IN THE UNITED STATES DISTRICT COURT
FOR THE EASTERN DISTRICT OF TEXAS
TYLER DIVISION

BLUE SPIKE, LLC,	§	
	§	
	§	
Plaintiff,	§	**CIVIL ACTION NO. 6:16-CV-00430**
	§	**JDL**
	§	
v.	§	
	§	
TOSHIBA AMERICA, INC. et al.,	§	
	§	
Defendants.	§	

MEMORANDUM OPINION AND ORDER

On May 13, 2016, Plaintiff filed this action for patent infringement. Plaintiff alleges that Defendants Toshiba America, Inc., Toshiba Corporation, and Toshiba America Information Systems, Inc. (collectively, "Toshiba") infringe claims of U.S. Patent No. 5,745,569 ("the '569 Patent") and U.S. Patent No. 8,930,719 ("the '719 Patent") (collectively, "Asserted Patents"). (*See* Doc. No. 6 (Amended Complaint).)

This claim construction opinion construes disputed claim terms in the Asserted Patents. Blue Spike has filed an Opening Claim Construction Brief (Doc. No. 29), Defendants have filed a Response (Doc. No. 32), and Blue Spike has filed a Reply (Doc. No. 33). The parties additionally submitted a Joint Claim Construction Chart pursuant to P.R. 4-5(d). (Doc. No. 34.) On April 21, 2017, the Court held a claim construction hearing. Upon consideration of the parties' arguments and for the reasons stated herein, the Court adopts the constructions set forth below.

OVERVIEW OF THE PATENTS

The Asserted Patents generally relate to improving computer security through digital watermarking and relocating all or portions of software in memory. *See* '569 Patent, at Abstract, 2:21–22; '719 Patent, at 1:21–23. The '569 Patent, entitled "METHOD FOR STEGA-CIPHER PROTECTION OF COMPUTER CODE," issued on April 28, 1998 from an application filed Jan. 17, 1996. The '719 Patent, entitled "DATA PROTECTION METHOD AND DEVICE," issued on Jan. 6, 2015 and claims priority to series of parent applications, the earliest filed March 24, 1998. The '719 Patent does not claim priority to the '569 Patent. However, substantially all of the specification of the '569 Patent appears in the specification of the '719 Patent.[1] Blue Spike asserts Claims 16–20 of the '569 Patent and Claims 10–12 and 22 of the '719 Patent. (*See* Doc. No. 36.) Claim 16 of the '569 Patent is an independent claim and Claims 17–20 depend from Claim 16. Claim 16 of the '569 Patent recites:

> 16. A method for copy protecting a software application executed by a computer system, the software application including a plurality of executable code resources loaded in a memory of the computer system, said method comprising the steps of:
>
> determining an address within the memory of the computer system associated with each of the plurality of executable code resources; and
>
> intermittently relocating each of the plurality of executable code resources to a different address within the memory of the computer during execution of the software application.

'569 Patent, at 10:9–19.

Claims 10 and 22 of the '719 Patent are independent claims and Claims 11 and 12 depend from Claim 10. Claim 10 recites:

> 10. A system for executing application software code, comprising:
>
> a memory designed to store data in non transitory form, and storing executable code resources;

[1] For common portions of the specification, the Court cites to the '569 Patent.

wherein said executable code resources comprise a memory scheduler and other executable code resources;
wherein said memory scheduler is designed to shuffle said other executable code resources in said memory; and
wherein said memory scheduler is designed to modify a stack frame in said memory.

'719 Patent, at 16:60–17:2.

Claim 22 of the '719 Patent recites:

22. A system comprising:
a processor designed to process instructions;
a memory designed to store data in non transitory form;
wherein said processor is coupled to said memory;
wherein said system is configured to load an executable program comprising at least two code resources into said memory and to randomize the location of at least one of the at least two code resources in the said memory, using said processor; and
wherein one of the at least two code resources is designed to modify a stack frame in said memory.

'719 Patent, at 17:52–62.

The Court has previously construed terms of the '569 Patent claims in *Blue Spike v. Huawei*, No. 6:13-cv-679-RWS, Doc. No. 194 (E.D. Tex. May 16, 2016) ("*Huawei* Order").

LEGAL STANDARD

"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) (quoting *Innova/Pure Water, Inc. v. Safari Water Filtration Sys., Inc.*, 381 F.3d 1111, 1115 (Fed. Cir. 2004)). The Court examines a patent's intrinsic evidence to define the patented invention's scope. *Id.* at 1313–14; *Bell Atl. Network Servs., Inc. v. Covad Commc'ns Group, Inc.*, 262 F.3d 1258, 1267 (Fed. Cir. 2001). Intrinsic evidence includes the claims, the rest of the specification, and the prosecution history. *Phillips*, 415 F.3d at 1312–13; *Bell Atl. Network Servs.*, 262 F.3d at 1267. The Court gives claim terms their ordinary and customary meaning as understood by one of ordinary skill in the art at the time of

the invention. *Phillips*, 415 F.3d at 1312–13; *Alloc, Inc. v. Int'l Trade Comm'n*, 342 F.3d 1361, 1368 (Fed. Cir. 2003). Claim language guides the Court's construction of claim terms. *Phillips*, 415 F.3d at 1314. "[T]he context in which a term is used in the asserted claim can be highly instructive." *Id.* Other claims, asserted and unasserted, can provide additional instruction because "terms are normally used consistently throughout the patent." *Id.* Differences among claims, such as additional limitations in dependent claims, can provide further guidance. *Id.*

"[C]laims 'must be read in view of the specification, of which they are a part.'" *Id.* (quoting *Markman v. Westview Instruments, Inc.*, 52 F.3d 967, 979 (Fed. Cir. 1995)). "[T]he specification 'is always highly relevant to the claim construction analysis. Usually, it is dispositive; it is the single best guide to the meaning of a disputed term.'" *Id.* (quoting *Vitronics Corp. v. Conceptronic, Inc.*, 90 F.3d 1576, 1582 (Fed. Cir. 1996)); *Teleflex. Inc. v. Ficosa N. Am. Corp.*, 299 F.3d 1313, 1325 (Fed. Cir. 2002). In the specification, a patentee may define his own terms, give a claim term a different meaning than it would otherwise possess, or disclaim or disavow some claim scope. *Phillips*, 415 F.3d at 1316. Although the Court generally presumes terms possess their ordinary meaning, this presumption can be overcome by statements of clear disclaimer. *See SciMed Life Sys., Inc. v. Advanced Cardiovascular Sys., Inc.*, 242 F.3d 1337, 1343–44 (Fed. Cir. 2001). This presumption does not arise when the patentee acts as his own lexicographer. *See Irdeto Access, Inc. v. EchoStar Satellite Corp.*, 383 F.3d 1295, 1301 (Fed. Cir. 2004).

The specification may also resolve ambiguous claim terms "where the ordinary and accustomed meaning of the words used in the claims lack sufficient clarity to permit the scope of the claim to be ascertained from the words alone." *Teleflex, Inc.*, 299 F.3d at 1325. For

example, "[a] claim interpretation that excludes a preferred embodiment from the scope of the claim 'is rarely, if ever, correct." *Globetrotter Software, Inc. v. Elan Computer Group Inc.*, 362 F.3d 1367, 1381 (Fed. Cir. 2004) (quoting *Vitronics Corp.*, 90 F.3d at 1583). But, "[a]lthough the specification may aid the court in interpreting the meaning of disputed language in the claims, particular embodiments and examples appearing in the specification will not generally be read into the claims." *Constant v. Advanced Micro-Devices, Inc.*, 848 F.2d 1560, 1571 (Fed. Cir. 1988); *see also Phillips*, 415 F.3d at 1323.

