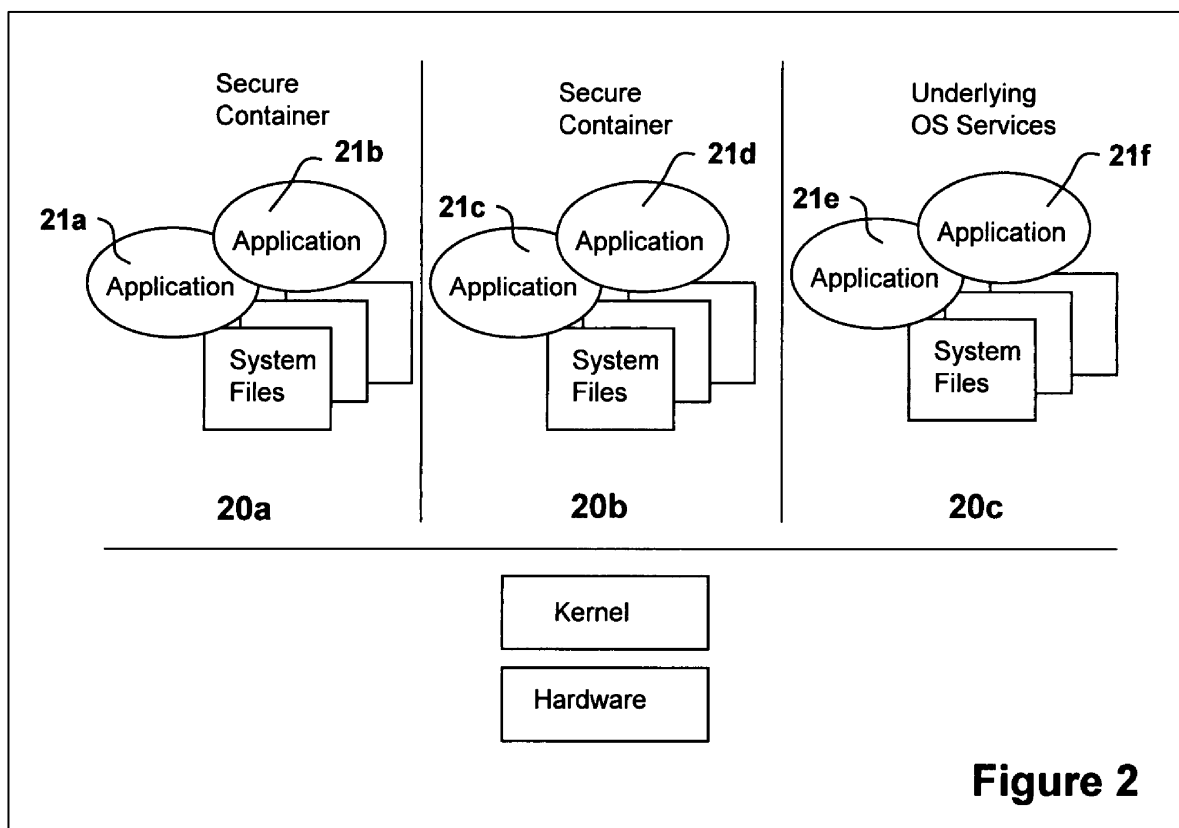
⁷ Per the '762 Patent, “system files” are the “files provided within an operating system and which are available to applications as shared libraries and configuration files.” '762 Patent at 5:10–12.

1 security-conflicts problem, as well as the incompatibility problem. *See id.* at 1:27–2:3, 7:4–15. It
 2 is directed to a method, in a system “having servers with operating systems that may differ,
 3 operating in disparate computing environments, . . . of providing at least some of the servers in the
 4 system with secure, executable, applications related to a service . . . wherein the applications each
 5 include an object executable by at least Some of the different operating systems . . . [by] storing in
 6 memory accessible to at least some of the servers a plurality of secure containers of application
 7 software.” *Id.* at (57) (Abstract). Independent claim 1 of the ’814 Patent recites:

8 In a system having a plurality of servers with operating systems that
 9 differ, operating in disparate computing environments, wherein each
 10 server includes a processor and an operating system including a
 11 kernel a set of associated local system files compatible with the
 12 processor, a method of providing at least some of the servers in the
 13 system with secure, executable, applications related to a service,
 14 wherein the applications are executed in a secure environment,
 15 wherein the applications each include an object executable by at
 16 least some of the different operating systems for performing a task
 17 related to the service, the method comprising:

18 storing in memory accessible to at least some of the servers a
 19 plurality of secure containers of application software, each
 20 container comprising one or more of the executable
 21 applications and a set of associated system files required to
 22 execute the one or more applications, for use with a local
 23 kernel residing permanently on one of the servers; wherein
 24 the set of associated system files are compatible with a local
 25 kernel of at least some of the plurality of different operating
 26 systems, the containers of application software excluding a
 27 kernel, wherein some or all of the associated system files
 28 within a container stored in memory are utilized in place of
 the associated local system files that remain resident on the
 server, wherein said associated system files utilized in place
 of the associated local system files are copies or modified
 copies of the associated local system files that remain
 resident on the server, and wherein the application software
 cannot be shared between the plurality of secure containers
 of application software, and wherein each of the containers
 has a unique root file system that is different from an
 operating system's root file system.

24 ’814 Patent at 17:30–61. The core of this method is storing in memory a plurality of “secure
 25 containers,” each comprising the applications to be executed and the “set of associated system
 26 files required to execute” them. *Id.* So, rather than relying on the local system files, these
 27 container-system files are used to run the application(s). *See id.* Such a system of secure
 28 containers and system files is illustrated in Figure 2 of the ’814 Patent:

**Figure 2**

'814 Patent, Fig. 2.

II. LEGAL STANDARD

A. Claim Construction Generally

"It is a bedrock principle of patent law that the claims of a patent define the invention to which the patentee is entitled the right to exclude." *Phillips v. AWH Corp.*, 415 F.3d 1303, 1312 (Fed. Cir. 2005) (cleaned up).

Because the meaning of a claim term as understood by persons of skill in the art is often not immediately apparent, and because patentees frequently use terms idiosyncratically, the court looks to "those sources available to the public that show what a person of skill in the art would have understood disputed claim language to mean." Those sources include "the words of the claims themselves, the remainder of the specification, the prosecution history, and extrinsic evidence concerning relevant scientific principles, the meaning of technical terms, and the state of the art."

