Ancora Techs. v. Apple, Inc.

United States Court of Appeals for the Federal Circuit

March 3, 2014, Decided

2013-1378, 2013-1414

Reporter
744 F.3d 732 *; 2014 U.S. App. LEXIS 3895 **; 109 U.S.P.Q.2D (BNA) 2135 ***; 2014 WL 803104

ANCORA TECHNOLOGIES, INC., Plaintiff-Appellant, v. APPLE, INC., Defendant-Cross Appellant.

Subsequent History: US Supreme Court certiorari denied by *Apple, Inc. v. Ancora Techs., Inc., 135 S. Ct. 957, 190 L. Ed. 2d 832, 2015 U.S. LEXIS 277 (U.S., 2015)*

Prior History: [1]** Appeals from the United States District Court for the Northern District of California in No. 11-CV-6357, Judge Yvonne Gonzalez Rogers.

Ancora Techs., Inc. v. Apple Inc., 2012 U.S. Dist. LEXIS 183045 (N.D. Cal., Dec. 31, 2012)

Disposition: AFFIRMED IN PART, REVERSED IN PART, AND REMANDED.

Case Summary

Overview

HOLDINGS: [1]-In a patent infringement case involving a patent that claimed methods for verifying that a software program on a computer was licensed to be there, the district court erred in its construction of "program" as limited to application programs, thereby excluding operating systems from the class of programs that the claimed method checks for authorization under a license. There was no support in this case, in either the specification or prosecution history, to depart from the ordinary meaning of "program," as the district court's construction did; [2]-District court was correct to reject a challenge to "volatile memory" and "non-volatile memory" as indefinite and not meeting the standards of *35 U.S.C.S. § 112(b)* because there was no dispute that the terms had a meaning that was clear, settled, and objective in content, and the specification did not supplant that understanding.

Outcome

District court's construction of "program" as limited to application programs reversed. District court's conclusion that the terms "volatile memory" and "non-volatile memory" were not indefinite affirmed.

Counsel: JOHN S. LEROY, Brooks Kushman P.C., of Southfield, Michigan, argued for plaintiff-appellant. With him on the brief were MARK A. CANTOR, MARC LORELLI, and JOHN P. RONDINI.

DEANNE E. MAYNARD, Morrison & Foerster LLP, of Washington, DC, argued for defendant-cross appellant. With her on the brief were BRIAN R. MATSUI and NATALIE R. RAM, OF WASHINGTON, DC; MICHAEL A. JACOBS, RICHARD S.J. HUNG, and FRANCIS C. HO, of San Francisco, California; and BITA RAHEBI, of Los Angeles, California.

Judges: Before RADER, Chief Judge, TARANTO, and CHEN, Circuit Judges.

Opinion by: TARANTO

Opinion

[*733] [*2136]** TARANTO, *Circuit Judge*.

Ancora Technologies, Inc., owns U.S. Patent No. 6,411,941, which claims methods for verifying that a software program on a computer is not there without authorization, but is licensed to be there. In December 2010, Ancora sued Apple Inc., alleging that products running Apple's iOS operating system infringed the '941 patent. The United States District Court for the Northern District of California construed the claims. *Ancora Techs., Inc. v. Apple Inc., 11-CV-06357, 2012 U.S. Dist. LEXIS 183045, 2012 WL 6738761 (N.D. Cal. Dec. 31, 2012)*. **[**2]** Ancora stipulated to summary judgment of non-infringement under the district court's construction of the claim term "program." The district court subsequently entered final judgment dismissing all claims and counterclaims. Ancora appeals the district court's construction of "program," while Apple cross-appeals the district court's holding that the terms "volatile memory" and "non-volatile memory" are not indefinite. We affirm in part, reverse in part, and remand.

BACKGROUND

The '941 patent, entitled "Method of Restricting Software Operation within a License Limitation," describes a method of preventing unauthorized software use by checking whether a software program is operating within a license and stopping the program or taking other remedial action if it is not. The specification states that methods for checking license coverage of software were known in the art at the time the inventors applied for the '941 patent. But some of those methods were vulnerable to hacking, the specification observes, while others were expensive and inconvenient to distribute. '941 patent, col. 1, lines 19-32.

The specification describes **[**3]** a method that it says over-comes those problems. In particular, it discloses using the memory space associated with the computer's basic input/output system (BIOS), rather than other memory space, to store appropriately encrypted license information to be used in the verification process. *See,* lines 45-48; *id*., col. 5, lines 19-24. It states that, while the contents of the BIOS memory space may be modified, the level of programming expertise needed to do so is unusually high, and the risk of accidentally damaging the BIOS and thereby rendering the computer inoperable "is too high of a risk for the ordinary software hacker to pay." *Id*., col. 3, lines 4-14. Thus, the inventors stated that their method makes use of the existing computer hardware (eliminating the expense and inconvenience of using additional **[*734]** hardware), while storing the verification information in a space that is harder and riskier for a hacker to tamper with than storage areas used by earlier methods.