The prosecution history is another tool to supply the proper context for claim construction because a patentee may define a term during prosecution of the patent. *Home Diagnostics Inc. v. LifeScan, Inc.*, 381 F.3d 1352, 1356 (Fed. Cir. 2004) ("As in the case of the specification, a patent applicant may define a term in prosecuting a patent."). The well-established doctrine of prosecution disclaimer "preclud[es] patentees from recapturing through claim interpretation specific meanings disclaimed during prosecution." *Omega Eng'g Inc. v. Raytek Corp*., 334 F.3d 1314, 1323 (Fed. Cir. 2003). The prosecution history must show that the patentee clearly and unambiguously disclaimed or disavowed the proposed interpretation during prosecution to obtain claim allowance. *Middleton Inc. v. 3M Co.*, 311 F.3d 1384, 1388 (Fed. Cir. 2002); *see also Springs Window Fashions LP v. Novo Indus., L.P.*, 323 F.3d 989, 994 (Fed. Cir. 2003) ("The disclaimer . . . must be effected with 'reasonable clarity and deliberateness.'") (citations omitted)). "Indeed, by distinguishing the claimed invention over the prior art, an applicant is indicating what the claims do not cover." *Spectrum Int'l v. Sterilite Corp.*, 164 F.3d 1372, 1378–79 (Fed. Cir. 1988) (quotation omitted). "As a basic principle of claim interpretation, prosecution disclaimer promotes the public notice function of the intrinsic evidence and protects the public's reliance on definitive statements made during prosecution."

Omega Eng'g, Inc., 334 F.3d at 1324. Statements in the prosecution history that are subject to multiple reasonable interpretations do not constitute a clear and unmistakable departure from the ordinary meaning of a claim term. *Golight, Inc. v. Wal-Mart Stores, Inc.*, 355 F.3d 1327, 1332 (Fed. Cir. 2004).

Although "less significant than the intrinsic record in determining the legally operative meaning of claim language," the Court may rely on extrinsic evidence to "shed useful light on the relevant art." *Phillips*, 415 F.3d at 1317 (quotation omitted). Technical dictionaries and treatises may help the Court understand the underlying technology and the manner in which one skilled in the art might use claim terms, but such sources may also provide overly broad definitions or may not be indicative of how terms are used in the patent. *Id.* at 1318. Similarly, expert testimony may aid the Court in determining the particular meaning of a term in the pertinent field, but "conclusory, unsupported assertions by experts as to the definition of a claim term are not useful." *Id.* Generally, extrinsic evidence is "less reliable than the patent and its prosecution history in determining how to read claim terms." *Id.*

In patent construction, "subsidiary fact finding is sometimes necessary" and the court "may have to make 'credibility judgments' about witnesses." *Teva v. Sandoz*, 135 S.Ct. 831, 838 (2015). In some cases, "the district court will need to look beyond the patent's intrinsic evidence and to consult extrinsic evidence in order to understand, for example, the background science or the meaning of a term in the relevant art during the relevant time period." *Id.* at 841. "If a district court resolves a dispute between experts and makes a factual finding that, in general, a certain term of art had a particular meaning to a person of ordinary skill in the art at the time of the invention, the district court must then conduct a legal analysis: whether a skilled artisan would ascribe that same meaning to that term *in the context of the specific patent claim under*

review." *Id.* (emphasis in original). When the court makes subsidiary factual findings about the extrinsic evidence in consideration of the "evidentiary underpinnings" of claim construction, those findings are reviewed for clear error on appeal. *Id.*

A patent claim may be expressed using functional language. *See* 35 U.S.C. § 112, ¶ 6; *Williamson v. Citrix Online, LLC*, 792 F.3d 1339, 1347–49 & n.3 (Fed. Cir. 2015) (en banc in relevant portion). Section 112, paragraph 6,[2] provides that a structure may be claimed as a "means . . . for performing a specified function" and that an act may be claimed as a "step for performing a specified function." *Masco Corp. v. United States*, 303 F.3d 1316, 1326 (Fed. Cir. 2002).

But section 112, paragraph 6 does not apply to all functional claim language. There is a rebuttable presumption that section 112, paragraph 6 applies when the claim language includes "means" or "step for" terms, and that it does not apply in the absence of those terms. *Masco Corp.*, 303 F.3d at 1326; *Williamson*, 792 F.3d at 1348. The presumption stands or falls according to whether one of ordinary skill in the art would understand the claim with the functional language, in the context of the entire specification, to denote sufficiently definite structure or acts for performing the function. *See Media Rights Techs., Inc. v. Capital One Fin. Corp.*, 800 F.3d 1366, 1372 (Fed. Cir. 2015) (§ 112, ¶ 6 does not apply when "the claim language, read in light of the specification, recites sufficiently definite structure" (quotation marks omitted) (citing *Williamson*, 792 F.3d at 1349; *Robert Bosch, LLC v. Snap-On Inc.*, 769 F.3d 1094, 1099 (Fed. Cir. 2014))); *Williamson*, 792 F.3d at 1349 (§ 112, ¶ 6 does not apply when "the words of the claim are understood by persons of ordinary skill in the art to have sufficiently definite meaning as the name for structure"); *Masco Corp.*, 303 F.3d at 1326 (§ 112,

[2] The America Invents Act renumbered section 112, paragraph 6 to section 112(f). However, because each of the patents at issue in this case was originally filed before September 16, 2012, the Court will refer to this code section by its previous numbering, section 112, paragraph 6.

¶ 6 does not apply when the claim includes an "act" corresponding to "how the function is performed"); *Personalized Media Commc'ns, L.L.C. v. Int'l Trade Comm'n*, 161 F.3d 696, 704 (Fed. Cir. 1998) (§ 112, ¶ 6 does not apply when the claim includes "sufficient structure, material, or acts within the claim itself to perform entirely the recited function . . . even if the claim uses the term 'means.'" (quotation marks and citation omitted)).

DISCUSSION

I. AGREED CLAIM TERM CONSTRUCTIONS

In its reply brief, Blue Spike agreed to Toshiba's proposed construction of the terms "a software application/application" as "a software program run by an operating system." (Doc. No. 33, at 2–3.) The term "a software application" appears in Claims 16 and 17 of the '569 Patent and the term "application" appears in Claim 10 of the '719 Patent. In light of the parties' agreement and a review of the asserted claims, specifications, and prosecution history, the Court adopts Defendant's proposed construction.

II. DISPUTED CLAIM TERMS

a. program

Claim Term	Plaintiffs' Proposal	Defendants' Proposal
"program" ('719 patent, Claim 22)	Plain and ordinary	"a set of instructions run by an operating system" <u>Alternate construction:</u> "A software program run by an operating system"

The sole dispute between the parties with respect to the term "program" (occurring in the context of Claim 22 of the '719 Patent as "an executable program . . .") is whether it

encompasses an operating system. Blue Spike argues, without citation, that the '719 Patent uses "program" in its plain and ordinary sense. (Doc. No. 29, at 17.) Blue Spike objects to Toshiba's construction as "an attempt to distinguish the operating system as something separate from a program." (*Id.*) According to Blue Spike, an operating system is a program "developed by software developers just as a program is developed." (*Id.*)

Toshiba argues that the claims refer to a system configured to "load an executable program" into memory. *See* '719 Patent, Claim 22. Toshiba argues that in the *Huawei* Order, the Court found that loading preceded execution. (Doc. No. 32, at 8 (citing *Huawei* Order, at 7).) Toshiba contends that it is "without question" that the operating system loads the executable program. (*Id.*) Therefore, Toshiba argues, the operating system must be part of the system of Claim 22 and not a loadable program. (*Id.*) Toshiba further contends that the specification consistently draws a distinction between an executable program and an operating system. (*Id.* (citing '569 Patent, at 7:30-34 ("[o]nce the code resources of a program are loaded into memory, they typically remain in a fixed position, unless the computer operating system finds it necessary to rearrange certain portions of memory during 'system time,' when the operating system code, not application code is running."); 7:38-39 ("in order to allow the operating system to rearrange memory transparently, underneath a running program.")).) Toshiba also argues that the specification equates "program" and "application" by explaining to an engineer that a "computer program is variously referred to as an application, from the point of view of a user." (*Id.* at 9 (citing '569 Patent, at 3:44-45).)