Id. at 1314 (quoting *Innova/Pure Water, Inc. v. Safari Water Filtration Sys., Inc.*, 381 F.3d 1111, 1116 (Fed. Cir. 2004)).

Claim terms are defined by their "ordinary and customary meaning," which is "the meaning that the term would have to a person of ordinary skill in the art in question at the time of

the invention.” *Id.* at 1312-13. In ascertaining the ordinary meaning of a term, courts first look to the patent’s claim terms, specification, and prosecution history, which is the patent’s “intrinsic record.” *Id.* The claim language is “of primary importance” when determining the bounds of the claimed invention and “[t]he specification necessarily informs the proper construction of the claims.” *Id.* at 1316. Extrinsic evidence may elucidate relevant art, but such evidence is only considered within the context of intrinsic evidence. *Id.* at 1317-19. “Extrinsic evidence consists of all evidence external to the patent and prosecution history, including expert and inventor testimony, dictionaries, and learned treatises.” *Markman v. Westview Instruments, Inc.*, 52 F.3d 967, 980 (Fed. Cir. 1995), *aff’d*, 517 U.S. 370 (1996).

“[D]istrict courts are not (and should not be) required to construe every limitation present in a patent’s asserted claims.” *O2 Micro Int’l Ltd. v. Beyond Innovation Tech. Co.*, 521 F.3d 1351, 1362 (Fed. Cir. 2008). “Rather, claim construction is a matter of resolution of disputed meanings and technical scope, to clarify and when necessary to explain what the patentee covered by the claims.” *Id.* (internal alterations and quotation marks omitted). Thus, when “parties present a fundamental dispute regarding the scope of a claim term, it is the court’s duty to resolve it.” *Id.*

A. Indefiniteness

35 U.S.C. § 112 requires the claims of a patent to particularly point out and distinctly claim the subject matter regarded as the invention. *Nautilus, Inc. v. Biosig Instruments, Inc.*, 572 U.S. 898, 902 (2014). “[A] patent is invalid for indefiniteness if its claims, read in light of the specification delineating the patent, and the prosecution history, fail to inform, with reasonable certainty, those skilled in the art about the scope of the invention.” *Id.* at 901. Though “absolute precision is unattainable[,]” “a patent must be precise enough to afford clear notice of what is claimed, thereby apprising the public of what is still open to them.” *Id.* at 909-10 (internal alterations omitted). A claim is sufficiently definite if it provides objective boundaries for those of skill in the art. *Interval Licensing LLC v. AOL, Inc.*, 766 F.3d 1364, 1371 (Fed. Cir. 2014).

Patents, however, are presumed valid. 35 U.S.C. § 282. The party challenging a patent’s validity has the burden of proving indefiniteness by clear and convincing evidence. *Dow Chem. Co. v. Nova Chemicals Corp. (Canada)*, 809 F.3d 1223, 1227 (Fed. Cir. 2015); *see also Microsoft*

Corp. v. I4I Ltd. P'ship, 564 U.S. 91, 95 (2011) (holding a party asserting an indefiniteness invalidity defense must prove it by clear and convincing evidence). Even so, “indefiniteness is a question of law and in effect part of claim construction.” *ePlus, Inc. v. Lawson Software, Inc.*, 700 F.3d 509, 517 (Fed. Cir. 2012). General principles of claim construction apply when a court evaluates indefiniteness. *HHZNP Medicines LLC v. Actavis Lab'ys UT, Inc.*, 940 F.3d 680, 688 (Fed. Cir. 2019).

III. DISCUSSION

A. The Disputed Dispositive Terms

VirtaMove and Google initially disputed eleven terms for the '762 Patent and five terms for the '814 Patent. *Google* Action, ECF No. 165. VirtaMove and Amazon initially disputed two terms, both for the '814 Patent. *Amazon* Action, ECF No. 82. Of these, the parties agree⁸ that the following five terms are potentially case- or claim-dispositive:

1. “an incompatible computer platform in which the first software application is not intended to execute” ('762 Patent at 17:26–29) (at issue in independent claim 17);
2. “the second incompatible computer platform” ('762 Patent at 18:11) (same);
3. “the local operating system libraries” ('762 Patent at 18:19–20) (same);
4. “disparate computing environments” ('814 Patent at 17:31–31) (at issue in independent claim 1 and dependent claims 2, 4, 6, 8–10, 13–14); and
5. “an operating system’s root file system” ('814 Patent at 17:60–61) (same).

The disputes as to terms 1–3 and 5 are between VirtaMove and Google only; the dispute as to term 4 is common to all parties, with both Google and Amazon asserting the term is indefinite.

Compare Google Action, ECF No. 165 *with Amazon* Action, ECF No. 82.

The Court addresses each term in turn and, finding at least one term invalid as to each of the asserted claims, determines that those claims are invalid. Based on that determination, the

⁸ In the *Google* Action, the parties’ identification of “the 10 terms likely to be most significant to resolving the parties’ dispute” pursuant to Patent Local Rule (“PLR”) 4-1(b) confusingly omitted two terms of the '762 Patent that Google contends are indefinite. *See Google* Action, ECF No. 165. VirtaMove and Google are admonished that a term that one side contends is indefinite is one that “may be case or claim dispositive” and should have been identified. PLR 4-1(b).

1 Court does not address the parties' remaining disputed terms.

2 **B. The Court Adopts the Parties' Relevant Agreed Constructions**

3 In their joint claim construction statements ("JCCS") and in stipulations following the
 4 *Markman* hearing, the parties agreed to certain constructions. *See Google* Action, ECF. No. 165
 5 (VirtaMove's and Google's JCCS); *Amazon* Action, ECF No. 82 (VirtaMove's and Amazon's
 6 JCCS); *Google* Action, ECF No. 202 (VirtaMove's and Google's post-hearing stipulation);
 7 *Amazon* Action, ECF No. 217 (VirtaMove's and Amazon's post-hearing stipulation). Although
 8 the Court need not at this juncture enter most of the agreed constructions, a few constructions are
 9 relevant to the Court's indefiniteness findings:

- 10 • All parties agree that the preambles of claim 17 of the '762 Patent and claim 1 of
 11 the '814 Patent are limiting (*Google* Action, ECF. No. 165 at 4; *Amazon* Action,
 12 ECF No. 82 at 5);
- 13 • VirtaMove and Google agree that, in claim 17 of the '762 Patent, a "computer
 14 platform" means a "computer system with a single instance of a fully functional
 15 operating system installed." *Google* Action, ECF No. 165 at 4. This term does not
 16 appear in claim 1 of the '814 Patent;
- 17 • VirtaMove and Google agree as to the antecedent bases of various terms in the '762
 18 and '814 Patent (*Google* Action, ECF No. 165 at 6);

19 The Court adopts the parties' agreed construction that the preambles are limiting, and
 20 VirtaMove and Google's agreed construction as to the meaning of "computer platform." And,
 21 although most of the "antecedent bases" agreements appear obvious, for the clarity of the parties,
 22 the Court also adopts all these agreed constructions.

