

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

AMAZON.COM, INC., AMAZON.COM SERVICES LLC,
Petitioner,

v.

NOKIA TECHNOLOGIES OY,
Patent Owner.

Case IPR2024-01043
Patent 8,918,741

DECLARATION OF MICHAEL KOGAN

TABLE OF CONTENTS

TABLE OF CONTENTS.....	i
I. INTRODUCTION	1
II. QUALIFICATIONS	1
A. Materials Considered.....	6
III. PERSON OF ORDINARY SKILL IN THE ART (POSITA)	7
IV. LEGAL STANDARD FOR PATENTABILITY	8
A. Claim Interpretation	9
B. Prior Art.....	10
C. Obviousness.....	10
V. OVERVIEW OF THE '741 PATENT	15
VI. PROSECUTION HISTORY OF THE '741 PATENT.....	19
VII. PRIOR ART REFERENCES	23
A. Hassan (Ex. 1004)	23
B. Becker (Ex. 1006).....	25
VIII. CLAIM CONSTRUCTION	28
IX. ARGUMENT.....	29
A. Grounds 1 and 2 – Hassan does not render obvious claims 1, 5-7, 10-11, 15, 17, and 19 of the '741 Patent and Hassan and Nirhamo do not render obvious claims 1-20 of the '741 Patent	31
B. Grounds 3 and 4 – Becker does not render obvious claims 1, 5-7, 10-11, 15, 17, and 19 of the '741 Patent and Becker and Nirhamo do not render obvious claims 1-20 of the '741 Patent.	47
X. CONCLUSION.....	59

I, Dr. Michael Kogan, declare as follows:

I. INTRODUCTION

1. I have been retained by Patent Owner Nokia Technologies Oy (“Patent Owner”) to provide expert opinions in connection with this *inter partes* review.

2. I am being paid at my standard consulting rate of \$650 per hour for my work in this matter. Additionally, I will be reimbursed for all expenses incurred. My compensation is not dependent on the outcome of this action, and I have no other interests in this matter or with any of the parties involved.

3. My opinions are also based on my education, training, research, experience, knowledge, and personal and professional experience, as well as my investigation and study of relevant materials.

II. QUALIFICATIONS

4. My qualifications for forming the opinions set forth in this report are listed in this section and provided in my *curriculum vitae* (“CV”), attached here as Exhibit 2002. *See* Ex. 2002. My CV also includes a list of my publications.

5. Based on my work experience, described in detail in my curriculum vitae, I am knowledgeable about operating systems software at all levels, including PC and embedded systems architecture, multitasking, hardware and device drivers, power management, memory management, inter-process communications,

program loading, and graphical user interfaces (GUIs). I have experience with operating system development in connection with the following operating systems: CP/M, DOS, OS/2, Windows, a wide variety of UNIX variants including XENIX, PC/IX, AIX, Mach, ATT Unix, Berkeley Unix, Linux, Android, and iOS, as well as numerous embedded and real-time operating systems.

6. I am an expert in PC and embedded system software with specialized skills in operating systems, system software on microprocessor-based platforms including the areas of operating system kernels, multitasking and inter-process communications, memory management and protection, I/O programming and device drivers, system initialization and boot code, power management, communications, cross-platform portability, cross-platform development, and GUI subsystems.

7. I am knowledgeable and skilled in the development of consumer electronics platforms, from both a hardware and operating system perspective. I have designed and implemented embedded operating systems and reference platforms for the development of one- and two-way pagers, smart phones, PDAs, and various general consumer-device architectures and SoCs. I have significant experience with embedded operating systems including OS9, OS9000, pSOS, QNX, MDSOS, MEMOS, eCos, VxWorks, Android, and iOS.

8. While at IBM from 1985-1992, I served as a developer, designer, and ultimately as chief architect of the OS/2 operating system products. I am responsible for two patents and numerous invention disclosures pertaining to multitasking operating systems technology.

9. From 1985-1987, I was a lead developer for the design, development, and testing of 16-bit OS/2. My main areas of responsibility included device drivers, memory management, debugging, queues, DOS compatibility, system initialization, and BIOS support.

10. From 1987-1991, I was the chief architect for the 32-bit version of OS/2, which shipped in 1992 as OS/2 2.0. My responsibilities included overall design responsibility for the kernel, subsystems such as Presentation Manager, 32-bit API, compatibility architecture, and performance. I implemented prototype 32-bit systems and provided new designs in multitasking, memory management, inter-process communications, and compatibility areas. I also provided high-level and low-level design documentation, complete product specifications, invention disclosures, and patent filings.

11. From 1991-1992, I was technical staff for the OS/2 programming area. I resolved key design and implementation issues for OS/2 2.0 delivery, provided technical and strategic briefings for customers, and was a technical interface to trade press and independent software vendors. I provided technical

direction for the implementation of OS/2 multimedia extensions and performed advanced technical research and prototyping for future systems including multiprocessing prototypes.

12. In 1992, I co-authored the book *The Design of OS/2*, which describes the motivation, design, and internals of the 16-bit and 32-bit OS/2 systems. It includes deep details on OS/2 including multitasking, inter-process communications, and GUI subsystems.

13. From 1992-1995, after leaving IBM, I consulted in the OS/2 market providing both design and development services to clients utilizing OS/2. I also was featured as a speaker at OS/2 technical seminars and public forums and wrote for several magazines.

14. While at Motorola from 1995 to 1996, I was the development lead for implementing Microware OS9 and OS9000 on advanced paging platforms and ported the OS9 operating system to numerous Motorola platforms.

15. While at Ratio Design Labs from 1996 to 1997, I led a small group responsible for rewriting an embedded operating system known as MDSOS for the ARM, Motorola, and Intel embedded platforms. This experience includes some of the first embedded devices to utilize advanced system-on-chip architectures with high integration in combination with Flash memory technology, advanced power management, and graphical LCD displays. As part of my work while at Ratio

Design Labs, I supported the development of AirScript, which enabled GUIs on platforms with advanced graphical LCD displays.

16. While at Cygnus Solutions from 1997 to 1998, my responsibilities included engineering direction and management for Cygnus's cross-platform GUI Integrated Development Environment (IDE) product, and Cygnus's board support tools for enabling rapid software development on multiple embedded hardware platforms.

17. Outside the aforementioned engagements, I have written numerous custom device drivers and subsystems for consulting clients in many operating system environments and have been involved in the architecture, design, and implementation of numerous desktop products and embedded devices. I also have significant experience building client-server applications, dynamic websites, web applications, web services, and mobile applications.

18. In addition to my knowledge and experience with how Operating Systems are designed and implemented, I also have extensive experience with and knowledge of subsystems, frameworks, and applications which run in conjunction with the operating system, how they are designed and implemented, and how they operate. As a lead operating system developer and lead designer, my understanding of how these other software components operate has enabled me to provide not only designs and implementations of operating systems services to enable their

operation, but also to provide recommendations on designs and implementations of frameworks and applications to enable them to provide functionality and services optimally.

19. In all of my experiences with both desktop operating systems and embedded operating systems in consumer devices, I have had extensive experience designing and implementing multitasking, scheduling, and I/O and device management including interrupt dispatching and handling.

20. I received my Masters of Science degree in Computer Science from Nova University in 1986, and my Doctor of Sciences degree in Computer Science from Nova University in 1991.

21. My experience coupled with my education enables my full understanding of the methods and technologies addressed by the '741 patent.

A. Materials Considered

22. My opinions are based on my education, training, research, knowledge, and personal and professional experience. In connection with this report, I have reviewed the following:

- Ex. 1001 – U.S. Patent No. 8,918,741 (“the '741 patent”)
- Ex. 1002 – Prosecution History of the '741 patent
- Ex. 1003 – Declaration of Benjamin B. Bederson, Ph.D. (“Bederson”)
- Ex. 1004 – U.S. Patent Application Publication No. 2007/0180492 (“Hassan”)

- Ex. 1005 – U.S. Patent Application Publication No. 2006/0246955 (“Nirhamo”)
- Ex. 1006 – U.S. Patent No. 7,689,939 (“Becker”)
- Ex. 1007 – U.S. Patent Application Publication No. 2008/0222545 (“Lemay”)
- Ex. 1010 – European patent No. 0664020B1 (“Matheny”)
- Ex. 1011 – Macintosh Human Interface Guidelines, Apple, 1992
- Ex. 1012 – The New Oxford American Dictionary, 2nd Edition, 2005
- Ex. 1013 – Webster’s College Dictionary, 2005
- Paper 1 – Petition for Inter Partes Review of U.S. Patent No. 8,918,741

III. PERSON OF ORDINARY SKILL IN THE ART (POSITA)

23. I understand that the meanings of terms should be interpreted from the perspective of an individual with average skills in the relevant field.

