MORRISON & FOERSTER LLP
MICHAEL A. JACOBS (Bar No. 111664)
mjacobs@mofo.com
MARC DAVID PETERS (Bar No. 211725)
mdpeters@mofo.com
DANIEL P. MUINO (Bar No. 209624)
dmuino@mofo.com
755 Page Mill Road, Palo Alto, CA 94304-1018
Telephone: (650) 813-5600 / Facsimile: (650) 494-0792

BOIES, SCHILLER & FLEXNER LLP
DAVID BOIES (Admitted *Pro Hac Vice*)
dboies@bsfllp.com
333 Main Street, Armonk, NY 10504
Telephone: (914) 749-8200 / Facsimile: (914) 749-8300
STEVEN C. HOLTZMAN (Bar No. 144177)
sholtzman@bsfllp.com
1999 Harrison St., Suite 900, Oakland, CA 94612
Telephone: (510) 874-1000 / Facsimile: (510) 874-1460

ORACLE CORPORATION
DORIAN DALEY (Bar No. 129049)
dorian.daley@oracle.com
DEBORAH K. MILLER (Bar No. 95527)
deborah.miller@oracle.com
MATTHEW M. SARBORARIA (Bar No. 211600)
matthew.sarboraria@oracle.com
500 Oracle Parkway, Redwood City, CA 94065
Telephone: (650) 506-5200 / Facsimile: (650) 506-7114

Attorneys for Plaintiff
ORACLE AMERICA, INC.

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN FRANCISCO DIVISION

ORACLE AMERICA, INC. Plaintiff, v. GOOGLE INC. Defendant.	Case No. CV 10-03561 WHA **ORACLE AMERICA, INC.'S OPPOSITION TO GOOGLE INC.'S MOTION TO STRIKE PORTIONS OF THE MITCHELL PATENT REPORT** Date: September 29, 2011 Time: 8:00 a.m. Dept.: Courtroom 8, 19th Floor Judge: Honorable William H. Alsup

TABLE OF CONTENTS

Page

TABLE OF AUTHORITIES

Page(s)

CASES

OTHER AUTHORITIES

I. INTRODUCTION

Oracle opposes Google's motion to strike portions of the Mitchell Opening Patent Infringement Report ("Report").[1] Oracle's Infringement Contentions ("ICs") notified Google in detail how Android infringes the asserted claims and which Android "instrumentalities" are accused. The ICs span more than 400 pages. That Google needed to attach a 59-page excerpt from the ICs just to explain Oracle's infringement theories as to two asserted claims reveals how detailed the ICs are. The Report provides only additional explanation and evidence to support the infringement theories disclosed in Oracle's ICs. Where the Report quotes from additional Android source code files, it does so only to explain in more detail how Android infringes the asserted patent claims. To support Oracle's contention that all Android devices have the same infringing elements, the Report shows that device manufacturers use the Google-provided source code on their Android devices without modifying those elements. The Report is fully supported by the ICs. It does not introduce any new infringement theories or accused instrumentalities.

Google's motion is not properly directed to whether the Report deviates impermissibly from the ICs. Instead, Google's motion is based on a belated challenge to the adequacy of Oracle's ICs. But the time for challenging the ICs passed long ago. Moreover, at a Court-directed meet-and-confer on the ICs, Oracle invited Google to pose any questions it wished if it had difficulty understanding the scope of Oracle's infringement theories. Google never responded. Google is responsible for failing to seek clarification, whether informally as invited or formally by moving to compel more detailed ICs. Google should not be allowed to use the Report as a vehicle to express its dissatisfaction with the ICs at this late stage in the case.

II. LEGAL STANDARD

The applicable legal standard may be found in the text of the local rule and from decisions in this district and elsewhere that address similar issues. The Patent Local Rules require an identification of each Accused Instrumentality and a "chart identifying specifically where each

[1] Portions of Prof. Mitchell's Opening Patent Infringement Report are attached as Exhibit A to the Declaration of Mark H. Francis in Support of Defendant Google Inc.'s Motion to Strike Portions of the Mitchell Patent Report ("Francis Decl.").

