IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF DELAWARE

PERSONAL AUDIO, LLC,	)	
	)	
Plaintiff,	)	
	)	
v.	)	Civil Action No. 17-1751-CFC-CJB
	)	
GOOGLE LLC,	)	
	)	
Defendant.	)	

REPORT AND RECOMMENDATION

In this action filed by Plaintiff Personal Audio, LLC ("PA" or "Plaintiff") against Google LLC ("Google" or "Defendant"), PA alleges infringement of United States Patent Nos. 6,199,076 (the "'076 patent") and 7,509,178 (the "'178 patent" and collectively with the '076 patent, "the asserted patents"). Presently before the Court is the matter of claim construction. The Court recommends that the District Court adopt the constructions as set forth below.

I. BACKGROUND AND STANDARD OF REVIEW

The Court hereby incorporates by reference the summary of the factual and procedural background of this matter set out in its January 16, 2019 Report and Recommendation ("January 16 R&R"). (D.I. 331 at 1-4) It additionally incorporates by reference the legal principles regarding claim construction set out in the January 16 R&R. (*Id.* at 4-9)

II. DISCUSSION

The parties had disputes regarding 10 terms or sets of terms (hereafter, "terms"). The January 16 R&R addressed the first three terms. The Court addresses terms 4, 5 and 6 herein. The remaining terms will be addressed in a forthcoming Report and Recommendation(s).

A. "means for continuously reproducing" term

The claim term "means for continuously reproducing said program segments in the order established by said sequence in the absence of a control command" (the "means for continuously reproducing term") appears in claims 1-3, 5, 6 and 13-15 of the '076 patent. (Google's Markman Presentation, Slide 43) The use of the disputed term in claim 1 of the '076 patent is representative. Accordingly, this claim is reproduced below, with the disputed term highlighted:

> **1.** A player for reproducing selected audio program segments comprising, in combination:
>
> means for storing a plurality of program segments, each of said program segments having a beginning and an end,
>
> means for receiving and storing a file of data establishing a sequence in which said program segments are scheduled to be reproduced by said player,
>
> means for accepting control commands from a user of said player,
>
> *means for continuously reproducing said program segments in the order established by said sequence in the absence of a control command,*
>
> means for detecting a first command indicative of a request to skip forward, and
>
> means responsive to said first command for discontinuing the reproduction of the currently playing program segment and instead continuing the reproduction at the beginning of a program segment which follows said currently playing program in said sequence.

('076 patent, col. 46:13-33 (emphasis added))

The parties agree that this limitation is a means-plus-function limitation, governed by 35 U.S.C. § 112(6). (D.I. 146, Appendix A at Ref. 6) The parties' competing proposed constructions for the term are set out in the chart below, with the main disputed points noted by bold, underlined language:

Term	**Plaintiff's Proposed Construction**	**Defendant's Proposed Construction**
"means for continuously reproducing said program segments in the order established by said sequence in the absence of a control command"	<u>Function</u>: "[C]ontinuously reproducing said program segments in the order established by said sequence in the absence of a control command." <u>Structure</u>: "A sound card that includes a digital to analog converter; headphones or one or more speakers; and a general purpose computer programmed to perform the algorithm that is illustrated in the flow chart of Figure 3 at items 233, 235, 237, 239, and 261 and more fully described at column 12, line 16 to column 13, line 11 and column 34, line 28 to column 35, line 44. Specifically, this algorithm includes the following steps: 1. beginning playback with the program segment identified by the ProgramID contained in the Selection_Record specified by the CurrentPlay variable; 2. when the currently playing program segment concludes, incrementing the CurrentPlay variable by one and fetching and playing the program segment identified by the ProgramID contained in the	<u>Function</u>: "Continuously reproducing said program segments in the order established by said sequence in the absence of a control command." <u>Structure</u>: "A sound card that includes a digital to analog converter; headphones or one or more speakers; and a general purpose computer programmed to perform the algorithm that is illustrated in the flow chart of Figure 3 at items 233, 235, 237, 239, and 261 and described at column 12, line 16 to column 13, line 11 and column 34, line 28 to column 35, line 44. Specifically, this algorithm includes the following steps: (1) beginning playback with the program segment identified by the ProgramID contained in the Selection_Record specified by the CurrentPlay variable; (2) when the currently playing program segment concludes, **<u>(a) if the concluded segment is a topic or subject announcement, incrementing the CurrentPlay variable by</u>**

	next Selection_Record in the sequencing file; 3. repeating step (2) **until a command is issued or that the sequence is completed**."	**one and fetching and playing the program segment identified by the ProgramID contained in the next Selection Record in the received sequencing file, and** **(b) if the concluded segment is a program segment,** **(i) scanning forward in the received sequencing file to locate the next Selection_Record containing the appropriate LocType;** **(ii) resetting the CurrentPlay variable to the record number of that Selection_Record; and** **(iii) fetching and playing the program segment identified by the ProgramID contained in the new Selection_Record;** (3) repeating step (2) **until a rewind Selection_Record (LocType: R) in the received sequencing file is reached, which resets the CurrentPlay variable to the location value contained in the rewind Selection_Record which is set to "1" to begin the playing sequence again with the first Selection_Record in the received sequencing file**."

(D.I. 146, Appendix A at Ref. 6) As reflected in these proposals, the parties have two primary disputes with respect to this term: whether the corresponding algorithmic structure requires (1) the sequence to be repeated in an endless loop; and (2) scanning for the next record of appropriate LocType. (D.I. 146 at 15, 17-18; D.I. 159 at 22; Tr. at 128, 132) The Court will take up these disputes in turn.

1. Does the algorithm require playback to continue in an endless loop?

The parties have agreed upon certain of the required structure for the recited function. Google asserts that such agreed-upon structure requires the program segments to play in an endless loop, absent a user command. (D.I. 159 at 22-24; D.I. 186 at 7-8; Tr. at 129-30) PA argues to the contrary that the structure does *not* require the playback of programs in an endless loop, and instead that "continuously reproducing" merely requires that playback of programs proceeds without interruption. (D.I. 146 at 15; D.I. 176 at 9-10; Tr. at 138-39)

The parties agree that Figure 3 of the patent discloses the algorithm corresponding to the claimed function. (D.I. 146 at 13-14; D.I. 159 at 22) Step 261 shows that if there is no user command, the player proceeds to "Continue Playback" at Step 235. ('076 patent, FIG. 3; *see also* Tr. at 129; D.I. 146 at 14-15) From there, the player determines if a new segment has started at Step 237; if so, the player then handles the new segment in Step 239. ('076 patent, FIG. 3; *see also* D.I. 146 at 14)

The parties also agree that the algorithm for "continuously reproducing said program segments in the order established by said sequence in the absence of a control command" is "[more fully] described" in the specification of the '076 patent at column 34, line 28 to column 35, line 44. (D.I. 146, Appendix A at Ref. 6) At one point in this portion of the specification, the patent explains that:

> When the player is first activated, CurrentPlay is set to 1 to begin play with the first topic announcement specified by the ProgramID **357**. The end of the selections file **351** is marked with an "R" Selection_Record **380** which contains the location value 1. When the player encounters this record, it resets the CurrentPlay register to 1, and the playing sequence begins again.