24. I understand that the level of ordinary skill in the art may depend on factors such as: (A) the types of problems encountered in the field; (B) existing solutions to those problems; (C) how quickly innovations are made; (D) the complexity of the technology; and (E) the educational level of workers in the field. It's important to note that not every factor may be present in a specific situation, and some factors may be more significant than others.

25. A person with ordinary skill in the art would have at least a bachelor’s degree in electrical engineering, computer engineering, or a related field, with two to three years of experience in user interfaces, operating systems, and device power

management. More work or practical experience may qualify one not having the requisite education as a person with ordinary skill in the art, while a higher level of education could offset less experience.

26. I understand that Petitioner takes the position that the field of the '741 patent is user interfaces and that a person with ordinary skill in the art would have had a Bachelor's degree in electrical engineering, computing engineering, computer science, or similar field with two to three years of experience in user interfaces or a master's degree in electrical engineering, computer engineering, computer science, or a similar field with a specialization in user interfaces. I also understand that person with less education but more relevant practical experience may also meet the Petitioner's standard.

27. I disagree with the level of ordinary skill in the art for the Challenged Patent that Petitioner proposes, although the proposed definitions are very similar. I have surpassed both the Patent Owner's and the Petitioner's levels of ordinary skill in the art as of the conception date of the '741 Patent. Therefore, regardless of which level of skill in the art is applied, my analysis and conclusions in this report remain unchanged.

IV. LEGAL STANDARD FOR PATENTABILITY

28. I use the legal principles below, of which counsel informed me, as a guide in formulating my opinions.

A. Claim Interpretation

29. My opinions concern what I believe the person of ordinary skill in the art would have understood the meaning of specific claim terms to be based on the patent documents.

30. I have learned that evaluating the legitimacy of a patent claim involves a two-step process. The first step entails interpreting the claim language accurately to establish its scope and significance. The second step involves comparing the properly interpreted claim to the alleged prior art to determine if the elements of the claim are satisfied by the prior art.

31. In order to understand the meaning of claim phrases, I began by carefully studying the patent. I focused on the claims themselves, followed by the background, detailed specifications, figures, and other patent content. After that, I examined the prosecution history to identify any clarifications or limitations that might be associated with claim terms. In certain cases, I also referred to other documents, such as references cited by the patent office.

32. I understand that during an inter partes review, claim terms should be interpreted based on their ordinary and customary meaning as understood by a person of ordinary skill in the relevant field, considering the entire disclosure at the time of the invention. It's important not to interpret a specific example from the written description as part of the claim if the claim language is broader.

Additionally, any special definition for a claim term must be clearly, deliberately, and precisely stated.

33. I have been informed that this standard is the same as that used in United States district courts, in which claim terms are given their ordinary and customary meaning as would be understood by a person of ordinary skill in the art at the time of the invention having taken into consideration the language of the claims, the specification, and the prosecution history of record under the *Phillips* claim construction standard.

B. Prior Art

34. It has been explained to me by Counsel that information that satisfies one of the categories of prior art outlined in 35 U.S.C. § 102 (pre-AIA) may be used in an invalidity analysis under §§ 102 or 103. If information is not classified correctly as prior art under one of the subsections of § 102 of the Patent Code, it may not form the basis of an anticipation or obviousness determination.

C. Obviousness

35. I have been informed that a patent claim is considered invalid under 35 U.S.C. § 103 (pre-AIA) if it would have been obvious to a person of ordinary skill in the relevant field in view of the prior art at the time the application was filed. In simpler terms, even if the claimed invention is not exactly described in a single prior art reference, a patent claim can be invalidated for obviousness if the

differences between the claimed invention and the prior art are such that the overall subject matter would have been obvious at the time the invention was made to a person with ordinary skill in the relevant field.

36. I have been informed that when making a determination of obviousness, the following factors are considered: (i) determining the scope and content of the prior art; (ii) ascertaining the differences between the prior art and the claims at issue; and (iii) resolving the level of ordinary skill in the pertinent art.

37. I understand that whether there are any relevant differences between the prior art and the claimed invention is to be analyzed from the view of a person of ordinary skill in the art at the time of the invention. For purposes of this IPR proceeding, I have been informed to assume that the alleged date of invention for the challenged claims is no later than June 29, 2007.

38. In analyzing the relevance of the differences between the claimed invention and the prior art, I understand that I must consider the impact, if any, of such differences on the obviousness or non-obviousness of the invention as a whole, not merely some portion of it.

39. The person of ordinary skill faced with a problem can utilize their experience and abilities to solve the problem and may also refer to any available prior art to aid in the solution.

40. I understand that a reference may be modified and/or combined with one or more other references and/or with the knowledge of one of ordinary skill in the art if such person of ordinary skill in the art would have found the modification and/or combination obvious. In other words, I understand that I do not need to look for precise teaching in the prior art directed to the subject matter of the claimed invention. I understand that I may take into account the inferences and creative steps that a person of ordinary skill in the art would have employed in reviewing the prior art at the time of the invention.

41. In determining whether a piece of prior art could have been combined with other prior art or with other information within the knowledge of a person having ordinary skill in the art, it is my understanding that the following are examples of approaches and rationales that may be considered to support a conclusion of obviousness:

- Combining prior art elements according to known methods to yield predictable results;
- Simple substitution of one known element for another to obtain predictable results;
- Use of a known technique to improve similar devices, methods, or products in the same way;
- Applying a known technique to a known device, method, or product ready for improvement to yield predictable results;

- Applying a technique or approach that would have been “obvious to try” (choosing from a finite number of identified, predictable solutions, with a reasonable expectation of success);
- Known work in one field of endeavor may prompt variations of it for use in either the same field or a different one based on design incentives or other market forces if the variations would have been predictable to a person having ordinary skill in the art; or
- Some teaching, suggestion, or motivation in the prior art that would have led one of ordinary skill to modify the prior art reference or to combine prior art reference teachings to arrive at the claimed invention.

42. I understand that if the claimed invention combined elements known in the prior art and the combination yielded results that were predictable to a person of ordinary skill in the art at the time of the invention, then this evidence would make it more likely that the claim was obvious.

43. I understand that an invention would be considered obvious if a person with ordinary skills in the relevant field would have recognized an obvious benefit to the solutions attempted by the applicant, given the wide range of needs arising from advancements in the field. When there is a need for a design or market pressure to solve a problem and only a finite number of identified, predictable solutions exist, it would be obvious to a person with ordinary skills to try the known options. If a technique has been used to improve one device, and a person

with ordinary skills in the field would recognize that it could improve similar devices in the same way, using the technique would be considered obvious.

44. I understand that a key inquiry is whether any improvement allegedly achieved by a claimed invention is more than the predictable use of prior art elements according to their established functions.

45. I understand that a claimed invention may be obvious if some teaching, suggestion, or motivation exists that would have led a person of ordinary skill in the art to combine the invalidating references. I also understand that this suggestion or motivation may come from sources such as explicit statements in the prior art, or from the knowledge of a person having ordinary skill in the art. Alternatively, any need or problem known in the field at the time and addressed by the patent may provide a reason for combining elements of the prior art.

46. I also understand that when there is a design need or market pressure, and there are a finite number of predictable solutions, a person of ordinary skill may be motivated to apply both his skill and common sense in trying to combine the known options in order to solve the problem.

47. I understand that a reason to modify or combine prior art may support an obviousness determination, provided there is a reasonable expectation of success. I further understand that whether a proposed modification or combination

of the prior art has a reasonable expectation of success is determined at the time the invention was made.

48. On the other hand, if the combination of known elements yielded unexpected or unpredictable results, or if the prior art teaches away from combining the known elements, then this evidence would make it more likely that the claim that successfully combined those elements was not obvious. I understand that hindsight must not be used when comparing the prior art to the invention for obviousness.

49. I am also aware that before concluding a claim is obvious, secondary considerations of non-obviousness, to the extent they exist, must be considered, such as: long-felt need, failure of others, copying by others, praise by others in the industry, licensing of the claimed invention, departure from accepted principles, widespread recognition by those in the art of the invention's significance, and commercial success. I understand that there must be a demonstrated nexus between the claimed invention and these secondary considerations for them to be considered.

V. OVERVIEW OF THE '741 PATENT

50. The '741 Patent, titled "Unlocking a touch screen device," relates generally to "user interfaces and in particular to transitioning from an idle/locked state of a device to an active/unlocked state of a device." Ex. 1001, 1:15-17.

