

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

CISCO SYSTEMS INC.,

Petitioner

IPR2026-00211

U.S. Patent No. 12,279,116

**DECLARATION OF BRUCE MCNAIR
UNDER 37 C.F.R. § 1.68 IN SUPPORT OF PETITION
FOR *INTER PARTES* REVIEW**

TABLE OF CONTENTS

I.	INTRODUCTION	6
II.	QUALIFICATIONS AND PROFESSIONAL EXPERIENCE	9
III.	LEVEL OF ORDINARY SKILL IN THE ART	13
IV.	RELEVANT LEGAL STANDARDS	14
V.	BACKGROUND	15
A.	Early Machine-Readable Instructions and Compiling	15
B.	Standard Functions and Program Libraries in Common Use	16
C.	Compilation and Linking: From Human-Readable Source Code to Executable Binary	17
D.	Dynamic Linking and Shared Libraries	18
VI.	OVERVIEW OF THE '116 PATENT	21
VII.	CLAIM CONSTRUCTION	23
VIII.	IDENTIFICATION OF HOW THE CLAIMS ARE UNPATENTABLE....	24
A.	Ground 1: Claims 1-8, 11-14, 16-20, 22-27, and 55 would have been Obvious over Abuan.....	25
1.	Summary of Abuan	25
2.	Claim 1	31
3.	Claim 2	52
4.	Claim 3	55
5.	Claim 4	56
6.	Claim 5	57
7.	Claim 6	59
8.	Claim 7	62
9.	Claim 8	64
10.	Claim 11	65
11.	Claim 12	71

12.	Claim 13	73
13.	Claim 14	78
14.	Claim 16	82
15.	Claim 17	84
16.	Claim 18	86
17.	Claim 19	88
18.	Claim 20	88
19.	Claim 22	91
20.	Claim 23	92
21.	Claim 24	93
22.	Claim 25	94
23.	Claim 26	95
24.	Claim 27	96
25.	Independent Claim 55	97
B.	Ground 2: Claim 15 would have been Obvious over Abuan in view of Eisenberg	100
1.	Summary of Eisenberg	100
2.	Reasons to Combine	102
3.	Claim 15	103
C.	Ground 3: Claim 21 would have been Obvious over Abuan in view and Beilis	105
1.	Summary of Beilis	105
2.	Reasons to Combine	107
3.	Claim 21	108
D.	Ground 4: Claims 9-10, 28-41, 43-47, and 49-54 would have been Obvious over Abuan in view of Platt and Guzman.	109
1.	Summary of Guzman	109

2.	Reasons to Combine	110
3.	Claim 9.....	113
4.	Claim 10.....	117
5.	Independent Claim 28.....	119
6.	Claim 29.....	122
7.	Claim 30.....	123
8.	Claim 31.....	123
9.	Claim 32.....	123
10.	Claim 33.....	123
11.	Claim 34.....	123
12.	Claim 35.....	124
13.	Claim 36.....	124
14.	Claim 37.....	124
15.	Claim 38.....	124
16.	Claim 39.....	124
17.	Claim 40.....	125
18.	Claim 41.....	125
19.	Claim 43.....	125
20.	Claim 44.....	125
21.	Claim 45.....	125
22.	Claim 46.....	126
23.	Claim 47.....	126
24.	Claim 49.....	126
25.	Claim 50.....	126
26.	Claim 51.....	126
27.	Claim 52.....	126
28.	Claim 53.....	127

29.	Claim 54.....	127
E.	Ground 5: Claim 42 would have been Obvious over Abuan in view of Platt, Guzman, and Eisenberg.	127
1.	Claim 42.....	127
F.	Ground 6: Claim 48 would have been Obvious over Abuan in view of Platt, Guzman, and Beilis.....	128
1.	Claim 48.....	128
IX.	CONCLUSION.....	129

I, Bruce McNair, do hereby declare as follows:

I. INTRODUCTION

1. I am making this declaration at the request of Cisco Systems, Inc. in the matter of the *Inter Partes* Review of U.S. Patent No. 12,279,116 (“the ’116 patent”) to Chaturvedi et al.

2. I am being compensated for my work in this matter at my standard hourly rate. I am also being reimbursed for reasonable and customary expenses associated with my work and testimony in this proceeding. My compensation is not contingent on the outcome of this matter or the specifics of my testimony.

3. I have been asked to provide my opinions regarding whether the subject matter of claims 1-55 (“the Challenged Claims”) of the ’116 patent would have been obvious to a person having ordinary skill in the art (“POSITA”) at the time of the alleged invention, in light of the prior art. It is my opinion that the Challenged Claims would have been obvious to a POSITA.

4. In the preparation of this declaration, I have studied:

- a. the ’116 patent, Ex.1001;
- b. the prosecution history of the ’116 patent (“’116 File History”), Ex.1002;
- c. U.S. Patent Publication No. 2011/0249077 to Abuan et al. (“Abuan”), Ex.1005;

d. U.S. Patent Publication No. US 2012/0092438 to Guzman Suarez et al. (“Guzman”), Ex.1007;

e. U.S. Patent Publication No. 2010/0066807 to Eisenberg (“Eisenberg”), Ex.1019; and

f. U.S. 2013/0290982 to Beilis et al. (Beilis), Ex.1046.

5. In forming the opinions expressed below, I have considered: the documents listed above; the relevant legal standards, including the standard for obviousness; and my own knowledge and experience based upon my work in the field of network communications and security as described below, as well as portions of the following additional materials:

a. U.S. Patent No. 8,396,490 to Platt et al. (“Platt”), Ex.1006;

b. U.S. Patent No. US 5,689,641 to Ludwig et al. (“Ludwig”), Ex.1008;

c. D. E. Knuth, Selected Papers on Computer Science (Cambridge Univ. Press), 1996, Ex.1009;

d. B. W. Boehm, Software Engineering Economics (Prentice-Hall, Inc.), 1981, Ex.1010;

e. U.S. Patent No. 5,339,431, Ex.1011;

f. U.S. Patent No. 5,615,400, Ex.1012;

g. U.S. Patent No. 5,946,486, Ex.1013;

Declaration of Bruce McNair
Inter Partes Review of U.S. 12,279,116

- h. U.S. Patent No. 6,243,764, Ex.1014;
- i. U.S. Patent No. 5,802,367, Ex.1015;
- j. U.S. Patent No. 5,218,697, Ex.1016;
- k. U.S. Patent No. 5,498,003, Ex.1017;
- l. U.S. Patent No. 6,163,858, Ex.1018;
- m. U.S. Patent Publication No. 2008/0178264, Ex.1021;
- n. U.S. Patent Publication No. 2005/0235044, Ex.1022;
- o. U.S. Patent Publication No. 2006/0037064, Ex.1023;
- p. U.S. Patent Publication No. 2008/0201438, Ex.1024;
- q. U.S. Patent Publication No. 2012/0170722, Ex.1025;
- r. U.S. Patent Publication No. 2005/0265322, Ex.1026;
- s. U.S. Patent No. 6,314,402, Ex.1027;
- t. U.S. Patent Publication No. 2005/0060183, Ex.1028;
- u. U.S. Patent Publication No. 2009/0193394, Ex.1029;
- v. U.S. Patent Publication No. 2008/0022327, Ex.1030;
- w. U.S. Patent Publication No. 2002/0199019, Ex.1031;
- x. U.S. Patent No. 6,101,607, Ex.1032;
- y. U.S. Patent No. 6,311,283, Ex.1033;
- z. U.S. Patent No. 6,295,048, Ex.1034;
- aa. U.S. Patent Publication No. 2009/0310023, Ex.1035;

- bb. U.S. Patent Publication No. 2008/0127183, Ex.1036;
- cc. U.S. Patent No. 7,574,202, Ex.1037;
- dd. U.S. Patent No. 7,461,144, Ex.1038;
- ee. U.S. Patent Publication No. 2007/0185998, Ex.1039;
- ff. U.S. Patent No. 6,820,235, Ex.1040;
- gg. RFC 3264, Ex.1041;
- hh. U.S. Patent No. 9,058,655, Ex.1042;
- ii. Apple Newsroom, “Apple Announces iPhone 2.0 Software Beta,” 2008, Ex.1043;
- jj. The Big Blog, “Skype now available for Android phones,” 2010, Ex.1044;
- kk. U.S. Patent Publication No. 2012/0019610, Ex.1045; and
- ll. iOS Reference Library, “Core Media Framework Reference,” 2010, Ex.1048.

6. Unless otherwise noted, all **emphasis** in any quoted material has been added. Claim terms are *italicized*.

II. QUALIFICATIONS AND PROFESSIONAL EXPERIENCE

7. My complete qualifications and professional experience are described in my *Curriculum Vitae*, a copy of which can be found in Exhibit 1004. The

following is a brief summary of my relevant qualifications and professional experience.

8. I received my Bachelors of Engineering and Masters of Engineering—Electrical Engineering from the Stevens Institute of Technology in 1971 and 1974, respectively. I also completed qualifiers and course work for a Ph.D. in computer science.

9. I have over 55 years of industry experience in the areas of computer systems and electronic circuits, including substantial time spent working with software, security and access control technologies. Before starting high school, I obtained an FCC amateur radio license and commercial radio telephone license and began learning about electronic circuits. While in high school in the 1960s, I was first exposed to computer programming and continued to add to my knowledge in both areas in college. My work experience includes seven years as a GS-0855 Electronic Engineer working for the US Army Electronics Command, a year as a Junior Member of Technical Staff at ITT Defense Communications Division, 24 years as a Member of Technical Staff and Technical Manager at AT&T/Bell Laboratories and 15 years as a full-time faculty member of the Electrical and Computer Engineering Department at Stevens Institute of Technology.

10. While I was working for the US Army Electronics Command at Fort Monmouth, NJ, I was responsible for support of TSEC/KG-27 bulk encryptor, a

classified encryption device used by the Army for protecting multi-channel digital voice communications lines. Also, while I was at Fort Monmouth, my responsibilities included research and development for the cryptographic aspects of SINCGARS, the Single Channel Ground/Airborne Radio System. As part of this work, I wrote computer simulation code for speech processing and system simulations and programmed microcontrollers to create real-time prototypes.

11. During my employment at AT&T/Bell Labs, in the late 1970s and early 1980s, I designed protocols for wide area public data networks, similar to the Internet. I also designed software and hardware for analog modems and secure voice terminals. This work included all of the user interface and real-time hardware driver software for a secure voice terminal and signal processing software to perform real-time speech compression using software development kits (SDKs) and application program interfaces (APIs) for standard microprocessors in the x86 family and various custom digital signal processors (DSPs).

12. From 1985 until 1987, I supervised a group at AT&T Bell Labs involved in the research and design of speech recognition and speaker verification technologies. This work resulted in the design and implementation of one of the earliest cellular phones with speech recognition technology, the AT&T 3050/3450.

13. From 1987 until 1994, I led the AT&T Bell Labs Security and System Reliability Architecture Group. In the course of this activity, I investigated security

in all aspects of AT&T products and services, particularly focused on access control and user authorization techniques. As part of this effort, I also led efforts to employ the newly developed AT&T Smart Card as a personal authenticator for use in the AT&T networks operations centers providing operations support personnel controlled access to the network support terminals. I was also involved in video processing for a planned AT&T service providing personal video conferencing much like today's Zoom, FaceTime, or WebEx as well as early work with General Magic on multiapplication PDAs.

14. From 1994 until 2002, I was a member of the AT&T Bell Labs and AT&T Labs research organization investigating emerging wireless technologies including extensions to the capability of 2G wireless systems and exploration of technology for what we now know as 4G and 5G systems. To perform these investigations, I was responsible for digital, analog and RF hardware design and extensive real-time signal processing software design with a multiprocessor system I created.

15. I am named on multiple domestic and international patents related to wireless systems, geolocation, encryption devices, voice-based security systems, and computer/network security.

16. Since 2002, I have taught undergraduate and graduate courses at Stevens Institute of Technology including courses in digital system design and

computer architecture. Since 2017, I have continued to teach graduate courses in the Stevens on-line graduate program. Between 2002 and 2017, I advised numerous undergraduate design projects including several involving web based systems and computer interfaces and drivers.

17. Also, since 2002, I have consulted on computer and network systems for various technical organizations and for intellectual property matters with technologies similar to those of the '116 patent.

III. LEVEL OF ORDINARY SKILL IN THE ART

18. I understand there are multiple factors relevant to determining the level of ordinary skill in the pertinent art, including (1) the levels of education and experience of persons working in the field at the time of the invention; (2) the sophistication of the technology; (3) the types of problems encountered in the field; and (4) the prior art solutions to those problems.

19. A Person of Ordinary Skill in The Art (“POSITA”) in July 2013 would have had a working knowledge of the software programming art that is pertinent to the '116 Patent, including the use of software modules and application programming interfaces (APIs). A POSITA would have had a bachelor’s degree in computer science, computer engineering, or an equivalent, and three years of professional experience relating to computer programming. Lack of professional experience can be remedied by additional education, and vice versa.

20. For purposes of this Declaration, in general, and unless otherwise noted, my statements and opinions, such as those regarding my own experience and what a POSITA would have understood or known generally (and specifically related to the references I consulted herein), reflect the knowledge that existed in the relevant field as of the priority date of the '116 patent.

IV. RELEVANT LEGAL STANDARDS

21. I am not an attorney. In preparing and expressing my opinions and considering the subject matter of the '116 patent, I am relying on certain basic legal principles that Cisco's counsel has explained to me.

22. I understand that prior art to the '116 patent includes patents and printed publications in the relevant art that predate the priority date of the '116 patent. For purposes of this Declaration, I am applying July 16, 2013, as the priority date of the '116 patent.

23. I have been informed by Cisco's counsel that a claimed invention is unpatentable under 35 U.S.C. § 103 if the differences between the claimed invention and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a POSITA. I have also been informed by Cisco's counsel that the obviousness analysis considers factual inquiries, including the level of ordinary skill in the art, the scope and content of

the prior art, and the differences between the prior art and the claimed subject matter.

24. I have been further informed by Cisco's counsel that there are several recognized rationales for combining references or modifying a reference to show obviousness. These rationales include: (a) combining prior art elements according to known methods to yield predictable results; (b) simple substitution of one known element for another to obtain predictable results; (c) use of a known technique to improve a similar device (method, or product) in the same way; (d) applying a known technique to a known device (method, or product) ready for improvement to yield predictable results; (e) choosing from a finite number of identified, predictable solutions, with a reasonable expectation of success; and (f) some teaching, suggestion, or motivation in the prior art that would have led a POSITA to modify the prior art or to combine prior art teachings to arrive at the claimed invention.

V. BACKGROUND

A. Early Machine-Readable Instructions and Compiling

25. The fundamental concepts of computer science and software programming have been developing as technical fields of study for several decades—even millennia. Ex.1009, 5, 185. One of those fundamental concepts was how to embody instructions for execution by a computer processor. Many of the

earliest iterations of computer-executable instructions included physical media, such as punch cards having patterns of holes representing a sequence of encoded numbers. Computers “read” those sequences and interpreted them as computer instructions. Ex.1009, 227-229. As these systems evolved, the media for writing and reading computer-readable instructions became more sophisticated, including perforated tape, teletype materials, and eventually magnetic media that allowed for more interactive input via a keyboard and a computer display.

26. At the same time, programmers created programs that translated human-readable notations into machine-executable encodings—a transformative process termed “compiling.” *See* Ex.1009, 231 (describing the IT compiler). The advent of compilers established the foundational workflow in which higher-level program representations are transformed into executable instructions. These developments enabled larger and more complex software systems and accelerated the adoption of higher-level languages and tooling around compilation and program construction.

B. Standard Functions and Program Libraries in Common Use

27. With the growth in software complexity, computer engineers turned toward the practice of incorporating reusable functionality into standard libraries and program libraries that could be invoked by application programs. Ex.1010, 654. By at least the 1980s, mainstream languages such as FORTRAN, COBOL,

and Pascal provided standard functions accessible to developers. *See* Ex.1010, 654. For instance, FORTRAN provided a “Scientific Subroutine Package,” which was a set of standard libraries that could be compiled into a FORTRAN program as early as the late 1960s. Ex.1041. More extensive functionality was distributed as “Program Libraries,” which application code would call as subroutines. Developers commonly relied on stable function interfaces (e.g., declarations or prototypes) so that separately written modules could interact predictably. This paradigm—applications invoking well-defined, reusable library routines—quickly became ubiquitous and standard practice for generations of programmers.

C. Compilation and Linking: From Human-Readable Source Code to Executable Binary

28. To support the modular use of libraries, the conventional build process for software was structured as a two-stage pipeline. In the first stage, a compiler translated human-readable source code into machine-readable object code. *See* Ex.1011, Abstract. In the second stage, a linker combined object code from multiple sources—including the application’s own source files and separately provided program libraries—into a complete, stand-alone executable program. *See* Ex.1011 at 1:24–28; Ex.1012 at 1:25–29. In doing so, the linker resolved external references (symbols) and performed relocations so that call sites in one module correctly target routines and data provided by another. *See* Ex.1040, 518-519. This

second stage is commonly referred to as “static linking.” *See* Ex.1012, 1:29; Ex.1013, 1:52–54, 1:66–2:2; Ex.1014, 2:5–17.

29. In static linking, the static-link library is integrated with the client program at link time such that the resulting executable contains the necessary library code and data alongside the application’s own code and data. As contemporaneous references explain, a static-link library operates in the client program’s execution context and has access to the client’s code and data. *See* Ex.1015 at 1:18–27, 1:39–40. The outcome is a self-contained binary that does not require separately invoking the library at runtime. This arrangement reflects a straightforward extension of the fundamental principle that application code can rely on and call functions furnished by other modules during execution.

D. Dynamic Linking and Shared Libraries

30. While static linking made software development more efficient, the growing body of software products that relied on the same or similar software functionalities raised interest in sharable, accessible libraries. Those libraries would share commonly used code across multiple applications so that redundant copies of that code would not have to be included in each executable, as was the case with static linking. Dynamic linking addressed this need by providing dynamic-link libraries (“DLLs”), which could be loaded and invoked by multiple programs at runtime. In contrast to static-link libraries, several client programs can

share a single copy of the code contained in a DLL, thereby conserving memory and reducing duplication. *See* Ex.1015 at 1:27–29.

31. DLLs, like static-link libraries, execute in the context of the invoking application. A DLL can access the client program’s code, data space, and stack space when a function it exposes is called by the client program. *See* Ex.1015 at 1:35–38. Thus, both static and dynamic linking reflect the same core model: an application invokes functions supplied by external libraries, and those functions execute without leaving the calling program’s execution context. This model was implemented across widely used platforms and toolchains, further underscoring its status as routine practice. In this way, static libraries and dynamic libraries respectively offer different advantages and attributes. Static libraries offer simplicity and inherent security, as all the required program code is compiled together. Dynamic libraries allow executable programs to be smaller in size, as a significant portion of the program code is provided by a DLL file. On the other hand, there is additional time overhead for DLL-based applications to find and load any required DLLs.

32. By the relevant time frame, these concepts were well known and widely implemented. Contemporary references describe DLLs as a long-established feature and as specific instances of application programming interfaces available as common resources to applications and other executable code. *See*

Ex.1016, 6:26. They further note that procedures for incorporating new DLLs into applications were well known in the art. *See* Ex.1017, 16:33–34. It was also standard to distinguish the two common forms of linking—static and dynamic. *See* Ex.1018, 6:65–66.

33. Also well-known before the '116 patent's earliest priority date was a modular approach for providing the functions to the calling programs, as noted in the introduction section above. In the modular approach, functionalities, such as video conferencing functionalities, were made accessible to calling programs in the form of self-contained components or modules. Ex.1008, 18:50-52 and FIG. 20 (“Software modules 161-168 communicate with operating system/GUI software 180 and other applications 170 utilizing standard function calls and interapplication protocols,” where the modules include “videophone 169,” “application sharing 166,” “computer-integrated telephony 167,” as shown in Figure 20); Ex.1049, [0054] and [0059]-[0060] (“a video call service device 602 comprises ... a network monitor module 614, a video call settings module 616, and a client management module 618,” where “the video call settings module 616 is invoked by the video call server 612” and “the client management module 618, when invoked, instructs a video call client 506 to reduce a display size used for a video call UI 104”); Ex.1019, Abstract (“a user [] select[s] a third party communication application from the user interface of the core application. The

selected third party communication application is executed to enable the user to initiate a communication session with another user using the services of the selected third party communication application.”); *see also, generally* Ex.1005 and Ex.1006.

34. All of the features and functionality described in this section were well-known for decades before the priority date of the ’116 patent, and a POSITA would have been familiar with them.

VI. OVERVIEW OF THE ’116 PATENT

35. The ’116 patent purports to address “system and method ... for improving functionality in software applications.” Ex.1001, Abstract. For example, “the functionality of [a] superblock application is constrained to some degree by [a] device 100 and by the instructions of the superblock itself.” Ex.1001, 2:67-3:3. The ’116 patent explains that “additional functionality” can be provided to the superblock 104 by a “function block 200 [that] includes instructions for providing the superblock 104 with one or more functions (e.g., capabilities) that are not otherwise possessed by the superblock 104.” Ex.1001, 3:35-37, 4:35-38. “[T]he function block 200 may be provided as a set of instructions that are included in the superblock 104.” Ex.1001, 4:54-56. An example of “the function block 200 may be ... a software developer's kit (SDK).” Ex.1001, 4:56-59. “The function block is

configured to provide functions that are accessible to the superblock application via an application programming interface (API).” Ex.1001, Abstract.

36. The '116 patent describes “compil[ing] or otherwise includ[ing] the function block instructions in the superblock 104 [to] ensure[] that the function block 200 will occupy the same memory space as the superblock 104.” Ex.1001, 4:61-64. Figure 3A of the '116 patent below shows a superblock application incorporating therein a function block where “an application programming interface (API)” “enable[es] the superblock 104 to interact with the function block 200.” Ex.1001, 6:25-27. For instance, “the superblock 104 may make API calls to the function block 200 to access the function block's capabilities. The function block 200 then provides services to the superblock 104 in response to the API calls.” Ex.1001, 6:27-31.

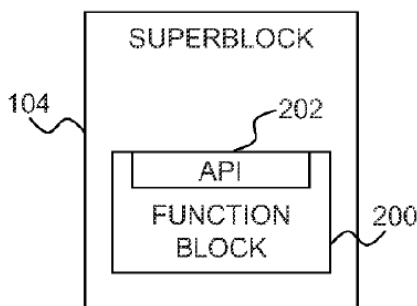


FIG. 3A

Ex.1001, Fig. 3A

37. The '116 patent provides “audio, video, ... [and] conferencing” as example functions provided by the function block. Ex.1001, 4:38-43. For example,

“the function block 200 ... connect[s],” “us[ing] one or more external services 204,” “to another device in order to establish [an audio/video call] session.” Ex.1001, 5:39-42, 33-38. “The external services 204 may be provided via a server.” Ex.1001, 5:30-33. When “provid[ing] ... video capabilities for the superblock 104,” the function block 200 “provid[es] those capabilities within the superblock application,” as illustrated in Figure 2B below. Ex.1001, 4:44-47. The ’116 patent describes “visual functions [being] presented without leaving the superblock application display. In other words, there is no need to switch context ... to any other application in order to access the additional functionality provided by the function block 200.” Ex.1001, 4:50-53.