('076 patent, col. 35:38-44) The specification then immediately continues on to say that "[t]his arrangement *creates, in effect, an endless loop*, allowing the user to skip forward in circular fashion through the entire program selection to locate desired programming, regardless of where the CurrentPlay register is set." (*Id.*, col. 35:44-48 (emphasis added)) Thus, Google's observation that "[t]he [a]greed [a]lgorithm [i]dentified by the [p]arties as [p]erforming the '[c]ontinuously [r]eproducing' [f]unction [r]equires a [c]ircula[r] [l]oop" seems accurate. (D.I. 159 at 22 (emphasis omitted); *see also* Tr. at 130)

Google's position is one that has also garnered support from other sources in the past (and, at times, even from PA). For example:

> (1) During the *Apple* litigation, the Eastern District of Texas Court agreed with Google's position, construing the structure corresponding to the function of this term as requiring an endless loop.[1] The Eastern District of Texas Court subsequently denied PA's motion for reconsideration of the Court's inclusion of the endless loop requirement. (D.I. 160, ex. 16 at 24-31) In a thorough analysis spanning several pages, the Eastern District of Texas Court explained that the only disclosed algorithm for performing the claimed function of "continuously reproducing" "includes playing program segments in an endless loop"; the Court cited in support to some of the above-referenced portions of the specification. (*Id.* at 26) The Eastern District of Texas Court further noted that there is no alternative structure disclosed in the specification—such as a different value that might be contained in the last

[1] Steps 1 and 2 of the *Apple* Court's construction were similar to PA's here, but then Step 3 recited: "repeating step (2) until the last Selection_Record in the sequencing file is reached, which resets the CurrentPlay variable to '1' to begin the playing sequence again with the first Selection_Record in the sequencing file." (D.I. 160, ex. 16 at 24-25 (emphasis omitted))

Selection_Record in the sequencing file, or a different algorithmic step that could occur when the last record in the sequence is encountered. (*Id.* at 26-29)

(2) During the *inter partes* review ("IPR") proceeding for the '076 patent, both Google and PA agreed that the endless loop requirement should be adopted. (D.I. 147, ex. E at 11-13)

(3) In the parties' Joint Claim Construction Statement filed in this case, PA did not dispute that the disclosed structure for this term requires an endless loop. (D.I. 76, ex. B at 5-6) It was not until the submission of claim construction briefing that PA revisited its old position that the corresponding structure does not require an endless loop. (D.I. 146 at 15; *see also* D.I. 159 at 23-24; Tr. at 128)

In assessing which side is right as to this dispute, it is instructive that the *Apple* Court, the PTAB, and even PA itself (at certain points) all agreed with Google's position.

What does PA point to in support of its current position to the contrary? PA first asserts that the person of ordinary skill in the art ("POSITA") at the time of the invention would understand the term "continuous reproduction" to "merely require the reproduction of programs absent a control command and does not further [require] playback in an endless loop." (D.I. 146 at 15; *see also* D.I. 176 at 9-10; Tr. at 138-39) PA's expert, Dr. Kevin C. Almeroth, cites to several dictionary definitions, product manuals and patents that all seem to support this interpretation of "continuous reproduction." (D.I. 147 at ¶¶ 71-78) Next, PA points to claim 4 of the '076 patent, (D.I. 146 at 15-16; D.I. 176 at 10, 11; Tr. at 139), which depends from claim 1; claim 4 explicitly recites an endless loop: "[a] player . . . wherein said sequence established by said data forms an endless circular sequence of program segments[,]" ('076 patent, col. 46:49-51). According to PA, this demonstrates that claim 1 *does not* require an endless loop "because claims 1 and 4 cannot be coterminous in scope[.]" (D.I. 146 at 15) PA also contends that the plain language of the term at issue (i.e., "reproducing [] program segments *in the order* *established by said sequence* in the absence of a control command") simply requires that the audio files of the sequence be continuously played through once without being repeated in the absence of a control command. (*Id.* at 16 (certain emphasis omitted); *see also* D.I. 176 at 10)

In the Court's view, PA's various arguments are insufficient to supplant what the intrinsic record discloses as the corresponding structure for this function. What is clearly lacking from PA's assertions is a description of *where in the specification* the patent talks about performing this function in some alternative way. Google correctly notes that "this is a means-plus-function term, so PA's claims must be limited to algorithms disclosed in the specification. PA has pointed to no algorithm other than the one that requires an endless loop." (D.I. 186 at 8; *see also* D.I. 159 at 23-24 ("[PA] points to nothing in the agreed structure to support [its] new construction."); Tr. at 130-31, 147-48; D.I. 160, ex. 16 at 29 ("The [*Apple*] [C]ourt has located no place in the text of the specification describing an alternative step wherein playback simply ends when the last Selection_Record in the sequencing file is encountered.")); *Saffran v. Johnson & Johnson*, 712 F.3d 549, 563 (Fed. Cir. 2013) ("Under [Section 112, paragraph 6], the question is . . . what structures are specifically disclosed and tied to that function in the specification.").[2]

[2] In its reply brief, PA does make an attempt to fill in this gap. It points to Figure 7, which depicts a sequencing file 470 that does not contain a LocType R, as illustrating that the patent does not disclose only one embodiment requiring an endless loop. (D.I. 176 at 10-11; Tr. at 142 (PA's counsel asserting that there is no LocType R in sequencing file 470, and therefore "this sequence would not implement a loop, and that's a clear disclosure of an alternative" embodiment)) But as Google persuasively retorts, (D.I. 186 at 8; Tr. at 131-32), the specification explains that Figure 7 "shows an example of the content of a *portion* of an illustrative HTML text file indicated generally at **450** used to create an audio file seen at **460** and a selections file indicated at **470**[,]" ('076 patent, col. 44:10-13 (emphasis added)). The patent explains that the invention can include the presentation of narrative text, and the last record depicted in the portion of Selections File 470 depicted in Figure 7 is K, which the patent explains "specifies the point at which the current image display should end." (*Id.*, FIG. 7 & col. 44:63-64; Google's Markman Hearing Presentation, Slide 52) In light of the fact that the specification refers to Figure 7 as displaying a "portion" of the relevant files, the Court is unpersuaded that sequencing file 470

