UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

GOOGLE LLC

Petitioner

v.

ART RESEARCH AND TECHNOLOGY LLC

(record) Patent Owner

Patent No. 10,681,103

DECLARATION OF DR. HENRY HOUH

TABLE OF CONTENTS

TABLE OF CONTENTS....2
I. ENGAGEMENT AND COMPENSATION....4
II. SUMMARY OF OPINIONS....5
III. QUALIFICATIONS....5
IV. MY UNDERSTANDING OF THE RELEVANT LAW....8
A. Claim Construction....8
B. Anticipation....8
C. Obviousness....9
V. SUMMARY OF THE '103 PATENT....12
VI. RELEVANT TIMEFRAME FOR DETERMINING OBVIOUSNESS....18
VII. LEVEL OF ORDINARY SKILL IN the ART....19
VIII. DETAILED EXPLANATION OF THE REASONS FOR UNPATENTABILITY....19
Ground 1. Claims 1-10 and 13-22 are obvious over Hedinsson in view of Ford 19
A. Overview of the Ground....20
1. Overview of Hedinsson....21
2. Overview of Ford....25
B. Rationale (Motivation) Supporting Obviousness....26
C. Reasonable Expectation of Success....31
D. Graham Factors....31
E. Analogous Art....32
F. Claim Mapping....32

Ground 2. Claims 11-12 and 23-24 are obvious over Hedinsson in view of Ford and Kostello....77

A. Overview of the Ground....78

1. Overview of Kostello....78

B. Rationale (Motivation) Supporting Obviousness....82

C. Reasonable Expectation of Success....87

D. Graham Factors....87

E. Analogous Art....87

F. Claim Mapping....88

IX. OATH....111

1. I, Henry Houh, hereby declare as follows:

I. ENGAGEMENT AND COMPENSATION

2. I have been retained by Google LLC ("Petitioner") as an expert in the *inter partes* review proceeding described above. I have been asked to provide my technical review, analysis, insights, and opinions regarding the references that form the basis for the grounds of unpatentability set forth in the Petition for *inter partes* review of US 10,681,103 (the '103 Patent). For this service, my billing rate is my standard hourly consulting rate, and I am reimbursed for actual expenses. My compensation in no way depends on the outcome of this matter.

3. This declaration sets forth my analyses and opinions based on the materials I have considered thus far and the bases for my opinions. I understand that this declaration will be used in the above mentioned *inter partes* review. In forming my opinions, I have considered the following Exhibits:

Exhibit No.	Description
1001	U.S. Patent No. 10,681,103 B2 ("**'103 patent**").
1004	U.S. Patent No. 8,793,282 B2 ("**Hedinsson**").
1005	Excerpts of J.L. Ford, *Programming for the Absolute Beginner* (2007) ("**Ford**").
1006	U.S. Patent No. 8,745,500 B1 ("**Kostello**").
1007	Prosecution History of U.S. Application No. 16/134,541 (which issued as the '103 patent).

1008	U.S. Patent Appl. Pub. No. 2005/0084232 A1.
1009	Declaration of June Ann Munford.
1010	U.S. Patent Appl. Pub. No. 2012/0284343 A1.
1011	U.S. Patent Appl. Pub. No. 2014/0229866 A1.
1012	U.S. Patent Appl. Pub. No. 2015/0185965 A1.
1013	Redline Versions of Claims 4, 5, 8-10, and 13-24 of U.S. Patent No. 10,681,103

II. SUMMARY OF OPINIONS

4. It is my opinion that:

- Ground 1: Claims 1-10 and 13-22 are obvious over Hedinsson and Ford.
- Ground 2: Claims 11, 12, 23, and 24 are obvious over Hedinsson, Ford, and Kostello.

III. QUALIFICATIONS

5. I hold a Ph.D. in electrical engineering and computer science from the Massachusetts Institute of Technology ("MIT"), awarded in 1998. I also hold a Master's Degree in electrical engineering and computer science from MIT, awarded in 1991, and two Bachelor's degrees (the first in electrical engineering and computer science and the second in physics) also from MIT, awarded in 1989 and 1990, respectively.

6. I have approximately 25 years of work experience related to topics in networking, Internet communications, Web development, social networking, Internet advertising, and Web-based digital media. I am currently a technical consultant specializing in Social Networking, Web 2.0, Web Site Development, Data Networking, Optical Networking, Telecommunications, Media Streaming and Voice over IP.

7. I have extensive experience with Web programming, Web-based digital media, and Web-based advertisement technologies. I am familiar with social networking, annotation of media, including static and playable media.

8. From 1994-1996, I was the founder and CEO of Agora Technology Group, where I conceived and oversaw development of targeted advertising-supported web sites. From 1997-1999, I was a Software Engineer and Senior Scientist and Engineer for NBX (acquired by 3Com), where I worked on IP telephony (telephony using Internet protocol). From 1999-2004, I worked for Teradyne and later its spinout (Empirix). During this period, I held the position of Chief Technologist of the Web Application Test Group. In that position, I was the Chief Architect for all Web testing products, re-architected the core testing product, and helped write a JavaScript interpreter.

9. In 2006, I was the Vice President of Operations and Technology for PodZinger Inc. There, I helped create and launch a consumer-facing search site

for audio and video and redeployed the web site from our company's machine room to a co-location facility. I was also responsible for evaluating advertisement models of revenue. The PodZinger search site indexed audio and video located all over the web, and used speech recognition and natural language processing to create a spoken-word transcripts of the media files, and augmented the audio/video search results with snippets of the transcripts. When a search result was selected from the results, the Podzinger web site provided a media page with a player integrated with a full transcript of the spoken words, where each word in the transcript was selectable by the user and would jump the player to the exact location of the selected spoken word within the audio or video file. Podzinger was named by Forbes Magazine as their favorite Video & Audio Search Engine in 2006, beating Google Video Search, Yahoo Video Search and others.

10. From 2007-2008, I was Vice President of Technology for the Delta Division of BBN Technologies, where I oversaw the development and deployment of a new fully-featured social networking web site. This social networking site allowed for user to share, discuss, and annotate media. From 2008-2009, I was the Chief Technology Officer for Eons. With Eons, I integrated a product I created at BBN that Eons acquired. I integrated the product into the Eons social networking platform. I also helped evaluate advertisement platform offerings and brought the "Boom Network" advertising network to market.

11. My qualifications, publications, and patents are set forth more fully in my curriculum vitae, attached as Ex-1003.

IV. MY UNDERSTANDING OF THE RELEVANT LAW

A. Claim Construction

12. I understand that a claim term in an *inter partes* review is to be interpreted according to standards applied by district courts. First, the language of the claims themselves is of primary importance in the effort to determine precisely what it is that is patented. The terms used in a claim are generally given the ordinary and customary meaning that the terms would have to a person of ordinary skill in the art in question at the time of the alleged invention, unless the term is expressly defined in the patent. The person of ordinary skill in the art reads the claim in the context of the entire patent, including the specification. Next to the language of the claims, the specification is the single best source for interpreting the claim terms. The claims may also be interpreted using the record of correspondence between the patent applicant and the Patent Office, and also with reference to other sources of evidence that can help to define the meaning of terms to a person of ordinary skill in the relevant time frame. I define the relevant time frame in ¶33, below.

B. Anticipation

13. I understand that a claim in an issued patent can be invalid if it is anticipated. In this case, "anticipation" means that there is a single prior art

reference that discloses every element of the claim, arranged in the way required by the claim.

14. I understand that an anticipating prior art reference must disclose each of the claim elements expressly or inherently. I understand that "inherent" disclosure means that the claim element, although not expressly described by the prior art reference, must necessarily be present based on the disclosure. I understand that a mere probability that the element is present is not sufficient to qualify as "inherent disclosure".

C. Obviousness

15. I understand that a claim in an issued patent can be invalid if it is obvious. Unlike anticipation, obviousness does not require that every element of the claim be in a single prior art reference. Instead, it is possible for claim elements to be described in different prior art references, so long as there is motivation or sufficient reasoning to combine the references.

16. I understand that a claim is invalid for obviousness if the differences between the claimed subject matter and the prior art are such that the subject matter as a whole would have been obvious at the time the alleged invention was made to a person having ordinary skill in the art to which said subject matter pertains.

17. I understand, therefore, that when evaluating obviousness, one must consider obviousness of the claim "as a whole". This consideration must be from the perspective of the person of ordinary skill in the relevant art, and that such perspective must be considered as of the "time the invention was made".

18. The level of ordinary skill in the art is discussed in ¶35 below.

19. The relevant time frame for obviousness, the "time the invention was made", is discussed in ¶33, below.

20. I understand that in considering the obviousness of a claim, one must consider four things. These include the scope and content of the prior art, the level of ordinary skill in the art at the relevant time, the differences between the prior art and the claim, and any "secondary considerations". I understand that these four things were specified by the U.S. Supreme Court in a case called *Graham v. John Deere*, and are therefore called the "*Graham* factors".

21. I understand that "secondary considerations" include real-world evidence that can tend to make a conclusion of obviousness either more probable or less probable. For example, the commercial success of a product embodying a claim of the patent could provide evidence tending to show that the claimed invention is not obvious. In order to understand the strength of the evidence, one would want to know whether the commercial success is traceable to a certain aspect of the claim not disclosed in a single prior art reference (i.e., whether there

is a causal "nexus" to the claim language). One would also want to know how the market reacted to disclosure of the invention, and whether commercial success might be traceable to things other than innovation, for example the market power of the seller, an advertising campaign, or the existence of a complex system having many features beyond the claims that might be desirable to a consumer. One would also want to know how the product compared to similar products not embodying the claim. I understand that commercial success evidence should be reasonably commensurate with the scope of the claim, but that it is not necessary for a commercial product to embody the full scope of the claim.

22. Other kinds of secondary considerations are possible. For example, evidence that the relevant field had a long-established, unsolved problem or need that was later provided by the claimed invention could be indicative of non-obviousness. Evidence that others had tried, but failed to make an aspect of the claim might indicate that the art lacked the requisite skill to do so. Evidence of copying of the patent owner's products before the patent was published might also indicate that its approach to solving a particular problem was not obvious. Evidence that the art recognized the value of products embodying a claim, for example, by praising the named inventors' work, might tend to show that the claim was non-obvious.

23. I further understand that prior art references can be combined where there is an express or implied rationale to do so. Such a rationale might include an expected advantage to be obtained, or might be implied under the circumstances. For example, a claim is likely obvious if design needs or market pressures existing in the prior art make it natural for one or more known components to be combined, where each component continues to function in the expected manner when combined (i.e., when there are no unpredictable results). A claim is also likely invalid where it is the combination of a known base system with a known technique that can be applied to the base system without an unpredictable result. In these cases, the combination must be within the capabilities of a person of ordinary skill in the art.

24. I understand that when considering obviousness, one must not refer to teachings in the specification of the patent itself. One can, however, refer to portions of the specification admitted to be prior art, including the "Background" section. Furthermore, a lack of discussion in the patent specification concerning how to implement a disclosed technique can support an inference that the ability to implement the technique was within the ordinary skill in the prior art.

V. SUMMARY OF THE '103 PATENT

25. I have reviewed the '103 patent. The '103 patent is titled "Social Networking with Video Annotation." Ex-1001. The '103 patent describes the

field of the invention as "relat[ing] to assemblies, methods, devices, and systems for managing information, and more particularly, to assemblies, methods, devices, and systems for sharing and annotating video data between members of a social network." Ex-1001, 1:21-25.

26. The '103 patent has 24 claims. Of those, there are 4 independent claims—claims 1, 7, 13, and 19. Independent claim 1 is reproduced below, with bracketed numbering added to each claim limitation:

> 1[a]. A method to make a composite virtual clip, comprising:
>
> [1b] selecting (N) saved virtual clips and an order of presentation for each virtual clip;
>
> [1c] initially setting (M)=1;
>
> [1d] configuring a (M)th pointer to indicate a storage location for a (M)th saved virtual clip, wherein a (M)th virtual clip comprises a (M)th content from one or more designated sources and (M) is greater than or equal to 1 and less than or equal to (N);
>
> saving said (M)th pointer;
>
> [1e] determining if (M) equals (N);
>
> when (M) is less than (N), setting (M) equal to (M)+1; and
>
> repeating said configuring, saving, and determining until (M) equals (N).

Ex-1001, claim 1. Independent claim 7 is reproduced below, with bracketed numbering added to each claim limitation:

7[a]. A method to display a composite virtual clip, comprising:

[7b] providing a storage location for a composite virtual clip stored in a composite virtual clip file, wherein the composite virtual clip is configured to access (N) saved virtual clips;

[7c] initially setting (P)=1;

[7d] activating a (P)th link encoded in the composite virtual clip file to stream a (P)th saved virtual clip to a user's device;

[7e] determining if (P) equals (N);

when (P) is less than (N), setting (P) equals (P)+1; and

repeating said activating and determining until (P) equals (N).

Ex-1001, claim 7. Independent claims 13 and 19 are directed to a "non-transitory computer usable medium encoded with a computer program product to" make or display "a composite virtual clip and usable with programmable computer processor disposed within a controller" comprising "computer readable program code which causes said programmable computer processor to" perform the steps recited in claims 1 and 7, respectively.

27. The claims of the '103 patent recite pseudocode[1] that combines two known concepts: (1) a **programming loop** and (2) a **composite virtual clip** that includes separate **virtual clips** (e.g., video clips) that are each identified using a **pointer** or **link**. The claimed programming loop executes once and repeats **until** a

[1] Pseudocode is a plain-language description of the steps of a computer program.

counting variable (**M** or **P**, which is initially set to one) equals the *number* of times the loop should execute (**N**). A PHOSITA would recognize the recited loop as a do-until loop. As I explain in more detail later, a do-until loop executes once and repeats until a particular condition is true. This type of programming loop substantially predates the claims, and it is shown in Ford. The claims also use pointers or links to identify the virtual clips in a larger composition, but those aspects of the claim were also known in the art, as shown by Hedinsson. In the next two paragraphs, I walk through the requirements of claims 1 and 7 to explain my summary in more detail. As I noted above, the other two independent claims (claims 13 and 19) are substantially similar for purposes of this introduction.

28. Element [1a] is the preamble and it recites a "method to make a composite virtual clip." In elements [1b] and [1c], the claim "selects" or "sets" the values of two variables—one of these variables (N) is used to establish the *number* of times a loop should execute, and the other variable (M) is used to *count* the loop's iterations. Element [1d] occurs within the loop and it identifies virtual clips with corresponding pointers or links. Specifically, in element [1d], "a (M)th pointer" (e.g., a first pointer during the first loop, a second pointer during the second loop) that "indicate[s] a storage location for a (M)th saved virtual clip" is configured and saved. Finally, in element [1e], the claim increments the counting variable (M) and

specifies that the loop repeats until the counting variable (M) equals the desired number of iterations (N).

29. Turning to claim 7, element [7a] is the preamble and it recites a "method to display a composite virtual clip." In elements [7b] and [7c], the claim again recites two variables—one for the *number* of times a loop should execute (N), and the other to *count* the loop's iterations (P)—and in element [7e], it also increments the counting variable (P) and specifies that the loop repeats until the counting variable (P) equals the desired number of iterations (N). Although claim 7 uses a different name for its counting variable, a PHOSITA would have understood that any variable name could have been used to achieve the same functionality. Also, similar to claim 1, element [7d] occurs within the loop and it identifies virtual clips with corresponding links. Specifically, consistent with element [7a], element [7d] recites "activating a (P)th link … to stream a (P)th saved virtual clip to a user's device."

30. The Specification includes flow charts relevant to the challenged claims. (Ex-1001, 11:1-14:29, Figs. 6-13). In particular, Figure 9 includes a flowchart that mirrors claim 1, and Figure 10 includes a flowchart that mirrors claim 7. However, the Specification's explanation of these flowcharts largely mirrors the language found in the claims. (Ex-1001, 12:31-13:9). I have reviewed the Specification in its entirety.

31. The application leading to the ’130 patent (U.S. Appl. No. 16/134,541) (“the ’541 application”) was filed on September 18, 2018. The ’541 application is identified as a division of U.S. Appl. No. 15/269,879, which was filed on September 19, 2016 and issued as U.S. Patent No. 10,084,840, which is identified as a continuation-in-part of an application filed on January 31, 2014, which subsequently issued as U.S. Patent No. 9,451,001. The ’541 application also claims the benefit of provisional application No. 61/759,219, which was filed on January 31, 2013.

32. I have also reviewed the file history for the ’103 patent as well as the file histories for the other applications in its family, including the provisional application. There were no substantive rejections of the ’103 patent’s claims during prosecution. Indeed, only one office action was mailed during prosecution of the ’541 application. (Ex-1007, 63-69). In it, the Examiner identified a “pertinent” prior art reference that “discloses filling two composite presentation part data structures with pointers or references to media clips assigned to [them]”; however, the Examiner did “not rel[y] upon” this reference to reject the claims, and he did not explain how the claims distinguished over it. (Ex-1007, 69). After the applicants made minor clarifying revisions to the Specification and the claims (Ex-1007, 41-59), the Examiner allowed all claims (Ex-1007, 18-24). In the Reasons for Allowance, the Examiner simply stated that “the prior art fails to teach or reasonably suggest the invention as claimed.” (Ex-1007, 23).

VI. RELEVANT TIMEFRAME FOR DETERMINING OBVIOUSNESS

33. I understand that obviousness must be evaluated at the time the alleged invention was made. From the first page of the ’103 patent (left column, field “(60)”), I understand that the first patent application cited by the ’103 patent is a provisional application filed on January 31, 2013. I will assume for the purpose of this report that January 31, 2013, is the date the purported invention was made, with the understanding that my testimony will also be applicable to some time prior to January 31, 2013, approximately in the 2012 to early 2013 timeframe. I may refer to this time period as the “relevant time frame”, and my testimony concerning obviousness is directed to this timeframe, even if I occasionally do not explicitly use a past tense.

