UNITED STATES DISTRICT COURT FOR THE
NORTHERN DISTRICT OF ILLINOIS
EASTERN DIVISION

APPLE INC. and NeXT SOFTWARE INC. (f/k/a NeXT COMPUTER, INC.),	)	
Plaintiffs,	)	No. 1:11-cv-08540
v.	)	
	)	Judge Richard A. Posner.
MOTOROLA, INC. and MOTOROLA MOBILITY, INC.,	)	
Defendants.	)	

ORDER

This order resolves motions, argued January 23, for summary judgment regarding four patents, plus claim construction of a fifth patent, also argued at that hearing.

As a result of these orders, the following patents involve, unless I am mistaken (please point out if I am), issues for trial: Apple '002, '263, '337, '354, '647, '867, '949; Motorola '516, '559, '898.

Motorola's summary judgment motion regarding Apple '486 and '852 is granted, and Apple's motion for summary judgment is dismissed as moot

Apple claims that certain Motorola devices running the Android operating system infringe U.S. Patent Number RE 39,486 E ("Extensible, Replaceable Network Component System") and related U.S. Patent Number 5,929,852 ("Encapsulated Network Entity Reference of a Network Component System"). I will refer to these patents collectively as the "Network Component Patents." Motorola has moved for summary judgment that its devices do not infringe the patents.

Claim 1 of the '486 patent describes "replaceable…component[s]" in a layered computing system. (The '852 patent incorporates the replaceable component system of '486 by reference at 13:22–44, and my interpretation of the terms applies equally to both patents' claims.) The use of replaceable components promotes program flexibility and facilitates customization because replaceability enables the user to alter functionality within a given program. The prior art was "appli-

cation-based," which meant that users had to accept an application's functionality as is or not use it at all.

Apple argues that the Java objects in the Android Runtime (the part of the Android system that facilitates communication between the processor and the applications) are the infringing replaceable components. Motorola asks that I construe "replaceable components" to mean components replaceable by Android end-users, and argues that the allegedly infringing Java objects are irreplaceable under that claim construction.

Apple's opening argument that apparatus claims, such as claim 1, may never be construed to require functional limitations doesn't mesh with the claim language of the patent. A patent's claims may describe a physical apparatus without reference to its components' capabilities, and when described in "purely structural terms" cannot be altered by functional limitations derived from the patent specification. *Schwing GmbH v. Putzmeister Aktiengesellschaft*, 305 F.3d 1318, 1324 (Fed. Cir. 2002); *Hewlett-Packard Co. v. Bausch & Lomb, Inc.*, 909 F.2d 1464, 1468 (Fed. Cir. 1990). But see *ACCO Brands, Inc. v. Micro Security Devices, Inc.*, 346 F.3d 1075, 1076–78 (Fed. Cir. 2003). But apparatus claims can incorporate functional limitations in the description of the claimed apparatus elements without invalidating the patent. *Typhoon Touch Technologies, Inc. v. Dell, Inc.*, 659 F.3d 1376, 1380 (Fed. Cir. 2011); *Microprocessor Enhancement Corp. v. Texas Instruments Inc.*, 520 F.3d 1367, 1374–75 (Fed. Cir. 2005). The replaceable components in the layered computing arrangement described in '486 are not a purely structural description, but a structural description defined in relation to a specific function of replaceability. (A car's brakes are replaceable, as are lightbulbs, but they require of the replacer different degrees of skill and effort.) The '486 apparatus is innovative because of the ability to replace components, and that claim term must be understood to interpret the computing environment patented by '486. I therefore reject Apple's argument that clarifying who must be able to replace the components would improperly impose a functional limitation on a purely structural apparatus claim.

Interpretation of patent claim language is based primarily on intrinsic evidence, including the patent specification. *Bell Atlantic Network Services, Inc. v. Covad Communications Group, Inc.*, 262 F.3d 1258, 1267 (Fed. Cir. 2001). The '486 specification emphasizes the flexibility of the replaceable components; "if a user does not like the way a particular component operates, that component can be replaced with a different component provided by another developer…. Clearly, the replaceability feature of the novel network component system provides a flexible alternative to the user" in contrast with the prior art of using "monolithic applications" that cannot be altered by the user. This is convincing evidence that

the system described in claim 1 intended its components to be replaceable by "users."

