Exhibit 20 Filed Under Seal Pursuant to Protective Order

EXHIBIT 21

SNELL & WILMER LLP
Alan L. Sullivan (3152)
Todd M. Shaughnessy (6651)
15 West South Temple
Gateway Tower West
Salt Lake City, Utah 84101-1004
Telephone: (801) 257-1900
Facsimile: (801) 257-1800

CRAVATH, SWAINE & MOORE LLP
Evan R. Chesler (admitted pro hac vice)
David R. Marriott (7572)
Worldwide Plaza
825 Eighth Avenue
New York, New York 10019
Telephone: (212) 474-1000
Facsimile: (212) 474-3700

Attorneys for Defendant/Counterclaim-Plaintiff
International Business Machines Corporation

IN THE UNITED STATES DISTRICT COURT

FOR THE DISTRICT OF UTAH

THE SCO GROUP, INC. Plaintiff/Counterclaim-Defendant, -against- INTERNATIONAL BUSINESS MACHINES CORPORATION, Defendant/Counterclaim-Plaintiff.	**DECLARATION OF RANDALL DAVIS** Civil No. 2:03CV-0294 DAK Honorable Dale A. Kimball Magistrate Judge Brooke C. Wells

I. INTRODUCTION

1. I am a professor of Computer Science at the Massachusetts Institute of Technology in Cambridge, Massachusetts. Addendum A provides more details of my technical background and experience, a list of publications, and a list of cases in which I have testified or been deposed. I received my undergraduate degree in Physics from Dartmouth College in 1970 and a Ph.D. in Computer Science from Stanford in 1976.

2. I have published some 50 articles on issues related to artificial intelligence and have served on several editorial boards, including *Artificial Intelligence, AI in Engineering,* and the MIT Press series on AI. I am a co-author of *Knowledge Based Systems in AI.*

3. In recognition of my research in artificial intelligence, I was selected in 1984 as one of America's top 100 scientists under the age of 40 by *Science Digest.* In 1986 I received the *AI Award* from the Boston Computer Society for contributions to the field. In 1990 I was named a Founding Fellow of the American Association for AI and in 1995 was elected to a two-year term as President of the Association. From 1995-1998 I served on the Scientific Advisory Board of the U.S. Air Force.

4. In addition to my work with artificial intelligence, I have also been active in the area of intellectual property and software. Among other things, I have served as a member of the Advisory Board to the US Congressional Office of Technology Assessment study on software and intellectual property, published in 1992 as *Finding a Balance: Computer Software, Intellectual Property, and the Challenge of Technological Change.* I have published a number of articles on the topic, including co-authoring an

article in the *Columbia Law Review* in 1994 entitled "A Manifesto Concerning Legal Protection of Computer Programs" and an article in the *Software Law Journal* in 1992 entitled "The Nature of Software and its Consequences for Establishing and Evaluating Similarity."

5. From 1998-2000 I served as the chairman of the National Academy of Sciences study on intellectual property rights and the emerging information infrastructure entitled *The Digital Dilemma: Intellectual Property in the Information Age*, published by the National Academy Press in February 2000.

6. I have been retained as an expert in over thirty cases dealing with alleged misappropriation of intellectual property, such as the allegations raised in this case, and have done numerous comparisons of code. I have been retained by plaintiffs who have asked me to investigate violations of intellectual property, by defendants who have asked me to investigate allegations made against them, and by both sides to serve as the sole arbiter of a binding arbitration.

7. In 1990 I served as expert to the Court (Eastern District of NY) in *Computer Associates v. Altai*, a software copyright infringement case that articulated the abstraction, filtration, comparison test for software. I have also been retained by the Department of Justice on its investigation of the INSLAW matter. In 1992 (and later in 1995) my task in that engagement was to investigate alleged copyright theft and subsequent cover-up by the Federal Bureau of Investigation, the National Security Agency, the Drug Enforcement Agency, the United States Customs Service, and the Defense Intelligence Agency.

II. ASSIGNMENT/SUMMARY OF FINDINGS

8. I have been asked by counsel for IBM to examine the 198 Items in SCO's December 22, 2005 Disclosure of Material Allegedly Misused by IBM (the "Final Disclosures") that are challenged by IBM in its Motion to Limit SCO's Claims Relating to Allegedly Misused Material ("IBM's Motion"). Specifically, I have been asked to (1) determine the extent to which SCO has specified its claims, by identifying versions, files and lines of code with respect to each of the items; and (2) describe the effort that would be required to evaluate SCO's allegations based on the level of specificity that it has provided.

9. In summary, SCO fails specifically to identify lines of System V, AIX or Dynix, and Linux material with respect to any of the 198 Items. As a result, it is impossible fully to evaluate SCO's claims.

III. ANALYSIS

10. In its Final Disclosures, SCO identifies 294 Items of allegedly misused material, including the 198 Items at issue in IBM's motion. I have reviewed the 198 Items to consider the extent to which they describe SCO's claims with specifity.

11. I conclude that SCO has failed to identify with specificity any of the 198 Items. SCO does not provide a complete set of reference points (version, file and line) for any of the 198 Items, which makes it practically impossible fully to evaluate SCO's claims.

12. As shown in Addendum B, SCO does not specifically identify lines of System V, AIX or Dynix, and Linux material for any of the 198 Items. SCO does not identify

with specificity System V, AIX, or Dynix version(s) or file(s) with respect to more than a few of the Items. Specific versions and files of Linux code are omitted with respect to many of the Items.