38. In summary, the ’116 patent merely combines known elements—application programming interfaces (APIs), an application utilizing APIs to call libraries, static-linked libraries with functions that can be executed without switching from the context of a calling application, a server that provides an audio/video session—into one system. This agglomeration of known ideas did not advance human knowledge or understanding, and the ’116 patent claims should not have been allowed to issue.

VII. CLAIM CONSTRUCTION

39. It is my understanding that in order to properly evaluate the ’116 patent, the terms of the claims must first be interpreted. It is my understanding that

for the purposes of this *inter partes* review, the claims are to be construed under the so-called *Phillips* standard, under which claim terms are given their ordinary and customary meaning as would have been understood by a POSITA in light of the specification and prosecution history, unless the inventor has set forth a special meaning for a term. I have also been informed that claim terms only need to be construed to the extent necessary to resolve the obviousness inquiry.

40. I have reviewed the entirety of the '116 patent, as well as its prosecution history. It is my opinion that for purposes of applying the prior art presented herein to evaluate the patentability of the Challenged Claims, no terms require express construction.

VIII. IDENTIFICATION OF HOW THE CLAIMS ARE UNPATENTABLE

41. The discussion in this Declaration provides a detailed analysis of how the asserted prior art references teach each limitation of the Challenged Claims.

42. As part of my analysis, I have considered, and discuss in detail, the scope and content of the prior art and any differences between the alleged invention and the prior art.

43. It is my opinion that the alleged invention recited in the Challenged Claims would have been obvious in view of the teachings of the asserted prior art and the knowledge of a POSITA.

A. Ground 1: Claims 1-8, 11-14, 16-20, 22-27, and 55 would have been Obvious over Abuan.

1. Summary of Abuan

44. U.S. Patent Publication No. 2011/0249077 to Abuan et al. (Ex.1005, “Abuan”) was filed on June 6, 2010, and published on October 13, 2011. Ex.1005, cover at (22), (43). Abuan is titled “Video Conference Network Management for a Mobile Device.” Ex.1005, cover at (54). Abuan was not before the Examiner during prosecution of the ’116 patent.

45. Abuan describes software architectures and associated methods for real-time video conferencing from mobile devices. Abuan’s software architecture organizes various video conferencing functions (e.g., video/audio encoding/decoding, compression/decompression, connection requests) into functional software modules and corresponding sub-modules. Ex.1005, [0043], [0099], [0056]. The modules’ functionalities are accessed via “a set of application program interfaces (APIs).” Ex.1005, [0044]. Figures 13 and 14 of Abuan below show “a video conferencing and processing module 1300” that “includes a client application 1365, a video conference module 1302, a media exchange module 1320, a buffer 1325, a captured image processing unit (CIPU) driver 1330, an encoder driver 1335, and a decoder driver 1340,” as well as “frame buffers 1405 and 1410, audio processing manager 1415, and audio driver 1420.” Ex.1005, [0170], [0183].

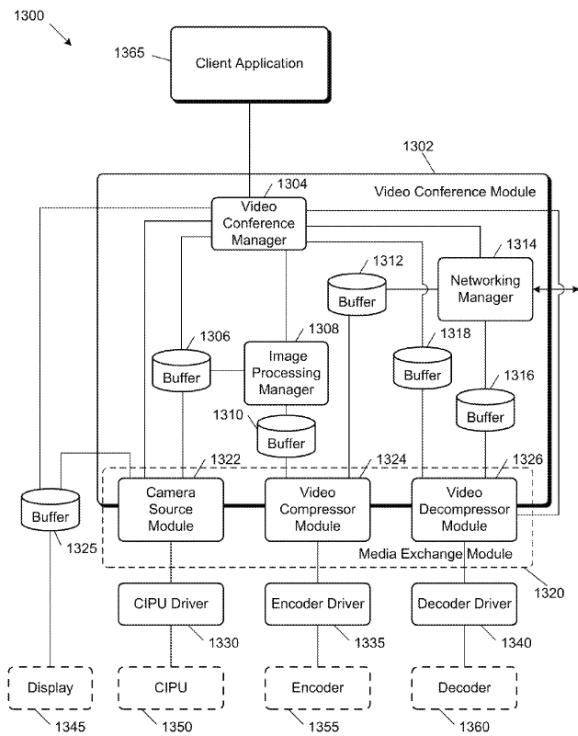


Figure 13

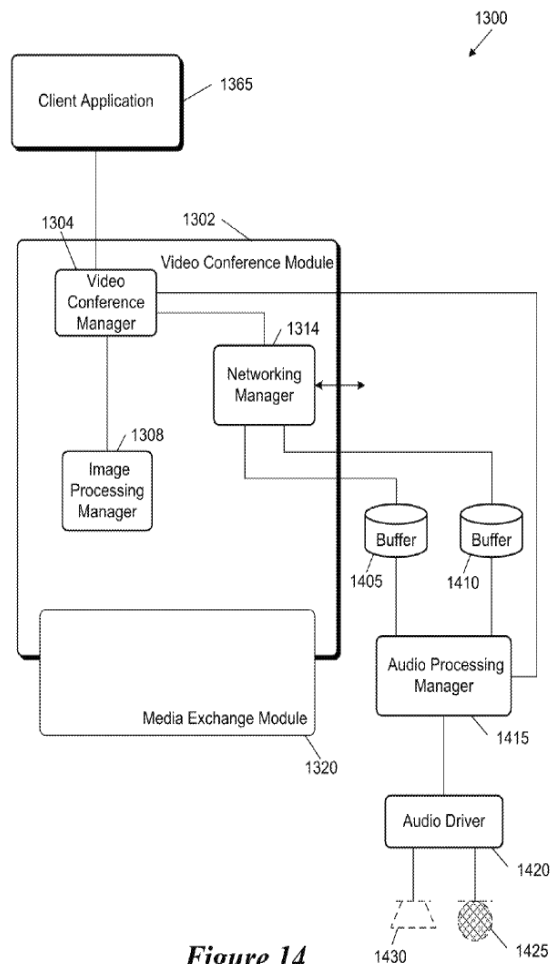


Figure 14

Ex.1005, Figs. 13 and 14

46. Abuan describes “the client application 1365,” which “may be ... implemented as a stand-alone application,” as “an application that uses the video conferencing functions of the video conference module 1302, such as a video conferencing application, a voice-over-IP (VoIP) application (e.g., Skype), or an instant messaging application.” Ex.1005, [0171]. “[T]o start a conference and end a conference,” the client application 1365 “sends instructions to the video conference

module 1302.” Ex.1005, [0172]. “The client application 1365 ... generates user interfaces that are displayed on the dual camera mobile device.” Ex.1005, [0172].

47. Abuan discloses that “the video conference manager 1304 is responsible for initializing some or all of the other modules of the video conference module 1302 (e.g., the image processing manager 1308 and the networking manager 1314) when a video conference is starting.” Ex.1005, [0175]. “The image processing manager 1308 ... processes images captured by the cameras of the ... mobile device” and “the networking manager 1314 manage one or more connections between the dual camera mobile device and the other device participating in the video conference.” Ex.1005, [0178]-[0179].

48. Figure 7 of Abuan shown below illustrates a “video conference request messaging sequence 700 [for establishing a video conference] among a video conference client 710 running on a device 705, a video conference server 715, and a video conference client 725 running on a device 720.” Ex.1005, [0108]. “[T]he video conference clients 710 and 725 are the same as” the client application 1365. Ex.1005, [0108] (“the video conference clients 710 and 725 are the same as the video conference client 645”), [0171] (“the client application 1365 is the same as the video conference client 645”).

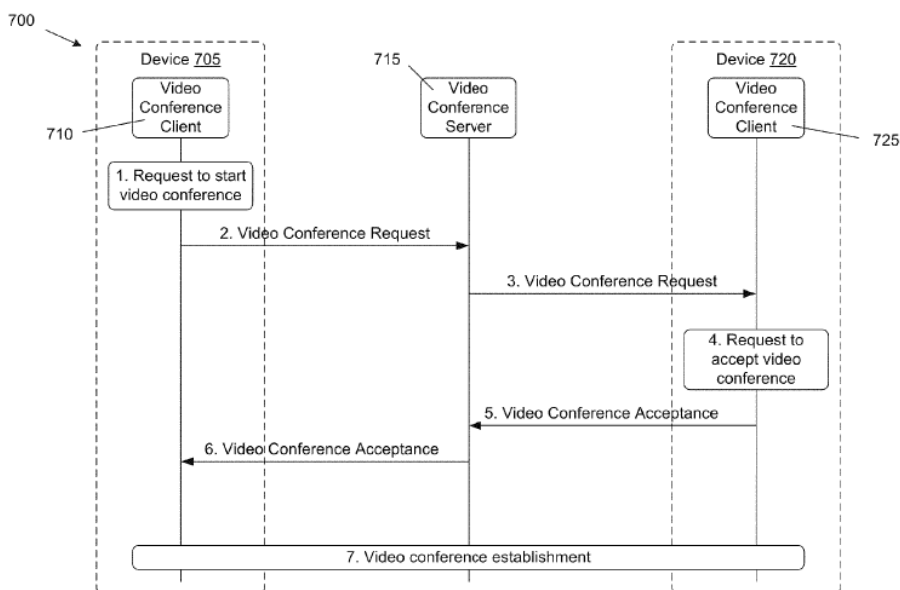


Figure 7

Ex.1005, Fig. 7

49. In Figure 7, “the video conference client 710 receives (at operation 1) a request from a user of the device 705 to start a video conference with the device 720” and in response “sends (at operation 2) a video conference request, which indicates the device 720 as the recipient based on input from the user, to the video conference server 715.” Ex.1005, [0110]-[0111]. “The video conference server 715 ... routes messages among [the] video conference clients,” for example, the video conference server 715 “forwards the video conference request to the video conference client 725” and “forwards (at operation 6) the video conference acceptance [from the video conference client 725] to the video conference client 710.” Ex.1005, [0109], [0111], [0113]. “[A] video conference between the device 705 and the device 720” is then established. Ex.1005, [0114]. The network layer of

the video conference modules “perform[] some or all of the networking functionalities for [the] video conferencing.” Abuan, [0104], [0102] (“The video conference module 625 includes ... a network layer”). Figure 8 “illustrates the start of such a video conference by a ... mobile device 800” and Figure 9 illustrates “the sequence of operations for presenting and accepting a video conference invitation at the remote user's device.” Ex.1005, [0117], [0129].

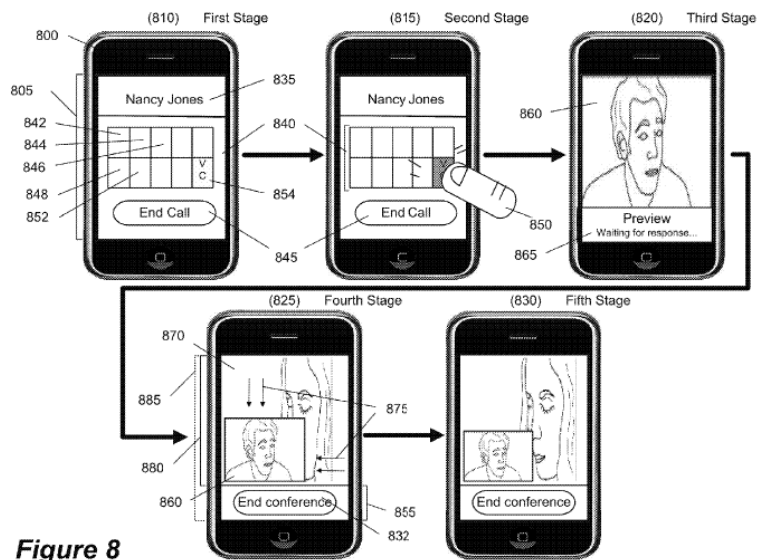


Figure 8

Ex.1005, Fig. 8

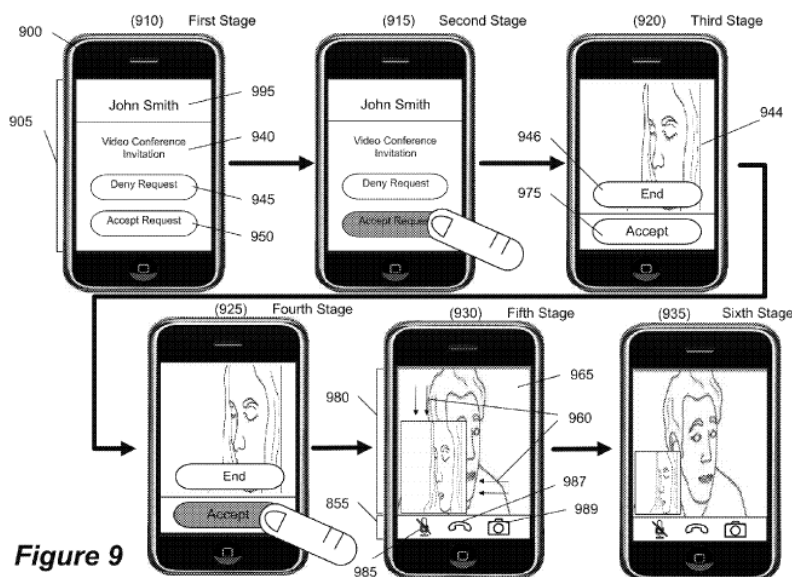


Figure 9

Ex.1005, Fig. 9

50. Abuan is analogous art to the '116 patent. First, Abuan is in the same field of endeavor as the '116 patent, namely, the “manner in which functionality is accessed in certain environments, such as mobile device environments.” Ex.1001, 1:66-67; see Ex.1005, [0001]-[0002] (describing ways that “functionality” is accessed on “portable devices, such as smartphones”). Abuan and the '116 patent also both generally relate to implementing functionality via software. See Ex.1001, Abstract (“providing additional functionality to existing software”); Ex.1005, [0006]-[0007], [0105] (describing how instant messaging or voice-over-IP applications “use the video conferencing functions of the video conference module”). Abuan and the '116 patent both describe at length features for facilitating videoconferencing. Ex.1001, Abstract, 4:43-51 (purpose of a function block is to provide functions including “audio, video,...conferencing, meetings,

and/or other functions”); Ex.1005, [0099] (describing a “video conference module...for performing a variety of video conferencing functions”). Second, Abuan is also reasonably pertinent to a problem allegedly addressed by the ’116 patent. The ’116 patent describes purported problems associated with using multiple applications on a mobile device, potentially requiring a user to “switch[] back and forth between the video window of [a] call and [a] superblock application.” Ex.1001, 4:22-26. The ’116 patent’s proposed solution is to provide functionality—such as communication functions—to a superblock application. Ex.1001, 4:43-61. Abuan similarly describes how communication functions are provided to a client application, which may be a “stand-alone application” or may itself be further “integrated into another application.” Ex.1005, [0105].

2. Claim 1

- a. **[1.0] *A method for providing a real-time communication session over the internet for a superblock application intended for use on a computing device, the method comprising:***

51. To the extent the preamble is limiting, Abuan discloses or renders it obvious.

52. Like the ’116 patent, Abuan illustrates “a method for managing a video conference between a first device and a second device.” Ex.1005, Abstract. More specifically, Abuan’s teachings pertain to the transmission of “captured images to one or more devices **during a real-time communication session**

between the users of” computing devices such as mobile devices. Ex.1005, [0003].

Abuan explains that:

The mobile device of some embodiments (1) can display the captured picture images and video images, (2) can store the captured images for later transmission to another device, (3) can transmit the captured images to one or more devices during **a real-time communication session between the users of the devices**, and (4) can encode the captured images for local storage or for transmission to another device.

One example of **a real-time communication session that involves the transmission of the captured video images is a video conference**. In some embodiments, the mobile device can only transmit one camera's captured video images at any given time during a video conference. In other embodiments, however, the mobile device can transmit captured video images from both of its cameras simultaneously during a video conference or other real-time communication session.

Ex.1005, [0033]-[0034].

53. Figure 7 of Abuan, below, illustrates one example sequence for commencing a video conference between two mobile devices. With reference to Figure 7, Abuan explains that:

FIG. 7 conceptually illustrates an example video conference request messaging sequence 700 of some embodiments. This figure shows the video conference request messaging sequence 700 among a video conference client 710 running on a device 705, a video conference server 715, and a video conference client 725 running on a device 720. In some embodiments, the video conference clients 710 and 725 are the same as the video conference client 645 shown in FIG. 6. As shown in FIG. 7, **one device (i.e., the device 705) requests a video conference and another device (i.e., the device 720) responds to such request.** The dual camera mobile device described in the present application can perform both operations (i.e., make a request and respond to a request).

Ex.1005, [0108].

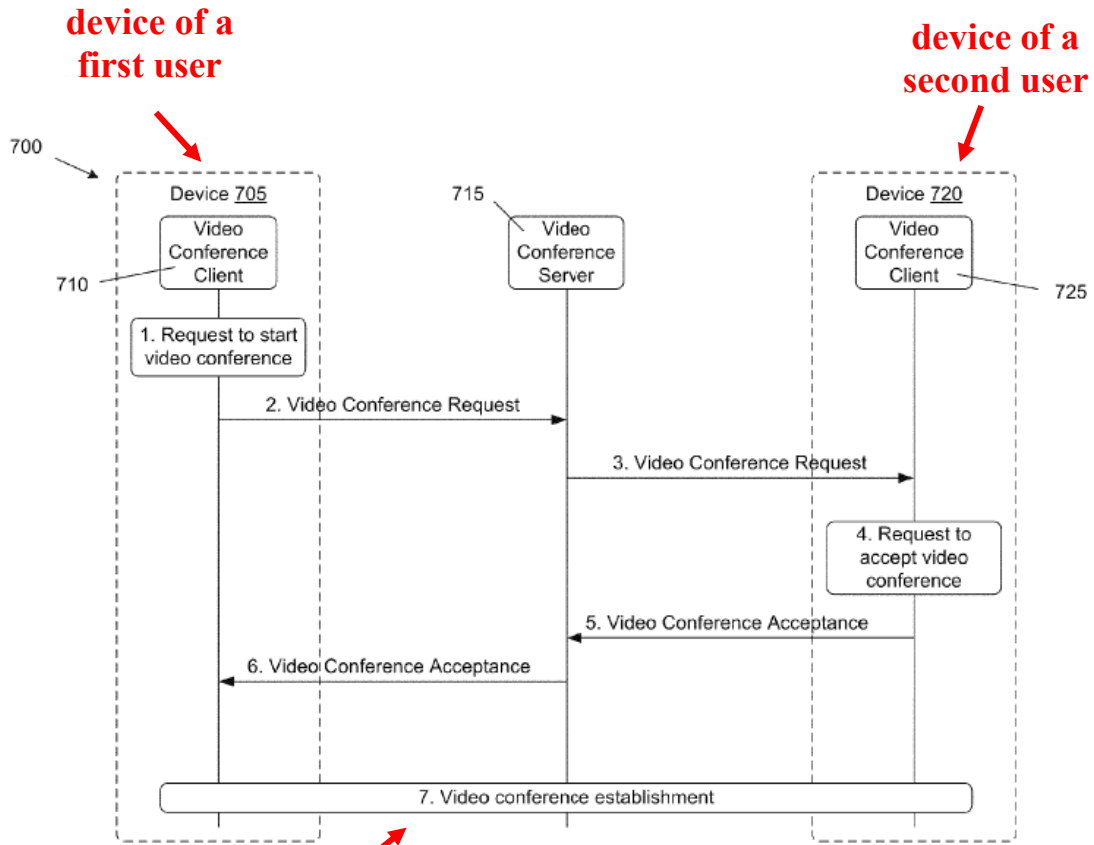


Figure 7

real-time communication session
(video conference)

Ex.1005, Fig. 7.

54. Further, Abuan illustrates a video conference client (*superblock application intended for use on a computing device*) such as video conference client 710 (operating on device 705) and video conference client 725 (operating on device 720).

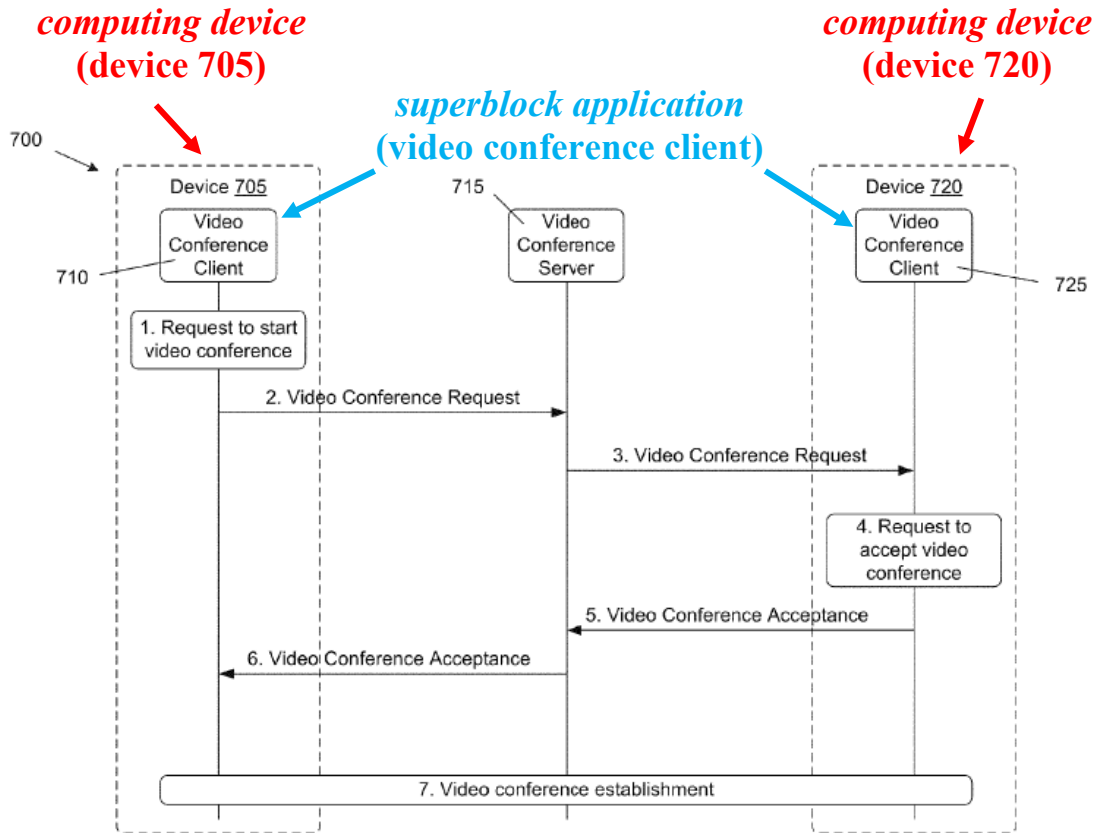


Figure 7

Ex.1005, Fig. 7.

55. Further, Abuan explains that the video conference client for each respective user communicates using a network connection (*over the internet*) such as the Internet:

The video conference server 715 of some embodiments routes messages among video conference clients. While some embodiments implement the video conference server 715 on one computing device, other embodiments

implement the video conference server 715 on multiple computing devices. In some embodiments, the video conference server is a publicly accessible server that can handle and route messages for numerous conferences at once. **Each of the video conference clients 710 and 725 of some embodiments communicates** with the video conference server 715 over a network (e.g., a cellular network, a local area network, a wireless network, a network of networks, **the Internet** etc.) through a network interface such as the network interface 650 described above.