But is an Android developer a "user"? Apple argues that he is, and cites the Android developers' guide as an instruction manual for replacing components of the Android system. And even if "user" is confined to end-user (the consumer), the Android developers' guide contemplates that some end-users skilled in Android programming will customize their phones' functionality. "Android Compatibility," http://source.android.com/compatibility/index.html (visited Jan. 24, 2012) ("a mobile phone is a highly personal, always-on, always-present gateway to the Internet. We haven't met a user yet who didn't want to customize it by extending its functionality. That's why Android was designed as a robust platform for running after-market applications").

But the developers' guide is geared towards modification of Android applications, not the Android Runtime environment. It states that "Android offers developers the ability to build extremely rich and innovative applications," and that the Android Runtime libraries provides the "functionality available in the core libraries of the Java programming language" to facilitate application development. "What is Android?" http://developer.android.com/guide/basics/what-is-android.html (visited Jan. 24, 2012). The parties agree that the Android Runtime environment, not the higher-level Android application environment, is the "software component architecture layer" which must contain replaceable components to infringe the Network Component Patents. Motorola's expert has said that "the Android Runtime, including its Java classes, is part of the core infrastructure of Android, and it cannot be modified or replaced by a user, other than through a full system update." Apple's response is that such a full system update is still "replacement" if the updated code replaces any Java objects of the Android Runtime libraries.

The examples of component replacement within the Android Runtime environment described by Apple's expert are remote from the interchangeable component system described in the '486 patent specification. The specification states that an "object[] of the ['486] invention is to simplify a user's experience" and to "provide a platform that allows third-party developers to extend a layered network component system by building new components that seamlessly interact with the system components." Components that are theoretically replaceable only by manually updating code during a full Android system update neither simplify the user's experience nor promote the formation of a third-party developer community creating interchangeable components to modify Android Runtime functionality.

Apple's expert asserts that he has replaced Java objects in his Motorola Droid X's Android Runtime environment, but has not explained why even sophisti-

cated users would want to make such modifications, or whether the components he substituted had to be written from scratch. He fails to rebut Motorola's evidence that the components of the Android Runtime environment are not "replaceable" as the term would be understood by a knowledgeable person in light of the '486 patent specification. The language of the patent specification, which touts the invention's ability to simplify users' experience and spur third-party development of interchangeable components, describes an invention similar to the application layer of the Android system, but not to the Android Runtime layer. Since the existence of meaningfully replaceable components within Android Runtime is a necessary element of Apple's infringement claim, Motorola is entitled to summary judgment of noninfringement. *Vivid Technologies, Inc. v. American Science & Engineering, Inc.*, 200 F.3d 795, 807 (Fed. Cir. 1999).

Motorola's motion for summary judgment regarding Apple '867 is denied

Motorola has moved for summary judgment of noninfringement of Apple's U.S. Patent Number 5,519,867 ("Object-Oriented Multitasking System"). The patent covers an apparatus for allowing incompatible applications and operating systems to communicate. Some programming languages are "object-oriented"; others are "procedural." Object-oriented applications are incompatible with procedural operating systems (and vice versa), much as an English speaker would be unable to follow instructions or answer questions in Japanese. The '867 patent bridges this linguistic gap by creating a "wrapper" for procedural operating systems that makes object-oriented applications compatible with procedural systems by providing an apparatus for enabling object-oriented applications to access procedural system services.

One aspect of this compatibility relates to "threads." A thread is a series of code to accomplish a discrete task. To run on a procedural operating system, object-oriented applications need to be able to get information from the operating system about threads; they may need to know about the priority (relative importance) of a thread, or its state (whether it's currently running, ready to execute, or blocked).

Limitation (e) of claim 1 is "means…for enabling said object-oriented application to access said services to spawn, control, and obtain information relating to a thread of execution." The parties agree that this language describes the function of enabling object-oriented application to access the operating system's services ("said services") to spawn (create), control, and obtain information relating to a thread. Motorola argues that its products don't infringe the '867 patent because they do not enable object-oriented applications to access the operating system to obtain thread information.