23 The constructions as to the '762 Patent are entered in the *Google* Action only, while the
 24 constructions as to the '814 Patent are entered in both actions. The Court does not adopt or reject
 25 the remaining agreed constructions at this juncture but will consider the parties' agreement to the
 26 extent future construction is required.

C. Disputed Terms Relevant to the '762 Patent⁹

1. Term 1 – “an incompatible computer platform in which the first software application is not intended to execute”

Term, Claim, and Patent	VirtaMove's Proposal	Google's Proposal	Court's Ruling
“an incompatible computer platform in which the first software application is not intended to execute” (’762 Patent, claim 17)	“a computer platform in which the first software application cannot run without change and in which the first software application is not designed to execute”	Indefinite	Indefinite as “an incompatible” Indefinite as to an application “not intended to execute”

As an initial matter, there are two discrete portions of this term that Google contends are indefinite: (1) what it means for a computer platform to be “incompatible” with a software application and (2) what it means for the application to be “not intended to execute” on the platform. *See, e.g., Google Action*, ECF No. 191 at 9–14 (Google’s argument as to “incompatible”), 14–15 (Google’s argument as to “not intended to execute”); *Google Action*, ECF No. 193 at 4–8 (VirtaMove’s reply, similarly divided).

a. “Incompatible”

Google argues that “incompatible” portion is indefinite because incompatibility can take many forms and the “claims do not provide the objective boundaries as to which form(s) of incompatibility are encompassed by the claimed invention,” and which forms of incompatibility are not. *Google Action*, ECF No. 191 at 10–12. Moreover, Google argues that VirtaMove’s proposed construction would contradict the surrounding claim language. *See id.* at 12–13. Google’s argument is persuasive as to the second point.

The Court begins, as it must, with the plain language of the claims. Claim 17 identifies two computer platforms: “a first computer platform” on which the software application “is designed for execution,” and another computer platform which is the “incompatible computer platform,” on which the application “is not intended to execute.” ’762 Patent at 17:25–29. In a vacuum, Google’s argument that “incompatible” might mean many things, *e.g.*, that an

⁹ The Court’s constructions of these terms are entered in the *Google Action* only.

“application may run but crash at fixed or random times, [or] it may lack some but not all of its functionality, etc.” has some force. *See Google Action*, ECF No. 191 at 11. But the Court does not construe claim language in a vacuum. *See, e.g., Phillips v. AWH Corp.*, 415 F.3d at 1313.

First, the surrounding language of the claim sheds some light on the meaning of “incompatibility” by using its antonym when it identifies “a set of associated local system files compatible with the local kernel and the processor of the incompatible computer platform.” *Id.* at 17:34–37. Moreover, the same word—incompatible—is used later in claim 17 in describing how “the execution of the first software application on the incompatible computer platform” is accomplished using the “second group of files in place of the associated local system files . . .” *Id.* at 18:5–10. All of this suggests that a person of ordinary skill in the art (“POSITA”) would understand “incompatible” as used in claim 17 to mean something along the lines of “able to be executed” (or “run”)¹⁰ on. VirtaMove also persuasively points to portions of the specification and prosecution history that further clarify the meaning of “incompatible” and point to essentially the same construction with one modification: an incompatible computer platform is one “in which the first software application cannot run without change.” *Google Action*, ECF No. 165 at 7 (excerpt of VirtaMove’s proposed construction (emphasis added)); *see* ’762 Patent at 1:43–45, 52–55 (in the prior art, a “software application[] . . . will not run on [a computer platform] it is not designed for”; the ’762 Patent provides “a system whereby . . . “applications can be run or executed on an incompatible computer platform [on] which heretofore, the applications could not be run.”); *see also id.* at 1:45–50 (explaining that, without the invention of the ’762 Patent, “programmers desirous of their software to be used on a plurality of different incompatible computer platforms have to designed specific versions of their software that are compatible . . .”); *Google Action*, ECF No. 185-3 (during prosecution, the applicant explained that, “without this invention,” the software application “would not and could not run due to an incompatibility in the operating

¹⁰ VirtaMove’s proposed construction, discussed below, uses the verb “run” instead of “execute.” Although “execute” may be a more precise term, the specification uses “execute” and “run” interchangeably, *e.g.*, in the Abstract, and Google has not argued that there is any relevant distinction between these terms. *See, e.g.,* ’762 Patent at (57) (Abstract).

1 system resident on the second computer.” (referring language from claim 1 that is identical in
2 relevant respects to claim 17)).

3 In light of the surrounding claim language, the specification, and the prosecution history,¹¹
4 the Court finds that term “incompatible” is not boundless or hopelessly vague; rather, “an
5 incompatible computer platform” means “a computer platform in which the first software
6 application cannot run . . .,” with some qualifier. But the scope of that qualifier is where problems
7 arise for VirtaMove.