13. In its memorandum in opposition to IBM's preclusion motion, SCO tells the Court that it has provided "color coded illustrations", "line by line source code comparisons" and "over 45,000 pages of supporting materials". However, tens of thousands of those pages concern Item 294, which SCO expressly abandons in its opposition brief. While the Final Disclosures include color-coded illustrations and line-by-line source comparisons, they either do not do so with regard to any of the 198 Items at issue or the materials provided do little to particularize SCO's claims.

14. Absent more specific information about SCO's claims, an extraordinary effort would be required to evaluate the claims. In fact, based on the information SCO has provided, it would be impossible fully to evaluate SCO's claims without considering the entire universe of potentially relevant code.

15. SCO's failure to specify its claims puts on IBM the impossible burden of looking for undefined needles in an enormous haystack. The multiple versions of Unix, AIX, Dynix, and Linux comprise more than 1 billion lines of code.

16. The size of the haystack is only part of the problem. With enough time, IBM would likely be able to search the haystack for the allegedly misused material, although I note that SCO's Mr. Sontag testified that it would take 25,000 man years to compare a single version of Linux (a mere 4,000,000 lines of code) to a single version of Unix.

17. The true difficulty with the Items at issue is that SCO does not describe the needles it is sending IBM to find. Instead of defining the 198 items at issue by providing version, file and line information, SCO describes them generally and imprecisely. As a result, the needles look just like hay. This suggests that SCO does not know what it claims or is hiding what it claims.

18. To take just one example (of many), in Item 146, SCO identifies IBM's "Use of Dynix/ptx for Linux development" by reference to an email that concerns "performance and profiling" and lists 11 Linux files without mentioning which versions of Linux these files come from.

19. This provides no meaningful information about what IBM is alleged to have done wrong. SCO does not say where such "profiling" was done in System V or Dynix or even where specifically it is allegedly done in Linux. Absent more information, it is practically impossible for IBM to conduct a proper investigation to fully defend itself.

20. I understand, and for this purpose assume, that SCO's claims require inquiry into, among other things, the origin of the code and concepts (which are, of course, embodied in code), the value of the code, whether SCO distributed the code under the GPL, whether it was developed to comply with publicly known standards, whether the code is dictated by externalities, whether the code is merely an unprotectable idea, whether the code ever shipped without a required copyright notice and whether the code is otherwise in the public domain. These questions must be answered on a line by line basis. And that cannot be done properly without knowing which versions, files and lines are at issue.

IV. SUMMARY

21. SCO has failed to provide the most basic information relating to the 198 Items at issue in IBM's motion. SCO has declined, as a practical matter, to tell IBM what is in dispute. SCO's failure to specify its claims puts on IBM the impossible burden of searching an enormous haystack for needles that look just like hay.

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

Randall Davis

Date: 29 March 2006

Place: Taipei, Taiwan

CERTIFICATE OF SERVICE

I hereby certify that on the 4th day of April, 2006, a true and correct copy of the foregoing was sent by U.S. Mail, postage prepaid, to the following:

Brent O. Hatch
Mark F. James
HATCH, JAMES & DODGE, P.C.
10 West Broadway, Suite 400
Salt Lake City, Utah 84101

Stephen N. Zack
Mark J. Heise
BOIES, SCHILLER & FLEXNER LLP
100 Southeast Second Street, Suite 2800
Miami, Florida 33131

Robert Silver
Edward Normand
BOIES, SCHILLER & FLEXNER LLP
333 Main Street
Armonk, New York 10504

/s/Todd M. Shaughnessy

ADDENDUM A

Addendum A

RANDALL DAVIS

Randall Davis received his undergraduate degree from Dartmouth, graduating summa cum laude, Phi Beta Kappa in 1970, and received a PhD from Stanford in artificial intelligence in 1976.

In 1978 he joined the faculty of the Electrical Engineering and Computer Science Department at MIT, where from 1979-1981 he held an Esther and Harold Edgerton Endowed Chair. He later served for 5 years as Associate Director of the Artificial Intelligence Laboratory. He is currently a Full Professor in the Department, and a Research Director of CSAIL, the newly-formed Computer Science and Artificial Intelligence Laboratory that resulted from the merger of the AI Lab and the Lab for Computer Science. He and his research group are developing advanced tools that permit natural, sketch-based interaction with software, particularly for computer-aided design and design rationale capture.

Dr. Davis has been one of the seminal contributors to the field of knowledge-based systems, publishing some 50 articles and playing a central role in the development of several systems. He serves on several editorial boards, including Artificial Intelligence, AI in Engineering, and the MIT Press series in AI. He is the co-author of Knowledge-Based Systems in AI, and was selected in 1984 as one of America's top 100 scientists under the age of 40 by Science Digest. In 1986 he received the AI Award from the Boston

Computer Society for his contributions to the field. In 1990 he was named a Founding Fellow of the American Association for AI and in 1995 was elected to a two-year term as President of the Association. In 2003 he received MIT's Frank E. Perkins Award for graduate advising. From 1995–1998 he served on the Scientific Advisory Board of the U.S. Air Force.

Dr. Davis has been a consultant to several major organizations, including Digital Equipment Corp, IBM, Aetna, and Schlumberger, and has been involved in the founding of three software companies.