51. The '741 Patent explains that mobile devices (e.g., mobile communication devices) generally include a variety of applications for tasks such as Internet communications, instant messaging capabilities, email facilities, web browsing and searching. *Id.*, 1:19-22. Such mobile devices include a locked mode to avoid inadvertent activation of applications. *Id.*, at 1:22-25. The '741 Patent further explains that, in existing systems, user inputs to the device are ignored in the idle or locked mode except to unlock the device. *Id.*, 1:26-28.

52. In the '741 Patent, the screen or user interface can be used to indicate whether an application is in an idle or locked mode. *Id.*, 3:24-25; *see also id.*, 4:13-14. For example, the icon for an application may appear relatively dimmed, shaded, less highlighted, or deemphasized. *Id.*, 3:26-27; 3:31-33; *see also id.*, 3:50-55. As an example, in FIG. 2A of the '741 Patent, the dimmed icon 202 represents a time or clock application. *Id.*, FIG 2A. The '741 Patent explains that, in the dimmed mode, the device is locked and it may not be possible to view any functions of the application, such as for example, the current time from the clock application. *Id.*, 6:60-63.

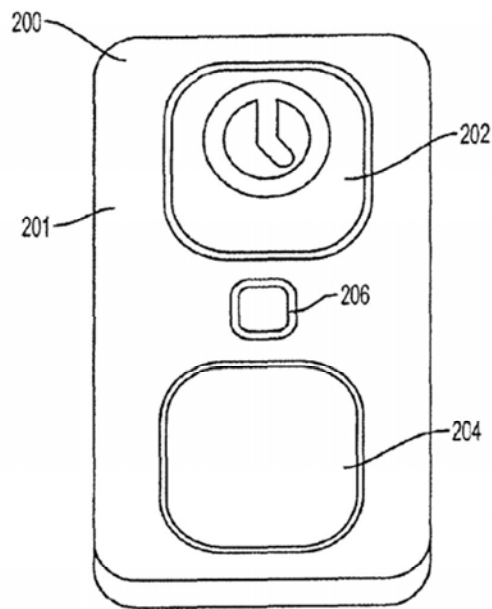


FIG. 2A

Id., FIG. 2A.

53. The '741 Patent provides the ability for a user to select an icon for an application to determine a state, utility, or function of the underlying application without the need to unlock the device or open the application. *Id.*, 6:56-59. For example, the '741 Patent discloses features that allow the user to interact with an icon to temporarily illuminate or otherwise highlight the icon. *Id.*, 6:64-66; FIG. 3. An example of this is shown for clock application icon 302 in FIG. 3 of the '741 Patent, which continues from the previous example of the clock application icon 202 from FIG. 2A of the '741 Patent.

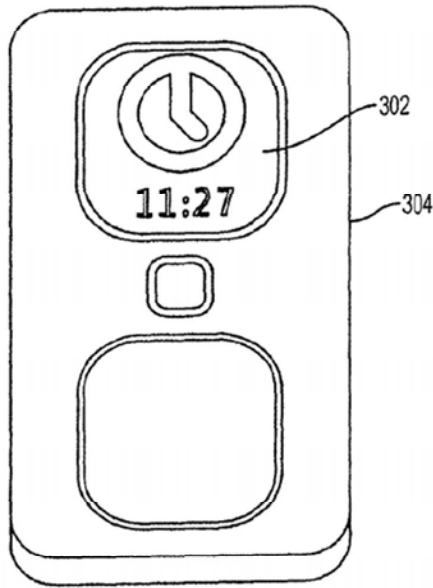


FIG. 3

Id., FIG. 3.

54. The '741 Patent explains that the state described above with respect to clock application icon 302 in FIG. 3 of the '741 Patent is referred to as a “passive active” or “semi-active” state. *Id.*, 6:66-7:2. This state allows for temporary activation of high level user interface elements of an application such as the clock. *Id.* Continuing with the example shown in FIG. 3 of the '741 Patent, in the passive (locked) active state, the clock application icon 302 is illuminated when the clock application icon 302 is selected by the user and the current time is then available and can be viewed. *Id.*, 7:2-6. In this passive active or “semi-active” state, the

clock function is illuminated temporarily, but does not become fully active. *Id.*, 7:8-9. The '741 Patent further explains that the application icon can remain illuminated for a predetermined period of time, for example five seconds. *Id.*, 7:9-11. This feature of the '741 Patent is one example that allows for quick activation of high-level user interface elements for passive information gathering, while the device remains in the idle and locked mode. *Id.*, 7:11-13.

VI. PROSECUTION HISTORY OF THE '741 PATENT

55. U.S. Patent Application No. 13/889,750 (“the '750 Application”) was filed May 8, 2023, as a continuation of U.S. Application No. 13/357,143, filed on January 24, 2012, which is a division of U.S. Application No. 11/770,929 filed on June 29, 2007 (now U.S. Patent No. 8,127,254). *Ex. 1002*, 53; 15-16.

56. On October 15, 2013, the Examiner issued a Non-Final Office Action rejecting claim 5 under 35 U.S.C. § 112(b) as allegedly being indefinite. *Id.*, 69. The Examiner also rejected claims 1, 7, 11-13, and 17-19 under 35 U.S.C. § 102(e) as allegedly being anticipated by U.S. Patent Application No. 2009/0205041 (“Michalske”). *Id.*, 69-71. The Examiner also rejected claims 2, 8-10, 14, and 20 under 35 U.S.C §103(a) as allegedly being obvious over Michalske in view of U.S. Patent Application No. 2007/0150842 (“Chaudhri”). *Id.*, 71-72. The Examiner also rejected claims 3, 6, and 15 under 35 U.S.C §103(a) as allegedly being obvious

over Michalske in view of U.S. Patent Application No. 2010/0070931 (“Nichols”).
Id., 73-74.

57. In response, Applicants filed a response to the Non-Final Office Action on January 15, 2014. *Id.*, 85-88. In the office action response, Applicants explained that the ’750 Application has a priority date of June 29, 2007, and argued that neither Michalske nor Nichols qualifies as prior art under 35 U.S.C. § 102 since they were filed after the priority date of the ’750 Application. *Id.*, 86-87.

58. On May 20, 2014, the Examiner issued a Non-Final Office Action rejecting claim 5 under 35 U.S.C. § 112(b) as allegedly being indefinite. *Id.*, 118-126. The Examiner also rejected claims 1, 7-9, 12, 13, 16, 17, and 19 under 35 U.S.C. § 102(e) as allegedly being anticipated by U.S. Patent Application No. 2007/0016958 (“Bodepudi”). *Id.*, 709-711. The Examiner also rejected claims 3-5, 11, 15, and 18 under 35 U.S.C §103(a) as allegedly being obvious over Bodepudi in view of U.S. Patent Application No. 2007/0184112 (“Chiang”). *Id.*, 711-712. The Examiner also rejected claims 2, 6, 14, and 20 under 35 U.S.C §103(a) as allegedly being obvious over Bodepudi in view of U.S. Patent Application No. 2007/0150842 (“Chaudhri”). *Id.*, 712-713. The Examiner also rejected claim 6 under 35 U.S.C §103(a) as allegedly being obvious over Bodepudi in view of Chiang and Chaudhri. *Id.*, 713. The Examiner also rejected claim 10 under 35

U.S.C §103(a) as allegedly being obvious over Bodepudi in view of U.S. Patent Application No. 2007/0120689 (“Zerhusen”). *Id.*, 713-714.

59. In particular, and with respect to “enabling a user to select the object while the device is in the locked mode, in order to determine a state, function, or utility of the application while the device remains in the locked mode,” the

Examiner cited paragraph 0051 of Bodepudi, which states:

Referring now to FIG. 6, an illustrative diagram shows a user interface during selective lock mode. As illustrated, a user interface 600 during selective lock mode, as compared with user interface 500 during normal mode, only includes a selective lock folder window 620 representing selective lock folder 408 with a selectable link 622 associated with application A and a selectable link 624 associated with application D. Responsive to a non-authorized user selection of application A, selective lock application 208 calls a wrapper function that opens application A, as illustrated by application A window 602 within user interface 600 through which a user has full access to the functions of application A.

Ex. 2003, ¶0051.

