Oracle v. Google

What Is A Patent?

- A copyright protects creative expression
- A patent protects an invention

17.

The copyright confers ownership over the particular expression of ideas in a work but it never confers ownership over ideas themselves. For example, if a book describes a strategy for playing a card game, the copyright prevents anyone (but

the b

strat

own

plag

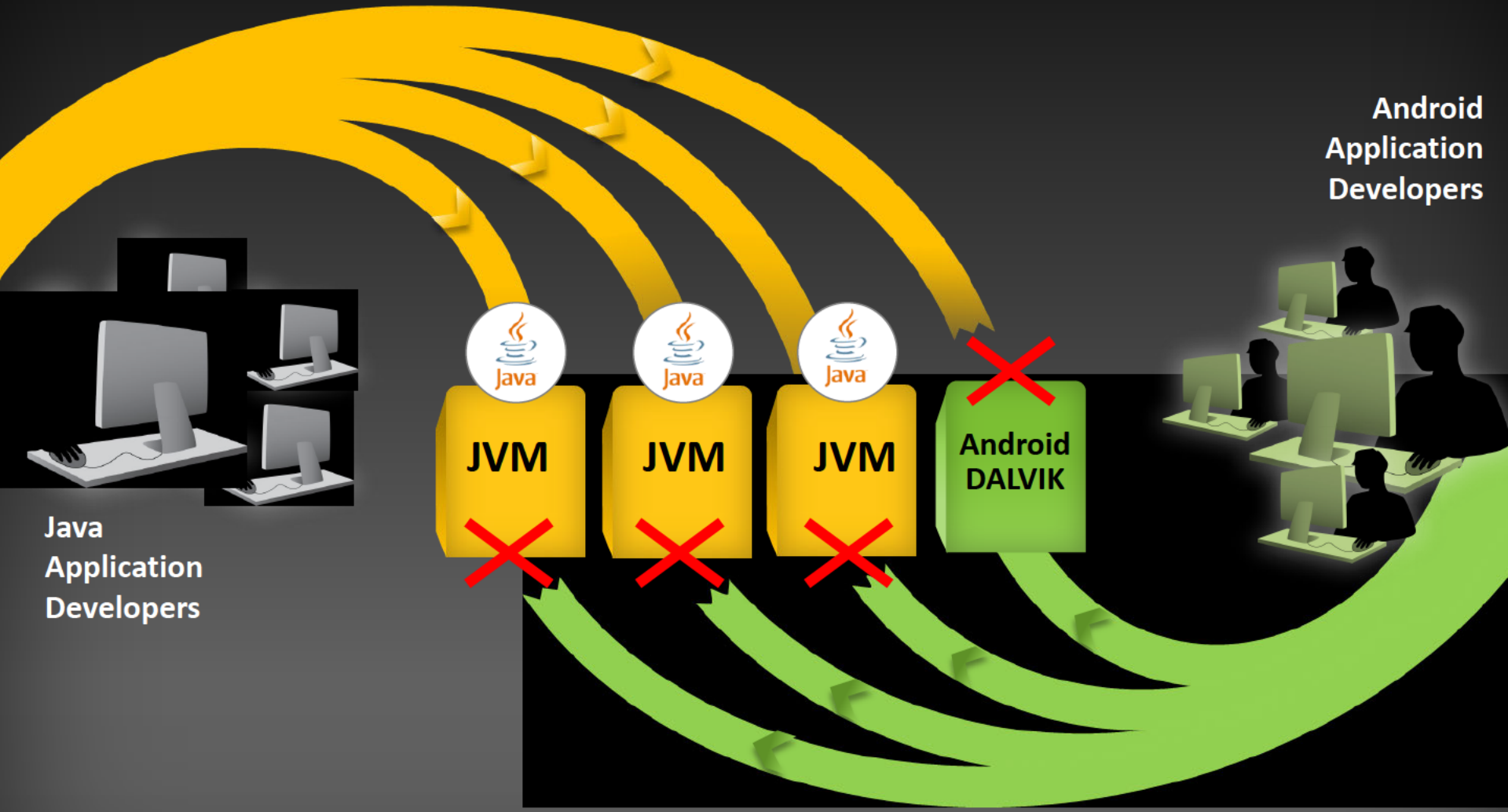
the p

neve

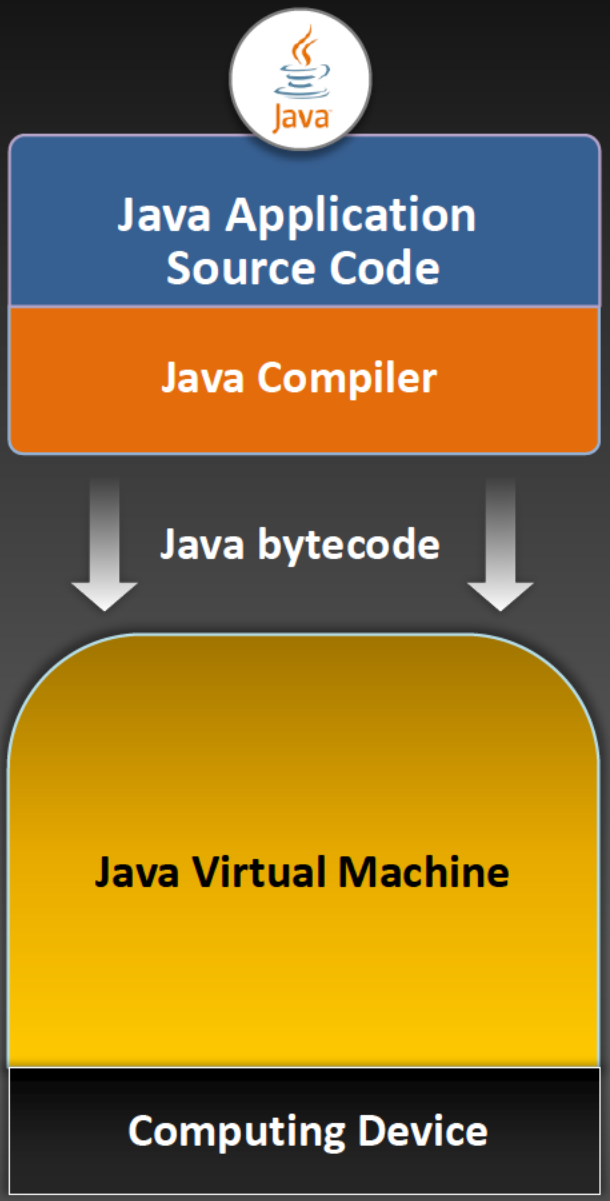
principle, or discovery. Possibly such things can be claimed under the patent system or by trade secret laws but they may not be claimed by copyright. For purposes of your deliberations, I instruct that the copyrights in question do cover the structure, sequence and organization of the compilable code.

Another statutory limitation on the scope of a copyright is that copyright never protects any procedure, process, system, method of operation, concept, principle, or discovery. Possibly such things can be claimed under the patent system or by trade secret laws but they may not be claimed by copyright. For purposes of your deliberations, I instruct that the copyrights in question do cover the structure, sequence and organization of the compilable code.

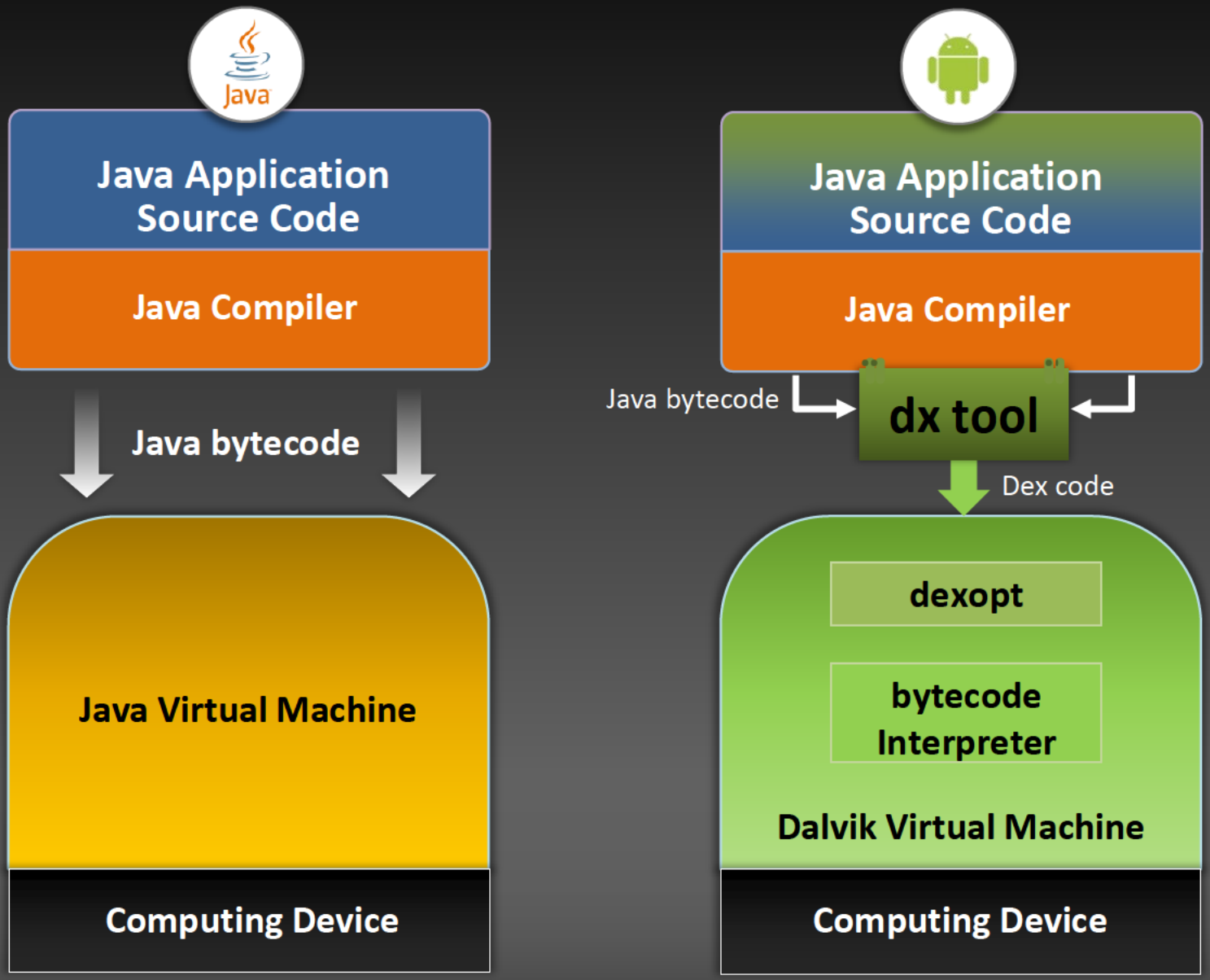
Android Uses “Dalvik” Virtual Machine



Java Platform Components



Java Platform vs. Android Platform Components



Java Source Code Compiled Into Bytecode

Java Application Source Code

```
public class HelloWorld {

  public static void main(String[] args) {
      System.out.println("Hello
      World!");
  }
}
```