The intrinsic evidence, described above, "trumps any extrinsic evidence, including dictionary definitions, that might contradict it." *WhatsApp Inc. v. Intercarrier Commc'ns, LLC*, Case No. 13-cv-04272-JST, 2014 WL 5306078, at *7 (N.D. Cal. Oct. 16, 2014) (citing *Phillips v. AWH Corp.*, 415 F.3d 1303, 1314 (Fed. Cir. 2005)). And PA's claim differentiation argument regarding dependent claim 4 is not dispositive on the question of whether the corresponding structure for the "continuously reproducing" term requires an endless loop. *See, e.g.*, *Laitram Corp. v. Rexnord, Inc.*, 939 F.2d 1533, 1538 (Fed. Cir. 1991) (explaining that "the interpretation of the 'means for joining' to include a cross member comes from the specification via section 112(6), not from [dependent] claim 24" and "[i]f a claim will bear only one interpretation, similarity [between claims] will have to be tolerated") (internal quotation marks and citation omitted).

For these reasons, the Court agrees with Google that the corresponding structure for the means for continuously reproducing term requires an endless loop.

2. Does the algorithm require scanning for the next record of appropriate LocType?

The dispute between the parties as to this second issue appears to be two-fold. That is, the parties dispute: (1) whether the LocType R structure is required to perform the recited function (i.e., as recited in Step 3 of Google's proposed structure); and (2) whether "scanning the file to locate the next record using LocType is required for advancing to the record representing the next program segment in the sequencing file in the course of 'continuously reproducing[]'" (i.e., as recited in Steps 2(a)-(b)(iii) of Google's proposed structure). (*See, e.g.*, D.I. 159 at 24;

represents (1) an entire sequencing file that (2) sheds light on the appropriate corresponding structure for this term.

D.I. 176 at 11) These steps are additions that Google is making in this litigation to the endless loop requirement that was recited in the corresponding structures for this term in the *Apple* litigation and in the IPR proceeding. (*See* D.I. 146, Appendix A at Ref. 6)

With respect to the first of these two issues, the Court agrees with Google that the rewind "R" LocType should be included in the corresponding structure. As described above, the specification explains that "[t]he end of the selections file **351** is marked with an 'R' Selection_Record **380** which contains the location value 1 [and] [w]hen the player encounters this record, it resets the CurrentPlay register to 1, and the playing sequence begins again." ('076 patent, col. 35:38-44) Figure 5, which depicts a Selections File, shows "R" as the last record in the file. (*Id.*, FIG. 5) PA acknowledges that the "R LocType is an implementation of the endless loop[.]" (D.I. 146 at 21) Nevertheless, PA argued in the briefing that even if an endless loop is required, the R LocType is still not necessary "because the endless loop could also be implemented using the hyperlink functionality disclosed (Almeroth Decl. ¶ 96) or resetting the CurrentPlay variable to 1 as set forth in the current construction." (*Id.*) But PA cites to nothing in the patent that discloses such "alternative implementations . . . as connected to the recited function." (D.I. 159 at 23 n.15) Indeed, during the *Markman* hearing, PA appeared to concede that implementation of an endless loop in fact requires an R LocType. (*See, e.g.*, Tr. at 140 ("[T]he reason why you have to have this LocType R to implement a loop is because this algorithm here doesn't do it naturally."); *id.* at 147 ("[Y]ou need a LocType R to have a loop, and it's optional to have the loop."))

With respect to the second issue—i.e., whether "scanning the file to locate the next record using LocType is required for advancing to the record representing the next program segment in the sequencing file in the course of 'continuously reproducing'"—the Court is not persuaded that these new steps are necessary structure. In arguing that LocType scanning is necessary, Google asserts that LocType is used to differentiate between various types of records in the sequence, such as program segments, subject announcements, topic announcements, highlight passages, hyperlinks, or rewind. (D.I. 159 at 24 (citing '076 patent, FIG. 5, cols. 32:24-50 & 24:32-44); D.I. 162 at ¶ 28) According to Google, the disclosed algorithm must scan LocType in order to play the next program segment (instead of playing a subject announcement, topic announcement, etc.). (*Id.* at 25) However, the Court agrees with PA that the specification seems to describe subdividing program segments by subject and topic announcements as optional features; thus, it does not seem necessary to require the structure to scan forward in the sequencing file to locate the next Selection_Record containing the appropriate LocType. (D.I. 146 at 19-20 (citing '076 patent, col. 15:21-42)) Instead, as PA explains, it seems that scanning forward to locate the appropriate LocType is a step taken when the player has received a *command to skip over* program segments in the sequence file that are not related to a specific subject or topic. (*Id.* at 20 n.1 (citing '076 patent, col. 15:28-38); Tr. at 145-46; *see also, e.g.*, '076 patent, col. 34:36-64) The recited function, however, requires playing the audio program segments "in the order established by said sequence in the absence of a control command[.]" ('076 patent, col. 46:23-25)

3. Conclusion

For the above reasons, the Court recommends that Steps 1 and 2 of PA's structure be adopted, and Step 3 of Google's structure be adopted (with the exception that "received" not be included in this step, for the reasons explained in the Court's January 16 R&R).

B. "processor . . . for discontinuing"

The claim term "processor . . . for discontinuing the reproduction of the currently playing audio program file and instead continuing the reproduction at the beginning of a listener-selected one of said audio program files in said collection in response to a program selection command from said listener" (also referred to by the parties as the "Go command") appears in claims 1-13 of the '178 patent. (*See* PA's Markman Presentation, Slide 150; Google's Markman Presentation, Slide 54) The use of the disputed term in claim 1 of the '178 patent is representative. Accordingly, this claim is reproduced below, with the disputed term highlighted:

> **1.** An audio program player comprising:
>
> a communications port for establishing a data communications link for downloading a plurality of separate digital compressed audio program files and a separate sequencing file from one or more server computers;
>
> a digital memory unit coupled to said communications port for persistently storing said separate digital compressed audio program files and said separate sequencing file, said sequencing file containing data specifying an ordered sequence of a collection of said separate digital compressed audio program files,
>
> an audio output unit including at least one speaker or headset for reproducing said audio program files in audible form perceptible to a listener,
>
> one or more manual controls for accepting commands from said listener, and
>
> *a processor* for continuously delivering a succession of said audio program files in said collection to said audio output unit in said ordered sequence specified by said sequencing file in the absence of a program selection command from said listener, and *for discontinuing the reproduction of the currently playing audio program file and instead continuing the reproduction at the beginning of a listener-selected one of said audio program files in said collection in response to a program selection command from said listener.*

