

# **EXHIBIT A**

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

In re Reexamination of	)	<b>MAIL STOP: <i>Ex Parte</i></b>
	)	<b><i>Reexamination</i></b>
U.S. Patent No.: 6,546,552	)	
	)	Group Art Unit: 3992
Sharon Peleg	)	
	)	Examiner: Andrew L. Nalven
Issued: April 8, 2003	)	
	)	Confirmation No.: 4316
Reexam Control No.: 90/009,670	)	
	)	
For: DIFFERENCE EXTRACTION	)	
BETWEEN TWO VERSIONS OF	)	
DATA-TABLES CONTAINING	)	
INTRA-REFERENCES	)	

**SUBSTANCE OF THE INTERVIEW**

The patent owner ("Owner") thanks Examiners Andrew Nalven, Eric Keasel, and Albert Gagliardi for kindly extending the courtesies of conducting a personal interview on August 11, 2010 with Owner's representatives Charles Wieland and Michael Weinberg. Owner's representative also thanks Examiner Nalven for the Interview Summary of August 11, 2010, which includes the statement that the "Examiner agrees that Wetmore does not teach the claimed invariant references or distinct label marks as claimed and defined by the specification."

In the interview, Owner's representatives explained distinguishing features of the claims of U.S. Patent 6,546,552 ("the '552 patent") over the cited U.S. Patent 5,481,713 ("Wetmore"), with reference to the Supplemental Response filed June 29, 2010 ("Response").

Initially, Owner's representatives discussed Wetmore, in particular how it is directed to a "vectorization" process that operates on compiled and/or assembled object files before linking of the object files takes place. Wetmore describes vectorization as a way to introduce a level of indirection to the resulting ROM-stored binary file so that when entering an external routine such as a subroutine or function call, program execution does not flow sequentially -- but is first redirected to one of several vector tables in RAM which will contain the address of the next line of code that should be executed.

Owner's representatives also emphasized that Wetmore does not utilize a binary difference result algorithm, but instead describes a far less granular comparison of the object files of two versions of vectorized ROM code to identify routines which are different or new. Accordingly, Wetmore permits updates of software by transmitting entire routines to a client, which are executed from RAM instead of obsolete routines in ROM. Notably, this would result in the transmission of large difference results, because, if even a single line of binary code were added to a routine, the entire routine would need to be included in the update of Wetmore.

Owner's representatives followed the discussion of Wetmore with a discussion of distinctions found in embodiments of the '552 patent, in particular how the '552 patent is not concerned with updating programs previously stored in a ROM, and does not require changes to the developer's compilation tool-chain in order to function. Owner's representatives also noted the inclusion of a glossary starting at column 2 of the '552 patent. The glossary, for example, defines a reference as "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. Entries that include references are designated also as reference entries."

Next, Owner's representatives discussed claim 8, a representative of "invariant reference" claims 8, 11, 21, 24, 28, 42, 45, 55, 58, and 62. Owner's representatives emphasized the points made on pages 8-15 of the Response, including that (1) the inputs to Wetmore are not executable, (2) no "references" in the Wetmore document are reflected as invariant, (3) Wetmore fails to disclose reflecting "substantially each" reference invariant, and (4) Wetmore fails to disclose generation of a "compact difference result".

After the discussion of claim 8, Owner's representatives discussed claim 1, a representative of "distinct label mark" claims 1, 4, 14, 17, 26, 35, 38, 48, 51, and 61. Owner's representatives emphasized the points made on pages 15-17 of the Response, including that (1) the inputs to Wetmore are not executable, (2) no "references" in the Wetmore document are replaced by a distinct label mark, (3) Wetmore fails to disclose "for substantially each reference entry ... replacing the

reference by a distinct label mark<sup>1</sup>, and (4) Wetmore fails to disclose generation of a "compact difference result".