limitation of each asserted claim is found within each Accused Instrumentality." Pat. L.R. 3-1(b)-(c). Google incorrectly states that "Oracle was also required to provide 'pinpoint citations' to the allegedly infringing source code in its ICs." (Mot. at 4.) The Patent Local Rules do not require such detail. Rather, infringement contentions "must be sufficient to provide reasonable notice to the defendant why the plaintiff believes it has a 'reasonable chance of proving infringement.'" *Shared Memory Graphics LLC v. Apple, Inc.*, No. C-10-2475 VRW (EMC), 2010 U.S. Dist. LEXIS 138868, at *10-11 (N.D. Cal. Dec. 30, 2010) (quoting *View Eng'g, Inc. v. Robotic Vision Sys., Inc.*, 208 F.3d 981, 986 (Fed. Cir. 2000)). Oracle's ICs did that, and more.

Although expert reports on patent infringement may not introduce infringement theories not previously set forth in the patentee's infringement contentions, the "scope of infringement contentions and expert reports are not, however, coextensive." *Fenner Invs., Ltd. v. Hewlett-Packard Co.*, No. 6:08-CV-273, 2010 U.S. Dist. LEXIS 17536, at *6-7 (E.D. Tex. Feb. 26, 2010). "[E]xpert reports are expected to provide additional information regarding the specific factual bases for plaintiffs' infringement contentions." *Sicurelli v. Jeneric/Pentron Inc.*, No. 03-CV-4934 (SLT) (KAM), 2005 U.S. Dist. LEXIS 42233, at *33 (E.D.N.Y. May 3, 2005). Thus, expert reports may provide additional explanation and evidence to support the infringement theories disclosed in the infringement contentions. *See DataTreasury Corp. v. Wells Fargo & Co.*, No. 2:06-CV-72 DF, 2010 U.S. Dist. LEXIS 110658, at *23 (E.D. Tex. Sept. 13, 2010) (denying defendant's motion to strike portions of plaintiff's infringement expert report because report did "not substantially deviate from [the] infringement contentions"). This is only common sense—otherwise, the Patent Local Rules would have done away with infringement and invalidity reports altogether.

III. ARGUMENT

A. Response to Critique A: The Report on Infringement of Claim 11 of the '104 Patent Is Supported by the ICs

The '104 patent is directed to systems and methods for resolving symbolic references to numerical references. (*See* ECF No. 137 ("Claim Construction Order") at 22.) Claim 11 of the '104 patent recites, inter alia, "determining a numerical reference corresponding to said symbolic reference [and] *storing* said numerical references." (Declaration of Mark H. Francis in Support of Defendant Google Inc.'s Motion to Strike Portions of the Mitchell Patent Report ("Francis Decl.") Ex. B, '104 patent col.7 ll.5-14 (emphasis added).)

Oracle's ICs for the '104 patent disclose two different infringement theories. (Declaration of Marc D. Peters in Support of Oracle America, Inc.'s Opposition to Google Inc.'s Motion to Strike Portions of the Mitchell Patent Report ("Peters Decl.") Ex. 1, ICs Ex. A.) The ICs include a detailed, sixty-seven-page claim chart mapping specific Android components or functionalities to each claim limitation. (*See id.*)

The first infringement theory involves the Android "dexopt" tool, which is a tool for verifying and optimizing the classes in a Dalvik executable (".dex") file. The ICs summarize this infringement theory by quoting an admission from Dan Bornstein, an Android engineer, that when a .dex file arrives on a device, "it will have symbolic references to methods and fields, but afterwards it might just be . . . a simple integer vtable offset so that when, for invoking a method, instead of having to do say a string-based lookup, it can just simply index into a vtable." (*Id.* at 2.) As Mr. Bornstein described, the dexopt tool tries to resolve the symbolic references in the instructions contained in the .dex file and stores or replaces the symbolic references with corresponding numerical references.

The ICs map the DexOptimize.c and Optimize.c Android source code files to the "storing" limitation. (*Id.* at 43-67.) Those source files include the dvmOptResolveInstField and dvmOptResolveStaticField functions, which, as their names suggest, resolve a symbolic reference to a field. (*Id.* at 49-53.[2]) The quoted source code shows that both functions call the dvmDexSetResolvedField function. (*Id.* at 50, 53.) The comments to those calls show that the dvmDexSetResolvedField function stores the resolved reference. (*Id.* at 50 ("Add it to the resolved table so we're faster on the next lookup."); *see also id.* at 52 (the same).) The ICs thus specify how the dexopt tool stores the resolved references for later use, and thereby infringe the '104 patent.