Ex.1005, [0109]. Indeed, from the point of view of each user device, data for a video conference is sent to and received from the other user's device over a network such as the Internet:

In some embodiments, the network interface 650 is a communication interface that allows **the video conference module 625 and the video conference client 645 to send data and receive data over a network** (e.g., a cellular network, a local area network, a wireless network, a network of networks, **the Internet**, etc.) through the network interface 650. For instance, if the video conference module 625 wants to **send data (e.g., images captured by cameras of the dual camera mobile device) to another device on the Internet**, the video conference module 625 sends the images to the other device through the network interface 650.

Ex.1005, [0106].

56. In sum, and as detailed further in the analysis of each element of claim 1 below, Abuan discloses or renders obvious *providing a real-time communication session* (establishing a real-time communication session such as a video conference) *over the internet* (communicating via the Internet) *for a superblock application intended for use on a computing device* (video conference client 710 (operating on device 705) and video conference client 725 (operating on device 720)).

b. **[1.1] *providing a function block for use in adding additional functionality to a third party superblock application that has its own functionality and display window,***

57. Abuan discloses or at least renders obvious this limitation.

58. Abuan describes a video conferencing and processing module 1300 that includes a client application 1365 and various modules and components, including a video conference module 1302, that provide services to the client application. Ex.1005, [0170], [0183]. The client application is a “video conference client...that uses the video conferencing functions of the video conference module 1302.” Ex.1005, [0170]-[0171]. Abuan’s “applications are implemented as software processes that are specified as a set of instructions recorded on a computer readable storage medium,” such as the memory of Abuan’s mobile device. Ex.1005, [0233]. The modules are comprised of software instructions, such

as video conferencing instructions 1986, that are stored in memory. Ex.1005, [0233], [0254], Fig. 19; see also [0254] (explaining how an “API-implementing component” may be a “module”); Ex.1045, [0032] (“programmable elements, often referred to as modules” “may include a program, a subroutine, a portion of a program, or a software component”). And Abuan contemplates that the client application 1365 is a third party application, such as Skype. Ex.1005, [0171]. Thus, the video conferencing and processing module 1300 (including the client application 1365 and the various modules) corresponds to the claimed “*third party superblock application.*”

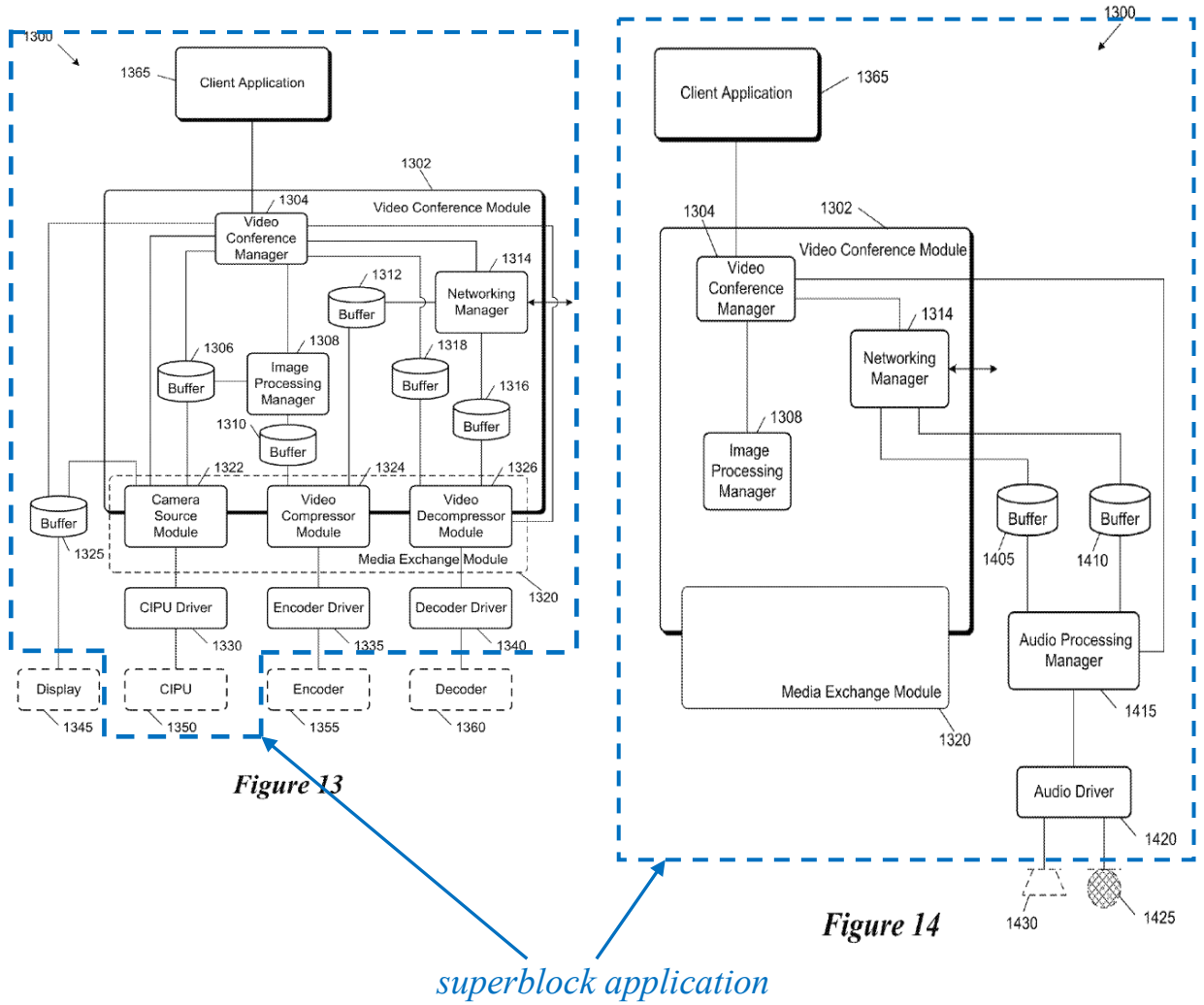


Figure 13

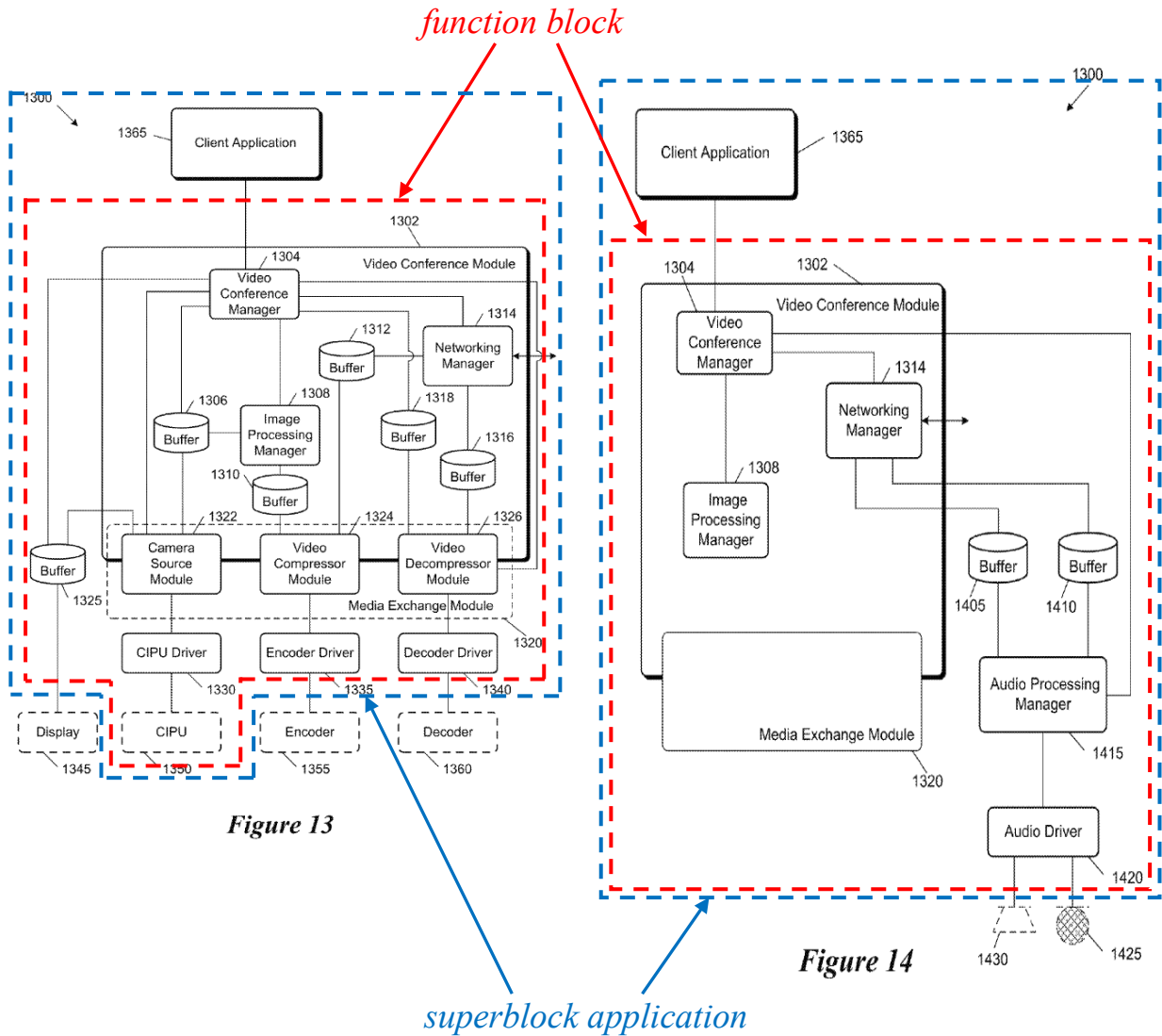
Figure 14

superblock application

Ex.1005, Figs. 13 and 14 (annotated).

59. Abuan’s various modules and components implementing functions provided to the client application 1365 (e.g., the video conference module 1302) correspond to the claimed “*function block*.” See, e.g., Ex.1005, [0056], [0099], [0170], [0183]. In other words, the video conferencing and processing module 1300, but excluding client application 1365, corresponds to the claimed “*function block*.”

60. The superblock application and function block are annotated in Abuan's Figures 13 and 14 below. Consistent with the '116 patent specification's examples, the function block is part of the superblock application. See, e.g., Ex.1001, Fig.3B.



Ex.1005, Figs. 13 and 14 (annotated).

61. Abuan explains that “client application 1365 [the superblock application]...uses the video conferencing functions [functionality] of the video conference module 1302 [the function block].” Ex.1005, [0171]. Further, Abuan describes the application employing the videoconferencing module and other modular components as “a stand-alone application,” such that it *has its own stand-alone functionality and display window*. Ex.1005, [0105], Figs. 8-11 (illustrating example user interfaces for videoconference functionality with display windows). A POSITA would have understood Abuan’s reference to a “stand-alone application” to mean that all the functionality of the application, including its module subcomponents (e.g., *the function block* functionality) is compiled into a single, unitary application. *See* Ex.1012, 1:26-29 (“Traditionally, an application's source files are compiled in object modules and then linked together with other object modules, generically called libraries, to form a complete stand-alone application.”). Thus, it would have been obvious to a POSITA for *the superblock applications* (Abuan’s client application operating on a users’ device) to include the module subcomponents provided as *a function block* and performed within the execution context of Abuan’s client application and not some other application.

62. For example, Abuan describes how an application accesses the functionality of the other software modules via defined application programming interfaces (APIs). Ex.1005, [0235]-[0237]. This structure puts functionality in the

API-implementing component (e.g., the function block) and not in the API-calling component (e.g., the superblock application):

An API allows a developer of an API-calling component (which may be a third party developer) to **leverage specified features provided by an API-implementing component**.

Ex.1005, [0239].

It will be appreciated that the **API-implementing component 1810 may include additional functions, methods, classes, data structures, and/or other features that are not specified through the API 1820 and are not available to the API-calling component 1830**.

Ex.1005, [0247].

63. The use of APIs to define standard, programmatic ways for an application to access functionality of a software module was well-known and commonly used. Ex.1029, [0087] (“APIs are well-known in the art”); Ex.1030, [0038] (“One of the primary purposes of an API is to provide a set of commonly-used functions”); Ex.1031, [0040] (“A well-known API”); Ex.1032, 5:32-48 (“An API is a well-known programming device.... [C]alls to API functions 60 are included in application program 56; the programmer is relieved of any task of having to incorporate or integrate the code itself of functions 60 in application program 56”).

64. Thus, Abuan renders obvious *providing a function block for use in adding additional functionality to a third party superblock application that has its own functionality and display window.*

c. **[1.2] wherein the function block is configured to be compiled into the superblock application and is configured to add the additional functionality to provide the real-time communication session using one or more servers connected over the internet, and**

65. Abuan renders obvious this limitation.

66. As discussed above in [1.1], Abuan describes the application employing the videoconferencing module and other modular components as “a stand-alone application.” Ex.1005, [0105]. A POSITA would have understood Abuan’s reference to a “stand-alone application” to mean that all the functionality of the application, including its module subcomponents (e.g., *the function block functionality*) is of contained in a single, unitary application. See Ex.1012, 1:26-29 (“Traditionally, an application's source files are compiled in object modules and then linked together with other object modules, generically called libraries, to form a complete stand-alone application.”). Thus, it would have been obvious to a POSITA that Abuan’s client application (*the superblock application*) includes the module subcomponents as *the function block configured to be compiled* therein and performed within the execution context of Abuan’s client application and not some other application. And Abuan explains that the API functions or methods discussed

above in [1.1] are compiled into stand-alone applications. Ex.1005, [0239] (explaining that an “API can be specified in terms of a programming language that can be interpreted or compiled when an application is built”).

67. Further, Abuan explains that its computing devices, such as device 705 or device 720, include a networking manager as part of the video conference module (*function block*) reflected in Figure 13 and shown in further detail in Figure 17. Abuan explains that “the networking manager 1700 manages network connections (e.g., connection establishment, connection monitoring, connection adjustments, connection tear down, etc.) between a dual camera mobile device on which it operates and a remote device in a video conference.” Ex.1005, [0208].

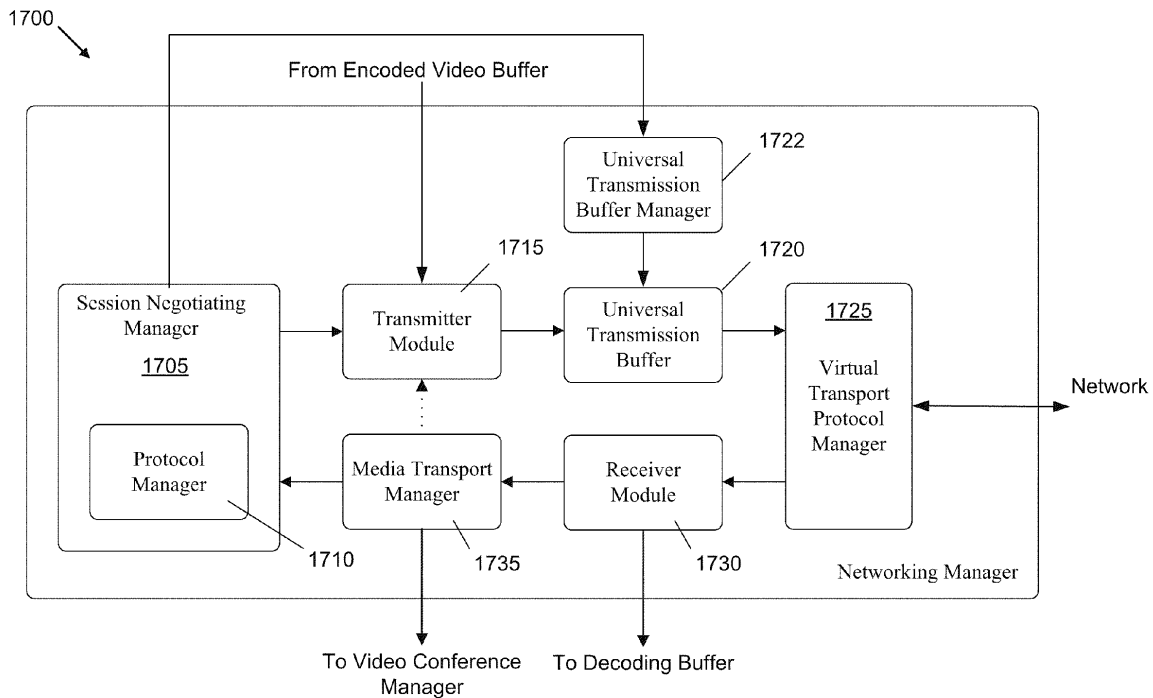


Figure 17

Ex.1005, Fig. 17.

68. As reflected in Figure 17, above, the networking manager 1700 “includes a session negotiating manager 1705, a transmitter module 1715, a universal transmission buffer 1720, a universal transmission buffer manager 1722, a virtual transport protocol (VTP) manager 1725, a receiver module 1730, and a media transport manager 1735.” Ex.1005, [0209]. These elements of the video conference manager (*function block*) are *configured to add the additional functionality to provide the real-time communication session using one or more servers connected over the internet* because they allow for the establishment and management of a

video conference. Ex.1005, [0208]. And as reflected in Figure 7, below, Abuan explains that a video conference is established between a first user and a second user using an intermediary video conference server:

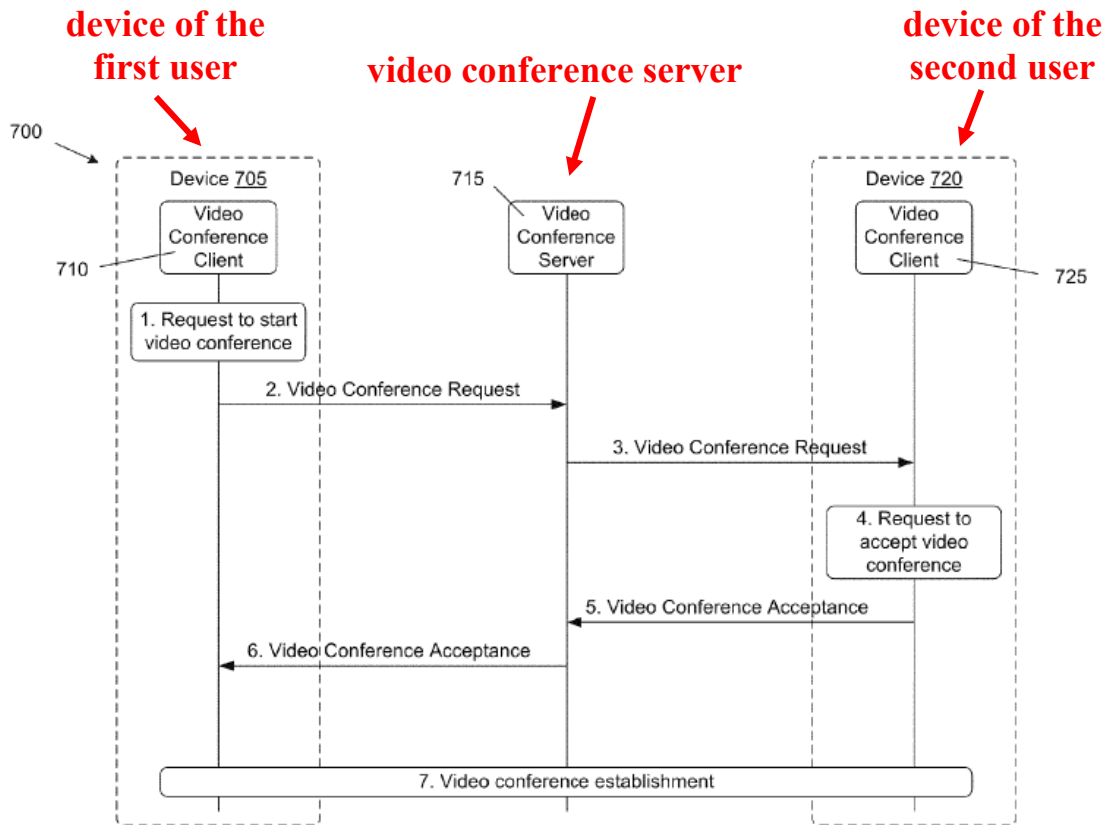


Figure 7

Ex.1005, Fig. 7.

The video conference server 715 of some embodiments routes messages among video conference clients. While some embodiments implement the video conference server 715 on one computing device, other embodiments implement the video conference server 715 on multiple

computing devices. In some embodiments, the video conference server is a publicly accessible server that can handle and route messages for numerous conferences at once. **Each of the video conference clients 710 and 725 of some embodiments communicates with the video conference server 715** over a network (e.g., a cellular network, a local area network, a wireless network, a network of networks, the Internet etc.) through a network interface such as the network interface 650 described above. The video conference server 715 of some embodiments routes messages among video conference clients. While some embodiments implement the video conference server 715 on one computing device, other embodiments implement the video conference server 715 on multiple computing devices. In some embodiments, the video conference server is a publicly accessible server that can handle and route messages for numerous conferences at once. **Each of the video conference clients 710 and 725 of some embodiments communicates with the video conference server 715 over a network** (e.g., a cellular network, a local area network, a wireless network, a network of networks, the Internet etc.) **through a network interface** such as the network interface 650 described above.

Ex.1005, [0109].

69. Accordingly, because Abuan's video conference manager (*function block*) provides the networking manager 1700 which in turn manages communications to the video conference server (*configured to add the additional functionality to provide the real-time communication session using one or more servers connected over the internet*), Abuan discloses or renders obvious this limitation.

d. **[1.3] wherein the function block is configured to interact with the superblock application through one or more application programming interface (API) calls; and**

70. Abuan discloses or at least renders obvious this limitation.

71. As explained above at [1.1], Abuan's video conferencing and processing module 1300 (including the client application 1365 and the various modules) corresponds to the claimed "*superblock application*."

72. As explained above at [1.1]-[1.2], Abuan's various modules and components implementing functions provided to the client application 1365 (e.g., the video conference module 1302) correspond to the claimed "*function block*."

73. And as detailed in [1.1]-[1.2], Abuan describes how an application accesses the functionality of the other software modules via defined *application programming interface (APIs) calls*. Ex.1005, [0235]-[0237]. Indeed, the use of APIs to define standard, programmatic ways for an application to access functionality of a software module was well-known and commonly used. Ex.1029,

[0087] (“APIs are well-known in the art”); Ex.1030, [0038] (“One of the primary purposes of an API is to provide a set of commonly-used functions”); Ex.1031, [0040] (“A well-known API”); Ex.1032, 5:32-48 (“An API is a well-known programming device.... [C]alls to API functions 60 are included in application program 56; the programmer is relieved of any task of having to incorporate or integrate the code itself of functions 60 in application program 56”). Relatedly, a POSITA would have been familiar with device drivers for output devices and would have recognized that they commonly included APIs for reporting on an output device’s capabilities. Ex.1033, 6:30-32 (using “software (e.g., the device drivers) to determine the capabilities” of hardware resources); Ex.1034, 11:8-26 (describing device drivers determining a “display device’s maximum resolution”); Ex.1035, Abstract (“determine capabilities of graphics hardware from a device driver”), [0004] (“provid[ing] a set of application programming interfaces (APIs) to ascertain device contexts and determine capabilities of graphics hardware from a device driver”); Ex.1036, [0033] (“device capabilities...may be exposed through one or more APIs”).