The foundational component of the operating system in the accused products is called the Linux kernel, which stores a cached copy of its thread information in the "Dalvik Virtual Machine," which is continuously updated. When Android applications need information from the operating system, they don't query the Linux kernel directly, but instead access Linux kernel services via a "virtual machine," a replication of a computer (computer in the broadest sense, meaning computing device) that functions just like the real machine but is achieved through software rather built from hardware. Motorola argues that there can't be infringement because Android applications never access the Linux kernel to obtain thread information; rather they obtain already-stored thread information from the virtual machine.

The dispute is over whether the two-step access system involving a virtual machine is an application that accesses the operating system's services. Apple's expert, Dr. Aldrich, says that the process of caching data and reading the cache instead of bothering the operating system directly with inquiries is a well-accepted technique for accessing system services, so one of ordinary skill in the art would conclude that Motorola's products enable object-oriented applications to access system services to obtain thread information. Motorola's experts disagree, arguing that Aldrich essentially admits that Motorola's products don't infringe the '867 patent: because these products merely return information already stored in the virtual machine and never query the actual operating system for thread information, they do not access the Linux kernel. This clash of experts created a disputed issue of material fact, precluding summary judgment.

Motorola's summary judgment motion regarding Apple '002 is denied

Motorola has moved for summary judgment of noninfringement of Apple's U.S. Patent Number 6,493,002 ("Method and Apparatus for Displaying and Accessing Control and Status Information in a Computer System"). The patent covers the well-known "control strips" or "toolbars" commonly found on personal computer operating systems. A toolbar is a window that, when it appears at all, appears in a top "window layer" that other windows—those displaying files or running programs—cannot overlap or block though they can overlap one another. The toolbar displays basic status information about the computer system through "display areas" (such as icons or other graphical representations), which might indicate the system's audio volume, screen brightness, remaining battery power, internet connection strength, clock time, and so forth. As described by the '002 patent, the user can interact with at least one of the display areas using the cursor or keyboard. Such interactive display areas are "buttons" on the toolbar—they can be clicked to make changes to the computer system (for instance by diminishing or increasing screen brightness). The point of including these display

areas on a single window (the toolbar) and of placing that window in a top window layer is to allow the user to view general information about his computer system and change basic settings quickly and conveniently.

Apple asserts that Motorola smartphones and tablets running the Android operating system infringe claims 1, 21, and 46 of the '002 patent. Each of the accused devices can display two windows (as we'll see, the two might in fact be only one) that Apple says infringe the '002 patent. The "status bar" is consistently displayed at the top of the accused products' screen and contains display areas—icons and other indicators—akin to those on a computer's toolbar, such as an internet connection icon, a new email icon, a battery life indicator, a clock, and so forth. The "notification window" is not usually displayed but can be brought up by the user by way of a downward finger swipe from the status bar. The notification window contains further display areas to indicate system status, some of which are interactive buttons; by contrast, none of the display areas on the status bar are interactive—none can be manipulated by the user to change system settings—though the status bar itself can be swiped to bring up the notification window.

Claim 1 of the patent, which is largely representative of the other claims asserted against Motorola, provides:

> An interactive computer-controlled display system comprising: a processor; a data display screen [e.g., a computer monitor] coupled to the processor; a cursor control device [e.g., a mouse or touchpad] coupled to said processor for positioning a cursor on said data display screen; a window generation and control logic coupled to the processor and data display screen to create an operating environment for a plurality of individual programming modules associated with different application programs that provide status and/or control functions, wherein the window generation and control logic generates and displays **a first window region having a plurality of display areas on said data display screen,** wherein the first window region is independently displayed and independently active of any application program, and wherein each of the plurality of display areas is associated with one of the plurality of individual programming modules, **the first window region and the plurality of independent display areas implemented in a window layer that appears on top of application programming windows that may be generated;** and an indicia generation logic coupled to the data display screen to execute at least one of the plurality of individual programming modules to generate information for display in one of the plurality of display areas in the first window region, **wherein at least one of the plurality of display areas and its associated programming module is sensitive to user input,** and further wherein the window generation and control logic and the indicia generation logic use **message-based communication to exchange information** to coordinate activities of the indicia generation logic to enable interactive display activity.