60. In response, Applicants filed a response to the Non-Final Office Action on June 23, 2014. Ex. 1002, 744-749. In the office action response, Applicants amended the claims to recite “enabling a user to select the ~~icon~~ object while the device is in the locked mode, in order to determine a state, function, or utility of the application without fully activating the application while the device

remains in the locked mode.” *Id.*, 748-749. Applicants argued that this claim element was not taught or suggested by Bodepudi or any of the cited references. *Id.* Applicants noted that support for this claim amendment is provided in at least paragraph [0036] of the published application, which is column 6: line 56 – column 7, line 13 of the issued patent and is provided below for reference.

Referring to FIGS. 2A and 3, in one embodiment, the user can select or contact the dimmed icons 202 and 204 to determine a state, utility or function of the underlying application, without the need to unlock the device or open the application. For example, in FIG. 2A, dimmed icon 202 represents a time or clock application. In the dimmed mode, when the device is locked, it may not be possible to view any functions of the application, such as for example, the current time. However, the features of the disclosed embodiments allow the icon 202 to be contacted to temporarily illuminate or somehow highlight the icon, as illustrated by icon 302 in FIG. 3. This state, which might be described as a passive active state, allow for temporary activation of an application, such as a clock. This is also referred to herein as the “semi-active” state. As shown in FIG. 3, the clock application icon 302 is illuminated when that area is selected, and the current time is available and can be viewed, particularly against the dimmed and/or darkened background 304. In alternate embodiments, any suitable way to temporarily highlight the selected icon can be implemented. The clock function is illuminated temporarily, but does not become fully active. The icon 302 can remain illuminated for a pre-determined period of time, such as for example, 5 seconds. This allows the quick

activation of high level user interface elements for passive information gathering, while the device remains in the idle and locked mode.

Ex. 1001, 6:56 – 7:13.

61. On June 24, 2014, the Examiner issued an initial Notice of Allowance in response to Applicant's arguments without providing further comments. *Id.*, 757. After receiving the Notice of Allowance, Applicants filed an Information Disclosure Statement and Request for Continued Examination on October 24, 2014. *Id.*, 824. On November 18, 2014, the Examiner issued a subsequent Notice of Allowance. *Id.*, 835.

VII. PRIOR ART REFERENCES

A. Hassan (Ex. 1004)

62. Hassan relates to “electronic user interactive devices and, in particular, to methods and systems for enabling sharing of such devices with other users without compromising security.” Ex. 1004, ¶0001.

63. Hassan explains that security is an issue with most sophisticated electronic devices which include handheld and mobile devices. *Id.*, ¶0002. Overtime, these devices began to offer greater levels of functionality and a wider range of applications, which has led to a greater desire to share these devices with others for limited purposes. *Id.*

64. In Hassan, a device is placed in a “locked state” that has an associated set of permitted tasks that are available for use during the “locked state.” *Id.*,

¶0013. This configuration allows the device owner to share the device with others but maintains security over aspects of the device. *Id.*

65. Hassan discloses using profile data or user profiles to indicate which applications a user can open or which tasks within a running application that a user is permitted to execute. *Id.*, ¶0040; ¶0070. Hassan provides several examples of how user profiles can be used to specify restrictions associated with a user. *Id.*, ¶0040; *see also id.* ¶0002. For example, a user profile may only permit a user to use a voice call application. *Id.*, ¶0040; *see also id.* ¶0002. In this example, the user associated with the user profile is only permitted by the task changer to use the voice call application and is prohibited from opening or accessing other applications on the device.

66. In another example, a user profile may only permit the user to open game applications. *Id.* ¶0040; *see also id.* ¶0002. In this example, the user associated with the user profile is only permitted by the task changer to use gaming applications and is prohibited from opening or accessing other applications on the device. In these examples, Hassan’s “locked state” discloses the ability to allow other users to have full access to some applications while preventing access to other applications on the device.

67. Hassan teaches task change APIs (application programming interfaces) which are implemented by “a process control level of the operating

system to determine whether the requested task change is permitted.” *Id.*, ¶0013.

These task change APIs are used by task changers, such as “a task switcher or window launcher” to evaluate whether the task change is permitted according to the data in the user profiles or profile data during the “locked state.” *Id.*

68. Hassan also provides examples using a task switcher to limit which tasks, using a task changer, within an open application a user is permitted to access. *Id.*, ¶0071. As an example, Hassan discloses that a user may be prohibited from accessing particular tasks or functions within an email application by the task changer. *Id.* In this example, Hassan discloses that while the user is able to open and access to the email application, the user can be prohibited from performing operations such as forwarding, replying to, composing, opening, deleting, moving, or otherwise dealing with messages because of the task changer. *Id.*

69. Overall, Hassan is directed to a process that relies on a task changer for controlling access to applications on a device for other users by using user profiles that specify which applications a user is able to open and/or which tasks a user can perform within an open application while the device is in the “locked state.”

B. Becker (Ex. 1006)

70. Becker relates to “screen blankers” (e.g., screen savers) and in particular “an application communicating with a user when a screen blanker is

activated.” Ex. 1006, 1:6-8. A “screen blanker” or “screen saver” is a program that automatically blanks the screen of a computer and/or displays a selected image or collection of images when the computer has not been used or has been idle for a selected period of time and the monitor remains activated. *Id.*, 1:50-57.

71. Becker is primarily directed toward an application where an IP softphone (e.g., Voice over IP (VoIP)) is running on a personal computer (PC). *Id.*, 1:12-13. An IP softphone includes software that emulates a telephone and communications signaling and voice information via an IP-network connection between the PC and a telecommunication switching system. *Id.*, 1:14-18.

72. In Becker, when a call is received via the network for the IP softphone, the software alerts the user via an internal or external speaker attached to the PC. *Id.*, 1:25-28. Becker explains that one common problem with prior systems is that a user may be unable to answer an incoming VoIP call when a password protected screen blanker or screen saver is activated on the PC. *Id.*, 1:48-50.

73. In prior systems, when the user hears the IP softphone ringing, the user must first unlock the screen blanker to access the PC. *Id.*, 1:61-66. The user must then select the software application and activate the soft button labeled “Answer” to answer the incoming call. *Id.* Only after completing these steps can the user use their handset to talk. *Id.*, 1:66-2:1. Becker explains that due to this

time-consuming process, frequently, the call is transferred to voicemail or the caller terminates the call before the user has completed this process. *Id.*, 2:1-4.

74. To address this problem, Becker discloses “[t]he invention is directed to a personal computer able to modify automatically a screen saver in response to the occurrence of one or more selected events.” *Id.*, 2:34-36. Becker’s process first involves receiving a notification of the occurrence of an event (e.g., a phone call) from an application that is running on PC while the display of the PC is in a screen saver mode. *Id.*, 2:42-44. Becker’s process further involves determining if the event is a member of a defined set of events and then displaying information associated with the event when the event is a member of the defined set of events. *Id.*, 2:45-49.

75. In Becker, when a communication device receives a communication (e.g., an incoming call) while the screen saver is active, the graphical user interface associated with the communication application is displayed to permit the user to answer the communication without first disabling or terminating the screen saver. *Id.*, 2:64-3:1. Becker is configured such that when an incoming communication is received by a softphone or communications application, an event notification agent is notified, which in turn notifies the screen saver application. *Id.*, 4:25-28; FIG. 3. The screen saver application then causes the standard graphical user interface of the softphone application to be displayed by a display device even though the

display device is still in the screen saver mode. *Id.*, 4:33-37; FIG. 3. “The event notification agent 304 provides the interface between the softphone and screen saver applications 300 and 308.” *Id.*, 4:65-67.

76. Becker is further configured such that when the communication is answered by the user while in the screen saver mode, communications are exchanged between the connected parties via the enabled softphone application. *Id.*, 6:32-34. If the communication is not answered within a predetermined amount of time, the call is transferred to voicemail or terminated and the softphone graphical user interface display is removed from the display device. *Id.*, 4:29-32; 6:41-43.

VIII. CLAIM CONSTRUCTION

77. I understand from Counsel that the claims are to be construed in light of the specification as would be read by a POSITA at the time the application was filed. I understand that the claim terms are given their ordinary and customary meaning as would be understood by a POSITA in the context of the entire disclosure. For the purposes of this report and the arguments made herein regarding institution, I have relied on the ordinary and customary meanings of the claim terms. If instituted, I reserve the right to address constructions of the claim terms, if needed.

IX. ARGUMENT

78. Claim 1 of the '741 Patent recites: “enabling a user to select the object while the device is in the locked mode, in order to determine a state, function or utility of the application *without fully activating the application* while the device remains in the locked mode.” In this way, the '741 Patent provides user-access to specific functionality of an application that is not fully activated, without unlocking the device.