Java Bytecode

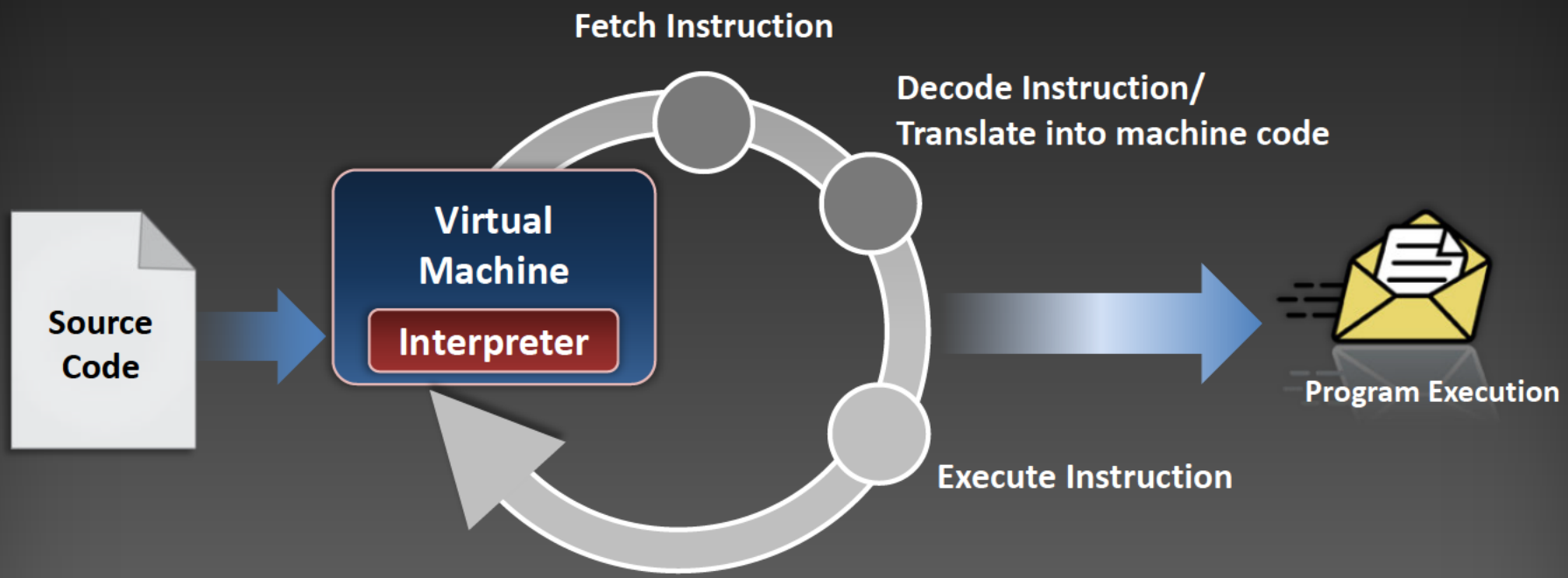
dx tool

Android dex code

```
Compiled from "HelloWorld.java"
public class HelloWorld extends java.lang.Object
  SourceFile: "HelloWorld.java"
  minor version: 0
  major version: 49
  Constant pool:
const #1 = Method                       #6.#15;          //
    java/lang/Object."<init>":()V
const #2 = Field     #16.#17;           //
    java/lang/System.out:Ljava/io/PrintStream;
const #3 = String    #18;               // Hello World!
const #4 = Method                       #19.#20;         //
    java/io/PrintStream.println:(Ljava/lang/String;)V
const #5 = class     #21;               // HelloWorld
const #6 = class     #22;               // java/lang/Object
const #7 = Asciz     <init>;
const #8 = Asciz     ()V;
const #9 = Asciz     Code;
const #10 = Asciz    LineNumberTable;
const #11 = Asciz    main;
const #12 = Asciz    ([Ljava/lang/String;)V;
const #13 = Asciz    SourceFile;
const #14 = Asciz    HelloWorld.java;
const #15 = NameAndType                 #7:#8;// "<init>":()V
const #16 = class    #23;               // java/lang/System
const #17 = NameAndType                 #24:#25;//
    out:Ljava/io/PrintStream;
const #18 = Asciz    Hello World!;
const #19 = class    #26;               // java/io/PrintStream
const #20 = NameAndType                 #27:#28;//
    println:(Ljava/lang/String;)V
const #21 = Asciz    HelloWorld;
const #22 = Asciz    java/lang/Object;
const #23 = Asciz    java/lang/System;
const #24 = Asciz    out;
```

```
000000: 6465 780a 3033|magic: "dex\n035\0"
000006: 3500         |
000008: e561 54a3    |checksum
00000c: f7a0 86cd e5b6|signature
000012: 4bc8 9fbe d011|
000018: 968a a9a6 7305|
00001e: 430d         |
000020: e802 0000    |file_size:      000002e8
000024: 7000 0000    |header_size:    00000070
000028: 7856 3412    |endian_tag:     12345678
00002c: 0000 0000    |link_size:      0
000030: 0000 0000    |link_off:       0
000034: 4802 0000    |map_off:        00000248
000038: 0e00 0000    |string_ids_size: 0000000e
00003c: 7000 0000    |string_ids_off:  00000070
000040: 0700 0000    |type_ids_size:   00000007
000044: a800 0000    |type_ids_off:    000000a8
000048: 0300 0000    |proto_ids_size:  00000003
00004c: c400 0000    |proto_ids_off:   000000c4
000050: 0100 0000    |field_ids_size:  00000001
000054: e800 0000    |field_ids_off:   000000e8
000058: 0400 0000    |method_ids_size: 00000004
00005c: f000 0000    |method_ids_off:  000000f0
000060: 0100 0000    |class_defs_size: 00000001
000064: 1001 0000    |class_defs_off:  00000110
000068: b801 0000    |data_size:      000001b8
00006c: 3001 0000    |data_off:       00000130
                     |
                     |string_ids:
                     |[0] "<init>"
000070: 7601 0000    |  string_data_off: 00000176
                     |[1] "Hello World!"
000074: 7e01 0000    |  string_data_off: 0000017e
                     |[2] "HelloWorld.java"
000078: 8c01 0000    |  string_data_off: 0000018c
```

Virtual Machines Slow Things Down



Google Android Developers Faced Performance And Memory Challenges

From: Brian Swetland
To: Andy McFadden and others
Sent: August 16, 2006

From: Brian Swetland Sent:8/16/2006 4:40 AM
To: [-] fadden@google.com; fcus@google.com; cjd@google.com
Cc: [-]
Bcc: [-]
Subject: feedback welcome.

The following is a document I'm calling DESIGN_MANIFESTO at the moment.
Does this stuff make sense? Am I smoking the crack?

Brian

- architecture: priority one is the device
- we are shipping the device, not the simulator
- if the device is not fast and stable we FAIL
- the emulator is the answer for desktop work
- yes, it is slow: we must make it faster
(the end users will never judge us by how fast the simulator was)
- performance is a problem NOW not LATER

"if the device is not fast and stable we FAIL"

- "it is complicated because it is powerful" is a lousy answer to "why is it so hard to do X"

- build on top of standard linux kernel services
- avoids making the kernel bigger
(can't remove core services: let's use em!)
- relies on well tested existing services
- in particular:
- unix domain sockets (dgram and stream)
- make use of the private linux-domain namespace
- shm passing using fd-over-socket
- rights checking using privs-over-socket

- avoid userland unix-ism
- we are an embedded system, not unix-on-a-phone
- do not keep state in a billion textfiles
- do not rely on the shell for anything besides debugging

- write it once
- never use two APIs or systems where one will do
- avoid excessive layering: in the long run, this overhead kills
- only one object model
- use the java object model
- do not build elaborate c++ object models

- minimal native code
- write as much as possible in java
- we are building a java based system: that decision is final

UNITED STATES DISTRICT COURT NORTHERN DISTRICT OF CALIFORNIA
TRIAL EXHIBIT 23
CASE NO. 10-03561 WHA
DATE ENTERED _____
BY _____ DEPUTY CLERK

HIGHLY CONFIDENTIAL – ATTORNEY'S EYES ONLY Oracle America v. Google, 3:10-cv-03561-WHA GOOGLE-04-00050098

Trial Exhibit 23, Page 1 of 2

TX 23 at p. 1

From: Brian Swetland
To: Andy Rubin
Sent: August 9, 2007

"We are building an embedded system. It is *massively* slower and has *massively* less memory than a modern desktop or server computer."

"We already use too much memory and execute too much code.

Embedded is about doing more with less. If it is not approached that way, you get terrible, slow, unusable systems. It's not pretty. Every cycle of work you do is further reduction of battery life."

"This really isn't a debate. The system is too slow. The system uses too much memory. Smaller, simpler, faster, more reliable wins. Across the whole system."

TX 218 at p. 1

Infringed Patents Improve Performance And Memory

US00RE38104E

(19) **United States**
(12) **Reissued Patent**
Gosling

(10) **Patent Number: US RE38,104 E**
(45) **Date of Reissued Patent: Apr. 29, 2003**

(54) **METHOD AND APPARATUS FOR RESOLVING DATA REFERENCES IN GENERATED CODE**

(75) Inventor: **James Gosling**, Redwood City, CA (US)

(73) Assignee: **Sun Microsystems, Inc.**, Palo Alto, CA (US)

(*) Notice: This patent is subject to a terminal disclaimer.

(21) Appl. No.: **09/261,570**

(22) Filed: **Mar. 3, 1999**

Related U.S. Patent Documents

Reissue of:
(64) Patent No.: **5,367,685**
Issued: **Nov. 22, 1994**
Appl. No.: **07/994,655**
Filed: **Dec. 22, 1992**

(51) **Int. Cl.**[7] **G06F 9/45**
(52) **U.S. Cl.** **717/140**; 717/106; 717/136; 717/139; 717/142; 717/146
(58) **Field of Search** 717/2, 5, 7, 8, 717/106–108, 114, 116, 146–147

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,636,940	A *	1/1987	Goodwin, Jr.	717/4
4,667,290	A	5/1987	Goss et al.	
4,667,930	A *	5/1987	Goss et al.	248/610
4,686,623	A *	8/1987	Wallace	717/8
4,729,096	A *	3/1988	Larson	717/5
4,773,097	A *	9/1988	Kanada et al.	717/9
5,201,050	A *	4/1993	McKeeman et al.	717/7
5,230,050	A *	7/1993	Iitsuka et al.	717/7
5,276,881	A	1/1994	Chan et al.	
5,280,613	A	1/1994	Chan et al.	
5,307,492	A *	4/1994	Benson	717/7
5,313,614	A *	5/1994	Goetelmann et al.	717/5
5,339,419	A	8/1994	Chan et al.	
5,347,632	A	9/1994	Filepp et al.	
5,428,792	A	6/1995	Conner et al.	

(List continued on next page.)

OTHER PUBLICATIONS

Adele Goldberg and David Robson, "Smalltalk-80-The Language and its Implementation", Xerox Palo Alto Research Center, 1983 (reprinted with corrections, Jul. 1985) pp. 1–720.

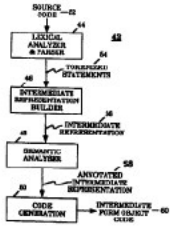
(List continued on next page.)

Primary Examiner—Thomas M. Heckler
(74) *Attorney, Agent, or Firm*—Finnegan, Henderson, Farabow, Garrett & Dunner, L.L.P.

(57) **ABSTRACT**

A hybrid compiler-interpreter comprising a compiler for "compiling" source program code, and an interpreter for interpreting the "compiled" code, is provided to a computer system. The compiler comprises a code generator that generates code in intermediate form with data references made on a symbolic basis. The interpreter comprises a main interpretation routine, and two data reference handling routines, a dynamic field reference routine for handling symbolic references, and a static field reference routine for handling numeric references. The dynamic field reference routine, when invoked, resolves a symbolic reference and rewrites the symbolic reference into a numeric reference. After re-writing, the dynamic field reference routine returns to the main interpretation routine without advancing program execution to the next instruction, thereby allowing the rewritten instruction with numeric reference to be re-executed. The static field reference routine, when invoked, obtain data for the program from a data object based on the numeric reference. After obtaining data, the static field reference routine advances program execution to the next instruction before returning to the interpretation routine. The main interpretation routine selectively invokes the two data reference handling routines depending on whether the data reference is an interaction in a symbolic or a numeric reference.

31 Claims, 5 Drawing Sheets



OAGOOGLE0000052236

TX 4015

US006061520A

United States Patent [19]
Yellin et al.

[11] **Patent Number: 6,061,520**
[45] **Date of Patent: May 9, 2000**

[54] **METHOD AND SYSTEM FOR PERFORMING STATIC INITIALIZATION**

[75] Inventors: **Frank Yellin**, Redwood City; **Richard D. Tuck**, San Francisco, both of Calif.

[73] Assignee: **Sun Microsystems, Inc.**, Palo Alto, Calif.

[21] Appl. No.: **09/055,947**

[22] Filed: **Apr. 7, 1998**

[51] **Int. Cl.**[7] **G06F 9/45**; G06F 3/00
[52] **U.S. Cl.** **395/705**; 395/704; 395/500.43; 709/100
[58] **Field of Search** 395/705, 707, 395/709

[56] **References Cited**

U.S. PATENT DOCUMENTS

5,361,350	11/1994	Conner et al.	707/103
5,367,685	11/1994	Gosling	395/707
5,421,016	5/1995	Conner et al.	395/707
5,437,025	7/1995	Bale et al.	707/103
5,615,400	3/1997	Cowear et al.	709/305
5,668,999	9/1997	Gosling	395/704
5,812,828	9/1998	Kaufer et al.	395/500.43
5,815,718	9/1998	Tuck	395/705
5,900,899	5/1999	Steele, Jr.	707/206
5,966,702	10/1999	Fraske et al.	707/1
5,999,732	12/1999	Bak et al.	395/708
6,003,038	12/1999	Chen	707/103

OTHER PUBLICATIONS

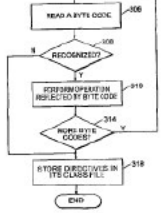
Tyma, P., "Tuning Java Performance ", Dr. Dobb's Journal [online], vol. 21, No. 4, pp. 52–58, Apr. 1996.
Cierniak et al., "Briki: an optimizing Java compiler". IEEE/IEEE Electronic Library, Proceedings, IEEE Compcon pp. 179–184, Feb. 1997.
Bell, D.; "Make Java fast: Optimize!". Javaworld[online]. Cramer et al.; "Compiling Java just in time". IEEE Electronic Library[online], vol. 17, Iss. 3, pp. 36–43, May 1997.
Lindholm, Tim et al, The Java Virtual Machine Specification, 1997.
Cowear et al; "Targeting GNAT to the Java virtual machine". ACM Digital Library[online], Proceedings of the conference on TRI–Ada '97, May 1997.
Hsieh et al.; "Compiles for improved Java Performance". IEEE Electronic Library[online] Computer[online], vol. 30, Iss. 6, pp. 67–75, Jun. 1997.
Armstrong, E.; "Hotspot: A new breed of virtual machine". Javaworld[online].
Gosling et al.; The Java Language Specification. Reading, MA, Addison-Wesley. Ch 12, pp. 215–236, Sep. 1996.