The boldface passages are those contested by the parties. In particular, Motorola argues that it is entitled to summary judgment because Apple has failed to

raise a genuine factual dispute that Motorola's products infringed either or both of two limitations of the '002 patent: the "first window region" limitation and the "message-based communication" limitation.

The parties have not adequately explained one of the grounds on which Motorola seeks summary judgment: the "message-based communication" element of claim 1 and the apparently similar "sending a message to a programming module" element in claims 21 and 46. The issue does not appear to be especially complex, but the parties' briefing of it is impenetrable. Because Motorola has failed to make its argument comprehensible, either in its brief or at oral argument, it is not entitled to summary judgment on this ground.

The second ground for summary judgment advanced by Motorola—the "first window region" limitation—has several elements: It is just a single window, rather than a group of windows; it has multiple display areas, and at least one such display area is "sensitive to user input," or interactive (in other words, it is a button); and it is "implemented in a window layer that appears on top of application programming windows that may be generated."

Both the status bar and the notification window satisfy the "top window layer" limitation. Each of them, when displayed, appears on top of any other windows and cannot be overlapped by another window. Finally, in none of the devices is any display area on the status bar interactive. The fact that the status bar may itself be interactive—a downward swipe on it brings up the notification window—does not mean that it contains any interactive display areas; Apple does not argue that it does. Motorola's status bar taken alone therefore does not infringe the '002 patent.

But the notification window does contain interactive display areas. It is true that the notification window, unlike a conventional personal computer toolbar or the accused devices' status bar, is not usually visible and can only be called up by the user's downward swiping motion. This prevents the notification window from fulfilling the toolbar's primary purpose, as stated in the patent's specification, of making core status indicators conveniently visible and accessible to users. But the claims, not the specification, determine the scope of the patent and so must be analyzed to determine infringement. *CollegeNet, Inc. v. ApplyYourself, Inc.*, 418 F.3d 1225, 1231 (Fed. Cir. 2005); *Bayer AG v. Biovail Corp.*, 279 F.3d 1340, 1348 (Fed. Cir. 2002). Motorola has not pointed to any language in the asserted claims that suggests that they are limited to windows that are always or usually visible onscreen or that appear automatically without user action.

Apple told the PTO that the "window layer" claim term meant that "the present invention is directed at using individual programming modules that generate displays that are *always visible on a top layer*" (emphasis added) and distinguished a prior reference from the '002 invention on the ground that the prior

reference "only allow[ed] the user an unobstructed view of the system if a button is selected"—just like the notification window, which comes into view only when the user performs a downward swipe from the status bar. Yet the asserted claims, as finally approved by the PTO, do not require that the window be always visible but only that it appear on top of all other windows and never be obstructed when it is generated; nor is the patent limited to toolbars that can be viewed without the user's pressing a button. "[B]ecause the prosecution history represents an ongoing negotiation between the PTO and the applicant, rather than the final product of that negotiation, it often lacks the clarity of the specification and thus is less useful for claim construction purposes." *Phillips v. AWH Corp.*, 415 F.3d 1303, 1317 (Fed. Cir. 2005). The patent's "first window region" limitation therefore does not provide a ground for granting summary judgment to Motorola—the status window cannot infringe the '002 patent, but for all that Motorola has shown, the notification window may.

My determination that the notification window might infringe is dispositive of this motion, but I add that another argument of Apple's—that the status bar and notification window together actually constitute a single window, which itself infringes the '002 patent—is a nonstarter. Whether the status bar and the notification window are two separate windows or a single "composite" window is a semantic question to which there is no answer that a jury could give; the parties have not suggested what kind of evidence would show that Motorola's status bar and notification window are only a single window that cannot be seen all at once, but only in two glances, by swiping the window down from the top of the screen. Apple bears the burden of proving that Motorola has infringed its patent, and thus would have to establish the unity of the status bar and notification window taken together to show infringement under this composite-window theory. It cannot do so, though it may be able to prove that the notification window taken alone infringes its patent.

