MORRISON & FOERSTER LLP
MICHAEL A. JACOBS (Bar No. 111664)
mjacobs@mofo.com
KENNETH A. KUWAYTI (Bar No. 145384)
kkuwayti@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'S RULE 50(A) MOTION AT THE CLOSE OF ALL EVIDENCE FOR PHASE II (PATENT PHASE)** Dept.: Courtroom 8, 19th Floor Judge: Honorable William H. Alsup

TABLE OF CONTENTS

Page

TABLE OF AUTHORITIES

Page(s)

CASES

STATUTES

I. INTRODUCTION

A reasonable jury may conclude that Google infringes Claims 11, 27, 29, 39, 40, and 41 of United States Patent No. RE38,104 ("the '104 patent") and Claims 1 and 20 of United States Patent No. 6,061,520 ("the '520 patent"). Google's admission that it sold Android phones, among other evidence, supports a finding that Google infringes by selling as well as by making and using regardless of Google's legal objection. Under existing law, the '104 patent is a valid broadening reissue patent. And there is sufficient evidence to support a finding that Google's infringement was willful.

The Court should deny Google's motion for judgment as a matter of law.

II. A REASONABLE JURY COULD FIND THAT GOOGLE INFRINGES THE ASSERTED CLAIMS OF THE '104 PATENT

The record evidence provides a legally sufficient basis for a reasonable jury to find that (1) Android's Resolve.c infringes Claims 11, 39, 40, and 41 of the '104 patent and (2) Android dexopt infringes Claims 27 and 29 of the '104 patent. There is substantial evidence that Dalvik bytecode instructions contain symbolic references that are resolved dynamically, rather than statically. (Because Google concedes that Resolve.c resolves symbolic references dynamically, the question of dynamic resolution of symbolic references applies only to infringement of Claims 27 and 29 by dexopt.) Oracle's infringement evidence proved literal infringement of the '104 patent claims, and there is no need to consider the doctrine of equivalents.

A. Dalvik bytecode instructions contain symbolic references

The dispute over the term "symbolic reference" boils down to whether a field index in a Dalvik bytecode instruction refers to a field in a data object by a name other than the numeric memory location. There is sufficient evidence for a reasonable jury to find that the answer is "yes."

1. A field index is a symbolic reference that is contained in a Dalvik bytecode instruction

The parties' trial evidence and argument focused on whether the field index in a Dalvik bytecode instruction is a symbolic reference. There is sufficient evidence for a reasonable jury to find that it is. Given the testimony that Dalvik and dexopt resolve type indices, method indices, and string indices in much the same way as field indices, a reasonable jury could conclude that these indices are symbolic references as well. See RT 3239:17-21 (McFadden), RT 3256:8-12 (McFadden), 3310:4-3311:1 (Mitchell).

The evidence proves that the Dalvik VM uses the resolver functions of Resolve.c to resolve a field index to a numeric memory location that is then used to obtain the value of a specific field. RT 3308:18-3309:24 (Mitchell), 3646:24-3647:25 (McFadden). Both parties' trial evidence and argument focused on a Dalvik bytecode instruction called the "IGET instruction." The IGET instruction (together with the IPUT instruction) "performs the identified object instance field operation with the identified field, loading or storing into the value register." TX 735 at 6. In other words, the IGET instruction "finds the instance of the object and retrieves the data from the specified field." RT 3221:2-7 (McFadden); *see also* RT 3968:10-15 (August). The IGET instruction contains the "IGET" opcode and three operands—vA, vB, and field@CCCC—where the third operand, field@CCCC, is the field index. TX 735 at 6; RT 3221:8-10 (McFadden).

Dr. Mitchell testified that the field index is a symbolic reference:

> Q. What's the relationship between the "y" in the source code program and that "field@01"?
>
> A. "Field@01" is really the Dalvik instruction's version of this field named "y" (indicating). So everywhere in the program that used "y," that the programmer wrote "y," the compiler and the dx tool turned that into "field 01." So this is really the symbolic name reference "field 01" in the bytecode program corresponding to the programmer's favorite name here "y," for the "x" "y" coordinate of a point.

RT 4023:8-16 (Mitchell).

> Q. Does the reference "01" -- so you regard "01" as a symbolic are [sic – or] numeric reference?
>
> A. Symbolic reference.
>
> Q. All right. Does the symbolic reference "01" need to be resolved to execute the IGET instruction?
>
> A. Right. What the programmer wrote is get the value over here to this object, add one to it, and do something else with it.

> So in order for that to happen, we have to find where that actual data is in the object using this symbolic reference. So the symbolic reference is resolved. There are a number of steps here. You've heard it three or four times, so let me save you the trouble of going through it.
>
> There is a process that looks up information here and then does some kind of a search. There was an animation of a linear search. Let's skip all that.
>
> But ultimately what's found as the resolution of this symbolic reference is this actual numerical reference, actual numeric location "48." So when that happens, that finishes, there is a little structure built in memory here that has a few parts. But the important part for us is it ends up with that memory location "48" in it. And that's what's used by the Dalvik Virtual Machine to then get this data value and operate on it and continue execution of the program.

RT 4024:23-4025:21 (Mitchell).

The field index is the reference that is resolved to determine a numeric reference to the corresponding instance field object. Google's Mr. McFadden testified that the indices are resolved by Dalvik:

> Q. The Dalvik VM stores pointers that result from resolving the indexes?
>
> A. Yes.
>
> Q. And the Dalvik VM then pulls them out of storage on subsequent Dalvik bytecode executions?
>
> A. Yes.

RT 3236:6-11 (McFadden). It is undisputed that the field index contained in the Dalvik bytecode instruction is not the numeric memory location of the specific field to which it refers. RT 3614:22-3615:16 (Bornstein); 3761:14-3762:6 (McFadden); 3970:20-3971:3 (August); 3533:21-25 (Mitchell).