74. Thus, Abuan discloses or renders obvious this limitation.

- e. **[1.4] *enabling establishment of the real-time communication session between the one or more servers and the function block compiled into the superblock application so that the function block can provide the***

real-time communication session to the superblock application.

75. Abuan discloses or renders obvious this limitation.

76. Abuan explains, with reference to Figure 7 above below, that a “video conference server 715 forwards (at operation 3) [a] video conference request [from device 705] to the video conference client 725 of the device 720” to establish a video conference between device 705 and device 720 (*enabling establishment of the real-time communication session*). Ex.1005, [0110]-[0111]. This communication between the video conference client 725—operating on device 720—and the video conference server 715, is performed by the video conference module 1302 on that device (*between the one or more servers and the function block compiled into the superblock application*). More specifically, Abuan discloses that “[e]ach of the video conference clients 710 and 725...communicates with the video conference server 715 over a network (e.g., a cellular network, a local area network, a wireless network, a network of networks, the Internet etc.) through a network interface such as the network interface 650.” Ex.1005, [0109]. Abuan explains that the networking manager 1314 of the video conference module 1302 (or network layer 640 of the video conference module 625) “establishes the connections between the dual camera mobile device and the other device of the video conference at the start of the video conference.” Ex.1005, [0179]; *see also* [0104]; Ex.1005, [0156] (“A more detailed version of this video conference module [625] will be described below by reference

to FIG. 13.”), [0171] (“the client application 1365 is the same as the video conference client 645 of FIG. 6.”). In this way, the video conferencing module 1302 (*the function block*) establishes the video conference session (*can provide the real-time communication session*) between the video conference client of device 705 (*the superbblock application*) and device 720.

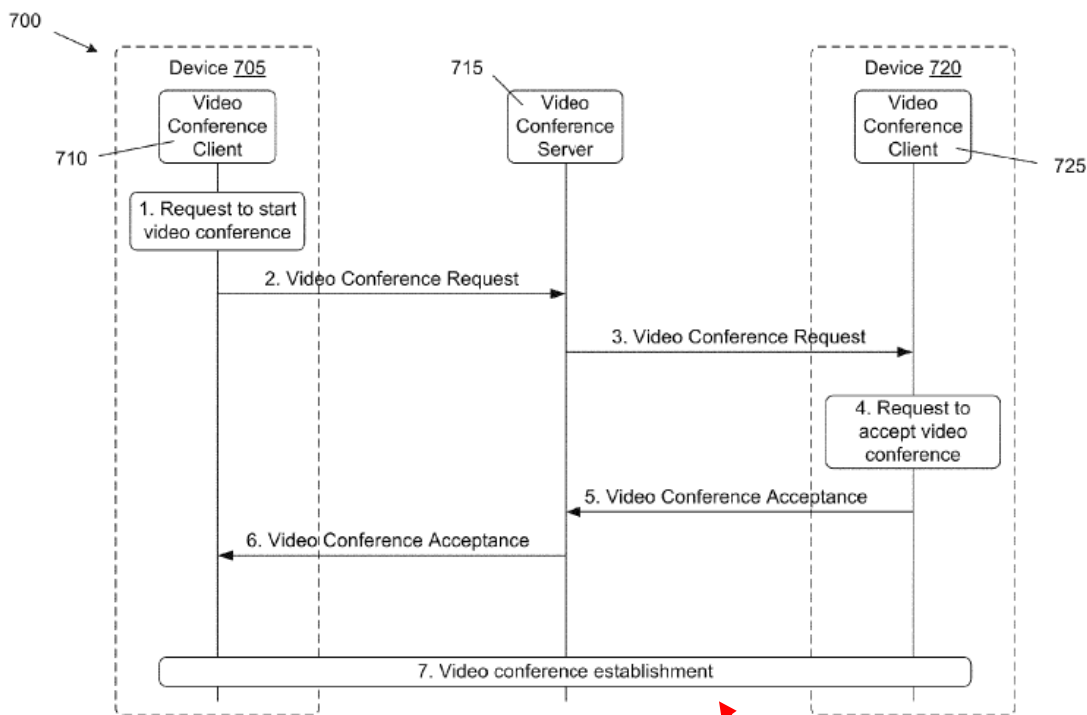


Figure 7

establishment of the real-time communication session

Ex.1005, Fig. 7.

77. Thus, Abuan renders obvious this limitation.

3. Claim 2

- a. **[2.0]** *The method of claim 1 wherein enabling the establishment of the real-time communication session includes signaling communications between the one or more servers and the function block.*

78. Abuan renders obvious this claim.

79. Abuan explains that its computing devices, such as device 705 or device 720, include a networking manager as part of the video conference module (*function block*) reflected in Figure 13 and shown in further detail in Figure 17. Abuan explains that “the networking manager 1700 manages network connections (e.g., connection establishment, connection monitoring, connection adjustments, connection tear down, etc.) between a dual camera mobile device on which it operates and a remote device in a video conference.” Ex.1005, [0208].

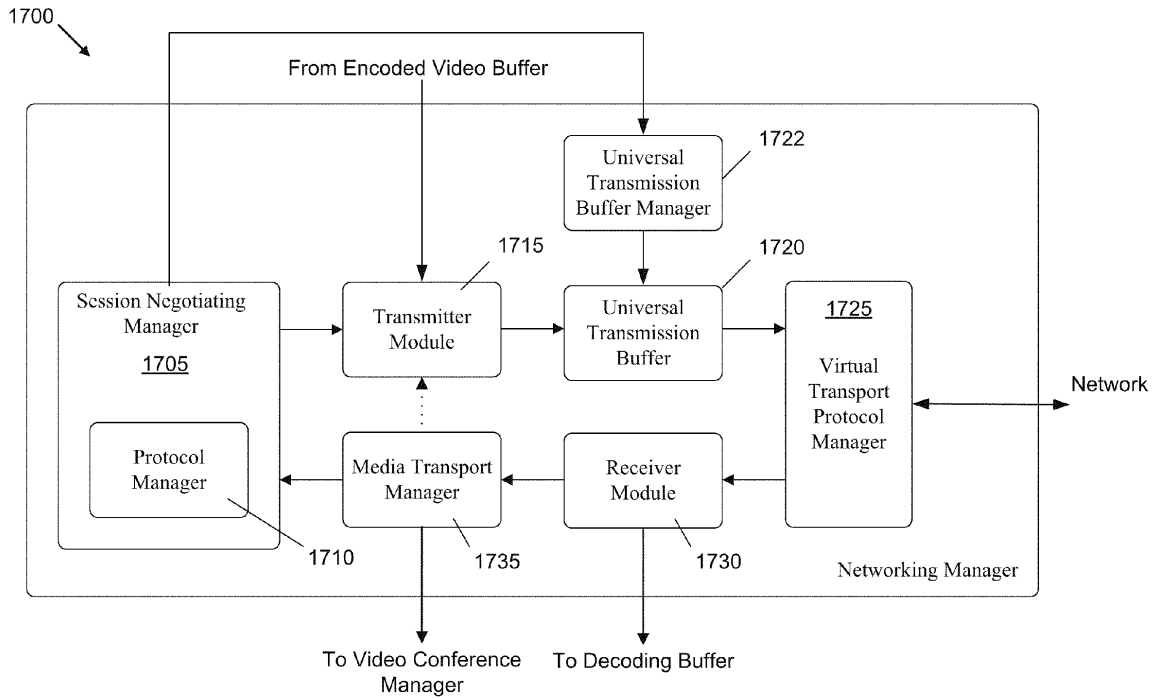


Figure 17

Ex.1005, Fig. 17.

80. As discussed above in claim 1, Abuan explains that a video conference is established between a first user and a second user via messages (*signaling communications*) exchanged with an intermediary video conference server:

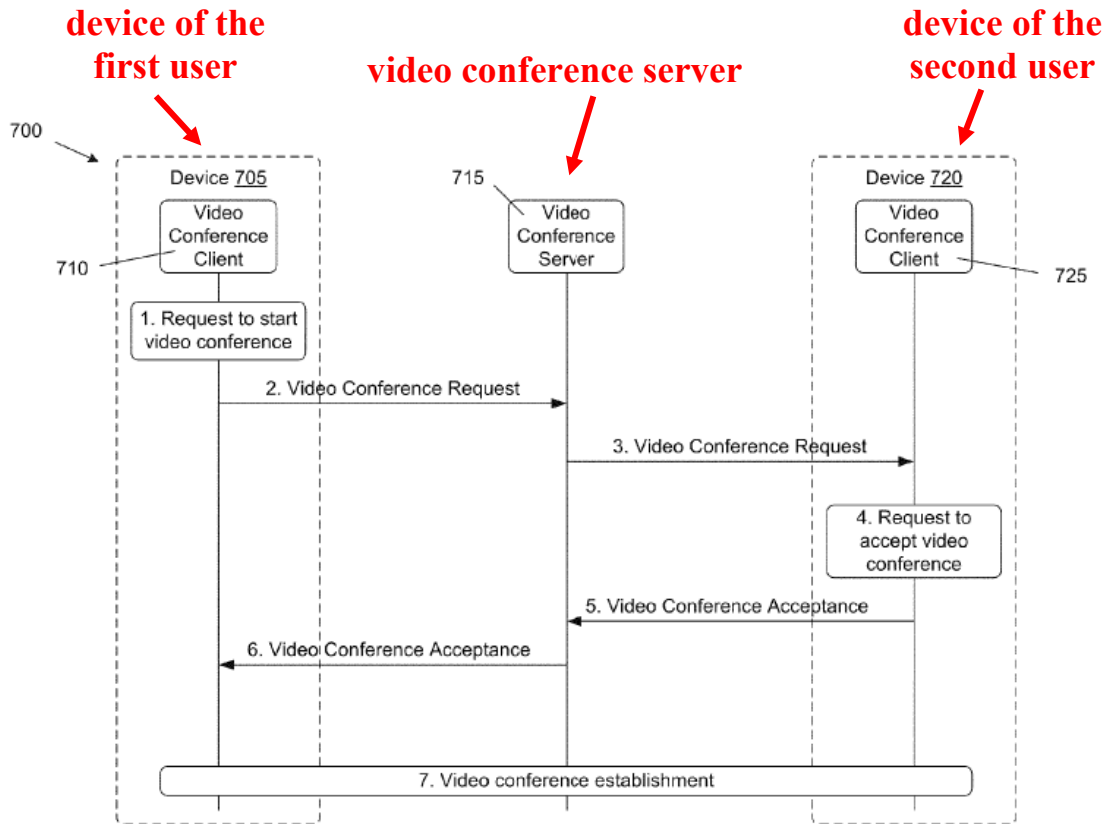


Figure 7

Ex.1005, Fig. 7.

81. More specifically, as discussed above in [1.2], Abuan teaches that *enabling the establishment of the real-time communication session includes signaling communications between the one or more servers and the function block* because “[e]ach of the video conference clients 710 and 725 of some embodiments communicates with the video conference server 715 over a network[.]” Ex.1005, [0109]. Abuan explains that the networking manager 1314 of the video conference module 1302 (or network layer 640 of the video conference module 625, *the function*

block) “establishes the connections between the dual camera mobile device and the other device of the video conference at the start of the video conference” via the intermediary video conference server 715 (*the one or more servers*). Ex.1005, [0179]; *see also* [0104].

82. Thus, Abuan renders obvious this claim.

4. **Claim 3**

a. **[3.0] *The method of claim 2 wherein the signaling communications include session setup, management, and teardown.***

83. Abuan renders obvious this claim.

84. As discussed above for claim 2, Abuan explains that its computing devices include a networking manager 1700 which manages the *signaling communications*. These *signaling communications include session setup, management, and teardown* because, as Abuan elaborates, “the networking manager 1700 **manages** network connections (e.g., **connection establishment**, connection monitoring, connection adjustments, **connection tear down**, etc.) between a dual camera mobile device on which it operates and a remote device in a video conference.” Ex.1005, [0208].

85. Thus, Abuan renders obvious this claim.

5. Claim 4

- a. [4.0] *The method of claim 2 wherein the signaling communications use Session Initiation Protocol (SIP) as a signaling protocol.*

86. Abuan renders obvious this claim.

87. As discussed above for claim 2, Abuan explains that its computing devices include a networking manager 1700 which manages the *signaling communications*. Abuan further explains that the networking manager 1700 uses *a signaling protocol*, such as *Session Initiation Protocol (SIP)*, for its signaling communications:

The session negotiating manager 1705 is responsible for establishing connections between the dual camera mobile device and one or more remote devices participating in the video conference, as well as tearing down these connections after the conference. In some embodiments, the session negotiating manager 1705 is also responsible for establishing multimedia communication sessions (e.g., to transmit and receive video and/or audio streams) between the dual camera mobile device and the remote devices in the video conference (e.g., using **a session initiation protocol (SIP)**).

Ex.1005, [0211].

88. Thus, Abuan renders obvious this claim.

6. Claim 5

- a. [5.0] *The method of claim 1 wherein the real-time communication session uses Real-time Transport Protocol (RTP) as a data transport protocol.***

89. Abuan renders obvious this limitation.

90. As discussed above for claim 2, Abuan explains that its computing devices include a networking manager 1700. As reflected in Figure 17, below, the networking manager 1700 “includes a session negotiating manager 1705, a transmitter module 1715, a universal transmission buffer 1720, a universal transmission buffer manager 1722, a virtual transport protocol (VTP) manager 1725, a receiver module 1730, and a media transport manager 1735.” Ex.1005, [0209].

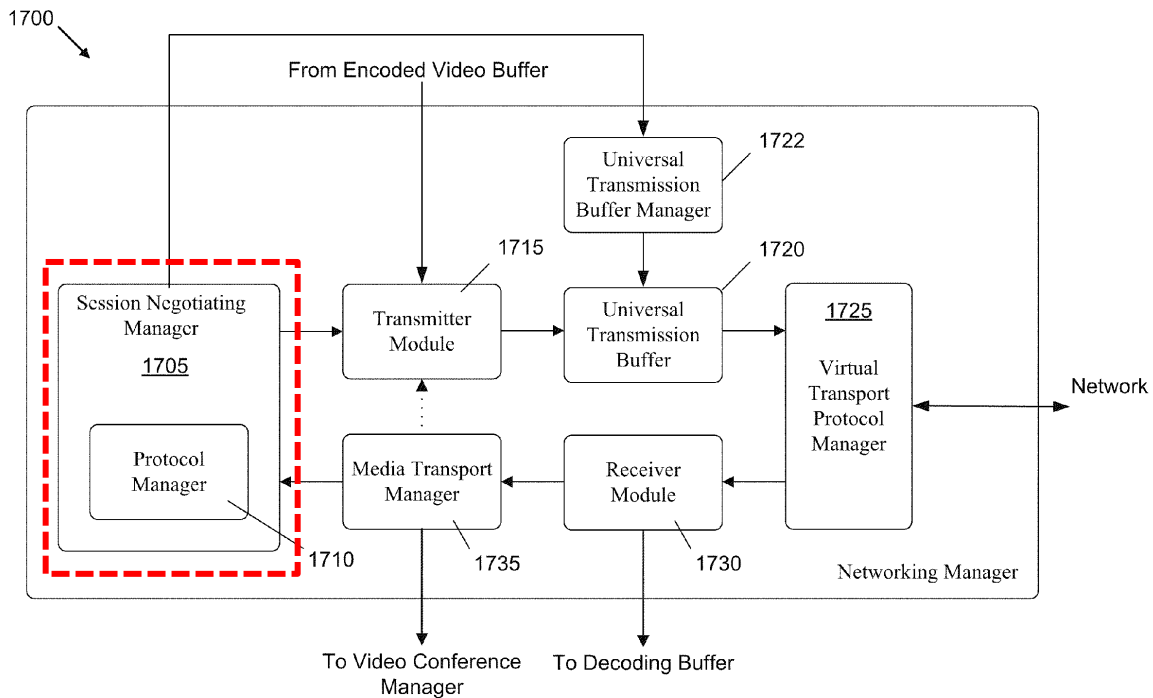


Figure 17

Ex.1005, Fig. 17.

91. Abuan’s networking manager 1700 includes a session negotiating manager 1705, which “is responsible for establishing connections between the dual camera mobile device and one or more remote devices participating in the video conference, as well as tearing down these connections after the conference.”

Ex.1005, [0211]. And the session negotiating manager 1705 further “includes a protocol manager 1710” that “ensures that the transmitter module 1715 uses a correct communication protocol to transmit data to a remote device during the video conference and enforces rules of the communication protocol that is used.”

Ex.1005, [0210]. Abuan explains that a number of communication protocols are supported by the protocol manager 1710, including “a real-time transport protocol (RTP),” rendering obvious that *the real-time communication session uses Real-time Transport Protocol (RTP) as a data transport protocol*. Ex.1005, [0210].

92. Thus, Abuan renders obvious this claim.

7. **Claim 6**

- a. **[6.0] *The method of claim 1 wherein enabling the establishment of the real-time communication session includes negotiating signaling and media parameters between the one or more servers and the function block.***

93. Abuan renders obvious this claim.

94. Abuan explains that after a user has accepted a video conference request, video conference establishment includes negotiating the connection:

Upon receiving the video conference acceptance, some embodiments establish (at operation 7) a video conference between the device 705 and the device 720. Different embodiments establish the video conference differently. For example, **the video conference establishment** of some embodiments **includes negotiating a connection** between the device 705 and the device 720, determining a bit rate at which to encode video, and exchanging video between the device 705 and the device 720.

Ex.1005, [0114].

95. More specifically, Abuan explains that, with reference to Figure 17, the networking manager 1700 (part of the *function block*) includes a session negotiating manager 1705, which is “is responsible for establishing connections between the dual camera mobile device and one or more remote devices participating in the video conference, as well as tearing down these connections after the conference.” Ex.1005, [0211]. Abuan’s session negotiating manager 1705 also “includes a protocol manager 1710” that “ensures that the transmitter module 1715 uses a correct communication protocol to transmit data to a remote device during the video conference and enforces rules of the communication protocol that is used.” Ex.1005, [0210]. And “[t]he manner in which the encoded images are created and sent to the transmitter module 1715 can be based on instructions or data received from...the session negotiating manager 1705.” Ex.1005, [0213].

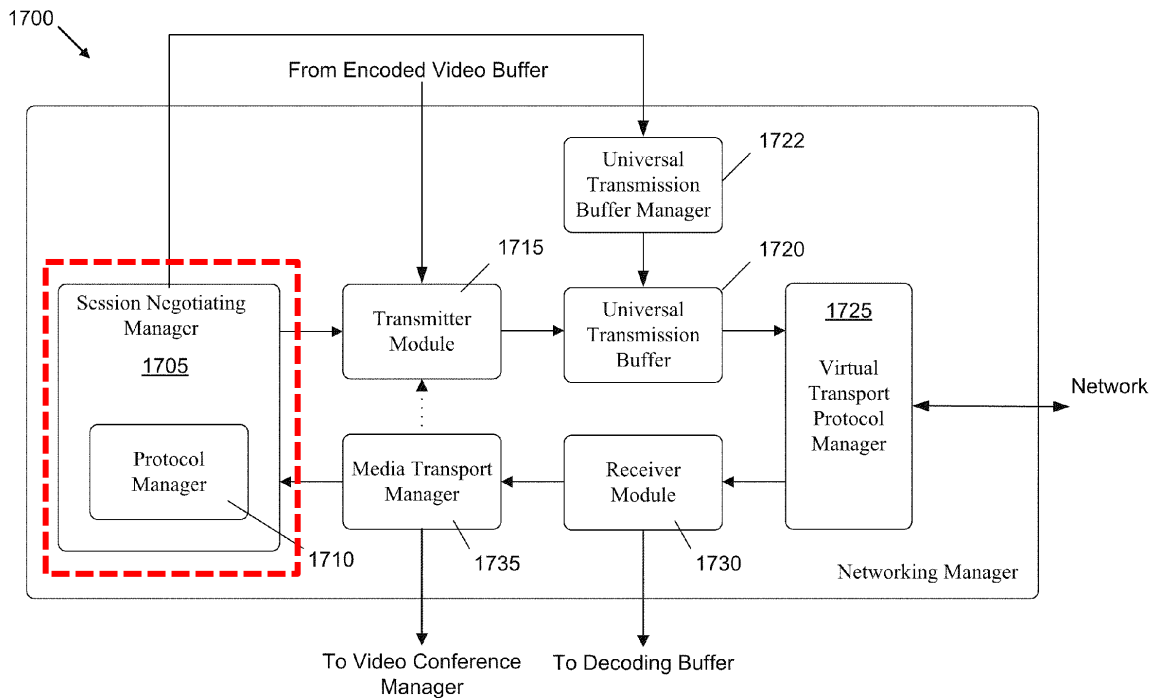


Figure 17

Ex.1005, Fig. 17.

96. Indeed, it was well-known that such parameter negotiation, e.g., “using a session initiation protocol (SIP)” as contemplated by Abuan. Ex.1005, [0211]. For instance, RFC3264 explains an “offer/answer model [] used by protocols like the Session Initiation Protocol (SIP)” to allow two entities to “arrive at a common view of a multimedia session between them,” including codecs to be used. Ex.1041, Abstract. In such implementation, “one participant in the session generates an SDP message that constitutes the offer - the set of media streams and codecs the offerer wishes to use, along with the IP addresses and ports the offerer

would like to use to receive the media. The offer is conveyed to the other participant, called the answerer.” Ex.1041, 1-2. “The answerer generates an answer, which is an SDP message that responds to the offer provided by the offerer. The answer has a matching media stream for each stream in the offer, indicating whether the stream is accepted or not, along with the codecs that will be used and the IP addresses and ports that the answerer wants to use to receive media.” Ex.1041, 2.

97. Accordingly, a POSITA would have recognized that in negotiating a connection between two user devices and video conference server 715 (*one or more servers*) to establish a video conference (*enabling the establishment of the real-time communication session*), Abuan’s session negotiating manager 1705 (part of *the function block*) ensures the correct communication protocol is used (*negotiating signaling...parameters*) and how encoded images are created (*negotiating...media parameters*).

98. Thus, Abuan renders obvious this claim.

8. Claim 7

a. [7.0] *The method of claim 6 wherein the signaling and media parameters include a bandwidth parameter.*

99. Abuan renders obvious this claim.

100. Abuan explains that the session negotiating manager 1705 (which negotiates *signaling and media parameters*, as discussed above in [6.0]) further

provides information including instructions related to a bit rate (*bandwidth parameter*):

The session negotiating manager 1705 also receives feedback data from the media transport manager 1735 and, based on the feedback data, determines the operation of the universal transmission buffer 1720 (e.g., whether to transmit or drop packets/frames) through the universal transmission buffer manager 1722. This feedback, in some embodiments, may include one-way latency and a bandwidth estimation bit rate. In other embodiments, the feedback includes packet loss information and roundtrip delay time (e.g., determined based on packets sent to the remote device in the video conference and the receipt of acknowledgements from that device). Based on the information from the media transport manager 1735, the **session negotiating manager 1705** can determine whether too many packets are being sent and **instruct the universal transmission buffer manager 1722 to have the universal transmission buffer 1720 transmit fewer packets (i.e., to adjust the bit rate as described in FIG. 12).**

Ex.1005, [0212]; *see also* Ex.1005, [0159] (“the set of network condition parameters include one-way latency and a bandwidth estimation bit rate.”), [0166] (“Since the available bandwidth for the video conference can change during the video conference, some embodiments continue to adjust the bit rate based on the

set of network condition parameters (i.e., the one-way latency and the bandwidth estimation bit rate) that are received from the remote device”).