34. For clarity, I note that the testimony I have provided in this report would remain the same even if I had assumed that the invention was made on September 19, 2016, and even if I had referred to late 2015 to 2016 as the “relevant time frame.” In my view, the ’103 patent is not entitled to a priority date before September 19, 2016, the filing date of U.S. Application No. 15/269,879. Although the face of the patent lists other earlier applications, none of those earlier-filed priority applications describe the subject matter recited in any of the claims of the ’103 patent. It is my understanding that the priority date of the ’103 patent does not need to be resolved in this proceeding because each of the references that form

the bases of the grounds (Hedinsson, Ford, and Kostello) were all filed before the earliest possible priority date for this patent (i.e., January 31, 2013).

VII. LEVEL OF ORDINARY SKILL IN THE ART

35. I understand that the legal question of obviousness must be assessed from the perspective of a person having ordinary skill in the art ("PHOSITA") at the time of the invention. In my opinion, a PHOSITA in the relevant timeframe would have had an undergraduate degree in electrical engineering, computer engineering, or computer science and two years of computer programming experience and experience with web-based digital media. A person may also have substantially the same knowledge, understanding, and skills if he or she had less practical experience but more education, or less or different education but more practical experience.

VIII. DETAILED EXPLANATION OF THE REASONS FOR UNPATENTABILITY

Ground 1. Claims 1-10 and 13-22 are obvious over Hedinsson in view of Ford

36. In my opinion, claims 1-10 and 13-22 would have been obvious over U.S. Patent No. 8,793,282 B2 ("**Hedinsson**") (Ex-1004) in view of "Programming for the Absolute Beginner" by Jerry Lee Ford, Jr. ("**Ford**") (Ex-1005).

37. Ford provides a high-level and simple introduction to programming. It is applicable to many programming languages, and it evidences concepts that would have been known to a PHOSITA. At the time, if a PHOSITA was looking for a

simple reference guide to implement basic programming logic, he would have searched the library's database for books with titles that include terms such as "programming" or "coding" as well as "beginner," "introduction," or the like. Such a search would have yielded Ford, which is entitled "Programming for the Absolute Beginner."

38. Neither Hedinsson nor Ford was cited during prosecution of any application leading to the '103 patent. (Ex-1001, code (56)).

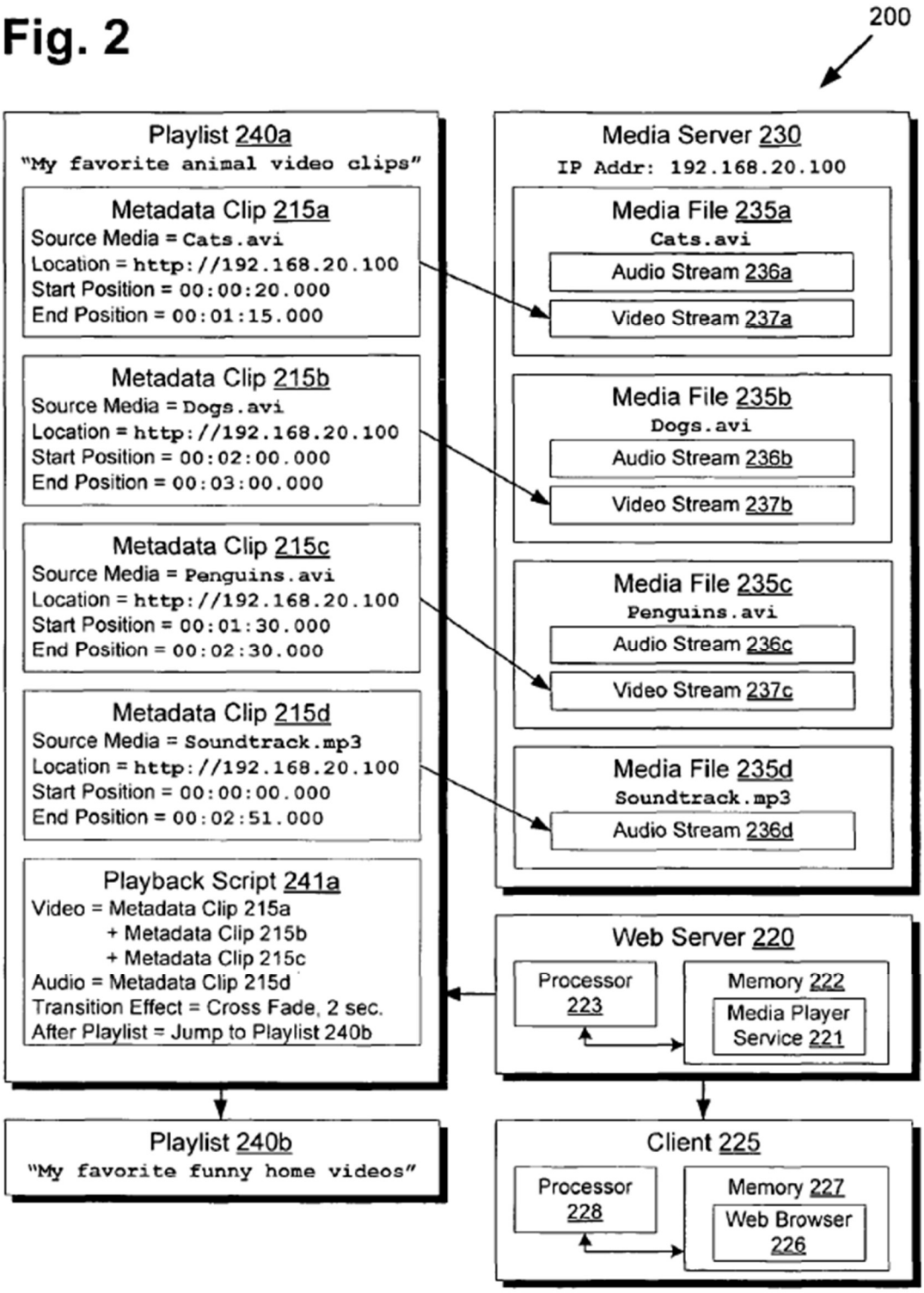
A. Overview of the Ground

39. In my opinion, the challenged claims would have been obvious over Hedinsson in view of Ford. Hedinsson describes a composite virtual clip that includes separate virtual clips (video clips) identified via pointers or links. Hedinsson's "media presentation" includes a plurality of video clips that are identified by corresponding "metadata clips." A PHOSITA would have been motivated to implement Hedinsson using well-known programming concepts, including programming loops, which are described in Ford. Specifically, a PHOSITA would have been motivated to make or display Hedinsson's composite virtual clip by stepping through each of the video clips using a do-until loop, and the resulting combination renders obvious the challenged claims.

1. Overview of Hedinsson

40. Hedinsson streams selected portions of different source media files as "one continuous sequential media stream" using a "playlist" that contains "metadata clips" that each point to an excerpt of a source media file (an audio or video clip). (Ex-1004, 5:37-47, Abstract). Each metadata clip "only provide[s] a reference to the actual media file[]," which allows Hedinsson's system to combine media file segments without generating a new media file. (Ex-1004, 4:11-14, 4:19-31, 4:59-67). Figure 2 illustrates the system.

Fig. 2

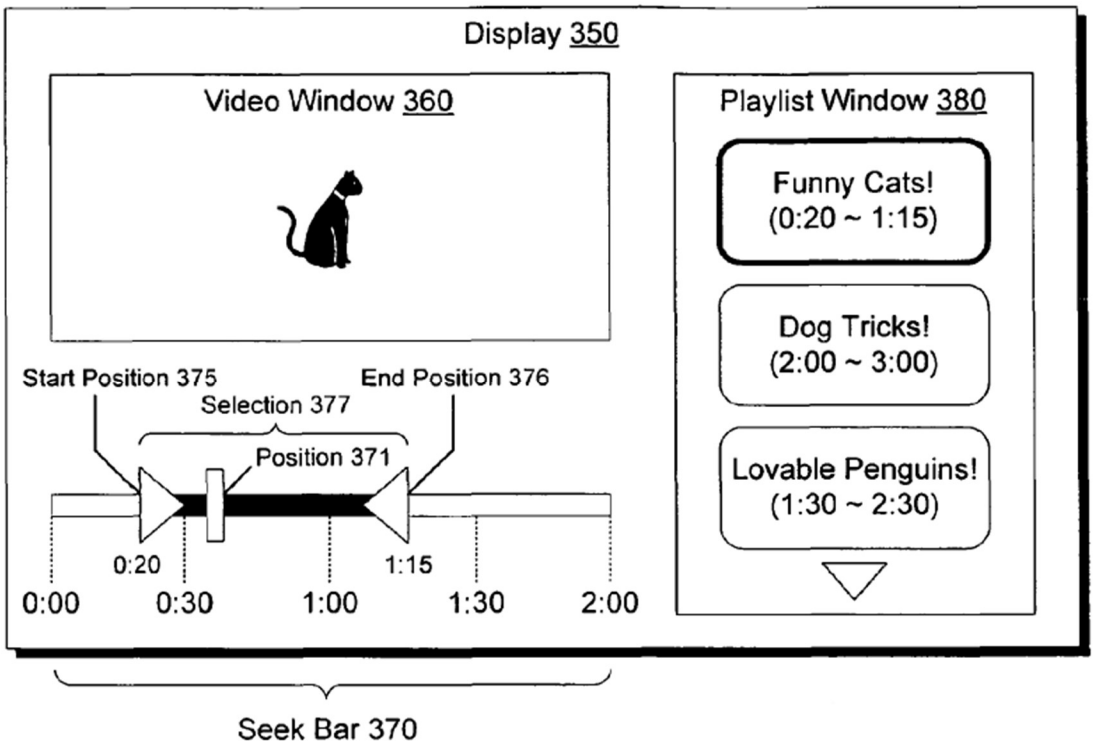


41. Hedinsson's metadata clip points to an audio or video clip.

Hedinsson's metadata clip identifies a specific excerpt of a media stream using: (1)

a filename, (2) "location information for retrieving the actual referenced source media file from a network, in the form of a uniform resource locator (URL) pointing to media server 130a via an Internet Protocol (IP) address," and (3) a start position and an end position. (Ex-1004, 4:33-42, 4:49-55, Fig. 2). In Figure 2 (above), Hedinsson's exemplary playlist (playlist 240a, top left) includes four metadata clips: one (215d) that points to an audio clip (in audio stream 236d) and three (215a-c) that point to video clips (in video streams 237a-c). (Ex-1004, 5:31-32, Fig. 2). In operation, Hedinsson uses playlist 240a to sequentially stream "the referenced video from metadata clips 215a, 215b, and 215c, in order, accompanied by the referenced audio from metadata clip 215d" to client 225 for playback on display 350. (Ex-1004, 5:37-47, 9:37-67).

42. An example of such playback is shown in Figure 3, which includes an exemplary "simplified player interface" (shown below) for displaying the presentation. (Ex-1004, 7:39-41).



Video window 360 (top left) displays a video, and playlist window 380 (right) shows "a sequential listing of metadata clips contained in the currently playing playlist, corresponding to playlist 240a of FIG. 2." (Ex-1004, 7:41-50, Figs. 2-3).

43. Hedinsson explains that "the player interface depicted in display 350" can support both "playlist playback" and "the creation and editing of playlists and metadata clips." (Ex-1004, 8:14-19). Hedinsson specifically states that a user can: (1) create a playlist using new or existing clips, (2) add a clip to a playlist, (3) adjust the start and/or end positions of a clip in a playlist, and (4) remove a clip from a playlist. (Ex-1004, 8:63-9:34, 8:34-45, Abstract). Hedinsson also contemplates a user locating and selecting a playlist created by another user. (Ex-1004, 9:32-34).

2. Overview of Ford

44. Ford describes "fundamentals of computer programming" that are applicable to "all programming languages." (Ex-1005, 11).[2]

45. Ford describes "loops," which "are code blocks that repeat a series of programming statements over and over again." (Ex-1005, 18). As Ford notes, there are several known types of loops, including "do…until" loops. (Ex-1005, 22-23). A do-until loop "executes at least once and repeats until a specified condition is true." (Ex-1005, 22-23). Ford provides an example:

```
i = 1

do

  print i
  i = i + 1

loop until i > 10
```

(Ex-1005, 26). This code prints the numbers 1 to 10. (Ex-1005, 21-22, 26). The variable "i" is used to track the number of loops; each iteration of the loop prints the value of "i"; and the loop repeats until "i" is greater than 10. (Ex-1005, 25-26). Ford

[2] For this exhibit, I refer to the page of the exhibit, shown in the bottom center of the page, rather than Ford's native pagination.

explains that "i > 10" is the loop's *condition*, and its *statements* are "print i" and "i = i + 1." (Ex-1005, 25). Also, Ford explains that, by changing "10" to a different number, the loop can repeat any number of times "without adding a single line of code." (Ex-1005, 22, 26).

B. Rationale (Motivation) Supporting Obviousness

46. As I explained in my summary of Hedinsson in ¶¶40-43, Hedinsson makes and displays a presentation that includes selected excerpts of audio and video streams (i.e., audio and video clips) using a playlist that includes "metadata clips" that each point to a corresponding audio or video clip.[3] (Ex-1004, 5:37-39, 8:16-19, Figs. 2-3). Hedinsson shows an example playlist (playlist 240a) in Figure 2; it describes displaying a presentation using that playlist (Ex-1004, 5:37-47, 9:25-67, Fig. 3); and it also states that a user can create, edit, and/or share playlists (Ex-1004, 8:16-19, 8:63-9:24, 8:34-51, 9:32-34). Hedinsson does not include pseudocode or code. As a result, it does not specify a precise manner of operation for either creating a new playlist or streaming using an existing playlist. These implementation details, however, were well within a PHOSITA's skill.

[3] I use the term "metadata clip" to refer to the data structure in Hedinsson's playlist, and the terms "audio clip" and "video clip" to refer to an excerpt of an audio or video media stream.

Indeed, as Hedinsson correctly states, "[t]he specific details not described in the present application are within the knowledge of a person of ordinary skill in the art." (Ex-1004, 2:26-28). Hedinsson also recognizes that Figure 3 is a "simplified … example" of an interface and notes that additional elements might be added to support "other editing and creation facilities" (Ex-1004, 7:39, 8:45-51), and that Figure 4 omits "details and features … that are apparent to a person of ordinary skill in the art" (Ex-1004, 8:55-57).

47. A PHOSITA would have been motivated to implement Hedinsson using code. Specifically, a PHOSITA would have been motivated to use programming loops both: (1) to create a new playlist that points to each of the desired video clips and (2) to stream each of the video clips identified by an existing playlist. Loops repeat a series of programming statements, and they were well-known to PHOSITAs. A PHOSITA would have been motivated to use a loop to create a new playlist or to display an existing one because each would both require performance of the same steps for each item in the playlist (i.e., for each metadata clip). In other words, when creating a new playlist that points to a set of video clips, some code will need to be executed to add a particular video clip to the playlist, and that code will be repeated for each of the video clips to be added. Likewise, when using a playlist to stream video clips, some code will need to be executed to stream a particular video clip, and that code will be repeated for each

of the video clips to be streamed. Ford demonstrates that loops and their benefits were well-known. Ford describes "fundamentals of computer programming" that are applicable to "all programming languages." (Ex-1005, 11). It explains that programmers use loops "to process the contents of arrays and text files" and "when processing files by extracting records a line at a time." (Ex-1005, 18, 22). Ford also discusses the benefits of loops. "Without loops, programmers would have to write enormous programs in order to be able to process large amounts of data." (Ex-1005, 21). "Likewise, without loops, performing repetitive actions, such as processing a collection of records stored in a file, becomes a tedious chore." (Ex-1005, 21). "Without loops, the programmer's job would be a lot harder." (Ex-1005, 22). "Obviously, using fewer programming statements to get something done makes for more streamlined and easier to support applications." (Ex-1005, 22).

48. Also, a PHOSITA would have been motivated to use loops to implement Hedinsson because this would facilitate playlists with different numbers of metadata clips, without having to create different blocks of code to handle playlists of differing lengths. Although Hedinsson's exemplary playlist (playlist 240a) references three video clips, a PHOSITA would have understood Hedinsson's description to be applicable to playlists that reference a different number of video clips. Indeed, Hedinsson indicates that its playlist can reference

more or fewer video clips. (Ex-1004, 3:1-6, 9:1-13, 9:25-29, 9:67-10:2). A PHOSITA would have understood that a loop can easily accommodate different numbers of clips with the same code. In particular, although Ford uses a particular number (i.e., 10) to configure the loop to iterate a predetermined number of times (i.e., ten times) (*e.g.*, Ex-1005, 22, 26), a PHOSITA would have understood that replacing this number with a variable, which can be set at run-time, would allow the number of iterations to be dynamically determined. This would have been beneficial for implementing Hedinsson so that its playlists can include differing numbers of metadata clips. In addition, when using a loop to add or access a set of metadata clips, a PHOSITA would have been motivated to add or access one metadata clip during each iteration of the loop because this would reduce repetitive code and increase the ability to accommodate different numbers of clips. If the loop instead added or accessed two metadata clips, for example, then the code would be longer, and the metadata clips would need to be added or accessed in pairs (or additional code would need to be added to allow the loop to sometimes add or access only one metadata clip).