The Android source code in evidence also shows that the field index contained in the IGET instruction is a symbolic reference. Android's Resolve.c has a dvmResolveInstField function for "[r]esolv[ing] an instance field." TX 47.6 at 8. The input to dvmResolveInstField is a field index and the output is a pointer to the instance field object. *See id.*; *see also* RT 3309:5-24 (Mitchell); RT 3638:8-17, 3646:24-3647:8 (McFadden). Mr. McFadden's source code comments explain that the Dalvik resolving functions convert an index contained in the instruction stream into a pointer:

> When a class, method, field, or string constant is referred to from Dalvik bytecode, **the reference takes the form of an integer index value**. This value indexes into an array of type_id_item, method_id_item, field_id_item, or string_id_item in the DEX file. The first three themselves contain (directly or indirectly) indexes to strings that **the resolver uses to convert the instruction stream index into a pointer to the appropriate object or struct**.

TX 46.14 at 1 (emphases added). Mr. McFadden confirmed that this was an accurate description of Dalvik. RT 3236:12-19 (McFadden). He also testified that if the instruction stream index were the numeric memory location, it would already be a pointer and there would be no reason to convert it to a pointer. RT 3234:22-3235:13 (McFadden).

A reasonable jury could find that a field index meets the Court's construction of "symbolic reference" exactly as the Court construed it. The undisputed fact that the field index and other indices are contained in Dalvik bytecode instructions proves that Dalvik bytecode instructions contain symbolic references as required by Claim 11 of the '104 patent. The same reasoning applies to the other asserted claims.

2. Google misrepresents Oracle's infringement argument and attempts to narrow the meaning of the term "data"

In its motion, Google misrepresents Oracle's infringement argument and accuses Oracle of redefining the Court's claim construction. Oracle's infringement argument is based on the Court's construction: that the Dalvik bytecode instructions contain field indices and other indices that refer to data to be obtained or used, and identify that data by a name other than the numeric memory location of the data. *See*, *e.g.*, RT 3303:2-3304:20 (Mitchell); TX 4015, 7:12-13. The evidence Oracle introduced at trial supports this argument.

Under the Court's claim construction, a symbolic reference "identifies data." In Android, actual field data in an instance object is "data," as Google's expert testified. RT 4002:5-16 (August). The actual field data in an instance object is "data" in the Court's claim construction, and the field index is the symbolic reference that identifies that data.

Google is the party misapplying the Court's claim construction. Its argument is that a reference is ***exclusively*** limited to what it ***directly*** references. But the Court's construction places no limitation on a symbolic reference other than it not being the numeric memory location of data to which it refers. A symbolic reference, by its nature, cannot be a direct reference to the data it refers to, because it must be resolved before the data can be obtained. There is no reason that a symbolic reference to actual field data in an instance object cannot ***also*** be the numeric memory location of other information that is useful in the resolution process that determines the numeric memory location of the actual field data. The infringement question is whether a field index identifies the actual field data to be obtained by a name other than the data's location—and it does.

In any case, it is clear that Google is looking at the wrong data under a proper reading of the claim. That actual data in an Android data object is the "data" in the Court's construction is supported by the '104 patent, because that is the data that is "obtained" or "thereafter used" in the asserted claims. RT 3311:23-3312:19 (Mitchell); RT 3759:12-3760:23 (McFadden); RT 3954:12-18 (August); 3958:1-3959:4 (August); TX 4015, 7:12-13, 12:16-17, 12:30-31, 12:44-45. Moreover, actual data in a data object is what is identified by an exemplary symbolic reference ("y") in the specification. TX 4015, 1:65-67 ("[A]n instruction that accesses or fetches y, such as the Load instruction 14' illustrated in FIG. 1, references the variable y by the symbolic name 'y'."); Fig. 1B (illustrating "data object" containing actual values 23 and 17). Google argues that a field index is a numeric reference to a location in the Field ID table (ECF No. 1166 at 2-3), but the '104 patent does not identify anything analogous to the information in the Field ID table the claim's as "data" that is "obtained."

Moreover, Oracle elicited testimony from which the jury could conclude that the contents of the Field ID table are not "data" at all:

> Q. So what this description of the overall file layout of a dex file shows is that the Field ID table is not stored in the Data area of a dex file; true, sir?
>
> A. It's not stored in the section that's labeled "Data."
>
> Q. Not stored in the section labeled "Data" by TX 736, Google's official definition of the dex file format; true, sir?
>
> A. True.

RT 3754:13-19 (McFadden). Accordingly, a reasonable jury could reject Google's argument as inconsistent with the Court's claim construction.

3. Dr. Mitchell's testimony provides substantial evidence of infringement

Dr. Mitchell's opinion, to which he testified at trial, is that the indices in Dalvik bytecode instruction are symbolic references. RT 4023:8-16, 4024:23-4025:21 (Mitchell). Google attacks that testimony by arguing that Dr. Mitchell had admitted in Paragraphs 269, 272, and 293 of his opening report that indices were numerical references. ECF No. 1166 at 3.

But Dr. Mitchell testified about sections of his opening expert report that support his opinion that the field indices in the IGET instructions are symbolic references. RT 4062:21-4066:19 (Mitchell) (testimony regarding Mitchell Opening Report paragraphs 250, 252, 283 and page 38 of Volume 2). He identified particular passages that disclose and support his opinion and establish that his opinion did not change in response to Dr. August's report. RT 4065:16-4066:19 (Mitchell). Although Google identified three instances of the same typographical error in Dr. Mitchell's opening report, he testified that he corrected the problem during his deposition when he first realized it, and the jury saw the deposition video in which he saw the error, consulted the code reproduced in the report, and corrected the error. RT 3489:21-3490:1 (Mitchell). He explained the errors again when cross-examined a second time. RT 3490:7-10, 3529:17-3530:5, 4038:10-4040:9 (Mitchell). Dr. Mitchell was quite clear that the mistakes in three paragraphs of his report were just that, and nothing more:

> Q. And you're now -- you've now testified that was just a mistake, right?
>
> A. Yes.

RT 4038:4-6 (Mitchell).

The jury is entitled to believe Dr. Mitchell's testimony. When deciding Google's motion for judgment as a matter of law, the Court must credit Dr. Mitchell's testimony. *City Solutions, Inc. v. Clear Channel Communications, Inc.*, 365 F.3d 835, 841 (9th Cir. 2004) ("In ruling on a motion for JMOL, 'the court must draw all reasonable inferences in favor of the nonmoving party, and it may not make credibility determinations or weigh the evidence.'") (quoting *Reeves v. Sanderson Plumbing Prods., Inc.*, 530 U.S. 133, 150 (2000)).