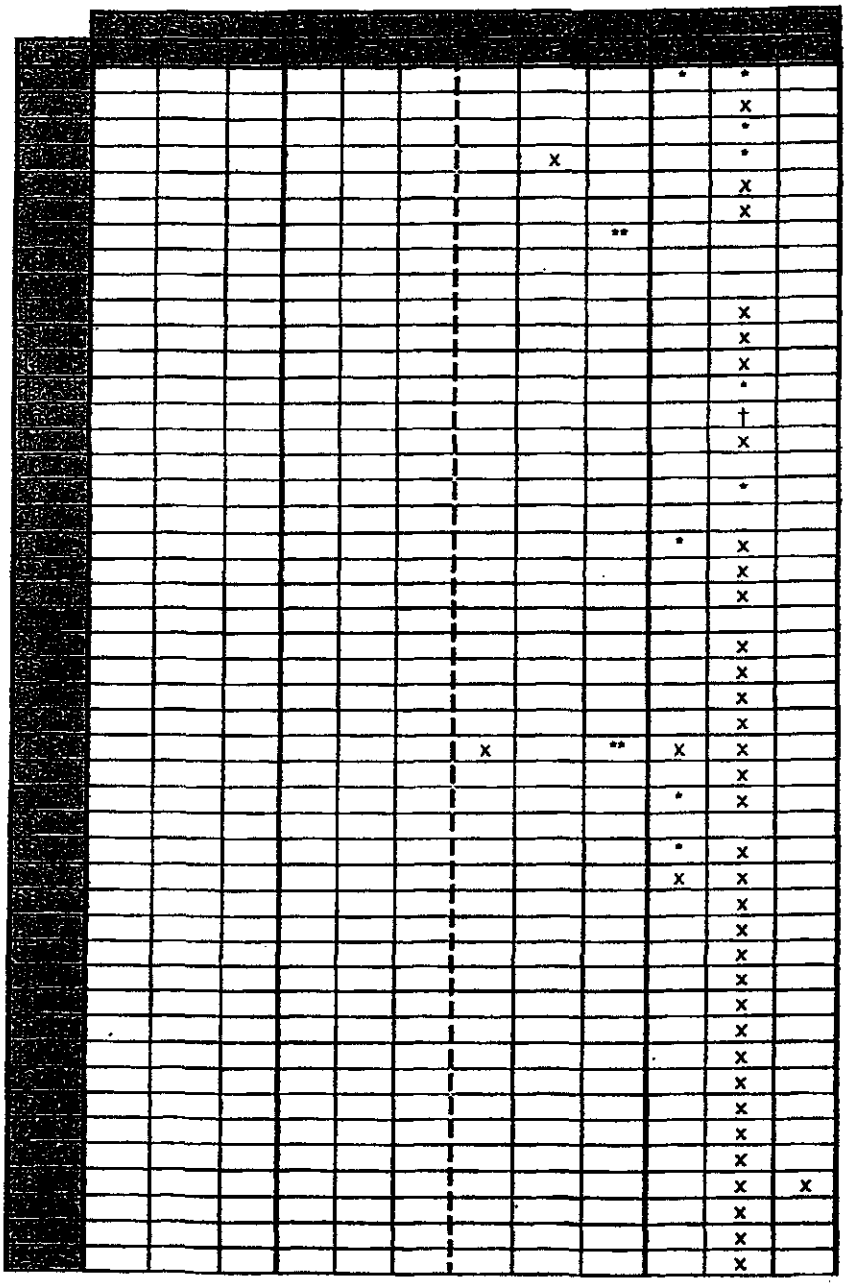
Dr. Davis has also been active in the area of intellectual property and software. In 1990 he served as expert to the Court in Computer Associates v. Altai, (775 F. Supp. 544 (E.D.N.Y. 1991); 982 F 2d 693) a case that produced the abstraction, filtration, comparison test for software copyright. He served on the panel run by the Computer Science and Telecommunications Board (CSTB) of the National Academy of Science in 1991 that resulted in Intellectual Property Issues in Software, and served as a member of the Advisory Board to the US Congressional Office of Technology Assessment study on software and intellectual property that was published in 1992 as Finding a Balance: Computer Software, Intellectual Property, and the Challenge of Technological Change. A 1994 paper in the Columbia Law Review analyzed the difficulties in applying intellectual property law to software and proposed a number of remedies.

