UNITED STATES DISTRICT COURT
DISTRICT OF MASSACHUSETTS
EASTERN DIVISION

RED BEND LTD. and RED BEND SOFTWARE INC., Plaintiffs, v. GOOGLE INC., Defendant.	CIVIL ACTION NO. 09-cv-11813
GOOGLE INC., Counterclaim-Plaintiff, v. RED BEND LTD. and RED BEND SOFTWARE INC., Counterclaim-Defendants.	

DECLARATION OF MARTIN G. WALKER
IN SUPPORT OF GOOGLE'S CLAIM CONSTRUCTION BRIEF

I, Martin G. Walker, Ph.D., declare:

I have been retained as an expert by the law firm Bingham McCutchen LLP, counsel of record for Google Inc. (hereafter "Google") in the above-captioned matter. This declaration is based on my personal knowledge and experience, as well as my investigation in this matter, and reflects my expert opinions on certain issues to which I may testify.

I. EDUCATION AND EXPERIENCE.

1. My CV (Exhibit A) contains an overview of my thirty years of experience in the high technology industry in Silicon Valley as well as a list of my recent

publications. I received a BSEE from the Massachusetts Institute of Technology in 1973, MSEE from Stanford University in 1976, and a Ph.D. in electrical engineering from Stanford University in 1979. During my career, I founded Analog Design Tools, and served as President, a Member of the Board of Directors, and as Chief Scientist. Later, I founded Symmetry Design Systems, Inc., and served as a Board Member and Executive Vice President. Next I founded Frequency Technology, and served as CEO (later Executive Vice President), and a Member of the Board of Directors. I have also served as the CTO of Knowledge Networks, an internet based consumer market research startup.

2. During my career, I have been responsible for design and development of several complex software programs. Additionally, to support my research activity at Stanford University, I wrote a special purpose program to aid in debugging a computer system that I designed. This program included a component called a "dis-assembler" that examines executable programs and identifies the individual instructions that make up the program. I have gained direct industry experience understanding and managing software programs of the type and complexity of the software program at issue. As part of my management of software development, I also became familiar with the process of providing software updates to end-users and the process of creating and applying update patches in particular.

3. Recently, I have undertaken numerous consulting assignments that involved evaluating computer software code in the context of litigation. For instance, I have analyzed software code to determine if the code practiced cited methods of patents. Thus I have a personal understanding of the normal standards in the industry regarding

analysis of source code in litigation contexts. Additionally, I have testified in patent litigation contexts.

III. OVERVIEW OF ASSIGNMENT AND OPINIONS EXPRESSED.

4. I understand that Plaintiffs in the above matter ("Red Bend") have accused Google's "Courgette" software of infringing certain claims of Red Bend's U.S. Patent No. 6,546,552 ("the '552 patent"). I have previously submitted a declaration to the Court in support of Google's Opposition to Red Bend's Motion for Preliminary Injunction, in which I discussed the non-infringement and invalidity of the '552 patent. As part of my investigation in this case, I have formed an opinion regarding the proper construction of certain terms used in the asserted claims of the '552 patent.

5. This declaration presents my findings at this time after having reviewed the claims asserted by Red Bend and the underlying evidence.

6. I am generally familiar with the claim construction analysis process through prior engagements I have had in patent infringement actions. I have been informed by counsel that claims are to be construed from the perspective of one of ordinary skill in the art at the time of the invention. I further understand that the meaning of claim terms may be understood based on the language of the claims themselves, the patent specification and the prosecution history. Additionally, other evidence (such as dictionaries) may also be used to determine the meaning of the words to a person of ordinary skill in the art at the time of the invention. I have also been informed that it is not necessary to rely on extrinsic evidence if the meaning of the claims is clear from the intrinsic evidence.