79. The determination of a state, function or utility of the application *without fully activating the application* refers to a constraint on the type of information that can be viewed in the locked mode because the claims require the application to not be fully activated. The '741 Patent provides multiple examples of such a constraint when determining a state, function or utility of the application. For example, “the icon 204 of FIG. 2A, representing a communication application, opens to user interface elements for contacts 222 and applications 224. This provides the user with the advantage of determining, prior to unlocking a device from an inactive state, an application or set of user interface elements for the device to open up to.” Ex. 1001, 6:48-53. Here, high level user interface elements, such as contacts 222 and applications 224, provide high level user interface elements that provide information regarding the functions of the fully activated communications application.

80. In addition, the '741 Patent provides a second example related to an icon 202, which represents a time or clock application. In this example, “when the device is locked, it may not be possible to view any functions of the application, such as for example, the current time.” *Id.*, 6:61-63. However, when the clock application is activated to determine a state, function or utility, the “clock application icon 302 is illuminated when that area is selected, and the current time is available and can be viewed, particularly against the dimmed and/or darkened background 304.” *Id.*, 7:3-6. Here again, the high-level user interface element, the clock, is presented to the user but the application itself is not fully activated. In fact, the '741 Patent teaches “the quick activation of high level user interface elements for passive information gathering, while the device remains in locked mode” *Id.*, 7:11-13.

81. In contrast to the claims and the '741 Patent specification, it is my opinion that neither Hassan nor Becker discloses or contemplates partially or temporarily activated applications. Instead, both Hassan and Becker disclose standard applications (e.g., Windows applications) that are stored in memory and fully activated upon execution. In each reference, the applications, even in a locked state / screen saver mode, include the same complete GUI used in an unlocked state / non-screen saver mode. As such, neither reference contemplates the quick

activation of high level user interface elements or operating an application in less than a fully activated state.

82. Instead, both references rely on a separate application or functions that operates to restrict a user request (e.g., a selection of an icon) based on permissions. But neither the claims nor the specification of the '741 Patent contemplate that restrictions imposed by another application (e.g., a task changer / screen saver application) result in an application that is not fully activated. Instead, similar to user based Windows restrictions imposed at the user/administrator level, the restrictions of the cited reference merely limit what task(s) that a user is able to perform without affecting the fully activated nature of the application.

83. As such, restrictions that result in certain functions of a fully activated application being unauthorized or blocked does not change the underlying fact that the applications in both cited references are fully activated. For these reasons and the reasons below, Petitioner fails to show that the cited references teach or suggest the claims of the '741 Patent.

A. Grounds 1 and 2 – Hassan does not render obvious claims 1, 5-7, 10-11, 15, 17, and 19 of the '741 Patent and Hassan and Nirhamo do not render obvious claims 1-20 of the '741 Patent

84. I understand that Petitioner has grouped Grounds 1 and 2 together for the purposes of its Petition. Petition, 8-9. It is further my understanding that the Petition relies on Hassan alone in its assertion that claims 1, 11, and 17 are

obvious. *Id.* As such, it is my understanding that Nirhamo is only applied to the dependent claims. While I do not address Nirhamo or the dependent claims herein, if this case is instituted, I reserve the right to address Nirhamo and/or the dependent claims, if needed.

1. Launching a Hassan application in a “locked state” does not teach or suggest the claims

85. Petitioner states that “[] Hassan teaches enabling a user to select the icon for an application to perform operations such as launching an application when the user requests a task change.” Petition, 23. However, in my opinion, opening an application to a fully activated state, even if in a locked mode, does not teach or suggest the claims.

86. During prosecution, the then pending claims were rejected based upon the Bodepudi reference. Ex. 1002 at 713-714. In particular, the Examiner relied on paragraph 51 of Bodepudi, which disclosed a fully activated application that launches in a wrapper application. Paragraph 51 of Bodepudi states:

Referring now to FIG. 6, an illustrative diagram shows a user interface during selective lock mode. As illustrated, a user interface 600 during selective lock mode, as compared with user interface 500 during normal mode, only includes a selective lock folder window 620 representing selective lock folder 408 with a selectable link 622 associated with application A and a selectable link 624 associated with application D. Responsive to a non-authorized user selection of application A, selective lock application 208 calls a wrapper function

that *opens* application A, as illustrated by application A window 602 within user interface 600 through which a user has **full access** to the functions of application A.

Ex. 2003, ¶0051 (emphasis added).

87. In addition, Bodepudi discloses: “[a]s will be further described, selective lock application 208 detects user selection of one of the designated selection of applications and calls a wrapper that opens the selected application to allow the user to access the full functionality of the application, but limits accesses to applications or files outside the wrapped application.” *Id.*, ¶0034. Bodepudi states that even though its application is bounded in memory, it is still fully activated—“[i]n particular, according to an advantage, by calling a wrapper function to open a selected application, the application is bounded in memory so that the non-authorized user is allowed full access to the functions of application, but blocked from accesses to other applications and files outside the wrapped application.” *Id.*, ¶0041. Finally, Bodepudi explains that a user name and password will allow the system to enter/exit the locked mode—“[i]f the authorized user enters the required password, then application lock controller 402 closes wrapped applications and may redirect the operating system to restore the memory to the system status before the selective lock mode initiated.” *Id.*, ¶0042. In each of these examples, it would have been clear to the

POSITA that the Bodepudi application was fully activated even though the overall functionality of the user was constrained by the wrapper application.

88. In response, the Applicants amended the claims to recite “without fully activating the application.” Applicants noted that support for this claim amendment is provided in at least paragraph [0036] of the published application, which is column 6: line 56 – column 7, line 13 of the issued patent and is provided below for reference.

Referring to FIGS. 2A and 3, in one embodiment, the user can select or contact the dimmed icons 202 and 204 to determine a state, utility or function of the underlying application, without the need to unlock the device or open the application. For example, in FIG. 2A, dimmed icon 202 represents a time or clock application. In the dimmed mode, when the device is locked, it may not be possible to view any functions of the application, such as for example, the current time. However, the features of the disclosed embodiments allow the icon 202 to be contacted to temporarily illuminate or somehow highlight the icon, as illustrated by icon 302 in FIG. 3. This state, which might be described as a passive active state, allow for temporary activation of an application, such as a clock. This is also referred to herein as the “semi-active” state. As shown in FIG. 3, the clock application icon 302 is illuminated when that area is selected, and the current time is available and can be viewed, particularly against the dimmed and/or darkened background 304. In alternate embodiments, any suitable way to temporarily highlight the selected icon can be implemented. The

clock function is illuminated temporarily, but does not become fully active. The icon 302 can remain illuminated for a pre-determined period of time, such as for example, 5 seconds. This allows the quick activation of high level user interface elements for passive information gathering, while the device remains in the idle and locked mode.

Ex. 1001, 6:56 – 7:13.

89. As a result, Applicants overcame the then pending rejection. That is, by amending the claims to recite that the claimed application is not fully activated, the Applicants distinguished references where a fully activated application was bounded by a wrapper application that intercepts user actions and selectively blocks access.

90. Like Bodepudi, which discloses a wrapper function that opens a fully activated application while still constraining the user, Hassan discloses the use of a type of wrapper function, a task changer, to restrict access to certain fully activated applications. Because Hassan's task changer is operating in Hassan's "locked state" and restricts access to what applications can be launched is the same as or equivalent to Bodepudi's wrappers restricting what applications can be launched, launching an application in Hassan when the user requests a task change does not satisfy the claim limitation "without fully activating the application."

2. Hassan's use of a task changer to restrict certain tasks or operations within a fully activated application does not teach or suggest the claims.

91. Petitioner states “Hassan teaches that the application (e.g., e-mail) is not fully activated because users may only access a limited set of ‘permitted’ tasks or operations within the application, rather than the full set of operations.” Petition, 25. And because the application cannot perform its full set of operations and/or functions, Petitioner alleges that the application itself is not fully activated. *Id.* Dr. Bederson recites this same argument. Ex. 1003, ¶0058. But, in my opinion, Petitioner and Dr. Bederson misread Hassan. Hassan never contemplates partially or temporarily activated applications, only fully activated applications whose execution is controlled by a task changer. Hassan's applications are fully activated once launched and access to their functions in a locked state are controlled by the use of task change APIs and the task changers (task switcher, window launcher, etc.). The use of a task changer to intercept certain tasks or operations of a fully activated application for the purpose of restricting access to certain tasks or operations within that application does not teach or suggest the claims.