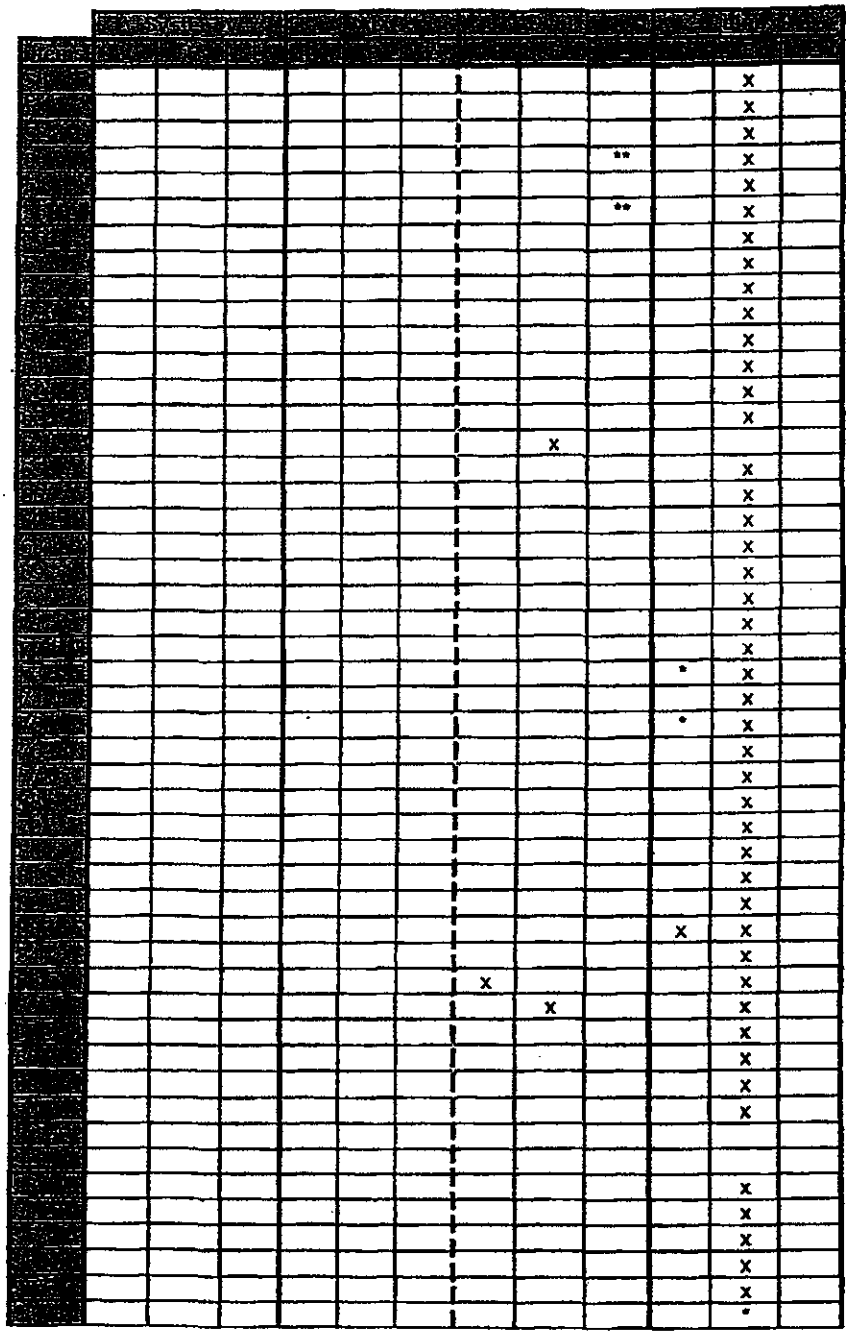
He has served as an expert in a variety of cases involving software, including the investigation by the Department of Justice of the Inslaw matter (40 Fed. Cl. 843; 1998 U.S. Claims), where he investigated allegations of copyright theft and cover-up by the Federal Bureau of Investigation, the National Security Agency, the Drug Enforcement Agency, the United States Customs Service, and the Defense Intelligence Agency. From 1998-2000 he served as the chairman of the National Academy of Sciences study on intellectual property rights and the information infrastructure entitled The Digital Dilemma: Intellectual Property in the Information Age, published by the National Academy Press in February, 2000.

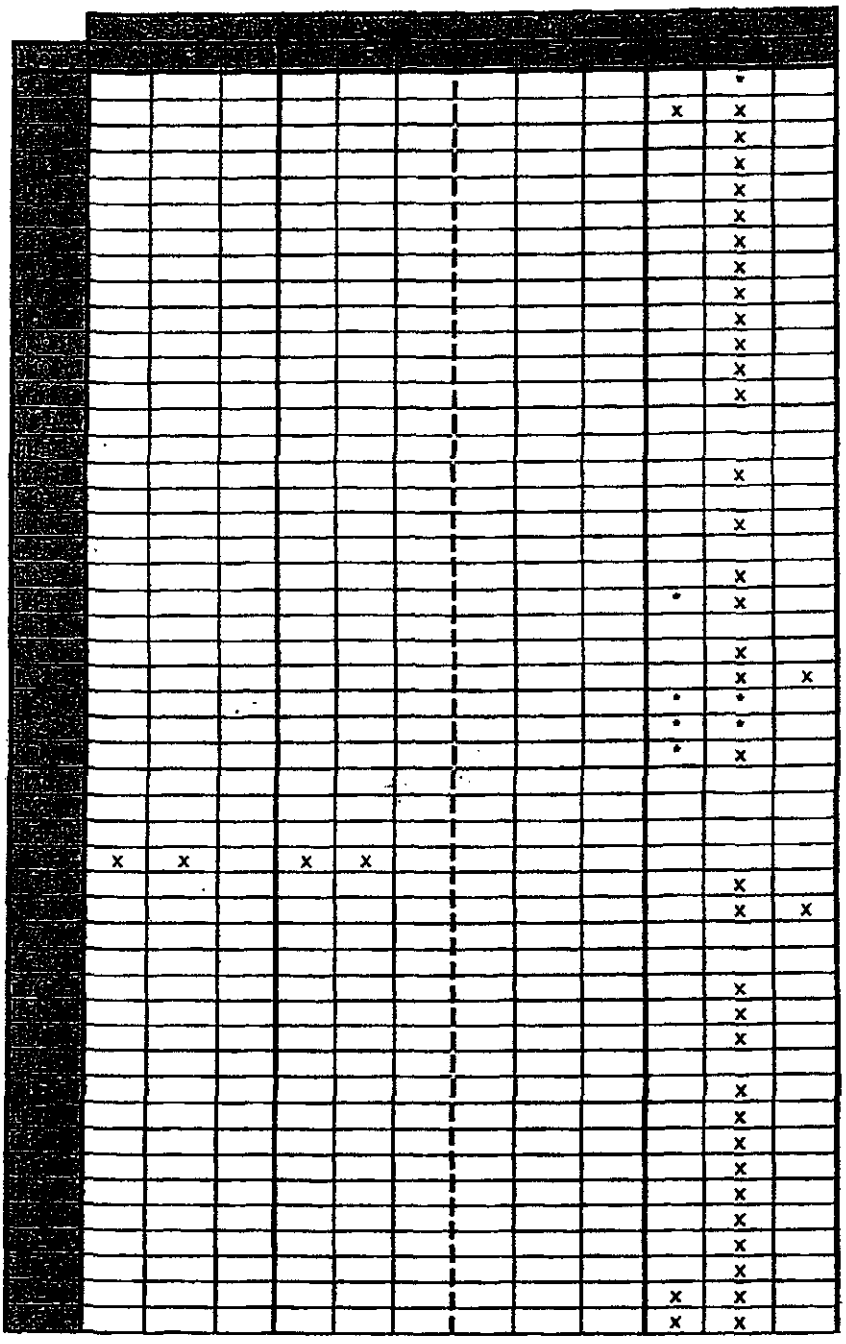
Dr. Davis has appeared on The Macneil/Lehrer Report and Innovations (WNET, NY), and played a major role in This Computer Thing, a pilot for an educational series (WGBH, Boston) about personal computers. He has been quoted in articles in The New York Times, The Wall Street Journal, Business Week, The Economist, The Boston Globe, High Technology, and Psychology Today. Interviews have appeared in Computerworld and on National Public Radio's All Things Considered. He has been a featured speaker in Texas Instrument's Satellite Symposium, and on Electronic Data Systems' internationally broadcast "Directions" program.