49. At the time, there were different types of programming loops, and a PHOSITA would have been aware of (and would have been able to use) each of these types. Ford describes four basic types: a do-while loop, a do-until loop, a for-next loop, and a while-wend loop. (Ex-1005, 22-23). As Ford notes, most

programming languages included these four basic types of loops, even though the specific syntax used for a particular type varied from programming language to programming language. (Ex-1005, 23). A PHOSITA would have known how to use each of these four basic types of loops, and a PHOSITA would have found the implementation of each to be predictable. Programming languages and coding techniques are predictable by design because a programmer needs to know what his or her code will do. Also, it would have been trivial for a PHOSITA to use any of these loops to write code. Thus, a PHOSITA would have understood these four types of code loops to be a finite number of identified, predictable solutions to the problem of writing code that repeats programming statements. Implementing each type of loop was within a PHOSITA's skill, and a PHOSITA would have found each type of loop to be predictable.

50. A PHOSITA would have been motivated to use a do-until loop when implementing Hedinsson because a do-until loop is executed at least once and repeats until a specified condition is true. (Ex-1005, 23). Specifically, a PHOSITA would have been motivated to create a new playlist using a do-until loop that: (1) adds one metadata clip (that points to a video clip), and (2) repeats until metadata clips associated with each desired video clip have been added. Also, a PHOSITA would have been motivated to stream Hedinsson's presentation using a do-until loop that: (1) accesses a metadata clip (to stream the linked video

clip), and (2) repeats until all video clips have been streamed. A PHOSITA would have recognized the benefit of using do-until loops for these operations because Hedinsson's playlist includes at least one metadata clip and looping should continue until all metadata clips have been processed.

C. Reasonable Expectation of Success

51. A PHOSITA would have had a reasonable expectation of success in making the proposed combination.

52. The art in the relevant timeframe was predictable. Specifically, in the relevant timeframe, the web server described by Hedinsson and the basic programming concepts explained in Ford were well-known in the art. Indeed, programming loops and their types were fundamental concepts taught in introductory computer programming books and classes. This is shown by Ford. I note that the '103 patent provides only a general and high-level discussion of the claimed subject matter, with no elaboration beyond the requirements recited in the claims.

D. Graham Factors

53. I addressed the level of ordinary skill in the art in ¶35.

54. The scope and content of the prior art are discussed throughout my declaration.

55. The differences between the prior art and the claims are discussed throughout my declaration.

56. I am not aware of any secondary considerations that would make an inference of non-obviousness more likely.

57. This discussion of Graham factors applies to all Grounds.

E. Analogous Art

58. In my view, both Hedinsson and Ford are analogous art that are in the same field as the ’103 patent and/or reasonably pertinent to the particular problems facing the inventors. Hedinsson is in the same field as the ’103 patent (“managing information … and more particularly … sharing and annotating video data”), and it would have been reasonably pertinent to the video-sharing problems facing the named inventors. (Ex-1004, Title, Abstract, 1:6-53; Ex-1001, Title, Abstract, 1:18-22). Ford describes basic computer programming concepts, which were known to and used by PHOSITAs. (Ex-1005, 11).

F. Claim Mapping

59. This section maps the challenged claims to the relevant disclosures of Hedinsson and Ford, where the claim text appears in bold-italics, and the relevant mapping follows.

CLAIM 1

"1[a]. A method to make a composite virtual clip, comprising:"

60. It is my understanding that a claim's preamble may or may not be limiting, but in any event, Hedinsson discloses **a method to make a composite virtual clip**. As I explain in more detail in the following paragraphs, Hedinsson **makes** a "sequential media stream" of audio and video clips from different source media files by creating a new playlist. (Ex-1004, 5:37-47, 8:16-19, 9:37-67, Abstract, Figs. 2, 4). This media stream and/or its playlist are **a composite virtual clip**.

61. Hedinsson discloses that a user may create a new playlist. For example, in its Abstract, Hedinsson states that "metadata clips and playlists may … be user generated." (Ex-1004, Abstract). It also states that "the player interface depicted in display 350" can support "the creation and editing of playlists" "via user input read from input device 329." (Ex-1004, 8:14-24). Hedinsson provides an example where clips are either added to or removed from a playlist by "click[ing] and dragg[ing]" an object corresponding to the clip either into the playlist or from the playlist into a trashcan. (Ex-1004, 8:40-45). Hedinsson explains that a user can "assemble playlist 340" by creating new metadata clips or adding existing ones. (Ex-1004, 9:1-24). Also, Hedinsson states that a user may save a playlist for later access by others. (Ex-1004, 9:32-34). Finally, in the

flowchart shown in Figure 4, Hedinsson generates one or more metadata clips (in step 410) and then "determine[s] a playlist" (in step 420). (Ex-1004, 9:25-36). From at least these passages of Hedinsson, a PHOSITA would understand Hedinsson to disclose creation of a new playlist by a user.

62. Hedinsson's playlist includes multiple metadata clips that each point to an audio or video clip. (Ex-1004, Fig. 2, 4:11-58, 5:31-47, 9:37-53). For example, playlist 240a includes one metadata clip (215d) pointing to an audio clip and three metadata clips (215a-c) pointing to video clips. (Ex-1004, Fig. 2). Also, Hedinsson teaches that a user may create a new playlist that includes multiple metadata clips. (Ex-1004, 8:63-9:24). In particular, in this passage Hedinsson refers to adding metadata clips (plural) to a playlist. For example, Hedinsson states that "[a] user at client 325 might search various media files from media server 330 to assemble playlist 340, providing start position 375 and end position 376 for metadata clips referencing those media files." (Ex-1004, 9:1-4). It also states that a user can set the start and end positions "for each media file." (Ex-1004, 9:7-9). Hedinsson says that client 325 notifies media player service 321 (executing in web server 320) "to store the newly created metadata clips in playlist 340." (Ex-1004, 9:10-13). Finally, Hedinsson states that "existing metadata clips may also be added to playlist 340." (Ex-1004, 9:14-16).

"[1b] selecting (N) saved virtual clips and an order of presentation for each virtual clip;"

63. In my opinion, Hedinsson expressly or inherently teaches **selecting** a particular number of saved video clips (**(N) saved virtual clips**) for inclusion in the new playlist. As I explained above in connection with element [1a], Hedinsson discloses that a user may create a new playlist that includes multiple metadata clips that each point to a video clip. From those passages, a PHOSITA would have understood Hedinsson to state that the user (and the system) selects the video clips for this new playlist. For example, Hedinsson's Abstract says that "metadata clips and playlists may … be user generated." (Ex-1004, Abstract). Also, Hedinsson states that "[a] user at client 325 might search various media files from media server 330 to assemble playlist 340, providing start position375 and end position 376 for metadata clips referencing those media files," and then "client 325 can notify media player service 321 … to store the newly created metadata clips in playlist 340." (Ex-1004, 8:63-9:24). In addition, a PHOSITA would have understood Hedinsson to state that the system also selects the video clips for this new playlist, at least because it adds existing or new metadata clips to a playlist. (Ex-1004, 9:14-16, 9:22-24). In addition, in my opinion, **selecting** a particular number of saved video clips (**(N) saved virtual clips**) for inclusion in the new playlist must necessarily be present given Hedinsson's disclosure. In order for a

user to create a new playlist of video clips, the video clips must necessarily first be **selected** by the user and/or by the system.

64. In either event, the number of selected video clips teaches **(N)**, which can be any number. For the creation of playlist 240a, **(N)** would equal 3, and the **virtual clips** would be the video clips referenced by metadata clips 215a-c. (Ex-1004, 5:37-47, 9:44-53, Fig. 2). Also, Hedinsson's video clips are **saved** on media server 230. (Ex-1004, 5:26-31, 2:46-50, Figs. 1-2).

65. In my opinion, Hedinsson expressly or inherently teaches **selecting an order of presentation for each virtual clip**. As I explained in element [1a], Hedinsson discloses that a user may create a new playlist to display multiple video clips. Also, Hedinsson's playlist specifies the order in which a plurality of video clips should be played: Hedinsson uses playlist 240a to stream "the referenced video from metadata clips 215a, 215b, and 215c, in order." (Ex-1004, 5:41-47, 9:37-53). From those passages, a PHOSITA would have understood Hedinsson to state that the system selects the order in which the video clips should be played. Also, in order to create a playlist that specifies an order in which video clips should be played, the system must necessarily first select the clips' order of presentation, which necessarily involves selecting the order of presentation for each video clip.

66. In the alternative, element [1b] would have been obvious over Hedinsson. The above-referenced discussion of creating a playlist at least suggests

selecting the video clips **and an order of presentation for each** video clip. In addition, a PHOSITA would have been motivated to implement Hedinsson's system so that a user can **select** the video clips **and** their **order of presentation** when creating a new playlist, and the PHOSITA would have had a reasonable expectation of success. A PHOSITA would have been motivated by Hedinsson's disclosure because Hedinsson seeks to provide users with the ability to "participate in the content creation process," "tailor[] content," and "[r]emix[] and juxtapos[e] clips." (Ex-1004, 1:32-39). Also, a PHOSITA would have been motivated to add this feature to increase user-friendliness, as a PHOSITA would have recognized that users would prefer (and in some cases would expect) to be able to select the video clips and their order when creating a new playlist.

"[1c] initially setting (M)=1;"

67. Ford describes an exemplary do-until loop that begins with "i = 1." (Ex-1005, 26). This teaches **initially setting (M)=1**, where **(M)** is taught by Ford's "i." Both "i" and "(M)" are arbitrary names for the variable that is used to count the number of iterations of the loop (also known as a "counter"), and the selection of the name of this variable has no impact on the function of the code. As I explained above in the section entitled the "Rationale Supporting Obviousness," and as I address in more detail in elements [1d] and [1e], a PHOSITA would have been motivated to create Hedinsson's new playlist using

Ford's do-until loop, and would have had a reasonable expectation of success in doing so. In particular, a PHOSITA implementing Hedinsson in this way would have been motivated to initially set its counting variable to "1," as described in Ford, so that the counting variable is equal to the current iteration number of the loop. (Ex-1005, 26). Although computer programmers frequently began counting with zero instead of than one, this variation would have been trivial to a PHOSITA. In particular, a PHOSITA would have been able to configure a loop by initializing this variable to either "0" or "1," and the PHOSITA could have easily modified the loop's code to switch from either of these options to the other one.

"[1d] configuring a (M)th pointer to indicate a storage location for a (M)th saved virtual clip, wherein a (M)th virtual clip comprises a (M)th content from one or more designated sources and (M) is greater than or equal to 1 and less than or equal to (N); saving said (M)th pointer;"

68. Hedinsson teaches that each metadata clip includes **a pointer to indicate a storage location** of a corresponding video clip (**saved virtual clip**) or audio clip. (Ex-1004, 4:33-64, Fig. 2). Hedinsson's metadata clip includes (1) "information for media player service 121 to locate the actual referenced source media file," (2) "information for retrieving the actual referenced source media file from a network, in the form of a uniform resource locator (URL) ***pointing to*** media server 130a via an Internet Protocol (IP) address," and (3) "a referenced start position … and a referenced end position" identifying the relevant "portion of [the]

media file." (Ex-1004, 4:33-64 (emphasis added), Fig. 2). The **pointer** is taught by: (1) the URL (which includes an IP address in Hedinsson's example) and source media filename, or (2) the URL, source media filename, and the start and end positions. In context, a PHOSITA would have understood the term "pointer" to mean that the system uses the *location* of the information, rather than the information itself. This understanding is also consistent with the references that were cited during prosecution of the '103 patent. For example, US2012/0284343 states that "the pointer to the multimedia content could be solely the web address or location of the video (i.e., without thumbnail and additional other information)" or that it could be an indirect reference that points to "intermediate content, which in turn leads to the full multimedia content." (Ex-1010, ¶47). Also, US2014/0229866 describes a "uniform resource locator ('URL')" as a type of "reference pointer." (Ex-1011, ¶107).

69. To the extent that Patent Owner argues that the term "pointer" should be interpreted to include additional requirements, Hedinsson still renders this obvious. The '103 patent uses the term "pointer" only once outside of the claims. (Ex-1001, 8:39-41). This supports my view that a PHOSITA would have been familiar with pointers and would have known how to use them. Indeed, in my opinion, a PHOSITA would have understood that a pointer was a known and predictable way to reference data, and a PHOSITA would have been able to

substitute Hedinsson's pointer (which identifies the location of a video clip within the media servers) with such a pointer with a reasonable expectation of success.

70. Hedinsson discloses or renders obvious **saving said pointer**. Hedinsson discloses that a user can create new metadata clips and "store the newly created metadata clips in playlist 340" (Ex-1004, 8:63-9:16, Fig. 4 (step 410)), and as I explained above, each metadata clip includes a **pointer**. Because Hedinsson stores new metadata clips in a new playlist, a PHOSITA would have understood that Hedinsson discloses **saving** the new metadata clips (and their **pointers**) in the playlist. In addition, PHOSITA would have been motivated to **save** the newly created metadata clips and the playlist because this would allow later use of those metadata clips and that playlist, as contemplated by Hedinsson. Hedinsson indicates that metadata clips are stored and can be accessed by other users. (Ex-1004, 9:14-18). Hedinsson describes "stor[ing] the newly created metadata clips in playlist 340," and it indicates that users can assemble and store playlists for later "browsing and selection by [a] user." (Ex-1004, 9:9-13, 9:32-34). Also, Hedinsson indicates that the playlist (which includes the metadata clips and their pointers) is saved because its "[e]nvironment 300 includes web server 320, client 325, input device 329, media server 330, playlist 340, and display 350." (Ex-1004, 7:15-18, Fig. 3).

71. Hedinsson discloses or renders obvious **configuring said pointer**. A PHOSITA would have understood that creating and saving a new metadata clip (with a **pointer**) involves **configuring** the **pointer**. In order for a new pointer to be saved, the appropriate data must first be added to it (i.e., the information specifying the destination referenced by the pointer). Some programming languages also require the pointer to be created and/or formatted before the data is added. These steps are generally referred to as configuring the pointer. The specific details of the configuration steps will depend upon the programming language used, but this would have been within the PHOSITA's skill. Also, a PHOSITA would have been motivated to implement the combination such that the **pointer** created for insertion into a new playlist was **configured** before being saved given Hedinsson's disclosure of saving metadata clips. In particular, as explained above, Hedinsson discloses that a metadata clip includes a pointer, and Hedinsson discloses saving the metadata clip. A PHOSITA would have been motivated to configure this pointer before saving it so that the system operates correctly.

72. Hedinsson discloses or renders obvious configuring and saving an **(M)th pointer** for an **(M)th saved virtual clip**. Initially, in the combination, **(M)** equals 1 (element [1c]), and for the reasons I provided earlier in my discussion of element [1d], Hedinsson at least discloses or renders obvious configuring and

saving a first pointer for a first saved video clip. In addition, to the extent required, Hedinsson discloses or renders obvious configuring and saving a pointer for *each* of the N video clips (element [1b]) because Hedinsson seeks to create a new playlist that includes a set of new and/or existing metadata clips. Hedinsson's disclosure encompasses a situation, for example, where a user seeks to create a new playlist that includes two new video clips. (As I noted in element [1b], N can be any number.) For the reasons previously explained, a PHOSITA would have understood Hedinsson to disclose configuring and saving a first pointer for the first video clip and a second pointer for the second video clip, or alternatively, a PHOSITA would have been motivated to implement Hedinsson in this way.

73. Also, Hedinsson teaches that each video clip (**(M)th virtual clip**) includes (**comprises**) corresponding content (**(M)th content**) **from** a particular media file 235 in a particular media server 230 (**one or more designated sources**). (Ex-1004, 9:37-67, 4:11-22, 5:26-36, 5:58-67, Fig. 2). In addition, in the combination, **(M) is greater than or equal to 1 and less than or equal to (N)** because, as I discuss in elements [1b] and [1e], M initially equals 1, and M is incremented until N video clips are added to the new playlist.

"[1e] determining if (M) equals (N); when (M) is less than (N), setting (M) equal to (M)+1; and repeating said configuring, saving, and determining until (M) equals (N)."

74. As explained in elements [1a], [1b], and [1d], Hedinsson discloses or renders obvious creating a new playlist that includes N metadata clips that each point to a corresponding video clip. (*E.g.*, Ex-1004, Abstract, 8:16-19, 8:63-9:24). Also, as I previously explained in the section entitled "Rationale Supporting Obviousness," a PHOSITA would have been motivated to create this new playlist using a do-until loop, as taught by Ford. In particular, a PHOSITA would have been motivated to use a do-until loop that configures and saves one metadata clip (including its pointer) during each iteration and repeats until all N metadata clips have been configured and saved, so that the playlist includes pointers to each of the N selected video clips (from element [1b]).

75. The language included in element [1e] is unclear because the last phrase ("repeating…") is unclear in view of the prior two phrases ("determining …" and "setting …"). To illustrate the problem, I will use an example where N equals 2 (i.e., 2 saved virtual clips and an order of presentation are selected in element [1b]). M is initially set to 1 (element [1c]), and a first pointer indicating a storage location of a first virtual clip is configured and saved (element [1d]). In element [1e], the claim recites "determining if (M) equals (N)," and the method determines they are not equal (because 1 does not equal 2). Next, the claim recites

"when (M) is less than (N), setting (M) equal to (M)+1." Here, 2 is less than 1, so the method sets M equal to M+1 (i.e., 1+1), which is 2. In other words, after "setting…," M equals 2. Finally, the last phrase recites "repeating … until (M) equals (N)," but it is unclear *when* the method evaluates "(M) equals (N)." If this phrase refers to the prior evaluation (in the "determining…" step), then the *method will repeat* (because it was previously determined that 1 does not equal 2). However, if this should be evaluated at the time of the "repeating…" step, then the *method will not repeat* (because it will determine that 2 does in fact equal 2). As a result, it is unclear whether the claimed loop iterates N times or N-1 times. However, in my view, Hedinsson-Ford renders the claim obvious under either interpretation. I explain in detail why it would have been obvious to iterate the loop N times in the proposed Hedinsson-Ford combination. But, if the claim only requires N-1 iterations, then the first N-1 iterations of the Hedinsson-Ford loop render this claim obvious (because the claim does not prohibit additional iterations of the loop).