7. In my opinion, having read the '552 patent and the prosecution history, I believe that one of ordinary skill in the art would understand the meaning of the claims and terms in dispute to require reference to the terms as they were defined by the inventor in the Glossary section of the '552 patent, and as they were further refined or clarified during prosecution of the patent. I have summarized here the key aspects of the patent and the events that occurred during prosecution that lead me to conclude that the terms in dispute should have the meanings stated below:

Claim term	**Google's Claim Construction**
Compact difference result	A difference result in which references that have changed due to delete/insert modifications do not appear.
Data table	A table of entries, each of which may have a different size. An executable program is one example of a data table. It cannot be source or other symbolic code.
Executable program	A program comprising machine language instructions and corresponding bytes of data used by the program that are ready to be run on a computer, excluding source or other symbolic code.
Invariant references	Values made the same in the modified old and new programs (or data tables) for corresponding reference entries so that the reference addresses are excluded from the difference result.
Modified old program Modified new program Modified new data table Modified old data table	A version of the actual program or data table in its original executable form, with certain portions replaced.

8. I understand that, in addition, the parties have stipulated that certain claim terms should be construed as follows:

Claim term	**Stipulated Claim Construction**
Difference result	Data representative of the difference between an old program (or data table) and a new program (or data table) used to carry out an update of the old program.
Old data table	A data table (or portion of a data table) that is to be updated.

Claim term	Stipulated Claim Construction
Old executable program	An executable program (or portion of an executable program) that is to be updated. Synonymous with "said old program."
Reference	A part of the data appearing in an entry in the data table which is used to refer to some other entry from the same data table. A reference can be either an address or a number used to compute an address.
Reference entries	Entries that include references. For this purpose, "entries" are addressable units of a data table or program. As an example, a reference entry could be an individual machine instruction (such as an individual jump instruction) specifying a target address.

V. PERSON OF ORDINARY SKILL IN THE ART.

9. I agree with Red Bend's expert, Dr. Edwards, that a person of ordinary skill in the art is someone with a bachelor of science degree in computer science (or an equivalent), who has approximately two years of software development experience, and some understanding of how source code is converted to machine instructions to be executed on a computer.

VI. THE '552 PATENT.

10. I have reviewed the '552 patent and its prosecution history. Based on my review, I find the following aspects of the '552 patent most important for purposes of this case.

11. The '552 patent relates to a very specific process for creating and distributing software upgrades to old programs. When a new program is created updating an old program, the '552 patent teaches using a specific process for capturing the differences between the old and the new program in a compact difference result which can then be distributed as a patch.

12. I believe that the following aspect of the ‘552 patent is key and fundamental to the process described:

> The present invention is based on the observation that the relatively large size of the difference result stems from the alterations of reference in reference entries as a result of other newly inserted entries (and/or entries that were deleted).
>
> On the basis of this observation, the invention aims at generating a modified old program and a modified new program, wherein the difference in references in corresponding entries in said new and old programs as explained above, will be reflected as invariant entries in the modified old and new programs. *The net effect is that the invariant reference entries (between the modified old program and the modified new program), will not appear in the difference result, thereby reducing its size as compared to a conventional difference result obtained by using hitherto known techniques.*

This quote is taken from the ‘552 patent at column 3, lines 31 - 46. Red Bend’s counsel and Dr. Edwards also have relied on this passage as describing a key aspect of the claimed invention of the ‘552 patent.

13. In essence (as set forth above), ‘the 552 patent is based on the fact that differences between an old and a new executable program are largely caused by alterations of references (*i.e.*, addresses) that are incidentally changed in reference entries (*i.e.*, instructions that refer to addresses) because of newly inserted (or deleted) instructions forming the transition from the old to the new program. The patent solves this problem by replacing corresponding addresses in the new and old program that changed only due to address shifting from insert/delete modifications with “invariant references.” The replacement is made in a modified old and a modified new program. Thereafter, the modified old and new programs are run through a difference generator

and the address changes due only to insert/delete are "neutralized" because of the invariant references assigned and disappear from the difference result. *See* '552 patent at 1:59-2:9, 3:27-3:46, 10:3-15 and 14:65-15:8.