Claim 1, the only independent claim Ancora asserts, is representative:

> 1. A method of restricting software operation within a license for use with a computer **[**4]** including an erasable, non-volatile memory area of a BIOS of the computer, **[***2137]** and a volatile memory area; the method comprising the steps of:
>
> selecting a program residing in the volatile memory,
>
> using an agent to set up a verification structure in the erasable, non-volatile memory of the BIOS, the verification structure accommodating data that includes at least one license record,
>
> verifying the program using at least the verification structure from the erasable non-volatile memory of the BIOS, and
>
> acting on the program according to the verification.

Id., col. 6, line 59, through col. 7, line 4.

The parties have not meaningfully disagreed about the ordinary meaning of the claim terms at issue on appeal: "program," "volatile memory," and "non-volatile memory." But Apple has relied on examples in the specification, as well as statements by the applicants and the examiner during prosecution, to argue that the terms do not have those ordinary meanings in this patent. Specifically, Apple has argued that the term "program" (which is to be verified for authorization under a license) is limited to an *application* program, *i.e*., one that relies on an operating system in order to run, thus excluding an **[**5]** operating system itself. Apple also has argued that the terms "volatile memory" and "non-volatile memory" are indefinite because an example given in the specification is irreconcilable with the ordinary meaning of the terms. The district court agreed with Apple on the first point (finding non-infringement on

that basis) but disagreed with Apple on the second (rejecting invalidity for indefiniteness on that basis). Both sides appeal. We have jurisdiction under 28 U.S.C. § 1295(a)(1).

DISCUSSION

Claim construction and indefiniteness are matters of law that this court reviews de novo. Cybor Corp. v. FAS Techs., Inc., 138 F.3d 1448, 1451 (Fed. Cir. 1998); Praxair, Inc. v. ATMI, Inc., 543 F.3d 1306, 1319 (Fed. Cir. 2008).

A

Ancora challenges the district court's conclusion that the term "program" is limited to application programs, thereby excluding operating systems from the class of programs that the claimed method checks for authorization under a license. We agree with Ancora. A claim term should be given its ordinary meaning in the pertinent context, unless the patentee has made clear its adoption of a different definition or otherwise disclaimed that meaning. *See, e.g.*, Thorner v. Sony Computer Entm't Am. LLC, 669 F.3d 1362, 1365 (Fed. Cir. 2012). **[**6]** There is no reason in this case to depart from the term's ordinary meaning.

Apple nowhere seriously disputes that the ordinary meaning of the word "program" in the computer context encompasses both operating systems and the applications that run on them (as well as other types of computer programs). And the district court explained that, although the term "program" may have many different meanings depending on the context, "to a computer programmer" a program is merely a "set of instructions" for a computer. **[*735]** Ancora, 2012 U.S. Dist. LEXIS 183045, 2012 WL 6738761, at *7. That clear meaning governs here, we conclude, because there is nothing sufficient to displace it.

The claims themselves point against a narrowing of the term "program" to application programs. Claim 1 recites a "method of restricting software operation" (if license coverage of the software cannot be verified) and refers to the restricted software simply as a "program." '941 patent, col. 6, line 59, through col. 7, line 4. In contrast, independent claim 18, which is not asserted here, recites a "method for accessing an *application* software program" and then repeatedly refers to the "*application* software program." *Id*., col. 8, lines 31-52 (emphases added). **[**7]** Although claim 18 is not a dependent principle is often of limited importance, the difference in terminology tends to reinforce, rather than undermine, adoption of the broad ordinary meaning of "program" by itself.

Nothing in the specification clearly narrows the term "program." The general disclosure in the specification refers to the to-be-verified software as a "software program," "software," or a "program," without limiting the subject matter to particular types of programs. *See, e.g., id*., col. 1, lines 7, 40; *id*., col. 2, lines 63, 66. The only instances in which the specification discusses using the claimed invention to verify *application* programs are **[***2138]** found in examples that the specification makes clear are not limiting. *See, e.g., id*., col. 1, line 45 (characterizing the example at col. 1, line 46, through col. 2, line 59, as "a specific non-limiting example"); *id*., col. 3, line 33 (describing a "preferred embodiment"); *id*., col. 4, line 66 (characterizing the preferred embodiment described in columns 5 and 6 as a "non-limiting example only"). Such examples are "not sufficient to redefine the term . . . to have anything other **[**8]** than its plain and ordinary meaning." IGT v. Bally Gaming Int'l, Inc., 659 F.3d 1109, 1118 (Fed. Cir. 2011). Thus, nothing in the specification would lead one of ordinary skill in the art to understand that the claims use "program" in a sense narrower than its ordinary meaning.