Blue Spike replies by noting that in the *Huawei* Order, the Court drew a distinction between "applications" and "programs." (Doc. No. 33, at 3.) In *Huawei,* the Court found that "it is more accurate to say that the intrinsic record treats applications as a subset of a computer

programs [sic] rather than equating the two." *Huawei* Order, at 10–11. Blue Spike thus contends that both an operating system and a software application are types of programs. (Doc. No. 33, at 3.) Blue Spike further argues, contrary to some of Toshiba's assertions, that "[a]n operating system is executed by a computer just as applications are." (*Id.* at 4.) Blue Spike reiterates that a person of ordinary skill in the art would agree that an operating system is a program. (*Id.*)

The parties do not dispute that in the abstract, an operating system is a program. The issue is whether the term "executable program" includes operating systems in the context of the Asserted Patents. Claim 22 of the '719 Patent recites, *inter alia*, "a system . . . wherein said system is configured to load an executable program comprising at least two code resources . . . ." Toshiba argues, without support from a person of skill in the art, that in order to practice the claim, the operating system must already be part of the claimed system such that it can load other programs. Blue Spike argues, without support from a person of skill in the art, that the claimed system does not require an operating system and indeed can load and execute an operating system the same as any other program. Based solely on a review of the claim language and the parties' conflicting attorney argument, it is entirely unclear whether the patent applicants intended to limit the term "executable program" to exclude operating systems.

The specification, on the other hand, does draw a clear distinction between an executable, or running, program and the operating system. *See* '569 Patent, at 7:34–39 ("[t]he MacOS for example, uses Handles, which are double-indirect pointers to memory locations, ***in order to allow the operating system to rearrange memory transparently, underneath a running program***.") Indeed, the specification equates the term "program" with the term "application," stating, "[a]n executable computer program is variously referred to as an application, from the

point of view of a user, or executable object code from the point of view of the engineer." *See, e.g.*, '569 Patent, at 3:44–47; *see also id.* at 1:50–52 ("Computer ***application programs*** can be watermarked . . ."); *id.* at 2:21–26; *id.* at 2:27–31 ("In this way, attempts to capture memory to determine underlying functionality or provide a 'patch' to facilitate unauthorized use of the ***'application,' or computer program***, without destroying the functionality and thus usefulness ***of a copyrightable computer program*** can be made difficult."); *id.* at 5:40–6:8 (interchanging the terms "program," "application," "application program," "executable program," and "executable application"); (Doc. No. 29, at 12). In turn, as the parties agree, the specification clearly distinguishes between an "application" and an "operating system." '569 Patent, at 7:30–34 (explaining that after code resources are loaded into memory, the computer operating system may find it necessary to rearrange portions of memory during "'system time,' when the operating system code, ***not application code***, is running."); '719 Patent, Claim 1 (claiming a computing device with an operating system and memory that stores an application software). Based on the specification's disclosure, the Court finds that the applicant clearly disclaimed the full scope of the term "program." In the context of the Asserted Patents, an "executable program" does not include an operating system.

Moreover, Claim 22 of the '719 Patent recites a system configured to "load an executable program." The specification never references "loading" in the context of initially loading an operating system. Rather, "loading" is used with reference to a program or application being loaded by the operating system. *See, e.g.*, '569 Patent, at 7:30–34; *see also id.* at 4:61–67 ("there are a certain sub-set of executable code resources, that comprise an application and that are 'essential' to the proper function of the application. In general, any code resource can be considered 'essential' in that *if the program proceeds to a point where it must 'call' the code*

resource and the code resource is not present in memory, ***or cannot be loaded****, then the program fails*.") This disclosure further indicates that within the context of the Asserted Patents, an executable program does not include an operating system.

Blue Spike refers to the *Huawei* Order to argue that this Court previously found software programs are not equivalent to software applications. *See Huawei* Order, at 10–11 ("the intrinsic record treats applications as a subset of [] computer programs rather than equating the two"). However, the *Huawei* Order did not construe the term "program," or specifically address whether the term "program" in the context of the patent specification includes operating systems. Further, *Huawei*'s conclusion was solely based on drawing a distinction between some claim preambles that refer to "copy protection of computer software" compared to others that refer to "copy protecting a software application." *Compare* '569 Patent, Claim 1 *with id.* at Claim 16. This comparison does not actually address the comparison of "program" to "application," but rather refers to "software" versus "application." Further, while there is a "general presumption that different terms have different meanings," the numerous portions of the specification, cited above, that use the terms "program" and "application" interchangeably overcome this presumption. *See CAE Screenplates, Inc. v. Heinrich Fiedler GmbH & Co. KG,* 224 F.3d 1308, 1317 (Fed.Cir.2000); *Fargo Elecs., Inc. v. Iris Ltd., Inc.*, No. 04-1017 JRT/FLN, 2005 WL 3241851, at *14 (D. Minn. Nov. 30, 2005), aff'd, 287 F. App'x 96 (Fed. Cir. 2008); *see, e.g.*, '569 Patent, at 3:44–47 ("[a]n executable computer program is variously referred to as an application. . . .").

Accordingly, the Court construes the term "a program" as "a set of instructions run by an operating system."

b. memory scheduler

Claim Term	Plaintiffs' Proposal	Defendants' Proposal
"memory scheduler" ('719 patent, Claim 10, 11, 12; '569 Patent, Claim 17, 18, 19, 20)**;**	Plain and ordinary	Means-plus-function limitation (Indefinite) Claimed Function: During execution time (i) shuffle or randomize other code resources in memory; (ii) modify a stack frame in memory; (iii) modify a value stored by a program counter; (iv) modify a calling address; (v) copy itself to a memory location associated with a calling address; (vi) maintain a list of addresses; (vii) shuffle itself randomly in memory. There must be a disclosed algorithm for performing the recited functions; if not, then it is indefinite. No corresponding structure is disclosed; there is no algorithm; therefore, the term is indefinite. <u>Alternate construction:</u> memory scheduler, wherein the memory scheduler is within the/a software application

Blue Spike contends, without citation, that a "scheduler" is "[a] computer program designed to perform functions such as scheduling, initiation, and termination of jobs." (Doc. No. 29, at 13.) Blue Spike states a "memory scheduler" specifically is "a scheduler of RAM or a 'memory scheduler.'" (*Id.*)