92. Hassan teaches a task changer and task change API which controls user access to fully loaded and activated applications and their operations, while in a locked state. Specifically, Hassan states:

The present application provides a device and a method for placing a device in a locked state having an associated set of permitted tasks so

as to permit the device owner to share the device with others but maintain security over aspects of the device. The user of a locked device may only perform tasks that are permitted without unlocking the device. A task change request is evaluated to determine whether the requested task is permitted and, if not, then an authorization window is opened to prompt the user to input authorization data. Upon verification of the authorization data, the device may be unlocked and the requested change executed. The permitted tasks may designate specific applications, specific operations or functions within applications or at the operating system level, one or more currently open windows, or other levels of granularity. ***Task change API calls by a task changer, such as a task switcher or window launcher, may be evaluated by a process control level of the operating system to determine whether the requested task change is permitted.*** If the task is not permitted, then the API call may be failed and the task changer may re-call the task change API supplemented with an argument containing the authorization data input by the user.

Ex. 1004, ¶0014 (emphasis added).

93. Hassan's task change API that is used by task change applications uses a process control level of the operating system to determine which aspects of the fully activated application's function(s) are permitted to be used or not ("locked"). *Id.*, ¶0033. Hassan states that "references to a task changer or task change module are intended to encompass the object or software element at an application level or operating system level for initiating instructions to execute an

operation within a specific application”. *Id.*, ¶0036. Hassan states: “[t]he term ‘task change’ or ‘task changer’ as used herein is meant to encompass both a change in the active window as between two or more open windows, and the launch or opening of a new window. In some embodiments, ***it is also meant to encompass selection of a specific operation within an application, where the operation implements an application-specific feature.***” *Id.*, ¶0037 (emphasis added).

94. Hassan’s task changer intercepts user requests. Specifically, Hassan states that “[t]he task changer is the software object or element that responds to that user input to execute or implement the requested operation or task. The task changer typically calls an application programming interface (API) instruction to implement or execute the requested function.” *Id.* Hassan provides examples for how the system operates by noting that the user interacts with an application as they normally would confirming that the task changer is a separate application or wrapper that modifies how the user interacts with the fully activated Hassan application. Hassan states: “[f]or example, in an e-mail application, a user may request a certain function like forwarding a message, replying to a message, opening a message, through an appropriate user input (like clicking on a designated icon).” *Id.* That is, regardless of a locked state or an unlocked state, the user is able to, for example, activate an icon to, for example, reply to a message, and it is the

task changer that processes task change requests and may interrupt the task request unbeknownst to the fully activated application to request user authentication.

95. But the presence of the task changer does not alter the fact that the applications of Hassan are fully activated. Indeed, the operation of Hassan's device and the applications stored thereon is the same in the locked state as it would be in the unlocked/open and accessible state. The sole difference in the locked state is that the background or wrapper application (e.g., task changer) is intercepting each request and confirming whether or not a particular task or operation is authorized before it allows a task to be executed. This procedure is shown in Figure 6 of Hassan.

96. As demonstrated in Figure 6, Hassan's applications are fully activated (see e.g., step 148) and it is the task changer that intercepts any task change request from a user. That is, from the applications perspective, the application is fully activated with all options fully enabled such that they can be executed upon request. That is, the fully activated application is operating as normal and is unaware of the task changer intercepting user requests.

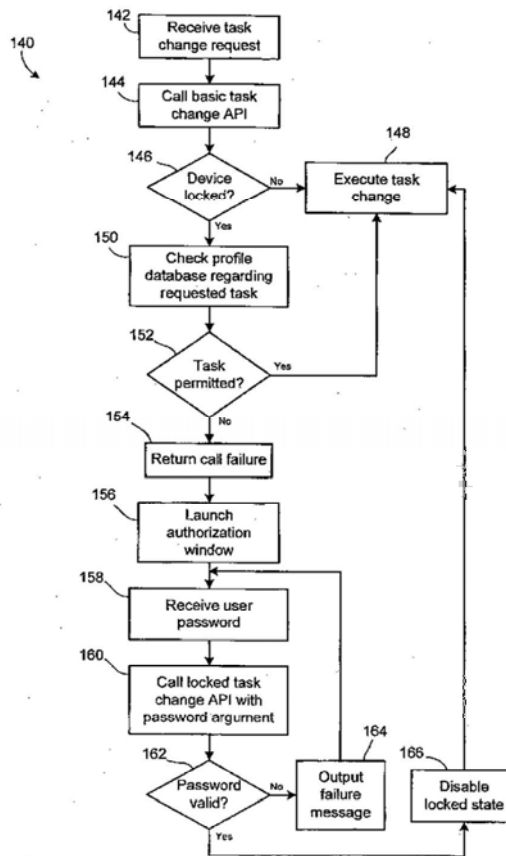


FIG. 6

Id., FIG. 6.

97. With regard to Figure 6, “[t]he task change request is received in step 142.” *Id.*, ¶0064. For there to be a task change request, there must be a request from a user to select an application, an operation within an application, or a task change. The POSITA would have understood that for the user to make such a request the menu or icons of available functions, such as functions of an applications, must be displayed. Displaying all of the available functions or operations would suggest to a POSITA that an application is fully activated.

98. Hassan confirms as it provides numerous examples of its applications being fully activated in that each available operations (e.g., icons) are activated and shown to a user. “For example, within a word processing program a user request to open a new document or an existing document causes the launch of a new window. Similarly, within an e-mail messaging program, clicking on a message in an “inbox” view launches a new window for displaying the selected message. Within the displayed message, selection of a “reply” or “forward” button or menu item launches yet another window for composing the reply or forward message.” *Id.*, ¶0035. Again, as I noted above, the application publishes the “reply” or “forward” button and it is the task changer that intercepts any user input related to these buttons so as to create the “task change request” that is processed at step 142. The fact that the application publishes the “reply” or “forward” button even though they may be unauthorized by the task changer indicates to the POSITA that the Hassan application is fully activated despite the presence of the task changer.

99. Moreover, Hassan relies on different user profiles for different users as it discloses:

For example, one profile may lock the user into a voice call application program. Another profile may only permit the user to activate games programs. Yet another profile may lock the device to the then-active window open on the device. This latter type of lock may be useful when reading an e-mail message opened in the active

window, thereby allowing the user to scroll through the message, but preventing access to other e-mail messages on the device and/or preventing forwarding or replying to the e-mail message being viewed.

Id., ¶0040. Because different users can have different user profiles giving them different sets of permitted functions, a POSITA would have understood that the application must be fully activated to be able to support such different profiles.

100. In response, at step 146 of Figure 6, the “process control layer 64 assesses whether the device is locked in step 146.” *Id.*, ¶0064. If not, the change request is passed to the operating system for implementation, as is shown in step 148. *Id.* Here, the POSITA would understand that the Hassan device is unlocked and passing a “change request” or task request that originated from an application to the operating system would result in the device operating normally. That is, in response to a request for a new task or application, the operating system would process the request and execute the task or open the application in the normal course. Such an operation evidences that the application is fully activated and is operating as normal even though the task changer is performing the steps of Figure 6.¹ Indeed, there is no indication there is any change to the Hassan application to

¹ If the application was not fully activated, there would be no need to, for example, check to see if the device was locked (Step 148) or check the profile data base

be able to perform the task only that the intercepted task is finally passed along to the application for execution.

101. If the device is locked at step 146, the process control layer determines whether the requested task change is a permitted task change. *Id.*, ¶0065. “This evaluation as to whether the task is permitted, as shown in step 152, may be based upon the identity of the application program requested, the type of task or operation being performed, or other factors.” *Id.* Regardless, if the task change is permitted, “the method 140 continues to step 148, whereupon the task change API call is passed through to the operating system 52 for implementation of the requested change.” *Id.* Again, because a task change is implemented just the same as it was implemented in an unlocked state, the application is fully activated and is operating as normal even though the task changer is intercepting any requests and performing the steps of Figure 6.

102. If the task is impermissible, “in step 156, the locking component 60 launches an authorization window to prompt the user to provide input authorization data, such as a password. Through the authorization window, the locking

regarding the requested task (Step 150) because the application itself would include just those high level user interface elements that were authorized in a locked state. *See* Ex. 1004, ¶0035.

component 60 receives user input in step 158.” *Id.*, ¶0066. Here, the presence of an authorization window does not change my analysis above that the application itself is fully activated and is operating as normal in the unlocked state because the user was able to request an unauthorized task or operation in the application.

103. Importantly, nothing has changed with the application and its status of being fully activated between steps 146 and 156 of Figure 6. That is, the fully activated application at step 146 remains fully activated at step 156. In fact, from the applications perspective nothing has changed at all (e.g., the user interface and application functionality has not changed). The only distinction or change that has occurred is that a separate application, a task changer, has launched an authorization window as a result of the intercepted request. This authorization window has not changed the activated status of the application or altered the application, but instead has intercepted a task request before it is implemented by the operating system even though the fully activated application is itself able to execute the task. As such, the POSITA would have understood that the Hassan application is fully activated.