8 In its portion of the joint claim construction statement (“JCCS”), VirtaMove proposed that
9 the qualifier is that the application “cannot run without change.” *Google* Action, ECF No. 165 at
10 7. Thus, VirtaMove’s proposed construction was “a computer platform in which the first software
11 application cannot run without change . . .” *Id.* But in its opening and reply briefs VirtaMove
12 tweaked its construction, saying that “an ‘incompatible computer platform’ [] in the context of the
13 invention [is]: a computer platform . . . which cannot run the application without modification (***or***
14 ***without practicing the claimed invention***).” *Google* Action, ECF No. 185 at 9; *id.*, ECF No. 193
15 at 6 (emphasis by VirtaMove). It is easy to see why VirtaMove had to adjust its construction: as
16 Google argues, VirtaMove’s original proposed construction required “that the platform cannot run
17 the application without first changing the application.” *Google* Action, ECF No. 191 at 12–13.
18 But that requirement is “nonsensical and contradicts explicit claim language,” because the very
19 language of claim 17 also *requires* that the “first software application” be run on the incompatible
20 computer platform in step (b), *without any change to the application itself*—merely by using the
21 provided “second group of files in place of the associated local system files normally used to
22 perform a same task . . .” *Google* Action, ECF No. 191 at 12–13; ’762 Patent at 17:25–18:28.
23 “Contradictory limitations within a single claim or between different claims can render a claim
24 indefinite.” *In re Shafovaloff*, 2025 WL 1779173, at *1 (Fed. Cir. June 27, 2025) (citing *Trs. of*
25 *Columbia Univ. in City of New York v. Symantec Corp.*, 811 F.3d 1359, 1366–67 (Fed. Cir. 2016).

26
27
28 ¹¹ To the extent the Court considers extrinsic evidence, Google’s own dictionary definition defines
“incompatible” as “not capable of being used in combination” and so supports a reading of
“incompatible” as “not capable of being run/executed with.” *See*

1 To avoid contradiction, VirtaMove’s initially proposed construction would have to be modified.

2 But here, VirtaMove’s change would render the claim circular – an incompatible computer
3 platform is one in which the application cannot run without practicing the claimed invention, *i.e.*,
4 without having the very incompatible computer platform the constructions seeks to define. The
5 fact that a proposed construction is circular is “[a] compelling reason for rejecting [it].” *Harris*
6 *Corp. v. IXYS Corp.*, 114 F.3d 1149, 1152 (Fed. Cir. 1997). And courts in this District have
7 rejected circular constructions, even when doing so rendered the claims indefinite. *E.g.*, *Sunstone*
8 *Info. Def., Inc. v. F5, Inc.*, No. 21-cv-09529-YGR, 2023 WL 2746762, at *7 (N.D. Cal. Mar. 30,
9 2023) (citing *id.*).

10 The Court declines to add the tweaked qualifier, “or without practicing the claimed
11 invention,” which would render the term circular, and adopts VirtaMove’s original proposed
12 construction that an “incompatible computer platform” is one “in which the first software
13 application cannot run without change.” But because that construction renders the claim
14 contradictory, even if the word “incompatible” is not completely meaningless, Google has shown
15 by clear and convincing evidence¹² that the term is indefinite and claim 17 is invalid. *See, e.g.*,
16 *Interval Licensing LLC v. AOL, Inc.*, 766 F.3d 1364, 1371 (Fed. Cir. 2014) (“The Supreme Court
17 explained that a patent does not satisfy the definiteness requirement of § 112 merely because ‘a
18 court can ascribe some meaning to a patent’s claims.’ The claims, when read in light of the
19 specification and the prosecution history, must provide objective boundaries for those of skill in
20 the art.” (quoting *Nautilus*, 572 U.S. at 911)).

21 **b. “Not Intended to Execute”**

22 Google argues that the second part of this term, that the software application is “not
23 intended to execute” on the incompatible computer platform, is also indefinite because this

24
25 ¹² VirtaMove argues that Google’s arguments on this and other terms fail as an evidentiary matter,
26 because Google has not submitted any expert testimony on indefiniteness. *See Google Action*
27 *ECF No. 185 at 7*. “But that is not required. Indefiniteness is a question of law, and a party need
28 not [necessarily] present ‘extrinsic evidence that one of ordinary skill in the art would fail to
understand that a term connotes a definite structure.’” *E.g. VTT Tech. Rsch. Ctr. of Finland Ltd. v. SiTime Corp.*, No. 4:19-cv-1174-YGR, 2020 WL 4193374, at *13 (N.D. Cal. July 21, 2020), *aff’d*,
849 F. App’x 928 (Fed. Cir. 2021) (quoting *Diebold Nixdorf, Inc. v. International Trade Commission*, 899 F.3d 1291, 1299 (Fed. Cir. 2018) (emphasis in original)).

1 limitation is subjective. *Google Action*, ECF No. 191 at 14. The Court agrees.

2 At the outset, “exclusively subjective language,” such as intent, “may be invalid.” *Malibu*
3 *Boats, LLC v. Skier’s Choice, Inc.*, No. 18-cv-15-JPM, 2020 WL 5026852, at *10 (E.D. Tenn.
4 Aug. 25, 2020) (citing *Datamize, LLC v. Plumtree Software, Inc.*, 417 F.3d 1342, 1350 (Fed. Cir.
5 2005), *abrogated on other grounds, Nautilus*, 572 U.S. 898 (2014)). “Federal Circuit precedent
6 does not automatically require that claim language that looks subjective is necessarily so
7 subjective that the resulting term is indefinite as a matter of law.” *Id.* For example, in some cases,
8 courts have found phrases “intended to be used for” or “intended to prevent” not to be indefinite
9 where the patents’ specifications and prosecution histories reflect that the phrases “introduce a
10 functional limitation” that “bound[s] the scope of the claims by specifying the operations that the
11 claimed invention must undertake.” *Williamson v. Google Inc.*, No. 15-cv-00966-BLF, 2017 WL
12 3232582, at *11 (N.D. Cal. July 27, 2017) (finding that “substituting ‘intended’ with ‘configured’
13 would be equally acceptable” and that the phrase “[a] first request signal includes information
14 intended to prevent said first request signal from being blocked” was not indefinite).

15 VirtaMove urges that this case is like *Williamson*, and that “intended” could be substituted
16 for “designed,” such that the application is “not designed to execute” on the incompatible
17 computer platform. *See Google Action*, ECF No. 185 at 11–13; *id.*, ECF No. 193 at 7–9.
18 VirtaMove is correct in one respect: the Court is convinced that the patentee appears to have used
19 “intended” as a synonym for “designed” throughout the patent. *See* ’762 Patent at (57), 1:25–51,
20 2:48–67.¹³ But substituting the word “designed” for the word “intended” does not resolve the
21 fundamental problem: the patent provides no guidance as to what objective parameters are used to
22 assess the “design.”