('178 patent, cols. 45:60-46:18 (emphasis added))

The parties agree that this limitation is a means-plus-function limitation, governed by 35 U.S.C. § 112(6). (D.I. 146, Appendix A at Ref. 10) The parties' competing proposed constructions for the "processor . . . for discontinuing" term are set out in the chart below, with the main disputed points noted by bold, underlined language:

Term	Plaintiff's Proposed Construction	Defendant's Proposed Construction
"processor . . . for discontinuing the reproduction of the currently playing audio program file and instead continuing the reproduction at the beginning of a listener selected one of said audio program files in said collection in response to a program selection command from said listener"	<u>Function</u>: "[I]n response to a 'Go' command, discontinuing the reproduction of the currently playing audio program file and instead continuing the reproduction at the beginning of a listener-selected one of said audio program files in said collection." <u>Structure</u>: "The structure corresponding to the claimed function can be the following structures and equivalents thereof: A general purpose computer programmed to perform the algorithm that is illustrated in the flow chart of Figure 3 at items 269, 235, 265, and 267 and more fully described at column 14, lines 25 to 29; column 14, lines 35 to 39; column 33, line 3 to line 8; and column 34, line 19 to column 35, line 52. Specifically, this algorithm includes the following steps:	<u>Function</u>: "In response to a 'Go' command, discontinuing the reproduction of the currently playing audio program file and instead continuing the reproduction at the beginning of a listener-selected one of said audio program files in said collection." <u>Structure</u>: "A general purpose computer programmed to perform the algorithm in the flow chart of Figure 3 at items 269, 235, and described at column 14, lines 25 to 26; column 14, lines 35 to 39; column 33, line 11 to line 16; column 34, line 19 to column 35, line 52; and column 35, line 54 to column 36, line 9. Specifically, the algorithm includes the following steps: **<u>(1) identifying the listener-selected Selection Record identified by the offset within the "L" Selection Record of the</u>**

	1. resetting the CurrentPlay variable to the record number of the listener-selected Selection_Record; and 2. fetching and playing the audio program file identified by the ProgramID contained in the new Selection_Record."	**hyperlink passage in the received sequencing file;** (2) resetting the CurrentPlay variable to the record number of the listener-selected Selection_Record; and (3) fetching and playing the audio program file identified by the ProgramID contained in the new Selection_Record."

(D.I. 146, Appendix A at Ref. 10) As reflected in these proposals, the crux of the parties' dispute regarding this term is whether the corresponding algorithmic structure requires an initial step of "identifying the listener-selected Selection_Record identified by the offset within the 'L' Selection_Record of the hyperlink passage in the received sequencing file" (the "identifying step").

Google asserts that this step is required because the processor must first identify the listener-selected file before going to that file. (D.I. 159 at 15-16; D.I. 186 at 11; Tr. at 150, 152 (Google's counsel asserting that "for the [G]o command, figuring out where to go is something that needs to be disclosed as part of the algorithm")) To that end, according to Google, the identifying step "matches the only disclosure in the specification" of responding to the "Go command[,]" (D.I. 159 at 15):

> Whenever the user issues a "Go" command (seen at **265** in FIG. **3**), the player will execute a hyperlink jump to the location indicated by the last "L" record in the selection file. When the jump is made, the location in the "L" record is inserted into the CurrentPlay register **353** . . . .

('076 patent, col. 35:61-66 (quoted in D.I. 159 at 15-16)) Google asserts that PA's proposed construction (that does not include this identifying step) is deficient because while it does recite how a user *initiates* a Go command, it does not disclose how to *respond* to the Go command. (D.I. 159 at 16; D.I. 186 at 11; Tr. at 152, 154)

The Court, however, is not persuaded that Google's identifying step is necessary structure to perform the claimed function. It so concludes for three primary reasons.

First, the Court notes that in other related proceedings, the claims were not construed in the manner Google now proposes. In the claim construction proceedings in the *Apple* litigation, the Eastern District of Texas Court did not include any identifying step in its corresponding structure for the Go command. (*See* D.I. 147, ex. C at Appendix B at 10) Nor did the PTAB's construction in the IPR proceeding relating to the '178 patent include an identifying step as corresponding structure. (D.I. 147, ex. H at 9-10) Instead, both tribunals' constructions mirrored PA's proposal here. It is true that in providing its Final Written Decision, the PTAB was utilizing a different standard for claim construction (the broadest reasonable interpretation, or "BRI," standard) than the one the Court must use here. (*See* D.I. 147, ex. H at 8)[3] Even so, the PTAB took pains to note that its interpretation of the Go command claim term "is consistent" with the findings of the Eastern District of Texas Court, "which interpreted [this term] . . . under 35 U.S.C. § 112, ¶ 6." (*Id.* at 9) While the decisions of these other entities are not dispositive of the issue here, the Court does find it persuasive that two other tribunals did not think it necessary to have the identifying step in the structure in order to execute the performance of the Go

[3] Pursuant to a recent rule change, beginning on November 13, 2018, the PTAB began to construe claims (including in IPR proceedings) by utilizing the same *Phillips* claim construction standard used for claim construction in a civil action in federal district court. *See* Changes to the Claim Construction Standard for Interpreting Claims in Trial Proceedings Before the Patent Trial and Appeal Board, 83 Fed. Reg. 51,340 (Oct. 11, 2018) (to be codified at 37 C.F.R. pt. 42). Thus, while the PTAB correctly utilized the BRI standard during the IPR proceedings relevant to this case, the PTAB no longer uses that standard.

command. *Cf. W.L Gore & Assoc., Inc. v. C.R. Bard, Inc.*, Civil Action No. 11-515-LPS-CJB, 2014 WL 3950663, at *6 (D. Del. Aug. 8, 2014) (treating another court's construction of identical claim terms in a related patent as "'persuasive authority'") (quoting *Biovail Labs. Int'l SRL v. Intelgenx Corp.*, Civ. No. 09-605-LPS, 2010 WL 5625746, at *5 (D. Del. Dec. 27, 2010)); *see also Clear with Computs., LLC v. AGCO Corp.*, CASE NO. 6:12-CV-622, 6:13-CV-161, 2014 WL 2700376, at *2 (E.D. Tex. June 13, 2014) (stating that the court "gives reasoned deference to prior claim construction rulings involving common terms in related patents").