Toshiba argues that the terms "memory scheduler code resource" and "memory scheduler" are means-plus-function limitations for which both Asserted Patents fail to disclose

any corresponding structure. (Doc. No. 32, at 16.) Toshiba contends that the term "scheduler" is a nonce term that functions as a "black box." (*Id.*) Toshiba contends the Asserted Patents' only description of the "memory scheduler" simply introduces the term as a "'***special***" code resource." (*Id.* at 16 (emphasis in original) (citing '569 Patent, at 8:1–19).) Further, Toshiba argues that the fact that the applicants introduced the term "memory scheduler" using quotation marks shows that it is effectively a nonce word without established meaning. (*Id.*) Toshiba contends that in the context of software technology, Blue Spike's definition of "scheduler" confirms that "memory scheduler" has no "well understood structural meaning in the computer technology field." (*Id.* at 17.) Toshiba notes that Blue Spike has provided no evidence as to how one skilled in the art would interpret "memory scheduler" to connote structure. (*Id.*)

On reply, Blue Spike contends that "memory scheduler" is itself sufficiently descriptive; it is not a non-structural term. (Doc. No. 33, at 4–5 (citing *Smartflash LLC v. Apple Inc.*, 77 F. Supp. 3d 535, 541 (E.D. Tex. 2014) (noting that the term "processor" is "not a generic nonstructural term such as 'means', 'element', and 'device' that typically do not connote sufficient structure")).) Blue Spike contends that unlike "module" in *Williamson*, the term "scheduler" is readily understood in the art: "scheduler n. A computer program designed to perform functions such as scheduling, initiation, and termination of jobs." (*Id.* at 5 (quoting Doc. No. 33, Ex. 8 (Dictionary of IBM & Computing Terminology) at 80).) Blue Spike argues that the intrinsic record fully supports its proposed construction of "memory scheduler." (*Id.* at 5–6.)

Toshiba's argument that the term "scheduler" lacks a known meaning in the computer technology field is belied by the fact that the term is specifically defined in computer dictionaries. The Dictionary of IBM & Computing Terminology specifically defines a "scheduler" as a type of computer program that performs certain functions. (Doc. No. 33, Ex. 8.)

This demonstrates that the term "scheduler," like the term "processor," has a generally understood meaning in the art. Toshiba argues that "the performance of functions by a computer program is nothing more than a generic phrase," thus rendering this dictionary definition meaningless. (Doc. No. 32, at 18.) Toshiba, however, does not rely on an expert declaration or any other recitation of a person of skill in the art to support this argument. Nor does Toshiba cite to caselaw holding that a known class of computer programs fails to have sufficient structure. Like "circuit" or "processor," the definition of "scheduler" as "a computer program designed to perform functions such as scheduling, initiation, and termination of jobs" implicates a general class of "structures," *i.e.* certain types of computer programs. *See Linear Tech. Corp. v. Impala Linear Corp.*, 379 F.3d 1311, 1320 (Fed. Cir. 2004); *Syncpoint Imaging, LLC v. Nintendo of Am. Inc.*, No. 215-CV-00247-JRG-RSP, 2016 WL 55118, at *19–21 (E.D. Tex. Jan. 5, 2016).

Blue Spike does not appear to dispute that the term "memory scheduler" in the context of the patents is a coined term. Toshiba, meanwhile, argues that because the term "memory scheduler" is a coined term (and the term "scheduler" allegedly fails to connote structure), "memory scheduler" must be a nonce word. Just because a term is coined, however, does not automatically make it a nonce word. A review of the claims indicates that the patent applicants refer to a "memory scheduler" as a type of "scheduler" as that term is understood in the field of computer technology.

For instance, Claim 10 of the '719 Patent recites, *inter alia*, "wherein said executable code resources comprise a memory scheduler and other executable code resources . . . wherein said memory scheduler is designed to shuffle said other executable code resources in said memory; and wherein said memory scheduler is designed to modify a stack frame in said memory." Similarly, Claim 1 of the '719 Patent recites, *inter alia*, "wherein said application

software comprises (1) a memory scheduler code resource and (2) other code resources; wherein said application software is designed to call said memory scheduler code resource . . . ." The '569 Patent claims refer to a "memory scheduler" only in the context of dependent claims, but likewise indicate that a memory scheduler is a unique computer program. *See, e.g.*, '569 Patent, Claim 17 ("The method of claim 16, further comprising the step of modifying the software application to direct a call to an executable code resource to a memory scheduler."); *id.* at Claim 18 ("The method of claim 17, wherein said step of intermittently relocating each of the plurality of executable code resources is performed in response to a call to an executable code resource received by the memory scheduler."). The claims provide a sufficient description of the memory scheduler's operations such that, in combination with the structure-connoting term "scheduler," sufficient structural meaning is conveyed to a person of skill in the art. *See Linear Tech. Corp. v. Impala Linear Corp.*, 379 F.3d 1311, 1320 (Fed. Cir. 2004) (finding that the term "circuit" had an understood meaning in the art and was not a means-plus-function term when it was coupled with sufficient description of the circuit's operation).

The specification further describes the term "memory scheduler":

> Under the present invention, ***the application contains a special code resource which knows about all the other code resources in memory***. During execution time, this special code resource, called ***a "memory scheduler," can be called periodically, or at random or pseudo random intervals, at which time it intentionally shuffles the other code resources randomly in memory***, so that someone trying to analyze snapshots of memory at various intervals cannot be sure if they are looking at the same code or organization from one "break" to the next. This adds significant complexity to their job. The scheduler also randomly relocates itself when it is finished. In order to do this, the scheduler would have to first copy itself to a new location, and then specifically modify the program counter and stack frame, so that it could then jump into the new copy of the scheduler, but return to the correct calling frame. Finally, the scheduler would need to maintain a list of all memory addresses which contain the address of the scheduler, and change them to reflect its new location.

'569 Patent 8:1–19 (emphasis added). As evident from this passage, the applicants first introduce the "memory scheduler" and subsequently refer to it as simply a "scheduler", again showing that the applicants intend a "memory scheduler" to be a type of "scheduler" as that term is understood in the field of computer technology. The patent applicants also specifically define a "memory scheduler" as special code resources of an application, which knows about other code resources in memory. Further, the applicants indicate that a "memory scheduler" can adjust the other code resources. *See id.* This disclosure provides sufficient information such that a person of ordinary skill in the art would understand the "structure" associated with the term "memory scheduler." Although Toshiba appears to challenge the specification's definition of the memory scheduler as a type of special code resource, the parties did not request construction of the term "special code resource," a term that also appears in the claims.[3]

Accordingly, the Court finds that the term "memory scheduler" is not subject to § 112 ¶ 6 and should be construed as "special code resources of an application, which knows about other code resources in memory."

c. copy protecting a software application

Claim Term	Plaintiffs' Proposal	Defendants' Proposal
"copy protecting a software application" ('569 Patent, Claim 16)	Plain and ordinary Alternate construction: "protecting against the unauthorized copying, unauthorized use, or unintended use of a software application"	"protecting a software application from unauthorized copying or the use of unauthorized copies"

[3] The term "special code resource" was construed in the *Huawei* Order as "one or more compiled functions or procedures." *Huawei* Order, at 12.