101. Thus, Abuan renders obvious this claim.

9. **Claim 8**

a. **[8.0] *The method of claim 6 wherein the signaling and media parameters include a codec parameter.***

102. Abuan renders obvious this claim.

103. Abuan explains that the session negotiating manager 1705 (which negotiates *signaling and media parameters*, as discussed above in [6.0]) further provides information including how encoded images are created and transmitted:

The transmitter module 1715 retrieves encoded images (e.g., as a bitstream) from a video buffer (e.g., the buffer 1312 of FIG. 13) and packetizes the images for transmission to a remote device in the video conference through the universal transmission buffer 1720 and the virtual transport protocol manager 1725. **The manner in which the encoded images are created and sent to the transmitter module 1715 can be based on instructions or data received from the media transport manager 1735 and/or the session negotiating manager 1705.** In some embodiments, packetizing the images involves breaking the received bitstream into a group of packets each having a particular size (i.e., a size specified by the session negotiating manager 1705 according to a particular

protocol), and adding any required headers (e.g., address headers, protocol specification headers, etc.).

Ex.1005, [0213].

104. And as discussed in [6.0], it was well-known that such parameter negotiation, e.g., “using a session initiation protocol (SIP)” as contemplated by Abuan. Ex.1005, [0211]; *see* Ex.1041, 1-2 (describing the negotiation of codecs that will be used via SIP messages).

105. Thus, Abuan renders obvious this claim.

10. Claim 11

- a. **[11.0] *The method of claim 1 further comprising sending a notification from the one or more servers to the function block.***

106. Abuan discloses or at least renders obvious this limitation.

107. Abuan teaches that as part of the establishment of real-time communication session between the users, a video conference request sent to the video conference server:

After the video conference client 710 receives the request, **the video conference client 710 sends (at operation 2) a video conference request**, which indicates the device 720 as the recipient based on input from the user, **to the video conference server 715**. The video conference server 715 forwards (at operation 3) the video conference request to the video conference client 725 of the device 720. In some

embodiments, the video conference server 715 forwards the video conference request to the video conference client 725 using push technology. That is, the video conference server 715 initiates the transmission of the video conference request to the video conference client 725 upon receipt from the video conference client 710, rather than waiting for the client 725 to send a request for any messages.

Ex.1005, [0111].

108. This is reflected in operation 2 of Figure 7, below:

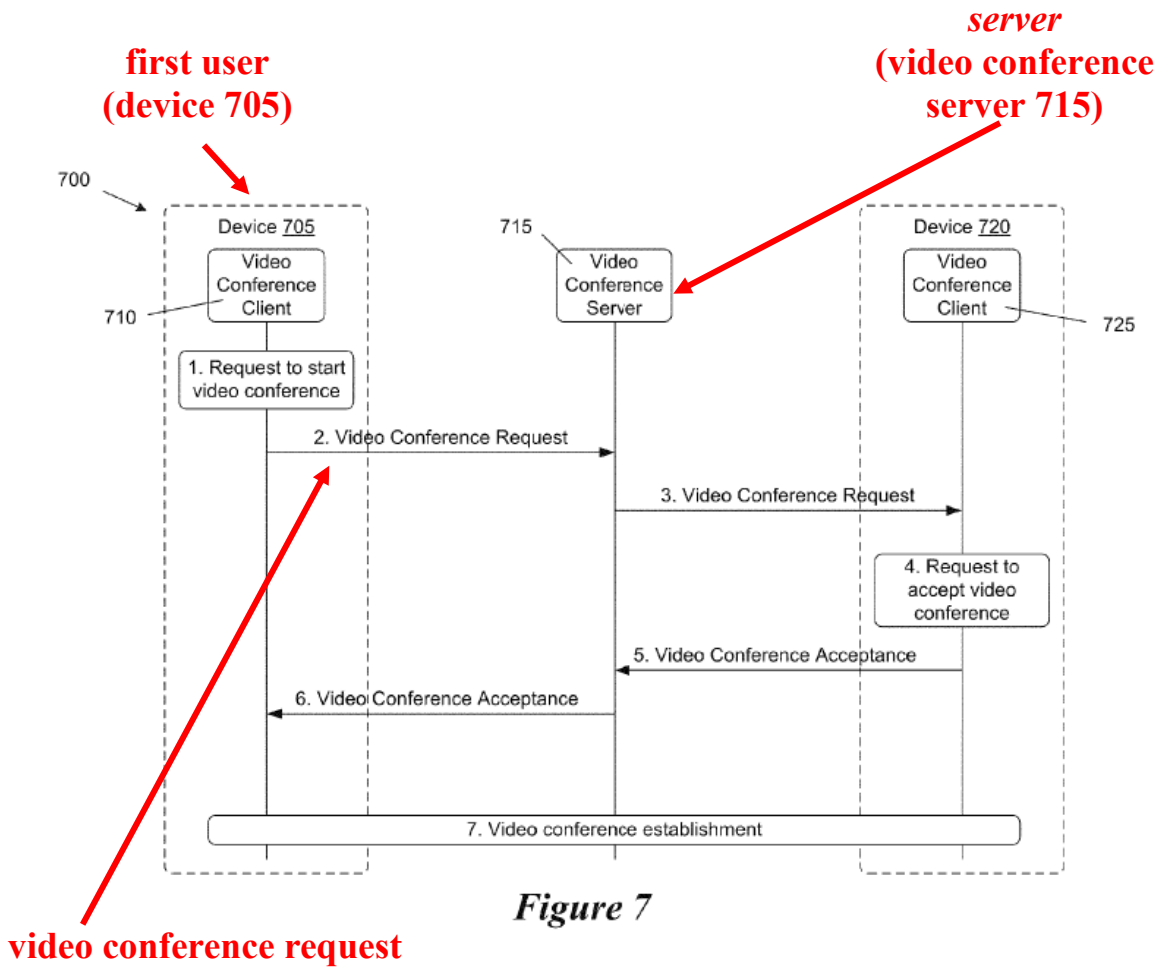


Figure 7

Ex.1005, Fig. 7.

109. Abuan explains that the video conference request is then sent (*sending a notification*) from the video conference server to the video conference client of a second device (*from the one or more servers to the function block*), where user interface is displayed to indicate that a request to start a video call by the first user:

FIG. 7 conceptually illustrates an example video conference request messaging sequence 700 of some embodiments. This figure shows the video conference

request messaging sequence 700 among a video conference client 710 running on a device 705, a video conference server 715, and a video conference client 725 running on a device 720. In some embodiments, the video conference clients 710 and 725 are the same as the video conference client 645 shown in FIG. 6. As shown in FIG. 7, **one device (i.e., the device 705) requests a video conference and another device (i.e., the device 720) responds to such request.** The dual camera mobile device described in the present application can perform both operations (i.e., make a request and respond to a request).

Ex.1005, [0108].

110. This is reflected in operation 4 of Figure 7, below:

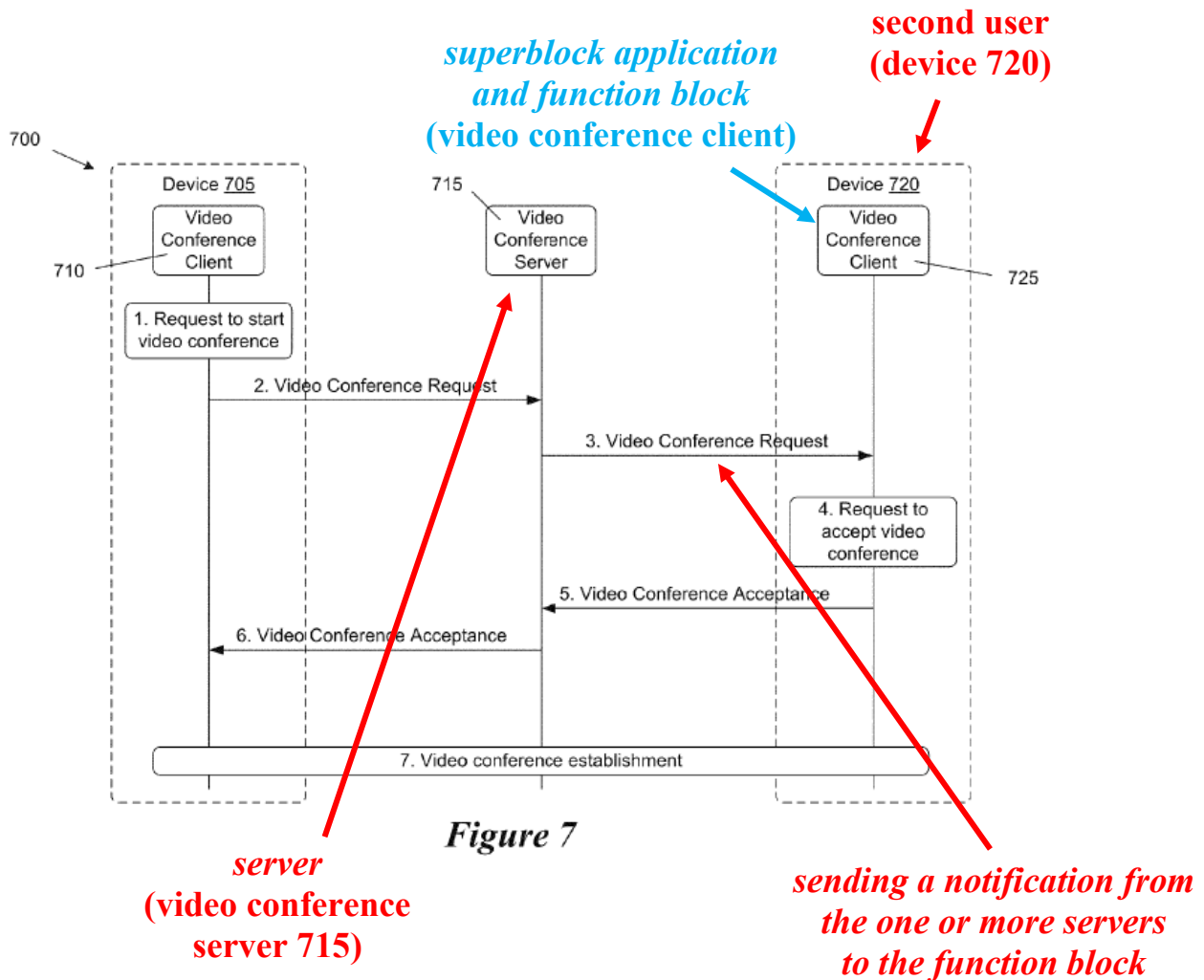


Figure 7

Ex.1005, Fig. 7.

111. Abuan explains that the video conference client operating on device 720 (i.e., including *the function block*) receives a video conference request, the user interface displays *a notification*:

When the video conference client 725 of some embodiments receives the video conference request, a user interface is displayed on the device 720 to indicate

to the user of the device 720 that the user of the device 705 sent a request to start a video conference and to prompt the user of the device 720 to accept or reject the video conference request. An example of such a user interface is illustrated in FIG. 9, which is described below. In some embodiments, when **the video conference client 725 receives (at operation 4) a request to accept the video conference request from the user of the device 705**, the video conference client 725 sends (at operation 5) a video conference acceptance to the video conference server 715. **The video conference client 725 of some embodiments receives the request to accept the video conference request when the user of the device 720 selects a user interface item of a user interface as illustrated in FIG. 9, for example.**

Ex.1005, [0112].

112. This is further illustrated in Figure 9, below, as a user interface displays a notification with items including “a name field 995, a message field 940, and two selectable UI items 945 and 950.” Ex.1005, [0130]. “Upon seeing the ‘Video Conference Invitation’ notation displayed in the message field 940, the invite recipient may deny or accept the request by selecting the Deny Request option 945 or Accept Request option 950 in the UI, respectively.” Ex.1005, [0131]. “The selectable UI item 975 is an Accept button 975 that the user may select to start video conferencing.” Ex.1005, [0134].

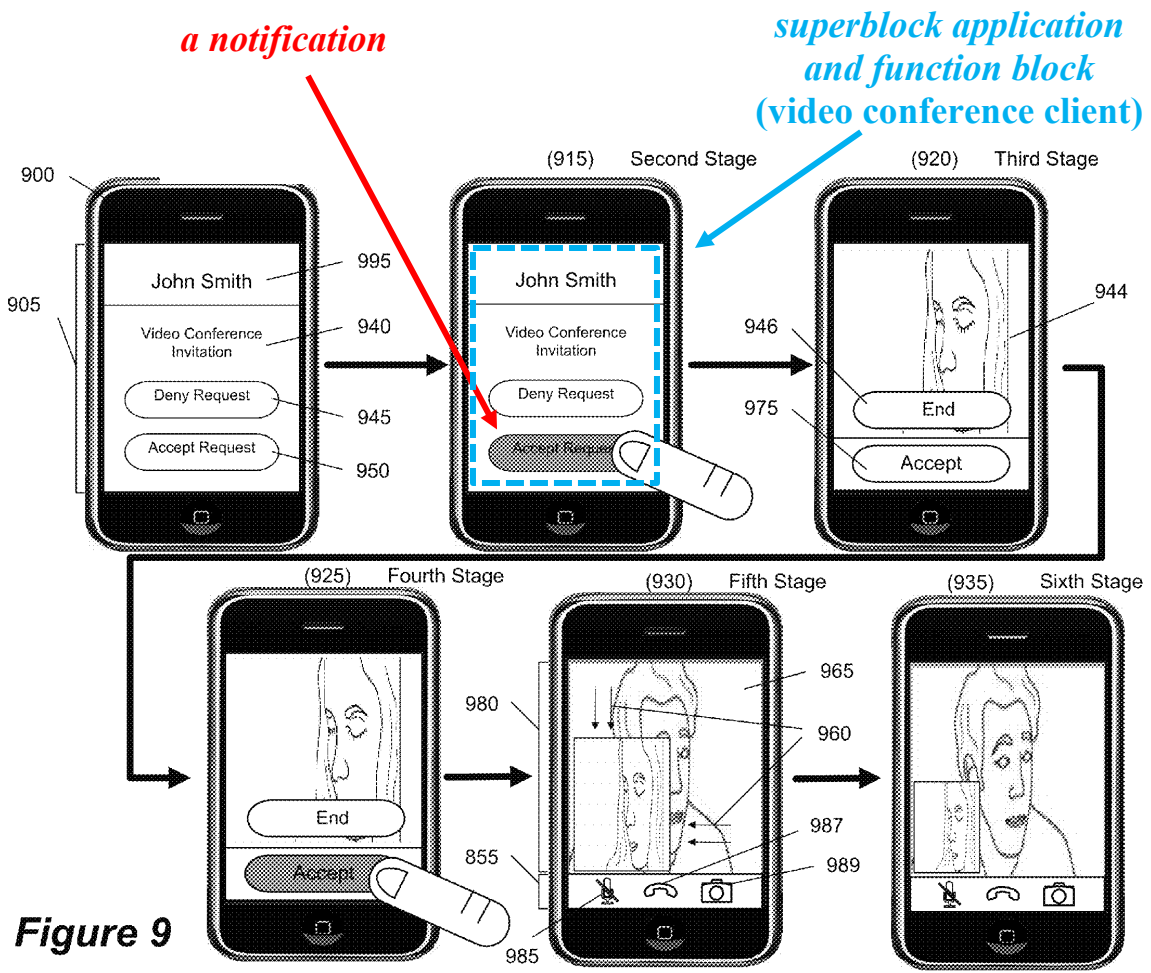


Figure 9

Ex.1005, Fig. 9.

113. Thus, Abuan discloses or at least renders obvious *sending a notification from the one or more servers to the function block.*

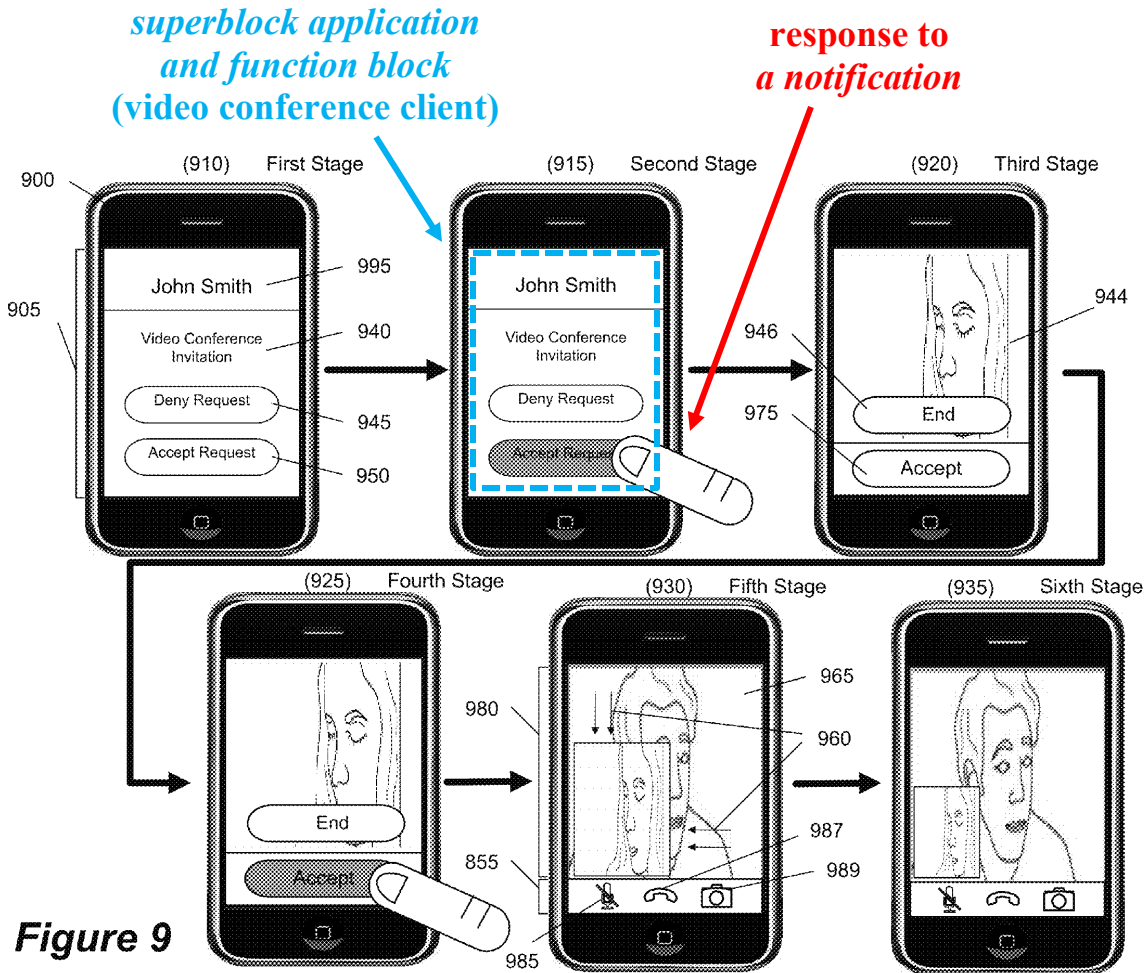
11. Claim 12

- a. [12.0] *The method of claim 11 wherein the notification requires a response and the establishment of the real-*

time communication session is the response to the notification.

114. Abuan renders obvious this claim.

115. As discussed at [11.0], Abuan explains that a user may respond to a video conference request by accepting or denying the request shown at step 915 (*“the notification”*). “Upon seeing the ‘Video Conference Invitation’ notation displayed in the message field 940, the invite recipient may deny or accept the request by selecting the Deny Request option 945 or Accept Request option 950 in the UI, respectively.” Ex.1005, [0130]. Since the user interface at step 915 offers only two mutually exclusive options, it would have been obvious *the notification requires a response*. When “Accept Request” is selected, a video conference is established “between the device 705 and the device 720” (such that *the establishment of the real-time communication session is the response to the notification*). Ex.1005, [0111]-[0112]. Figure 9, below, provides an example of where “Accept Request” is selected:



Ex.1005, Fig. 9.

116. Thus, Abuan renders obvious this claim.

12. **Claim 13**

- a. [13.0] *The method of claim 11 wherein the notification is a request to initiate the real-time communication session, and an API call indicates whether the request*

has been granted or denied by the superblock application.