Primary Examiner—Tariq R. Hafiz
Assistant Examiner—Kelvin E. Booker
Attorney, Agent, or Firm—Finnegan, Henderson, Farabow, Garrett & Dunner, L.L.P.

[57] **ABSTRACT**

The disclosed system represents an improvement over conventional systems for initializing static arrays by reducing the amount of code executed by the virtual machine to statically initialize an array. To realize this reduction, when consolidating class files, the preloader identifies all <clinit> methods and play executes these methods to determine the static initialization performed by them. The preloader then creates an expression indicating the static initialization performed by the <clinit> method and stores this expression in the .nolass file, replacing the <clinit> method. As such, the code of the <clinit> method, containing many instructions, is replaced by a single expression instructing the virtual machine to perform static initialization, thus saving a significant amount of memory. The virtual machine is modified to recognize this expression and perform the appropriate static initialization of an array.

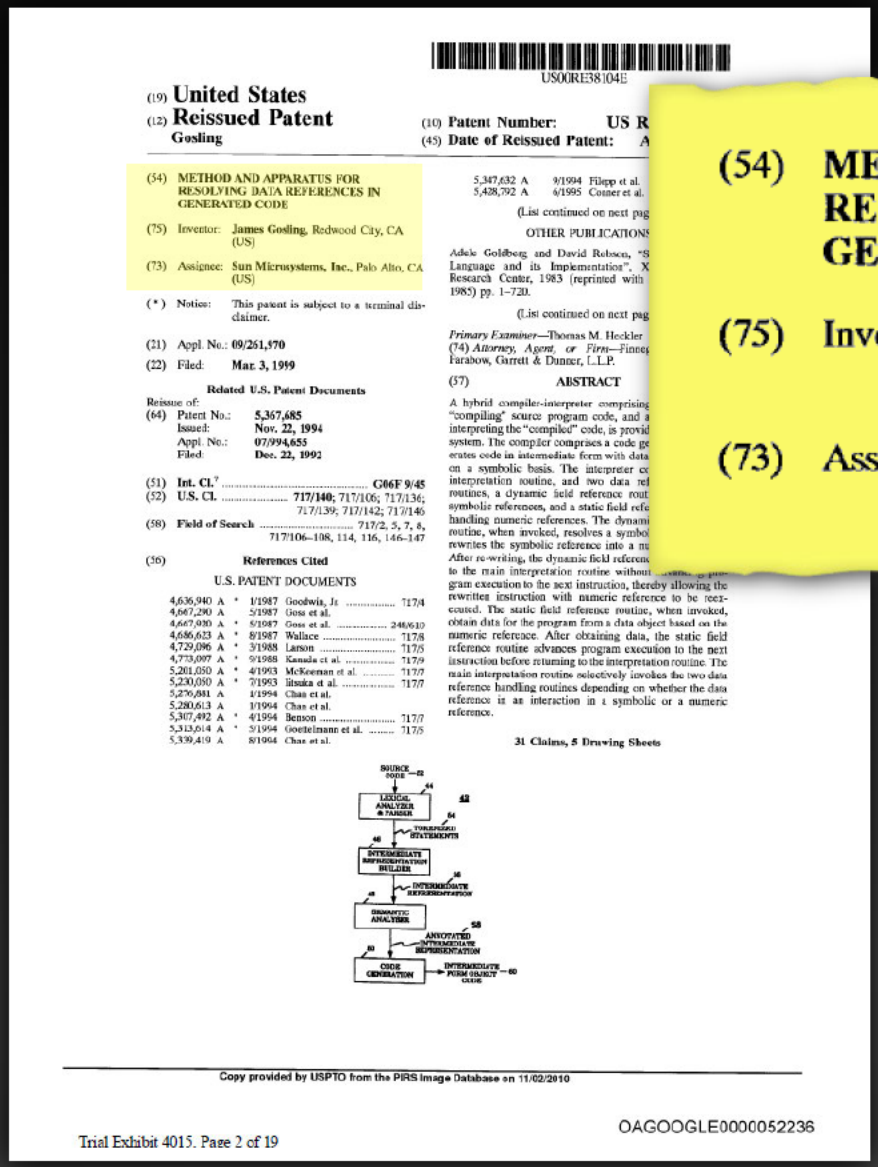
23 Claims, 3 Drawing Sheets



TX 4011

Google’s Infringement of ’104 Patent

RE38,104 Patent Solves Performance Problem



US00RE38104E

(19) **United States**
(12) **Reissued Patent**
Gosling

(10) **Patent Number:** US R
(45) **Date of Reissued Patent:** A

(54) METHOD AND APPARATUS FOR RESOLVING DATA REFERENCES IN GENERATED CODE

(75) Inventor: James Gosling, Redwood City, CA (US)

(73) Assignee: Sun Microsystems, Inc., Palo Alto, CA (US)

(*) Notice: This patent is subject to a terminal disclaimer.

(21) Appl. No.: 09/261,570

(22) Filed: Mar. 3, 1999

Related U.S. Patent Documents

Reissue of:
(64) Patent No.: 5,367,685
Issued: Nov. 22, 1994
Appl. No.: 07/994,655
Filed: Dec. 22, 1992

(51) Int. Cl.[7] G06F 9/45
(52) U.S. Cl. 717/140; 717/106; 717/136; 717/139; 717/142; 717/146
(58) Field of Search 717/2, 5, 7, 8, 717/106–108, 114, 116, 146–147

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,636,940 A * 1/1987 Goodwin, Jr. 717/4
4,667,290 A 5/1987 Goss et al.
4,667,930 A * 5/1987 Goss et al. 248/610
4,686,623 A * 8/1987 Wallace 717/8
4,729,096 A * 3/1988 Larson 717/5
4,773,097 A * 9/1988 Kanada et al. 717/9
5,201,050 A * 4/1993 McKeeman et al. 717/7
5,230,050 A * 7/1993 Iitsuka et al. 717/7
5,276,881 A 1/1994 Chan et al.
5,280,613 A 1/1994 Chan et al.
5,307,492 A * 4/1994 Benson 717/7
5,313,614 A * 5/1994 Goetelmann et al. 717/5
5,339,419 A 8/1994 Chan et al.
5,347,632 A 9/1994 Filepp et al.
5,428,792 A 6/1995 Conner et al.

(List continued on next pag

OTHER PUBLICATIONS

Adele Goldberg and David Robson, "S Language and its Implementation", X Research Center, 1983 (reprinted with 1985) pp. 1–720.

(List continued on next pag

Primary Examiner—Thomas M. Heckler
(74) *Attorney, Agent, or Firm*—Finneg Farabow, Garrett & Dunner, L.L.P.

(57) **ABSTRACT**

A hybrid compiler-interpreter comprising "compiling" source program code, and a interpreting the "compiled" code, is provid system. The compiler comprises a code ge erates code in intermediate form with data on a symbolic basis. The interpreter co interpretation routine, and two data ref routines, a dynamic field reference rout symbolic references, and a static field refe handling numeric references. The dynami routine, when invoked, resolves a symbol rewrites the symbolic reference into a nu After re-writing, the dynamic field referenc to the main interpretation routine without gram execution to the next instruction, thereby allowing the rewritten instruction with numeric reference to be reex-ecuted. The static field reference routine, when invoked, obtain data for the program from a data object based on the numeric reference. After obtaining data, the static field reference routine advances program execution to the next instruction before returning to the interpretation routine. The main interpretation routine selectively invokes the two data reference handling routines depending on whether the data reference is an interaction is a symbolic or a numeric reference.

31 Claims, 5 Drawing Sheets

OAGOOGLE0000052236

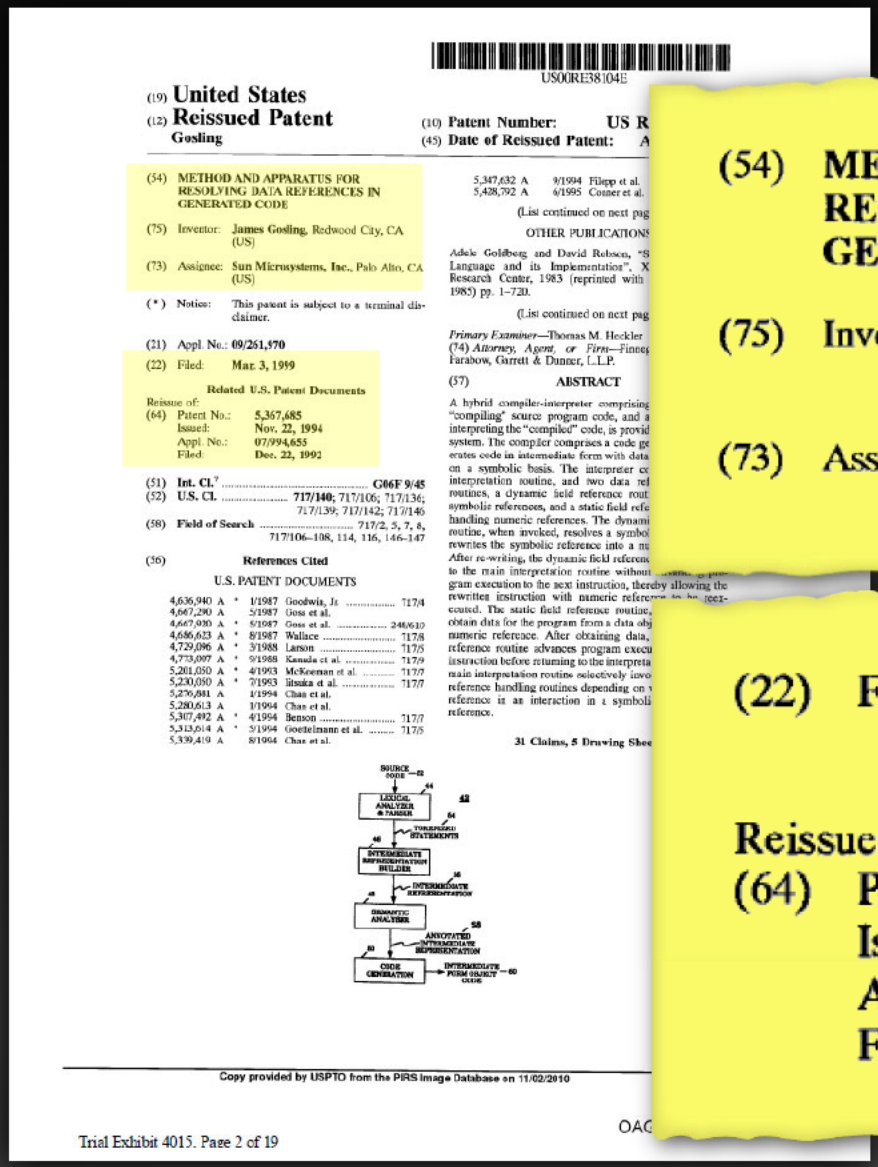
(54) **METHOD AND APPARATUS FOR RESOLVING DATA REFERENCES IN GENERATED CODE**

(75) Inventor: **James Gosling**, Redwood City, CA (US)

(73) Assignee: **Sun Microsystems, Inc.**, Palo Alto, CA (US)

TX 4015

Patent Office Thoroughly Examined '104 Patented Solution



US00RE38104E

(19) **United States**
(12) **Reissued Patent**
Gosling

(10) **Patent Number:** US R
(45) **Date of Reissued Patent:** A

(54) METHOD AND APPARATUS FOR RESOLVING DATA REFERENCES IN GENERATED CODE

(75) Inventor: James Gosling, Redwood City, CA (US)

(73) Assignee: Sun Microsystems, Inc., Palo Alto, CA (US)

(*) Notice: This patent is subject to a terminal disclaimer.

(21) Appl. No.: 09/261,570

(22) Filed: Mar. 3, 1999

Related U.S. Patent Documents

Reissue of:
(64) Patent No.: 5,367,685
Issued: Nov. 22, 1994
Appl. No.: 07/994,655
Filed: Dec. 22, 1992

(51) Int. Cl.[7] G06F 9/45
(52) U.S. Cl. 717/140; 717/106; 717/136; 717/139; 717/142; 717/146
(58) Field of Search 717/2, 5, 7, 8, 717/106–108, 114, 116, 146–147

(56) References Cited

U.S. PATENT DOCUMENTS

4,636,940	A *	1/1987	Goodwin, Jr	717/4
4,667,290	A	5/1987	Goss et al.	
4,667,930	A *	5/1987	Goss et al.	248/610
4,686,623	A *	8/1987	Wallace	717/8
4,729,096	A *	3/1988	Larson	717/5
4,773,097	A *	9/1988	Kanada et al.	717/9
5,201,050	A *	4/1993	McKeenan et al.	717/7
5,230,050	A *	7/1993	Iitsuka et al.	717/7
5,276,881	A	1/1994	Chan et al.	
5,280,613	A	1/1994	Chan et al.	
5,307,492	A *	4/1994	Benson	717/7
5,313,614	A *	5/1994	Goetelmann et al.	717/5
5,339,419	A	8/1994	Chan et al.	
5,347,632	A	9/1994	Filepp et al.	
5,428,792	A	6/1995	Conner et al.	