The main dispute between the parties is whether "copy protection" is limited to protecting a software application from the unauthorized copying or use of unauthorized copies. Blue Spike contends that copy protection in the context of the Asserted Patents is broader and also includes "attempts at memory capture or object code analysis," "modifying, in an unintended manner, the functioning of the application," "attacks at disabling the system," and "attempts to tamper" with the software. (Doc. No. 29, at 5–6 (citing '569 Patent, 2:21-26; 3:37-40).) Blue Spike states that copy protection also applies to original works, as evidenced by the fact that it is alternatively referred to as "content protection" or "copyright protection." (*Id.* at 6.) Blue Spike further argues that the Court in *Huawei* mistook Blue Spike's statements at the Markman Hearing for full agreement to the defendant's proposed construction in that case—the same construction as Toshiba's proposed construction here—when there was "no agreement." (*Id.*) Blue Spike argues, without explanation, that the construction in *Huawei* has "proven inadequate." (*Id.*)

Toshiba contends that Blue Spike agreed to the construction proposed and adopted by the Court in the *Huawei* case. (Doc. No. 32, at 10.) Toshiba contends that the '569 Patent uses the term "copy protecting" in accordance with its plain and ordinary meaning, *i.e.*, to protect a software application from unauthorized copying. (*Id.* (citing '569 Patent, at 4:49–53, 5:51–57, 713–17 ("distribution and exchange of content would be made more secure from unauthorized copying"; the disclosed invention prevents "unauthorized copies" and prevents hackers from "forc[ing] the application program to run as an unauthorized copy.").) Toshiba argues that the '569 Patent incorporates by reference a patent that describes "copy protection" as preventing unauthorized copies. '569 Patent 2:34–44 (citing U.S. Patent No. 5,613,004); *see also* '004 Patent, at 2:57–60. Toshiba also contends that during prosecution of the '569 Patent the

applicants argued that "a significant benefit of the claimed invention" is that it protects software from unauthorized copying. (Doc. No. 32, at 11 (citing Doc. No. 32, Ex. 8 ('569 Patent prosecution history, Response to Office Action dated July 23, 1997), at 2–3).) Toshiba contends that Blue Spike provides no explanation as to why the "prior [*Huawei*] construction has proven inadequate." (*Id*. at 11-12.)

On reply, Blue Spike reiterates that Toshiba's proposal is too limiting. (Doc. No. 33, at 6–7 (citing '569 Patent, at 2:21-26).) Blue Spike also reiterates that the Court in *Huawei* "misperceived an agreement between the parties." (*Id.* at 7 n.10.) Blue Spike contends that during the Markman Hearing in the *Huawei* case, Blue Spike continued to argue that defendants' construction misconstrued "'copy' as meaning 'copying'" rather than "copy as the copy of the software." (*Id*.)

In *Huawei*, the Court found the term "copy protecting a software application" should be construed (as opposed to being left with no construction) in order avoid jury confusion. The *Huawei* Order construed the term as "protecting a software application from unauthorized copying or the use of unauthorized copies" based on a perceived agreement between the parties at the Markman Hearing. *Huawei* Order, at 5. In other words, the Court ultimately deemed it was unnecessary to conduct a detailed analysis of the intrinsic record with respect to this term. Meanwhile, in this case, the parties have clearly maintained disagreement with respect to their understanding of the proper construction of the term "copy protecting a software application."[4] Thus, the Court proceeds to conduct its own analysis of the record in determining its construction of this term. Whether or not Blue Spike actually agreed to the construction in the *Huawei* Order is inconsequential to the analysis.

[4] Blue Spike has not objected to construction of this term on the basis that it appears in the preamble of Claim 16. Thus, the Court does not consider this issue in its analysis.

The ’569 Patent specification indicates that “copy protecting a software application” has a broader meaning than simply protecting a software application from unauthorized copying or the use of unauthorized copies. For example, the specification states:

> It is also desirable to randomly reorganize program memory structure intermittently during program run time, to prevent attempts at memory capture or object code analysis aimed at eliminating licensing or ownership information, or otherwise modifying, in an unintended manner, the functioning of the application.

’569 Patent, at 2:21-26. In other words, the ’569 Patent contemplates that a purpose of the invention is to prevent hackers from analyzing or modifying key security portions of software code. There is no indication that the Asserted Patents are solely limited to protecting against analyzing or modifying *copies* of software. The specification likewise states:

> Note that the application can be copied in an uninhibited manner, but must contain the license code issued to the licensed owner, to access its essential code resources. The goal of the invention, copyright protection of computer code ***and*** establishment of responsibility for copies, is thus accomplished.

’569 Patent, at 6:32–37 (emphasis added). Other passages of the specification similarly refer to “modifying, in an unintended manner, the functioning of the application,” “attacks at disabling the system,” and “attempts to tamper” with software code. ’569 Patent, at 2:21–26; 3:37–40. Thus, again, the ’569 Patent contemplates that copy protection requires protection of the original code itself, not just “responsibility for copies.”

Despite the fact that Blue Spike cites many of these portions of the specification in its opening brief, Toshiba does not address them in its responsive claim construction brief. Instead, Toshiba argues that (1) “[t]he ’569 patent specification uses [‘copy protecting’] in accordance with its plain and ordinary meaning,” and (2) that plain and ordinary meaning is “to protect a software application from unauthorized copying.” (Doc. No. 32, at 10.) Based on a review of the specification, including the passages cited above, these arguments are unpersuasive.

Nor does the prosecution history of the '569 Patent provide a basis to limit the term "copy protection" as Toshiba requests. In distinguishing a prior art reference, Houser, the patent applicants stated:

> Houser discloses an electronic document verification system…thereby enabling a user to 'sign' the document. … Unlike Houser, which deals only with the ***protection of electronic documents***…claim 13 is directed to a method for ***copy protection of computer software*** … Application of Houser to computer software would thus fail to achieve ***a significant benefit*** of the claimed invention; ***namely, rendering an unauthorized copy of the computer software effectively inoperable***.

(Doc. No. 32, Ex. 8 ('569 Patent prosecution history, Response to Office Action dated July 23, 1997), at 2–3 (emphasis added)).) The patent applicants' statements are a far cry from disclaimer with respect to the term "copy protecting." Instead, the patent applicants distinguished Houser as disclosing a system allowing users to sign electronic documents. Further, the applicants state that "a significant benefit" of the claimed invention relates to unauthorized copies; not the only benefit. The Asserted Patents thus contemplate that "copy protection" means more than simply protecting copies.

Given the parties' conflicting views of the "plain and ordinary" meaning of the term, however, a construction of "copy protection" is necessary. The Court adopts the same construction as *Huawei's* tentative construction: "protecting a software application from unauthorized copying or unauthorized use." Blue Spike further requests that the Court include the phrase "unintended use" in the construction of this term. However, the phrase "unauthorized use" already covers, to some extent, the concept of "unintended use." The phrase "unintended use" also goes further to imply a level of accidental use. The '569 Patent, however, does not reference protection against accidental use of a software application. Rather, each of the '569 Patent examples addresses concerns of hackers intentionally tampering with software

applications or copies of software in one form or another. Thus, including the phrase "unintended use" in the Court's construction would be, to some extent, redundant and, to some extent, inappropriate.

Accordingly, the Court construes the term "copy protecting a software application" as "protecting a software application from unauthorized copying or unauthorized use."

d. during execution of the software application

Claim Term	Plaintiffs' Proposal	Defendants' Proposal
"during execution of the software application" ('569 Patent, Claim 16)	Plain and ordinary Alternate construction: "the time between when an application begins and ends"	"while the software application is running"

Blue Spike argues that the term "during execution of the software application" should be understood by its plain and ordinary meaning. (Doc. No. 29, at 7.) Specifically, Blue Spike argues that the plain and ordinary meaning of "execution" encompasses the "loading" of an application. (*Id.* at 7–8 (citing Doc. No. 29, Ex. 5 (*Microsoft Computer Dictionary*, Fifth Edition, 2002) ("In programming, execution implies loading machine code of the program into memory and then performing the instructions.")).) Blue Spike argues that both running and loading occur during execution, *i.e.*, execution begins when a user double-clicks an application on a desktop and continues even if an application goes idle, passes control to another application, "and so on." (*Id.* at 8.)