117. Abuan renders obvious this limitation.

118. Abuan illustrates, with respect to Figure 7, below, the establishment of a video conference, where a video conference request sent to the video conference server (step 2), which is then sent from the video conference server to the video conference client of a second device (step 3), where a user interface displays a notification to indicate that a request to start a video call has been made by the first user (*the notification is a request to initiate the real-time communication session*):

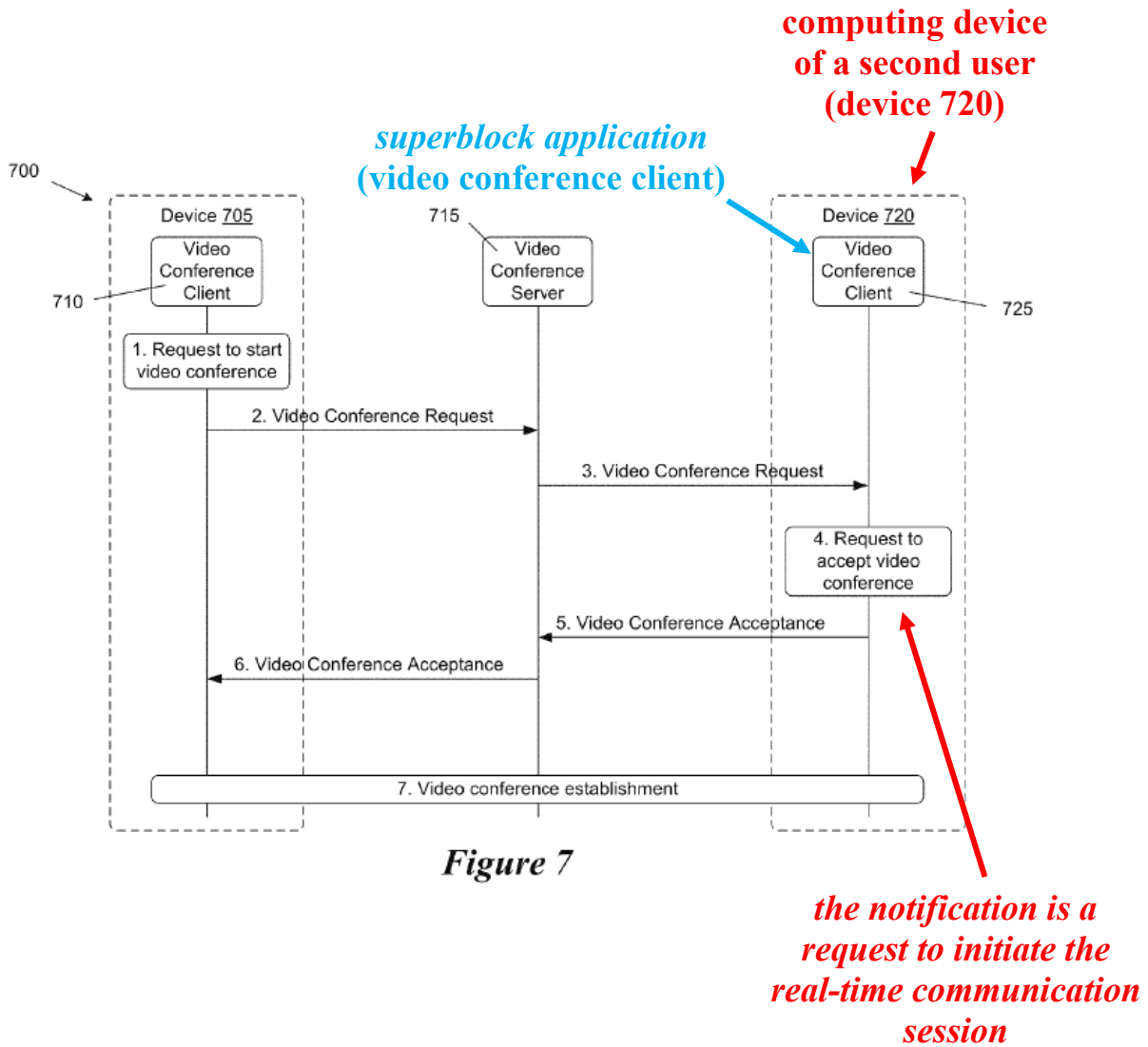
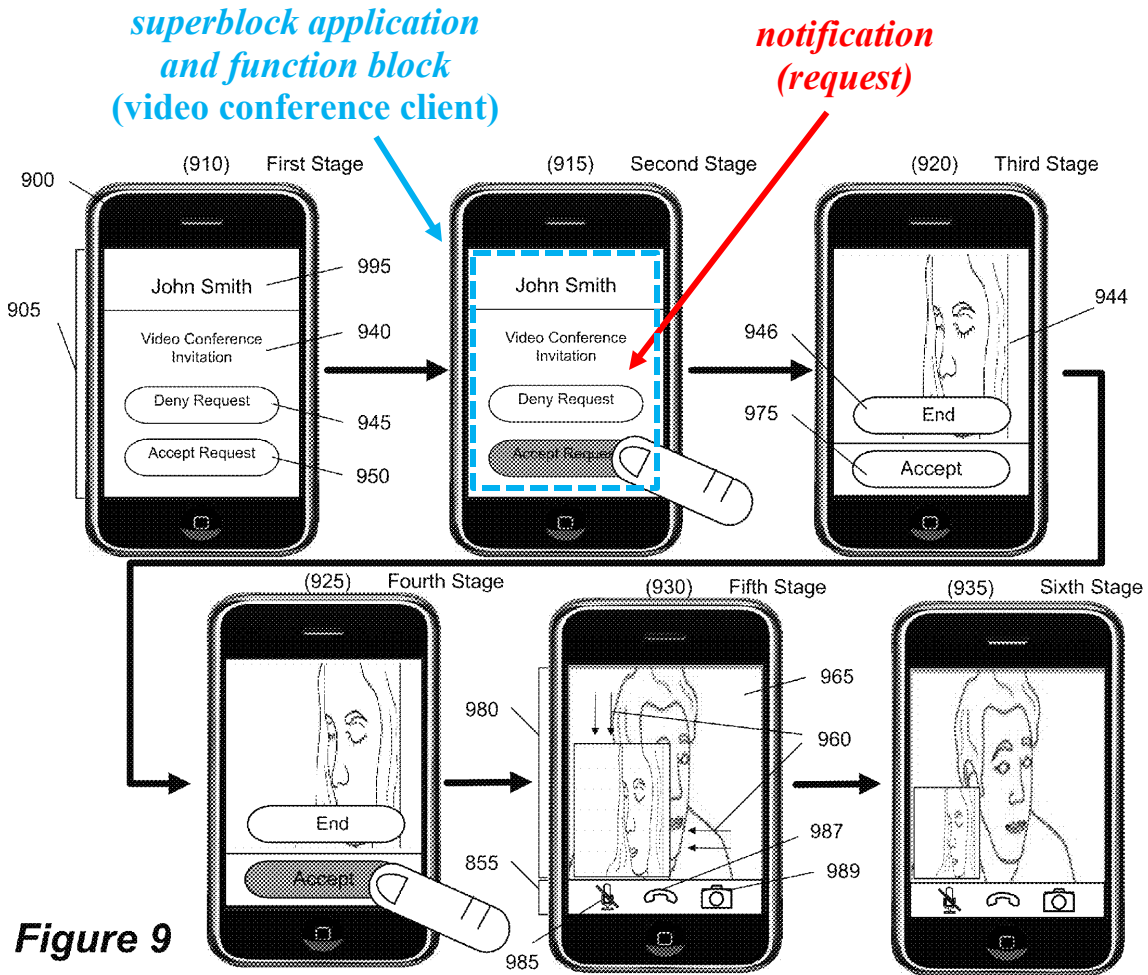


Figure 7

Ex.1005, Fig. 7.

119. This request is further illustrated in Figure 9, below:



Ex.1005, Fig. 9.

120. Further, as discussed above for [1.3], Abuan teaches that API calls are used to allow a client program to access services provided by libraries, such that *an API call indicates whether the request has been granted or denied by the superblock application*. More specifically, Abuan explains that the video conference client

application sends instructions from a user of the application to the video conference module 1302, including instructions to start a video conference:

The **client application 1365 of some embodiments sends instructions to the video conference module 1302 such as instructions to start a conference** and end a conference, receives instructions from the video conference module 1302, **routes instructions from a user of the dual camera mobile device to the video conference module 1302**, and generates user interfaces that are displayed on the dual camera mobile device and allow a user to interact with the application.

Ex.1005, [0172] (referencing Fig. 13).

The process 1500 begins by receiving (at 1505) instructions to start a video conference. In some embodiments, **the instructions** are received from the client application 1365 or **are received from a user through a user interface displayed on the dual camera mobile device and forwarded to the video conference manager 1304 by the client application 1365**. For example, in some embodiments, when a user of the dual camera mobile device accepts a video conference request, the instructions are received through the user interface and forwarded by the client application. On the other hand, when a user of the other device accepts a request sent from the local device, some embodiments receive the

instructions from the client application without user interface interaction (although there may have been previous user interface interaction to send out the initial request).

Ex.1005, [0187] (referencing Fig. 15); *see* Ex.1005, [0186] (“The operation of the video conference manager 1304 of some embodiments will now be described by reference to FIG. 15.”).

121. Accordingly, because the client application 1365 (*superblock application*) communicates with the video conference module (*function block*) via API calls, including instructions from a user such as instructions to start a video conference, Abuan renders obvious *an API call indicates whether the request has been granted or denied by the superblock application.*

13. Claim 14

a. [14.0] *The method of claim 11 wherein the notification is a presence notification.*

122. Abuan renders obvious this claim.

123. As discussed above in [11.0], Abuan teaches that as part of the establishment of real-time communication session between the users, a video conference request sent to the video conference server, which is sent in turn to the device of the user receiving the request, as reflected in Figure 7, below:

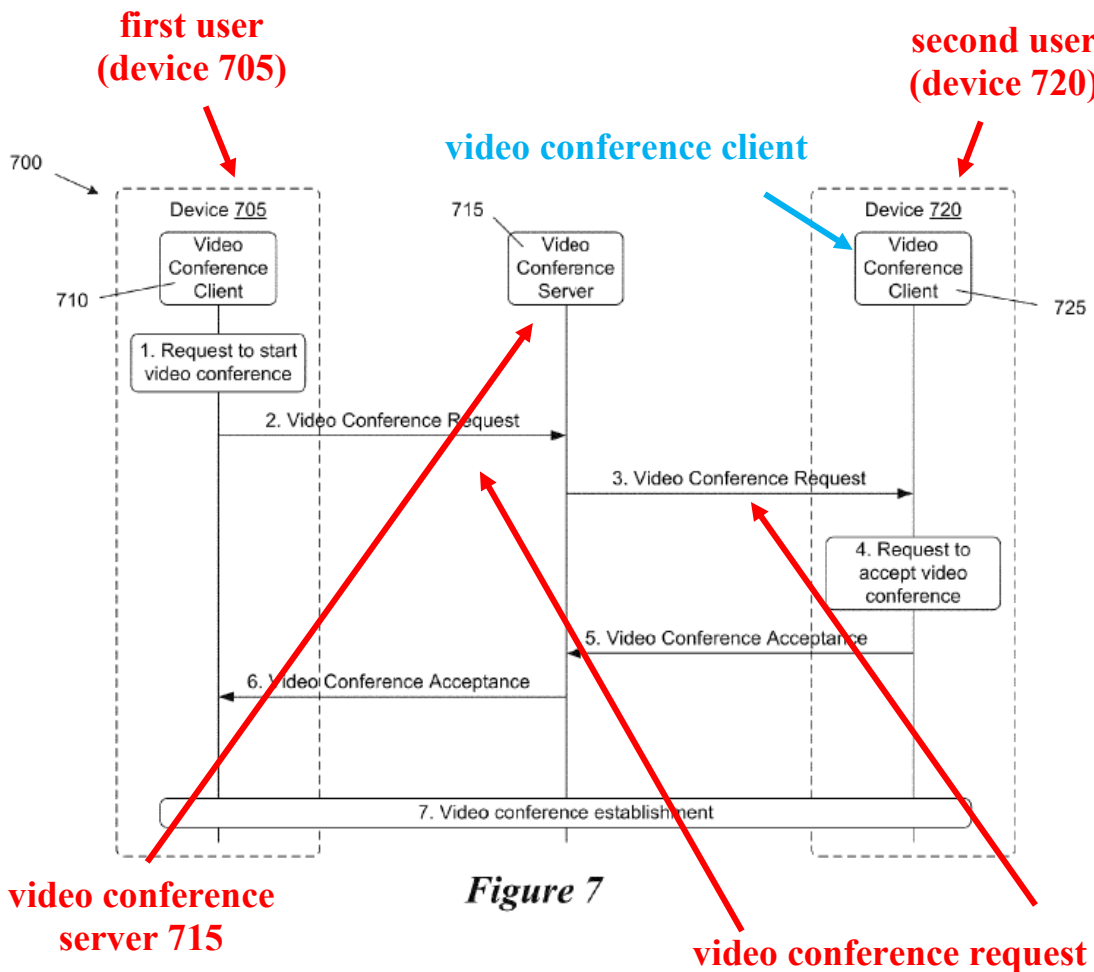


Figure 7

Ex.1005, Fig. 7.

124. Abuan explains that information regarding the incoming call, including the identity of the caller and options to accept or deny the video conference request, are displayed to the user when a video conference is requested:

The first stage 910 illustrates **the UI 905 when the invite recipient receives an invitation to a video conference from the invite requestor, John Smith.** As shown in FIG.

9, the UI 905 in this stage includes a name field 995, a message field 940, and two selectable UI items 945 and 950. **The name field 995 displays the name of a person who is requesting a video conference.** In some embodiments, the name field 995 displays a phone number of the person who is requesting a video conference instead of the name of the person. The message field 940 **displays an invite from the invite requestor to the invite recipient.** In this example, the “Video Conference Invitation” in the field 940 indicates that the invite requestor is requesting a video conference with the invite recipient. The selectable UI items 945 and 950 (which can be implemented as selectable buttons) provide selectable Deny Request and Accept Request options 945 and 950 for the invite recipient to use to reject or accept the invitation. Different embodiments may display these options differently and/or display other options.

Ex.1005, [0130].

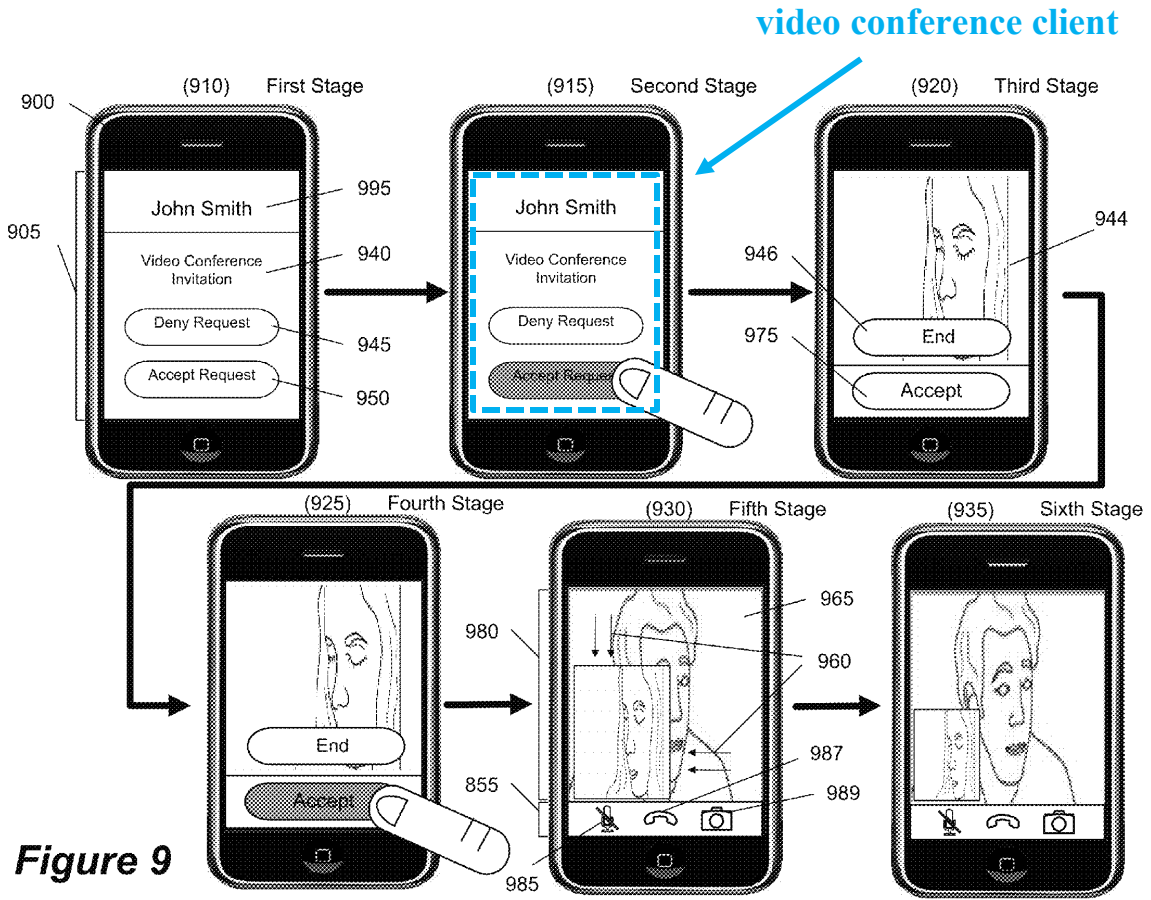


Figure 9

Ex.1005, Fig. 9.

125. Abuan's video conference invitation represents the *presence* of the user sending the request. For example, by sending the request, the user demonstrates that they are both online and awaiting a response to the request. *See* Ex.1005, [0111]. By communicating that a video conference is being requested and providing the identity of the user sending the video conference invitation, Abuan renders obvious that the

notification transmitted from a first user, to the server, to the second user *is a presence notification*.

126. Thus, Abuan renders obvious this claim.

14. **Claim 16**

- a. **[16.0] *The method of claim 1 further comprising receiving, by the one or more servers, a notification from the function block.***

127. Abuan discloses or at least renders obvious this claim.

128. As discussed above in [11.0], Abuan teaches that as part of the establishment of real-time communication session between the users, a video conference request sent to the video conference server:

After the video conference client 710 receives the request, **the video conference client 710 sends (at operation 2) a video conference request**, which indicates the device 720 as the recipient based on input from the user, **to the video conference server 715**. The video conference server 715 forwards (at operation 3) the video conference request to the video conference client 725 of the device 720. In some embodiments, the video conference server 715 forwards the video conference request to the video conference client 725 using push technology. That is, the video conference server 715 initiates the transmission of the video conference request to the video conference client 725 upon receipt from the video conference client 710, rather than

waiting for the client 725 to send a request for any messages.

Ex.1005, [0111].

129. This is reflected in operation 2 of Figure 7, below:

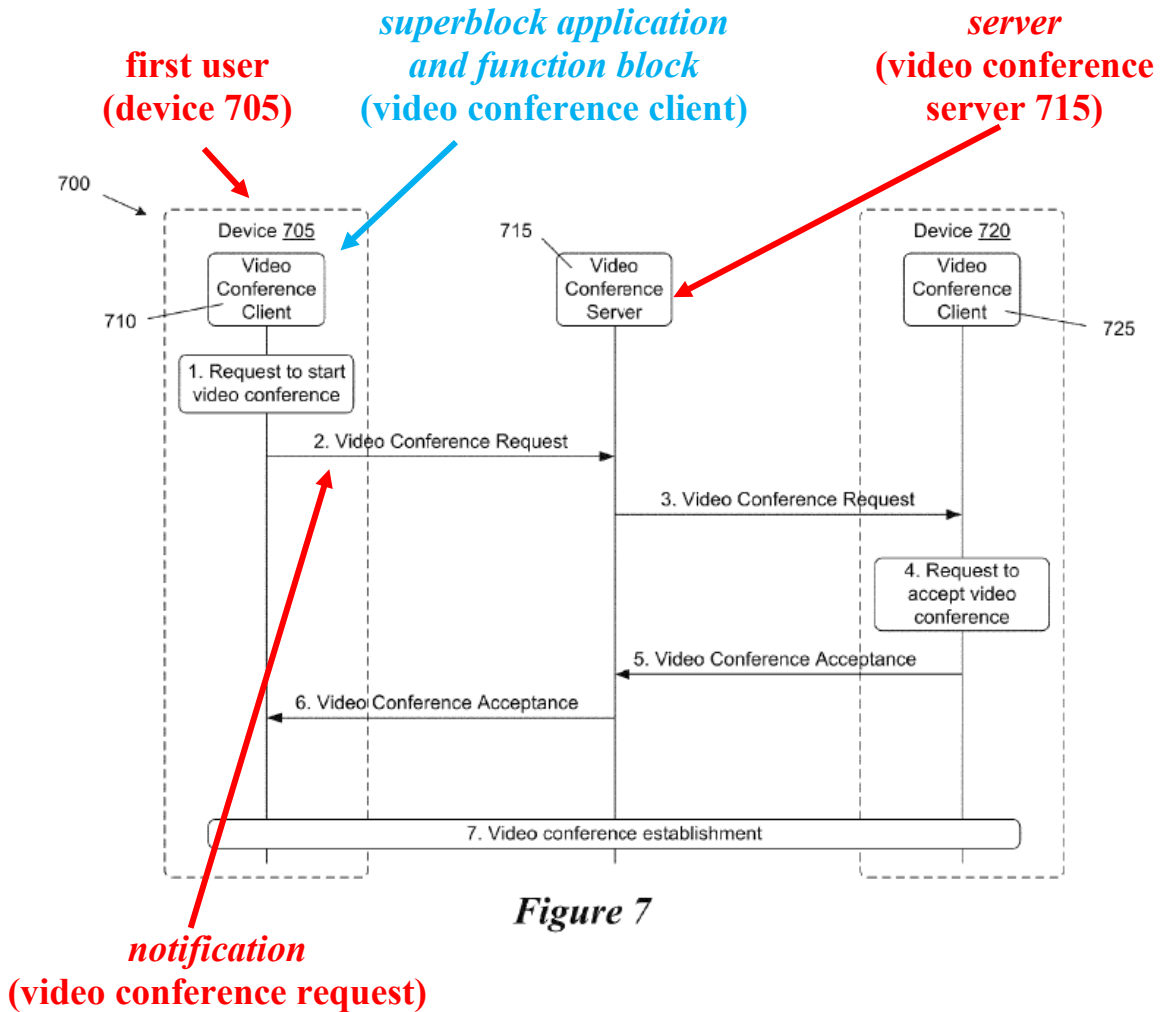


Figure 7

Ex.1005, Fig. 7.

130. As discussed in further detail above in [11.0], this *notification* is forwarded from the video conference server to device 720 in step 3, where the second user accepts or denies the request for a video conference. Ex.1005, [0112].

Accordingly, the video conference request sent in step 2 of Figure 7, above, is a *notification from the function block that is receiv[ed] by the one or more servers*, i.e., video conference server 715.

131. Thus, Abuan discloses or renders obvious this claim.

15. **Claim 17**

a. [17.0] *The method of claim 1 wherein the real-time communication session is for instant messaging.*

132. Abuan renders obvious this claim.

133. Abuan describes how establishing a videoconference session includes establishing “a control communication channel [] for messaging between the local mobile device and a remote device.” Ex.1005, [0218]. This control communication channel is used for “sending and receiving requests, notifications, and acknowledgements,” plus “remote control instruction messages.” Ex.1005, [0218]. A POSITA would have recognized that such messages would be exchanged substantially instantly because they relate to the establishment, maintenance, and control of the real-time videoconference communication session. As confirmation, Abuan explains that the control communication channel uses “protocols like a **real-time** transport control protocol (RTCP).” Ex.1005, [0218]. Abuan’s discussion of establishing a real-time messaging channel for controlling videoconference session renders obvious that “*the real-time communication session is for instant messaging.*” See also Ex.1001, 17:34-35 (“instant messaging

(IM) control module”). A POSITA would have been familiar with instant messaging, which was a technique for exchanging text-based messages in real time. Ex.1024, [0003] (“Instant messaging (“IM”) is a popular form of computer-based communication in which text-based messages are exchanged between users in real-time.”); Ex.1025, [0005] (a user “has access to a (text only) Instant Messaging client”).

134. Further, Abuan explains that its invention contemplates various types of *real-time communication session[s]*, including ones *for instant messaging* over an instant messaging application:

In some embodiments, the client application 1365 is the same as the video conference client 645 of FIG. 6. As mentioned above, the client application 1365 may be integrated into another application or implemented as a stand-alone application. **The client application 1365 may be an application that uses the video conferencing functions of the video conference module 1302, such as a video conferencing application, a voice-over-IP (VoIP) application (e.g., Skype), or an instant messaging application.**

Ex.1005, [0171].

In some embodiments, the communication system 2100 is operated by a service carrier who initially provisions a mobile device 2115 to allow the mobile device 2115 to use

the communication system 2100. Some embodiments provision a mobile device 2115 by configuring and registering a subscriber identity module (SIM) card in the mobile device 2115. In other embodiments, **the mobile device 2115** is instead configured and registered using the mobile device 2115's memory. Moreover, **additional services can be provisioned** (after a customer purchases the mobile device 2115) such as data services like GPRS, multimedia messaging service (MMS), and **instant messaging**. Once provisioned, the mobile device 2115 is activated and is thereby allowed to use the communication system 2100 by the service carrier.

Ex.1005, [0270].

135. Thus, Abuan renders obvious this claim.

16. **Claim 18**

a. **[18.0] *The method of claim 1 wherein the real-time communication session is a voice call.***

136. Abuan renders obvious this claim.

137. Abuan explains that its invention contemplates various types of *real-time communication session [s]*, including *a voice call* such as over a voice-over-IP application:

In some embodiments, the client application 1365 is the same as the video conference client 645 of FIG. 6. As mentioned above, the client application 1365 may be

integrated into another application or implemented as a stand-alone application. The **client application 1365 may be an application that uses the video conferencing functions of the video conference module 1302, such as a video conferencing application, a voice-over-IP (VoIP) application (e.g., Skype), or an instant messaging application.**

Ex.1005, [0171]. A POSITA would have been familiar with voice-over-IP, which was a technique for exchange audio communications in real time. Ex.1026, [0085] (“well-known Voice-over-IP standards that are cornerstone technology for the transmission of real-time audio”); Ex.1027, 6:39-47 (“devices which perform these functions [such as analog-to-digital (A/D) conversion,...echo cancellation, and other types of signal conditioning, as well as a voice activity detector (VAD)...function for determining the temporal boundaries of a telephone caller's speech] are well-known in the art and are commercially available”); Ex.1028, [0026] (“audio (such as voice-over-IP)”).