Second, the Court agrees with PA that the patent discloses sufficient algorithmic structure for the claimed Go command (including "where to go" in response to the Go command) and so does not require inclusion of Google's identifying step. (Tr. at 153; *see also* D.I. 146 at 26) To that end, the patent discloses an embodiment of the Go command as simply receiving identification of the listener-selected segment, and then going to that segment. (*See* D.I. 146 at 26)

In making the argument that Google's proposed identifying step is unnecessary, PA first emphasizes that the recited function for this term requires: (1) stopping play of the currently playing program file in response to a command from the listener; and (2) instead continuing play at the beginning of the audio program file selected by the listener. (*Id.*) The specification, in describing one embodiment involving a player implemented by a personal computer with a screen display, explains the following:

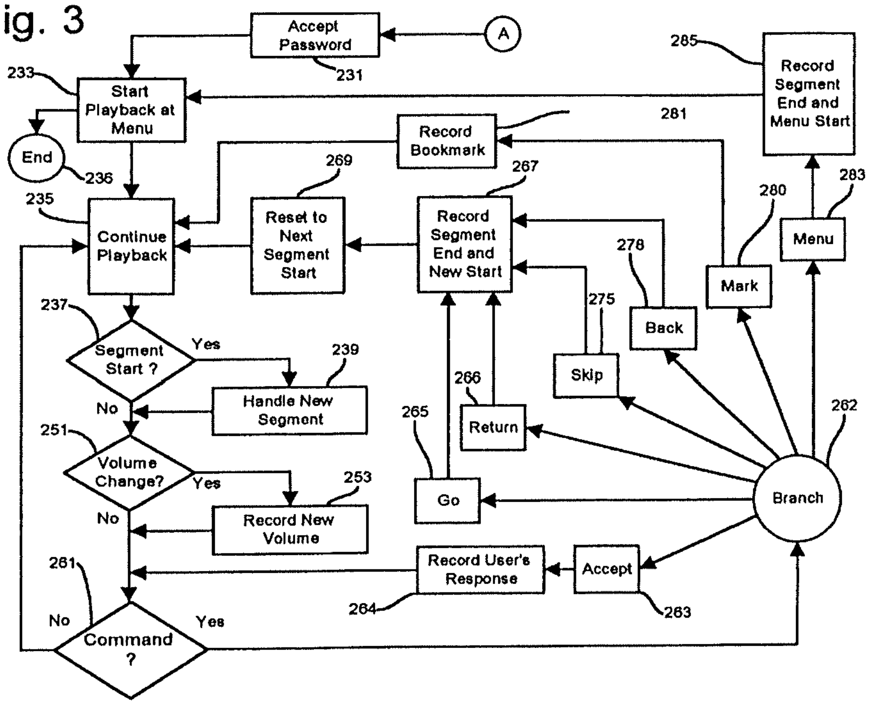
> [T]he current playback position may be advantageously indicated by displaying the program segment topic descriptions in a scrolling listing, with the description of the program currently being displayed being highlighted. The scheduled duration of each program segment may be displayed, along with the elapsed time remaining to be played in the currently playing segment, to enable the user to more easily determine when to skip the remainder of the

currently playing segment. *When such a concurrent visual display is available, means may also be included to respond to the user[']s selection of a given program on the scrollable listing by means of a mouse or the like, and then automatically continue the play at the beginning of the program segment thus selected.*

('178 patent, col. 12:49-63 (emphasis added)) With this embodiment, the user's act of *selecting* a program from the menu will also provide the requisite identification of *where to go next* (i.e., how to respond to the command). (*See* D.I. 176 at 12)

This embodiment is revisited later when the patent expressly discusses the "Go command." As can be seen in Figure 3 below, the "Go" command at 265 is one of the commands available from step 262. In response to the Go command, the player discontinues reproduction of the current segment and resets to the listener selected segment at step 269, and continues playback of the new selection at step 235. ('178 patent, FIG. 3; *see also* D.I. 146 at 26)

Fig. 3



The specification further explains that:

> A first command, "Go" indicated at **265**, causes the player to make an *immediate* shift to a different program segment. For example, the spoken voice command "FIVE" can indicate a request to go to a *predetermined* numbered program segment while the spoken command "NEWS" could switch to the subject announcement segment for news programs. Alternatively, a mouse click on a screen-displayed menu of items, or a push-button on a hand controller connected by an infrared link to the player computer, could similarly be processed as a command to go to a *predetermined* program segment associated with that command signal. In such cases, the system records the start of the new segment on the log file (seen at **215** in FIG. **2**) at **267** and switches the current playback position in the program sequence file **214** to the new setting at **269**, and the playback continues at **235**.

('178 patent, col. 14:25-39 (quoted in D.I. 146 at 26) (emphasis added)) This disclosure supports PA's description of the Go command as "simply [] receiving the identification of the segment from the listener" and then playing that segment. (*See* D.I. 146 at 26) That is, according to this description, the listener could: (1) provide a spoken command; (2) use a mouse to click an item on a menu; or (3) push a button on a hand controller—each of which would be processed by the player as a command to "immediate[ly]" go to a "predetermined" segment associated with that command. (*Id.*; *see also* Tr. at 153-54 (PA's counsel explaining that the algorithm "does tell you where to jump to. . . . [W]hen you say the spoken word five, [for example,] that takes you to position five.")) In other words, it is not necessary for the algorithmic structure to include a separate step dedicated to figuring out where to go "in order to respond to the go command issued by the user[,]" (Tr. at 154-55), because the Go command receives a request from the user that *tells the player where to go*, (*see id.* at 153; PA's Markman Presentation, Slide 166). This portion of the specification describing the Go command makes no mention of using an L LocType to perform the function. (*See* D.I. 146 at 27; D.I. 176 at 12)

Third, after carefully considering the specification language that Google points to as supporting its identifying step, the Court agrees with PA that this disclosure merely describes an alternative embodiment. This portion of the specification explains that:

> Hyperlinks are implemented by means of anchor passage identifiers, the "A" and "B" Selection records which respectively identify the anchor passage, and a "L" link identifier which holds the location of a subject, topic or highlight Selection_Record. The "A" and "B" selection records enable the player to add an audio cue (such as a tone, low-level chime, or the like) to the beginning, end, or during any passage in any program selection. Whenever the user issues a "Go" command (seen at **265** in FIG. 3), the player will execute a hyperlink jump to the location indicated by the last "L" record in the selection file.