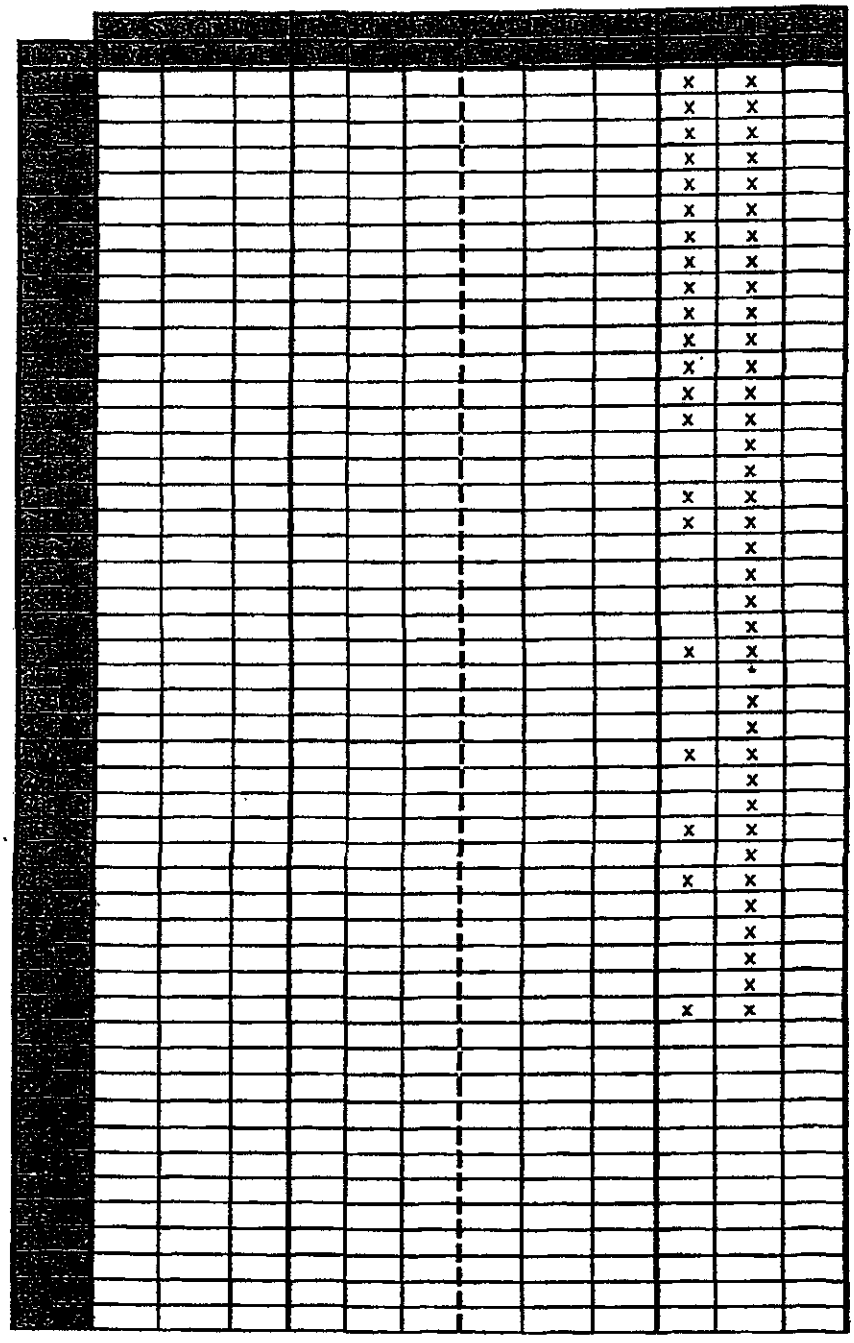
ADDENDUM B

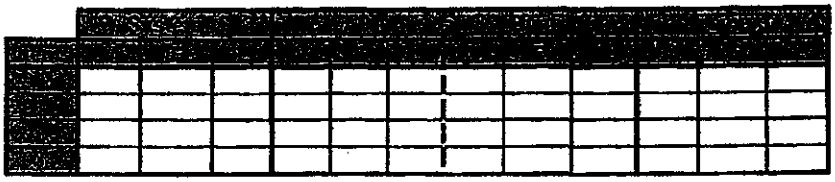
Addendum B











V = Product version identified
F = File name identified
L = Lines of code identified