23 This case is unlike *Williamson* in two important respects. First, in *Williamson*, the

24
25 ¹³ Google points to claim differentiation and unasserted claim 16 to support the argument that
26 “intended” should not be construed to mean “designed.” *See Google Action*, ECF No. 191 at 14–
27 15. “The doctrine of claim differentiation stems from ‘the common sense notion that different
28 words or phrases used in separate claims are presumed to indicate that the claims have different
meanings and scope.’” *Seachange Int’l, Inc. v. C-COR, Inc.*, 413 F.3d 1361, 1368 (Fed. Cir.
2005) (internal citation omitted). But the doctrine is “not a hard and fast rule”; it “only creates a
presumption,” (*see id.*), and in this case, that presumption is rebutted by the overwhelmingly
consistent, synonymous usage of the terms in the specification.

1 specification and prosecution history supported specific boundaries as to the intended (or
2 “designed” or “configured”) signal. For example, “[t]he prosecution history show[ed] that the
3 claim amendment incorporating the phrase containing ‘intended’ was not to introduce subjective
4 intent, but to overcome an enablement issue” – that the patent examiner had found a stricter phrase
5 (“that the ‘signal cannot be blocked’”) was not enabled. *Williamson*, 2017 WL 3232582, at *11
6 (N.D. Cal. July 27, 2017). Moreover, the specification supported “the use of the word ‘intended’
7 as introduc[ing] a functional limitation in connection with [recited] ‘information,’” *i.e.*, the
8 information had to do something specific, namely, be configured “to prevent [the] first request
9 signal from being blocked.” *Id.* Here, VirtaMove has pointed to no statements from the
10 prosecution history or specifications defining any objective parameters for the “design” of the
11 application.

12 This runs into the second difference between the patent in *Williamson* and this case. The
13 closest thing to an objective property of the application’s design that the ’762 Patent provides is
14 that the application either is or is not “designed for execution on” or “designed to execute on” the
15 particular computer platforms. *See, generally*, ’762 Patent. But if that purpose were used to
16 bound the term, it would render the “not intended to execute” limitation completely redundant and
17 superfluous, because, according to VirtaMove, the definition of the incompatible computer
18 platform is *already* one “in which the first software application cannot run [*i.e.*, execute] without
19 change.” It is a core canon of claim construction that “[c]laims must be interpreted with an eye
20 toward giving effect to all terms in the claim.” *Becton, Dickinson & Co. v. Tyco Healthcare Grp.,*
21 *LP*, 616 F.3d 1249, 1257 (Fed. Cir. 2010) (internal quotation marks and citations omitted); *see*
22 *also Bicon, Inc. v. Straumann Co.*, 441 F.3d 945, 951 (Fed. Cir. 2006) (a claim construction that
23 reads a limitation out of a claim is “contrary to the principle that claim language should not [be]
24 treated as meaningless”). The only way that “not intended to execute” (or “not designed to
25 execute”) does not render “incompatible” superfluous is if the intent or design points toward
26 something more than the ultimate ability of the application to execute – and the ’762 Patent
27
28

provides no objective boundaries as to what that “something more” might be.¹⁴

For this alternative reason, because the term “not intended to execute” as used in the ’762 Patent, even if construed as “not designed to execute” as VirtaMove urges, is either wholly unbounded or, if bounded, bounded only by subjective intent, Google has shown by clear and convincing evidence the term is indefinite and claim 17 is invalid.

2. Term 2 – “the second incompatible computer platform”

Term, Claim, and Patent	VirtaMove’s Proposal	Google’s Proposal	Court’s Ruling
“the second incompatible computer platform” (’762 Patent, claim 17)	“the incompatible computer platform”	Indefinite	“the incompatible computer platform”

As already explained, the ’762 Patent provides for two computer platforms: “a first computer platform” on which the software application “is designed for execution,” and another “incompatible computer platform,” on which the application “is not intended to execute.” ’762 Patent at 17:25–29. In step (c) of the method, however, the ’762 Patent recites “the second incompatible computer platform.” *Id.* at 18:11–12. Google rightly points out that this second incompatible computer platform lacks an antecedent basis and argues this makes the claim indefinite and invalid.

In general, “[a] claim is ‘indefinite if a [claim] term does not have proper antecedent basis.’” *Sensor Elec. Tech., Inc. v. Bolb, Inc.*, No. 18-cv-05194-LHK, 2019 WL 4645338, at *31 (N.D. Cal. Sept. 24, 2019) (quoting *Halliburton Energy Servs., Inc. v. M-I LLC*, 514 F.3d 1244, 1249 (Fed. Cir. 2008)). On some occasions the Federal Circuit has held that the lack of an antecedent basis is not fatal, “as long as the claim [nonetheless] apprises one of ordinary skill in the art of its scope.” *In re Downing*, 754 F. App’x 988, 996 (Fed. Cir. 2018). But here,

¹⁴ For example, an application may not be intended for execution on a later computer platform but, through either chance or intentional efforts at backward compatibility when building that platform, the application runs with no problem. In such a case, the “intended” or “designed” limitation would be meaningful and distinct – but it would rely on the subjective intent of the programmer/designer. Moreover, the ’762 Patent is completely silent as to whether this or some other boundary was what the patentee intended.

1 VirtaMove’s attempt to rehabilitate the claim as-is explicitly requires the Court to strike the word
 2 “second” from the term. *See Google Action*, ECF No. 185 at 13–15; *id.*, ECF No. 193 at 9–10.
 3 The Court cannot do so by following the usual rules of claim construction. *See Becton*, 616 F.3d
 4 at 1257 (holding that “[c]laims must be interpreted with an eye toward giving effect to all terms in
 5 the claim.” (internal quotation marks and citations omitted)); *Bicon*, 441 F.3d at 951 (a claim
 6 construction that reads a limitation out of a claim is “contrary to the principle that claim language
 7 should not [be] treated as meaningless”). Rather, the Court agrees with Google that, “[w]hile not
 8 acknowledging it, what VirtaMove is really asking the Court to do is *correct* the claim
 9 language[.]” *Google Action*, ECF No. 191 at 17 (emphasis by Google).