[2] The referenced functions and comments are highlighted in the claim charts attached to the Peters Declaration.

The Report tracks the "dexopt" infringement theory. The Report quotes the same admission by Bornstein and explains that the dexopt tool "looks at every instruction in a method and tries to resolve symbolic references and determines, stores, or replaces symbolic references with corresponding numeric references (*i.e.*, Android quickens the instructions where Android can)." (*See* Report ¶¶ 233, 248.) The Report quotes source code from the Optimize.c source file, just like the ICs, to show how Android meets the "storing" limitation. (*Id.* ¶¶ 250-54.) Specifically, the Report explains that the dvmOptResolveInstField function implemented in the Optimize.c source file can resolve a field and store data about the resolved field in a table for later use. (*Id.* ¶ 253.) The Report traces the dvmOptResolveInstField function to a call to the dvmDexSetResolvedField function. (*Id.*) That called function is implemented in the DvmDex.h file. (*See id.* ¶ 255.) The Report quotes from the DvmDex.h file to explain that the called function uses "a table indexed by the field index in the constant pool to write . . . the resolved field information." (*Id.*) Accordingly, the Report quotes from the DvmDex.h file to detail the role of the dvmDexSetResolvedField function, which is already disclosed in the ICs. The Report simply cites additional support for the theory disclosed in the ICs.

Oracle's second infringement theory involves the Dalvik virtual machine itself. The ICs summarize this infringement theory by referencing comments from an Android engineer, which are located in the Resolve.h file, that "constant pool references are resolved into pointers to VM structs." (Peters Decl. Ex. 1 at 2 (internal quotation marks omitted).) The ICs map the Resolve.c source file to the "storing" limitation. (*Id.* at 17-34.) That source file implements the dvmResolveClass function (among others), which, according to the comment to this call in the quoted source code, "cache[s] a copy of the lookup in the DexFile's 'resolved class' table, so future references to 'classIdx' are faster." (*Id.* at 18.) The dvmResolveClass function further contains a call to the dvmDexSetResolvedClass function, as identified in the ICs. (*Id.* at 21.) The comment to this call in the quoted source code explains that the called function stores the numerical reference (resClass, the pointer to the class object) to a resolved class. (*Id.* ("Add what we found to the list so we can skip the class search next time through.").)

The Report tracks this second infringement theory by also quoting from the Resolve.c source file to show how the Dalvik virtual machine meets the "storing" limitation. (Report ¶¶ 265-69.) The Report explains that the dvmResolveClass function implemented in the Resolve.c source file resolves symbolic references such as class names to numerical references such as a pointer to the class object. (*Id.* ¶ 269.) The Report traces the dvmResolveClass function to a call to the dvmDexSetResolvedClass function. (*Id.* ¶¶ 269, 271.) Again, that called function is implemented in the DvmDex.h file. (*See id.* ¶ 271.) The Report quotes from the DvmDex.h file to explain that the called function "serves to store the resolved symbolic reference after the class name gets resolved . . . ." (*Id.*) Accordingly, the Report quotes from the DvmDex.h file to provide more explanation about how the Dalvik virtual machine meets the "storing" limitation.

Similarly, Oracle's ICs identify calls in Resolve.c to functions for storing numerical references to methods (dvmDexSetResolvedMethod function, Peters Decl. Ex. 1 at 25, 28), fields (dvmDexSetResolvedField, *id.* at 30, 32), and strings (dvmDexSetResolvedString, *id.* at 33). All of those called functions are implemented in the DvmDex.h file. The Report discusses the Resolve.c source file and further explains the roles of the functions called in Resolve.c by quoting from the DvmDex.h file. (*See* Report ¶¶ 270, 271, 273, 277, 279, 282, 287, 289, 292, and 296.) So Google is overreaching in its motion—there is no cause to strike the discussion of Resolve.c in those Report paragraphs.

Oracle's ICs gave Google detailed accounts of Oracle's two Claim 11 infringement theories. That the ICs do not quote directly from the DvmDex.h file could not have "telegraphed an intention . . . *not* to rely on them." (Mot. at 3.) The claim charts identified the functions that are implemented in the DvmDex.h file, mapped them to the "storing" limitation, and thereby clarified Oracle's intent to rely on the DvmDex.h file for this limitation. Accordingly, the Report's reference to the DvmDex.h file is consistent with the two infringement theories disclosed in Oracle's ICs.