* Indicates that only Linux patch information is given
** Indicates that some lines of code are displayed, but none specifically identified as misused
† Indicates that only Linux Test Project (LTP) information is given

EXHIBIT 22

COPY

Brent O. Hatch (5715)
HATCH, JAMES & DODGE, PC
10 West Broadway, Suite 400
Salt Lake City, Utah 84101
Telephone: (801) 363-6363
Facsimile: (801) 363-6666

Robert Silver, Esq. (admitted pro hac vice)
BOIES, SCHILLER & FLEXNER LLP
333 Main Street
Armonk, New York 10504
Telephone: (914) 749-8200
Facsimile: (914) 749-8300

Stephen N. Zack (admitted pro hac vice)
Mark J. Heise (admitted pro hac vice)
BOIES, SCHILLER & FLEXNER LLP
Bank of America Tower – Suite 2800
100 Southeast Second Street
Miami, Florida 33131
Telephone: (305) 539-8400
Facsimile: (305) 539-1307

Frederick S. Frei (admitted pro hac vice)
Aldo Noto (admitted pro hac vice)
John K. Harrop (admitted pro hac vice)
ANDREWS KURTH LLP
1701 Pennsylvania Avenue, Ste. 300
Washington, DC 20006
Telephone: (202) 662-2700
Facsimile: (202) 662-2739

Attorneys for Plaintiff The SCO Group, Inc.

IN THE UNITED STATES DISTRICT COURT

FOR THE DISTRICT OF UTAH

THE SCO GROUP, INC. Plaintiff/Counterclaim Defendant vs. INTERNATIONAL BUSINESS MACHINES CORPORATION Defendant/Counterclaim Plaintiff	**DECLARATION OF CHRIS SONTAG IN SUPPORT OF SCO'S OPPOSITION TO IBM'S MOTION FOR PARTIAL SUMMARY JUDGMENT** **Case No. 2:03-CV-0294 DAK** Honorable Dale A. Kimball Magistrate Judge Brooke C. Wells

DECLARATION OF CHRIS SONTAG

1. My name is Chris Sontag and I am Senior Vice President and General Manager of The SCO Group, Inc. My office is located in Lindon, Utah. Unless otherwise noted or evident from their context, this declaration is based on my personal knowledge and information available to me from reliable sources. To the best of my knowledge, information and belief, the facts set forth herein are true and correct.

2. I submit this Declaration in support of SCO's Memorandum in Opposition to Defendant/Counterclaim-Plaintiff IBM's Cross-Motion for Partial Summary Judgment on Its Claim for Declaratory Judgment of Non-Infringement, dated May 18, 2004.

3. The Court should allow SCO to conduct discovery to rebut IBM's Cross-Motion for Partial Summary Judgment on its Tenth Counterclaim for Declaratory Judgment of Non-Infringement ("IBM's Cross-Motion").

I. Introduction

4. For SCO to obtain all necessary and reasonable evidence to support its claims and to oppose IBM's Tenth Counterclaim, SCO must undertake a line-by-line comparison of Linux code and UNIX code. Based on our review to date, SCO believes that such comparison will reveal substantial similarity between the Linux and UNIX code.

5. There are inherent obstacles in identifying substantial similarities between UNIX and Linux. Both the UNIX and Linux operating systems are large and complex computer programs with many lines of code to compare. Furthermore, Linux code that is modified or derived from UNIX code may not necessarily bear line-for-line character similarity.

6. A kernel is the core portion of the operating system. The kernel performs the most essential operating system tasks, such as handling disk input and output operations and managing the internal memory.

7. The operating system kernel is a lengthy, complex computer program comprising numerous modules and files, and millions of lines of code. The Linux kernel (ver. 2.4) comprises 4 million lines of code and the UNIX SVR 4.2 MP kernel comprises 3.4 million lines of code.

8. To show that Linux code is substantially similar to UNIX code requires a comparison of that code which, as described below, is an undertaking of great magnitude and complexity. In other words, the 4 million lines of Linux code must be compared with the 3.5 million lines of UNIX code, line-by-line, or in groups of lines according to the structure, sequence or function of the group of lines. In the paragraphs that follow, I will describe a time-consuming and resource intensive approach to this process, and ways in which this process can be streamlined.

9. There are two basic ways to execute the code comparison: 1) using an automated process or computer program, and 2) manual review by a knowledgeable individual.

10. Attempting to use an automated process to perform a complete comparison of all of the source code in UNIX and Linux computer operating systems is not feasible. Automated tools to find copied lines of code are available "off-the-shelf." The tools are designed to find lines of code that are identical in every detail; they perform that function well. SCO and its experts have sought to modify and improve the tools to locate lines of code that are not identical but that are nearly identical; the tools have not always performed that function well. The

automated tools occasionally assist a programmer locate blocks of code that might have similarities. The programmer must then visually review the code in a difficult and labor-intensive process. Often this review is only possible if each version of the code can be reviewed to follow the changes from one version to the next.