Toshiba contends that "loading" and "execution" are different claim terms, and different claim terms are presumed to have different meanings. (Doc. No. 32, at 13.) Toshiba contends that the preamble of Claim 16 of the '569 Patent refers to resources that have already been

loaded (*i.e.*, "plurality of code resources loaded in a memory"), and then refers to a "software application" and "executable code resources," indicating that the code resources have the potential to be executed in the future (*i.e.*, *after* load time). (*Id.*) Toshiba further notes that the '569 Patent describes "'intentional[] shuffl[ing]' of executable code resources '[d]uring execution time' at 'periodic[]' or 'random or pseudo-random intervals.'" (*Id.* (citing '569 Patent, at 8:1–9).) Toshiba also argues that during patent prosecution of a related patent, the applicant equated "execution" with "running" and treated an application's run-time separate from "system time." (*Id.* at 13.) Toshiba objects to Blue Spike's assertion, without any citation, that double-clicking on an application means that "an application is executed." (*Id.* at 12.) Toshiba contends that just because a "user double-clicks on an application," which may cause the loading and the successive executing of the application, does not mean the two concepts are the same. (*Id.*)

In reply, Blue Spike contends, without support, that loading is a subset of the executing phase: when an application is executed its instructions are first loaded and then carried out. (Doc. No. 33, at 7.) Blue Spike contends that the Court adopted this position in *Huawei* by finding that "the claim does not require that loading be completed before a program is executed." (*Id.* (citing *Huawei* Order, at 7).) Blue Spike argues with respect to the prosecution history that the applicants' use of the phrase "while running" had nothing to do with the concept of "after loading." (*Id.* at 8.)

The *Huawei* Order addressed similar arguments to the arguments the parties make here. In *Huawei*, the Court stated

> Defendants are correct that the '569 Patent treats loading and executing differently, that loading precedes execution, and that the code resources are intermittently relocated during software execution. For example, loading is the act of "transfer[ring] a program from a storage device into a computer's memory" and executing is "run[ning] a program, carry[ing] out a command, or peform[ing] a function." [(Doc. No. 32, Ex. 13 (*Dictionary of Computer Words, Am. Heritage*

> *Dictionaries Revised Ed.*, at 94, 160 (1995)).)] However, the claim does not require that loading be completed before a program is executed. Likewise, the '569 Patent does not exclude "idle" time from "execution time." That is, referring to Defendants' app example, just because the application is in the background or waiting for an input from a user does not necessarily mean the app is not executing. As acknowledged by Defendants, there may still be "a few things happening." . . . As long as the app has started to perform instructions, it could potentially read onto this term.

Huawei Order, at 7. The Court agrees that loading and execution are separate concepts. For instance, Claim 16 of the '569 Patent refers separately to "a software application executed by a computer system" and "executable code resources loaded in a memory." However, as the *Huawei* Order makes clear, the Asserted Patents do not exclude the possibility of execution occurring concurrently with loading.

Neither party has set forth a sufficient basis to depart from the analysis in *Huawei*. However, the Court ultimately declines to follow *Huawei's* construction, which matches Toshiba's proposed construction in this case. *See Huawei* Order, at 7 (construing "during execution of the software application" as "while the software application is running"). Swapping out "during execution" with "while running" does not add any clarity to a jury's understanding of this term. Further, resolution of the real dispute between the parties as to whether loading occurs during execution is not impacted by exchanging "during execution" for "while running." As noted above, "loading" and "execution" are separate concepts. "During the execution" does not encompass mere "loading." By rejecting Blue Spike's position with respect to loading and execution, no further construction is necessary. *See O2 Micro Int'l Ltd. v. Beyond Innovation Tech. Co.*, 521 F.3d 1351, 1362 (Fed. Cir. 2008) ("district courts are not (and should not be) required to construe every limitation present in a patent's asserted claims."); *Finjan, Inc. v. Secure Computing Corp.*, 626 F.3d 1197, 1207 (Fed. Cir. 2010) ("Unlike O2 Micro, where the court failed to resolve the parties' quarrel, the district court rejected Defendants' construction.").

The Court thus concludes that no construction is necessary for the term "during execution of the software application."

e. call

Claim Term	Plaintiffs' Proposal	Defendants' Proposal
"call" ('569 Patent, Claims 17, 18)	Plain and ordinary Alternate construction: "A statement in a computer program that references a subroutine or program." Proposal on Reply "a transfer of control from one software module to another, usually with the implication that control will be returned to the calling module" (Doc. No. 33, at 8.)	"Transfer of control during execution time."

Blue Spike contends that one of ordinary skill in the art would understand that when a program is called, the program is triggered to perform its assigned function. (Doc. No. 29, at 14.) Blue Spike contends that Toshiba's construction of "call" introduces confusion because an underlying program need not always transfer control or end when making a call. (*Id.*) Blue Spike further argues that Toshiba's reference to the phrase "execution time" in the construction of "call" "only causes additional confusion." (*Id.*)

Toshiba contends that the asserted claims use the term "call" with reference to the "memory scheduler." (Doc. No. 32, at 20.) Toshiba argues that according to the specification, the "memory scheduler" is only called during execution time. (*Id.* (citing '569 Patent at 8:3–4).)

Toshiba argues that dictionaries support its proposed construction of "call" as requiring a transfer of control. (*Id.*).

Blue Spike replies by noting that its proposed alternative construction aligns with one of the dictionary definitions identified by Toshiba: "[a] transfer of control from one software module to another, usually with the implication that control will be returned to the calling module." (Doc. No. 33, at 8 (citing Doc. No. 32, Ex. 16 (*The IEEE Standard Dictionary of Electrical and Electronic Terms*)).)

The term "call" only appears in the specification in two places: With respect to "essential" code resources, the specification states:

> In general, any code resource can be considered "essential" in that if the program proceeds to a point where it must "***call***" the code resource and the code resource is not present in memory, or cannot be loaded, then the program fails.

'569 Patent, at 4:60–67 (emphasis added). With respect to the "memory scheduler," the specification states, "[d]uring execution time, this special code resource, called a 'memory scheduler,' can be ***called*** periodically, or at random or pseudo random intervals . . . ." '569 Patent, at 8:3–6 (emphasis added). Neither of these passages emphasizes "execution time" or "transfer of control" specifically with regard to the concept of "calling". In other words, while the specification and claims may indicate the "memory scheduler" is only called during execution time, the specification does not require that calls in their general sense only occur during execution time. Further, including the phrase "during execution time" in the construction of "call" in the manner that Toshiba proposes adds ambiguity. As written, Toshiba's construction is unclear whether the phrase refers to the execution time of an operating system, the application software, or the call itself.

Nor do either of the quoted passages of the specification require a "transfer of control" with respect to calls. Indeed, it would be improper to impute the phrase "transfer of control" into the construction of this term. As indicated in Toshiba's dictionary definitions, calls may be used to summon "subroutines," "procedures," or other software modules. (Doc. No. 32, Exs. 14 & 15.) There is no requirement in these definitions and no disclosure in the specification limiting the term "call" to require a *full* shutdown and *full* transfer of control from the original program to another subroutine, program, procedure, or module. Thus, although the "transfer of control" language may appear in some dictionary definitions of "call," this language alone could mislead a fact-finder to think 1) transfer of control must occur in its entirety and 2) further, to the extent control is transferred, it is permanently transferred.

The specification uses the term "call" in its plain and ordinary sense in the field of computer technology. Neither party has pointed to an inherent ambiguity in the meaning of the term "call" that would warrant construction. Rather, Toshiba's proposal is an attempt to limit the scope of the term "call" that is unwarranted in the context of the Asserted Patents and would be more confusing than helpful for a jury.