14. In addition to the patent, the prosecution history also emphasizes a process of identifying insert/delete modifications and assigning invariant references to the affected addresses in the old and new program so that they disappear from the difference result:

> A major problem arises when applying these [prior art diff] methods to executable program files . . . . The problem arises from the fact that executable programs are generated from sources and in that process many references are inserted into these executable files. These references do not refer symbolically to other location of the program, as may be the case in source files, but they refer to addresses - sequential locations in the program file. . . .
>
> Consider, for example, an extreme example where a change in the first source of line [sic.] may lead to actual change of some first executable file, in which few bytes were added but also all references must change since they refer to locations that now have been moved farther for the amount of bytes added at the beginning. To simply reflect all the changed references when computing a difference, one must include them all. In accordance with the present application such a need is reduced or eliminated, and what is required, is just to send the first few modified bytes . . . . The modification is effected in such as way that the references become "invariant" . . . .

Declaration of Susan Baker Manning in Support of Google Inc.'s Claim Construction Brief ("Manning Decl."), Ex. C at RedBend 0000150.

15. The patent also is clear that it applies to forming difference results on executable programs. '552 patent at 2:60-62; Declaration of Susan Baker Manning in

Support of Google Inc.'s Claim Construction Brief ("Manning Decl."), Ex. C at RedBend 0000149, RedBend 0000151-155.

16. The prosecution history further clarifies that in extracting the difference between two versions of executable files as defined in independent Claim 1, "there is no source involved, and neither statements, not any textual or other symbolic representation of the program even exist." *See* Manning Decl., Ex. C at RedBend0000151. The applicant makes the same argument relative to all claims of the patent, regardless of whether the claim uses the terms "executable program" or "data table" to describe on what the difference is being generated.

> Claims 35 to 68 are basically similar to claims 1 to 34, respectively, except for the fact that they recite data table instead of executable program. Data table is discussed on page 4, line 9 of the application and do not embrace source code as in Okuzumi. It is accordingly submitted that Claims … 42-44 [and] 55-57 are not anticipated by Okuzumi for the reasons discussed in detail above with reference to claims 1 to 3, 8-10. . . and 21-23.

Id. at RedBend0000154 (emphasis added).

17. The specification of the '552 Patent also emphasizes the importance of replacing "substantially each" reference entry in the old and new programs. For instance, in the section entitled "SUMMARY OF THE INVENTION," the description of the invention includes "scanning the old program and for substantially each reference entry perform" performing certain steps (*see* '552 Patent at 3:53-54). It is significant that this same or similar language is repeated more than 20 times in the specification. Thus addressing "substantially all" of the reference entries should be thought of as an important part of the invention described in the '552 Patent.

18. The asserted claims of the ‘552 Patent can be broken into “program” claims (8-13, 21-25, 28-31, 32-34) and parallel “data table” claims (42-47, 55-60, and 62-68). Other than the difference in terminology between the words “program” and “data table,” the parallel asserted claims are effectively identical to one another.

A. “Executable program”

19. The term “executable program” is properly construed as “a program comprising machine language instructions and corresponding bytes of data used by the program that are ready to be run on a computer, excluding source or other symbolic code.”

20. The term “executable” in the ‘552 patent refers to a type of program. In particular, an executable program is “a loaded program in machine-memory” or “an executable-file.” ‘552 patent at 2:61-65.

21. A person of ordinary skill in the art would understand that for a program to be loaded in machine memory, it would comprise machine language instructions and corresponding bytes of data that are ready to be run on the chipset specific to that particular computer; as opposed to symbolic code such as source code.

B. “Data table”

22. The term “data table” should be defined as “A table of entries, each of which may have a different size. An executable program is one example of a data table. It cannot be source or other symbolic code.”

23. The specification of the ‘552 patent defines a “data table” in the “Glossary” as “a table of entries, each may have a different size.” *See* ‘552 patent at

2:33-34. The specification later explains "[a]n executable program is one example of a data table." *Id.* at 2:61.