76. This proposed Hedinsson-Ford combination discloses or renders obvious **determining if (M) equals (N)**. Ford's do-until loop "evaluate[s]" whether a *condition* is true (**determining if**) (Ex-1005, 25-26), and in the proposed Hedinsson-Ford combination, a PHOSITA would have been motivated to use **(M) equals (N)** as the *condition*. Ford initially sets "i" to 1, increments "i" in the loop,

and repeats the loop until "i" exceeds a particular value. (Ex-1005, 26). As explained in element [1c], Ford's "i" teaches **(M)** —both "i" and "(M)" are variables used to count the number of loop iterations. A PHOSITA would have known that loops commonly include a variable that is used to count the number of loops, and another example of that is shown in the introductory example provided by Ford. (Ex-1005, 21-22). Based on this teaching, a PHOSITA would have been motivated to use, as the loop's condition, a comparison of M with a number indicative of the number of times the loop should repeat. In other words, a PHOSITA would have been motivated to use **(M) equals (N)** as the condition because M is used to count the number of loops, M begins at 1, and the loop should repeat N times.

77. Ford teaches **setting (M) equal to (M)+1** because Ford increments "i" in the loop (i.e., "i = i + 1"). (Ex-1005, 26). Also, the combination discloses or renders obvious setting (M) equal to (M)+1 **when (M) is less than (N)**. Because the loop increments the value of M during *each* iteration, M will be incremented whether or not this condition is satisfied (and, thus, will be incremented when this condition is satisfied). Moreover, if required by the claim, a PHOSITA would also have been motivated to increment M *only when* M is less than N because this would avoid an unnecessary step (i.e., incrementing M even after the loop has been executed the desired number of times). In other words, when M equals N there is

no need to further increment M because the desired number of loop iterations has already occurred.

78. The Hedinsson-Ford combination discloses or renders obvious **repeating said configuring, saving, and determining until (M) equals (N)**. Ford's "*statements* … are executed each time the loop repeats," and the "*condition* … is evaluated at the end of each iteration of the [do-until] loop." (Ex-1005, 26). The do-until loop "repeats until [the] specified condition is true." (Ex-1005, 23). Thus, Ford teaches **repeating** both (1) the *statements* inside the loop and (2) the **determining** of the *condition*, **until** the *condition* is true. As I explained above, a PHOSITA would have been motivated to use **(M) equals (N)** as the *condition*. Also, a PHOSITA would have been motivated to include the **configuring** and **saving** steps (i.e., element [1d]) in the loop's *statements* so that each metadata clip (and its pointer) is configured and saved in the playlist. In particular, as explained in element [1d], a pointer corresponding to a given metadata clip is **configured** and **saved** when adding the metadata clip to the playlist, and a PHOSITA would have been motivated to repeat this for each metadata clip to be added to the playlist so that the playlist includes all of the desired information. A PHOSITA would have understood that this information would need to be added to the newly created playlist in order for the playlist to be used to stream each of the video clips, as contemplated by Hedinsson. (Ex-1004, 5:37-47, 9:37-67).

79. Also, to the extent required by the claim, the combination discloses or renders obvious element [1d] when it is **repeated** pursuant to element [1e]. For the reasons I explained above in connection with element [1d], Hedinsson discloses or renders obvious performing the steps recited in element [1d] for *each* of the N video clips (**saved virtual clips**).

80. Finally, to the extent that the combination of elements [1d] and [1e] requires the pointers to be saved in sequential order, this is disclosed or rendered obvious by Hedinsson. Hedinsson discloses processing the video clips in sequential order and depicts playlist 240a in the same order. (Ex-1004, 9:37-67, Fig. 2). Although Hedinsson's playback script 241a would allow the video clips to be easily reordered after playlist creation (*see* Ex-1004, 5:37-47), a PHOSITA would have understood Hedinsson to create its playlist 240a with metadata clips 215 in the original sequential order. Also, this ordering would have been obvious over Hedinsson because a PHOSITA would have been motivated to sequentially store the metadata clips in the playlist when originally creating the playlist. Using this ordering would have been a very straightforward way to write the code, and it would have been one of a finite number of identified, predictable solutions to the problem of storing data that were well within the skill of a PHOSITA.

CLAIM 2

"2[a]. The method of claim 1, wherein said composite virtual clip further comprises one or more first type virtual clips, wherein the one or more first type virtual clips comprise content from a Media File, playable or static, from a beginning to a first designated end point,"

81. *See* claim 1. As I explained above in my discussion of element [1a], Hedinsson makes a **composite virtual clip** that **comprises** audio and video clips (either of which, in the alternative, teaches the **one or more first type virtual clips**). (*E.g.*, Ex-1004, Fig. 2). The audio and video clips each **comprise content from a Media File** (one of the media files 235a-d) **playable from a beginning** (the metadata clip's start position) **to a first designated end point** (the metadata clip's end position). (Ex-1004, 4:11-22, 4:52-58, 9:37-53, Figs. 1-2).

82. For the audio alternative: playlist 240a includes metadata clip 215d, which causes the system to play an audio clip—"Soundtrack.mp3" from "00:00:00" to "00:02:51." (Ex-1004, Fig. 2, 9:43-44, 5:46-47). For the video alternative: playlist 240a includes metadata clip 215a, which causes the system to play a video clip—"Cats.avi" from "00:00:20" to "00:01:15." (Ex-1004, Fig. 2, 9:44-48).

"[2b] wherein the Media File, playable or static, is selected from the group consisting of AVI file, MP3 file, MP4 file, WMA file, WAV file, FLASH, MPEG file, an image file, a PDF file, a text file, a VISIO file, a .ppt file, a .key file, a spreadsheet file, 3D media file

which provides a Stereoscopic visual display that can be viewed through stereoscopic eyewear or played on 3D display technology, and a Virtual Reality/Augmented Reality file that can be viewed through Virtual Reality devices."

83. Hedinsson discloses element [2b] under either alternative mapping of the **first type virtual clip** (*i.e.*, audio or video). Hedinsson's audio clip comprises content from an MP3 file (Ex-1004, 9:43-44, Fig. 2), and Hedinsson's video clip comprises content from an AVI file (Ex-1004, 9:46-48, Fig. 2).

CLAIM 3

84. I have reordered the elements of this claim to address the limitations pertaining to a **first interactable graphical object** (elements [3b] and [3d]) before addressing a **second interactable graphical object** (elements [3c] and [3e]).

"3[a]. The method of claim 2, further comprising:"

85. See my analysis of claim 2, which is provided above.

"[3b] displaying a first interactable graphical object, wherein the first interactable graphical object comprises a Cancel Clip link;"

"[3d] when activating the first interactable graphical object, a begin time location, end time location, and first media duration of the first type virtual clip are not stored; and"

86. As I discussed in connection with claim 2, the **first type virtual clip** is either Hedinsson's video clip or its audio clip. In the following paragraphs, I address each of these alternatives separately.

Video Alternative:

87. When the "first type virtual clip" is a video clip, Hedinsson discloses or renders elements [3b] and [3d] obvious for two reasons.

88. First, Hedinsson teaches that a user may edit playlists using display 350. (Ex-1004, 7:39-41, 8:16-19, Fig. 3). Specifically, "items from playlist window 380 might be clicked and dragged into a trashcan icon using input device 329 to remove items from playlist." (Ex-1004, 8:42-45). That trashcan icon is the **first interactable graphical object** that **comprises a Cancel Clip link**, and when it is **activated** (by a user dragging a video clip from playlist window 380 to the trashcan icon), the corresponding video clip is removed from the playlist. As a result, the clip's **begin time location, end time location, and first media duration are not stored**.

89. Second, it would have been obvious to add buttons to Hedinsson's display 350 that allow a user to "save" or "cancel" a particular clip, given Hedinsson's disclosure and the knowledge of a PHOSITA.

90. Hedinsson teaches that a user can "generat[e] … metadata clips" by "search[ing] various media files from media server" and "providing start position 375 and end position 376." (Ex-1004, 8:63-9:9, 8:34-40, Fig. 3). Hedinsson also teaches that "the newly created metadata clips" can be "store[d]" in the playlist, and Hedinsson indicates that those clips can be saved for future use by others.

(Ex-1004, 9:9-24). Hedinsson's "simplified player interface example" in Figure 3 does not show how a user would specify whether to save a new clip (or discard changes to a clip), but Hedinsson notes that "other editing and creation facilities" may be added. (Ex-1004, 7:39, 8:45-48).

91. Given Hedinsson's disclosure, a PHOSITA would have been motivated to add a save-clip button and a cancel-clip button to the display: the former would save a user's edits to a particular video clip (either by modifying an existing clip or creating a new one), and the former would discard any edits (and cancel the add-clip operation). A PHOSITA would have known that user interfaces frequently provided such buttons to allow a user to specify whether to save or discard changes. An example of such an interface is shown in US2015/0185965 (Ex-1012, "Belliveau"), which was filed before the '103 patent (and, in my view, was filed before its priority date, as I discuss above in ¶34). Both Hedinsson and this published application relate to creating video clips for inclusion in a compilation, and in fact, they both list Disney Enterprises, Inc. as the assignee. (Ex-1012, ¶¶3, 6, 39, 41). In connection with Figure 2, Belliveau provides an interface that can be used to create a video clips using clip scrubbers 20a, 20b that "are selectable by a user and determine the start point (start scrubber 20a) and end point (end scrubber 20b) of the clip that the user wishes to select." (Ex-1012, ¶31). In addition, Belliveau's interface shows a "Save" button and a

"Cancel" button, which a user would have understood as providing a way to indicate when/if the clip should be saved or the user's changes should be discarded. (Ex-1012, Fig. 2, ¶34). In addition, adding these buttons to a user interface was a predictable design choice within the skill of a PHOSITA, and a PHOSITA would have known that it would increase the user-friendliness of Hedinsson's display. In particular, by providing both of these buttons, a user could specify when to save a newly created metadata clip, and a user would have a clear way to specify when a newly created metadata clip (or changes to an existing clip) should not be saved.

92. Thus, Hedinsson renders obvious a **first interactable graphical object** (the cancel-clip button) that **comprises a Cancel Clip link** that, when **activated** by user input, does **not store** the **begin time location, end time location, and first media duration** of the video clip.

Audio Alternative:

93. Although not specifically shown in Figure 3, given Hedinsson's disclosure and the knowledge of a PHOSITA, it would have been obvious to implement Hedinsson so that audio clip(s) are created, added, changed, and/or removed using the same (or substantially the same) **interactable graphical objects** discussed in the video alternative for claim 3.

94. Hedinsson's disclosure suggests that the same (or substantially the same) **interactable graphical objects** should be used for both video and audio clips. Hedinsson uses an exemplary video clip to illustrate the functionality, but Hedinsson refers to creating, adding, changing, and removing a playlist's "metadata clips" generally, and some metadata clips (such as metadata clip 215d) point to audio clips. (Ex-1004, 8:16-51, 8:63-9:24, 5:46-47, Fig. 2). Also, Hedinsson indicates that a user can specify the audio clip to be included in a playlist. Hedinsson shows a variety of available audio streams in Figure 2, and Hedinsson's background references "replac[ing] audio tracks." (Ex-1004, 1:34-36, 5:26-31, Fig. 2 (audio streams 236a-d)). Hedinsson explains that audio clips are tagged to facilitate searching, and a clip's attributes might specify "audio sampling rate" or "number of audio channels." (Ex-1004, 3:34-38, 3:44-48). Also, Hedinsson states that "disparate video clips can be unified" "[b]y choosing a separate audio stream." (Ex-1004, 5:46-57). In fact, although Hedinsson's example playlist identifies only one audio clip ("Soundtrack.mp3"), Hedinsson states that "[m]ultiple audio tracks might be specified" in a playlist. (Ex-1004, 5:54-57). In addition, Hedinsson recognizes that its display 350 is a "simplified player interface example," and it suggests using additional "interface elements" to support "referencing outside audio sources." (Ex-1004, 7:39, 8:45-48).

95. Moreover, a PHOSITA would have been motivated to implement Hedinsson in a manner that allows a user to create, add, change, and remove audio clip(s) using the same (or substantially similar) graphical objects discussed in the video alternative to provide the user with the flexibility to define a playlist's audio clip. Also, using the same (or substantially similar) graphical objects for both the video and audio clips would increase user-friendliness because the user would interface with the system in substantially the same way when working with audio and video clips, thereby gaining additional familiarity with the system's operation.

96. As a result, when the "first type virtual clip" is an audio clip, Hedinsson renders elements [3b] and [3d] obvious for two reasons.

97. First, given Hedinsson's disclosure of dragging metadata clips to a trashcan icon to remove them from the playlist (Ex-1004, 8:42-45), it would have been obvious to use the same process to remove an audio clip from the playlist. That trashcan icon is the **first interactable graphical object** that **comprises a Cancel Clip link**, and when it is **activated** by dragging an audio clip to the trashcan, the **begin time location, end time location, and first media duration** of the corresponding audio clip are deleted from the playlist and **are not stored**.

98. Second, for the reasons I explained above with respect to video clips, Hedinsson's disclosure would have motivated a PHOSITA to add buttons that allow a user to either "save" or "cancel" a new audio clip. Thus, Hedinsson

renders obvious a **first interactable graphical object** (cancel-clip button) that **comprises a Cancel Clip link** that, when **activated** by user input, does **not store** the **begin time location, end time location, and first media duration** of the audio clip.

> ***"[3c] simultaneously displaying a second interactable graphical object, wherein the second interactable graphical object comprises an End Clip link;"***
>
> ***"[3e] when activating the second interactable graphical object, the begin time location, end time location, and first media during of the first type virtual clip are stored."***

Video Alternative:

99. When the "first type virtual clip" is a video clip, Hedinsson teaches or renders elements [3c] and [3e] obvious for two reasons.

100. First, Hedinsson teaches that a user can add a clip to a playlist using display 350. (Ex-1004, 7:39-41, 8:16-19, Fig. 3). "Video window 360 might be clicked and dragged into playlist window 380 using input device 329 to add the presently playing metadata clip to playlist 340." (Ex-1004, 8:40-42). The playlist window 380 is the **second interactable graphical object** that **comprises an End Clip link**; when it is **activated** (by a user dragging a video clip from video window 360 to playlist window 380), a metadata clip associated with the video clip is

stored in the playlist. As a result, the video clip's **begin time location** and **end time location are stored** in the playlist.

101. Second, as I explained above in connection with the second video alternative for elements [3b] and [3d] (starting at ¶89), a PHOSITA would have been motivated to implement Hedinsson's display 350 to include save-clip and cancel-clip buttons. For those reasons, Hedinsson renders obvious a **second interactable graphical object** (the save-clip button) that, **when activated** by user input, **stores** the **begin time location** and the **end time location** of the video clip.

Audio Alternative:

102. When the "second type virtual clip" is an audio clip, Hedinsson renders these elements obvious because, as I explained in connection with the audio alternative for elements [3b] and [3d] (starting at ¶93), a PHOSITA would have been motivated to implement Hedinsson so that audio clip(s) are created, added, changed, and/or removed using the same (or substantially the same) **interactable graphical objects** discussed in the video alternative. As a result, when the "second type virtual clip" is an audio clip, Hedinsson renders elements [3c] and [3e] obvious for two reasons.

103. First, given Hedinsson's disclosure of dragging metadata clips to playlist window 380 to add them to the playlist (Ex-1004, 8:40-42), it would have been obvious to similarly use a playlist window to add an audio clip to the playlist.

That playlist window teaches the **second interactable graphical object** that **comprises an End Clip link**, and when **activated** by dragging an audio clip to it, the **begin time location** and **end time location** of the audio clip **are stored** in the playlist.

104. Second, for the reasons I explained in elements [3b] and [3d] with respect to video clips (starting at ¶89), Hedinsson's disclosure would have motivated a PHOSITA to add buttons that allow a user to either "save" or "cancel" a new audio clip. Thus, Hedinsson renders obvious **a second interactable graphical object** (save-clip button) that, **when activated** by user input, **stores** the **begin time location** and the **end time location** of an audio clip.

Analysis for Both Alternatives:

105. In each alternative, the **second interactable graphical object** (i.e., the playlist window or save-clip button) is **displayed simultaneously** with the first interactable graphical object (i.e., the trashcan icon or cancel-clip button) because these graphical objects are shown at the same time in the browser window that includes display 350 (Ex-1004, Fig. 3), and because a PHOSITA would have been motivated to display these objects at the same time in the proposed combination. A PHOSITA would have been motivated to show a consistent user interface for creating, adding, changing, and removing video and audio clips to increase ease of use. In addition, the decision to show these objects simultaneously (or to instead

display them on different screens) was a standard design choice with well-understood and predictable advantages and disadvantages. Moreover, a PHOSITA would have been specifically motivated to simultaneously display the two corresponding objects that allow a user to select whether to add or remove information, in order to reduce user confusion and frustration. As a result, a PHOSITA would have been motivated to simultaneously display: (1) the playlist window and the trashcan icon, or (2) the save-clip and cancel-clip buttons. For example, as shown in Belliveau, the user interface shows both a save-clip button and a cancel-clip button. (Ex-1012, Fig. 2). A PHOSITA would have understood that this would be a beneficial way to configure the interface because a user could easily discern how to either save a clip or discard his/her changes.