The Court thus finds that no construction is necessary for the term "call".

f. intermittently relocating

Claim Term	**Plaintiffs' Proposal**	**Defendants' Proposal**
"intermittently relocating" ('569 Patent, Claims 16, 18, 19, 20)	**Intermittently Relocating** Plain and ordinary Alternate construction 1: "relocating one or more times" Alternate construction 2: "moving to a new location one	**Intermittently Relocating** "intentionally shuffling at periodic, random or pseudo-random intervals"

	or more times" <u>Proposal on Reply</u> "shuffling randomly, seemingly-randomly, or non-randomly" (*see* Doc. No. 33, at 9)	

In their briefing, the parties submitted proposed constructions and argument with respect to both the terms "intermittently relocating" and "relocating." However, Claims 16, 18, 19, and 20 each recite "intermittently relocating"; none of them recite simply "relocating." Thus, the Court will only address the parties' arguments relating to "intermittently relocating."

Blue Spike contends that Toshiba replaces "intermittently relocating," a plainly understood phrase, with hyper-technical language that is likely to confuse a jury. (Doc. No. 29, at 9.) Blue Spike specifically objects to the use of the word "intentionally" in Toshiba's construction. (*Id.* at 10.) Blue Spike contends that the specification refers to intentionality in the context of the memory scheduler and merely indicates that the memory scheduler shuffles code in memory when called. (*Id.*) Blue Spike argues that including the word "intentional" in the construction of "intermittently relocating" risks importing an additional limitation into Claim 16 of the '569 Patent, which does not claim a "memory scheduler". (*Id.* at 10–11.) Blue Spike also sets forth alternative proposed constructions that clarify "intermittently relocating" can occur one or more times. (*Id.* at 11 (citing *Huawei* Order, at 9 (holding with respect to the term "intermittently relocating" that "[t]he patentee's statement [in prosecution history] could include multiple shuffles but does not necessarily exclude a single shuffle.")).)

Toshiba argues that the *Huawei* Order's interpretation of "intermittently relocating" as covering a single shuffle is incorrect and that the term should require more than one shuffle. (Doc. No. 32, at 21 (citing *Huawei* Order, at 9).) Toshiba contends that the term "relocating"

alone could include a single shuffle, but the addition of "intermittently" demonstrates that the patent applicants intended the term to encompass only multiple shuffles. (*Id.* at 21.) Toshiba argues that the term "intervals" as used in its plural form in the specification, prosecution history, and *Huawei* Order further conveys "the meaning of more than one interval," and thus at least two shuffles. (*Id.* at 22 (citing '569 Patent, at 8:3–7; Doc. No.32, Ex. 17 ('569 Patent prosecution history, Response to Final Office Action June 26, 1997), at 7; *Huawei* Order, at 9).) Toshiba also specifically argues that the prosecution history, relied on in the *Huawei* Order, shows that the applicants distinguished prior art systems on the basis that they did not rearrange memory in an intentional effort to secure copy protection. (*Id.* at 22 (citing '569 Patent, at 7:30–36).) Toshiba argues that the term is not "readily understood" because 1) the examiner needed clarification as to what "intermittently relocating" meant, and 2) in *Huawei*, the Court noted that the term has "inherent ambiguity." (*Id.* at 23 (citing Doc. No. 32, Ex. 18 ('569 Patent prosecution history, Office Action May 22, 1997), at 2 ("it is unclear what is meant by 'relocating' and by 'intermittently relocating' in context."); *Huawei* Order, at 9).)

Blue Spike replies by arguing that "shuffling one or more times" is sufficient to capture the meaning of the term "intermittently relocating." (Doc. No. 33, at 9.) Blue Spike proposes an alternative construction of "shuffling randomly, seemingly-randomly, or non-randomly." (*Id.*)

The *Huawei* Order construed the term "intermittently relocating" as "intentionally shuffling at periodic, random, or pseudo-random intervals." *Huawei* Order, at 10. This is the same construction that Toshiba proposes in this case.

Blue Spike does not provide a sufficient explanation for why its alternative proposed phrasing of "randomly, seemingly-randomly, or non-randomly" adds more clarity than the phrasing "periodic, random, or pseudo-random." The parties do not appear to have a serious

dispute on this point, and the Court finds no basis to depart from the "periodic, random, or pseudo-random" phrasing used in both the specification and in *Huawei's* construction. *See* '569 Patent, 8:3–10; (*see also* Doc. No. 32, Ex. 17 ('569 Patent prosecution history, Response to Final Office Action June 26, 1997), at 7 (referring to this passage of the specification to define "intermittently relocating.").) The two main disputes remaining between the parties are thus 1) whether "intermittently relocating" requires intentional shuffling and 2) whether "intermittently relocating" requires more than one shuffle.

The *Huawei* Order refers to Blue Spike's dispute with the term "intentional" in the construction of "intermittently relocating," but does not address the dispute at length. *See Huawei* Order, at 9. The Order does quote from the relevant portion of the '569 Patent stating:

> Under the present invention, the application contains a special code resource which knows about all the other code resources in memory. During execution time, this special code resource, called a "memory scheduler," can be called periodically, or at random or pseudo random intervals, at which time it ***intentionally shuffles*** the other code resources randomly in memory, so that someone trying to analyze snapshots of memory at various intervals cannot be sure if they are looking at the same code or organization from one "break" to the next.

'569 Patent at 8:1–10. The Order also notes, "[d]efendants further clarified that 'intentional' does not require user initiated relocating or shuffling, addressing Blue Spike's concern on this point." *Huawei* Order, at 9. Blue Spike argues, however, that the quoted portion of the specification merely indicates that the *memory scheduler* intentionally shuffles other code resources; it does not mean that the term "intermittently relocating" requires intentional shuffling. (Doc. No. 29, at 10.) Blue Spike also argues that including the term could lead to jury confusion "as to whom or what is intending the shuffling to occur." (*Id.* at 11.) These arguments are well-taken. Construing "intermittently relocating" to require "intentional shuffling" opens up the potential for jury confusion by implying a level of intentionality that the

Asserted Patents do not require. Toshiba responds by noting that the Asserted Patents distinguish the claimed invention from prior art systems that "rearrange certain portions of memory during 'system time'. . . Typically, this is done in low memory systems, to maintain optimal memory utilization." (Doc. No. 32, at 22 (citing '569 Patent, at 7:30–36).) However, Claim 16 of the '569 Patent specifically requires that the code resources are intermittently relocated "during execution of the software application." Thus, the claims on their face already distinguish from these prior art scenarios. There is no basis to import "intentional shuffling" into the meaning of "intermittently relocating."

Huawei did analyze whether "intermittently relocating" requires more than one shuffle. *See Huawei* Order, at 9. The *Huawei* Order found that there was not a clear disclaimer or disavowal in the prosecution history excluding a single shuffle. *Id.* According to *Huawei*, "[t]he patentee's statement [during prosecution history] could include multiple shuffles but does not necessarily exclude a single shuffle. For example, a shuffle at a random time would be consistent with the patentee's statement." *Id.* *Huawei* was specifically referring to a portion of the prosecution history where the applicants stated:

> Applicants respectfully direct the Examiner's attention to the disclosure at [Col.8:1-19] of the Specification, which describes an embodiment of the present invention wherein the code resources are "shuffled" in memory at periodic, random or pseudo-random intervals. *See also* [Col. 3:38-41 of the] Specification, ("Attempts to tamper or 'patch' substitute code resources can be made highly difficult by randomizing the location of said resources in memory on an intermittent basis . . ."). Given the foregoing, the Applicant respectfully submits that the language of claim[] 28 would be clear to a person of ordinary skill in the art.