Google's argument that Dr. Mitchell "conjure[d] up a re-interpretation of the term 'symbolic reference'" is not supported by the evidence. ECF No. 1166 at 3. Dr. Mitchell's testimony was that because a field index refers to "data," in this case actual field data, a field index is a symbolic reference under the Court's construction and therefore the '104 patent is infringed. Because his testimony regarding the "data" that the field indices refer to is consistent with the Court's construction, the evidence elicited from Google's own witnesses, and the Android source code itself, there is no reason his testimony should be discredited.

Given the evidence indicating that field indices and other indices contained in the Dalvik bytecode instructions refer to data by a name other than the numeric memory location of the data, a reasonable juror could find that Dalvik bytecode instructions contain symbolic references, and that the Dalvik VM and dexopt infringe the asserted claims of the '104 patent on that basis.

B. Android dexopt resolves symbolic references dynamically rather than statically

Google devotes a single paragraph of its brief to trying to show that Android dexopt resolves symbolic references statically rather than dynamically. Google ignores the contradictory testimony of its own witnesses and expert and documents authored by its own engineers that refute its position. No reasonable jury could find in Google's favor on this issue (*see* ECF No. 1168 at 8-9), and certainly there is more than enough evidence in the record to support a finding against it.

Google relies on testimony from Android engineer Andy McFadden that dexopt "perform[s] a set of static optimizations." RT 3730:16-22. But Google ignores other McFadden testimony establishing that dexopt is dynamic. Mr. McFadden admitted that because the resolution process depends on the conditions actually existing on the handset, dexopt needs to rerun when those conditions change by way of a system update. RT 3769:13-17 (McFadden); *see also* RT 3255:20-25 (McFadden) (admitting need to run dexopt when performing system update because memory layout could change). Similarly, Google relies on testimony from its expert Dr. August that dexopt is static because it "runs and must run before the program executes." RT 3940:17-20 (August). But when asked whether "dexopt processes the dex files when the Dalvik Virtual machine is running," Dr. August conceded that it does so "[s]ometimes." RT 3988:14-

3989:23 (August). Even Mr. Bornstein admitted that dexopt processes dex files while the Dalvik Virtual Machine is running. RT 3580:21-23 (Bornstein).

The three documents Google relies on cannot support judgment either. TX 739 is a duplicate of TX 105, with a different exhibit number. Relying on a cropped quote, Google cites it for the proposition that "dexopt performs optimizations that 'can be inferred statically.'" ECF No. 1166 at 5. But the full sentence directly refutes Google's position. It actually states that, "*Some of these require information only available at runtime, others can be inferred statically* when certain assumptions are made." TX 739 at 5. Oracle, of course, does not have to show that every single dexopt optimization is performed dynamically to prove infringement. "Some" is enough. A reasonable jury could find in Oracle's favor—but not Google's—on the basis of this document alone. That is particularly true given that, when responding to a customer query about dexopt asking, "Why do we need to do this during runtime? Couldn't it be done in compile time?" Android engineers quoted verbatim from this very part of TX 739 stating that it shows "why some of these optimizations can only be performed at runtime." *Compare* TX 1094 *with* TX 739.

Similarly, TX 735 states only that some opcodes "are reasonable candidates for static linking," and TX 737 identifies some "statically linked" instruction formats. (TX 735, 737). But stating that something is a "reasonable candidate" for static linking does not establish that it does not operate dynamically. And labeling a process as "static linking" cannot change the fact that it operates dynamically in any event.

The jury could also accept the testimony of Oracle's expert. That testimony did not consist of mere "conclusory statements" as Google contends. ECF No. 1166 at 5. Dr. Mitchell testified at length, explaining in detail how dexopt operates dynamically and how claims 27 and 29 are infringed, expressly relying on and referring to the dexopt documentation, testimony from Mr. McFadden and Mr. Bornstein, and demonstrative exhibits. RT 3322:12-3333:13, 3538:9-3539:24, 4028:14-4029:8. By way of example only, Dr. Mitchell provided the following testimony:

> There are a number of reasons why this is a dynamic process. It requires runtime information from the runtime environment.
>
> As I – the class name, for example, refers to a location of a class loaded into memory. The position of a class in memory depends on how classes are loaded into the specific phone or device. So there is runtime information that's necessary in order to do this resolution.
>
> Q. And therefore?
>
> A. Therefore, it's a dynamic operation.
>
> It's also implemented in the Dalvik Virtual Machine, in the dexopt that uses the same code as the—or overlapping code with the bytecode interpreter.
>
> This isn't done, couldn't be done during dx tool on the developer's platform, as I believe Andy McFadden explained. So it's really a dynamic runtime environment optimization.

RT 3539:9-24 (Mitchell).

TX 105/739 was part of the dexopt documentation Dr. Mitchell relied upon. As Dr. Mitchell testified, this document shows "that it starts the virtual machine up, does this kind of dynamic boot process there, loads files from the class path because, as I explained, we need to know where they sit in memory in order to find numeric references, and then sets about verifying and optimizing and doing other things with that bytecode. So this is the sense in which it's dynamic and it's part of the runtime environment of the Android platform." RT 3332:1-14 (Mitchell); *see* TX 105/739 at 2.

There is more than enough evidence in the record for a reasonable jury to find in favor of Oracle. Google's JMOL should be denied.

C. Google infringes the asserted claims of the '104 patent literally

Oracle presented evidence of Google's literal infringement of the '104 patent and argued that Google literally infringes the '104 patent. There is sufficient evidence for a jury to find literal infringement—indeed, as shown in the previous sections, the evidence is compelling. *See also* ECF No. 1168. The jury was not instructed on the doctrine of equivalents, nor was it argued. The doctrine of equivalents provides no basis for granting judgment as a matter of law of non-infringement in this case.

III. A REASONABLE JURY COULD FIND THAT GOOGLE INFRINGES THE ASSERTED CLAIMS OF THE ’520 PATENT

Oracle proved that Android’s dx tool literally infringes the asserted claims of the ’520 patent by simulating execution of initialization bytecodes to determine their static initialization. As admitted by Google’s expert, the dx tool identifies the static initialization of bytecodes *without executing them*. That is the definition of “simulating execution” set forth in the claims. If there was any uncertainty about what the dx tool does, the file that commences the process is called “Simulator.java” and explicitly states that it “simulate[s] the effects of executing bytecode.” TX 46.16 at line 37.