Google's effort to limit the scope of the Report to the specific source code reproduced verbatim in Oracle's ICs is unavailing. The decisions on which Google relies are inapposite. As a preliminary matter, *Vasudevan Software, Inc. v. Int'l Business Machines Corp.*, No. C09-05897 RS (HRL) (N.D. Cal. Feb. 18, 2011), is marked "NOT FOR CITATION" and should not have been cited to this Court. *See* Civ. L.R. 3-4(e). *Genentech, Inc. v. Trustees of the Univ. of Pa.*, No. C 10-2037-LHK (PSG), 2010 U.S. Dist. LEXIS 142450 (N.D. Cal. Dec. 13, 2010), involved claims of infringement against a breast cancer therapeutic drug, not source code. The patentee there was ordered to provide "pinpoint citations" to the accused drug in the context of a motion to compel detailed infringement contentions. *Id.* at *7-11. All of the decisions on which *Genentech* relied for requiring "pinpoint citations" to the accused instrumentalities involved motions to compel detailed infringement contentions. *See Big Baboon Corp. v. Dell, Inc.*, 723 F. Supp. 2d 1224, 1225 (C.D. Cal. 2010) (motion to compel adequate infringement contentions); *Diagnostic Sys. Corp. v. Symantec Corp.*, Nos. SACV 06-1211 DOC (ANx), SACV 07-960 DOC (ANx), 2009 U.S. Dist. LEXIS 53916, at *7-8 (C.D. Cal. June 5, 2009) (motion to compel "a more definite infringement statement"); *Am. Video Graphics, L.P. v. Elec. Arts, Inc.*, 359 F. Supp. 2d 558, 560-61 (E.D. Tex. 2005) (motion to supplement infringement "charts with specific references to the source code"). Requiring pinpoint citations to defendant's source code may be appropriate if the defendant moves to compel such citations on the ground that the defendant does not understand the scope of the infringement contentions without them. *See Big Baboon*, 723 F. Supp. 2d at 1227-28. But these cases do not stand for the proposition that expert reports must be limited to the lines of source code identified in the infringement contentions.

Google's actions demonstrate that Oracle's ICs are sufficient. Google complained to the Court about the adequacy of Oracle's initial ICs in February 2011. (ECF No. 79.) Although the initial ICs provided sufficient notice of Oracle's infringement theories and accused instrumentalities, Oracle explained them in a response letter brief to this Court. (ECF No. 82.) In the resulting Court-directed meet-and-confer on the ICs, Oracle invited Google to pose any questions it wished if it had difficulty understanding Oracle's ICs. (Peters Decl. ¶ 1.) Google *never* posed any queries that correspond to the complaints it makes in its current motion. (*Id.*) As the parties agreed, Oracle served second supplemental ICs in April 2011. Again Google *never* moved to compel Oracle to provide more detailed ICs or additional pinpoint citations to Android source code, nor even asked informally for more details such as the ones in the Report it now complains of. The Court-imposed deadline for bringing discovery motions has passed. Google has thus waived its right to challenge the adequacy of Oracle's ICs. *See Orion IP, LLC v. Staples, Inc.*, 407 F. Supp. 2d 815, 818 (E.D. Tex. 2006) ("A defendant cannot lay behind the log until late in the case and then claim it lacks notice as to the scope of the case or the infringement contentions."); *see also Bowoto v. Chevron Corp.*, No. C 99-02506 SI, 2006 U.S. Dist. LEXIS 65139, at *5 (N.D. Cal. Aug. 29, 2006) (denying motion to compel additional discovery because defendants had "no good reason for the month-long delay in bringing their motion to compel"). After failing to move to compel supplemental ICs in a timely manner, Google cannot now disguise its discontent with Oracle's ICs as a motion to strike portions of the Report. The ICs gave sufficient notice of Oracle's infringement theories and the Report properly provides additional detail to explain those theories. There are no new infringement theories in the Report.