(Doc. No. 32, Ex. 17, at 7.) This Court respectfully disagrees with *Huawei's* interpretation of "intermittently relocating" as not excluding a single shuffle for two reasons. First, both the specification and the prosecution history refer to "intervals," not the singular term "interval."

(*See id.*); *see also* '569 Patent, at 8:1–19. Second, the plain meaning of the term "intermittently" (and the phrase "on an intermittent basis") implies that an event occurs on a repeated basis: something that happens only once has not occurred "intermittently." (*See* Doc. No. 29, Ex. 5 (*Microsoft Dictionary*, 5th Ed.) ("intermittent: adj. pertaining to something, such as a signal or connection, that is not unbroken, but occurs at periodic or occasional intervals"), at 280.) If the applicants intended to encapsulate just a single movement of memory, they could have easily drafted Claims 16, 18, 19, and 20 of the '569 Patent to say simply "relocating", "randomizing", or "shuffling." The Court thus finds that "intermittently relocating" requires more than one shuffle.

Accordingly, the Court construes "intermittently relocating" as "shuffling at periodic, random, or pseudo-random intervals."

g. shuffle & randomize

Claim Term	Plaintiffs' Proposal	Defendants' Proposal
"shuffle" ('719 Patent, Claim 10, 11, 12)	Plain and ordinary <u>Alternate construction:</u> "move a portion of a sequence to a new location in a sequence" <u>Proposal on Reply</u> "rearrange" or "reorganize" (*See* Doc. No.33, at 9 n.12)	"Randomly reorganize during program run time."

Claim Term	Plaintiffs' Proposal	Defendants' Proposal
"randomize" ('719 Patent, Claim 22)	"employ random selection" <u>Proposal on Reply</u> "rearrange" or "reorganize" (*See* Doc. No. 33, at 9 n.12)	"Randomly reorganize during program run time."

Blue Spike objects to Toshiba's constructions as applying the same definitions for "randomize," and "shuffle." (Doc. No. 29, at 15.) Blue Spike contends the terms are not used synonymously. (*Id.*) Blue Spike also objects to Toshiba's constructions (1) as being more confusing than the original terms, (2) as improperly importing the idea of "during program run time," (similar to Blue Spike's objection to "execution time" in the "call" term above), and (3) as construing "shuffle" and "randomize" the same, even though both terms are used in the same claims. (*Id.*) With respect to the term "shuffle," Blue Spike contends that its alternative construction as "move a portion of a sequence to a new location in a sequence" embodies the idea that a collection of code resources may be moved to different locations and affords "shuffle" its own meaning rather than sharing it with disparate terms. (*Id.*) With respect to the term "randomize," Blue Spike asserts that the plain and ordinary meaning is to "employ random selection." (*Id.*)

Toshiba acknowledges that, as a default, different claim terms have different meanings. (Doc. No. 32, at 24.) Toshiba contends, however, that a court can construe different terms to have the same meanings where the patent and prosecution history do not reveal a difference between the terms. (*Id.* (citing *Fargo Elecs., Inc. v. Iris Ltd., Inc.*, No. 04-1017 JRT/FLN, 2005 WL 3241851, at *14 (D. Minn. Nov. 30, 2005), aff'd, 287 F. App'x 96 (Fed. Cir. 2008) ("In so doing, the Court is aware that it is defining different terms to have essentially the same meaning. Although a patentee's use of different terms normally indicates that it intended those terms to carry different meanings, a reading of the patent and prosecution history does not reveal what that difference might be in this case")).) Toshiba contends that "the core of the invention disclosed in the '569 Patent is the shuffling of code resources randomly in memory to prevent

unauthorized copying and increase protection." (*Id.*) Toshiba contends that the patent refers to the shuffling process "in many ways throughout the patent – relocate, randomize, shuffle – but all of these terms have the same meaning." (*Id.*) Further, Toshiba contends that the Asserted Patents emphasize "during program run time" and "during execution time," consistent with Toshiba's construction. (*Id.* at 24–25 (citing '569 Patent, at 2:21–26, 8:3–7).)

On reply, Blue Spike alternatively proposes constructions of the terms "shuffle" and "randomize" as either "rearrange" or "reorganize." (Doc. No. 33, at 9 n.12.) Blue Spike contends that "anyone who has shuffled a deck of cards understands that to 'shuffle' means to 'rearrange' or 'reorganize.'" (*Id.* at 9.) Blue Spike contends that shuffling implies movement of multiple items. (*Id.*) Blue Spike contends that unlike "randomize," the term "shuffle" does not require random movement. (*Id.* at 9–10 (citing '569 Patent, 8:6–7 ("shuffles the other code resources randomly in memory.")).) Blue Spike argues that "during program run time" does not even match Toshiba's cited portion of the specification, which states "[d]uring execution time." (*Id.* at 10.) Blue Spike argues that while some of the claims specifically recite "during execution time," others do not, making it inappropriate to import the limitation into the "shuffle" and "randomize" terms. *Id.* at 10–11.

As with the term "call," there is not a clear disclaimer or disavowal in the intrinsic record with respect to the terms "shuffle" and "randomize" such that the terms should be limited to only occurring "during execution time." Although embodiments in the specification refer to "shuffling" or "relocating" as occurring "during execution time," (*see, e.g.*, '569 Patent, at 2:21–26; 8:3–7), "particular embodiments and examples appearing in the specification will not generally be read into the claims." *Constant v. Advanced Micro-Devices, Inc.*, 848 F.2d 1560, 1571 (Fed. Cir. 1988); *see also Phillips*, 415 F.3d at 1323. Indeed, as Blue Spike notes, only

some of the claims of the Asserted Patents that refer to "shuffling," "randomizing," and "relocating," also claim "during execution time."

The parties seem to agree at a high level that all the terms relate to "rearranging" or "reorganizing." However, it would be inappropriate to construe both "shuffle" and "randomize" as "randomly reorganize." The specification uses the term "shuffle" in a manner that does not always require "random" shuffling. For example, the specification states "shuffle the other code resources randomly." '569 Patent, at 8:3–7. By specifically indicating that the shuffling occurs randomly in this embodiment, the specification implicitly acknowledges that "shuffling" is not inherently random. *See Phillips*, 415 F.3d at 1314 ("To take a simple example, the claim in this case refers to "steel baffles," which strongly implies that the term "baffles" does not inherently mean objects made of steel."). On the other hand, in some instances the specification refers to "randomly reorganize[ing]." '569 Patent, 2:21–22; 7:22–24 ("[a] second method according to the present invention is to randomly re-organize program memory structure to prevent attempts at memory capture or object code analysis."). The word "randomize" is simply a conjugation of the words "randomly" and "reorganize"; by its very nature, the word "randomize" implies that reorganization is random.

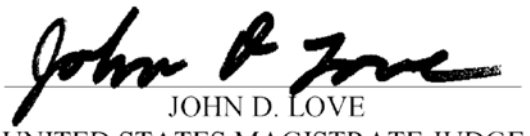
The Court thus construes "shuffle" as "reorganize" and construes "randomize" as "randomly reorganize."

CONCLUSION

For the foregoing reasons, the Court adopts the constructions set forth above

.

So ORDERED and SIGNED this 28th day of June, 2017.



JOHN D. LOVE
UNITED STATES MAGISTRATE JUDGE