The prosecution history requires more extended discussion, but it too does not require a meaning that substitutes for the ordinary one. In reading the prosecution history, it is important to keep in mind the distinction between a program whose coverage by a license is being checked and a piece of software that embodies the patent's claimed method of checking. The term "program" in the claims refers exclusively to the to-be-verified program. Indeed, neither the specification nor the claims use the term "program" to refer to software (a set of instructions) that, when run, performs the claimed verification steps, instead referring to the invention as a "method," "system," or, in one instance, a "license verifier application." *See, e.g*., '941 patent, col. 1, lines 6-8; *id*., col. 2, line 14.

The prosecution-history statements that Apple cites are focused on the verifying software, not clearly (or in any event relevantly) **[**9]** on the to-be-verified program, and so cannot support Apple's narrowing argument. Specifically, the applicants distinguished their invention over a combination of two references: one disclosed storage in the BIOS memory area by the BIOS software itself; the other disclosed software implemented in or through an operating system. The applicants explained

that their invention differed from the prior art in that it *both* operated as an application running through an operating system *and* used the BIOS level for data storage and retrieval—a combination that was not previously **[*736]** taught and that an ordinarily skilled application writer would not employ:

> [T]here is no suggestion or motivation to combine Misra and Ewertz in the manner suggested in the Office Action. BIOS is a configuration utility. Software license management *applications, such as the one of the present invention*, are operating system (OS) level programs. . . . [W]hen BIOS is running, the computer is in a configuration mode, hence OS is not running. Thus, BIOS and OS level programs are normally mutually exclusive.
> . . .
>
> [T]he present invention proceeds against conventional wisdom in the art. Using BIOS to store application data such **[**10]** as that stored in Misra's local cache for licenses is not obvious. The BIOS area is not considered a storage area for computer applications. An ordinary skilled artisan would not consider the BIOS as a storage medium to preserve application data for at least two reasons.
> First, . . . [a]n ordinary person skilled in the art makes use of OS features to write data to storage mediums. There is no OS support whatsoever to write data to the system BIOS. Therefore, an ordinary person skilled in the art would not consider the BIOS as a possible storage medium. . . .
> Second, no file system is associated with the BIOS. . . . This is further evidence that OS level application programmers would not consider the BIOS as a storage medium for license data.

Amendment dated Feb. 5, 2002, at 6-7, in Appl. No. 09/164,777 (emphasis added).

The reference to the invention as a "license management application[]" and the identification of persons of ordinary skill in the relevant art as "application programmers" who "make[] use of OS features" demonstrate that the applicants understood that their claimed methods would be implemented as application software, rather than lower-level system software. But those representations, **[**11]** made in distinguishing prior art, concerned software that implemented the invented method. The to-be-verified software is different from the verifying software. The statements from the prosecution history on which Apple relies do not say that the program being verified must be an application program. Even the reference to "application data" in describing Misra, even **[***2139]** if read to refer to data about a to-be-verified program (which is not clear), does not distinguish Misra, or limit the present claims, on that basis.[1]

Other prosecution statements cited by Apple no more establish the narrowing it urges. Although Apple makes much of language about storing "application data" in the BIOS area, Amendment dated Feb. 5, 2002, at 7, nothing in the applicants' statements indicates that the "application" in question is the to-be-verified software, as opposed to the verifying software; and in any event, the language does not rise to the level of a disclaimer regarding nature of the to-be-verified software. Likewise, although **[**12]** the examiner stated in his reasons for allowance that "the closest prior art systems, singly or collectively, do not teach licensed programs running at the OS level interacting with a program verification structure stored in the BIOS," Notice of Allowability dated Feb. 20, 2002, at 4, in Appl. No. 09/164,777, that statement is at worst a slip: under the claims, it is indisputably **[*737]** the verifying software that interacts with the verification structure. In any event, the statement is not the applicants' statement. *See* Salazar v. Procter & Gamble Co., 414 F.3d 1342, 1345 (Fed. Cir. 2005) (remarks in the examiner's statement of reasons for allowance insufficient to limit claim scope). And, as quoted above, the applicants were clear that the OS-level language referred to the verifying software.

Nor, finally, did the applicants represent in the prosecution history, or elsewhere, that verification must occur before the to-be-verified program is loaded (so that software for performing verification that depended on a running operating system could not verify the operating system). To the contrary, the first step in claim 1 is "selecting a program residing in the volatile memory," '941 patent, **[**13]** col. 6, line 63, and the examiner understood that "software would have to be loaded a priori in order to reside in volatile memory." Office Action dated Jan. 15, 2002, at 3, in Appl. No. 09/164,777 (emphasis added). The specification does describe an embodiment in which the verifying software is "a priori running in the computer" when a to-be-verified program is loaded into memory. '941 patent, col. 2, lines 14-15. But that is part of what is merely a "non-limiting example" that is "by no means binding." *Id*., col. 1, line 45; *id*., col. 2, line 61.