('178 patent, col. 35:43-53) According to Google, the language "whenever the user issues a 'Go' command" signals that the identifying step at issue is not optional. (D.I. 159 at 16) But when considered as a whole, this passage—a passage located over 20 columns after the patent's first explicit description of the "Go" command—is clearly discussing the implantation of hyperlinks. And it seems to be describing, at a minimum, a special scenario where hyperlinked Selection_Records embedded in the sequence can play an audio cue, and the user can then issue a "Go" command causing the player to retrieve the linked "L" Selection_Record. (*See* D.I. 146 at 27; D.I. 176 at 12-13) This seems to be a different scenario than, for example, a Go command issued by a listener via selecting an item on a visual menu, as described above. (PA's Markman

Presentation, Slide 165)[4] Accordingly, this description of one embodiment should not limit the algorithmic structure for the Go command.[5]

C. "means for detecting a first command"

---

[4] PA points to an earlier portion of the specification (which comes right after the initial description of the Go command) that supports the idea that Google's identifying step is merely one embodiment of many, and thus should not limit the algorithmic structure for the "processor . . . for discontinuing" term:

> In the preferred arrangement, described in more detail in conjunction with FIG. **5** of the drawings, the program being played may contain passages which relate to other program segments in the compilation. These passages may be indicated by direct announcement, such as: "Say 'Go' when any of the following automotive companies are named to obtain additional information: . . . Ford . . . General Motors . . . Chrysler . . . Honda . . ." *Alternatively, an audible cue signal, such a distinctive tone or chime, might immediately precede a passage which anchors a link to another program segment. . . . When a cue signal indicates a hyperlink passage, a simple "Go" voice command causes the player to reset to a new location from which playing continues*[.]

('178 patent, col. 14:40-55 (emphasis added) (quoted in PA's Markman Presentation, Slide 161)) The language emphasized in this passage mimics the specification language that Google relies on in advocating for the identifying step, and describes that scenario as an "[a]lternative[]" one.

[5] The specification does go on to note that "[t]he hyperlink capability described above may be used to implement a program menu of the type described earlier in connection with FIG. **3**." ('178 patent, col. 35:64-66) Google latches onto this language to argue that the specification thus states that the identifying step is used whenever the user issues a Go command—including in the context of going to a user selected file on a visual menu display. (D.I. 186 at 11) This argument lacks merit, however. The specification says that the identifying step *may* be used in relation to program menus. And as PA points out, the very next sentence indicates that this portion of the specification is referring to a *spoken* menu, not the visually displayed menu discussed earlier in conjunction with the Go command. (PA's Markman Presentation, Slide 171 (quoting '178 patent, cols. 35:66-36:3 ("A menu program segment may be included in the program compilation which includes a series of *spoken* descriptions of subjects or topics, each description being the anchor portion of a hyperlink to the corresponding subject or topic.") (emphasis added))) In the Court's view, this language just underscores that the identifying step Google seeks to include in the construction is merely an alternative embodiment.

The claim term "means for detecting a first command indicative of a request to skip forward" appears, *inter alia*, in claim 1 of the '076 patent. (*See* Google's Markman Presentation, Slide 63) For ease of reference, this claim is reproduced again below, with the disputed term highlighted:

> **1.** A player for reproducing selected audio program segments comprising, in combination:
>
> means for storing a plurality of program segments, each of said program segments having a beginning and an end,
>
> means for receiving and storing a file of data establishing a sequence in which said program segments are scheduled to be reproduced by said player,
>
> means for accepting control commands from a user of said player,
>
> means for continuously reproducing said program segments in the order established by said sequence in the absence of a control command,
>
> *means for detecting a first command indicative of a request to skip forward*, and
>
> means responsive to said first command for discontinuing the reproduction of the currently playing program segment and instead continuing the reproduction at the beginning of a program segment which follows said currently playing program in said sequence.

('076 patent, col. 46:13-33 (emphasis added))

The parties agree that this limitation is a means-plus-function limitation, governed by 35 U.S.C. § 112(6). (D.I. 146, Appendix A at Ref. 11) The parties' competing proposed constructions for the "means for detecting a first command" term are set out in the chart below, with the main disputed points noted by bold, underlined language:

Term	Plaintiff's Proposed Construction	Defendant's Proposed Construction
"means for detecting a first command indicative of a request to skip forward"	<u>Function</u>: "'[D]etecting a first command indicative of a request to skip forward.'" <u>Structure</u>: "The structure corresponding to the claimed function is the following structures and equivalents thereof: A general purpose computer programmed to perform the algorithm that is illustrated in the flow chart of Figure 3 at items 261, 262, and 275. Specifically, this algorithm includes the following steps: 1. determining whether input from the means for accepting control commands is a command using a <u>**conditional**</u> programming construct; and 2. if the input is a command, using a 'branch' programming construct to select one of the player's available commands, which include a 'Skip' command, for execution."	<u>Function</u>: "Detecting a first command indicative of a request to skip forward." <u>Structure</u>: "A general purpose computer programmed to perform the algorithm that is illustrated in the flow chart of Figure 3 at items 261, 262, 275. Specifically, this algorithm includes the following steps: (1) determining whether input from the means for accepting control commands is a command using a <u>**'if-then-else'**</u> programming construct; and (2) if the input is a command, using a 'branch' programming construct to select one of the player's available commands, which include a 'Skip' command, for execution."

(D.I. 146, Appendix A at Ref. 11) The parties agree that the corresponding structure for this term includes items 261 (Command?), 262 (Branch), and 275 (Skip) or 278 (Back) of Figure 3. (D.I. 146 at 28; D.I. 159 at 26) With respect to this limitation, the specification explains that the player is equipped with a keyboard and microphone that accept keyed or voice commands, respectively, which control playback. ('076 patent, col. 13:49-52) Such a command may be entered while playback is continuing, thus interrupting playback; this command is depicted by the "Command?" diamond at Item 261 (the "Command diamond"). (*Id.*, col. 13:52-53 & FIG. 3) When an input is received from the means for accepting control commands, the player must determine whether that input is a command. (*Id.* at FIG. 3 at Item 261 ("Command?")) If the player determines that the input is a command, operation proceeds to Item 262 in Figure 3, the "Branch" circle, as illustrated by the "Yes" arrow from Item 261 to Item 262 in Figure 3. (*Id.* at Items 261 & 262) At the Branch step, "the character of the command is evaluated . . . to select one of six different types of functions." (*Id.*, col. 13:53-55) One such function is the Skip command, indicated at item 275 in Figure 3, which "causes the player to advance to the beginning of the next program segment in the program sequence[.]" (*Id.*, col. 15:22-23)

The parties have a single dispute with respect to this term. That is, they disagree about whether the "Command?" diamond depicted in Item 261 is limited to an "if-then-else" programming construct (Google's position), or whether it encompasses any "conditional" programming construct (PA's position). (D.I. 146 at 28; D.I. 159 at 26; D.I. 186 at 14; Tr. at 156)