24. The "data table" of the claims cannot include source or other symbolic code because during prosecution of the '552 patent in the Response to Office Action at RedBend0000151, Red Bend disclaimed source code and other symbolic code for all claims, both the "data table" claims and the "executable program" claims. The applicant stated, "In extracting [a] diff between 2 versions of executable files … there is no source involved, and neither statements, not any textual or other symbolic representation of the program even exist". *Id.; see also* Manning Decl. Ex. C at RedBend0000154 ("Claims 35 to 68 are basically similar to claims 1 to 34, respectively, except for the fact that they recite data table instead of executable program. *Data table is discussed on page 4, line 9 of the application and do not embrace source code as in Okuzumi.* It is accordingly submitted that … 42-44 … 55-57 are not anticipated by Okozumi for the reasons discussed in detail above with reference to claims 1 to 3, 8-10. . . .") (emphasis added).

C. "Modified old data table;" "Modified old program;" "Modified new data table;" "Modified new program"

25. The asserted independent claims each use the term "modified" similarly. *See* claims 8, 21, 42 and 55. In the specification, "step e" describes "replacing" address references associated within certain reference entries of a program with label marks. *See* '552 patent at 12:11-28. Furthermore, the Abstract describes "replacing the reference of the entry by a distinct label mark, whereby a modified old [or new] program is generated." *See id.* at 3:53-62; *see also* 4:8-19 (describing replacement of references in reference entries to create the modification).

26. In the prosecution history, the applicant argued that the modified old and new programs (or data tables) remain in executable format after modification. *See* RedBend0000151. During prosecution, the applicant also limited modifications to replacement of "the reference of said entry by distinct label mark" and not "the generation of auxiliary index data structure." *Id.* at RedBend0000173 (arguing the applicant "[does] not agree with the Examiner's contention that Miller creates modified old programs in the sense of the invention."). The applicant further distinguished Miller because "in addition to an old file, an index or hash table is created [] so as to facilitate searching for character strings from the new file." *Id.*

27. A hash table lists index values and corresponding data. In short, the table data may be retrieved by value as well as by using a pointer into the table.

D. "Compact difference result"

28. "Compact difference result" is properly construed as "a difference result in which references that have changed due to delete/insert modifications do not appear."

29. The summary of the '552 patent explains that "the invention aims at generating a modified old program and a modified new program, wherein the difference in references in corresponding entries in said new and old programs as explained above, will be reflected as invariant entries in the modified old and new programs. The net effect is that the invariant reference entries (between the modified old program and the modified new program), will not appear in the difference result, thereby reducing its size." '552 patent at 3:31-46. Later, the patent explains that "[t]he modification in the reference . . . is caused by the insertion of entries . . . it is desired to neutralize this

change, since it has occurred solely due to the fact that other entries have been affected . . . modifications of this kind will be modified to invariant references with the obvious consequence that they are not reflected in the difference result, thereby keeping the latter relatively compact." *Id.* at 9:66-10:15.

30. During prosecution of the '552 patent, the applicant further explained "pre-processing is applied to both old and new files (giving rise to modified old and modified new programs) and applying a diff extraction utility to the modified old and modified new programs. The modification is effected in such a way that references become 'invariant' (see page 5, lines 22 to 29), thereby reducing the diff result and rendering it compact." Manning Decl., Ex. C at RedBend0000150. The pages referenced by the applicant now correspond to the '552 patent specification at 3:36-46, quoted in paragraph 29 above.

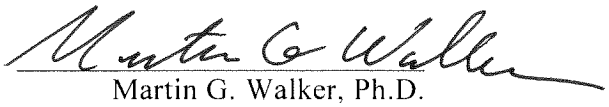
E. "Invariant references"

31. "Invariant references" are "values made the same in the modified old and new programs (or data tables) for corresponding reference entries so that the reference addresses are excluded from the difference result."

32. *See* paragraphs 29 - 30 above.

I declare under penalty of perjury that the foregoing is true and correct.

Executed on: July 15, 2010 in San Francisco, CA.



Martin G. Walker, Ph.D.

Certificate of Service

I hereby certify that this document filed through the ECF system will be sent electronically to the registered participants as identified on the Notice of Electronic Filing (NEF) and paper copies will be sent to those indicated as non-registered participants, by federal express, on July 15, 2010.

/s/ David M. Magee
David M. Magee