Google has only two arguments to avoid liability for infringing the ’520 patent. Google argues that it should escape liability because its expert chooses to call the simulation process in the dx tool something else: “pattern matching.” But there is nothing in the claim language that excludes simulation involving pattern matching from the ambit of the claims. Google’s position is based on reading limitations into the claims regarding what “simulating execution” allegedly is and is not. But the claims say what “simulating execution” is—identifying the static initialization of bytecodes *without executing them*. TX 4011, 9:56-57. It is undisputed that the dx tool does that.

As Dr. Mitchell testified, the Android dx tool faces the same problem as described in the ’520 patent and solves it in the same way. When a Java program includes a static array (*i.e.*, a set of data items), the Java compiler will create a long list of bytecode instructions to initialize the array. TX 4011 (’520 patent), 1:57-2:58; RT 3335:10-19 (Mitchell). This set of bytecode instructions is larger than the array itself and takes up more memory space. *Id*. To reduce the number of instructions needed for initialization, the ’520 patent invention simulates execution of the instructions (*i.e.*, examines them without executing them) to determine the static initialization they perform, and replaces them with a shorter instruction to initialize the array. TX 4011 (’520 patent), 2:64-3:7, 3:66-4:17.

The Android dx tool does the same thing. It receives Java bytecode files compiled by a Java compiler. RT 3547:5-19 (Bornstein). Files containing static arrays will have long lists of bytecode instructions for initialization. RT 3334:16-3335:19 (Mitchell). The dx tool simulates execution of those instructions to identify the static initialization they perform, and replaces them with a single instruction to initiate the array. RT 3338:19-3339:19 (Mitchell). This process is commenced in a file called "Simulator.java" within the dx tool. TX 46.16 (Simulator.java); RT 3340:5-3341:16 (Mitchell). As the engineer comments clearly state, Simulator.java "simulate[s] the effects of executing bytecode" to figure out what the bytecode does. TX 46.16 at lines 37-43, 86-105. Simulator.java calls upon certain parsing methods (parseInstruction and parseNewarray) to assist with understanding the instructions. TX 46.16 at line 99; TX 46.17 at lines 211, 887; RT 3341:17-3344:7 (Mitchell). As a result of Simulator.java and the methods it invokes, the bytecode instructions are examined without being executed, their static initialization is determined, and a shorter "fast instruction" is generated to replace the long list of bytecode instructions. *Id*. This matches the "simulating execution" step of the asserted claims.

Google argues that the dx tool cannot be simulating execution of bytecodes because its process (1) involves "pattern-matching," and (2) does not manipulate a stack. But neither of those limitations exist in the asserted claims. Claims 1 and 20 do not say "simulating execution of bytecodes except where it involves pattern-matching," nor do they say "simulating execution by manipulating a stack."

While insisting that the dx tool does not simulate execution of bytecodes, Dr. Parr nonetheless admitted that it performs all of the constituent recited steps of simulation. He conceded that the dx tool works as follows (Dr. Parr's testimony in italics):

(1) The dx tool "identif[ies] the static initialization of the array"

"A. We agree that the dx tool identifies static initialization of an array."

(2) by examining the "byte codes of the clinit method against a memory"

Q. And you also agree that what we're talking about when we're doing the function in the dx tool is: ". . . directed to the bytecodes of the clinit method against a memory." True, sir?
A. Yes. The static initialization of an array is done with instructions in the clinit method.

(3) and does so "without executing the byte codes."

"Q. . . . you would agree with me that this is done without executing the bytecodes in the dx tool; true, sir?
A. In this context, yes."

RT 3822:17-3823:13. Accordingly, the dx tool is simulating execution of bytecodes as described in the asserted claims.

Google and Dr. Parr admit, moreover, that the dx tool's handling of initialization bytecodes begins in the Simulator.java file. ECF No. 1166 at 7; RT 3830:12-19, 3834:8-16, 3834:25-3835:5 (Parr). The engineer comments in that file explicitly describe it as a "[c]lass which knows how to simulate the effects of executing bytecode." TX 46.16 at lines 37. Google tries to evade this description by arguing that much of the process happens in another file via certain parsing methods (parseInstruction and parseNewarray) that are called by Simulator.java. ECF No. 1166 at 7-8. But the fact that Simulator.java calls on methods to parse instructions does not mean it is not still simulating execution of bytecodes. Dr. Parr admitted that parsing instructions *is part of the simulation process*:

Q. In order to simulate execution of a bytecode instruction, that code would make a call to another method called parseInstruction on line 119; true, sir?

A. True.

Q. So in order to simulate execution of a bytecode instruction using this code, you need to parse the instruction; true, sir?

A. I would use the term "decode," but, yes, I understand what the developer meant.

Q. So parsing the instruction here is part of simulation execution?

A. Parsing a single instruction, yes.

. . .

Q. And, again, to simulate execution of the bytecodes, the method makes a call to a method calls parseInstruction that's on Line 99; true, sir?

A. True.

Q. So, again, in order to simulate execution, this case of a block of bytecode instructions, you parse the instruction; true, sir?

A. It individually parses an instruction, yes.

RT 3828:5-16, 3829:8-17.

Google also cites to Dr. Parr's experiment in which he added a fake bytecode instruction to the stream of initialization instructions to break the pattern of instructions and cause the dx tool process to fail. ECF No. 1166 at 7. Google argues that "[a] process using simulated execution would simulate the execution of each instruction, including the useless instruction . . ." *Id*. This interpretation of "simulating execution" is not rooted in the claim language, which requires "simulating execution of the byte codes" but not necessarily separate simulation of each and every bytecode. In fact, the specification states that the preloader of the invention will *not* simulate execution of each bytecode, but rather only certain bytecodes that are "generally used to perform static initialization of an array." TX 4011, 5:17-51 ("Any byte codes other than those listed above are not recognized. . . . If a byte code is not recognized, the preloader considers it unsuitable for optimization"). Whether the dx tool simulates each instruction separately or blocks of them together (*i.e.*, pattern-matching), it still simulates execution of bytecodes as recited in the claims.