10 “A district court may correct an ‘obvious’ error in a patent.” *See VTT Tech. Rsch. Ctr. of*
 11 *Finland Ltd. v. SiTime Corp.*, No. 4:19-cv-1174-YGR, 2020 WL 4193374, at *8 (N.D. Cal. July
 12 21, 2020), *aff’d*, 849 F. App’x 928 (Fed. Cir. 2021). But it can do so only if “(1) the correction is
 13 not subject to reasonable debate based on consideration of the claim language and the specification
 14 and (2) the prosecution history does not suggest a different interpretation of the claims.” *Id.*
 15 (quoting *Novo Indus., L.P. v. Micro Molds Corp.*, 350 F.3d 1348, 1354 (Fed. Cir. 2003)). “The
 16 error and the correction are both evaluated from the perspective of one of ordinary skill in the art.”
 17 *Id.* Here, the Court parts ways with Google – the error is obvious on the face of the patent, the
 18 correction is not subject to reasonable debate, and the prosecution history confirms, rather than
 19 undermines, the correction.

20 First, the error is obvious from the face of the patent. The entire specification and each of
 21 the claims speaks only of two computer platforms – the first computer platform and another
 22 incompatible one. *See, generally*, ’762 Patent. A POSITA would understand that the introduction
 23 of another, “second incompatible computer platform” is an error and that the patentee intended
 24 merely to refer to the second computer platform, *i.e.*, the incompatible one; Google does not argue
 25 otherwise. *See Google Action*, ECF no. 191 at 16–18.

26 The correction is equally clear: striking the word “second.” Google argues that this could
 27 be the case, but the claim could also be read as requiring a separate, second incompatible computer
 28 platform to perform the filtering step. *Id.* The Court is not persuaded. As VirtaMove points out

on reply, the claim language and specification consistently refer to “capsule runtime software” or “runtime software,” which performs the filtering of step (c), as being installed on and running on *the* incompatible computer platform, *i.e.*, the same one as referred to elsewhere in the claims and specification. *E.g.*, ’762 Patent at 2:19–47, 2:63–3:33; 6:14–26. Similarly, claim 1 of the ’762 Patent is explicit that the runtime software “provid[es] values from” an “application filter library stored in the second memory,” and that “second memory” is “the second memory on the incompatible computer platform.” ’762 Patent at 15:14–67. Simply put, while Google’s alternative interpretation is grammatically possible, it makes no sense given the patent’s specification, and it is therefore not reasonable. Overall, the Court finds that the claims and specification leave no room for debate that the “second” incompatible computer platform is intended to refer to the same “incompatible computer platform” as used elsewhere in the claims.

Finally, the prosecution history does not suggest a different interpretation. In fact, it confirms the Court’s correction, and it makes clear how the inclusion of the “second incompatible computer platform” occurred. It confirms that this was a drafting error, introduced when the applicant was broadly introducing a suite of “second” hardware components for the second computer platform: “a second incompatible operating system,” a “second memory,” and a “second processor,” all of which pertained to the original “incompatible computer platform.” *See, e.g.*, *Google Action*, ECF No. 185-3 at 13. But the applicant’s explanation made clear that “without this invention, if the first software application is run on **the second computer platform**, it would not and could not run due to an incompatibility And, with this invention, the first software application is enabled to run on the second incompatible computer platform.” *Id.* at 12–13 (emphasis added). This final consideration leaves no doubt that the ’762 Patent is concerned with only two computer platforms: the first one and an incompatible one, *i.e.*, a second one that is incompatible. The prosecution history confirms that striking “second” from this term is the appropriate correction.

For these reasons, the Court corrects claim 17 by striking the word “second” from “the second incompatible computer platform,” and does not find the claim indefinite on this ground. *See VTT Tech. Rsch.*, 2020 WL 4193374, at *10 (correcting patent by inserting a temperature

range that was supported by the specification and the prosecution history, and declining to find the claim indefinite).

3. Term 3 – “the local operating system libraries”

Term, Claim, and Patent	VirtaMove’s Proposal	Google’s Proposal	Court’s Ruling
“the local operating system libraries” ('762 Patent, claim 17)	No construction necessary; plain and ordinary meaning.	Indefinite	“the local operating system” means “the operating system,” <i>i.e.</i> , the operating system of the incompatible computer platform; The Court gives the term “libraries” its plain and ordinary meaning.

The issue here, addressed by both parties, is the same as the preceding term. Claim 17 introduces a phrase, “the local operating system libraries,” without an antecedent basis. Here, however, the question is not one of correction – VirtaMove does not propose any corrected construction, and none is obvious on the face of the patent. So, the term is indefinite, and the claim is invalid, unless a POSITA would understand what “the local operating system libraries” refers to. Here again, the Court agrees with VirtaMove.

There are two potential ambiguities in this term. The first is, “what is the local operating system?” The Court agrees with VirtaMove that a POSITA would readily understand that the local operating system refers to the operating system of the incompatible computer platform, not the first computer platform, for two reasons. First, when “a local operating system” is defined in claim 17, it is the operating system “of the incompatible computer platform”; claim 17 never defines or refers to the operating system of the first computer platform, and a POSITA would understand that it does not suddenly do so now. *E.g., In re Downing*, 754 F. App’x 988, 996 (Fed. Cir. 2018) (finding that the introduction of “the end user” without an antecedent basis was not ambiguous and indefinite where the claim “only references one ‘end user,’ because “[w]ho else could the end user be?”).

The second ambiguity is the question of what is meant by “libraries”? To be sure, a lay-person might be confused by the term “libraries” without an introduction, particularly where claim

17 of the '762 Patent previously refers only to “local system files,” and “system files” are defined by the specification as “files provided within an operating system and which are available to applications as shared libraries and configuration files.” See '762 Patent at 5:10–12, 17:25–18:28. But a POSITA, *i.e.*, a person of ordinary skill in the art of “computer software,” (*see id.* at 1:26–32 (Field of the Invention)), would likely understand what libraries or library files refers to. This is the point at which Google’s failure to provide any evidence as to the specific understanding of a POSITA, *e.g.*, via expert testimony, is relevant – without such testimony here, Google has not shown by clear and convincing evidence that this term is indefinite, and the Court does not invalidate the claim on this basis.

The Court construes “the local operating system” to mean “the operating system,” *i.e.*, the operating system of the incompatible computer platform, and gives the term “libraries” its plain and ordinary meaning to one of skill in the art.