(List continued on next pag

OTHER PUBLICATIONS

Adele Goldberg and David Robson, "S Language and its Implementation", X Research Center, 1983 (reprinted with 1985) pp. 1–720.

(List continued on next pag

Primary Examiner—Thomas M. Heckler
(74) *Attorney, Agent, or Firm*—Finne Farabow, Garrett & Dunner, L.L.P.

(57) ABSTRACT

A hybrid compiler-interpreter comprising "compiling" source program code, and a interpreting the "compiled" code, is provid system. The compiler comprises a code ge erates code in intermediate form with data on a symbolic basis. The interpreter co interpretation routine, and two data ref routines, a dynamic field reference rout symbolic references, and a static field refe handling numeric references. The dynami routine, when invoked, resolves a symbol rewrites the symbolic reference into a nu After re-writing, the dynamic field referenc to the main interpretation routine without gram execution to the next instruction, thereby allowing the rewritten instruction with numeric referen counted. The static field reference routine, obtain data for the program from a data obj numeric reference. After obtaining data, reference routine advances program execu instruction before returning to the interpreta main interpretation routine selectively invo reference handling routines depending on reference is an interaction in a symboli reference.

31 Claims, 5 Drawing Shee

(54) **METHOD AND APPARATUS FOR RESOLVING DATA REFERENCES IN GENERATED CODE**

(75) Inventor: **James Gosling**, Redwood City, CA (US)

(73) Assignee: **Sun Microsystems, Inc.**, Palo Alto, CA (US)

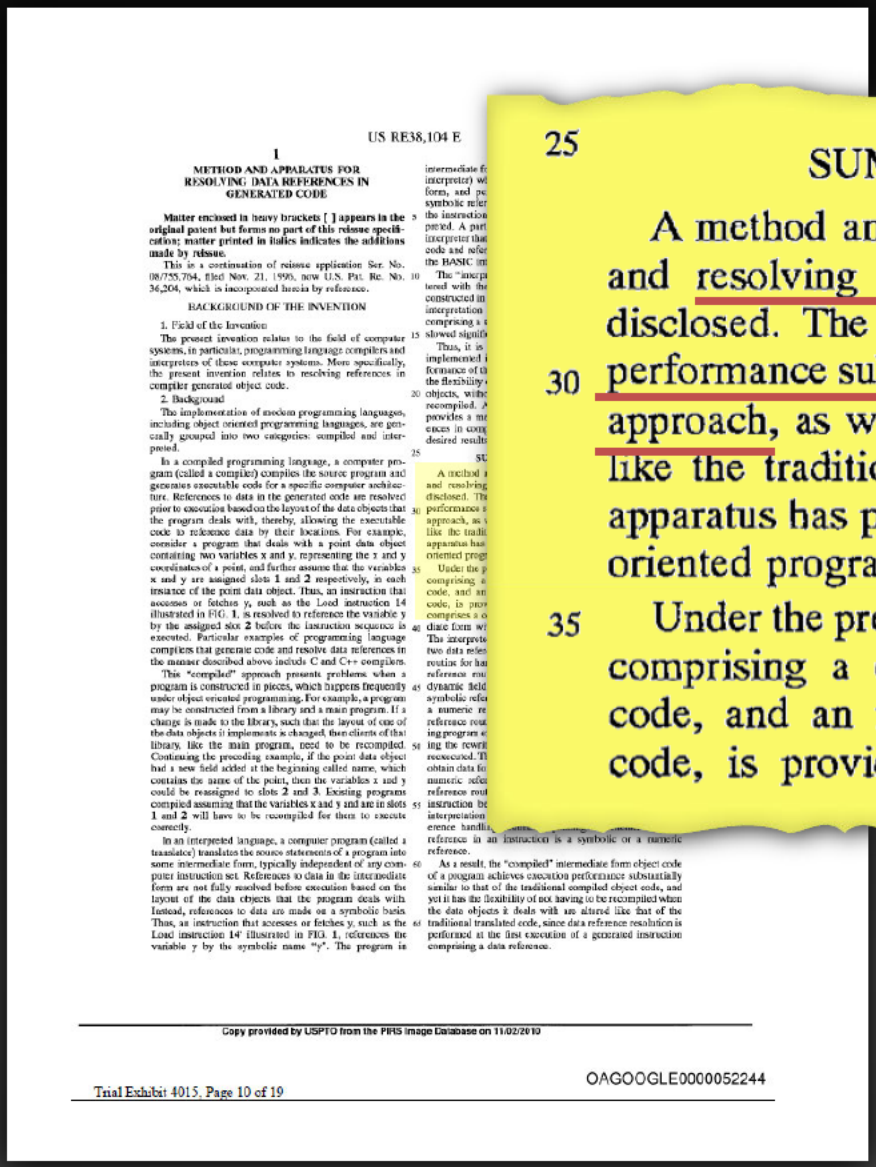
(22) Filed: **Mar. 3, 1999**

Related U.S. Patent Documents

Reissue of:
(64) Patent No.: **5,367,685**
Issued: **Nov. 22, 1994**
Appl. No.: **07/994,655**
Filed: **Dec. 22, 1992**

TX 4015

’104 Patented Solution



US RE38,104 E

1

METHOD AND APPARATUS FOR RESOLVING DATA REFERENCES IN GENERATED CODE

Matter enclosed in heavy brackets [ ] appears in the original patent but forms no part of this reissue specification; matter printed in italics indicates the additions made by reissue.

This is a continuation of reissue application Ser. No. 08/755,704, filed Nov. 21, 1996, now U.S. Pat. Re. No. 36,204, which is incorporated herein by reference.

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates to the field of computer systems, in particular, programming languages compilers and interpreters of these computer systems. More specifically, the present invention relates to resolving references in compiler generated object code.

2. Background

The implementation of modern programming languages, including object oriented programming languages, are generally grouped into two categories: compiled and interpreted.

In a compiled programming language, a computer program (called a compiler) compiles the source program and generates executable code for a specific computer architecture. References to data in the generated code are resolved prior to execution based on the layout of the data objects that the program deals with, thereby, allowing the executable code to reference data by their locations. For example, consider a program that deals with a point data object containing two variables x and y, representing the x and y coordinates of a point, and further assume that the variables x and y are assigned slots 1 and 2 respectively, in each instance of the point data object. Thus, an instruction that accesses or fetches y, such as the Load instruction 14 illustrated in FIG. 1, is resolved to reference the variable y by the assigned slot 2 before the instruction sequence is executed. Particular examples of programming language compilers that generate code and resolve data references in the manner described above include C and C++ compilers.

This "compiled" approach presents problems when a program is constructed in pieces, which happens frequently under object oriented programming. For example, a program may be constructed from a library and a main program. If a change is made to the library, such that the layout of one of the data objects it implements is changed, then clients of that library, like the main program, need to be recompiled. Continuing the preceding example, if the point data object had a new field added at the beginning called name, which contains the name of the point, then the variables x and y could be reassigned to slots 2 and 3. Existing programs compiled assuming that the variables x and y and are in slots 1 and 2 will have to be recompiled for them to execute correctly.

In an interpreted language, a computer program (called a translator) translates the source statements of a program into some intermediate form, typically independent of any computer instruction set. References to data in the intermediate form are not fully resolved before execution based on the layout of the data objects that the program deals with. Instead, references to data are made on a symbolic basis. Thus, an 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". The program in

Copy provided by USPTO from the PIRS Image Database on 11/02/2010

OAGOOGLE0000052244

Trial Exhibit 4015, Page 10 of 19

25

SUMMARY OF THE INVENTION

A method and apparatus for generating executable code and resolving data references in the generated code is disclosed. The method and apparatus provides execution 30 performance substantially similar to the traditional compiled approach, as well as the flexibility of altering data objects like the traditional interpreted approach. The method and apparatus has particular application to implementing object oriented programming languages in computer systems.

35 Under the present invention, a hybrid compiler-interpreter comprising a compiler for "compiling" source program code, and an interpreter for interpreting the "compiled" code, is provided to a computer system. The compiler

TX 4015 at p. 10

’104 Patent Claims Cover Symbolic Reference Resolution

’104 Patent, Claim 11

11. An apparatus comprising:

a memory containing intermediate form object code constituted by a set of instructions, certain of said instructions containing one or more symbolic references; and

a processor configured to execute said instructions containing one or more symbolic references by determining a numerical reference corresponding to said symbolic reference, storing said numerical references, and obtaining data in accordance to said numerical references.

Google contends missing from Android

Meaning Of “symbolic reference”

symbolic reference

A reference that identifies data by a name other than the numeric memory location of the data, and that is resolved dynamically rather than statically.

Android Resolves Symbolic References

Google programmers stated in Android code that Android resolves "symbolic references":

```
/*
 * Link (prepare and resolve).  Verification is deferred until later.
 *
 * This converts symbolic references into pointers.  It's independent of
 * the source file format.
```

(Comments for "dvmLinkClass" routine in dalvik\vm\oo\Class.c)

symbolic references

TX 47.15, 46.15

Android Relies On Symbolic Reference Resolution To Run Faster

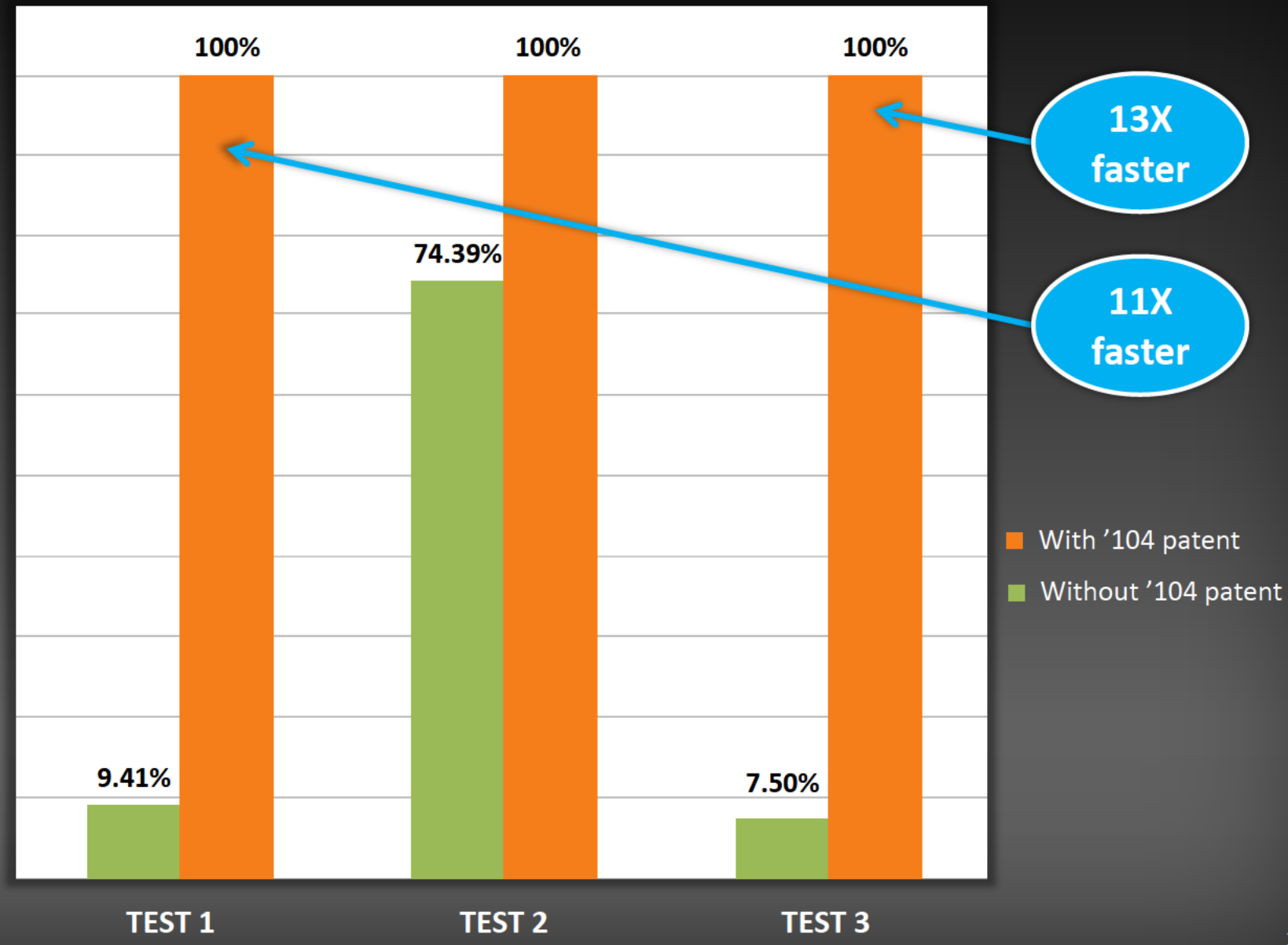
Dan Bornstein
Former Google Android Tech Lead
Google I/O 2008 Video entitled "Dalvik Virtual Machines Internals"

"We do optimization. And, so the first time that, that a data-dex file lands on a device, we, we do that verification work, we also, we also augment that file, if we have to we will do byte swapping and pad, pad out structures and in addition, we, we have a bunch of other things that we do such that when, when it comes time to run, we can run that much faster. So as an example of static linking, before when a dex files arrives on a, on a device it will have symbolic references to methods and fields, but afterwards it might just be a simple integer v-table 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 v-table."