Finally, Google's argument improperly imports "stack manipulation" into the asserted claims. ECF No. 1166 at 7-8. As Dr. Parr admitted, the asserted claims make no mention of stack manipulation. TX 4011, 9:47-62, 12:3-7; RT 3794:20-23. Google had two opportunities to request construction of claim terms and did not seek to construe "simulating execution" to require "stack manipulation." Nor would there be any basis for limiting the claims to an exemplary embodiment. *Kara Tech. Inc. v. Stamps.com Inc.*, 582 F.3d 1341, 1345, 1348 (Fed. Cir. 2009) ("The patentee is entitled to the full scope of his claims, and we will not limit him to his preferred embodiment or import a limitation from the specification into the claims."). Indeed, dependent Claim 3 confirms that independent Claim 1 is not limited to stack manipulation, since Claim 3 includes "stack manipulation" as an express limitation. *Aspex Eyewear, Inc. v. Marchon Eyewear, Inc.*, 672 F.3d 1335, 1348 (Fed. Cir. 2012) (reversing summary judgment of noninfringement because "the presence of a dependent claim that adds a particular limitation gives rise to a presumption that the limitation in question is not present in the independent claim"). The absence of stack manipulation in the dx tool is irrelevant to infringement, as the asserted claims do not require that feature.

Contrary to Google's suggestion, Oracle's position is not "an implied theory of equivalent infringement." ECF No. 1166 at 9. Oracle has shown that the dx tool literally infringes Claims 1 and 20 by simulating execution of bytecodes (*i.e.*, examining the bytecodes *without executing them*) to identify their static initialization, as recited in the claims. Google is not entitled to judgment on the '520 patent; to the contrary, Oracle is entitled to judgment as a matter of law as set forth in its counterpart motion. ECF No. 1168 at 12-16.

IV. GOOGLE CANNOT ESCAPE LIABILITY FOR PATENT INFRINGEMENT BY CLAIMING IT GIVES AWAY ANDROID FOR FREE

Google next argues it cannot be held liable for patent infringement because its business model is to make Android available for "free" and then make its money indirectly through advertising. This is not the law, and the Court has already rejected this argument twice.

In the Parties' Joint Proposed Jury Instructions, Google proposed that the jury be instructed that for purposes of patent infringement a sale or offer to sell "does not include giving something away for free." ECF No. 539 at 98. The Court did not accept this language. Google made the same argument at the charging conference and the Court rejected it outright, adopting Oracle's proposed language that "Distributing or offering a product for free constitutes a use or sale:"

> That's got to be the law. At least on the facts of this case. It would be wrong for the jury to have the idea that Google, if it is infringing, is excused from infringing merely because of this free thing on the Internet. We all know that Google makes millions off of this, and this is – indirectly, maybe, but it's the way it works. So this should not be – this is a no-brainer. The law has to be this.

RT 3665:24-3667:7. *See also* ECF No. 1153 ¶ 14 (final jury instructions).

The Court's position is well supported by the case law. But the Court does not even have to reach this question because Google's admissions established direct infringement independent of this issue. Google admitted in an RFA response that it sold the Android Nexus One phone in the United States. RT 3105:4-5 ("Google admits that it sold Nexus One devices in the United States."). And Google Developer Programs engineer Daniel Morrill testified that Google developers use the Android SDK to develop applications for Android and then run these applications on Android phones to test them. TX 1126 (Morrill Dep. Tr. at 24:3-23, 36:5-37:23, 38:09-39:07). Google admitted this in RFA responses that were entered into the record as well. RT 3103:19-3104:23.[1]

In any event, many decisions have held that giving away an infringing product for free is a "use" or "sale" under section 271(a), particularly where it is associated with anticipated economic gain. In *Thorn EMI N. Am., Inc. v. Micron Tech., Inc.*, for example, the court found the plaintiff infringed by giving away free samples of the infringing product:

> Viewing ownership of a patent as ownership of the right to exclude, each delivery of a free sample consequently impinges on the patentee's right to exclude others from interfering with the patentee's monopoly on the patented product. *A patentee's monopoly would be worthless, for example, if, in an attempt to solicit business, a person delivered free samples to all of the patentee's potential customers.* Mere solicitation, in contrast, does not tangibly affect the patentee's monopoly. In sum, because delivery of free samples of allegedly infringing samples tangibly impinges on the patentee's monopoly rights in a way that ordinary solicitation does not, such delivery must be regarded as "use" of an allegedly infringing product for the purposes of § 271(a).

821 F. Supp. 272, 275 (D. Del. 1993) (emphasis added).

The scenario *Thorn* foresaw is precisely what is happening here. Google has gained a dominant market position by making its infringing Android operating system available for free. As Oracle CFO Safra Catz noted, "It's pretty hard to compete with free." RT 2322:11-12 (Catz). *See also Applied Biosystems, Inc. v. Cruachem, Ltd.*, 772 F. Supp. 1458, 1466 (D. Del. 1991) (mailing samples of an infringing product may constitute direct patent infringement); *Patent Tube Corp. v. Bristol-Myers Co.*, 25 F. Supp. 776, 777 (S.D.N.Y. 1938) ("The distribution of the patented device only for advertising purposes and without actual monetary compensation therefore, in my opinion, creates no exception to the general rule that use of the patented device is forbidden. To hold so would be permitting the doing of something indirectly which is forbidden to be done directly.").

[1] To the extent Google is implying that this issue may be relevant to indirect infringement, it is foreclosed from doing so. The parties stipulated that "if the jury finds that Google directly infringes any claims of either or both of the '104 and '520 patents, the verdict shall be treated as a verdict in favor of Oracle with respect to both its direct and indirect infringement claims relating to such directly infringed patent claims." ECF No. 1139 ¶ 1.