106. Finally, a PHOSITA would have found the phrase "and first media during" in element [3d] to be unclear. If the claim is interpreted to require storage of the clip's *duration*, this would have been an obvious variation of Hedinsson. Hedinsson states that, instead of an end position, metadata clip may "specify a length or duration of playback, from which the referenced end position can be derived." (Ex-1004, 4:55-58). Storing both an end position and duration would have been a trivial variation of Hedinsson's teachings. A PHOSITA would have been motivated to store both pieces of data in an implementation that showed the total length of the composite presentation in display 350 because it would reduce

computational steps required to display both the total duration and the end position of each clip. In addition, a PHOSITA would have been motivated to store both pieces of data in the metadata clip because it requires a minimal amount of additional storage and increases the compatibility of the metadata clip's format with other (or future) systems.

CLAIM 4

107. Dependent claim 4 is substantially the same as dependent claim 2, except: (1) claim 4 recites **a second type virtual clip** (rather than "a first type virtual clip" recited in claim 2), (2) claim 4 recites a **designated beginning** (rather than "a beginning" recited in claim 2), and (3) claim 4 recites a **second designated end point** (rather than "a designated endpoint" recited in claim 2). (Ex-1013, 2). As a result, it is my opinion that dependent claim 4 is obvious for the reasons that I provided above for independent claim 1 and dependent claim 2. In the next few paragraphs, I address the differences between claims 4 and 2.

108. As I explained above in connection with element [2a], Hedinsson's **composite virtual clip** comprises audio and/or video clips (either of which teaches the **one or more second type virtual clips**). To the extent the "first type virtual clip" (recited in claim 2) must be different than the **second type virtual clip** (recited in claim 4), these elements are taught by the audio and video clips, respectively, or vice versa.

109. To the extent claim 4 requires the **designated beginning** to specify a particular starting location within the media file, Hedinsson teaches this for both video and audio clips. Each of the video metadata clips identified in Hedinsson's exemplary playlist 240a start at a particular starting location within the media file. (Ex-1004, Fig. 2). For example, metadata clip points to video stream 237b starting at 00:02:00. (Ex-1004, 9:48-51). In addition, Hedinsson's exemplary playlist 240a includes a metadata clip 215d that points to an audio clip, and it specifies a beginning time for that audio clip within that audio stream. (Ex-1004, Fig. 2). Hedinsson also contemplates starting an audio clip at 00:02:51. (Ex-1004, 6:37-41). Further, as I discussed in connection with elements [3c] and [3e], Hedinsson teaches that a user can create new metadata clips and can select the start and end positions for those clips. (*E.g.*, Ex-1004, 9:1-13). Also, to the extent not disclosed in Hedinsson, a PHOSITA would have been motivated by Hedinsson's disclosure to provide a user with the flexibility to specify a particular beginning for the selected audio clip(s). (Ex-1004, Fig. 2, 4:22-31 (explaining that "any arbitrary media file segments" can be streamed "without the need to generate separate media files"), 1:32-37 (indicating that users may want to "replace[] audio tracks" to "enhance the entertainment value of content")). Indeed, as I explained in elements [3b] and [3d], Hedinsson indicates that audio clips are created and added to a

playlist in the same (or a substantially similar) way as video clips. (*E.g.*, Ex-1004, 8:16-51, 8:63-9:24, 5:46-47, 4:59-64, Fig. 2).

CLAIM 5

110. Dependent claim 5 is substantially the same as claim 3, except: (1) claim 5 depends from claim 4 (where claim 3 depends from claim 2), and (2) claim 5 recites **the second type virtual clip**, which was introduced in claim 4 (where claim 3 recites "the first type virtual clip," which was introduced in claim 2). (Ex-1013, 2). As a result, it is my opinion that dependent claim 5 is obvious for the reasons that I provided above for independent claim 1 and dependent claims 3 and 4.

CLAIM 6

> ***"6. The method of claim 1, wherein said composite virtual clip further comprises one or more first type virtual clips and one or more second type virtual clips."***

111. It is my opinion that dependent claim 5 is obvious for the reasons that I provided above for independent claim 1 and dependent claims 2 and 4. As I explained in elements [2a] and [4a], the video clips are the **first type virtual clips**, and the audio clips are the **second type virtual clips**, or vice versa.

CLAIM 7

"7[a]. A method to display a composite virtual clip, comprising:"

112. It is my understanding that a claim's preamble may or may not be limiting, but in any event, Hedinsson discloses **a method to display a composite virtual clip**. Hedinsson **displays** a "sequential media stream" of video clips from different source media files using a playlist, e.g., playlist 240a. (Ex-1004, 5:37-47, 9:37-67, 8:16-19, Figs. 2-4). As I explained above in element [1a], Hedinsson teaches that its playlist includes multiple metadata clips that each point to a specific audio or video clip. (*E.g.*, Ex-1004, Fig. 2, 4:11-58, 5:31-47). The media stream and/or its playlist are **a composite virtual clip**.

"[7b] providing a storage location for a composite virtual clip stored in a composite virtual clip file, wherein the composite virtual clip is configured to access (N) saved virtual clips;"

113. Hedinsson discloses **providing a storage location for** the playlist (**a composite virtual clip stored in a composite virtual clip file**). Hedinsson's "[e]nvironment 200 includes web server 220, client 225, playlists 240a-240b, and media server," and a PHOSITA would understand Hedinsson's playlist to be stored. (Ex-1004, 5:21-23, 7:15-17). Also, Hedinsson discloses that a user can create metadata clips and "store the newly created metadata clips in playlist 340," which also indicates that the playlist is stored. (Ex-1004, 8:63-9:13, Fig. 4 (step

410)). In addition, given the playlist's structure, function, and contents, a PHOSITA would understand Hedinsson to disclose that its playlist is stored as a **file**. (Ex-1004, Figs. 2-3). A file was (and is) the basic unit of storage that enables a computer to distinguish one set of information from another. It was (and is) a complete, named collection of information (such as a set of data used by a program. Hedinsson's playlist is a complete, named collection of information used by a program (the media player service on the web server) to display a sequential media stream. Hedinsson specifically identifies its playlist (e.g., playlist 240a), and the playlist has an associated title ("My favorite animal video clips."). (Ex-1004, Fig. 2). From this, a PHOSITA would have understood Hedinsson's playlist to be **stored** in a **file**.

114. In the alternative, it would have been obvious to **store** Hedinsson's playlist as a **file** because Hedinsson states that the playlist renders a new "*media* file" unnecessary, suggesting that the playlist is a different type of file. (Ex-1004, 4:27-31, 5:63-67). Also, Hedinsson indicates that the playlist can later be retrieved by the user or others. (Ex-1004, 9:32-34). This would encourage a PHOSITA to store Hedinsson's playlist in a file because this would facilitate access by different users. Moreover, storing the playlist in a file would have been an obvious implementation of Hedinsson because a file is a basic unit of storage, would have simplified access to the playlist, and was well-known and predictable.

115. Hedinsson discloses or renders obvious that **the composite virtual clip is configured to access (N) saved virtual clips**. As I explained above in my discussion of elements [1a] and [1b], Hedinsson's playlist includes multiple metadata clips that are used to sequentially stream a number of saved video clips, **(N)**, which can be any number. For example, for playlist 240a, **(N)** equals 3, and the **virtual clips** are the video clips referenced by metadata clips 215a-c. (Ex-1004, 5:37-47, 9:37-53, Fig. 2). Also, Hedinsson teaches video clips **saved** on media server 230. (Ex-1004, 5:26-31, 9:1-4, Fig. 2). Hedinsson also teaches video clips **saved** on media servers 130a, 130b and explains that these can correspond to media server 230. (Ex-1004, 2:46-50, 5:32-36).

116. The claim language's reference to both "a composite virtual clip" and a "composite virtual clip file" is unclear, but Hedinsson teaches or renders obvious the claim for substantially the same reasons I have provided under any reasonable interpretation of this claim language. In my opinion, the claim language is unclear for three reasons. First, both the preamble and this limitation refer to "a composite virtual clip," and it is not clear whether this must be the same composite virtual clip. Second, the claim refers to "a composite virtual clip stored in a composite virtual clip file," but later states that "the composite virtual clip is configured to access (N) saved virtual clips" where the activated link is "encoded in the composite virtual clip file." Third, the corresponding portion of the specification

simply states: "provid[ing] a storage location for a composite virtual clip file configured to access [the] saved clips." (Ex-1001, 12:61-64). In any event, to the extent the claim requires that the **composite virtual clip** be different than the **composite virtual clip file**: Hedinsson's playlist (the **composite virtual clip**) is stored in a file (the **composite virtual clip file**); the playlist is configured to access the video clips; and as required by element [7d], links to those clips are stored in the playlist's file. In other words, the **composite virtual clip file** is the electronic manifestation of the **composite virtual clip**. In my analysis of other elements, I have used the term "playlist" to refer both to the information itself as well as the file in which it is located.

"[7c] initially setting (P)=1;"

117. Ford describes an exemplary do-until loop that begins with "i = 1." (Ex-1005, 26). This teaches **initially setting (P)=1**, where **(P)** is taught by Ford's "i" because both "i" and "(P)" are arbitrary names for a variable (a counter), as I explained in element [1c]. Both "i" and "(P)" are arbitrary names for the variable that is used to count the number of iterations of the loop, and the selection of the name of this variable has no impact on the function of the code. As I explained above in the section entitled the "Rationale Supporting Obviousness," and as I address in more detail in elements [7d] and [7e], a PHOSITA would have been motivated to display Hedinsson's playlist using Ford's do-until loop, which counts

the iterations of the loop by first setting a variable to 1, and would have had a reasonable expectation of success in doing so. For the reasons I explained above in element [1c], a PHOSITA implementing Hedinsson in this way would have been motivated to initially set its counting variable to "1," as described in Ford. (Ex-1005, 26).

"[7d] activating a (P)th link encoded in the composite virtual clip file to stream a (P)th saved virtual clip to a user's device;"

118. As I explained above in my discussion of element [1d], Hedinsson's metadata clip points to a particular video clip (**saved virtual clip**) using a source media filename, URL, and start and end positions. The specification states that "[t]he link may be in the form of a pointer, electronic bookmark, hyperlink, or any other type of link." (Ex-1001, 8:39-41). In my opinion, the **link** is taught by: (1) the URL (which includes an IP address in Hedinsson's example) and source media filename, and/or (2) the URL, source media filename, and start and end positions. Also, as I explained above in my discussion of element [7b], Hedinsson teaches or suggests that the metadata clip (including the **link**) is stored in the playlist's file (**encoded in the composite virtual clip file**). Indeed, Hedinsson states that a user can create metadata clips and "store the newly created metadata clips in playlist 340." (Ex-1004, 8:63-9:13).

119. Hedinsson expressly or inherently teaches **activating a link** in a metadata clip **to stream a saved virtual clip** (the corresponding video clip) **to a user's device** (client 225 for playback on display 350). (Ex-1004, 9:37-67, 5:37-47, 2:51-52, 4:28-31). Hedinsson uses a metadata clip's source media filename, URL (e.g., IP address), and start and end positions to "locate[]," "retrieve[]," and "stream[]" the corresponding video clip. (Ex-1004, 4:32-64, 5:37-47, 9:37-67). From this, a PHOSITA would have understood Hedinsson to disclose **activating** the metadata clip's **link to stream** a video clip to the user's device. Also, even if this is not expressly disclosed, it is inherently disclosed by Hedinsson because using a link to locate, retrieve, and stream a video clip necessarily involves **activating the link**. Alternatively, this would have been an obvious implementation of Hedinsson given Hedinsson's disclosure and because a PHOSITA would have been motivated to activate the link to facilitate access to the video clip.

120. Hedinsson discloses or renders obvious activating a **(P)th link to stream a (P)th saved virtual clip**. Initially, **(P)** equals 1 (element [7c]), and as I explained in the preceding paragraphs, Hedinsson discloses or renders obvious activating a first link to stream a first saved video clip. For example, Hedinsson's playlist 240a includes a first metadata clip 215a, which points to the first video clip (in stream 237a), and Hedinsson activates that link to stream the first video clip.

(Ex-1004, Fig. 2, 9:46-48, 4:32-55). In addition, to the extent required, Hedinsson discloses or renders obvious activating a link for *each* of the N video clips (element [7b]) because Hedinsson displays a playlist that includes a set of metadata clips. Hedinsson shows an example playlist that includes three video clips (identified by metadata clips 215a, 215b, and 215c), and Hedinsson teaches that each of these metadata clips are used to stream the corresponding video clips in sequential order. (Ex-1004, 9:37-67, 4:33-64). For the reasons previously explained, a PHOSITA would have understood Hedinsson to disclose (or alternatively would have been motivated to implement Hedinsson to include): (1) activating a first link (in metadata clip 215a) to stream a first video clip (an excerpt of stream 237a), (2) activating a first link (in metadata clip 215b) to stream a first video clip (an excerpt of stream 237b), and (3) activating a first link (in metadata clip 215c) to stream a first video clip (an excerpt of stream 237c). Indeed, Hedinsson's claim 1 recites "process[ing] the playlist without generating a new media file to retrieve <u>from each of the at least two metadata clips</u> in the playlist the referenced source media, the referenced start position, and the referenced stop position, for a playback to begin at the referenced start position and end at the referenced end position."

"[7e] determining if (P) equals (N); when (P) is less than (N), setting (P) equals (P)+1; and repeating said activating and determining until (P) equals (N)."

121. As I explained in my discussion of elements [7a], [7b], and [7d], Hedinsson displays a sequential media stream of video clips using a playlist that includes N metadata clips, which each include a link to a corresponding video clip. (*E.g.*, Ex-1004, 9:37-67). As I explained in the section entitled "Rationale Supporting Obviousness," a PHOSITA would have been motivated to use a do-until loop, taught by Ford, to display the media stream using the playlist. In particular, a PHOSITA would have been motivated to use a do-until loop that uses one metadata clip to stream a video clip (by activating its link) during each iteration and repeats until all N metadata clips have been used, so that the media stream includes each of the N video clips (from element [7b]).

122. The language included in element [7e] is unclear for substantially same reasons as I provided in connection with element [1e] (in ¶75). In short, it is unclear whether the loop iterates N times or N-1 times. However, in my view, Hedinsson-Ford renders the claim obvious under either interpretation. I explain in detail why it would have been obvious to iterate the loop N times in the proposed Hedinsson-Ford combination. But, if the claim only requires N-1 iterations, then the first N-1 iterations of the Hedinsson-Ford loop render this claim obvious (because the claim does not prohibit additional iterations of the loop).

123. The proposed Hedinsson-Ford combination discloses or renders obvious element [7e] for substantially the same reasons I provided for element [1e]. Elements [1e] and [7e] differ in that claim 7 uses P to count the number of loops (element [7c]), where claim 1 uses M; however, the combination applies equally no matter which name is given to this variable. As I explained in elements [1c] and [7c], the variable names are arbitrary, and the selection of a particular name has no impact on the function of the code. Thus, for the reasons explained in element [1e], the combination discloses or renders obvious **determining** the *condition*, **setting (P) equals (P)+1 when (P) is less than (N)** and **repeating** the loop's *statements* **and determining until** the *condition* is true. Also, a PHOSITA would have been motivated to use **(P) equals (N)** as the *condition* because P is used to count the number of loops, P begins at 1, and the loop should repeat N times (once for each video clip to be streamed). Further, for this do-until loop, a PHOSITA would have been motivated to include the **activating** step (i.e., element [7d]) in the loop's *statements* so that each video clip is streamed (by activating the link in its corresponding metadata clip). (Ex-1004, 9:37-67, 4:32-58).

124. Also, to the extent required by the claim, the combination discloses or renders obvious element [7d] when it is **repeated** pursuant to element [7e]. For the reasons I explained above in connection with element [7d], Hedinsson discloses or

renders obvious performing the steps recited in element [7d] for *each* of the N video clips (**saved virtual clips**).

125. Finally, to the extent that the combination of elements [7d] and [7e] requires the pointers to be activated in sequential order, this is disclosed or rendered obvious by Hedinsson. Hedinsson processes the video clips in sequential order and depicts playlist 240a in the same order. (Ex-1004, 9:37-67, Fig. 2). Alternatively, this ordering would have been obvious in view of Hedinsson because a PHOSITA would have been motivated to activate the links associated with the video clips in sequential order to facilitate Hedinsson's sequential play back. (Ex-1004, 5:37-47, 7:45-47, 9:44-53, 9:61-67).

CLAIMS 8-10

126. Dependent claim 8 is substantially the same as claim 2, except claim 8 depends from claim 7 (where claim 2 depends from claim 1). (Ex-1013, 3). As a result, it is my opinion that dependent claim 8 is obvious for the reasons that I provided above for independent claim 7 and dependent claim 2.

127. Dependent claim 9 is substantially the same as claim 4, except claim 9 depends from claim 7 (where claim 4 depends from claim 1). (Ex-1013, 3). Dependent claim 9 recites "a designated start point" (where claim 4 uses the term "a designated beginning"), but it is my opinion that a "designated start point" is satisfied for the reasons I provided for "a designated beginning." As a result, it is

my opinion that dependent claim 9 is obvious for the reasons that I provided above for independent claim 7 and dependent claim 4.

128. Dependent claim 10 is substantially the same as claim 6, except claim 10 depends from claim 7 (where claim 6 depends from claim 1). (Ex-1013, 3). As a result, it is my opinion that dependent claim 10 is obvious for the reasons that I provided above for independent claim 7 and dependent claim 6.