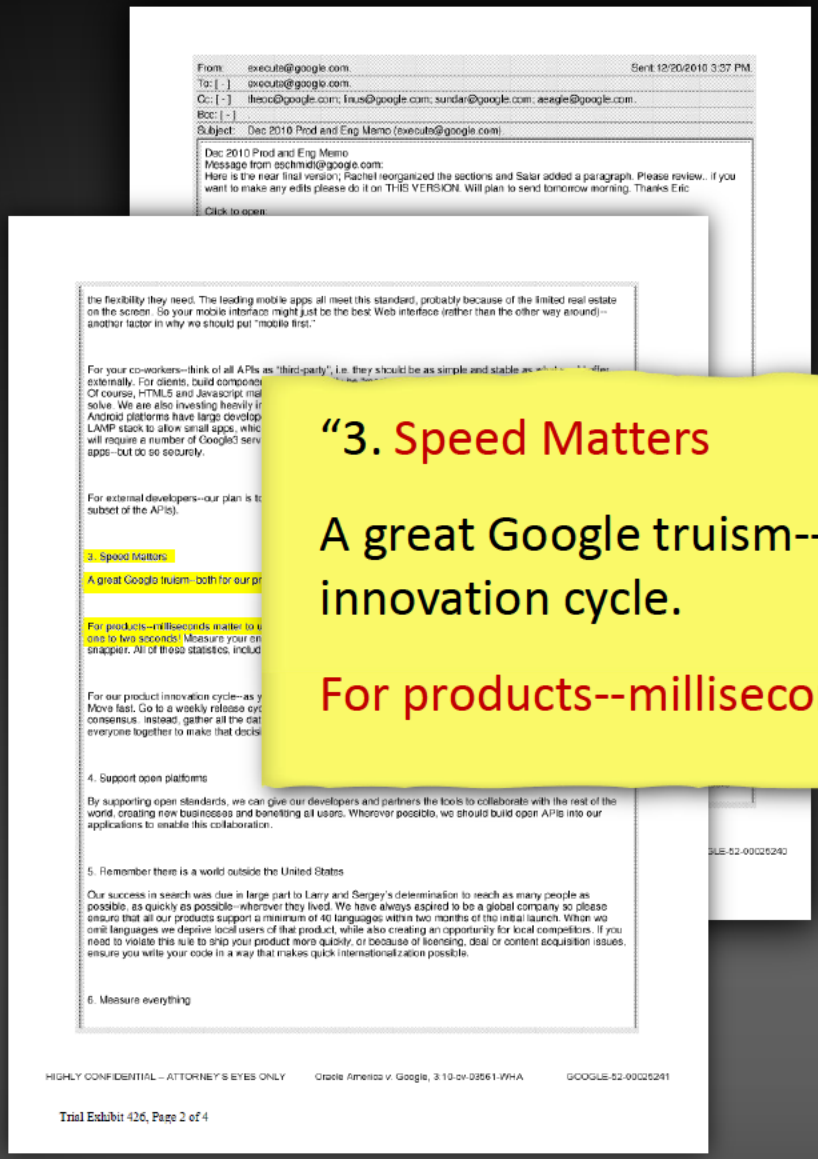
TX 816

symbolic references

Benchmark Tests Prove Android Runs Faster Because Of ’104 Patent



Speed Matters


From: execute@google.com Sent: 12/20/2010 3:37 PM
To: [-] execute@google.com
Cc: [-] theoc@google.com; linus@google.com; sundar@google.com; aeagle@google.com
Bcc: [-]
Subject: Dec 2010 Prod and Eng Memo (execute@google.com)

Dec 2010 Prod and Eng Memo
Message from eschmidt@google.com:
Here is the near final version; Rachel reorganized the sections and Salar added a paragraph. Please review.. if you want to make any edits please do it on THIS VERSION. Will plan to send tomorrow morning. Thanks Eric

Click to open

the flexibility they need. The leading mobile apps all meet this standard, probably because of the limited real estate on the screen. So your mobile interface might just be the best Web interface (rather than the other way around)--another factor in why we should put "mobile first."

For your co-workers--think of all APIs as "third-party", i.e. they should be as simple and stable as

externally. For clients, build compone

Of course, HTML5 and Javascript ma

solve. We are also investing heavily i

Android platforms have large develop

LAMP stack to allow small apps, whic

will require a number of Google8 serv

apps--but do so securely.

For external developers--our plan is t

subset of the APIs).

3. Speed Matters

A great Google truism--both for our pr

For products--milliseconds matter to u

are to two seconds! Measure your on

snapper. All of these statistics includ

For our product innovation cycle--as y

Move fast. Go to a weekly release cyc

consensus. Instead, gather all the dat

everyone together to make that decis

4. Support open platforms

By supporting open standards, we can give our developers and partners the tools to collaborate with the rest of the world, creating new businesses and benefiting all users. Wherever possible, we should build open APIs into our applications to enable this collaboration.

5. Remember there is a world outside the United States

Our success in search was due in large part to Larry and Sergey's determination to reach as many people as possible, as quickly as possible--wherever they lived. We have always aspired to be a global company so please ensure that all our products support a minimum of 40 languages within two months of the initial launch. When we omit languages we deprive local users of that product, while also creating an opportunity for local competitors. If you need to violate this rule to ship your product more quickly, or because of licensing, deal or content acquisition issues, ensure you write your code in a way that makes quick internationalization possible.

6. Measure everything

HIGHLY CONFIDENTIAL – ATTORNEY'S EYES ONLY Oracle America v. Google, 3:10-cv-03561-WHA GOOGLE-52-00025241

Trial Exhibit 426, Page 2 of 4

GLE-52-00025240

From: Eric Schmidt
To: execute@google.com
Sent: December 20, 2010

"3. Speed Matters

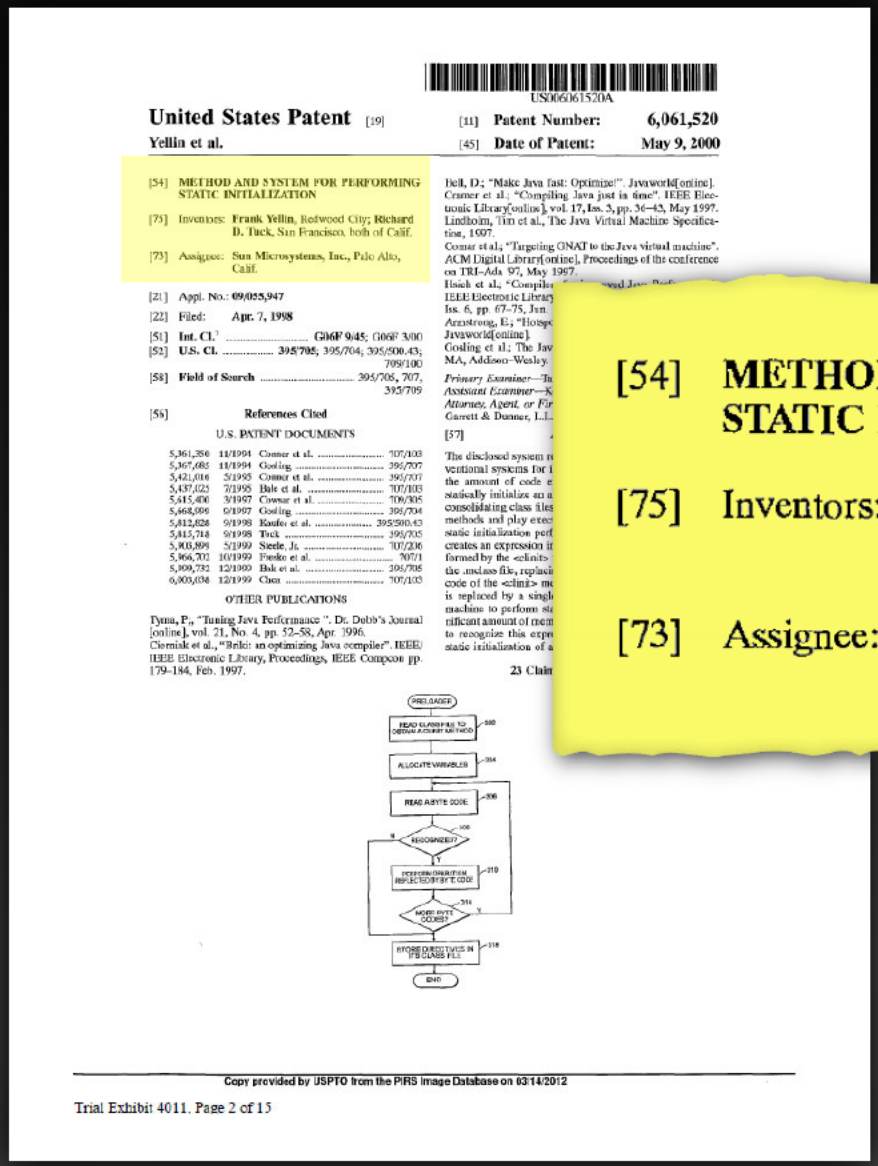
A great Google truism--both for our products and our product innovation cycle.

For products--milliseconds matter to users."

TX 426 at p.2

Google’s Infringement of ’520 Patent

6,061,520 Patent Solves Memory Problem



US006061520A

United States Patent [19]
Yellin et al.

[11] **Patent Number: 6,061,520**
[45] **Date of Patent: May 9, 2000**

[54] METHOD AND SYSTEM FOR PERFORMING STATIC INITIALIZATION

[75] Inventors: **Frank Yellin**, Redwood City; **Richard D. Tuck**, San Francisco, both of Calif.

[73] Assignee: **Sun Microsystems, Inc.**, Palo Alto, Calif.

[21] Appl. No.: **09/055,947**

[22] Filed: **Apr. 7, 1998**

[51] **Int. Cl.**[7] G06F 9/45; G06F 3/00

[52] **U.S. Cl.** 395/705; 395/704; 395/500.43; 709/100

[58] **Field of Search** 395/705, 707, 395/709

[56] **References Cited**

U.S. PATENT DOCUMENTS

5,361,356	11/1994	Conner et al.	707/103
5,367,685	11/1994	Gosling	395/707
5,421,016	5/1995	Conner et al.	395/707
5,437,025	7/1995	Bale et al.	707/103
5,515,490	5/1997	Conner et al.	709/305
5,668,999	9/1997	Gosling	395/704
5,812,828	9/1998	Kaufer et al.	395/500.43
5,815,718	9/1998	Tuck	395/705
5,819,099	5/1999	Steele, Jr.	707/206
5,966,702	10/1999	Fresko et al.	707/1
5,999,732	12/1999	Bak et al.	395/705
6,003,038	12/1999	Chen	707/103

OTHER PUBLICATIONS

Tyma, P., "Tuning Java Performance ". Dr. Dobb's Journal [online], vol. 21, No. 4, pp. 52–58, Apr. 1996.
Cierniak et al., "Briki: an optimizing Java compiler". IEEE/IEEE Electronic Library, Proceedings, IEEE Compcon pp. 179–184, Feb. 1997.

Bell, D.; "Make Java fast: Optimize!". Javaworld[online].
Cramer et al.; "Compiling Java just in time". IEEE Electronic Library[online], vol. 17, Iss. 3, pp. 36–43, May 1997.
Lindholm, Tim et al., The Java Virtual Machine Specification, 1997.
Conner et al., "Targeting GNAT to the Java virtual machine". ACM Digital Library[online], Proceedings of the conference on TRI–Ada '97, May 1997.
Hsieh et al., "Compile
IEEE Electronic Library
Iss. 6, pp. 67–75, Jun.
Armstrong, E.; "Hotspe
Javaworld[online]
Gosling et al.; The Jav
MA, Addison–Wesley.

Primary Examiner—T
Assistant Examiner—K
Attorney, Agent, or Fir
Garrett & Dunner, L.L.