As the Federal Circuit stated in *Medical Solutions, Inc. v. C. Change Surgical LLC*, a case cited by Google, "[t]he inquiry as to what constitutes a 'use' of a patented item is highly case-specific." 541 F.3d 1136, 1141 (Fed. Cir. 2008) (citation omitted). It is not surprising, then, that all the cases Google relies upon have dramatically different facts from those at issue here, involving jurisdictional questions and isolated instances of a demonstration or offer to donate a product. The holding in *Medical Solutions*, for example, was that the demonstration of a device at a trade show "which appears to fall short of practicing all of the elements of any one claim" was not a sufficient basis for jurisdiction. *Id. See also Advanced Semiconductor Materials Am., Inc. v. Applied Materials, Inc.*, 1995 WL 419747 (N.D. Cal. July 10, 1995) at *6 (finding triable issue of material fact as to whether product demonstration constituted infringement); *HollyAnne Corp. v. TFT, Inc.*, 199 F.3d 1304, 1309 (Fed Cir. 1999) (plaintiff "conceded it was attempting to base personal jurisdiction on one offer to donate and nothing more").

In contrast, by distributing Android for "free," Google is generating hundreds of millions of dollars in revenue annually. *See* TX 1061 at 7. One internal Google document refers to Android as a $10 billion opportunity. TX 431. As Google Chairman Eric Schmidt testified, "the primary reason to have something like Android is that people will do more searches, and then we'll get more money as a result." RT 1458:9-16.[2] In an industry where a free distribution business model like Google's is hardly unique, Google's argument would gut patent law.

The Court should deny this part of Google's motion as well.

V. THE '104 PATENT IS A VALID BROADENING REISSUE

As Google acknowledges, under existing law Google has no defense under 35 U.S.C. § 251. Two months ago, the Federal Circuit held that "after a broadening reissue application has been filed within the two year statutory period, an applicant is 'not barred from making further broadening changes' after the two year period 'in the course of [the] prosecution of the reissue

[2] In addition, Android is not given away for free—it is licensed in exchange for consideration. Under the license selected by Google (the Apache License, Version 2.0), licensees agree to certain conditions for further distribution and to waive Google's liability arising out of Android, among other obligations. *See, e.g.*, TX 46.21 (identifying Android source code as copyrighted and referring readers to license link at http://www.apache.org/licenses/LICENSE-2.0 for "the specific language governing permissions and limitations under the License.").

application.'" *In re Staats*, 671 F.3d 1350, 1355 (Fed. Cir. 2012) (quoting *In re Graff*, 111 F.3d 874, 877 (Fed. Cir. 1997)). In *Staats*, a patentee first filed its application in 1996, was granted a patent in 1999, and subsequently filed three broadening reissue applications, adding broadened claims as late as 2007, eight years after the original patent issued. The Federal Circuit reversed the rejection of the third set of claims by the Board of Patent Appeals and Interferences.

In this case, Sun filed its original application on December 22, 1992, and U.S. Pat. No. 5,367,685 issued on November 22, 1994. TX 4015 at (64). Sun filed a reissue application on November 21, 1996, which matured into U.S. Pat No. RE36,204 on April 27, 1999. TX 4013 at (22), (45). While the '204 reissue application was pending, Sun timely filed a reissue continuation application on March 3, 1999 in compliance with 35 U.S.C. § 120.[3] *See* Certified File History of '104 Patent, TX 4018 at 4. The 1999 application matured into the '104 patent. TX 4015 at (22), (45). Sun filed the '104 reissue application less than five years after the '685 patent issued, and the claims added in the '104 application directly related to the errors identified in the oath accompanying the '204 application. *See* Reissue Declaration, TX 4018 at 29-30; Supplemental Reissue Declaration, TX 4018 at 1133. The claims at issue in *Staats*, by contrast, were not even related to the claims in the original reissue application. 671 F.3d at 1353. Under Federal Circuit precedent and Sections 120 and 251 of the Patent Act, the '104 patent is a valid patent resulting from a proper broadening reissue application.

VI. A REASONABLE JURY COULD FIND THAT GOOGLE'S INFRINGEMENT WAS WILLFUL

The evidence at trial established, clearly and convincingly, that Google's infringement of the '104 and '502 patents was willful, and a reasonable jury could so find.

[3] "An application for patent for an invention disclosed in the manner provided by section 112(a) . . . which is filed by an inventor or inventors named in the previously filed application *shall have the same effect*, as to such invention, *as though filed on the date of the prior application*, if filed before the patenting or abandonment of or termination of proceedings on the first application or on an application similarly entitled to the benefit of the filing date of the first application and if it contains or is amended to contain a specific reference to the earlier filed application." 35 U.S.C. § 120 (emphasis added). Thus a reissue continuation application is treated as if it were filed on the date of its parent reissue application.

Under Federal Circuit law, willfulness is established if Oracle shows that (1) Google acted "despite an objectively high likelihood that its actions constituted infringement of a valid patent" and (2) that the objectively defined risk was "either known or so obvious that it should have been known" to Google. *See In re Seagate Tech., LLC*, 497 F.3d 1360, 1371 (Fed. Cir. 2007) (citing *Safeco Ins. Co. of Am. v. Burr*, 551 U.S. 47, 68-69 (2007)). With the exception of a single sentence that baldly—and incorrectly—asserts that Oracle introduced no evidence sufficient to meet the second prong (ECF No. 1166 at 17), Google devotes its entire argument to the first prong.

Willfulness is determined based on the "totality of the circumstances," not individual facts in isolation. *See Knorr-Bremse Systeme Fuer Nutzfahrzeuge GmbH v. Dana Corp.*, 383 F.3d 1337, 1342 (Fed. Cir. 2004) (en banc); *see also id.* at 1343 ("'[W]illfulness' in infringement, as in life, is not an all-or-nothing trait, but one of degree." (quoting *Rite-Hite Corp. v. Kelley Co.*, 819 F.2d 1120, 1125-26 (Fed. Cir. 1987))).