11. Minor changes to a line of code such as punctuation, renaming a variable, changing comments, spelling changes, or alterations to the text will prevent the automated system from locating the matches in the lines of code. Similarly, inserting, deleting, or reordering lines of code will prevent the automated system from identifying a block of similar code. The reordering of lines of code may render the automated system useless.

12. Despite these shortcomings, and as described further below, SCO and its experts have used automated tools to locate lines of identical code, and they have visually analyzed the larger blocks of code in which those lines appear. For example, in a block of code having 100 lines, if two or three lines were found to match, a visual review would then be undertaken of the entire 100 line block of code looking for other similarities.

13. The automated tools may provide "false positives" that need to be manually reviewed. Some automated tools can provide a numerical value or percentage that represents a degree of similarity. In practice, however, files with very low similarity numbers are sometimes found to be substantially similar, while others with high values of numerical similarity have been found not to be substantially similar. Therefore, files need to be checked manually and the numerical similarity number is of little assistance.

14. Because of shortcomings with automated code comparison processes, SCO and its experts must rely largely on manual comparisons. Such manual comparisons are very labor and

time intensive. SCO and its experts must know or learn both the UNIX and Linux operating systems in detail. This process can take many months. To execute the comparison, without some roadmaps or list of "hot spots" in Linux, SCO and its experts must compare page after page of code. The 4 million lines of Linux kernel code takes up 66,000 pages; the 3.4 million lines of UNIX code takes up 58,000 pages. A simplistic manual comparison would involve placing the pages of code side by side in some ordered manner and then looking for the same or similar structure, sequence and organization of the code. Assuming each page comparison takes one (1) minute, and that there are 66,000 x 58,000 comparisons, this "initial" review could take on the order of 25,000 man-years. Following the initial review, SCO and its experts must conduct a detailed comparison of likely copying candidates. This "second-level" review would also be very lengthy.

15. One shortcut to comparing the UNIX and Linux code might be comparing similar directory structures of the UNIX and Linux operating systems. For example, version 2.4 of the Linux kernel contains 530 subdirectories and about 8750 source and assembly files. *See* Understanding The Linux Kernel, D. Broet, O'Reilly, 2003. Assuming each of the 8750 files requires one (1) day to investigate, about 35 man-years would be required to review all the Linux kernel files. However, this calculation ignores the possibility that the two operating systems use different file names, and that similar code sequences may reside in entirely different files.

16. Another shortcut may be to compare files from the UNIX and Linux operating systems that share the same or similar names (because the names of certain files correspond to the file's function). However, any significant overlap in the names of files between the UNIX and Linux operating systems is statistically unlikely.

17. Two ways of determining the number of files in any two computer operating systems that share the same or similar names are: (i) look for files whose names share the exact same characters in the exact same order, or (ii) look for files whose names share almost all of the same characters in almost the exact same order. To produce the most relevant results, each such search should take account of certain pre-defined language tokens (whose similarity between operating systems is not particularly probative of copying).

18. To "look" for such files, SCO and its experts have used computer programs to compare the thousands of files in UNIX and Linux operating systems. That is, based on the limited discovery to date and the operating systems that SCO already possessed or were publically available, SCO performed initial searches to find files that share the same or similar names. These comparisons represent only a small fraction of the total number of comparisons that could be made among the numerous versions of the UNIX, AIX/Dynix, and Linux operating systems.

19. The foregoing searches have permitted SCO and its experts to identify numerous files that, as between the UNIX and Linux operating systems, share the same or very similar names. SCO and its experts have used the results of the file searches to then turn to comparing the source code in those files.

20. Once they identified particular files with the same or similar names in UNIX and Linux, SCO and its experts used a combination of a computer program and manual review of the results of the program to find instances of substantial similarity in the operating systems. There are significant limitations to any such approach, and there is no way to eliminate human review and assessment of the program's results -- both of which are extremely time consuming.

21. SCO and its experts have used computer programs to identify the extent of similarity between lines of source code in any two given files. The results of one computer program shows where there are any differences between the lines of code. When a second computer program is run on those results, it shows where the lines of code (although different in some way) nevertheless contain the same code in the same sequence.

22. Once both computer programs have been run, SCO and its experts manually reviewed the results to assess the similarity between the lines of code at issue. The manual reviewer searches for instances where parts of the lines of code being compared are syntactically synonymous. That is, the reviewer determines whether the line of code in one file uses different words or characters to describe the same structure, function, declaration or subroutine as a line of code in the other file.

23. To date, this combination of automatic and manual review has been completed for only a very small portion of the Linux and UNIX operating systems, despite a significant man-hour expenditure, on the order of two-man years.

24. Another way for SCO to obtain all of the reasonably available and necessary evidence to support its claims and to oppose IBM's Tenth Counterclaim is to access the numerous IBM and Sequent engineers and programmers who have, over the years, developed AIX and Dynix code, contributed AIX and Dynix code to Linux, or assisted others in contributing to Linux. These engineers have access to and have studied UNIX based operating systems that have been enterprise hardened and made multiprocessor capable.