[57]

The disclosed system re
ventional systems for i
the amount of code e
statically initialize an a
consolidating class files
methods and play exec
static initialization perf
creates an expression it
formed by the <clinit>
the .mclass file, replaci
code of the <clinit> me
is replaced by a single
machine to perform st
nificant amount of mem
to recognize this expre
static initialization of a

23 Clai

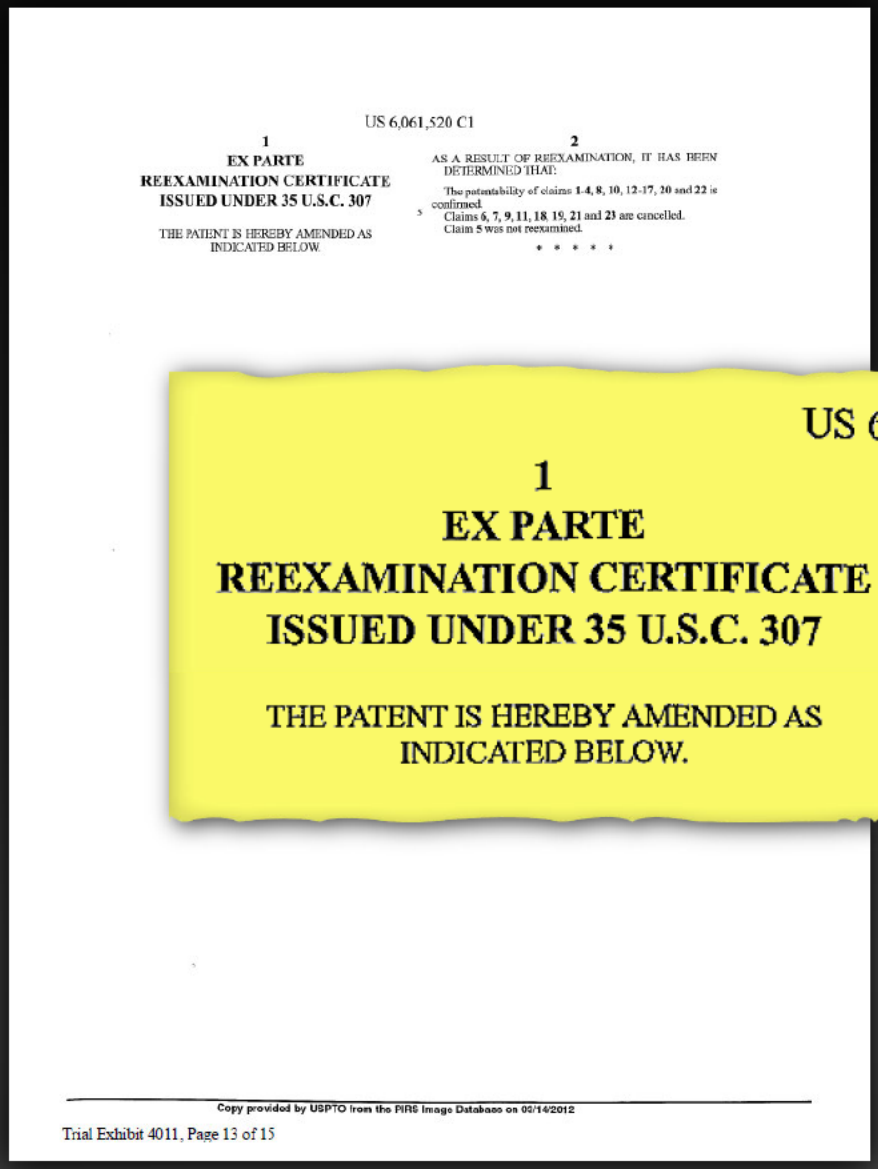
[54] **METHOD AND SYSTEM FOR PERFORMING STATIC INITIALIZATION**

[75] Inventors: **Frank Yellin**, Redwood City; **Richard D. Tuck**, San Francisco, both of Calif.

[73] Assignee: **Sun Microsystems, Inc.**, Palo Alto, Calif.

TX 4011

Patent Office Thoroughly Examined '520 Patented Solution



US 6,061,520 C1

1

EX PARTE
REEXAMINATION CERTIFICATE
ISSUED UNDER 35 U.S.C. 307

THE PATENT IS HEREBY AMENDED AS INDICATED BELOW.

2

AS A RESULT OF REEXAMINATION, IT HAS BEEN DETERMINED THAT:

The patentability of claims **1-4, 8, 10, 12-17, 20** and **22** is confirmed.

5 Claims **6, 7, 9, 11, 18, 19, 21** and **23** are cancelled.

Claim **5** was not reexamined.

* * * * *

US 6,061,520 C1

1

EX PARTE
REEXAMINATION CERTIFICATE
ISSUED UNDER 35 U.S.C. 307

THE PATENT IS HEREBY AMENDED AS INDICATED BELOW.

2

AS A RESULT OF REEXAMINATION, IT HAS BEEN DETERMINED THAT:

The patentability of claims **1-4, 8, 10, 12-17, 20** and **22** is confirmed.

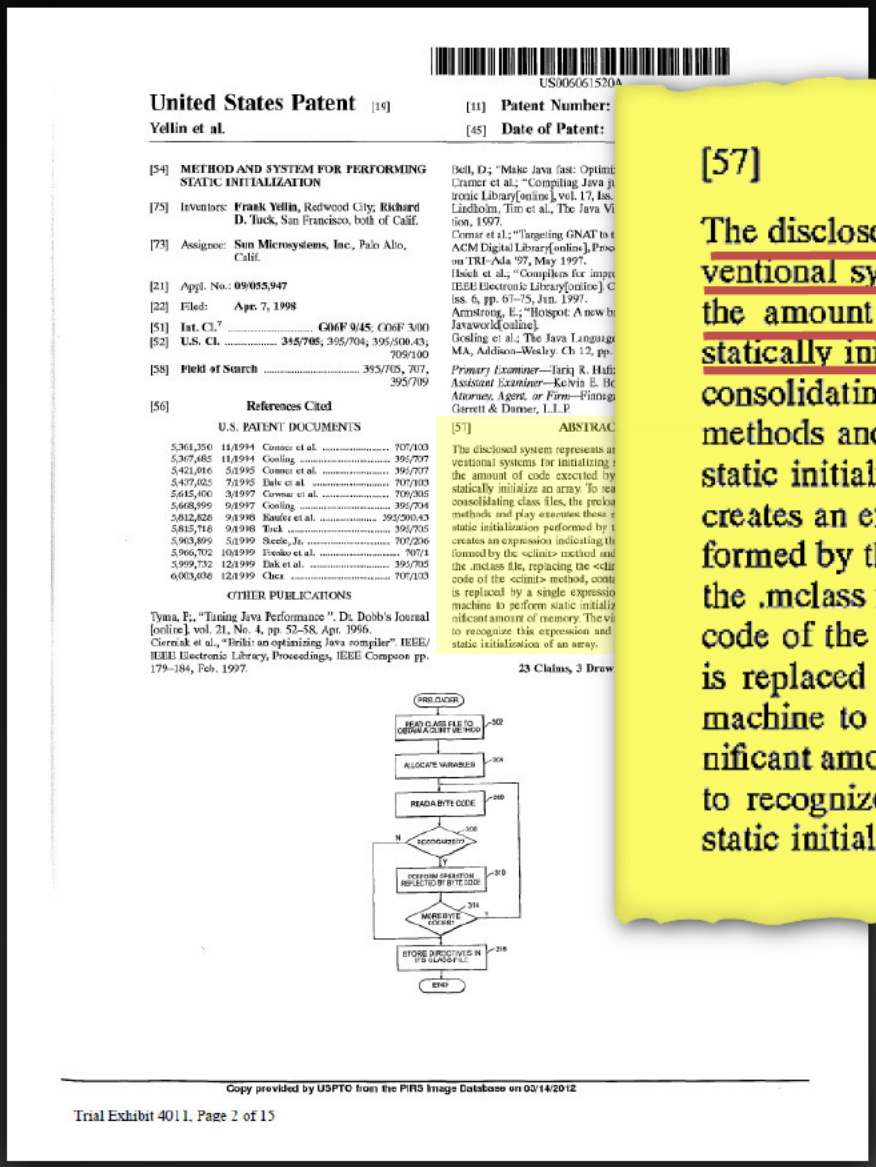
5 Claims **6, 7, 9, 11, 18, 19, 21** and **23** are cancelled.

Claim **5** was not reexamined.

* * * * *

TX 4011 at p.13

’520 Patented Solution



US005061520A

United States Patent [19]
Yellin et al.

[11] **Patent Number:**
[45] **Date of Patent:**

[54] **METHOD AND SYSTEM FOR PERFORMING STATIC INITIALIZATION**

[75] Inventors: **Frank Yellin**, Redwood City; **Richard D. Tuck**, San Francisco, both of Calif.

[73] Assignee: **Sun Microsystems, Inc.**, Palo Alto, Calif.

[21] Appl. No.: **09/055,947**

[22] Filed: **Apr. 7, 1998**

[51] **Int. Cl.**[7] G06F 9/45; G06F 3/00

[52] **U.S. Cl.** **345/705**; 395/704; 395/500.43; 709/100

[58] **Field of Search** 395/705, 707, 395/709

[56] **References Cited**

U.S. PATENT DOCUMENTS

5,361,350	11/1994	Conner et al.	707/103
5,367,685	11/1994	Gosling	395/707
5,421,016	5/1995	Conner et al.	395/707
5,437,025	7/1995	Bale et al.	707/103
5,615,400	3/1997	Cowear et al.	709/305
5,668,999	9/1997	Gosling	395/704
5,812,826	9/1998	Raufer et al.	395/500.43
5,815,718	9/1998	Tuck	395/705
5,903,899	5/1999	Steele, Jr.	707/206
5,966,702	10/1999	Fresko et al.	707/1
5,999,732	12/1999	Bak et al.	395/705
6,003,038	12/1999	Chen	707/103

OTHER PUBLICATIONS

Tyma, P., "Tuning Java Performance", Dr. Dobb's Journal [online], vol. 21, No. 4, pp. 52–58, Apr. 1996.
Cierniak et al., "Briki: an optimizing Java compiler", IEEE/IEEE Electronic Library, Proceedings, IEEE Compcon pp. 179–184, Feb. 1997.

[Bell, D.; "Make Java fast: Optimi
Cramer et al.; "Compiling Java ju
tronic Library[online], vol. 17, Iss.
Lindholm, Tim et al., The Java Vi
tion, 1997.
Cornar et al.; "Targeting GNAT to
ACM Digital Library[online], Proc
on TRI-Ada '97, May 1997.
Hsieh et al.; "Compilers for impr
IEEE Electronic Library[online], C
Iss. 6, pp. 67–75, Jun. 1997.
Armstrong, E.; "Hotspot: A new b
Javaworld[online].
Gosling et al.; The Java Language
MA, Addison-Wesley. Ch 12, pp.

Primary Examiner—Tariq R. Hafiz
Assistant Examiner—Kelvin E. Bo
Attorney, Agent, or Firm—Finnegan Garrett & Dunner, L.L.P.

[57] ABSTRAC

The disclosed system represents an
vention systems for initializing
the amount of code executed by
statically initialize an array. To rea
consolidating class files, the preloa
methods and play executes these
static initialization performed by
creates an expression indicating th
formed by the <clinit> method and
the .mclass file, replacing the <cli
code of the <clinit> method, cont
is replaced by a single expressio
machine to perform static initiali
nificant amount of memory. The vi
to recognize this expression and
static initialization of an array.

23 Claims, 3 Draw

Copy provided by USPTO from the PIRS Image Database on 03/14/2012

Trial Exhibit 4011, Page 2 of 15

[57] **ABSTRACT**

The disclosed system represents an improvement over conventional systems for initializing static arrays by reducing the amount of code executed by the virtual machine to statically initialize an array. To realize this reduction, when consolidating class files, the preloader identifies all <clinit> methods and play executes these methods to determine the static initialization performed by them. The preloader then creates an expression indicating the static initialization performed by the <clinit> method and stores this expression in the .mclass file, replacing the <clinit> method. As such, the code of the <clinit> method, containing many instructions, is replaced by a single expression instructing the virtual machine to perform static initialization, thus saving a significant amount of memory. The virtual machine is modified to recognize this expression and perform the appropriate static initialization of an array.

TX 4011

’520 Patent Claims Cover Simulating Execution

’520 Patent, Claim 1

1. A method in a data processing system for statically initializing an array, comprising the steps of:

compiling source code containing the array with static values to generate a class file with a clinit method containing byte codes to statically initialize the array to the static values;

receiving the class file into a preloader;

simulating execution of the byte codes of the clinit method against a memory without executing the byte codes to identify the static initialization of the array by the preloader;

storing into an output file an instruction requesting the static initialization of the array; and

interpreting the instruction by a virtual machine to perform the static initialization of the array.