Ignoring this settled principle, Google begins with the mistaken argument that it cannot be liable for willful infringement unless and until it received actual notice from Oracle of the specific patents that Google infringed. ECF No. 1166 at 13 (describing "actual notice" as a "prerequisite" to willfulness). That is not the law, and federal courts have repeatedly rejected the argument Google makes now. For example, in *Black & Decker Inc. v. Robert Bosch Tool Corp.*, the Court rejected as "without merit" defendant's argument that willfulness required knowledge of specific patents. No. 04 C 7955, 2006 WL 3783006, at *3 (N.D. Ill. Dec. 22, 2006). "Indeed, the Federal Circuit instructs that it is the infringer's knowledge of ***patent rights*** that is relevant." *Id.* (emphasis in original) (citing *nCube Corp. v. Seachange Int'l, Inc.*, 436 F.3d 1317, 1324 (Fed. Cir. 2006) and *Imonex Servs., Inc. v. W.H. Munzprufer Dietmar Trenner GMBH*, 408 F.3d 1374, 1377-78 (Fed. Cir. 2005)). *See also Power Integrations, Inc. v. Fairchild Semiconductor Int'l, Inc*., 725 F. Supp. 2d 474, 479-80 (D. Del. 2010) ("If Fairchild was not busy studying and copying Power Integrations' technology as discussed above, it was essentially ignoring it, which in the Court's view, also rises to the level of objectively reckless behavior."); *PalTalk Holdings, Inc. v. Microsoft Corp.*, No. 2:06-CV-367 (DF), 2009 U.S. Dist. LEXIS 131087, at *6-7 (E.D.

Tex. Feb. 2, 2009) (holding that constructive knowledge is sufficient to satisfy objective prong of *Seagate* test; denying defendant's motion for summary judgment on willfulness because defendant "was aware both of the innovative characteristics of [patentee's] technology and [its] development of a patent portfolio").

The two cases that Google cites do not hold otherwise. *Seagate* says nothing whatsoever about requiring knowledge of specific patents, at the page cited by Google or elsewhere. *See Seagate*, 497 F.3d at 1371. To the contrary, *Seagate* repeats the rule that willfulness requires only that the defendant act "despite an objectively high likelihood that its actions constituted infringement of a valid patent." *Id.* In Google's only other case, *Gustafson*, the Federal Circuit reversed a finding of willfulness where the only finding from the district court related to knowledge was that the defendant first learned of the patent when it was served with the complaint. *See Gustafson, Inc. v. Intersystems Indus. Prods., Inc.*, 897 F.2d 508, 511 (Fed. Cir. 1990).

The evidence of Google's recklessness is overwhelming. Google decided at the very beginning of Android's development that it would rely on Java. TX 7 (Rubin email stating that "Android is building a Java OS"); TX 23 (Swetland email stating that "we are building a java based system: that decision is final"). More specifically, Google planned from the very beginning to build a Java virtual machine. TX 4 (Android was working on a JVM even before Google acquired it); TX 1 ("current scenario" in July 2005 included JVM). Although Google tried to argue at trial that "Dalvik" is not a JVM, the evidence at trial established that in many significant ways, it is. Because Android uses the Java language, Dalvik follows the Java virtual machine specification with respect to class loading, linking, and initialization. RT 3223:13-3224:5 (McFadden). Dalvik also requires developers to use the Java compiler, further compelling similarities between Dalvik and Sun's JVMs. RT 3547:5-10 (Bornstein). Internally, as late as August 2009, Andy Rubin and other Android team members still used the terms "Dalvik" and "Java Virtual Machine" "interchangeably to describe [their] work at the time internally," and, in fact, Rubin testified that "most of the time I used the term JVM." TX 219; RT 3179:12-22 (Rubin).

Sun, of course, invented Java, and has thousands of Java patents. RT 1471:24-1473:4 (Schmidt); RT 549:12-14 (Screven). Those facts were well known to Google and the world. Eric Schmidt, Google's CEO during most of the events in question and its chairman to this day, was the Chief Technology Officer of Sun until 1997. RT 1471:8-1472:1; 1479:2-3 (Schmidt). Not only did Google hold an Executive Committee seat in the Java Community Process (RT 303:1-6 (Ellison))— a seat it still holds—Google in fact designated Android engineers such as Bob Lee and Josh Bloch to represent it in the JCP. RT 1181:15-20 (Lee); RT 827:18-21 (Bloch). By 2005, Google had hired numerous Sun engineers with extensive experience—and with patents attributed to them as inventors—related specifically to Java virtual machines. TX 5; RT 2993:1-24, RT 2997:14-2999:12 (Lindholm). Tim Lindholm's book, *The Java Virtual Machine Specification*, contains a notice on the copyright page itself that the specification it describes is covered by patents, and an entire chapter of the book is devoted to describing an implementation of the predecessor to the '104 patent. TX 25, at 6, 401; TX 4015. *See Depomed, Inc. v. Ivax Corp.*, 532 F. Supp. 2d 1170, 1186 (N.D. Cal. 2007) (denying summary judgment on willfulness where "there is evidence that the '475 patent and an agreement to license the patent to a third party were well publicized"). Mr. Rubin testified that he had had discussions with Sun about patents relating to the virtual machine. RT 3205:1-3 (Rubin). These facts provide objective evidence that there was a high likelihood that Google's Java-based Dalvik virtual machine would infringe one or more Sun Java patents.

In addition to Google's knowledge of Java technology, licensing, and business, the long, continuing pattern of licensing negotiations between the two companies also supports a finding of willfulness. *See Walker Digital, LLC v. Facebook, Inc.*, CIV. 11-313-SLR, 2012 WL 1129370, at *7 (D. Del. Apr. 4, 2012) (finding that allegations "Amazon had pre-suit knowledge of the patents-in-suit due to Amazon's interactions with Walker Digital's representatives, which occurred before the original complaint was filed" precluded motion to dismiss willfulness claim); *see also* TX 565 at p. 2 (8/2/07: Vineet Gupta: "Andy cannot say he was not aware of the licensing requirements - as he had to go thru this at Danger - and we discussed this during Project Android Phase, and then during the Sun/Google collaboration attempt as well.").