138. Further, as discussed above in claims 6-7, Abuan explains that video conference module 1302 (part of *the function block*) initializes modules for *voice call[s]*, including audio processing manager 1415, microphone 1425, speaker 1430, and performs “encoding [of] audio data captured by the microphone 1425 and decoding [of] audio data stored in the buffer 1410” Ex.1005, [0189].

139. Thus, Abuan renders obvious this claim.

17. Claim 19

- a. **[19.0] *The method of claim 1 wherein the real-time communication session is a video call.***

140. Abuan renders obvious this claim.

141. Abuan explains that its invention contemplates various types of *real-time communication session[s]*, including *a video call* over a video conferencing application:

In some embodiments, the client application 1365 is the same as the video conference client 645 of FIG. 6. As mentioned above, the client application 1365 may be integrated into another application or implemented as a stand-alone application. **The client application 1365 may be an application that uses the video conferencing functions of the video conference module 1302, such as a video conferencing application, a voice-over-IP (VoIP) application (e.g., Skype), or an instant messaging application.**

Ex.1005, [0171].

142. Thus, Abuan renders obvious this claim.

18. Claim 20

- a. **[20.0] *The method of claim 1 wherein the superblock application is further adapted to operate with iOS as an operating system for the computing device.***

143. Abuan renders obvious this claim.

144. Abuan's teachings support that frameworks provided by Apple, which are intended to be used on the iOS operating system, can be implemented as a media exchange module:

In some embodiments, the media exchange module 310 allows programs on the device that are consumers and producers of media content to exchange media content and instructions regarding the processing of the media content. In the video processing and encoding module 300, the media exchange module 310 of some embodiments routes instructions and media content between the video processing module 325 and the CIPU driver 305, and between the video processing module 325 and the encoder driver 320. To facilitate the routing of such instructions and media content, the media exchange module 310 of some embodiments provides a set of application programming interfaces (APIs) for the consumers and producers of media content to use. In some of such embodiments, the media exchange module 310 is a set of one or more frameworks that is part of an operating system running on the dual camera mobile device. One example of such a media exchange module 310 is the **Core Media framework provided by Apple Inc.**

Ex.1005, [0044]. Indeed, Apple's iOS operating system is executed on mobile devices and includes frameworks for media processing and exchange, such as the Core Media framework:

Media processing module 115 registers sensed image 125 to one or more reference images 135 to produce registered image 140. In an embodiment targeted for implementation on **a mobile device executing the iOS™ operating system**, media processing module 115 may be a framework that provides a low-level programming interface for managing and playing audiovisual media. (iOS is a trademark of Apple Inc.) **One such framework is the Core Media framework.** Once registration is complete, the registered image may be passed to user application 120 that may then further manipulate the image to generate final output image 145. In one embodiment, user application 120 could be an image processing application such as Aperture® or iPhoto®. (APERTURE and iPHOTO are registered trademarks of Apple Inc.)

Ex.1042; *see also* Ex.1043, 1 (iOS, the Apple mobile operating system reviously referred to as iPhone 2.0 software, beta announcement included APIs for “Core Services” and “Media...technologies”); Ex.1048 (Wayback Machine capture showing “Core Media Framework Reference” from Apple iPhone developer documentation for “iOS application”).

145. As explained in [1.1], Abuan’s video conferencing and processing module 1300 (including the client application 1365 and the various modules) is *the superbloc application*, which includes media exchange module 1320. Ex.1005,

[0181] (“is the same as the media exchange module 310 shown in FIG. 3, with more detail provided”). Apple’s Core Media framework is part of “a rich set of [APIs] and tools to create innovative applications for iPhone and iPod® touch.” Ex.1043, 1. Using Core Media framework—provided by Apple for applications running on iOS—as the media exchange module in Abuan’s video conferencing application 1365 means the application is *adapted to operate with iOS as an operating system*, as the framework provides interfaces for “iOS application[s].” Ex.1048. Thus, Abuan renders obvious *the superblock application is further adapted to operate with iOS as an operating system for the computing device*.

19. Claim 22

- a. **[22.0] *The method of claim 1 wherein the computing device is a cellular phone.***

146. Abuan renders obvious this claim.

147. Abuan explains that its invention contemplates a *computing device*, such as user device 705 and device 720 discussed above, including implementations where the computing device is a mobile phone (i.e., *cellular phone*):

Some embodiments of the invention provide a mobile device with two cameras that can take pictures and videos. **Examples of mobile devices include mobile phones,** smartphones, personal digital assistants (PDAs), laptops, tablet personal computers, or any other type of mobile

computing device. As used in this document, pictures refer to still picture images that are taken by the camera one at a time in a single-picture mode, or several at a time in a fast-action mode. Video, on the other hand, refers to a sequence of video images that are captured by a camera at a particular rate, which is often referred to as a frame rate. Typical frame rates for capturing video are 25 frames per second (fps), 30 fps, and 60 fps. The cameras of the mobile device of some embodiments can capture video images (i.e., video frames) at these and other frame rates.

Ex.1005, [0032].

148. Thus, Abuan discloses or at least renders obvious *the computing device is a cellular phone*.

20. Claim 23

a. [23.0] *The method of claim 1 wherein the computing device is a smart phone.*

149. Abuan renders obvious this claim.

150. Abuan explains that its invention contemplates a *computing device*, such as user device 705 and device 720 discussed above, including implementations where the computing device is a *smart phone*:

Some embodiments of the invention provide a mobile device with two cameras that can take pictures and videos. **Examples of mobile devices include** mobile phones, **smartphones**, personal digital assistants (PDAs), laptops,

tablet personal computers, or any other type of mobile computing device. As used in this document, pictures refer to still picture images that are taken by the camera one at a time in a single-picture mode, or several at a time in a fast-action mode. Video, on the other hand, refers to a sequence of video images that are captured by a camera at a particular rate, which is often referred to as a frame rate. Typical frame rates for capturing video are 25 frames per second (fps), 30 fps, and 60 fps. The cameras of the mobile device of some embodiments can capture video images (i.e., video frames) at these and other frame rates.

Ex.1005, [0032].

151. Thus, Abuan discloses or at least renders obvious *the computing device is a smart phone*.

21. **Claim 24**

a. [24.0] *The method of claim 1 wherein the computing device is a tablet.*

152. Abuan renders obvious this claim.

153. Abuan explains that its invention contemplates a *computing device*, such as user device 705 and device 720 discussed above, including implementations where the computing device is a *tablet*:

Some embodiments of the invention provide a mobile device with two cameras that can take pictures and videos. **Examples of mobile devices include** mobile phones,

smartphones, personal digital assistants (PDAs), laptops, **tablet personal computers**, or any other type of mobile computing device. As used in this document, pictures refer to still picture images that are taken by the camera one at a time in a single-picture mode, or several at a time in a fast-action mode. Video, on the other hand, refers to a sequence of video images that are captured by a camera at a particular rate, which is often referred to as a frame rate. Typical frame rates for capturing video are 25 frames per second (fps), 30 fps, and 60 fps. The cameras of the mobile device of some embodiments can capture video images (i.e., video frames) at these and other frame rates.

Ex.1005, [0032].

154. Thus, Abuan discloses or at least renders obvious *the computing device is a tablet*.

22. **Claim 25**

a. [25.0] *The method of claim 1 wherein the computing device is a personal digital assistant.*

155. Abuan renders obvious this claim.

156. Abuan explains that its invention contemplates a *computing device*, such as user device 705 and device 720 discussed above, including implementations where the computing device is a *personal digital assistant*:

Some embodiments of the invention provide a mobile device with two cameras that can take pictures and videos.

Examples of mobile devices include mobile phones, smartphones, **personal digital assistants (PDAs)**, laptops, tablet personal computers, or any other type of mobile computing device. As used in this document, pictures refer to still picture images that are taken by the camera one at a time in a single-picture mode, or several at a time in a fast-action mode. Video, on the other hand, refers to a sequence of video images that are captured by a camera at a particular rate, which is often referred to as a frame rate. Typical frame rates for capturing video are 25 frames per second (fps), 30 fps, and 60 fps. The cameras of the mobile device of some embodiments can capture video images (i.e., video frames) at these and other frame rates.

Ex.1005, [0032].

157. Thus, Abuan discloses or at least renders obvious *the computing device is a personal digital assistant*.

23. Claim 26

a. [26.0] *The method of claim 1 wherein the computing device is a laptop computer.*

158. Abuan renders obvious this claim.

159. Abuan explains that its invention contemplates a *computing device*, such as user device 705 and device 720 discussed above, including implementations where the computing device is a *laptop computer*:

Some embodiments of the invention provide a mobile device with two cameras that can take pictures and videos. **Examples of mobile devices include** mobile phones, smartphones, personal digital assistants (PDAs), **laptops**, tablet personal computers, or any other type of mobile computing device. As used in this document, pictures refer to still picture images that are taken by the camera one at a time in a single-picture mode, or several at a time in a fast-action mode. Video, on the other hand, refers to a sequence of video images that are captured by a camera at a particular rate, which is often referred to as a frame rate. Typical frame rates for capturing video are 25 frames per second (fps), 30 fps, and 60 fps. The cameras of the mobile device of some embodiments can capture video images (i.e., video frames) at these and other frame rates.

Ex.1005, [0032].

160. Thus, Abuan discloses or at least renders obvious *the computing device is a laptop computer.*

24. Claim 27

a. [27.0] *The method of claim 1 wherein the computing device is a desktop computer.*

161. Abuan renders obvious this claim.

162. While Abuan explains that its invention contemplates devices such as user device 705 and device 720 discussed above, Abuan further explains that its

“embodiments are used in cases involving a video conference between a dual camera mobile device and another device, such as a single camera mobile device, a **computer**, a phone with video conference capability, etc.” Ex.1005, [0279]. Accordingly, a POSITA would have recognized that Abuan’s teachings would be applicable to a *desktop computer*.

163. Thus, Abuan renders obvious *the computing device is a desktop computer*.

25. Independent Claim 55

- a. **[55.0] *A non-transitory computer readable medium embodying a computer program for providing a real-time communication session over the internet for a superblock application intended for use on a computing device, the computer program comprising instructions that include***

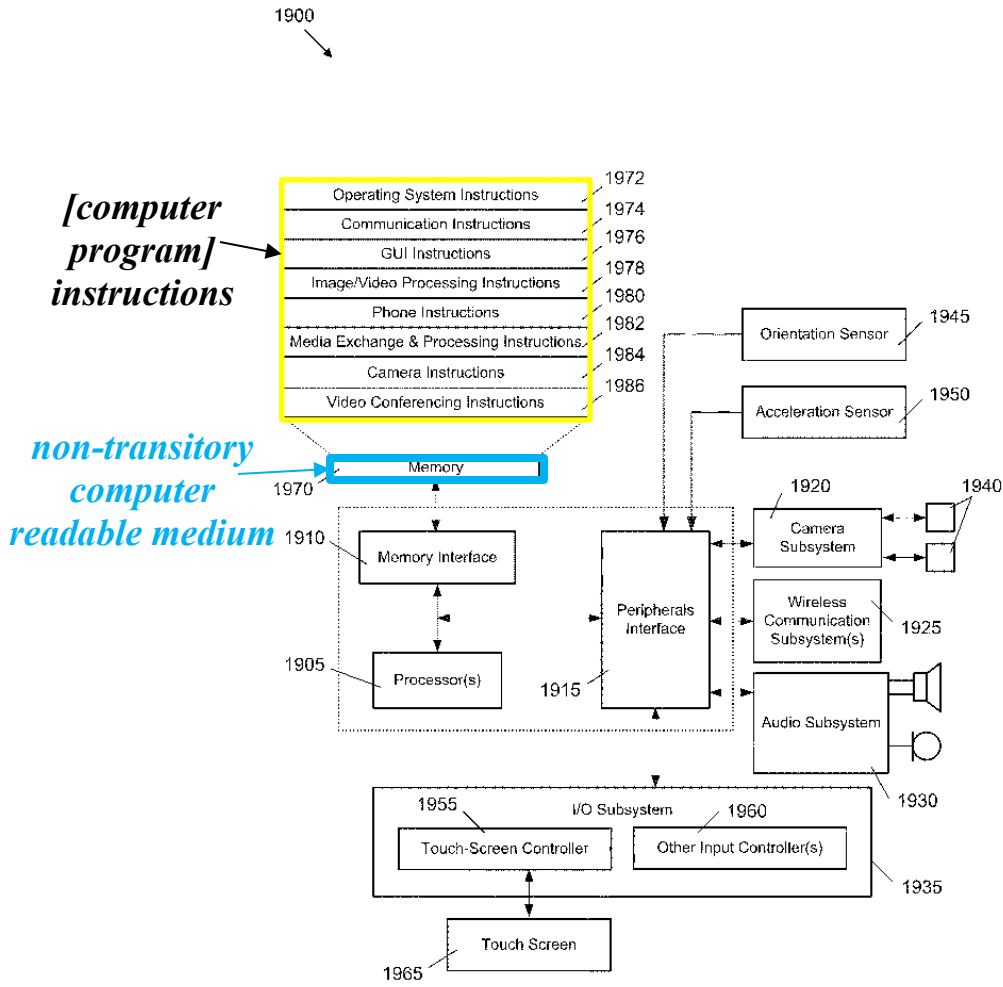
164. To the extent the preamble is limiting, Abuan discloses or renders it obvious.

165. As discussed above in [1.0], Abuan discloses or at least renders obvious *providing a real-time communication session over the internet for a superblock application intended for use on a computing device*. Further, Abuan’s methods “are implemented as software processes.” Ex.1005, [0235]. The software comprises program code embodied as computer program instructions residing in read-only memory:

In this specification, the term “software” is meant to include firmware **residing in read-only memory** or applications stored in magnetic storage which can be read into memory for processing by a processor. Also, in some embodiments, multiple software inventions can be implemented as sub-parts of a larger program while remaining distinct software inventions. In some embodiments, multiple software inventions can also be implemented as separate programs. Finally, any combination of separate programs that together implement a software invention described here is within the scope of the invention.

Ex.1005, [0234].

166. Further, Abuan explains that the software (*computer program comprising instructions*) is stored in “a machine readable medium,” and “in a form readable by...a computer.” Abuan, [0248].



Ex.1005, Fig. 19.

167. Accordingly, Abuan’s “memory” storing computer program instructions for a video conference client application is a *non-transitory computer readable medium embodying a computer program... the computer program comprising instructions* that perform the functionality described above in the analysis of claim 1.

- b. ***[55.1] a function block for use in adding additional functionality to a third party superblock application that has its own functionality and display window,***

168. See [1.1] above.

- c. ***[55.2] wherein the function block is configured to be compiled into the superblock application and is configured to add the additional functionality to provide the real-time communication session using one or more servers connected over the internet, and***

169. See [1.2] above.

- d. ***[55.3] wherein the function block is configured to interact with the superblock application through one or more application programming interface (API) calls, and***

170. See [1.3] above.

- e. ***[55.4] wherein the function block is further configured to enable establishment of the real-time communication session between the one or more servers and the function block compiled into the superblock application so that the function block can provide the real-time communication session to the superblock application.***

171. See [1.4] above.

B. Ground 2: Claim 15 would have been Obvious over Abuan in view of Eisenberg.

1. Summary of Eisenberg

172. U.S. Patent Publication No. 2010/0066807 to Eisenberg (Ex.1019, “Eisenberg”) was filed on October 20, 2009, and published on March 18, 2010. Ex.1019, cover at (22), (43). Eisenberg is titled “Initiation and support of video

conferencing using instant messaging.” Ex.1019, cover at (54). Eisenberg was not before the Examiner during prosecution of the ’116 patent.

173. Eisenberg describes a “system and method for initiating and supporting network video conferences[.]” Ex.1019, Abstract. Like Abuan, Eisenberg contemplates that “one or more instant messaging enabled clients can initiate a video conference.” Ex.1019, Abstract. Further, an intermediary server, “e.g., a video conferencing server, supports video conferences between video conference participants, a video conference being initiated between video conference participants in response to an instant message transmitted between the at least two client nodes.” Ex.1019, Abstract. Eisenberg recognizes that “[i]nstant messaging (IM) is becoming an increasingly popular utility on networks such as the Internet.” Ex.1019, [0003]. Eisenberg details that within such services, a “presence” feature is provided “which permits IM clients to know the status of other clients on the network serviced by the messenger service.” Ex.1019, [0003]. This allows users to communicate their availability or otherwise block incoming messages when unavailable. Ex.1019, [0003].

174. Eisenberg is analogous art to the ’116 patent because it is in the same field of endeavor as the ’116 patent, namely, the “manner in which functionality is accessed in certain environments, such as mobile device environments.” Ex.1001, 1:58-60; *see* Ex.1019, [0005]-[0006] (describing “the convenience of instant

messaging with video conferencing capability to allow a user to initiate a video conference using IM” on “client nodes” or other types of “video conferencing system[s]”). Eisenberg and the ’116 patent also both generally relate to implementing functionality via software. *See* Ex.1001, Abstract (“providing additional functionality to existing software”); Ex.1005, [0027] (describing a device including “an IM client software module 16, a video conference initiation software module 18 and a video conferencing client software module 28.”), [0023] (“each IM client 12 includes a video conference module 18 which interfaces with both the IM client module 16 and the CTM server 20 to initiate a video conference”). Eisenberg and the ’116 patent both describe at length features for facilitating videoconferencing. Ex.1001, Abstract, 4:43-51 (purpose of a function block is to provide functions including “audio, video,...conferencing, meetings, and/or other functions”); Ex.1019, [0005] (“a video conference is initiated between video conference participants”), [0023] (“each IM client 12 includes a video conference module 18 which interfaces with both the IM client module 16 and the CTM server 20 to initiate a video conference”).

2. Reasons to Combine

175. A POSITA would have been motivated to include a presence indicator, as taught by Eisenberg, in Abuan’s video conference client because doing so would have been the combination of prior art elements (Abuan’s video conference client

with Eisenberg's presence features) according to known methods to yield predictable results (providing information regarding the availability of a user before placing a call to them using Abuan's video conference client). Eisenberg explicitly teaches a presence feature for instant message clients, and more broadly contemplates a "system and method for initiating and supporting network video conferences." Ex.1019, Abstract. And Abuan similarly teaches an instant message client, such as its video conference client, that is able to establish video conferences over a network between two devices. Ex.1005, [0109]. A POSITA would have had a reasonable expectation of success in making such a combination because Eisenberg is directed to video conferencing, and compliments Abuan's teachings of establishing a video conference between two devices with additional information related to the users interacting with Abuan's video conference client.

3. Claim 15

- a. ***[15.0] The method of claim 14 wherein the function block is further configured to update a presence indicator within the superblock application in response to receiving the presence notification.***

176. Abuan in view of Eisenberg renders obvious this claim.

177. As discussed above for [1.0], Abuan discloses "an application that uses the video conferencing functions of the video conference module 1302, such as a video conferencing application, a voice-over-IP (VoIP) application (e.g., Skype), or **an instant messaging application.**" Ex.1005, [0171]. Eisenberg

provides further implementation details regarding an instant messaging application, such as contemplated by Abuan. More specifically, Eisenberg explains that “[i]nstant messaging (IM) is becoming an increasingly popular utility on networks such as the Internet.” Ex.1019, [0003]. For such applications, Eisenberg notes that it is typical to include a *presence indicator* regarding whether a client is online or offline:

The messenger service also typically provides a “presence” feature which permits IM clients to know the status of other clients on the network serviced by the messenger service. The status information for a client includes whether the client is online and available to receive instant messages from other clients and whether the client presently at his workstation, which can be determined by the level of activity at the client's workstation. The presence feature also allows clients to declare themselves unavailable to receive instant messages so that incoming messages can be blocked.

Ex.1019, [0003].

178. To that end, Eisenberg’s teaching of providing a presence feature is advantageous for a video conference client, such as the one contemplated by Abuan, because Eisenberg explains that it allows users to communicate their availability or otherwise block incoming messages when unavailable. Ex.1019, [0003]. And as discussed above, a POSITA would have been motivated to combine Eisenberg’s

teaching of a presence feature in Abuan's video conference client because doing so would have been the combination of prior art elements (Abuan's video conference client with Eisenberg's presence features) according to known methods to yield predictable results (providing information regarding the availability of a user before placing a call to them using Abuan's video conference client). Eisenberg explicitly teaches a presence feature for instant message clients, and more broadly contemplates a "system and method for initiating and supporting network video conferences." Ex.1019, Abstract. And Abuan similarly teaches an instant message client, such as its video conference client, that is able to establish video conferences over a network between two devices. Ex.1005, [0109]. A POSITA would have had a reasonable expectation of success in doing so because Eisenberg specifically contemplates such features in the context of video conferencing.

179. Thus, Abuan in view of Eisenberg renders obvious this claim.

C. Ground 3: Claim 21 would have been Obvious over Abuan in view and Beilis.

1. Summary of Beilis

180. Beilis is related to "the field of mobile communications," and telecommunications applications running "on a mobile computing platform." Ex.1046, [0003]. In an example, Beilis discloses an "android device," that "run[s] a mobile platform such as the well-known android platform," and is running a "telecommunications video application." Ex.1046, [0024]. Beilis further explains

that an application may command the telecommunications video application “to launch a video call.” Ex.1046, [0027].

181. Beilis is in the same field of endeavor as the ’116 patent, namely, the “manner in which functionality is accessed in certain environments, such as mobile device environments.” Ex.1001, 1:58-60; see Ex.1046, [0003]-[0005] (describing the ways mobile applications run in different computing platforms, such as Android). Beilis and the ’116 patent also both generally relate to implementing functionality via software. See Ex.1001, Abstract (“providing additional functionality to existing software”); Ex.1046, [0011], [0013], [0024] (describing mobile device running software and software applications, such as telecommunications video applications). Beilis and the ’116 patent both describe features for facilitating videoconferencing. Ex.1001, Abstract, 4:43-51 (purpose of a function block is to provide functions including “audio, video,...conferencing, meetings, and/or other functions”); Ex.1046, [0027]-[0028], [0039] (describing running a mobile application to “launch,” “render,” “transfer,” and “resize” a video call).

182. Beilis is reasonably pertinent to a problem allegedly addressed by the ’116 patent. The ’116 patent describes purported problems associated with using multiple applications on a mobile device, potentially requiring a user to “switch[] back and forth between the video window of [a] call and [a] superblock application.”

Ex.1001, 4:22-26. The '116 patent's proposed solution is to provide functionality—such as communication functions—to a superblock application. Ex.1001, 4:43-61. Beilis similarly describes the complexities of having multiple applications running in parallel on a mobile device, where “a user must select the application and engage in manual task performance to replace the original display with new data from the other application.” Ex.1046, [0008]. Beilis integrates the capability to perform function calls while continuously occupying the full screen so that the user does not need to switch back and forth between applications. Ex.1046, [0029].