D. Disputed Terms Relevant to the '814 Patent¹⁵

1. Term 4 – “disparate computing environments”

Term, Claim, and Patent	VirtaMove’s Proposal	Google’s Proposal	Amazon’s Proposal	Court’s Ruling
“disparate computing environments” (’814 Patent, claim 1)	“Environments run by standalone or unrelated computers”	Indefinite	Indefinite	Indefinite

The parties’ arguments on this term agree on much. All parties agree that the patentee engaged in lexicography in defining this term, and the Court concurs. “Disparte computing environments” is defined to mean “Environments where computers are stand-alone or where there are plural computers and where they are unrelated.” ’814 Patent at 2:16–19 (using the introductory phrase: “The following definitions are used herein:”); *E.g.*, *Abbott Lab ’ys v. Andrx Pharms., Inc.*, 473 F.3d 1196, 1210 (Fed. Cir. 2007) (phrase “as used herein, means . . .” is “unambiguous” definitional language). Amazon and Google argue that, accepting this definition, there are two problems with the claim: the definition is either contradictory of the surrounding

¹⁵ The Court’s constructions of these terms are entered in both the *Google* and *Amazon* Actions.

claim language, or else it lacks objective boundaries. *Google* Action, ECF No. 63 at 12–13; *Amazon* Action, ECF No. 71 at 7–9. For the following reasons, the Court agrees.

First, in its briefs and at the hearing, VirtaMove conceded that the servers in the system are “related” (to whatever extent that term is understood). *See, e.g., Amazon* Action, ECF No. 77 at 5. But it urges that, rather than being contradictory, this simply means that the claim does not “extent to ‘unrelated computers, [and] only the ‘standalone portion of th[e] description [is] relevant to the scope of the claim.’” *Id.* The Court agrees with VirtaMove because the lexicography controls, and the lexicography is disjunctive.¹⁶ In other words, as VirtaMove argued at the hearing, if a claim recited an “orchard of apple trees bearing fruit, and then the specification defined fruit as apples or oranges,” the claim is not contradictory just because an apple tree cannot bear oranges. *See* October 1, 2026 Hrg. Tr. at 61:9–20. The claim is still cohesive to the extent it is limited to apples or, in this case, “standalone computers.”

But applying the “standalone” portion of the definition, *i.e.*, that the claim requires “[e]nvironments where computers are stand-alone,”¹⁷ the Court finds this term indefinite because it is either contradictory or lacks boundaries that can be discerned by a POSITA with “reasonable certainty.” “Standalone” would appear to mean computers that either are independent of one another or (as VirtaMove urges) that have “the ability” to “operate independently of each other.” *See Amazon* Action, ECF No. 77 at 5. The Court is persuaded by Amazon’s argument that the former understanding contradicts the plain language of the claims, because the servers at issue are not in fact independent of one another – they “form a single system,” as explained by Amazon’s technical expert, Dr. Long. *Amazon* Action, ECF No. 79 at 5–6 (citing *Amazon* Action, ECF No. 71-6 (“Long Report”) ¶ 36); *see also* Long Report, ¶¶ (explaining that, “[f]or multiple servers to

¹⁶ While, as the Court has explained, rendering a limitation meaningless is highly disfavored as a canon of claim construction, lexicography means that “[w]hen a patentee explicitly defines a claim term in the patent specification, the patentee’s definition controls.” *Martek Biosciences Corp. v. Nutrinova, Inc.*, 579 F.3d 1363, 1380 (Fed. Cir. 2009). This is true even if the definition “is contrary to the conventional meaning of the term.” *Id.*

¹⁷ The Court is not persuaded by Amazon’s argument that this definition cannot apply to the claim because it only applies in the case of a single computer, because the definition specifies “computers,” plural, that “are stand-alone.” ’814 Patent at 2:16–19.

1 form a single ‘system,’ a person of ordinary skill would understand that they must somehow be
2 connected (e.g., networked),” contradicting the requirement that “computers be ‘stand-alone’”).¹⁸

3 This leaves only VirtaMove’s contention that “stand-alone” is not contradictory to the
4 extent it is “used to indicate the ability of computers to operate independently of each other,” *i.e.*,
5 that the computers could be independently operated even if they are not in fact independent in the
6 claim. *Amazon* Action, ECF No. 77 at 5. There is some logic to this argument; as VirtaMove’s
7 counsel argued at the hearing, two laptops might be “standalone” computers even if, for example,
8 they are connected via a network. October 1, 2026 Hrg. Tr. at 78:24–79:7. But the problem of
9 such a definition for VirtaMove is that it has no boundaries. The specification of the ’814 Patent
10 uses the word “stand-alone” precisely *once*, in the lexicographical definition. None of the figures,
11 text, nor the prosecution history in the record, shed any light on what it means for a computer to be
12 “stand-alone” if it does not mean unconnected. The term only raises a host of questions: Do
13 computers need to be completely unconnected? What if they are connected over a network, but
14 could be disconnected and still run but not execute any applications? What if they are connected,
15 could be disconnected and still function, but lose some functionality (*e.g.*, be unable to run some
16 applications)? How much operation is required before a POSITA determines that a computer
17 cannot operate independently of other computers? The intrinsic evidence provides no guidance.
18 And Dr. Long testifies that “a person of ordinary skill would be unable to understand with
19 reasonable certainty what the patentee’s ‘stand-alone’ or ‘unrelated’ definition requires in the
20 context of the claimed system of multiple servers.” Long Report, ¶ 37.

21 Applying the patentee’s lexicography that “disparate computing environments” are
22 “[e]nvironments where computers are stand-alone or where there are plural computers and where
23 they are unrelated,” and even assuming as VirtaMove urges that the “unrelated” portion of the
24

25 ¹⁸ At the hearing, VirtaMove urged the Court to discount Dr. Long’s testimony. But VirtaMove
26 has not submitted any competing testimony as to the level of skill of a POSITA or a POSITA’s
27 understanding, and the Court credits the unrebutted testimony of Dr. Long in this respect. *E.g.*,
28 *Dyfan, LLC v. Target Corp.*, 28 F.4th 1360, 1367 (Fed. Cir. 2022) (explaining that “the district
court erred by ignoring key evidence—unrebutted deposition testimony from Target’s own expert,
Dr. Goldberg—regarding how a person of ordinary skill would have understood the
‘code’/‘application’ limitations.”).

definition simply drops out, the Court finds the “stand-alone” portion of the definition either contradictory or fatally ambiguous. So, the term “disparate computing environments” is indefinite and Amazon¹⁹ has shown, by clear and convincing evidence, that claim 1 of the ’814 Patent is invalid. Because independent claim 1 is invalid for indefiniteness, asserted dependent claims 2, 4, 6–10, and 13–14 are also invalid. *See Berkheimer v. HP Inc.*, 881 F.3d 1360, 1364 (Fed. Cir. 2018) (holding that because a claim term was indefinite, the independent claim and all its dependent claims were invalid).