B. Response to Critique B: The Report on Infringement of Claim 1 of the '702 Patent Is Supported by the ICs

The '702 patent is directed to systems and methods for pre-processing and packaging class files by removing duplicate information from them. (*See* Claim Construction Order at 8-11.) Claim 1 of the '702 patent recites, inter alia, "*removing* said duplicated elements from said plurality of class files . . . ." (Francis Decl. Ex. D, '702 patent col.51 ll.2-11 (emphasis added).)

Oracle's ICs disclose that Android's dx tool infringes Claim 1 when it transforms class files compiled by a Java language compiler into a .dex file. (Peters Decl. Ex. 2, ICs Ex. C.) Oracle's ICs for the '702 patent cite to Android source code, presentation slides entitled "Dalvik Virtual Machine Internals, Google I/O 2008," which were presented by Dan Bornstein, the author of the dx tool, and other documentary evidence. (*Id.*) To show that the dx tool performs the "removing" limitation, Oracle chose to use Bornstein's presentation slides, which were filled with admissions that the dx tool operates in a way that would infringe the '702 patent, rather than quote from Android source code files. Specifically, Bornstein's presentation slides on "Dex File Anatomy" illustrate how the dx tool removes duplicated constants from multiple class files to create a single constant pool table in a .dex file. (*Id.* at 13-16.) The presentation slides point out that the resulting .dex file has five types of constant pools with no duplicated constants: string_ids, type_ids, proto_ids, field_ids, and method_ids. (*Id.*) The ICs identify other sections of the .dex file (the "class_defs" section and other data) as the reduced class files. (*Id.*)

Even though the slides provide sufficient notice of Oracle's infringement theory, Oracle's ICs also cite to the TypeIdsSection.java file as meeting the "removing" limitation of Claim 1. (*Id.* at 17.) TypeIdsSection, as its name suggests, corresponds to the type_ids constant pool. As Prof. Mitchell explained in a recent deposition, "these different sections [of the constant pool table] are all coded similarly." (Peters Decl. Ex. 3 (9/6/2011 Mitchell Dep. 174:6-9).) Given the link between type_ids and the TypeIdsSection.java file, that citation constitutes sufficient notice to Google that other similarly named files are also relevant to this limitation—the StringIdsSection.java file for removing duplicate string_ids, the FieldIdsSection.java file for removing duplicate field_ids, and so on. All these sections are identified in the slides reproduced in the ICs.

The ICs provide further notice that the StringIdsSection.java and FieldIdsSection.java files are relevant. The ICs' disclosure for "determining plurality of duplicated elements" provides an antecedent basis for "removing *said* duplicated elements . . . ." (Francis Decl. Ex. D, '702 patent col.51 ll.2-11 (emphasis added).) The ICs quote from the DexFile.java file to show how the dx tool determines the duplicated constants. (Peters Decl. Ex. 2 at 4-9.) That file, in its toDex0 function, calls the "prepare" functions to create .dex file sections, including classDefs, methodIds, fieldIds, protoIds, typeIds, and stringIds. (*Id.* at 5-6.) In order to understand how the dx tool removes duplicated constants and obtains reduced class files as it prepares a .dex file, one would trace the called functions to the ClassDefsSection.java, StringsIdsSection.java, FieldIdsSection.java, and other similarly named files. Accordingly, the ICs provided ample notice to Google of which dx tool's functionality meets the "removing" limitation of Claim 1.

The Report provides more description of the .dex file anatomy by citing to the ClassDefsSection.java file, because the ClassDefsSection "generally includes the Dalvik byte code associated with the input class files after the identified duplicates have been removed." (Report ¶ 476; *see also id.* ¶¶ 478-479.) The Report also cites to the Section.java and

UniformItemSection.java files, simply because the ClassDefsSection.java file is a subclass of those two files and thus inherits some of their members and functions. (*Id.* ¶ 459.)

Moreover, the Report quotes from Android source code files to explain how the dx tool removes duplicated constants from class files in forming a single .dex file with a shared constant pool. The Report explains that the toDex0 function implemented in the DexFile.java file calls the "prepare" function for the sections of the class file. (*Id.* ¶¶ 458-59.) The Report explains the role of those called functions by tracing the calls to the StringIdsSection.java, FieldIdsSection.java, and other similarly named files, in which duplicated constants are removed. (Report ¶¶ 460, 461, 468.) The Report quotes from those files the source code for storing the determined constants in a TreeMap structure. (*Id.*) The Report explains that the TreeMap structure is designed to store a single copy of each constant, thereby necessarily removing any duplicate constants from each constant pool. (*Id.* ¶ 462.) Therefore, the Report further explains the Bornstein slides shown in the ICs by describing the accompanying Android source code.