Google focuses on just one piece of evidence, and tries to argue that Mr. Lindholm had forgotten the contents of the book he wrote, or had not actually read the patent that he wrote about in so much detail. Far less exposure to the patent and patented technology has been held to establish knowledge—even when the defendant employee with knowledge of the patent disclaimed knowledge of the infringing work. *See Stryker Corp. v. Intermedics Orthopedics, Inc.*, 96 F.3d 1409, 1415-16 (Fed. Cir. 1996). But even if the jury credited Mr. Lindholm's testimony, Google's argument misses the point: Mr. Lindholm, along with Eric Schmidt, Andy Rubin, Bob Lee, Josh Bloch, and many other Google employees, was personally and directly aware that Sun, and later Oracle, had invested years of research into developing technology for Java virtual machines, and that technology was covered by patents, including one of the specific patents that Google actually infringed.[4]

Accordingly, Mr. Rubin had no choice to admit that, with regard to Sun,

> Look, like I said before, I assume they're running a business, they're inventing intellectual property, they're protecting it through the patent system. Through GPL, I didn't know what they were, but I knew that it was dangerous to use the stuff without knowing exactly what it was.

TX 1128 (Rubin Dep.) at p. 14, 16:04-16.

Despite the fact that Android was building a "Java Virtual Machine," despite knowing that Java technology was protected by Sun patents, despite having numerous employees with the technical ability and direct experience to assess whether Google infringed Sun's Java patents, and despite having the actual inventor of the '520 patent, Frank Yellin, on its payroll, Google looked away. Such conduct—proceeding with a "dangerous" course of action without availing oneself

[4] There is no dispute that Google did have actual knowledge of the patents it infringed at least as of July 20, 2010. RT 3100:10-20 (stipulation as to notice). Google argues that the period from July 20 to August 13, 2010, when the lawsuit was filed, is too short to support a finding of willful infringement. There is no such rule of law. *See Walker Digital, LLC v. Facebook, Inc.*, CIV. 11-313-SLR, 2012 WL 1129370 (D. Del. Apr. 4, 2012) (refusing to dismiss claim for willful infringement based on pre-suit knowledge where complaint alleged that plaintiff and defendant discussed allegations of infringement ***just one month*** before the complaint was filed); *Power Lift, Inc. v. Lang Tools, Inc.*, 774 F.2d 478, 481-82 (Fed. Cir. 1985) (finding infringement willful based on post-notice conduct where there were only ***9 days*** between notice and filing of suit).

of means to prevent harm, when those means are directly at hand—is the very definition of recklessness.

Significantly, Google does not spend a single word of the willfulness section of its brief trying to argue that it had credible defenses to infringement. Seeing the weakness of its position, Google tries to go outside the trial record to argue that it asserted, ***in its Answer***, what it calls "sound legal defenses to those allegations." ECF No. 1166 at 14. Even if that were evidence—and it is not—the mere assertion of affirmative defenses is of no help to Google in avoiding a willfulness finding. *See i4i Ltd. P'ship v. Microsoft Corp.*, 670 F. Supp. 2d 568, 581-82 (E.D. Tex. 2009) ("the number of creative defenses that Microsoft is able to muster in an infringement action after years of litigation and substantial discovery is irrelevant to the objective prong of the *Seagate* analysis"), *aff'd*, 589 F.3d 1246 (Fed. Cir. 2009), *and aff'd*, 131 S. Ct. 2238 (2011). *See also id.* at 595 ("The fact that Microsoft presented several defenses at trial, including noninfringement and invalidity, does not mean the jury's willfulness finding lacks a sufficient evidentiary basis."). In any event, Google insisted that the evidence of willfulness be limited to pre-complaint conduct, which the Court accepted. RT 3902:15-3903:17; ECF No. 1166 at 13-14 n.2. It cannot now turn around and rely on its post-complaint conduct to avoid willfulness.

All that remains of Google's willfulness argument is the contention that it could not have acted willfully because Jonathan Schwartz said kind things about Android in a blog post before any of Android's technology had been released, Larry Ellison said he was "flattered" by Android before Oracle had even acquired Sun, and Eric Schmidt was of the "opinion" (a word that Google excises from the quoted testimony) that "Sun management was comfortable that we had done -- that what we had done was free and clear of any intellectual property of Sun's." ECF No. 1166 at 15-16. This evidence of Google's supposed state of mind would be relevant, if at all, to the second, subjective prong of *Seagate*, not the objective prong. In any event, the jury plainly and properly rejected that very argument in Phase 1 when it answered Question 4.B of the verdict form with a unanimous "No."

Finally, Google asserts, in a throwaway line, that Oracle "failed to introduce any evidence sufficient to meet the requirements of the second, subjective prong." ECF No. 1166 at 17. A bare assertion such as this would provide no basis to award JMOL, but the statement is demonstrably false in any event. The "objectively defined risk" of infringement described above not only was known or should have been known to Google, the existence of that risk was established entirely through Google documents and the testimony of Google employees.

The risk of infringement came as no surprise to Google. Google knew it was building a Java-based solution that included a Java virtual machine. Google knew that it needed a license to use Sun Java technology, and had no such license. TX 1 at p. 9 (7/26/2005: "Must take license from Sun"); TX 3 at p.3 (7/29/2005: "Google needs a TCK license"); TX 7 at p.1 (10/11/2005: "My proposal is that we take a license") TX 12 (12/20/2005: "Either a) we'll partner with Sun as contemplated in our recent discussions or b) we'll take a license"); TX 17 (2/10/2006: "critical license"). Google knew that Sun was willing to assert its patents against Google. (TX 230, RT 3193:3-5 (Rubin) ("Sun had threatened Google before, in different areas of its business, on patents."). Google worried about the risk of "Java lawsuits" over patents and copyrights. TX 326, TX 406, RT 1559:20-1560:12 (Schmidt). Google decided that if it could not reach an agreement with Sun, it would "Do Java anyway and defend our decision, perhaps making enemies along the way." TX 7 at p. 2. Having done just that, Google knew that Android was in "dangerous territory" in dealing with Sun. TX 1029. On March 24, 2008, Mr. Rubin urged his team not to demonstrate Android to any "Sun employees or lawyers." TX 29. And on August 6, 2010, Mr. Lindholm needed little time to conclude, yet again, that Google needed to take a Java license from Oracle for Android. TX 10. A reasonable jury could decide that Google's conduct was willful.

VII. CONCLUSION

For the foregoing reasons, the Court should deny Google's motion for judgment as a matter of law.

Dated: May 17, 2012

MORRISON & FOERSTER LLP

By: /s/ *Michael A. Jacobs*
Michael A. Jacobs
Attorneys for Plaintiff
ORACLE AMERICA, INC.