Google contends missing from Android

Android Simulates Execution

```
36  /**
37   * Class which knows how to simulate the effects of executing bytecode.
38   *
39   * <p><b>Note:</b> This class is not thread-safe. If multiple threads
40   * need to use a single instance, they must synchronize access
     explicitly
41   * between themselves.</p>
42   */
43  publi
44      /
45       * {@code non-null;} canned error message for local variable
46       * table mismatches
47       */
48      private static final String LOCAL_MISMATCH_ERROR =
49          "This is symptomatic of .class transformation tools that ignore
             " +
50          "local variable information.";
51
52      /** {@code non-null;} machine to use when simulating */
53      private final Machine machine;
54
55      /** {@code non-null;} array of bytecode */
56      private final BytecodeArray code;
```

simulate the effects of executing bytecode

TX 47.16, 46.16 Simulator.java

Android Relies On Simulating Execution To Save Memory And Run Faster

Dan Bornstein
Former Google Android Tech Lead
Google I/O 2008 Video entitled "Dalvik Virtual Machines Internals"

"So, sometimes you really need to have just a big array of data and if you've ever looked at what something like this looks like in a .class file, it's not pretty. ... And each time you add another element to say an int array, it's 11 bytes of code and constant combined and another four instruction dispatches. … So knowing that, this is what we do in a dex file. This is the same code, turned, which has been translated into a .dex file. You can see that for, in this case we're really, we're only using 46 bytes for this example and as you add elements to that int array it's just four more bytes per element which is exactly the data represented. And we only have to interpret one opcode to do that entire initialization and that's the third one down, the fill array data…. And this is both a speed and a space efficiency win. Measured on our system libraries it saves us something like a 100K. …"

TX 816

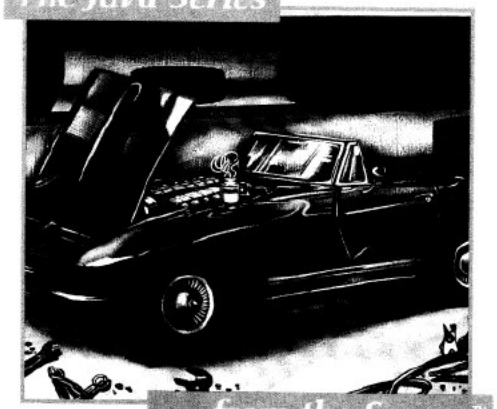
Google’s Knowledge of Patents

Google Knew About '104 Patent And Its Importance

Tim Lindholm • Frank Yellin

The Java™ Virtual Machine Specification

The Java Series



... from the Source™

JAVA

UNITED STATES DISTRICT COURT
NORTHERN DISTRICT OF CALIFORNIA
TRIAL EXHIBIT 25
CASE NO. 10-03561 WHA
DATE ENTERED
BY
DEPUTY CLERK

Trial Exhibit 25, Page 1 of 488

TX 25

CHAPTER 9

An Optimization

THIS chapter describes an optimization implemented in Sun's version of the Java Virtual Machine. In this optimization, compiled Java Virtual Machine code is modified at run time for better performance.

The optimization takes the form of a set of pseudo-instructions. These are variants of normal Java Virtual Machine instructions that take advantage of information learned at run time to do less work than the original instructions. The pseudo-instructions are distinguishable by the suffix *_quick* in their mnemonics.

It is important to understand that these pseudo-instructions are *not* part of the Java Virtual Machine specification or instruction set. They are invisible outside of a Java Virtual Machine implementation. However, inside a Java Virtual Machine implementation they have proven to be an effective optimization.

The technique documented in this chapter is covered by U.S. Patent 5,367,685.

9.1 Dynamic Linking via Rewriting

A compiler targeting the Java Virtual Machine must only emit instructions from the instruction set documented in Chapter 6, "Java Virtual Machine Instruction Set." The optimization described in this chapter works by dynamically replacing occurrences of certain of those instructions, the first time they are executed, by internal, more efficient variants. The new instructions take advantage of loading and linking work done the first time the associated normal instruction is executed.

For instructions that are rewritten, each instance of the instruction is replaced on its first execution by a *_quick* pseudo-instruction. Subsequent execution of that

389

Trial Exhibit 25, Page 401 of 488

TX 25 at p. 389

Google Knew About '104 Patent And Its Importance

Tim Lindholm • Frank Yellin

The Java™ Virtual Machine

CHAPTER 9

An Optimization

THIS chapter describes an optimization implemented in Sun's version of the Java Virtual Machine. In this optimization, compiled Java Virtual Machine code is modified at run time for better performance.

The optimization takes the form of a set of pseudo-instructions. These are variants of normal Java Virtual Machine instructions that take advantage of information learned at run time to do less work than the original instructions. The pseudo-instructions are distinguishable by the suffix *_quick* in their mnemonics.

It is important to understand that these pseudo-instructions are *not* part of the Java Virtual Machine specification or instruction set. They are invisible outside of a Java Virtual Machine implementation. However, inside a Java Virtual Machine implementation they have proven to be an effective optimization.

The technique documented in this chapter is covered by U.S. Patent 5,367,685.

Trial Exhibit 25. Page 401 of 488

The technique documented in this chapter is covered by U.S. Patent 5,367,685.

TX 25 at p. 389

US00RE38104E

(19) United States
(12) Reissued Patent
Gosling

(10) Patent Number: US RE38,104 E
(45) Date of Reissued Patent: Apr. 29, 2003

(22) Filed: Mar. 3, 1999

Related U.S. Patent Documents

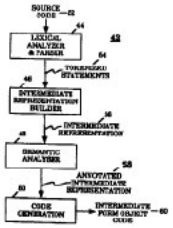
Reissue of:

(64) Patent No.: 5,367,685
Issued: Nov. 22, 1994
Appl. No.: 07/994,655
Filed: Dec. 22, 1992

5,201,050	A *	4/1993	McKeeman et al.	717/7
5,230,050	A *	7/1993	Iitsuka et al.	717/7
5,276,881	A	1/1994	Chan et al.	
5,280,613	A	1/1994	Chan et al.	
5,307,492	A *	4/1994	Benson	717/7
5,313,614	A *	5/1994	Goetelmann et al.	717/5
5,339,419	A	8/1994	Chan et al.	

main interpretation routines selectively invokes the two data reference handling routines depending on whether the data reference is an interaction in a symbolic or a numeric reference.

31 Claims, 5 Drawing Sheets



Copy provided by USPTO from the PIRS Image Database on 11/02/2010

Trial Exhibit 4015. Page 2 of 19

OAGOOGLE0000052236

TX 4015

Google Worried About Its Java Patent Infringement

From: Greg Stein. Sent:11/12/2006 10:48 PM.
To: [-] Andy Rubin.
Cc: [-] Chris DiBona; opensource-team.
Bcc: [-] .
Subject: Re: [Opensource-team] Fwd: Java News from Sun.

The news.com article notes that you can buy a non-GPL'd version.

On 11/12/06, Andy Rubin <arubin@google.com> wrote:
> I have been advised that Sun will offer a "link exception" to GPL so
> that you can link your app with their class libraries and not have
> the copy-left force you to open source your app. If they do not then
> it's a play for a dual license.
>
> They still have patents and trademarks.
>
>
> On Nov 12, 2006, at 10:30 PM, Chris DiBona wrote:
>
> > I have to admit I keep thinking "what's the catch", which means I'm a
> > little cynical about this. I hope it's just the regular GPL and they
> > don't try for additional attribution clauses. OSI is apparantly
> > considering allowing these within the aupices of "open source" as they
> > see it.
> >
> > I'm chatting with Mike Tiemann a
> >
> > Chris
> >
> > On 11/12/06, Greg Stein <gstein
> > On news.com: http://news.com
> > +code/2100-7344_3-6134584.h
> >>
> > On 11/12/06, Chris DiBona <cd
> > > FYI
> > >
> > > ---------- Forwarded message ----------
> > > From: Simon Phipps <Simon.Phipps@sun.com>
> > > Date: Nov 12, 2006 6:28 PM
> > > Subject: Java News from Sun
> > > To: Trusted-Friends@webmink.net
> > >
> > >
> > > Tomorrow, Sun will be making an announcement about the Java
> > platform
> > > that many of us have been anticipating eagerly. We wanted you to
> > hear
> > > it direct from us rather than through the press, so I'm sending you
> > > the news release that will go to the press tomorrow. If you have
> > any
> > > questions, please do not hesitate to contact me.
> > >
> > > Best regards
> > >
> > > Simon
> > >
> > >
> > >
> > > _____
> > >
> > > Simon Phipps, Chief Open Source & Free Software Officer, Sun
> > Microsystems

UNITED STATES DISTRICT COURT
NORTHERN DISTRICT OF CALIFORNIA
TRIAL EXHIBIT 155
CASE NO. 10-03561 WHA
DATE ENTERED _____
BY _____
DEPUTY CLERK

HIGHLY CONFIDENTIAL – ATTORNEY'S EYES ONLY Oracle America v. Google, 3:10-cv-03561-WHA GOOGLE-01-00025468

Trial Exhibit 155, Page 1 of 5

From: Andy Rubin
Sent: November 12, 2006

"They still have patents and trademarks."

TX 155 at p. 1

Google Worried About Its Java Patent Infringement

Andy Rubin
Co-Founder of Android
July 27, 2011
Dep. Tr. 16:4-16

Q. When you wrote "they still have patents and trademarks," what was in your mind about what patents Sun had?

A. 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.

So effectively you have to go back to Sun, ask them what they considered their intellectual property and, you know, try to figure out what the trick was if you wanted to use the technology.

Google Worried About Its Java Patent Infringement

From:	Andy Rubin. Sent:8/11/2007 4:56 PM.
To: [-]	Bob Lee.
Cc: [-]	Brian Swetland; Dan Bornstein.
Bcc: [-]	
Subject:	Re: [jc-user] New OpenJDK Community Technology Compatibility Kit License (TCK).

..and as far as GPL-ing the VM, everything that is linked with the VM would get infected.

The problem with GPL in embedded systems is that it's viral, and there is no way (for example) OEMs or Carriers to differentiate by adding proprietary works. We are building a platform where the entire purpose is to let people differentiate on top of it.

Finally, Sun has a different license for its library for SE and ME. The SE library is LGPL, ME library is GPL. That means anything that links with the ME library gets infected. And the SE library is not optimized for embedded systems.

Sun chose GPL for this exact reason so that companies would need to come back to them and take a direct license and pay royalties.

Tricky, no? Why would we want to do anything to support this behavior? We want to distance ourselves as much as possible from Sun.

- andy

PS -- we negotiated 9 months with Sun and decided to walk away after they threatened to sue us over patent violations

On Aug 11, 2007, at 3:47 PM,

> On 8/11/07, Bob Lee <crazybob@google.com> wrote:
>> I thought you guys might be interested in the reactions to Sun's
>> TCK licensing announcement
>> on the Java Champions mailing list. Sun's not fooling anyone. (The
>> list isn't public, so please
>> don't forward further.)
>
> I'm sure you agree that this is a totally unsurprising move by Sun.
>
>> Would it make sense for us to use Sun's Java library
>> implementation and license Dalvik under
>> the GPL? I think that would count as "substantial derivation" in
>> which case we'd get a TCK
>> license, right? :)
>
> I see the smiley, but in a word, no.
>

UNITED STATES DISTRICT COURT
NORTHERN DISTRICT OF CALIFORNIA
TRIAL EXHIBIT 230
CASE NO. 10-03561 WHA
DATE ENTERED:
BY:
DEPUTY CLERK

HIGHLY CONFIDENTIAL – ATTORNEY'S EYES ONLY Oracle America v. Google, 3:10-cv-03561-WHA GOOGLE-02-00020474

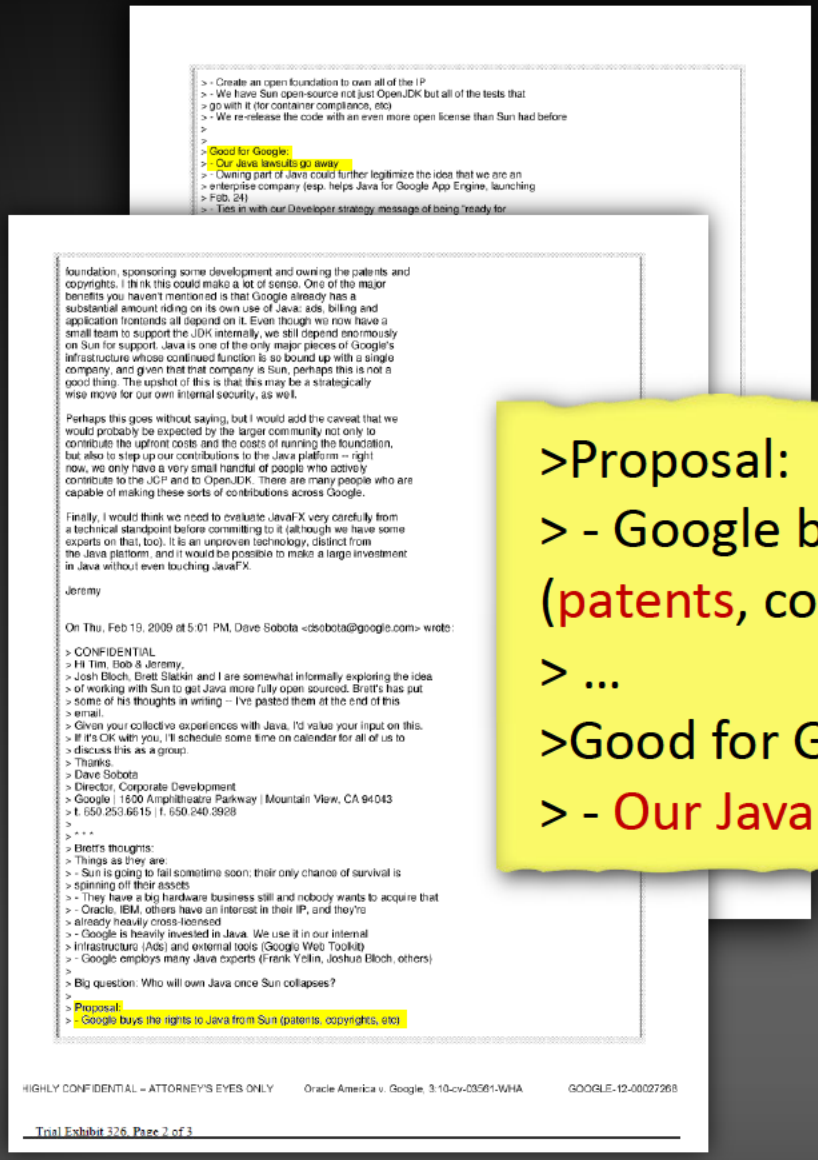
Trial Exhibit 230, Page 1 of 2

From: Andy Rubin
To: Bob Lee
Sent: August 11, 2007

"PS --we negotiated 9 months with Sun and decided to walk away after they threatened to sue us over patent violations"