The Report does not introduce a "new" infringement theory against the dx tool. Oracle has, all along, contended that the dx tool's functionality for removing duplicated constants from multiple class files to create a shared constant pool and reduced class files in a single .dex file infringes the '702 patent. Nothing about that theory has changed. Prof. Mitchell explained that in providing a narrative explanation of the infringement theories in his Report, he "ended up with . . . some different examples or following the call chain up and down a little bit to explain more fully. But it's the same basic . . . infringement story, a reason why the system infringes as shown in the chart." (Peters Decl. Ex. 3 (9/6/2011 Mitchell Dep. 177:18-178:2).) Therefore, the Report merely provides "additional information regarding the specific factual bases" for Oracle's ICs by tracing through the identified functions. *Sicurelli*, 2005 U.S. Dist. LEXIS 42233, at *33.

Again, it is too late for Google to challenge Oracle's ICs. In any event, Google's own conduct shows it understood Oracle's '702 infringement theory from the ICs perfectly well. Based on the ICs, Google was able to serve Invalidity Contentions and respond to Oracle's interrogatory seeking Google's non-infringement position. (Peters Decl. Ex. 4 at 16-17.) In its interrogatory response, Google contended that the dx tool does not meet the "removing" limitation because "there would be no intermediate step of removing duplicated elements from class files to obtain a plurality of reduced class files prior to forming a multi-class file." (*Id.* at 16.) Prof. Parr, Google's expert on non-infringement of the '702 patent, also opined in his expert report that "there is no 'remove' step performed by the Android dx tool to create reduced class files." (Peters Decl. Ex. 5 at 19.) Google would not have been able to assert this non-infringement position in its interrogatory response unless the ICs provided sufficient notice of Oracle's infringement theories to Google. Also, that Google did not change its non-infringement position after reviewing the Report indicates that the Report tracks the infringement theories disclosed in the ICs. The Report explains those infringement theories by providing only additional explanation and evidence; it does not add new theories.

Accordingly, the Report is fully supported by Oracle's ICs.

C. Response to Critique C: The Report on the Scope of the Accused Instrumentalities Is Supported by the ICs

Oracle's ICs identify the following "Accused Instrumentalities" as infringing:

> (i) "Android" or "the Android Platform"; (ii) Google devices running Android; and (iii) other mobile devices running Android. Representative examples of Google devices running Android include the Google Dev Phones, the Google Nexus One, and the Google Nexus S. Representative examples of other mobile devices running Android include HTC's EVO 4G, HTC's Droid Incredible, HTC's G2, Motorola's Droid, and Samsung's Captivate.

(Francis Decl. Ex. F at 2.) Google complains that the Report includes three "additional" Android devices: LG Optimus, HTC Droid Incredible 2, and Motorola Atrix. However, all of those devices fall under the category of "other mobile devices running Android" and thus are within the scope of the ICs. In fact, Google does not dispute that the three Android devices are "mobile devices running Android." The "other mobile devices" named in the foregoing section are specifically "representative examples," not an exhaustive list.

Oracle's contention is that all Android devices function in the same way with respect to the patented technology, and that contention was clearly made in the ICs. Google did not need to "guess" which Android devices are encompassed by Oracle's ICs—they all are. To be sure, because Google gives away Android (and hence Oracle's Java intellectual property) for free, Android has been widely adopted. Oracle could not reasonably examine every Android device for direct proof that the manufacturer has deployed Android in an infringing way. Instead, Oracle will rely on a mix of direct and circumstantial evidence to prove its allegations. The record is replete with evidence that device manufacturers install the infringing elements of the Android platform on their Android devices. Publicly available examples of such evidence include the following:

- Google's Compatibility Definition Document ("CDD"), which specifies the requirements for a device to be Android-compatible, forbid device manufacturers from modifying aspects of Android that are copied from the Java platform: "[D]evice implementers MUST NOT make any prohibited modifications . . . to these package namespaces: java.*, javax.*, sun.*, android.*, com.android.*." (Compatibility Definition Document for Android 2.2 at 8, *available at* http://source.android.com/compatibility/android-2.2-cdd.pdf.) All CDDs impose the same restrictions.