CLAIM 13

129. Element [13a] recites "**[a] non-transitory computer usable medium encoded with a computer program product to make a composite virtual clip and usable with programmable computer processor disposed within a controller**." It is my understanding that a claim's preamble may or may not be limiting, but in any event, Hedinsson discloses this subject matter. Hedinsson teaches that "media player service 321 executing on processor 323 of web server 320" creates and updates a playlist and associated metadata clips (Ex-1004, 9:9-13, 8:14-33), and that "[w]eb server 320 includes processor 323 and memory 322 storing media player service 321" (Ex-1004, 7:17-18). Thus, Hedinsson teaches a **non-transitory computer usable medium** (memory 322) **encoded with a computer program product** (media player service 321) **to make a composite virtual clip** (a new playlist and/or the corresponding media presentation, as I discussed in connection with element [1a]) **and usable with programmable**

computer processor disposed within a controller (processor 323 in web server 320). (*See* Ex-1004, 7:15-31). Moreover, to the extent required, Hedinsson discloses or renders obvious a non-volatile (persistent) memory for storing media player service 321. Hedinsson states that "memory 322 stor[es] media player service 321." (Ex-1004, 7:17-18). A PHOSITA would have known that software (like media player service 321) is typically stored in a non-volatile (persistent) memory. As a result, a PHOSITA would have understood Hedinsson to teach that memory 322 is (or includes) a non-volatile memory. Moreover, even if this is not taught by Hedinsson, Hedinsson suggests that media player service 321 is stored in a non-volatile memory somewhere because Hedinsson uses this software to implement its method. Further, a PHOSITA would have been motivated to implement Hedinsson to store media player service 321 in a non-volatile memory so that the software would not be lost if/when the web server is rebooted and/or loses power.

130. The remaining elements of claim 13 (i.e., elements [13b] - [13e]) are the same as the corresponding elements in claim 1 except that they are directed to **computer readable program code which causes said programmable computer processor to** perform the method steps recited in claim 1. (Ex-1013, 4). As a result, it is my opinion that these elements are disclosed and/or rendered obvious

by the Hedinsson-Ford combination for the reasons that I provided above for elements [1b] through [1e].

CLAIMS 14-18

131. Dependent claim 14 is substantially the same as dependent claim 2, except claim 14 depends from claim 13 (where claim 2 depends from claim 1). (Ex-1013, 5). As a result, it is my opinion that dependent claim 14 is obvious for the reasons that I provided above for independent claim 13 and dependent claim 2.

132. Dependent claim 15 is substantially the same as dependent claim 3, except: (1) claim 15 depends from claim 14, which depends from claim 13 (where claim 3 depends from claim 2, which depends from claim 1), and (2) claim 15 is directed to **computer readable program code which causes said programmable computer processor to** perform the method steps recited in claim 3. (Ex-1013, 5). As a result, it is my opinion that dependent claim 15 is obvious for the reasons that I provided above for independent claim 13 and dependent claims 3 and 14.

133. Dependent claim 16 is substantially the same as dependent claim 4, except claim 16 depends from claim 13 (where claim 4 depends from claim 1). (Ex-1013, 6). As a result, it is my opinion that dependent claim 16 is obvious for the reasons that I provided above for independent claim 13 and dependent claim 4.

134. Dependent claim 17 is substantially the same as dependent claim 5, except: (1) claim 17 depends from claim 16, which depends from claim 13 (where

claim 5 depends from claim 4, which depends from claim 1), and (2) claim 17 is directed to **computer readable program code which causes said programmable computer processor to** perform the method steps recited in claim 5. (Ex-1013, 6). As a result, it is my opinion that dependent claim 17 is obvious for the reasons that I provided above for independent claim 13 and dependent claims 5 and 16.

135. Dependent claim 18 is substantially the same as dependent claim 6, except claim 18 depends from claim 13 (where claim 6 depends from claim 1). (Ex-1013, 7). As a result, it is my opinion that dependent claim 18 is obvious for the reasons that I provided above for independent claim 13 and dependent claim 6.

CLAIM 19

136. Element [19a] recites "**[a] non-transitory computer usable medium encoded with a computer program product to display a composite virtual clip and usable with programmable computer processor disposed within a controller**." As a result, similar to element [13a], element [19a] recites "**[a] non-transitory computer usable medium encoded with a computer program product … usable with programmable computer processor disposed within a controller**," and similar to element [7a], element [19a] recites "**to display a composite virtual clip**." Thus, to the extent it is limiting, Hedinsson discloses or renders obvious element [19a] for substantially the same reasons explained for elements [13a] and [7a]. In particular, Hedinsson teaches that "media player

service 321 executing on processor 323 of web server 320" uses playlist to retrieve audio and video clips to display a media presentation (Ex-1004, 5:41-47, 9:37-67), and that "[w]eb server 320 includes processor 323 and memory 322 storing media player service 321." (Ex-1004, 7:17-18). Thus, Hedinsson teaches a **non-transitory computer usable medium** (memory 322) **encoded with a computer program product** (media player service 321) **to display a composite virtual clip** (the playlist and/or resulting media presentation, as I discussed in connection with element [7a]) **and usable with programmable computer processor disposed within a controller** (processor 323 in web server 320). Moreover, to the extent required, Hedinsson discloses or renders obvious a non-volatile (persistent) memory for storing media player service 321 for the reasons I explained in connection with element [13a].

137. The remaining elements of claim 19 (i.e., elements [19b] - [19e]) are the same as the corresponding elements in claim 7 except that they are directed to **computer readable program code which causes the programmable computer processor** to perform the method steps recited in claim 1. (Ex-1013, 7). As a result, it is my opinion that these elements are disclosed and/or rendered obvious by the Hedinsson-Ford combination for the reasons that I provided above for elements [7b] through [7e].

CLAIMS 20-22

138. Dependent claim 20 is substantially the same as dependent claim 8, except claim 20 depends from claim 19 (where claim 8 depends from claim 7). (Ex-1013, 8). As a result, it is my opinion that dependent claim 20 is obvious for the reasons that I provided above for independent claim 19 and dependent claim 8.

139. Dependent claim 21 is substantially the same as dependent claim 9, except claim 21 depends from claim 19 (where claim 9 depends from claim 7). (Ex-1013, 8). As a result, it is my opinion that dependent claim 21 is obvious for the reasons that I provided above for independent claim 19 and dependent claim 9.

140. Dependent claim 22 is substantially the same as dependent claim 10, except claim 22 depends from claim 19 (where claim 10 depends from claim 7). (Ex-1013, 8). As a result, it is my opinion that dependent claim 22 is obvious for the reasons that I provided above for independent claim 19 and dependent claim 10.

Ground 2. Claims 11-12 and 23-24 are obvious over Hedinsson in view of Ford and Kostello

141. In my opinion, claims 11-12 and 23-24 would have been obvious over Hedinsson and Ford (as discussed under Ground 1 and incorporated herein), further in view of U.S. Patent No. 8,745,500 B1 ("**Kostello**")(Ex-1006).

142. Kostello was not cited during prosecution of any application leading to the '103 patent. (Ex-1001, code (56)).

A. Overview of the Ground

143. Dependent claims 11, 12, 23, and 24 recite pseudocode that uses a do-until loop to add a set of **transition effect links** or **effect links** to the **composite virtual clip file**. Specifically, claims 11 and 23 recursively "updat[e] the composite virtual clip file to include [an] (i)th transition effect link" that is configured "to be activated after activating an (i)th virtual clip link and before activation of an (i+1)th virtual clip link." Claims 12 and 24 recursively "updat[e] the composite virtual clip file to include [a] (k)th effect link" that is configured "to be activated simultaneously with an (k)th virtual clip link."

144. In my opinion, these claims would have been obvious over Hedinsson in view of Ford and Kostello. Hedinsson contemplates applying effects to video clips (including both image and transition effects), but it does not expressly discuss selecting *different* effects for *different* video clips. However, a PHOSITA would have been motivated to implement Hedinsson-Ford to provide this functionality, which is disclosed in Kostello, so that a user has additional flexibility in playlist creation, as contemplated by Hedinsson.

1. Overview of Kostello

145. Kostello describes the creation of a seamless composite presentation that includes segments of different video clips. (Ex-1006, 5:57-6:3, 8:11-23). Kostello's editing interfaces allow a user to: highlight segments of video clips for

inclusion in the presentation (Ex-1006, 6:4-49, 8:44-48, Fig. 4) and to specify video, transition, and music effects (Ex-1006, 6:57-8:10, 8:48-53, Figs. 5-7). Once these "enhancements" are finished, the user's device generates either "a single video file" or "a metadata file … in connection with a video file that identifies one or more of the segments, the highlights, [and] the enhancements." (Ex-1006, 8:11-23, 5:22-29, 8:53-56).

146. In Figure 5 (below), Kostello shows an interface that allows a user to add effects to the presentation's segments. (Ex-1006, 3:28-30).

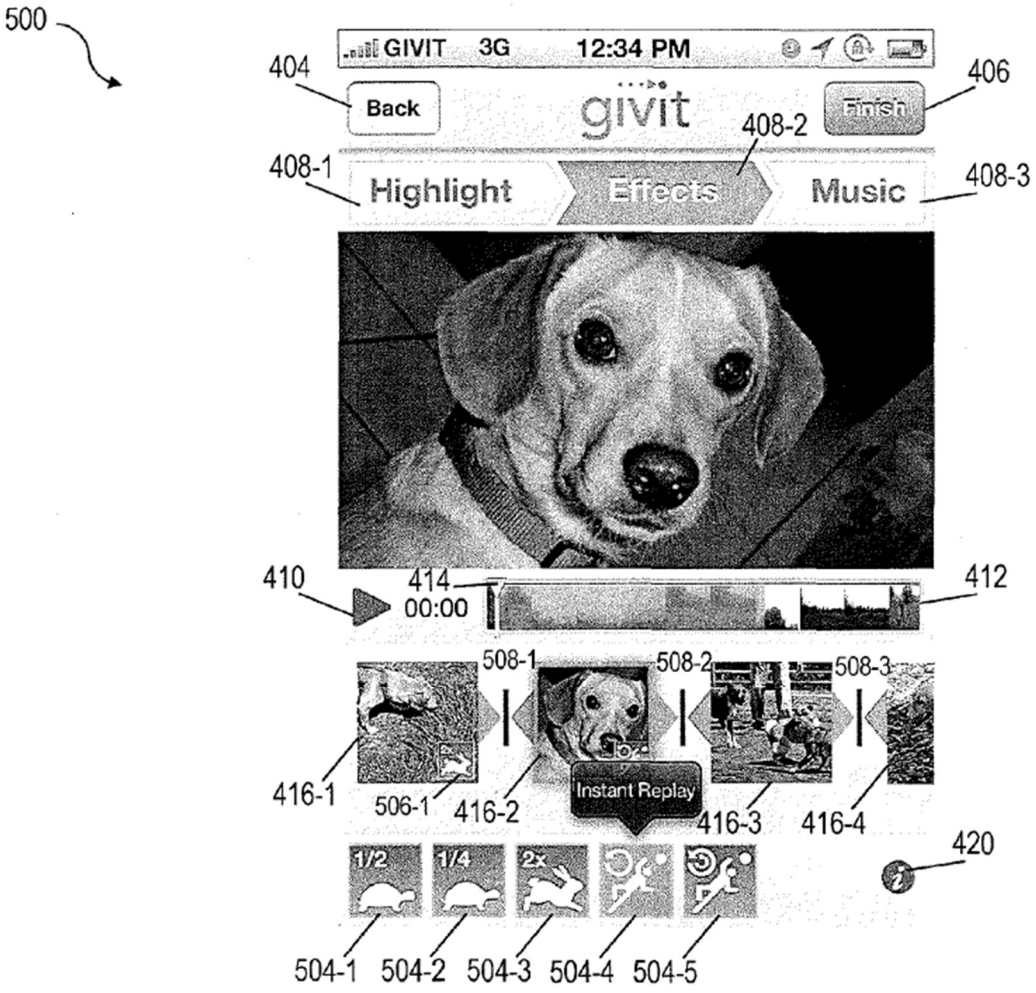


FIG. 5

A user applies an effect to a particular segment "by activating both the corresponding thumbnail 416 and a corresponding effects element 504." (Ex-1006, 7:8-11, 6:58-63). "For example, a first effect element 504-1 can be used for playing back the corresponding video segment at half speed," and "a fourth effect element 504-4 can be used to cause the corresponding video segment to be replayed once within the greater video clip (i.e., a single instant replay)." (Ex-1006, 6:63-7:4). Effects can also include a "visual overlay" or "change of color of the segment." (Ex-1006, 2:6-10, claim 9).

147. Figure 6 (below) shows an interface for selecting a transition effect between any two consecutive segments. (Ex-1006, 3:31-33).

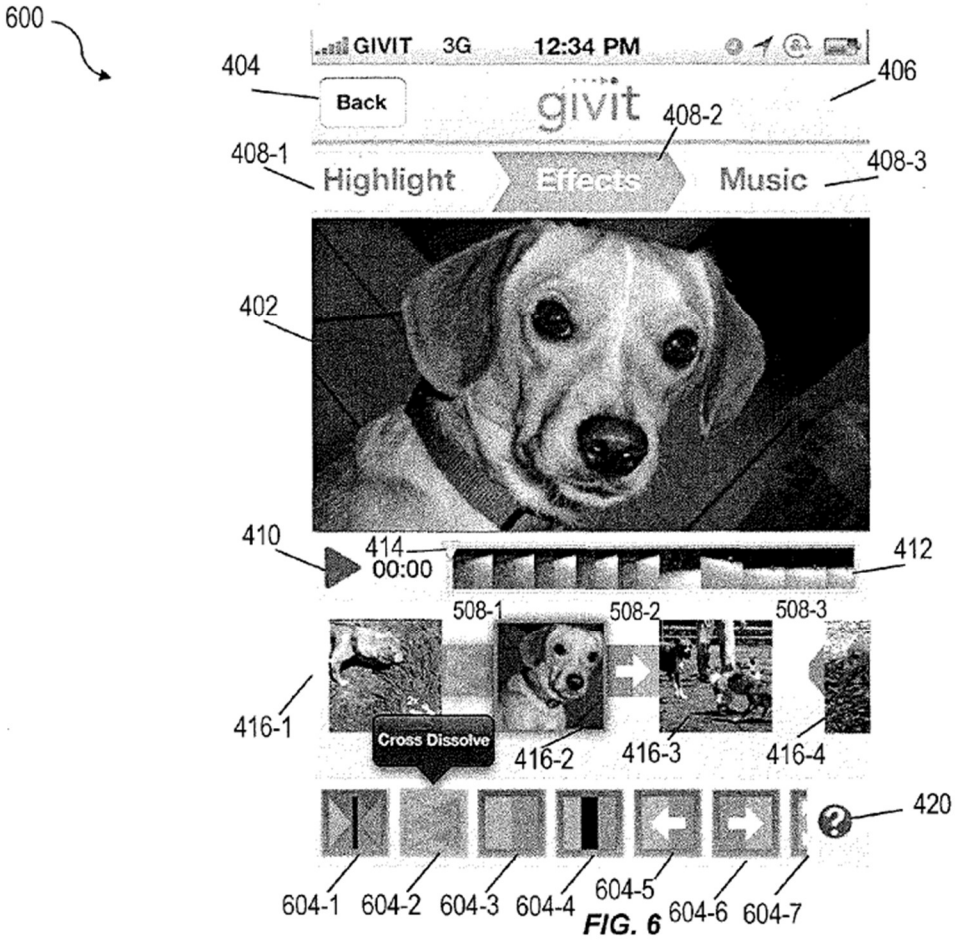


FIG. 6

Transition effects elements 604 are displayed when a user activates "a space between two thumbnails 416." (Ex-1006, 7:15-24). If the user selects a particular transition element 604, the presentation will display the corresponding effect when transitioning between the video clip segments associated with the adjacent thumbnails 416. (Ex-1006, 7:22-60, Fig. 6). "For example, [activating] a first transition effects element 604-1 … [causes] a fade to black effect" between the corresponding segments, where activating "[a] second transition effects element 604-2 … [causes] a cross dissolve effect." (Ex-1006, 7:24-30).

B. Rationale (Motivation) Supporting Obviousness

148. The rationale supporting the combination of Hedinsson and Ford is explained in Ground 1, and for the reasons explained below, a PHOSITA would have been motivated to implement Hedinsson's effects by allowing a user to select different effects for different video clips, as taught by Kostello.

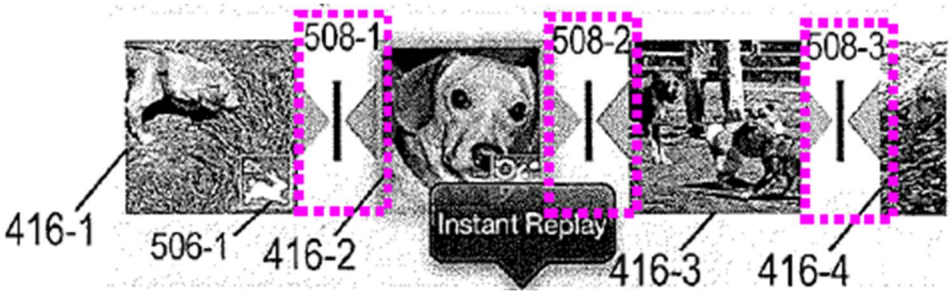
149. Hedinsson teaches that its playlist may include other details, such as "transition effects" between video clips and "image overlays" for video clips. (Ex-1004, 6:2-8). Hedinsson's exemplary playlist 240a specifies that "a two second cross fading effect" should be applied between video clips, and its independent claims recite "transitional effects between the at least two metadata clips retrieved in the playlist." (Ex-1004, 6:2-5, claims 1, 12, Fig. 2). The specific language used in Hedinsson indicates that multiple transition effects and/or image overlays can be applied to a single playlist: Hedinsson states that a playlist (singular) may specify "transition effects" and "image overlays" (plural). (Ex-1004, 6:2-5, claims 1, 12). These disclosures would have motivated a PHOSITA to implement Hedinsson's playlist in a way that allows a user to select effects (e.g., image and transition effects) for the video clips. They also would have motivated a PHOSITA to allow a user to select multiple image and/or transition effects in a playlist.