In discussing the point that the inputs to Wetmore are not executable, Owner's representatives emphasized that the object files of Wetmore include unresolved, symbolic references which are not numeric, rendering the object files not executable. Examiner Nalven asked about the distinction between symbolic references of object files and numerical references of executable programs. Owner's representatives explained that a symbolic reference does not point to any address until resolved. Further, it is noted herein that a related discussion appears in the remarks dated May 21, 2002 during prosecution of the application that issued as the '552 patent. For the Examiner's reference, an annotated excerpt from these remarks is provided below:

General diff extraction utilities (see page 2, lines 14 and 15 [col. 1, lines 22-25 of the '552 patent]) were originally designed for extracting differences between two versions of a source program (referred to also as 'program source' or as 'source'). Sources are the original text files that programmers write in some formal language known as 'programming language'. Such sources are purely symbolic in the sense that they do not mention actual physical location of other elements in the source nor their absolute sequential location. Rather, any of required references in a source are made through symbolic names which themselves are part of the source.

Such diff utilities and methods are known in the art and known to be sufficient for extracting differences of such source files. A major problem arises when applying these methods to executable program files aiming to extract the difference at the byte-level, regarding the files as a list of data bytes rather than list of text lines. 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. When

---

<sup>1</sup> As mentioned at the interview, the undersigned noted an incomplete thought at page 16 of Response. Specifically, the third paragraph should have begun "First, Wetmore causes modifications to entry points, not replacing references with distinct label marks through vectorization of the old and new object files." This is clear from Fig. 3 of Wetmore, where the "Reference to Entry Point" is not vectorized. Without vectorization, the executable code would be immediately following the entry point 304; vectorization results in the location of entry point 304 being modified to be a reference to a table pointer 305 in RAM.

regenerating an executable file from a modified source file, not only the actual changes are being reflected in the executable file, also the majority if not all the inserted references are modified as well since their referred addresses have changed their location in the executable file as a result of the actual changes. This phenomenon leads to a significant increase in the amount of differences. This problem is discussed in the "Background of the Invention" section of the specification, Page 2, lines 11 to 18 [*col. 1, lines 18-30 of the '552 patent*], and exemplified on Page 2, lines 18 to 30 [*col. 1, lines 31-46 of the '552 patent*]. (Emphasis Added.)

Owner's representatives also referred to Table 2 in column 8 of Wetmore in explaining how no references in MaxBlock, which the Examiner considers to be the "old program", are replaced, or reflected as invariant, in a modified old and modified new program. Rather, in performing the vectorization of Wetmore, the routine MaxBlock is not changed as shown in Table 2; only the entry name label of the routine is changed. The label of the routine does not constitute a reference, but is instead used as a symbol to be referenced by other code. According to Wetmore, the original label of MaxBlock is used to label the instruction "jmp ([ $\$0584$ ])", which will jump to an address contained at the address  $\$0584$  in the RAM. Absent any patches, the location  $\$0584$  will contain the address to the label `_v_MaxBlock`. Examiner Nalven appeared to acknowledge this difference between Wetmore and the claims of the '552 patent.


**Conclusion**

Owner looks forward to receiving a Notice of Intent to Issue a Reexamination Certificate in due course. In the event that there are any questions concerning the Response, or the '552 patent in general, the Examiner is respectfully requested to telephone the undersigned so that reexamination proceeding may be expedited.

Respectfully submitted,

BUCHANAN INGERSOLL & ROONEY PC

Date: August 16, 2010

By:   
\_\_\_\_\_  
Charles F. Wieland III  
Registration No. 33096

**Customer No. 21839**  
703 836 6620

**CERTIFICATE OF SERVICE**

It is hereby certified by the undersigned that a true copy of the Substance of Interview filed on August 16, 2010 was transmitted via e-mail to:

Jonathan A. David, Esq.  
Lerner, David, Littenberg, Krumholz & Mentlik, LLP  
600 South Avenue West  
Westfield, New Jersey 07090

Respectfully submitted,

BUCHANAN INGERSOLL & ROONEY PC

Date: August 16, 2010

By:



Charles F. Wieland III  
Registration No. 33096

**Customer No. 21839**  
703 836 6620