- Dan Morrill, who was formerly in charge of the CDDs, has stated: "Android is a chunk of software that you port to a device. For the most part, Android devices are running *the same code*." (Dan Morrill, *On Android Compatibility, available at*: http://android-developers.blogspot.com/2010/05/on-android-compatibility.html (emphasis added).)

Despite overwhelming evidence that device manufacturers install Google's infringing Android code as-is on their Android devices, Google continues to contend that it is not liable because its Android licensees might have made changes to the Android code before loading it onto their Android devices. To further investigate whether there was evidentiary support for Google's non-infringement argument, Prof. Mitchell double-checked the Android code installed on LG Optimus, HTC Droid Incredible 2, and Motorola Atrix devices. Exactly as one would expect, he confirmed that these devices made no changes to the infringing elements. The Report references those Android devices as additional evidence that Android devices include the infringing elements as Google intended. Those devices are not "new" to this case. Google has been on notice since Oracle served the initial ICs in December 2010 that all Android devices are accused of infringing the patents-in-suit.

Google has no evidence to rebut Oracle's contention that device manufacturers install the infringing elements of the Android platform on their Android devices. Google continues to represent that it does not know what source code is run on third-party Android devices. (*See* Mot. at 5-8; *see also* Peters Decl. Ex. 6 at 11 (Google's Responses to Oracle's Interrogatory No. 21) ("Google states that it has no direct, specific knowledge with regard to how third parties modify the accused Android source code and documentation.").) Google's position is disingenuous given its active role in ensuring that the devices are Android-compatible. Regardless, given its supposed lack of any relevant knowledge, Google cannot now challenge Oracle's position that the ICs apply to all Android devices.

In its motion, Google again argues that Oracle's ICs did not comply with the Patent Local Rules in identifying the scope of the Accused Instrumentalities. But this argument comes too late—Google has waived any claim that Oracle's ICs were inadequate and cannot contest them now. *See Fenner*, 2010 U.S. Dist. LEXIS 17536, at *11-13 ("If Defendants were unclear as to the scope of the contentions, it was their responsibility to work with Plaintiff, informally or through motion practice, to clarify the issue.").

Moreover, the ICs gave sufficient notice to Google as to the scope of Accused Instrumentalities. The ICs identify the Android elements that infringe each claim by referencing Android source code, presentations made by Android engineers, and other documentary evidence. From these, Google was on notice as to which components of Android devices it had to defend for some time. Moreover, because Google knew what Android devices were at issue, Google should not have had any difficulty in identifying the Accused Instrumentalities for the method claims and other claims that require physical components such as a computer processor or memory. As Oracle explained in its letter brief to this Court in February 2011, "[d]epending on whether the asserted claim is directed to a system, a method, or a computer software product, the accused instrumentalities are computers running Android or the Android development environment, the users of such computers (including application developers or end users), or the physical software medium itself (including Google's servers)." (ECF No. 82 at 2.) Oracle's ICs also identified the categories of direct infringers, including manufacturers of Android products, application developers, service providers, and end users.

Although Google characterizes Oracle's infringement theories disclosed in the ICs as "untenable," this is not the appropriate time or vehicle to argue the merits of the case. Infringement contentions only need "to provide reasonable notice to the defendant why the plaintiff believes it has a 'reasonable chance of proving infringement.'" *Shared Memory Graphics*, 2010 U.S. Dist. LEXIS 138868, at *10-11 (citation omitted). The ICs provided more than reasonable notice of the scope of Oracle's infringement theories. At trial, Oracle will present evidence to show how these Accused Instrumentalities, including all Android-branded phones, infringe the asserted patent claims according to the infringement theories disclosed in the ICs.

IV. CONCLUSION

The ICs gave sufficient notice to Google of the Accused Instrumentalities and Oracle's infringement theories. The Report is fully supported by the ICs. For the foregoing reasons, Oracle respectfully asks this Court to deny Google's motion to strike portions of the Mitchell Patent Report.

Dated: September 15, 2011

MICHAEL A. JACOBS
MARC DAVID PETERS
DANIEL P. MUINO
MORRISON & FOERSTER LLP

By: /s/ Marc David Peters

Attorneys for Plaintiff
ORACLE AMERICA, INC.