Previous constructions of the term (by the Eastern District of Texas Court in the *Apple* litigation and the PTAB in the IPR proceeding) are relevant to this dispute. First, during the *Apple* litigation, defendant Apple, Inc. had argued that the specification of the '076 patent failed to disclose any algorithm for evaluating whether an input is a command or a non-command. (D.I. 160, ex. 2 at 23-24) PA retorted that the specification *did* disclose such an algorithm, albeit a simple one (i.e., (1) receiving the command; and (2) evaluating the character of the command) that is sufficient to enable a POSITA to understand the bounds of the claim. (*See id.*) The *Apple*

Court concluded that from the perspective of the POSITA in the field of computer science, the Command diamond would indicate an "if-then-else" computer programming construct for discriminating between commands and non-commands, and that such an algorithm amounted to sufficient structure. (*Id.* at 24) In support of its conclusion, the *Apple* Court cited to the *IEEE Standard Dictionary of Electrical and Electronics Terms* (6th ed. 1996), (*id.*), which defines "if-then-else" as "[a] single-entry, single-exit two-way branch that defines a condition, specifies the processing to be performed if the condition is met and, optionally, if it is not, and returns control in both instances to the statement immediately following the overall construct[,]" (D.I. 160, ex. 15 at 500).

Later, during the IPR proceeding relating to the '076 patent, PA argued that the *Apple* Court's "if-then-else" programming construct was unduly narrow, in that it was limited to particular programming languages. (D.I. 147, ex. E at 22) PA asserted that a "'conditional' programming construct" would instead be more appropriate, as it would be consistent with the Command diamond while also encompassing other programming languages that use analogous terms such as "when-then-else" to accomplish the same function as "if-then-else." (*Id.* (certain internal quotation marks omitted)) The PTAB agreed with PA, noting that Google had "not pointed to persuasive evidence in the Specification that [the Command diamond] is limited to particular programming constructs appearing in specific programming languages." (*Id.* at 22-23) The PTAB also explained that it was not persuaded that a diamond-shaped box in and of itself "excludes analogous constructs from other programming languages." (*Id.* at 23) Accordingly, the PTAB modified the *Apple* Court's construction for the "means for detecting a first command" such that "if-then-else" was replaced with "conditional." (*Id.*)

Here, Google's argument against the "conditional" language is that it would amount to pure functional claiming; Google asserts that such a construction would allow the algorithm to be *any* conditional programming construct. (D.I. 159 at 27; D.I. 186 at 14; Tr. at 157-59) Meanwhile, PA claims that "if-then-else" is a specific term of art used in particular programming languages; it argues that reference to a "conditional" programming construct would be more appropriate, in that it would not limit the structure to a specific program language. (D.I. 146 at 28)

The record with respect to the appropriate construction of this term is not especially robust. As for the intrinsic record, beyond the Command diamond in the specification, the specification does not offer much in the way of guidance. (*See* D.I. 186 at 14 ("Neither party has cited intrinsic evidence on this point."); Tr. at 156-57 (Google's counsel reiterating that "[n]either side is really able to demonstrate something from the intrinsic record on this point. We're faced with just the diamond[] as in the figure"))

Turning to the extrinsic record, the parties' experts offer dueling opinions on the proper outcome here. PA's expert Dr. Almeroth first opined that the term "'conditional' is an apt description" for the Command diamond at Item 261, citing in support to two computer dictionary definitions of "conditional." (D.I. 147 at ¶ 109 (citing *id.*, ex. K (defining "conditional" as "[o]f, pertaining to, or characteristic of an action or operation that takes place based on whether or not a certain condition is true"); *id.*, ex. J (defining "conditional branch" as "an instruction that causes the computer to jump to another location in the program only if a specified condition is true"))) Dr. Almeroth also explained that the patents do not require the claimed inventions to use any particular programming language(s). (*Id.* at ¶ 108) He noted that while "'[i]f' and 'else' are actual operators used across a number of programming languages[,] [c]omputer languages also use slightly different syntax or operators, such as 'when/then/else' or a 'case' test (or both), which may be used to accomplish the same operation as if-then-else." (*Id.*)

For Google's part, its expert Dr. Ketan Mayer-Patel then responded that the Command diamond indicates "a particular type of conditional statement—a *simple* conditional statement with only two possible outcomes: Yes and No, with arrows indicating respective subsequent steps 262 and 235." (D.I. 162 at ¶ 21 (certain emphasis omitted)) Dr. Mayer-Patel explained that in programming, such a statement is known as an "if-then-else" statement, "a construct present in virtually all programming languages." (*Id.*) He then opined that PA's proposal to categorize the Command diamond as a conditional program construct "does not accurately or precisely define the type of programming construct disclosed" therein because the term "conditional" is much broader than an "if-then-else" programming construct "and could include any number of programming constructs" such as "a more complex conditional statement [that] could include a case statement, a switch statement, a branch statement, and a number of other programming constructs." (*Id.* at ¶ 22) And according to Dr. Mayer-Patel, these more complex conditional statements "may have any number of outcomes"—unlike a simple if-then-else outcome. (*Id.*)

In the Court's view, Google's position seems the better one. This is so for a few reasons, all of them stemming from the content of Dr. Almeroth's reply declaration on this issue. Indeed, in the Court's view, that reply declaration actually supports the conclusion that Google's proposal is the correct one.

For one thing, in his reply declaration, Dr. Almeroth does not substantively respond to Dr. Mayer-Patel's opinions at all with respect to this term. (D.I. 177 at ¶¶ 77-78)[6] That suggests that there may be no persuasive rebuttal.

Moreover, in his reply declaration, Dr. Almeroth significantly relies on the fact that "the PTO has already found that one skilled would understand the diamond box to correspond to a 'conditional' programming construct. [] This finding forms part of the prosecution history and is persuasive evidence for Plaintiff's position." (*Id.* at ¶ 78)[7] However, as discussed above, in the IPR proceeding, the BRI standard applied to claim construction issues. (*See* D.I. 159 at 27) Indeed, during the IPR proceeding, in arguing that the *Apple* Court's "if-then-else" language should not be adopted, *PA* expressly relied on the BRI standard in support. Patent Owner's Preliminary Response at 16, *Lenovo (United States) Inc. v. Personal Audio LLC*, Case IPR2015-00845 (P.T.A.B. June 20, 2015) ("Patent owner . . . disagrees with the incorporation of 'if-then-else' *as part of the broadest reasonable construction*.") (emphasis added); *id.* at 16-17 ("Patent Owner believes the *broadest reasonable construction* for [the Command diamond] should be a 'conditional' programming construct.") (emphasis added)) This suggests that while construing the algorithm at issue to encompass any "conditional" algorithm might be acceptable under the BRI standard, it may not be appropriate under the *Phillips* standard.