150. For those implementation details, a PHOSITA would have been motivated to look to Kostello, which describes a system similar to Hedinsson's

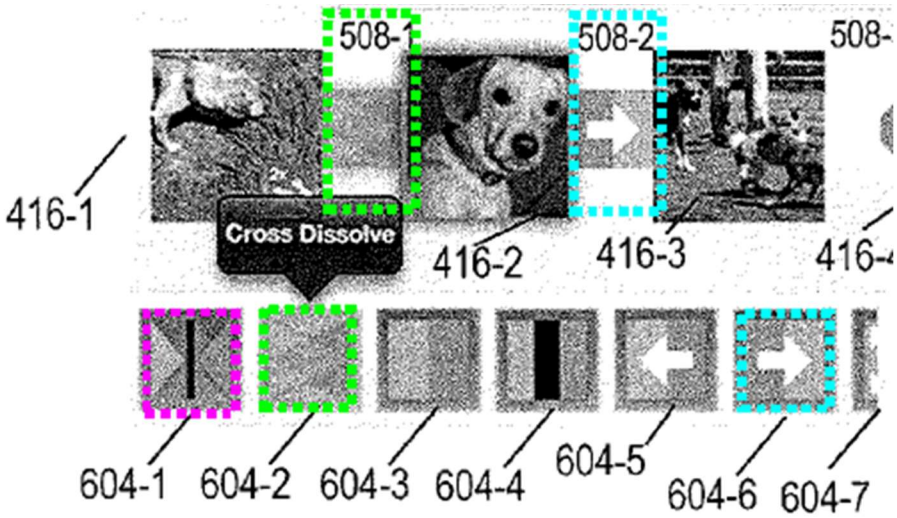
system. Kostello allows a user to create a seamless composite presentation from individual video clips. (Ex-1006, 5:57-6:3, 8:11-23, 8:53-56, Figs. 3-4). Kostello states that "multiple video clips or portions thereof can be concatenated into a single clip or a series of linked clips" by selecting thumbnails 302 (which each correspond to a clip) in the desired order. (Ex-1006, 5:57-67, Fig. 3). In Figure 4, Kostello shows an example where four video clips have been selected, each indicated by a corresponding thumbnail 416-1, 416-2, 416-3, or 416-4, and after a user applies any "enhancements," Kostello generates "one or more files" to "enable the playback of the video clips." (Ex-1006, 8:53-56, 8:11-23, Fig. 4). In fact, Kostello contemplates using a separate metadata file to store Kostello's enhancements. (Ex-1006, 8:11-23, 2:45-58, 5:24-29, 8:53-56). Kostello states that "a metadata file can be generated in connection with a video file that identifies one or more of the segments, the highlights, the enhancements such as the music and transitions, and the like." (Ex-1006, 8:19-23). Kostello also explains that the "generated or modified video file can comprise at least two related, but separate, data files. For example, one file may encapsulate metadata associated with the other file etc." (Ex-1006, 2:55-58).

151. As I explain in my summary of Kostello (¶¶145-147), Kostello's interface allows a user to select an image effect for each video clip (via effect element 504), and a transition effect for each transition between consecutive video

clips (via transition effects element 604). (Ex-1006, 6:57-7:60, Figs. 5-6). For example, Kostello states that "effects elements 504 can correspond to different effects to be applied to one or more of the segments associated with the thumbnails." (Ex-1006, 6:61-63). Also, Kostello shows an example where the user has selected *different* transition effects between *different* video clips: for transitions 508-1 and 508-2, a user has replaced a fade to black transition (604-1) with a cross dissolve effect (604-2) and a slide right effect (604-6), respectively, as illustrated in the following annotated excerpts of Figures 5 and 6.



Annotated Excerpt of Figure 5



Annotated Excerpt of Figure 6

152. A PHOSITA would have been motivated to implement Hedinsson-Ford to allow different image and transition effects for different videos, as contemplated by Kostello, because this would provide the user with additional flexibility in playlist creation. Indeed, Hedinsson seeks to allow users to flexibly create composite presentations that incorporate creative input from users. (Ex-1004, 1:32-53, 6:49-67, 8:45-48). Also, Hedinsson recognizes the benefits of "unif[ying]" "disparate video clips" in the presentation (Ex-1004, 5:47-50), and a PHOSITA would have understood that image and transition effects were known techniques to unify disparate video clips. For example, a PHOSITA would have known that image effects could unify video clips by adjusting different clips to enhance their visual similarity. Also, US2005/0084232, which was filed before the '103 patent, states that transition effects were known to unify video clips: "As is well known to those of ordinary skill in the art, video transition effects are used to soften abrupt scene changes between successive video clips." (Ex-1008, ¶35).

153. In addition, a PHOSITA would have been motivated to combine this feature of Kostello with Hedinsson-Ford because a PHOSITA would have understood that existing composite video clips that included separate videos also often included different image effects for different video clips and different transition effects between videos. For example, Herberger shows the use of different transition effects between different media clips in its Figures 4 and 5, and

Herberger states that it is "preferred" that the user be able to individually select which transition effect to apply at each transition. (Ex-1008, ¶35, Figs. 4-5). A PHOSITA would have known that users prefer to have the flexibility to select different effects and transitions in a presentation, as this is a longstanding concept in movies and other video creations. (Transitions and image effects have been included in movies for at least forty years.) Indeed, the '103 patent acknowledges that transition and image effects were known to PHOSITAs. (Ex-1001, 13:12-17, 13:49-52).

154. In the proposed Hedinsson-Ford-Kostello combination, a PHOSITA would have been motivated to use a programming loop to add these effects to Hedinsson's playlist. In particular, a PHOSITA would have been motivated to add, to the playlist, each of the image or transition effects. A PHOSITA would have understood that adding each of these effects to the playlist would require the repetition of programming statements, and as Ford explains, programmers prefer to use loops for repetitive actions (Ex-1005, 21-22).

155. Also, for the reasons I explained above in connection with Ground 1, a PHOSITA would have been motivated to use a do-until loop to add these effects. A PHOSITA would have known that a do-until loop would be beneficial because at least one effect should be added to the playlist and because looping should continue until all effects have been added. (Ex-1005, 23). Also, as I explained

above in connection with Ground 1, a do-until loop would have been obvious because there were a finite number of different types of loops, each of which were predictable, well-known, and within a PHOSITA's skill. In addition, a PHOSITA would have been motivated to use a do-until loop for adding effects because a PHOSITA would have been motivated to dupe and revise code when possible and, as explained in Ground 1, the PHOSITA would have been motivated to use this type of loop to create a new playlist and to stream the clips identified in an existing one.

C. Reasonable Expectation of Success

156. A PHOSITA would have had a reasonable expectation of success in making the proposed combination for the reasons I explained above in connection with Ground 1. In addition, in the relevant timeframe, transition effects and image effects were known and understood by PHOSITAs. I note that the '103 patent indicates that "those skilled in the art" were familiar with both. (Ex-1001, 13:12-17, 13:49-52). In addition, the use of these effects was predictable and within the skill of a PHOSITA.

D. Graham Factors

157. The Graham factors were discussed under Ground 1.

E. Analogous Art

158. In my view, Kostello is analogous art because it is in the same field as the '103 patent (maintaining information and sharing and annotating video data),

and it would have been reasonably pertinent to the video-sharing problems facing the named inventors. (Ex-1006, Title, Abstract, 1:5-22, 3:49-50, 4:13-16; Ex-1001, Title, Abstract, 1:18-22).

F. Claim Mapping

159. This section maps the challenged claims to the relevant disclosures of Hedinsson, Ford, and Kostello, where the claim text appears in bold-italics, and the relevant mapping follows.

CLAIM 11

"11[a]. The method of claim 7, further comprising adding a transition effect to the composite virtual clip by:"

160. I discuss claim 7 above in connection with Ground 1, and my analysis of claim 1 is also relevant to this claim. It is my understanding that a claim's preamble may or may not be limiting, but in any event, the proposed Hedinsson-Ford-Kostello combination discloses or renders obvious **a method** that **further comprises adding a transition effect to the composite virtual clip** (the media stream and/or its playlist). Kostello adds transition effects between consecutive video clips (508-1, 508-2, 508-3) using transition effects elements 604. (Ex-1006, 7:15-60, Fig. 6). As I explained in the section in Ground 2 titled "Rationale Supporting Obviousness," it would have been obvious to add this functionality to the Hedinsson-Ford combination. Indeed, Hedinsson teaches transition effects

between video clips (Ex-1004, 6:2-5, claims 1, 12, Fig. 2), and the patent admits transition effects were known to PHOSITAs (Ex-1001, 13:12-17).

"[11b] selecting the composite virtual clip, wherein the composite virtual clip is defined by (N) saved virtual clips and an order of presentation of each virtual clip from 1 to N;"

161. In my opinion, the proposed combination discloses or renders obvious **selecting the composite virtual clip** (the media stream and/or its playlist). In order to add Kostello's transition effects to a particular playlist, a PHOSITA would have been motivated to first select the playlist (and, by extension, its media stream) to facilitate the addition of those transition effects and to ensure they are added to the correct playlist.

162. For the reasons I explained in Ground 1 in connection with elements [1b] and [7b], Hedinsson discloses or renders obvious that **the composite virtual clip** (the media stream and/or its playlist) **is defined by (N) saved virtual clips** (video clips saved on a media server) **and an order of presentation of each virtual clip from 1 to N**. Hedinsson's playlist specifies the order in which a plurality of video clips should be played (Ex-1004, 5:37-47, 9:37-53, Fig. 2), and the number of video clips in the playlist teaches **(N)**, which can be any number.

"[11c] selecting a transition effect having a known storage location;"

163. In my opinion, Kostello teaches **selecting a transition effect**. (Ex-1006, 7:15-60, Fig. 6). For example, Kostello's Figure 6 shows that the user has selected a cross dissolve transition effect 604-2 to be applied to transition element 508-1, which occurs between the first video clip segment (represented by thumbnail 416-1) and the second segment (represented by thumbnail 416-2). (Ex-1006, Fig. 6). For the reasons I explained above in the section titled "Rationale Supporting Obviousness" in this Ground 2, Kostello also discloses or renders obvious selecting a transition effect 604 for each transition between consecutive segments. (Ex-1006, 7:15-24, 5:57-60, Figs. 5-6). Indeed, in Figure 6, Kostello shows that different transition effects have been applied between different segments (video clips). (Ex-1006, Fig. 6).

164. The combination discloses or renders obvious that each transition effect **has a known storage location**. In the interface shown in Figure 6, Kostello displays several different transition effects elements 604 that may be selected by a user, and Kostello specifies that its software can be downloaded to a mobile device or run on another device. (Ex-1006, 3:64-4:5, 9:47-65, Fig. 6). Also, Hedinsson teaches that its playlist "specifie[s]" transition effects. (Ex-1004, 6:2-5, Fig. 2). From this, a PHOSITA would have understood each reference to teach, or at least suggest, that each transition effect has a known storage location. In addition, a

PHOSITA would have been motivated to store each transition effect in a known location to facilitate its use in the proposed combination, as explained below in element [11e].

"[11d] initially setting (i)=1;"

165. Ford describes an exemplary do-until loop that begins with "i = 1." (Ex-1005, 26). This teaches **initially setting (i)=1**, where **(i)** is taught by Ford's "i." As I explained above in the section in Ground 2 entitled the "Rationale Supporting Obviousness," and as I address more in detail in elements [11e] and [11f], a PHOSITA would have been motivated to add transition effects to Hedinsson's playlist using Ford's do-until loop, and would have had a reasonable expectation of success in doing so. Also, as I explained in elements [1c] and [7c] in Ground 1, "i" is an arbitrary name for the variable that is used to count the number of iterations of the loop, and a PHOSITA implementing Hedinsson as proposed would have been motivated to initially set its counting variable to "1," as described in Ford, so that the counting variable is equal to the current iteration number of the loop.

"[11e] configuring an (i)th transition effect link pointing to a known (i)th transition effect storage location; configuring the (i)th transition effect link to be activated after activating an (i)th virtual clip link and before activation of an (i+1)th virtual clip link, wherein (i) is between 1 and (N)-1; updating the composite virtual clip file to include the (i)th transition effect link;"

166. As I explained in element [11c], the combination discloses or renders obvious **a known storage location** for each selected **transition effect**.

167. It would have been obvious to add a particular transition effect to the playlist by **configuring a transition effect link pointing to** the transition effect's **known storage location** and by **updating the composite virtual clip file** (the playlist) **to include** that **transition effect link**. A PHOSITA would have been motivated to add some sort of reference (a link) to the transition effect (**a transition effect link**) in the playlist (**the composite virtual clip file**) to identify the transition effect selected by the user for that transition. Indeed, the '103 patent uses the term "link" broadly to include "any … type of link." (Ex-1001, 8:39-41). Hedinsson states that its playlist can specify transition effects, and a PHOSITA would have understood this to mean that the playlist includes some type of reference (a link) to the desired transition effects. (Ex-1004, 6:1-8, Fig. 2). Stated differently, a PHOSITA would have understood Hedinsson to disclose that its playlist includes a reference to a transition effect, rather than the transition effect itself, because Hedinsson's playlist includes links to video clips, rather than a new composite media file itself, as I explained in elements [1d] and [7d]. A PHOSITA

would have understood using a link to be an efficient way to identify transition effects. Moreover, a PHOSITA would have been motivated to include a link in the playlist to facilitate a reference to the software code required to implement the desired transition effect. For example, Kostello contemplates "a cross dissolve effect" between two segments (Ex-1006, 7:27-30), and a PHOSITA would have understood that displaying this effect would require the system to execute code that blends, on a frame by frame basis, the end of one video and the beginning of the next video.

168. The combination discloses or renders obvious configuring an **(i)th** transition effect link pointing to a known **(i)th** transition effect storage location and updating the composite virtual file to include the **(i)th** transition effect link. Initially, **(i)** equals 1 (element [11d]), and as I explained in the preceding paragraphs, the combination discloses or renders obvious configuring a first transition effect link pointing to a first location associated with a first transition and updating the playlist to include that link. In addition, **(i) is between 1 and (N)-1**. As discussed in elements [11d] and [11f], **(i)** initially equals 1 and is incremented. Also, as discussed in element [11b], **(N)** is the number of video clips selected for inclusion in the playlist (i.e., a positive integer), and **(N)-1** equals the number of transitions between those video clips. The example shown in Kostello's Figure 6 includes four video clips (corresponding to thumbnails 416-1 to 416-4) and three

transitions between those video clips (508-1 to 508-3); in this example, (N) equals 4, and (N)-1 equals 3. In addition, to the extent required, the combination discloses or renders obvious selecting, configuring, and adding a link for *each* of the N-1 transitions because Kostello allows users to add different effects to a set of transitions. There are N video clips (as I explained in element [11b]), so there are N-1 transitions. Also, Kostello also discloses or renders obvious adding a transition effect 604 for each transition between consecutive segments, as I explained in element [11c] and in the section titled "Rationale Supporting Obviousness." (Ex-1006, 7:15-24, Figs. 5-6). This disclosure encompasses a situation, for example, where a user seeks to add transitions to a playlist that includes three video clips. For the reasons previously explained, a PHOSITA would have understood the combination to disclose (or alternatively would have been motivated to implement the combination to include) selecting, configuring, and adding a first link for a first transition (between the first video clip and the second video clip), and selecting, configuring, and adding a second link for a second transition (between the second video clip and the third video clip).

169. The combination discloses or renders obvious **configuring the i(th) transition effect to be activated after activating an (i)th virtual clip link and before activation of an (i+1)th virtual clip link**. A PHOSITA would have understood the combination to activate the transition effect link **after** the activation

of preceding video clip link and **before** the activation of the subsequent video clip link, because Kostello's transition effects are applied *between* two video clips (i.e., the preceding and subsequent video clips). (Ex-1006, 7:22-24). In addition, a PHOSITA would have been motivated to implement the combination in this manner given Kostello's disclosure and because transition effects between two sequential video clips were known in the art and contemplated by Hedinsson. (*E.g.*, Ex-1006, 7:22-24; Ex-1004, 6:2-5). This would have been a logical and desirable way to implement the proposed combination because a PHOSITA would have been motivated to activate links in the order in which the corresponding clips or effects are to be displayed. Further, a PHOSITA would have been motivated to implement the combination in this way because it activates links to video clips and transition effects in sequential order, as indicated by Hedinsson. (Ex-1004, 9:44-46).

"[11f] determining if (i) equals (N); when (i) is less than (N), setting (i) equals (i)+(1); and repeating selecting and configuring to add the (i)th transition effect to the composite virtual clip until (i) equals (N)-1."

170. As I explained above in the section in Ground 2 entitled the "Rationale Supporting Obviousness," a PHOSITA would have been motivated to add each of the transition effect links to the playlist using a programming loop

because loops are beneficial for repetitive actions like this, and it would have been obvious to use a do-until loop, as taught by Ford, for this purpose.

171. In particular, given Ford and Kostello, a PHOSITA would have been motivated to implement Hedinsson using a do-until loop that selects, configures, and adds one transition effect link during each iteration, and repeats until N-1 transition effects have been added to the playlist. This would allow the user to select a different transition effect between each video clip, as explained in element [11e].