104. Hassan confirms my interpretation as it states “[i]f the password is validated, then in step 164 the lock state may be disabled and the method 140 proceeds to step 148 to implement the requested task change.” *Id.*, ¶0067. That is, upon authorization, the “change request” or task request is forwarded to the

operating system for implementation. Again, there is no change to the application contemplated upon authorization only that the task changer will process a task request.

105. Regardless of success or failure, Figure 6 demonstrates that an application is fully activated in Hassan such that from the applications perspective, the user is able to execute a change request or task request, such as by selecting an icon within the application, rendering the application fully activated. Outside of the application, a task changer intercepts these requests and determines whether to pass the requests to the operating system. Regardless of the decision made by the task changer, the application itself remains fully activated.

106. Hassan never contemplates partially or temporarily activated applications, only fully activated applications whose execution is controlled by a task changer. As a result, Hassan's fully activated applications fails to teach or suggest "without fully activating the application" as is recited in claims 1, 11, and 17.

3. Hassan's applications are not temporarily activated.

107. Petitioner asserts that "[t]herefore, Hassan teaches that the application (e.g., email) is not fully activated while the device remains in locked mode; instead, it is *temporarily* activated with limited functionality, which serves the underlying purpose of Hassan." Petition, 26. As I noted above, Hassan's

applications are fully activated and Hassan does not teach or suggest that applications in Hassan are temporarily fully activated.

108. Petitioner appears to rely on the disclosure of the '741 Patent, which recites that an application is temporarily activated when the device is in the locked mode. For example, “[a]n application that does not become active, or fully active, when selected, such as a clock that can be *temporarily* illuminated, may be displayed with a 60% illumination level.” Ex. 1001, 3:65-4:1. Further, “[] the features of the disclosed embodiments allow the icon 202 to be contacted to *temporarily* illuminate or somehow highlight the icon, as illustrated by icon 302 in FIG. 3. This state, which might be described as a passive active state, allow for temporary activation of an application, such as a clock.” *Id.*, 6:64-7:1. Finally, “[t]he clock function is illuminated *temporarily*, but does not become fully active.” *Id.*, 7:8-9.

109. In contrast to the '741 Patent, Hassan discloses, and Petitioner confirms, that it is not the Hassan applications that are *temporarily* activated but that it is apparently the “locked mode [that] is only temporary.” Petition, 26. As I described above, each of Hassan’s applications are fully activated and there is nothing in Hassan that suggests that an application or operation of an application is temporary. As such, any application will remain fully activated in Hassan until it is closed, which would not be considered *temporary* to a POSITA.

110. Because the only portion of Hassan which Petitioner alleges is temporary is the locked mode, Hassan does not teach or suggest the temporary activation of applications.

B. Grounds 3 and 4 – Becker does not render obvious claims 1, 5-7, 10-11, 15, 17, and 19 of the '741 Patent and Becker and Nirhamo do not render obvious claims 1-20 of the '741 Patent.

111. I understand that Petitioner has grouped Grounds 3 and 4 together for the purposes of its Petition. Petition, 43-44. It is further my understanding that the Petition relies on Becker alone in its assertion that claims 1, 11, and 17 are obvious. *Id.* As such, it is my understanding that Nirhamo is only applied to the dependent claims. While I do not address Nirhamo or the dependent claims herein, if this case is instituted, I reserve the right to address Nirhamo and/or the dependent claims, if needed.

1. Becker fails to teach or suggest “without fully activating the application.”

112. Claim 1 recites: “enabling a user to select the object while the device is in the locked mode, in order to determine a state, function or utility of the application *without fully activating the application* while the device remains in the locked mode.”

113. I understand that Petitioner states: “Becker teaches that the communication application is not fully activated because the user may access only a restricted set of the features that only ‘allow[s] for limited calling with specified

call permissions (or limited calling to specified destinations) while the PC is password protected.” Petition, 57. And because Becker teaches “limiting user access ‘to a restricted set of the features of the communication application,’” Petitioner alleges that the application it is not fully activated. *Id.*, 57-58. Dr. Bederson recites this same argument. Ex. 1003, ¶130. But Petitioner and Dr. Bederson misread Becker. While Becker may disclose that the softphone or communications application may be restricted, a POSITA would have understood that these permissions are implemented and controlled by the screen saver application in conjunction with the event notification agent.

2. Becker’s softphone or communications application is fully activated.

114. It is my opinion that a POSITA would have understood that the Becker softphone or communications application is fully activated while operating in the Becker screen saver mode. Becker never contemplates partially or temporarily activated applications, only fully activated applications whose execution is controlled by a screen saver application.

115. Becker discloses a “screen saver mode,” whereby a personal computer may “modify automatically a screen saver in response to the occurrence of one or more selected events.” Ex. 1006, 2:34-36. Specifically, “[w]hen the communication device is receiving the communication, a graphical user interface associated with the communication application is displayed to permit the user to

answer the communication without first disabling or terminating the screen saver mode.” *Id.*, 2:61-3:1. In operation, Becker provides an “event notification agent” that notifies a screen saver application when an incoming communication is received so that the screen saver application can cause the GUI of the softphone or communications application to be presented. *Id.*, 4:25-28, 4:33-38, 5:1-16.

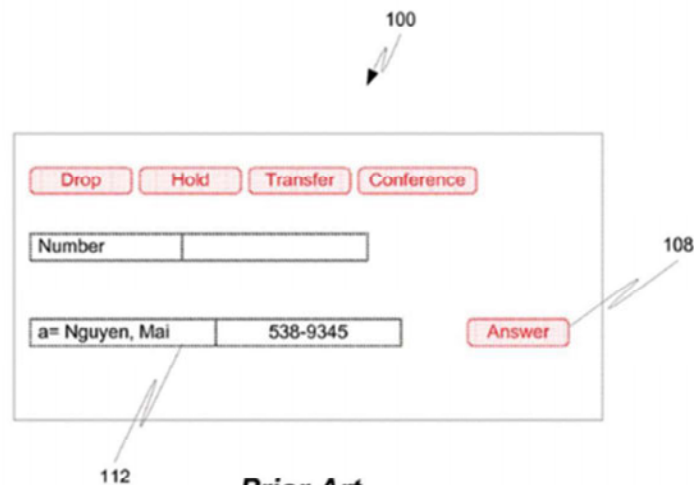
116. Becker discloses the operation of the GUI of the softphone or communications application in a non-screen saver mode with respect to Figures 1A, 1C, and 1D:

With reference to the Avaya IP Softphone™ display 100 of FIG. 1A, the user may answer the call by selecting the softphone software application (such as by clicking on the corresponding icon 104 of FIG. 1B) and selecting a button 108 labeled “answer”. Before and after the call is answered, the caller's name and number, if available, is depicted in display field 112. The user then conducts a conversation with the caller via either the handset or the headset. When the call is answered, the display changes automatically to the display 120 of FIG. 1D, which includes buttons for drop 124, hold 128, transfer 132, conference 136, and hangup 140. The call is terminated by the called user by selecting the hangup button 140. With reference to the Avaya IP Softphone™ display 150 of FIG. 1C, to place a call the user once again selects the software and selects a soft button 154 labeled “Number” and proceeds to dial the destination number using a keypad or a pre-stored list of telephone numbers. The number appears in the

field 158. The keypad can be either a soft keypad or the PC's keyboard.

Ex. 1006, 1:29-47. When the device is in screen saver mode, Becker discloses that the user can receive and place outgoing calls using the same fully activated softphone or communications application GUI as is described at Becker column 1, lines 29-47. *Id.*, 2:61-3:1; 6:52-54.

117. I understand that the Petition includes annotated Figure 1A as an example of the GUI that is displayed in screen saver mode. This GUI includes all of the user interface elements as would be included if the softphone or communications application was operating outside of the Becker screen saver mode. Because a POSITA would have understood that the softphone or communications application of Fig 1A was fully activated in the non-screen saver mode, the POSITA would have understood that the same application with the same user interface elements operating in the screen saver mode would also be fully activated.