2. Term 5 – “an operating system’s root file system”

Term, Claim, and Patent	VirtaMove’s Proposal	Google’s Proposal	Court’s Ruling
“an operating system’s root file system” (’814 Patent, Claim 1)	No construction necessary; plain and ordinary meaning.	Indefinite	“each of the operating systems’ root file systems,” wherein “the operating systems” are the “operating systems” of “the plurality of servers”

Finally, the ’814 Patent concludes with the requirement that “each of the containers” used by the invention must have “a unique root file system that is different from an operating system’s root file system.” ’814 Patent at 17:59–61. Google argues this term is indefinite because claim 1 recites “a system having a plurality of services with operating systems,” *plural*, so it is not clear which “operating system” is the comparator system to which the container’s root file system must be “different.” *Google Action*, ECF No. 63 at 20. Despite urging a “plain and ordinary meaning” construction, in its briefs, VirtaMove in fact argues that “an operating system” should be construed to mean “any” or “each” operating system, because a POSITA would understand that the invention requires that the container’s root file system be different from each of, *i.e.*, any of, the operating systems’ root file systems. ECF No. 65 at 15–16. The parties’²⁰ briefs on this issue

¹⁹ Unlike Amazon, Google did not submit an expert report and thus may have had more difficulty showing indefiniteness. However, “[g]iven the importance of uniformity in the treatment of a given patent,” the Court applies the same definition across both cases. *Cf. Finisar Corp.*, 523 F.3d at 1329 (Fed. Cir. 2008). Especially where the Court determines that a patent is invalid, it is not clear how the same patent could be invalid in one case but not another.

²⁰ Google and VirtaMove. Amazon did not take a position on this term.

were sparse and, at the hearing, the Court ordered the parties to provide, without further argument, supplemental authority on this issue. *See Google Action*, ECF No. 202 at 2.

Having reviewed the parties' authorities and arguments, the Court agrees with VirtaMove. While the phrase "an operating system's root file system" might be ambiguous standing alone, it would be clear to a POSITA—in light of the specification and surrounding claim language—that the term refers to "each of the operating systems' root file systems," *i.e.*, referencing the "plurality of servers with operating systems that differ." *See '814 Patent* at 1:30–61; *see also id.* at 4:43–46. Google's supplemental authority is unhelpful as Google refers to authority that largely cites to cases it already included, and all directed to problems where a definite article ("the" or "said") refers back to one item among a group of things, without specifying which one. *See eCardless Bancorp, Ltd. v. PayPal Inc.*, No. 24-cv-01054-BLF, 2024 WL 4804975, at *13–14 (N.D. Cal. Nov. 14, 2024) (citing and explaining, *inter alia*, *Baldwin Graphic Sys., Inc. v. Siebert, Inc.*, 512 F.3d 1338, 1343 (Fed. Cir. 2008) ("if two different levers are recited earlier in [a] claim, the recitation of 'said lever' in the same or subsequent claim would be unclear where it is uncertain which of the two levers was intended.")). Nor does Google contend that VirtaMove's proposed construction is unsupported or unreasonable, but instead argues that the construction raises new problems (an argument with which the Court disagrees). *Google Action*, ECF No. 66 at 10 (complaining that VirtaMove's proposal does not specify "the universe of 'any operating systems' to be analyzed"; this argument wholly ignores that the universe of "operating systems" is introduced in the first line of claim 1).

Google has failed to show by clear and convincing evidence that this term is indefinite. The Court construes "an operating system's root file system" to mean "each of the operating systems' root file systems," wherein "the operating systems" are the "operating systems" of "the plurality of servers" of claim 1. *See '814 Patent* at 17:30–31.

IV. CONCLUSION

The disputed terms are construed as follows:

Term, Claim, and Patent	Court's Ruling
“an incompatible computer platform in which the first software application is not intended to execute” (’762 Patent, claim 17 and its dependent claims)	Indefinite as to “an incompatible” Indefinite as to an application “not intended to execute” Construed in the <i>Google</i> Action
“the second incompatible computer platform” (’762 Patent, claim 17 and its dependent claims)	“the incompatible computer platform” Construed in the <i>Google</i> Action
“the local operating system libraries” (’762 Patent, claim 17 and its dependent claims)	“the local operating system” means “the operating system,” <i>i.e.</i> , the operating system of the incompatible computer platform; The Court gives the term “libraries” its plain and ordinary meaning. Construed in the <i>Google</i> Action
“disparate computing environments” (’814 Patent, claim 1 and its dependent claims)	Indefinite Construed in both actions.
“an operating system’s root file system” (’814 Patent, claim 1 and its dependent claims)	“each of the operating systems’ root file systems,” wherein “the operating systems” are the “operating systems” of “the plurality of servers” Construed in both actions.

Because Google and Amazon have shown by clear and convincing evidence that two terms in claim 17 of the ’762 Patent and one term in claim 1 of the ’814 Patent are indefinite, the Court finds that those claims are invalid. And, because those claims are invalid for indefiniteness, all claims that depend from them are also invalid.

VirtaMove and the Defendant in each action shall meet and confer as to the appropriate next steps in each action. *See, e.g., Fintiv, Inc. v. Paypal Holdings, Inc.*, No. 23-cv-00490-ADA, 2023 WL 5423078, at *1 (W.D. Tex. July 28, 2023), *aff’d*, 134 F.4th 1377 (Fed. Cir. 2025) (entering final judgment after issuing claim construction order finding all asserted claims invalid). The Court sets a status conference for **October 27, 2026 at 10:30 a.m.** in both actions.

IT IS SO ORDERED.

Dated: October 13, 2026


 Noël Wise
 United States District Judge