172. The language included in element [11f] is unclear for substantially the same reasons I provided in connection with element [1e] in in ¶75. In addition, because the last phrase ("repeating…") refers to "(i) equals (N)-1," it is unclear whether the loop iterates N, N-1, or N-2 times. However, in my view, Hedinsson-Ford-Kostello renders the claim obvious under any of these interpretations. I explain in detail why it would have been obvious to iterate the loop N-1 times in the proposed Hedinsson-Ford-Kostello combination. But, if the claim only requires N-2 iterations, then the first N-2 iterations of the Hedinsson-Ford-Kostello loop render this claim obvious (because the claim does not prohibit additional iterations of the loop).

173. In addition, if the claim requires configuring and saving N transition effect links, it would still have been obvious because a PHOSITA also would have

been motivated to include an additional loop iteration (for a total of N iterations) so that a user can select a transition effect to appear at the end of the last video clip in the playlist, in order to provide additional creative flexibility to the user. Indeed, Hedinsson contemplates displaying a different playlist immediately following the instant playlist (Ex-1004, 6:13-20), which would further have motivated a PHOSITA to include this functionality.

174. For the reasons explained in elements [1e] and [7e], the proposed combination discloses or renders obvious **determining if (i) equals (N); when (i) is less than (N), setting (i) equals (i)+(1); and repeating** the *statements* within the loop **until** a *condition* is satisfied. Although claims 1 and 7 use the variables M and P, respectively, to count the number of loops, where this claim uses "i" (element [11d]), the combination applies equally no matter what name is given to the variable, as I explained in element [11c].

175. In the context of this loop, a PHOSITA would have been motivated to include both the **selecting** and **configuring** steps in the loop's *statements* because a user may choose to **select** different transition effects between different pairs of adjacent video clips in the playlist, and so that the appropriate link is **configured** and added to the playlist, as discussed in element [11e].

176. For the reasons I have explained, a PHOSITA would have been motivated to set, as the loop's *condition*, either: (1) **(i) equals (N)-1** so that the

loop repeats N-1 times, once for each transition between video clips in the playlist, or (2) **(i) equals (N)**, in order to allow the user to also select a transition to occur after the playlist. A PHOSITA would have understood this to be a design choice with predictable results, and implementing each option would have been within a PHOSITA's skill.

177. In addition, a PHOSITA would have been motivated to **determine if "i" equals N**—even if this is not the loop's condition—to check for errors in the loop's code. As Ford teaches, when working with loops, programmers "must constantly be on the look out for … endless loops," i.e., a loop that never ends. (Ex-1005, 30). A PHOSITA would have understood that, if the loop's condition is (i) equals (N)-1, then (i) should never equal (N) inside the loop. However, this could happen if there is a typographical error made when programming the loop. As a result, a PHOSITA would have been motivated to add a few lines of code (i.e., to **determine if (i) equals (N)**, and if so, to exit the loop) for error checking, as contemplated by Ford. (Ex-1005, 31-32).

178. Also, to the extent required by the claim, the proposed combination discloses or renders obvious element [11e] even after (i) is incremented and this step is repeated because, as explained in element [11e], it would have been obvious to select, configure, and add *each* transition effect link.

CLAIM 12

"12[a]. The method of claim 7, further comprising adding an effect to the composite virtual clip, wherein the effect includes at least one of a lensing effect and a second effect, by:"

179. I discuss claim 7 above in connection with Ground 1, and my analysis of claim 1 is also relevant to this claim. It is my understanding that a claim's preamble may or may not be limiting, but in any event, the proposed Hedinsson-Ford-Kostello combination discloses or renders obvious **a method** that **further comprises adding an effect to the composite virtual clip** (the media stream and/or its playlist), where **the effect includes at least one of a lensing effect and a second effect**. Kostello adds effects to video clips (508-1, 508-2, 508-3) using effects elements 504. (Ex-1006, 6:57-7:14, Fig. 5). Kostello also contemplates adding a "visual overlay" or "change of color of a segment." (Ex-1006, 2:6-10, claim 9). As I explained in the section in Ground 2 titled "Rationale Supporting Obviousness," it would have been obvious to add this functionality to the Hedinsson-Ford combination. Indeed, Hedinsson teaches adding "image overlays" to video clips (Ex-1004, 6:5-8), and the patent admits that effects such as overlying

a "color filter" or "image distortions" were known to PHOSITAs (Ex-1001, 13:12-17). Kostello's effects teach a **lensing effect**[4] and/or a **second effect**.

"[12b] selecting the composite virtual clip, wherein the composite virtual clip defined by (N) saved virtual clips and an order of presentation of each virtual clip from 1 to N;"

180. In element [11b] above, I explain why the proposed combination discloses or renders obvious **selecting the composite virtual clip, wherein the composite virtual clip defined by (N) saved virtual clips and an order of presentation of each virtual clip from 1 to N**. Although that analysis is provided in the context of adding Kostello's transition effects, the same analysis applies equally when adding Kostello's image effects. For example, in order to add Kostello's image effects to a particular playlist, a PHOSITA would have been motivated to first select the playlist (and, by extension, its media stream) to facilitate the addition of those effects and ensure they are added to the correct playlist.

[4] Although the term "lensing effect" is unclear, the Specification states that it at least includes "overlay of one or more color filters, image distortions, and annotations." (Ex-1001, 13:49-52).

"[12c] selecting an effect having a known storage location;"

181. In my opinion, Kostello teaches **selecting an effect**. (Ex-1006, 6:57-7:14, Fig. 5). For example, as shown in Figure 5, the user has selected an instant replay effect 504-4 for the video clip associated with thumbnail 416-2. (Ex-1006, Fig. 5). Kostello also contemplates adding a "visual overlay" or "change of color of a segment." (Ex-1006, 2:6-10, claim 9). For the reasons I explained above in the section titled "Rationale Supporting Obviousness" in this Ground 2, Kostello also discloses or renders obvious selecting an effect 504 for each video clip segment. (Ex-1006, 6:61-63, 7:8-10, 5:57-60, Fig. 5). Indeed, Kostello states that "[t]he effects elements 504 can correspond to different effects to be applied to one or more of the segments associated with the thumbnails 416" (Ex-1006, 6:61-63), and Kostello states that a segment is associated with an effect "by activating both the corresponding thumbnail 416 and a corresponding effects element 504" (Ex-1006, 7:8-10). I understand these passages to state that different effects elements 504 can be applied to different segments (video clips).

182. For substantially the same reasons as I explained in element [11c], the combination discloses or renders obvious that each effect **has a known storage location**. Indeed, Kostello's Figure 5 shows the available options for image effects 504 (Ex-1006, 6:57-7:14), and Hedinsson teaches that playlist 240 can include

"[r]eferences to captions, subtitles, image overlays, commentary, external web links, and other related content" (Ex-1004, 6:5-8).

"[12d] setting (k) initially equals to 1;"

183. Ford describes an exemplary do-until loop that begins with "i = 1." (Ex-1005, 26). This teaches **setting (k) initially equals to 1**, where **(k)** is taught by Ford's "i." As I explained above in the section in Ground 2 entitled the "Rationale Supporting Obviousness," and as I address more in detail in elements [12e] and [12f], a PHOSITA would have been motivated to add effects to Hedinsson's playlist using Ford's do-until loop, and would have had a reasonable expectation of success in doing so. Also, as I explained in elements [1c] and [7c] in Ground 1, "i" is an arbitrary name for the variable that is used to count the number of iterations of the loop, and a PHOSITA implementing Hedinsson as proposed would have been motivated to initially set its counting variable to "1," as described in Ford, so that the counting variable is equal to the current iteration number of the loop.

"[12e] configuring an (k)th effect link pointing to a known (k)th storage location; configuring the (k)th effect link to be activated simultaneously with an (k)th virtual clip, wherein (k) is between 1 and (N); updating the composite virtual clip to include the (k)th effect link;"

184. As I explained in element [12c], the combination discloses or renders obvious **a known storage location** for each selected **effect**.

185. *See* my analysis of element [11e]. It would have been obvious to add an effect to the playlist by **configuring an effect link pointing to** the effect's **known storage location** and by **updating the composite virtual clip file** (the playlist) **to include** that **effect link**. A PHOSITA would have been motivated to add some sort of reference (a link) to the effect (**an effect link**) in the playlist (**the composite virtual clip file**) to identify the effect selected by a user for that video clip. Indeed, the '103 patent uses the term "link" broadly to include "any … type of link." (Ex-1001, 8:39-41). The reasons explained in [11e] for transition effects apply equally to the effect recited in this claim. A PHOSITA would have been motivated to include a link in the playlist to reference the software code required to implement the desired effect. Indeed, Kostello contemplates an "instant replay" of a particular clip (Ex-1006, 7:1-3, Fig. 5), and a PHOSITA would have understood that the system would repeat the particular video clip during play back. As another example, Kostello describes effect 504-1, which "can be used for playing back the corresponding video segment at half speed" (Ex-1006, 6:63-65), and a PHOSITA

would have understood that the system would alter the normal display of the clip throughout the duration of the clip.

186. The combination discloses or renders obvious configuring a **(k)th** effect link pointing to a known **(k)th** storage location and updating the composite virtual file to include the **(k)th** effect link. Initially, **(k)** equals 1 (element [12d]), and as explained in the preceding paragraphs, the combination discloses or renders obvious configuring a first effect link pointing to a first location associated with a first effect and updating the playlist to include that link. In addition, **(k) is between 1 and (N)** because **(k)** initially equals 1 (element [12d]) and **(k)** is incremented until it equals (N) (element [12f]), which is the number of video clips (element [12b]). In addition, to the extent required, the combination discloses or renders obvious selecting, configuring, and adding a link for *each* of the N video clips (element [12b]) because Kostello allows users to add different effects to a set of video clips. Kostello also discloses or renders obvious adding an effect 504 to each video clip, as I explained in element [12c] and in the section titled "Rationale Supporting Obviousness." (Ex-1006, 6:61-63, 7:8-10, Fig. 5). This disclosure encompasses a situation, for example, where a user seeks to add effects to a playlist that includes two video clips. For the reasons previously explained, a PHOSITA would have understood the combination to disclose (or alternatively would have been motivated to implement the combination to include):

(1) selecting, configuring, and adding a first link for a first video clip, and

(2) selecting, configuring, and adding a second link for a second video clip.

187. The combination discloses or renders obvious **configuring the k(th) effect link to be activated simultaneously with an (k)th virtual clip**. A PHOSITA would have understood the combination to activate the effect link **simultaneously** with the corresponding video clip because Kostello's effects are applied to video clip segments. (Ex-1006, 6:61-63). In addition, a PHOSITA would have been motivated to implement the combination in this manner given Kostello's disclosure and because effects were known in the art and contemplated by Hedinsson. (*E.g.*, Ex-1006, 6:61-63; Ex-1004, 6:5-8). This would have been a logical and desirable way to implement the proposed combination because it facilitates applying the effect to the entire video clip. For example, some of Kostello's effects are applied to the entire video clip (e.g., half-speed playback via element 504-1, "visual overlay," and color change) (Ex-1006, 6:63-65, 2:6-10), and a PHOSITA would have been motivated to activate the effect link with the corresponding video clip to facilitate execution of the appropriate code at the appropriate time.

"[12f] determining if (k) equals (N); when (k) is less than (N), setting (k) equals (k)+(1); and repeating selecting and configuring to add the (k)th effect to the composite virtual clip until (k) equals (N)."

188. As I explained above in the section in Ground 2 entitled the "Rationale Supporting Obviousness," a PHOSITA would have been motivated to add the effects links to the playlist using a programming loop because loops are beneficial for repetitive actions like this, and it would have been obvious to use a do-until loop, as taught by Ford, for this purpose. In particular, given Ford and Kostello, a PHOSITA would have been motivated to implement Hedinsson using a do-until loop that selects, configures, and adds one effect link during each iteration, and repeats until N links have been added to the playlist. This would allow the user to select a different effect for each video clip, as explained in element [12e].

189. The language included in element [12f] is unclear for substantially the same reasons I provided in connection with element [1e] in in ¶75. In short, it is unclear whether the loop iterates N times or N-1 times. However, in my view, Hedinsson-Ford-Kostello renders the claim obvious under either interpretation. I explain in detail why it would have been obvious to iterate the loop N times in the proposed Hedinsson-Ford-Kostello combination. But, if the claim only requires N-1 iterations, then the first N-1 iterations of the Hedinsson-Ford-Kostello loop render this claim obvious (because the claim does not prohibit additional iterations of the loop).

190. For the reasons explained in elements [1e] and [7e], the proposed combination discloses or renders obvious **determining if (k) equals (N); when (k) is less than (N), setting (k) equals (k)+(1); and repeating** the *statements* within the loop **until (k) equals (N)**. Although claims 1 and 7 use the variables M and P, respectively, to count the number of loops, where this claim uses "k" (*see* element [12d]), the combination applies equally no matter what name is given to the variable, as I explained in element [12c]. In addition, in the context of this loop, a PHOSITA would have been motivated to include both the **selecting** and **configuring** steps in the loop's *statements* because a user may choose to **select** different effects for each video clip in the playlist, and so that the appropriate link is **configured** and added to the playlist, as discussed in element [12e]. Also, it would have been obvious to use **(k) equals (N)** as the loop's *condition* for the reasons explained in elements [1e] and [7e].

191. Also, to the extent required by the claim, the proposed combination discloses or renders obvious element [12e] even after (k) is incremented and this step is repeated because, as explained in element [11e], it would have been obvious to select, configure, and add *each* effect link.

CLAIM 23

192. Dependent claim 23 is substantially the same as dependent claim 11, except: (1) claim 23 depends from claim 19 (where claim 11 depends from

claim 7), (2) claim 23 is directed to **computer readable program code which causes said programmable computer processor to** perform the method steps recited in claim 11, and (3) claim 23 recites "**(N)**" at the very end of element [23e] (where claim 11 recites "(N)-1" at the very end of element [11e]). (Ex-1013, 9). As a result, it is my opinion that dependent claim 23 is obvious for the reasons that I provided above for independent claim 19 (in Ground 1) and dependent claim 11 (in this Ground).

193. In particular, even though element [23e] recites "**(N)**" in the last clause, this would have been obvious for the reasons I provide above in the discussion of claim [11e]. In particular, it would have been obvious to implement Hedinsson to iterate the loop N times so that the user can select a transition effect to occur following the end of a playlist.

CLAIM 24

194. Dependent claim 24 is substantially the same as dependent claim 12, except: (1) claim 24 depends from claim 19 (where claim 12 depends from claim 7), (2) claim 24 is directed to **computer readable program code which causes said programmable computer processor to** perform the method steps recited in claim 12, (3) the preamble of claim 24 recites "**at least one of a lensing effect and a sound effect**" (where claim 12 recites "at least one of a lensing effect and a second effect"), and (4) claim 24 specifies that "**the composite virtual clip**

includes (N) saved virtual clips" (where claim 12 recites "the composite virtual clip defined by (N) saved virtual clips"). (Ex-1013, 10). As a result, it is my opinion that dependent claim 24 is obvious for the reasons that I provided above for independent claim 19 (in Ground 1) and dependent claim 12 (in this Ground).

195. In claim 24, it is not clear whether "**wherein (i) is between 1 and (N)**" (in element [24e]) should be interpretated as "wherein (k) is between 1 and (N)." If so, then this is satisfied for the reasons explained in connection with element [24e]. If not, then this limitation is still satisfied by the proposed combination because **(i)** can be any number between 1 and (N), as claim 24 includes no other reference to "**(i)**".

196. It is my understanding that a claim's preamble may or may not be limiting, but in any event, the proposed combination renders obvious **at least one of a lensing effect and a sound effect**. For the reasons I explained above in claim 12 (and particularly in element [12a]), Kostello's effects include a **lensing effect**. Kostello also teaches adding "audio" effects to individual segments in substantially the same way as other effects. (Ex-1006, 2:27-36, 7:65-8:10, 8:48-53, Fig. 7, Fig. 9 (step 930)). For example, Kostello states:

> At least one audio effects palette can be displayed in the graphical user interface comprising a plurality of audio effects elements. Each audio effect element can correspond to audio effects to be applied to a segment of the at least one video clip. User-generated input can be received via the graphical user interface selecting at least one thumbnail and at least one audio effects element. With such a

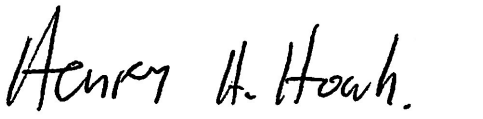
> scenario, the generated or modified file applies audio effects corresponding to the selected at least one audio affects element to the segment corresponding to the selected at least one thumbnail.

(Ex-1006, 2:27-36). In Figure 7, Kostello shows an example interface through with a user can add audio effects, and a user can, for example, adjust a volume slider to set the volume level for the audio. (Ex-1006, 7:65-8:10). Thus, it is my opinion that Kostello teaches adding an effect that includes a sound effect.

197. Hedinsson discloses or renders obvious that **the composite virtual clip** (the media stream and/or its playlist) **includes (N) saved virtual clips** (the number of video clips referenced by the playlist) for the reasons I previously explained in connection with element [12b]. For example, for playlist 240a, **(N)** would equal 3, and the **saved virtual clips** would be the video clips referenced by metadata clips 215a-c. (Ex-1004, 5:37-47, 9:37-53, Fig. 2).

IX. OATH

198. This declaration and my opinions herein are made to the best of my knowledge and understanding, and based on the material available to me, at the time of signing this declaration. I declare under penalty of perjury under the laws of the United States of America that all statements made of my own knowledge are true and that all statements made on information and belief are believed to be true. I understand that willful false statements and the like are punishable by fine or imprisonment, or both (18 U.S.C. 1001) and may jeopardize the validity of the application or any patent issuing thereon.



August 12, 2025
Date

Dr. Henry Houh