---

[1] We do not have before us a contention that the verification software must be an "application." We do not address whether

We conclude that the district court erred in construing "program" to mean "a set of instructions for software *applications* that can be executed by a computer." *Ancora, 2012 U.S. Dist. LEXIS 183045, 2012 WL 6738761, at *10* (emphasis added).

B

In its cross-appeal, Apple challenges the district court's rejection of its contention that the claims at issue are invalid because the terms "volatile memory" and "non-volatile memory" are indefinite. Under what is now *35 U.S.C. § 112(b)*, a claim must be "sufficiently definite to inform the public of the bounds of the protected invention, i.e., what subject matter is covered by the exclusive rights of the patent." *Halliburton Energy Servs., Inc. v. M-I LLC, 514 F.3d 1244, 1249 (Fed. Cir. 2008)*. **[**14]** The Supreme Court currently is considering how to refine the formulations for applying the definiteness requirement. *See Nautilus, Inc. v. Biosig Instruments, Inc.*, Sup. Ct. No. 13-369, *cert. granted*, *134 S. Ct. 896, 187 L. Ed. 2d 702, 2014 WL 92363 (2014)*. In this case, we think that we can reject the indefiniteness challenge without awaiting the Court's clarification. However other circumstances may be evaluated, it suffices to reject the challenge in this case that the claim language and the prosecution history leave no reasonable uncertainty about the boundaries of the terms at issue, even considering certain aspects of the specification that could engender confusion when read in isolation.

Most importantly, there is no dispute that the terms "volatile memory" and "non-volatile memory" have a meaning that is clear, settled, and objective in content. Both parties and the district court agreed that, as a general matter, "[t]o one of ordinary skill in the art, a volatile memory is memory whose data is not maintained when the power is removed and a non-volatile memory is memory whose data is maintained when the power is removed." *Ancora, 2012 U.S. Dist. LEXIS 183045, 2012 WL 6738761, at *4*. That meaning leaves the relevant public with a **[**15]** firm understanding of the scope of the claim terms, unless something exceptional sufficiently supplants that understanding. Apple argues that the specification does so. We conclude otherwise.

[*2140]** Apple's argument rests on the fact that, three times, the specification uses language **[*738]** referring to a hard disk as an example of volatile memory. '941 patent, col. 1, line 21; *id.*, col. 3, line 9; *id.*, col. 4, line 53. All sides agree that a hard disk maintains data when referred to as "volatile memory." Apple contends that because "a hard disk is a quintessential example of non-volatile memory" and "the specification does not explain *how* a hard disk can fall into the category of volatile memory . . . or what characteristics differentiate volatile from non-volatile memory . . . a person of ordinary skill would not know what falls within the scope of the claims." Cross-Appellant Br. at 38.

We are not persuaded that Apple's conclusion is properly drawn from the passages on which it relies. To begin with, the terms at issue have so clear an ordinary meaning that a skilled artisan would not be looking for clarification in the specification. There **[**16]** is no facial ambiguity or obscurity in the claim term. Moreover, the specification nowhere purports to set out a definition for "volatile" or "non-volatile" memory, and nothing in it reads like a disclaimer of the clear ordinary meaning. Under our claim-construction law, a clear ordinary meaning is not properly overcome (and a relevant reader would not reasonably think it overcome) by a few passing references that do not amount to a redefinition or disclaimer.

In this case, moreover, a skilled artisan would appreciate that the passages at issue have a possible meaning that is not (what would be surprising) starkly irreconcilable with the clear meaning of "volatile" and "non-volatile" memory, which are the claim terms. (The claims do not mention a hard disk at all, and the only specific example of "volatile" memory set out in the claims is Random Access Memory (RAM), '941 patent, col. 8, lines 1-2, which all agree is "volatile" in the ordinary sense.) As the district court observed, it is well known that a computer's hard disk is routinely used as "virtual" memory to provide temporary storage when there is insufficient RAM to complete an operation, *Ancora, 2012 U.S. Dist. LEXIS 183045, 2012 WL 6738761, at *5*, in **[**17]** which case (it is undisputed) the data become inaccessible through the usual means once power is removed (even if the data can still be found on the hard disk by more sophisticated means), Cross-Appellant Br. at 50; J.A. 1672. This explanation for the otherwise-perplexing example of a hard disk as "volatile" memory finds support in the specification's statement that "the volatile memory is a *RAM e.g. hard disk* and/or internal memory of the computer." '941 patent, col. 4, lines 53-54 (emphasis added). Although oddly phrased, the reference to a "hard disk" as an example of RAM suggests that the patentee meant to refer to the hard disk only in its capacity as supplemental memory in conjunction with the main RAM, rather than to assert, in a passing and indirect

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.