Claim construction of the Apple '263 patent

Apple's '263 patent (U.S. Patent No. 6,343,263) describes a computer system that performs realtime signal processing on serially transmitted data. Claim 1 lays out the architecture of this signal-processing system. It includes two subsystems—a "host central processing unit" and a "realtime signal processing subsystem"—connected by a realtime application program interface ("realtime API" for short). The realtime API requests realtime services (for instance, video-image processing) from the realtime signal processing subsystem on behalf of applications running on the host subsystem, receives instructions from the host subsystem, and supplies the realtime signal processing subsystem with instructions for signal processing.

Apple alleges that Motorola phones and tablets infringe the '263 patent. Motorola countered that its devices don't infringe because their signal-processing systems lack a realtime API, and on that basis it moved for a summary judgment ruling that it had not infringed. I decided this factual dispute couldn't be resolved without construction of the claim term "realtime API."

Claim 1 asserts "at least one *realtime application program interface* (API) coupled between the [host] subsystem and the realtime signal processing subsystem to allow the subsystem to interoperate with said realtime services" (emphasis added). There is no dispute over the meaning of the word "realtime" generally: to be "realtime" a system "must satisfy explicit (bounded) response-time constraints or risk severe consequences," namely degraded performance. Philip A. Laplante, *Real-Time Systems Design and Analysis: An Engineer's Handbook* 10 (1993). But the parties differ over how the word should be understood in conjunction with an API. Motorola asserts that a "realtime API" is a an API that has realtime functionality, which Motorola defines as "facilitating constant bit rate handling," while Apple defines it as an API that facilitates realtime signal processing, that is, that enables realtime interaction between the two subsystems.

Motorola points out that the inventors identified some components of their invention as "realtime," but left the word out of claim 31, which describes an API that "issu[es] requests to the realtime engine to perform data transformations." An API that does that falls within Apple's broad definition of realtime API as any API that links the realtime signal-processing system to the host system. Realtime API in claim 1 must, Motorola reasons, mean something different from the API in claim 31, since the latter, even though it fits Apple's proposed definition for purposes of claim construction, is not called a "realtime API."

There may indeed be a presumption that "API" in claim 31 and "realtime API" in claim 1 have different meanings, *Forest Laboratories, Inc. v. Abbott Laboratories*, 239 F.3d 1305, 1310 (Fed Cir. 2001); *Tandon Corp. v. International Trade Commission*, 831 F.2d 1017, 1023 (Fed. Cir. 1987), but that presumption can be overcome, for example by the patent specification. *Id*; see also *Philips v. AWH Corp.*, 415 F.3d 1303, 1315–17 (Fed Cir. 2005) (en banc). And '273's specification—"the single best guide to the meaning of a disputed term," *Vitronics Corp. v. Conceptronic, Inc.*, 90 F.3d 1576, 1582 (Fed. Cir. 1996)—consistently refers to the realtime API as just the "interface" or the "API." So too did the inventors during the patent's prosecution.

Neither the specification nor the prosecution history suggests that the realtime API disclosed in claim 1 must itself have realtime functionality, as by facilitating constant bit-rate handling (e.g., preventing a streaming video from being interrupted by incoming data). The specification makes clear that the realtime signal-processing subsystem must assure constant bit-rate handling, but not that

the realtime API must do so. The patent does describe one embodiment of the invention as involving the realtime API's instructing the real-time engine on how to process the proper number of bits per word, but this doesn't imply that the API itself does the constant bit-rate processing, and is therefore consistent with how the realtime API is described everywhere else in the patent: as an interface between the realtime subsystem and the host subsystem that receives commands from the host and passes the instructions along to the realtime subsystem, which does the realtime processing.

I therefore construe "realtime application program interface" in claim 1 of the '263 patent to mean an "API that allows realtime interaction between two or more subsystems."

United States Circuit Judge

January 25, 2012