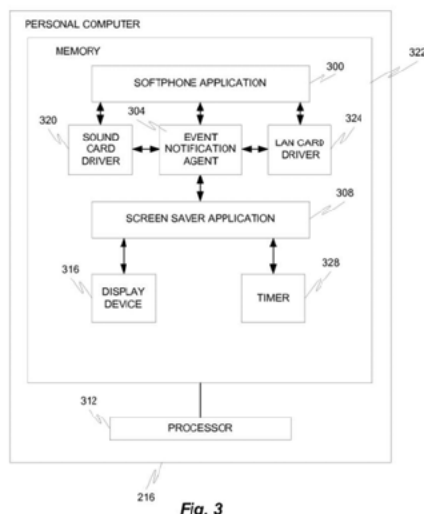
**Prior Art
Fig. 1A**

Petition, 52.

118. In addition to relying on the same user interface of the softphone or communications application in a screen saver mode, Becker provides additional disclosure confirming that its softphone or communications application is fully activated in the screen saver mode.

119. Similar to how a POSITA would have understood an application to be run in a typical operating system environment, such as a Microsoft Windows environment, a POSITA would have understood that the Becker softphone application is stored in memory and fully activated when executed. For example, as illustrated in Fig. 3 and the associated text: “FIG. 3 depicts, in block diagram form, the structure of the software instructions stored in the memory 322 (or a computer readable medium) of the PC 216 and executed by the processor 312 to

provide functions for the handset or for the headset and for the display device 316.” Ex. 1006, 4:51-55. As is shown below, softphone application 300 is one example of a software instruction that is stored in memory.



Ex. 1006, Fig. 3.

120. In addition, a POSITA would have understood that the softphone or communications application is fully activated in Becker because it is configured to continuously monitor for incoming communications and to handle them when they occur. For example, “[w]hen an incoming communication is received by softphone application intended for the user/subscriber of the PC 216, an event notification agent 304 (FIG. 3) is notified, which in turn notifies a screen saver application 308 (FIG. 3).” Ex. 1006, 4:25-28. In addition, “[w]hen the control information is received from PBX 204 indicating that there is an incoming communication, the information is relayed from LAN card driver 324 to the softphone application 300.

The softphone application 300 then initiates sound card driver 320 to produce a ringing sound on the internal or external speaker of PC 216. In addition, the softphone application 300, concurrently or immediately before or after, notifies the event notification agent 304 of the incoming communication.” *Id.*, 5:1-9.

121. In an instance in which an incoming communication is detected, the fully activated softphone or communications applications configures its complete GUI to be displayed in the Becker screen saver mode. For example, “[i]n response, the screen saver application 308 causes the GUI of the softphone application, such as the GUI of FIGS. 1A and 1B, to be displayed by the display device 316 (FIG. 3), even though the display device is still in screen saving mode.” Ex. 1006, 4:33-37. A POSITA would have understood that the Becker softphone or communications application would be fully activated because the same complete GUI used in non-screen saver mode is displayed.

122. Finally, and as I described above, Applicants overcame a reference similar to Becker during prosecution—Bodepudi. Like Bodepudi, which discloses a wrapper function that opens a fully activated application while still constraining the user, Becker also discloses the use of a type of wrapper function, a screen saver application, to restrict access to certain applications. As a result, it is my opinion that Becker’s presentation of a softphone or communications application GUI in a

screen saver mode fails to teach or suggest the “without fully activating the application” limitation of the independent claims.

3. Becker’s use of a screen saver application to restrict the fully activated softphone or communications application does not teach or suggest the claims.

123. In each instance of the Becker softphone or communications application, the application is fully activated even though the display device is still in screen saver mode. Indeed, even when certain permissions restrict the softphone or communications application, a POSITA would have understood that these permissions or restrictions are implemented by the screen saver application in conjunction with the event notification agent.

124. I understand that the Petition states: “Becker further teaches enabling a user to select the object (e.g., button) in order to determine a state, function or utility of the application **without fully activating the application** while the device remains in the locked mode.” Petition, 57 (emphasis in original). I understand that the Petition relies on Becker’s teaching of limiting user access to “a restricted set of the features of the communications application” while the PC is in the screen saver mode. Petition, 57-58 (citing Ex. 1006, 6:48-60 (“The user may freely access all of a restricted set of the features of the communications application to accept incoming or place outgoing contacts or otherwise conduct communications; ... For example, the incoming GUI display, such as that of FIG. 1A, can be displayed to

permit only incoming communications to be answered but the user -58- restricted from making or initiating outgoing communications.”)). Finally, I understand that Petitioner relies on the buttons of Becker’s communication application to demonstrate that Becker enables a user to select an object. Petition, 56 (“Likewise, Becker enables users to select the drop/hold/transfer/conference buttons to determine whether or not the application supports the corresponding functionality during screen saver mode.”). The descriptions in the Petition do not change my position that the Becker application is fully activated.

125. While a POSITA would have understood from Becker that there could be a restriction of functionality in a screen saver mode, such a restriction does not change that the Becker smartphone or communications application is fully activated. As I described above, the Becker smartphone or communications application is loaded in memory and executed the same complete GUI used in non-screen saver mode.

126. Indeed, Becker itself states that it is the screen saver application that controls any restrictions. For example, “Softphone application 300 provides the overall control of the communication of calls with PBX 204. Screen saver application 308 provides the *overall control of the display device 316* as discussed below. The event notification agent 304 provides the interface between the softphone and screen saver applications 300 and 308.” Ex. 1006, 3:61-68. “If so,

the [screen saver] application, in step 422, displays the GUI for the communication application (e.g., softphone or instant messaging application) (without receiving the defined information from a user) but *continues to restrict the user's access to a defined set of applications.*" *Id.*, 6:21-26; *see also id.*, 4:37-40 ("As will be appreciated, when the device is in screen saving mode the ability of the user to access/enable other software applications, such as those stored in PC 216, is restricted"). Becker provides no other teachings with regard to how a restriction would be implemented.

127. Thus, the only apparent change in Becker is that upon selection of a button, the Becker screen saver application may restrict the button's corresponding functionality. Similar to what I discussed with regard to Hassan, restrictions that are implemented by a separate application (e.g., a task changer or a screen saver application) do not change the fact that the application of Becker is fully activated, is displaying its same GUI used in the non-screen saver mode, and allows for selection of any of its buttons. Thus, the softphone or communications application is fully activated.

128. This further supports my opinion that Becker fails to teach or suggest "enabling a user to select the object while the device is in the locked mode, in order to determine a state, function or utility of the application without fully activating the application while the device remains in the locked mode."

4. Petitioner's ground requires each of Becker's buttons to be present on Becker's user interface.

129. To the extent it is argued that Becker restricts its application by removing functionality (e.g., buttons) from its GUI in the screen saver mode, which would be inconsistent with how a POSITA would read Becker and how Petitioner has interpreted Becker, then Petitioner's alleged rejection fails for another reason.

130. In order to allegedly render obvious, "enabling a user to select the object while the device is in the locked mode," I understand that Petitioner is relying on the selection of a "button in the GUI display" of Becker. Petitioner, 54. Petitioner identifies that "Becker teaches that users select the GUI display, including, for example, 'Answer,' 'Drop,' 'Hold,' 'Transfer,' 'Conference' buttons, to perform permitted operations when the PC is in screen saver mode." *Id.* Based on the alleged presence of these buttons, Petitioner asserts "Becker enables a user to select its buttons, to determine a function of the application (e.g., the ability to answer, drop, hold, transfer, or conference incoming calls) during screen saver mode." Petition, 56. Other than the buttons within the GUI of Becker, Petitioner does not identify any other "objects" that would teach or suggest "enabling a user to select the object while the device is in the locked mode."

131. Petitioner's alleged rejection relies on each of Becker's buttons being present so that they are selectable by the user to determine whether or not the

application supports the corresponding functionality during screen saver mode. If Becker's restriction is argued to mean that a restricted button (e.g., dial or answer) is removed from the GUI, then Petitioner's reliance on selection of such a button to determine its functionality is no longer possible.

132. As such, if Becker was interpreted to remove a restricted button from its GUI and to keep unrestricted buttons on the GUI, Petitioner would no longer be able to show "enabling a user to select the object while the device is in the locked mode." It is my opinion that Petitioner's proposed rejection also fails for this additional reason.

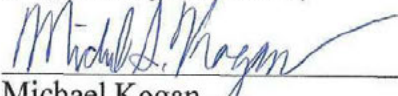
X. CONCLUSION

133. In signing this declaration, I recognize that this declaration will be filed as evidence in a contested case before the Patent Trial and Appeal Board of the United States Patent and Trademark Office. I also recognize that I may be subject to cross-examination in the case.

I hereby 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 further that these statements were made with the 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.

Executed this 24th day of October, 2024.

Respectfully submitted,


Michael Kogan