---

[6] In its reply brief, PA also did not provide a substantive counter to Dr. Mayer-Patel's position. PA's argument in that brief regarding this term was limited to a two-line conclusory sentence (citing to Dr. Almeroth's reply declaration). (*See* D.I. 176 at 15 (citing D.I. 177 at ¶ 77))

[7] During oral argument, PA's counsel echoed this point. (*See* Tr. at 159 ("I just want to say that the PTO took up this issue . . . [and] didn't limit the [Command diamond to an if-then-else construct] when the arrow suggests conditional. . . . There is an intrinsic record on this."))

Additionally, Dr. Almeroth's only other argument in his reply declaration is that the "conditional" programming construct would amount to sufficient structure because the "corresponding algorithm includes two separate steps: a first step that determines [whether the input is a command] and a second step that responds to the input using a 'branch' programming construct." (D.I. 177 at ¶ 78) He explains that the "claimed algorithm is not merely *any* conditional programming construct, but only those that use a 'branch' construct to select a player command for execution." (*Id.* (emphasis in original)) Thus, he opines, "regardless of how one interprets the first step, it is my opinion that the second step provides a further substantive limitation that renders the claim definite." (*Id.*)

But as Google points out, the law requires that sufficient structure be disclosed for the full scope of the claimed function of a means-plus-function term. (D.I. 186 at 14 (citing *Twin Peaks Software Inc. v. IBM Corp.*, 690 F. App'x 656, 665 (Fed. Cir. 2017); *Media Rights Techs. Inc. v. Capital One Fin. Corp.*, 800 F.3d 1366, 1374 (Fed. Cir. 2015)); Tr. at 158-59) And thus Dr. Almeroth's position here (that "regardless of how one interprets the first step" of determining whether the input is a command or not, the "branch" construct of the second step provides sufficient structure for the term) is not persuasive. Before it can proceed to that second step, the player must go through the first step of determining whether the input is a command, and there must be sufficient structure for doing so. It is not in dispute that a diamond shape, such as that representing the Command step, indicates an "if-then-else" construct, which provides sufficient structure for the "determining" part of the algorithm. Nor does it seem to be disputed that "a number of programming languages" utilize such a statement. (*See* D.I. 147 at ¶ 108 (Dr. Almeroth opining that "'[i]f'" and "'else'" are "actual operators used across a number of programming languages"); D.I. 162 at ¶ 21 (Dr. Mayer-Patel opining that an if-then-else construct is "present in virtually all programming languages"); Patent Owner's Preliminary Response at 17, *Lenovo (United States) Inc. v. Personal Audio LLC*, Case IPR2015-00845 (P.T.A.B. June 20, 2015) (PA noting that "'if' and 'else' are actual operators used across a number of programming languages"))

For these reasons, the Court recommends that Google's proposed structure for the "means for detecting a first command" term be adopted.

III. CONCLUSION

For the foregoing reasons, the Court recommends that the District Court adopt the following constructions:

1. For the term "means for continuously reproducing said program segments in the order established by said sequence in the absence of a control command" the function is "continuously reproducing said program segments in the order established by said sequence in the absence of a control command." The corresponding structure for this term is: "A sound card that includes a digital to analog converter; headphones or one or more speakers; and a general purpose computer programmed to perform the algorithm that is illustrated in the flow chart of Figure 3 at items 233, 235, 237, 239, and 261 and more fully described at column 12, line 16 to column 13, line 11 and column 34, line 28 to column 35, line 44. Specifically, this algorithm includes the following steps:

   1. beginning playback with the program segment identified by the ProgramID contained in the Selection_Record specified by the CurrentPlay variable;

   2. when the currently playing program concludes, incrementing the CurrentPlay variable by one and fetching and playing the program segment identified by the ProgramID contained in the next Selection_Record in the sequencing file;

3. repeating step (2) until a rewind Selection_Record (LocType: R) in the sequencing file is reached, which resets the CurrentPlay variable to the location value contained in the rewind Selection_Record which is set to "1" to begin the playing sequence again with the first Selection_Record in the sequencing file."

2. For the term "processor . . . for discontinuing the reproduction of the currently playing audio program file and instead continuing the reproduction at the beginning of a listener selected one of said audio program files in said collection in response to a program selection command from said listener" the function is "in response to a 'Go' command, discontinuing the reproduction of the currently playing audio program file and instead continuing the reproduction at the beginning of a listener-selected one of said audio program files in said collection." The corresponding structure for this term is: "A general purpose computer programmed to perform the algorithm that is illustrated in the flow chart of Figure 3 at items 269, 235, 265, and 267 and more fully described at column 14, lines 25 to 29; column 14, lines 35 to 39; column 33, line 3 to line 8; and column 34, line 19 to column 35, line 52. Specifically, this algorithm includes the following steps:

1. resetting the CurrentPlay variable to the record number of the listener-selected Selection_Record; and

2. fetching and playing the audio program file identified by the ProgramID contained in the new Selection_Record."

3. For the term "means for detecting a first command indicative of a request to skip forward" the function is "detecting a first command indicative of a request to skip forward." The corresponding structure for this term is: A general purpose computer programmed to perform the algorithm that is illustrated in the flow chart of Figure 3 at items 261, 262, 275. Specifically, this algorithm includes the following steps:

1. determining whether input from the means for accepting control commands is a command using a 'if-then-else' programming construct; and

2. if the input is a command, using a 'branch' programming construct to select one of the player's available commands, which include a 'Skip' command, for execution."

This Report and Recommendation is filed pursuant to 28 U.S.C. § 636(b)(1)(B), Fed. R. Civ. P. 72(b)(1), and D. Del. LR 72.1. The parties may serve and file specific written objections within fourteen (14) days after being served with a copy of this Report and Recommendation. Fed. R. Civ. P. 72(b)(2). The failure of a party to object to legal conclusions may result in the loss of the right to de novo review in the district court. *See Henderson v. Carlson*, 812 F.2d 874, 878-79 (3d Cir. 1987); *Sincavage v. Barnhart*, 171 F. App'x 924, 925 n.1 (3d Cir. 2006).

The parties are directed to the Court's Standing Order for Objections Filed Under Fed. R. Civ. P. 72, dated October 9, 2013, a copy of which is available on the District Court's website, located at http://www.ded.uscourts.gov.

Dated: March 13, 2019

Christopher J. Burke
Christopher J. Burke
UNITED STATES MAGISTRATE JUDGE