TX 230 at p. 1

Google Worried About Its Java Patent Infringement



> - Create an open foundation to own all of the IP
> - We have Sun open-source not just OpenJDK but all of the tests that
> go with it for container compliance, etc)
> - We re-release the code with an even more open license than Sun had before
>
>
> Good for Google:
> - Our Java lawsuits go away
> - Owning part of Java could further legitimize the idea that we are an
> enterprise company (esp. helps Java for Google App Engine, launching
> Feb. 24)
> - Ties in with our Developer strategy message of being "ready for

foundation, sponsoring some development and owning the patents and copyrights. I think this could make a lot of sense. One of the major benefits you haven't mentioned is that Google already has a substantial amount riding on its own use of Java: ads, billing and application frontends all depend on it. Even though we now have a small team to support the JDK internally, we still depend enormously on Sun for support. Java is one of the only major pieces of Google's infrastructure whose continued function is so bound up with a single company, and given that that company is Sun, perhaps this is not a good thing. The upshot of this is that this may be a strategically wise move for our own internal security, as well.

Perhaps this goes without saying, but I would add the caveat that we would probably be expected by the larger community not only to contribute the upfront costs and the costs of running the foundation, but also to step up our contributions to the Java platform -- right now, we only have a very small handful of people who actively contribute to the JCP and to OpenJDK. There are many people who are capable of making these sorts of contributions across Google.

Finally, I would think we need to evaluate JavaFX very carefully from a technical standpoint before committing to it (although we have some experts on that, too). It is an unproven technology, distinct from the Java platform, and it would be possible to make a large investment in Java without even touching JavaFX.

Jeremy

On Thu, Feb 19, 2009 at 5:01 PM, Dave Sobota <dsobota@google.com> wrote:

> CONFIDENTIAL
> Hi Tim, Bob & Jeremy,
> Josh Bloch, Brett Slatkin and I are somewhat informally exploring the idea
> of working with Sun to get Java more fully open sourced. Brett's has put
> some of his thoughts in writing -- I've pasted them at the end of this
> email.
> Given your collective experiences with Java, I'd value your input on this.
> If it's OK with you, I'll schedule some time on calendar for all of us to
> discuss this as a group.
> Thanks.
> Dave Sobota
> Director, Corporate Development
> Google | 1600 Amphitheatre Parkway | Mountain View, CA 94043
> t. 650.253.6615 | f. 650.240.3928
>
> * * *
> Brett's thoughts:
> Things as they are:
> - Sun is going to fail sometime soon: their only chance of survival is
> spinning off their assets
> - They have a big hardware business still and nobody wants to acquire that
> - Oracle, IBM, others have an interest in their IP, and they're
> already heavily cross-licensed
> - Google is heavily invested in Java. We use it in our internal
> infrastructure (Ads) and external tools (Google Web Toolkit)
> - Google employs many Java experts (Frank Yellin, Joshua Bloch, others)
>
> Big question: Who will own Java once Sun collapses?
>
> Proposal:
> - Google buys the rights to Java from Sun (patents, copyrights, etc)

HIGHLY CONFIDENTIAL – ATTORNEY'S EYES ONLY Oracle America v. Google, 3:10-cv-03561-WHA GOOGLE-12-00027268

Trial Exhibit 326, Page 2 of 3

From: Dave Sobota
To: Tim Lindholm,
Bob Lee
Sent: February 19, 2009

>Proposal:
> - Google buys the rights to Java from Sun (patents, copyrights, etc)
> ...
>Good for Google:
> - Our Java lawsuits go away

TX 326 at p. 2

Google Purposely Avoided Finding Out Whether They Were Infringing Oracle's Patents

Andy Rubin
Co-Founder of Android
July 27, 2011
Dep. Tr. 19:20-20:8

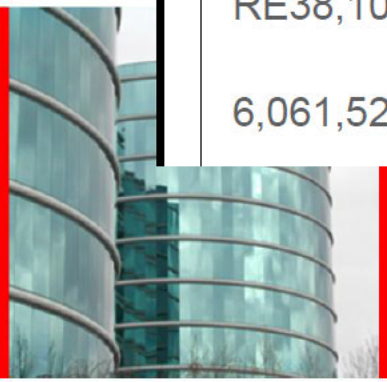
Q. To the best of your knowledge, between Google's acquisition of Android and the filing of the lawsuit in this action, did Google ever investigate Sun's, later Oracle America's patent portfolio as it might relate to Android?

A. Yeah, of the parts of Google that I manage and I operate, the -- there was no instruction to go investigate the breadth of Sun's patent portfolio.

Oracle Told Google It Infringed '104 And '520 Patents

Exemplary Oracle Patents Highly Relevant to Google

Patent	Short Description	Infringed By
RE38,104	Data References Resolution	Android Dalvik VM
6,061,520	Static Initialization	Android Dalvik VM



ORACLE

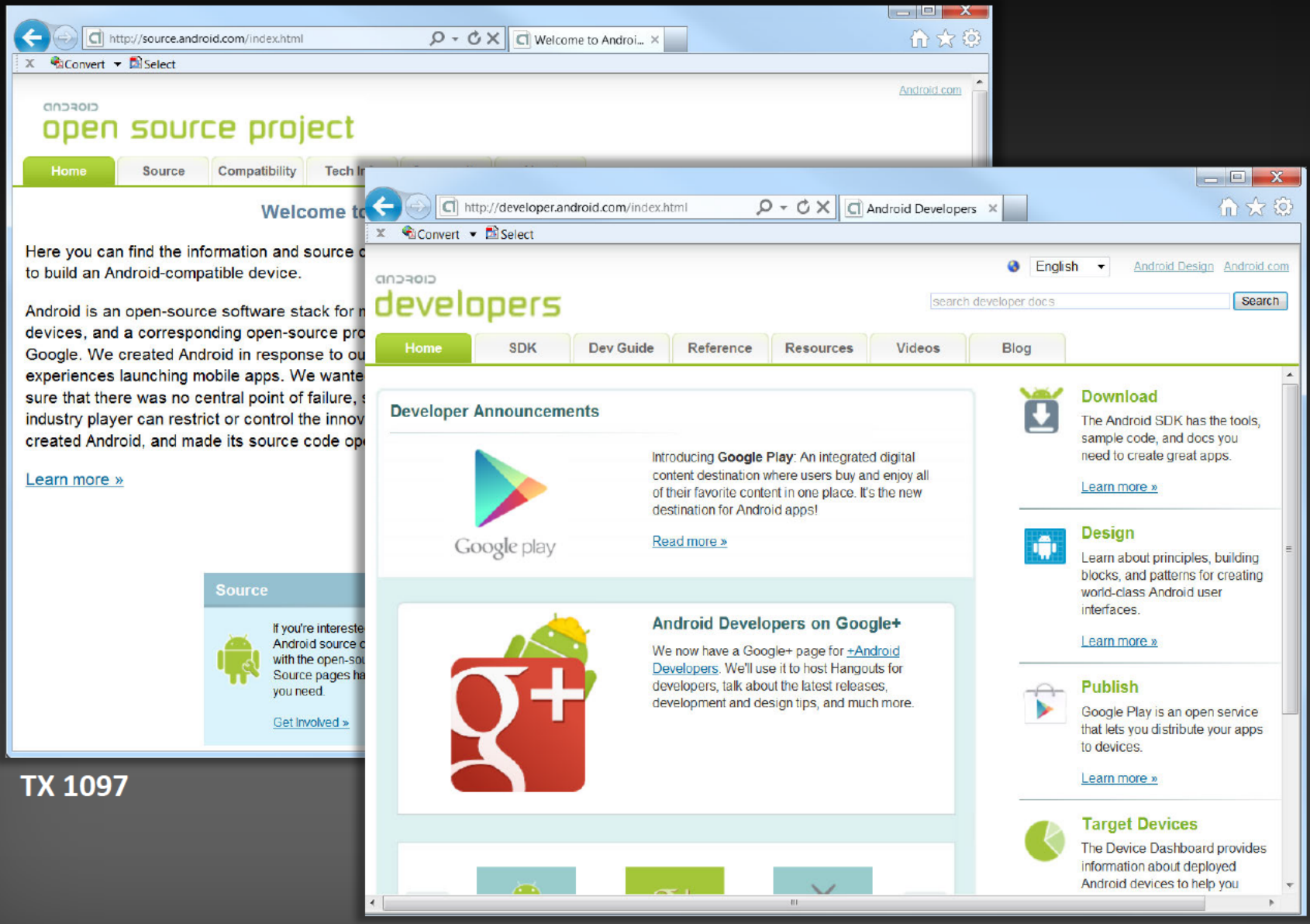
Oracle – Google Android Meeting

July 20, 2010

TX 920 (dated July 20, 2010)

Google Induces Patent Infringement By Phone Makers

Google Makes Android Available To Developers And Phone Makers



TX 1097

TX 1096

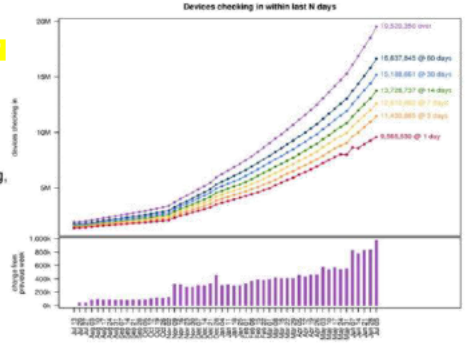
Google Induces Patent Infringement By Phone Makers

Android Activations in 2010:
"160K Android devices with Google services activated per day"

>20M Android Devices Activated

HIGHLIGHTS

- 20M Android devices sold to date (+82% Q/Q)
- 160K Android devices with Google services activated per day (+128% Q/Q)
- 70K apps in Android Market (+94% Q/Q)
- 32M daily SRP (+82% Q/Q vs. +14% overall mobile)
- 2.65 searches/day per 7-day active device
- $120M/year ads revenue run-rate (+47% Q/Q)
- $9.83/year ads revenue per 7-day active device
- New wave of OEMs launching strong products: Samsung, Sony Ericcson, LG
- All major operators (worldwide) committed to Android
- Decisive and successful transition of DTC
- Marketing: Android app videos on YouTube; Verizon Droid Apps $60M+ campaign, Sprint Evo, Samsung co-marketing



LOWLIGHTS

- Apple momentum continues with strong launches (iPad, iOS, iPhone 4) + iBooks
- Behind on music, video, books
- Market: Low rate of app purchases, policy issues
- Billing: Geographies, credit-card only, no in-app, no subscription
- Various competing app stores, and carrier walled gardens (JIL)
- Some Android devices shipping without Google search

- Device numbers are "all-time check-ins"; data as of 6/30/2010. Data from go/android-stats
- As of Mar-2010: 29 devices across 62 carriers, 50 countries and 19 languages
- Revenue/Searches data from mobile dashboard

TX 1061 at p. 6

Android activations in 2012: 750,000 Per Day

Dan Morrill,
Technical Program Manager
for Android Compatibility
April 26, 2012
Trial Tr. 1017:4-16

Clean Room Is No Defense To Patent Claims

Andy Rubin
Co-Founder of Android
July 27, 2011
Dep. Tr. 249:5-23

Q. And we've talked about your clean room, the way the clean room got established and how you communicated the rules of the clean room.

One thing we didn't talk about specifically in the context of clean room is patent issues. So did you have an understanding that the clean room would bear on the question of whether Android would infringe Sun/Oracle America patents?

A. No. Generally speaking, a clean room approach doesn't protect against patents. There's no expectation.

As I said previously, VM technology has been around forever. I didn't think the stuff that we were doing was going to be a violation of anybody's IP.

Q. And so that was based without reviewing the IP? That was kind of an intuition on your part?

A. That's correct. It was -- you know, it was my business judgment.

Google Has No Excuses

- Patented inventions are not “free”
- Google has no fair use defense
- The truth is in Google’s own source code
- Google needs a license to use patents