2. Reasons to Combine

183. A POSITA would have been motivated to configure Abuan's stand-alone video conferencing application available on the Android platform, as taught by Beilis, because doing so would have been a predictable, routine combination of prior art element using known software development techniques to yield predictable results (Abuan's video conferencing application configured to run on the well-established Android platform). A POSITA would have appreciated that configuring Abuan's application to run on the Android platform is an implementation detail that would not change the underlying videoconferencing functionalities disclosed by Abuan. A POSITA would have had reasonable expectation of success in the combination because Beilis shows that video

communications applications were available on, and operable within, the Android platform.

3. Claim 21

- a. **[21.0] *The method of claim 1 wherein the superblock application is further adapted to operate with Android as an operating system for the computing device.***

184. Abuan alone or in view of Beilis renders obvious this claim.

185. As discussed above, Abuan contemplates that its video conference client is “an application that may use the video conferencing functions” such as Skype. Ex.1005, [0105], [0171]. It was well-known that video conferencing applications, including Skype itself, were also available on the Android operating system. Ex.1044

(<https://web.archive.org/web/20101006000452/http://blogs.skype.com/en/2010/10/android.html>). And while Abuan provides frameworks provided by Apple Inc. as examples of its modules, Abuan does not limit itself to iOS as the only operating system it contemplates. Ex.1005, [0044]; *see also* Ex.1007, [0048].

186. Beilis provides additional implementation details for operating mobile applications on different devices and platforms, including “android device[s]” running the “android platform.” Ex.1046, [0016]. Beilis explains that an android device may run a “telecommunications video application” on the android platform (“*Android as an operating system*”), which application is used to “launch” and

“render” a video call. Ex.1046, [0024], [0027], [0029]. Thus, Beilis’s telecommunications application is “*adapted to operate with Android as an operating system for the computing device.*”

187. A POSITA would have been motivated to adapt[] Abuan’s video conference client (“*superblock application*”) to run on the “Android operating system,” as taught by Beilis, because doing so would have been a predictable, routine engineering choice aimed at making the same mobile video-conferencing functionality available to the large and well-established Android ecosystem, without changing the underlying conferencing concepts disclosed by Abuan. Beilis shows that video communications applications were available on, and operable within, the Android platform, and that launching and conducting video calls through an Android application was a known and achievable implementation detail.

188. Thus, Abuan alone or in view of Beilis renders obvious this claim.

D. Ground 4: Claims 9-10, 28-41, 43-47, and 49-54 would have been Obvious over Abuan in view of Platt and Guzman.

1. Summary of Guzman

189. U.S. Patent Publication No. US 2012/0092438 to Guzman Suarez et al. (“Guzman,” Ex.1007) was filed on Oct. 18, 2010, and published on Apr. 19, 2012. Ex.1007, 1 (fields 22 (filing date) and 43 (publication date)). Guzman is

titled “Overlay for a Video Conferencing Application.” Ex.1007, 1 (field 54).

Guzman was not before the Examiner during prosecution of the ’116 patent.

190. Guzman describes a video-conferencing application that facilitates account setup, authentication, contact management, and call initiation for video conferencing sessions. Ex.1007, Abstract, [0002]–[0006], [0033], [0037], [0049]–[0051]. Guzman’s application and its interface allows a user to sign in to a service (for instance, via a remote server) using login credentials to gain access to the user’s contacts, and thereby gain the ability to initiate video conference calls to those contacts. Ex.1007, [0044]–[0046], [0060]–[0063], [0074]–[0075], [0058], [0070]–[0071].

191. Guzman’s video conferencing application includes an “auto-login” option that allows the application to automatically provide stored login credentials to the authenticating system without the user having to input those credentials for every video conferencing session. Ex.1007, [0056]–[0059].

2. Reasons to Combine

192. A person of ordinary skill in the art would have seen clear, practical reasons to combine Abuan’s videoconferencing framework with Guzman’s login and auto-login mechanisms.

193. Abuan recognizes that a video call can only proceed with the recipient’s permission, as reflected in its discussion of converting a phone call into

a video conference “with the permission of the other party.” Abuan, [0117]. Abuan also shows, in Figure 7, that these video sessions depend on an intermediary device—a videoconference server—to set up, manage, and carry the session. In practice, that server is a shared, finite resource. A skilled engineer would thus appreciate that the server’s participation is not just functional but also conditional, in the sense that the server must decide whether to allow or deny additional sessions based on access control and capacity.

194. Without some gatekeeping, the system would be exposed to misuse and overload. A system that allows any client to start a session at any time without authentication increases the risk that the server will be overwhelmed, whether by intentional attacks or simply by volume spikes and abusive patterns. This is a textbook problem in networked services: denial-of-service conditions occur when requests flood a server, consuming bandwidth or compute such that legitimate users are deprived of service. Allowing unrestricted and unlimited use of a server-provided feature would create the undesirable risk the server becoming overwhelmed and failing, either intentionally (in a denial-of-service attack) or unintentionally. Ex.1037, 19:5-10 (“denial of service attack (also, DoS attack) is an attack...that causes a loss of service to users, typically...by consuming the bandwidth of the victim network or overloading the computational resources of the victim system”); Ex.1038, 2:30-39 (“denial of service (DOS) attacks, where...the

server at the target IP address is overwhelmed by fake requests for service, and is either unable to service ‘real’ requests for service, or crashes due to overload”); Ex.1039, [0002] (“In a denial of service attack, one or more attackers may send packets towards one or more servers, overwhelming the servers so that they are no longer able to serve their legitimate clients.”). Engineers routinely mitigate these risks by requiring authenticated access and by tying service initiation to verified user identity.

195. Guzman addresses precisely this issue by a login process through which a user supplies a username and password to access a service such as videoconferencing. Ex.1007, [0056]. Guzman further proposes an auto-login procedure to more conveniently allow a user who has already authenticated successfully to bypass repeated logins on subsequent uses. Ex.1007, [0056]. That approach is a well-known way to preserve access control while keeping the user experience efficient.

196. A skilled practitioner would find it natural and straightforward to incorporate Guzman’s login and optional auto-login into Abuan’s clients and server. Abuan already relies on a centralized server to broker and maintain sessions, and the addition of a credential check at the server boundary is a standard software pattern. The combination does not require novel algorithms or unconventional architectures; it calls for routine programming work within

ordinary skill to integrate authentication workflows, maintain session state, and, where desired, persist tokens or credentials to enable auto-login. The expected outcome is predictable: restricting access to videoconference services to authenticated users, thereby preventing unauthorized use and reducing the risk of server overload from anonymous or automated traffic.

197. Put differently, applying Guzman's login technique to Abuan's videoconferencing system is the application of a known access-control mechanism to another known server-mediated communication service that would have benefited from, and expected, an authorization mechanism. The resulting benefits—controlling who can initiate or join a session, improving system stability under load, and aligning with standard security practices—are entirely foreseeable. The same logic holds for both the client and server components in Abuan: implementing login at the client side to capture credentials and at the server side to verify and authorize access is the ordinary way similar systems are improved—namely, by limiting functionality to legitimate, logged-in users in order to protect shared server resources while maintaining a smooth user experience via auto-login.

3. Claim 9

- a. **[9.0] *The method of claim 1 further comprising providing an authorization key for use by the superblock application, wherein the authorization key***

unlocks the function block for use by the superblock application.

198. Abuan in view of Guzman renders obvious this claim.

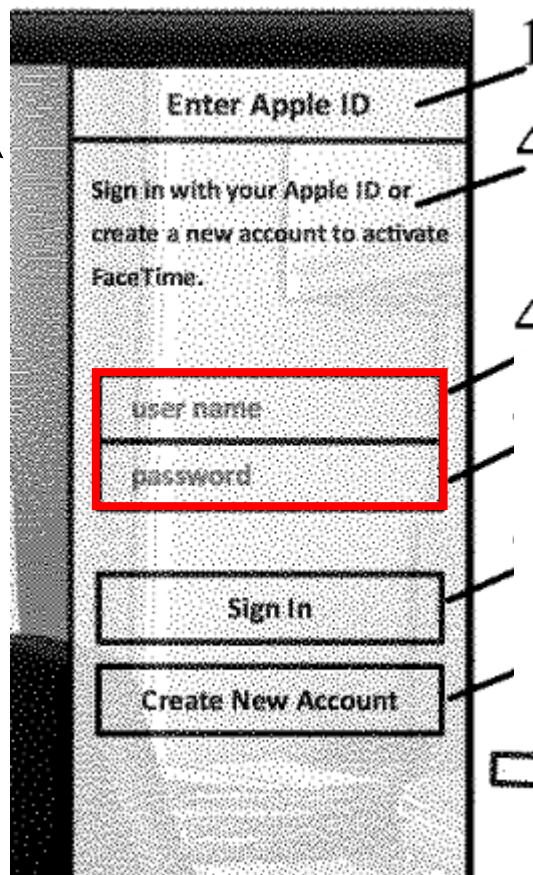
199. Abuan's Figure 7 assumes that the video conference client 710 is authorized to use the video conference module 1302 to communicate with the video conference server 715 and the user of device 705 is already logged into the video conference server 715. It would have been obvious to a POSITA, however, for the video conference server to require users to login to use the server's videoconferencing capabilities, because requiring a user to login was nearly universally employed by servers at the time. *See* Ex.1040, Fig. 6 (showing a "typical login screen" asking for "an account name 293 and a password 295"); Ex.1021, [0024] ("A user login typically includes a user name and password."); Ex.1022, [0028] (describing "operations typically performed" during login for "authentication and authorization"); Ex.1023, [0002] (describing the "well known" login process in which "a server computer is responsible for granting access to a remote user based on a valid combination of userID and password"). Consistent with that understanding and with Abuan's general architecture, Guzman describes a videoconferencing application for providing videoconference communications between two or more clients. Ex.1007, Abstract, [0034].

200. Guzman discloses a "video conferencing application [] for conducting a video conference between [a] first electronic device and a second electronic

device.” Ex.1007, Abstract. Like Abuan’s video conference module 1302 that provides video conferencing functions to the client application 1365 on a mobile device, Guzman’s video conferencing application “enables a user of [an electronic] device to engage in a video conference with a remote user of a second electronic device that also includes a video conferencing application.” Ex.1005, [0171]; Ex.1007, [0034], [0221] (describing application as a “client”).

201. Guzman describes a user inputting a “username” and “password” (*authorization credentials*) into an “overlay display area” of a “video conferencing application.” Ex.1007, [0050]. In one example, Guzman describes a log-in procedure wherein the application receives, as a user input from UI (part of the *superblock application*) a username and password. Ex.1007, [0074], Fig. 3. With respect to Fig. 4 below, Guzman shows the overlay display area UI with two “entry fields” for “user name” and “password.”

“superblock application”



**UI fields for application
to receive
“authorization
credentials”**

Ex.1007, Fig. 4 (partial annotated)

202. “[W]hen the user selects the “Sign In” item 465, the video conferencing application verifies whether the login information entered by the user is valid.” Ex.1007, [0077]. Guzman describes different ways to verify and authenticate the user’s credentials. First, the credentials can be verified by the video conferencing application itself. Ex.1007, [0057]. Second, the credentials can be verified “by accessing a remote server that stores login information.” Ex.1007, [0057].

203. Thus, Guzman's disclosure of verifying login credentials by the video conferencing application itself discloses "*providing an authorization key for use by the superbloc application.*" Further, the user interface interaction module 2205 (*function block*) receives the authorization credentials from the UI of the video conferencing application and facilitates login. When the credentials have been verified, the user's contacts are loaded and made available so that the user can initiate a video conference (*the authorization key unlocks the function block for use by the superbloc application*). Ex.1007, [0058], Fig. 4.

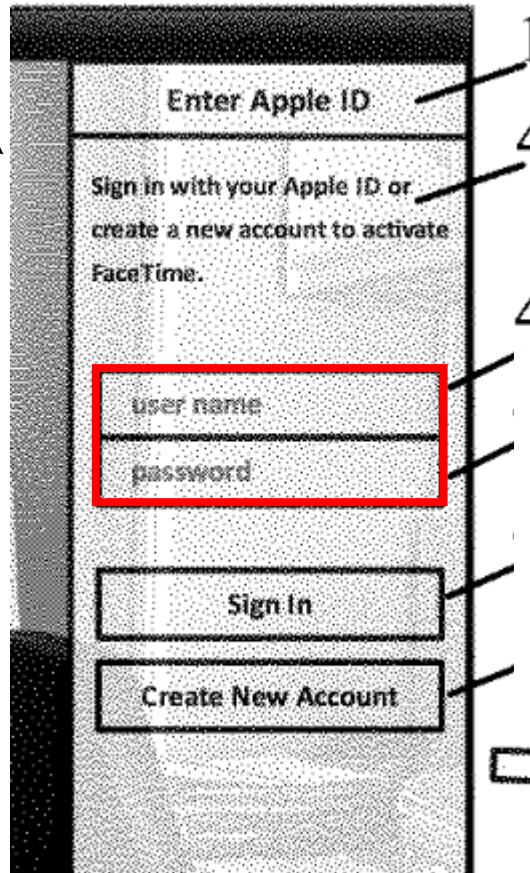
4. **Claim 10**

- a. **[10.0] *The method of claim 1 further comprising providing authentication credentials for use by the superbloc application, wherein the authentication credentials enable the function block to use the one or more servers for the real-time communication session.***

204. Abuan in view of Guzman renders obvious this claim.

205. As discussed above in [9.0] and reflected in Figure 4, below, Guzman describes a user inputting a "username" and "password" (*authorization credentials*) into an "overlay display area" of a "video conferencing application." Ex.1007, [0050].

“superblock application”



**UI fields for application
to receive
“authorization
credentials”**

Ex.1007, Fig. 4 (partial annotated)

206. “[W]hen the user selects the “Sign In” item 465, the video conferencing application verifies whether the login information entered by the user is valid.” Ex.1007, [0077]. Guzman describes different ways to verify and authenticate the user’s credentials. First, the credentials can be verified by the video conferencing application itself. Ex.1007, [0057]. Second, the credentials can be verified “by accessing a remote server that stores login information.” Ex.1007, [0057].

207. For instance, Guzman’s video conferencing application 2200 includes “an account setup and verification module 2210” (a “*function block*”) that facilitates a user logging into the videoconferencing server. Ex.1007, [0222], [0230]. In one example, a “user inputs an account name and password which is **passed by the user interface interaction module 2205 to the account setup and verification module 2210.**” Ex.1007, [0230].

208. Thus, Guzman’s disclosure of verifying login credentials by accessing a remote server discloses “*providing authorization credentials for use by the superblock application.*” Further, the user interface interaction module 2205 (*function block*) receives the authorization credentials from the UI of the video conferencing application and facilitate login with the videoconferencing server (*the authentication credentials enable the function block to use the one or more servers for the real-time communication session*). When the credentials have been verified, the user’s contacts are loaded and made available so that the user can initiate a video conference. Ex.1007, [0058], Fig. 4.

5. Independent Claim 28

209. Independent claim 28 is substantially similar to independent claim 1, with limitations [28.0] through [28.3] being word-for-word identical to limitations [1.0] through [1.3], respectively. Limitation [28.4] is substantially similar to claim [10.0], and is addressed below. Dependent claim 29 is substantially similar to

limitation [1.4], except it does not recite the language “*compiled into the superblock application*” found in limitation [1.4]. Dependent claims 30-32 are substantially similar to claims 2-4, respectively, claim 36 is substantially similar to claim 5, claims 33-37 are substantially similar to claims 6-9, claims 38-41 are substantially similar to claims 11-14, claims 43-47 are substantially similar to claims 16-20, and claims 49-54 are substantially similar to claims 22-27. Accordingly, claims 28-41, 43-47, and 49-54 obvious for the same reasons as discussed above. See Ex.1047 (comparing claims 1-27 with claims 28-54 in redline).

- a. **[28.0] *A method for providing a real-time communication session over the internet for a superblock application intended for use on a computing device, the method comprising:***

210. See [1.0] above.

- b. **[28.1] *providing a function block for use in adding additional functionality to a third party superblock application that has its own functionality and display window,***

211. See [1.1] above.

- c. **[28.2] *wherein the function block is configured to be compiled into the superblock application and is configured to add the additional functionality to provide the real-time communication session using one or more servers connected over the internet, and***

212. See [1.2] above.

- d. **[28.3] *wherein the function block is configured to interact with the superblock application through one or***

*more application programming interface (API) calls;
and*

213. See [1.3] above.

- e. **[28.4] *providing authentication credentials for use by the superblock application, wherein the authentication credentials enable the function block to use the one or more servers to establish the real-time communication session for the superblock application.***

214. Limitation [28.4] is substantially similar to claim 10. As discussed above at [10.0], Abuan and Guzman render obvious “*providing authentication credentials for use by the superblock application, wherein the authentication credentials enable the function block to use the one or more servers.*”

215. Limitation [28.4] further recites “*to establish the real-time communication session for the superblock application.*” As discussed for [1.4], Abuan explains, with reference to Figure 7 above below, that a “video conference server 715 forwards (at operation 3) [a] video conference request [from device 705] to the video conference client 725 of the device 720” to establish a video conference between device 705 and device 720 (*establish the real-time communication session*). Ex.1005, [0110]-[0111]. This communication between the video conference client 725—operating on device 720—and the video conference server 715, is performed by the video conference module 1302 on that device, such that *the function block [] use[s] the one or more servers* via communications over the network. More specifically, Abuan discloses that “[e]ach of the video conference clients 710 and

725...communicates with the video conference server 715 over a network (e.g., a cellular network, a local area network, a wireless network, a network of networks, the Internet etc.) through a network interface such as the network interface 650.” Ex.1005, [0109]. Abuan explains that the networking manager 1314 of the video conference module 1302 (or network layer 640 of the video conference module 625) “establishes the connections between the dual camera mobile device and the other device of the video conference at the start of the video conference.” Ex.1005, [0179]; *see also* [0104]; Ex.1005, [0156] (“A more detailed version of this video conference module [625] will be described below by reference to FIG. 13.”), [0171] (“the client application 1365 is the same as the video conference client 645 of FIG. 6.”). In this way, the video conferencing module 1302 (*the function block*) establishes the video conference session *establish the real-time communication session*) between the video conference client of device 705 (*the superblock application*) and device 720 using video conference server 715 (*use the one or more servers*).

6. Claim 29

- a. **[29.0] *The method of claim 28 further comprising enabling the establishment of the real-time communication session between the one or more servers and the function block so that the function block can provide the real-time communication session to the superblock application.***

216. See [1.4] above.

7. Claim 30

- a. **[30.0]** *The method of claim 29 wherein enabling the establishment of the real-time communication session includes signaling communications between the one or more servers and the function block.*

217. See [2.0] above.

8. Claim 31

- a. **[31.0]** *The method of claim 30 wherein the signaling communications include session setup, management, and teardown.*

218. See [3.0] above.

9. Claim 32

- a. **[32.0]** *The method of claim 30 wherein the signaling communications use Session Initiation Protocol (SIP) as a signaling protocol.*

219. See [4.0] above.

10. Claim 33

- a. **[33.0]** *The method of claim 29 wherein enabling the establishment of the real-time communication session includes negotiating signaling and media parameters between the one or more servers and the function block.*

220. See [6.0] above.

11. Claim 34

- a. **[34.0]** *The method of claim 33 wherein the signaling and media parameters include a bandwidth parameter.*

221. See [7.0] above.

12. Claim 35

- a. **[35.0] *The method of claim 33 wherein the signaling and media parameters include a codec parameter.***

222. See [8.0] above.

13. Claim 36

- a. **[36.0] *The method of claim 28 wherein the real-time communication session uses Real-time Transport Protocol (RTP) as a data transport protocol.***

223. See [5.0] above.

14. Claim 37

- a. **[37.0] *The method of claim 28 further comprising providing an authorization key for use by the superblock application, wherein the authorization key unlocks the function block for use by the superblock application.***

224. See [9.0] above.

15. Claim 38

- a. **[38.0] *The method of claim 28 further comprising sending a notification from the one or more servers to the function block.***

225. See [11.0] above.

16. Claim 39

- a. **[39.0] *The method of claim 38 wherein the notification requires a response and the establishment of the real-time communication session is the response to the notification.***

226. See [12.0] above.

17. Claim 40

- a. **[40.0]** *The method of claim 38 wherein the notification is a request to initiate the real-time communication session, and an API call indicates whether the request has been granted or denied by the superblock application.*

227. See [13.0] above.

18. Claim 41

- a. **[41.0]** *The method of claim 38 wherein the notification is a presence notification.*

228. See [14.0] above.

19. Claim 43

- a. **[43.0]** *The method of claim 28 further comprising receiving, by the one or more servers, a notification from the function block.*

229. See [16.0] above.

20. Claim 44

- a. **[44.0]** *The method of claim 28 wherein the real-time communication session is for instant messaging.*

230. See [17.0] above.

21. Claim 45

- a. **[45.0]** *The method of claim 28 wherein the real-time communication session is a voice call.*

231. See [18.0] above.

22. Claim 46

- a. **[46.0]** *The method of claim 28 wherein the real-time communication session is a video call.*

232. See [19.0] above.

23. Claim 47

- a. **[47.0]** *The method of claim 28 wherein the superblock application is further adapted to operate with iOS as an operating system for the computing device.*

233. See [20.0] above.

24. Claim 49

- a. **[49.0]** *The method of claim 28 wherein the computing device is a cellular phone.*

234. See [22.0] above.

25. Claim 50

- a. **[50.0]** *The method of claim 28 wherein the computing device is a smart phone.*

235. See [23.0] above.

26. Claim 51

- a. **[51.0]** *The method of claim 28 wherein the computing device is a tablet.*

236. See [24.0] above.

27. Claim 52

- a. **[52.0]** *The method of claim 28 wherein the computing device is a personal digital assistant.*

237. See [25.0] above.

28. Claim 53

- a. **[53.0] *The method of claim 28 wherein the computing device is a laptop computer.***

238. See [26.0] above.

29. Claim 54

- a. **[54.0] *The method of claim 28 wherein the computing device is a desktop computer.***

239. See [27.0] above.

E. Ground 5: Claim 42 would have been Obvious over Abuan in view of Platt, Guzman, and Eisenberg.

240. Claim 42 depends, indirectly, from claim 28, which is obvious over Abuan and Guzman as discussed above in Ground 4. Claim 42 further recites the same limitation recited in claim 15, shown to be obvious over Abuan above in Ground 1. Thus, claim 42 would have been obvious for the same reasons discussed above for claims 28 and 15. *See* Ex.1047 (comparing claims 1-27 with claims 28-54 in redline).

1. Claim 42

- a. **[42.0] *The method of claim 41 wherein the function block is further configured to update a presence indicator within the superblock application in response to receiving the presence notification.***

241. See [15.0] above.

F. Ground 6: Claim 48 would have been Obvious over Abuan in view of Platt, Guzman, and Beilis.

242. Claim 48 depends from claim 28, which is obvious over Abuan and Guzman as discussed above in Ground 4. Claim 48 further recites the same limitation recited in claim 21, shown to be obvious over Abuan above in Ground 1. Thus, claim 48 would have been obvious for the same reasons discussed above for claims 28 and 21. *See* Ex.1047 (comparing claims 1-27 with claims 28-54 in redline).

1. Claim 48

- a.** [48.0] *The method of claim 28 wherein the superblock application is further adapted to operate with Android as an operating system for the computing device.*

243. See [21.0] above.

IX. CONCLUSION

244. I declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true, and that these statements were made with knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under section 1001 of Title 18 of the United States Code.

Dated: January 13, 2026


Bruce McNair