25. Once identified, the programmers and engineers can be deposed and can: provide identities of Linux contributors for further discovery, discuss their own Linux contributions,

discuss assistance given to Linux contributors, and discuss specific code segments that were contributed to Linux. This will also assist SCO in identifying former IBM employees who are contributing to Linux.

26. As will be discussed below, a revision control system (RCS – implemented by IBM as the Configuration Management/Version Control (CMVC) is an excellent source for finding the programmers and engineers familiar with relevant UNIX based code that has been contributed by IBM and third parties to make Linux enterprise hardened and multiprocessor capable. Deposing these programmers and engineers will allow SCO to prioritize its efforts to find Linux code that is substantially similar to UNIX code.

27. As discussed in the Declaration of Sandeep Gupta, the Linux kernel uses a ULS routine to block and unblock access to shared data. The Linux ULS routine is substantially similar to a ULS routine in UNIX. A Mr. Russel of IBM helped a Mr. Jamie Lokier contribute the UNIX ULS code into Linux. If SCO had access to IBM's CMVC, then SCO might have discovered that Mr. Russel worked on ULS for IBM, and could have deposed Mr. Russel to determine what specific help he provided in the contribution of ULS to Linux and to whom he provided that help.

28. Using the CMVC, and by deposing individuals such as Mr. Russel of IBM, SCO can significantly reduce the burden of reviewing Linux and UNIX code. Mr. Russel and other programmers can identify areas of Linux code that are copies of or are derived from AIX and Dynix code. Mr. Russel and other programmers can also identify contributors to the Linux code and can show the necessary access to AIX and Dynix that these contributors had.

29. SCO also can streamline obtaining all of the reasonably available evidence to support its claims and to oppose IBM's Tenth Counterclaim and can prioritize its search of Linux code that is substantially similar to UNIX code by examining the lineages of AIX and Dynix. By examining the source code in successive versions of AIX and Dynix, SCO can trace its UNIX code through to current versions of AIX and Dynix to determine where in Linux, SCO's UNIX code is copied. This tracing will allow SCO to prioritize its search of Linux code for evidence of copying of UNIX code.

30. Software (i.e., source code) undergoes many changes during its initial development and later over its operational life. Changes may occur as frequently as daily and can continue for years. Software changes typically are driven by the need to correct "bugs," to improve features, or to add new features. Because of changes made to source code over time, a current code version may "look" different than the initial code version, making identification of the initial code version difficult and substantial similarity and derivation more difficult to establish.

31. Software developers rely on version control systems (VCSs), or version management systems (VMSs), to control changes and revisions to source code. Version control systems are automated tools that provide specific access and tracking features to allow multiple parties to operate on and revise source code. For example, a "Checkout" feature allows a user to retrieve, from a source code repository, a section of source code for which some changes are intended. A "Checkin" feature deposits the changed source code in a source code repository. Version control systems also provide an approval process, and many other features. In short, version controls systems are software tools that provide detailed software change histories.

32. Related to the VCS is a "bug" tracking system or log. The bug tracking system allows users to log problems encountered with source code. Some bug tracking systems are implemented as web service applications, and allow software users to register problems using a Web page-provided form. Other bug tracking systems are internal to the company developing or supporting the source code. Because software changes are often driven by problem reports, it is natural to integrate these bug tracking systems with version control systems: this allows for a framework where changes resulting from a bug report can be easily located, and where some measure of certainty is provided that changes have been integrated into a product release. For large-scale software development projects, such integration is mandatory.

33. Both VCSs and bug tracking systems typically allow for some type of commentary to explain why a source code change was needed and to explain what was changed. VCSs and bug tracking systems are typically maintained in an electronic format, although hardcopy printouts may be available.

34. The advantage of the VCS is that it is an ongoing snapshot of how the software development took place. The VCS allows a user to view, through time, all changes to software by time and date, author, and possibly a reference to the bug tracking system. The VCS is an essential tool for software developers. For example, a bug may be reported to a software company, and to develop a correction, a software developer may refer back several years, or even decades, to prior versions of code so as to understand how the error (bug) developed, and how to revise the code to eliminate the bug. Without the VCS, this process could not be completed.

35. The VCS is also an essential tool that SCO can use to support its claims and to oppose IBM's Tenth Counterclaim. By viewing each version of AIX and Dynix, along with the associated CMVC or similar system, SCO will be able to track a Dynix/AIX code segment through its many derivative versions to its ultimate location in Linux. This will significantly streamline SCO's efforts to find code in Linux that is substantially similar to UNIX code. Moreover, the CMVC will identify programmers who can be deposed and who can explain where in Linux the code was contributed. By viewing each version of the Dynix/AIX code, SCO will be better able to determine if the structure, sequence, and organization of the corresponding Linux code matches that of UNIX.

II. Discovery Required From IBM

36. Similar to a software developer chasing a bug through time, SCO should be able to trace the development of UNIX-based source code from its initial AIX and Dynix versions through to current versions of the AIX and Dynix code and then into Linux. AIX, Dynix, ptx, and Dynix/ptx consist of millions of lines of source code, much of which likely will have undergone numerous (possibly hundreds) of changes. Tracing the current AIX, Dynix, ptx, and Dynix/ptx code versions to earlier code versions, and then ultimately to the corresponding UNIX code, will not be possible within any workable timeframe without a detailed "road map." The VCSs *and* bug tracking systems provide this road map. Additionally, the VCSs identify the software developer who authored the change and may now be assisting with development of Linux. These developers can be deposed to provide information that will help SCO prioritize its efforts to locate Linux code that is substantially similar to UNIX code.

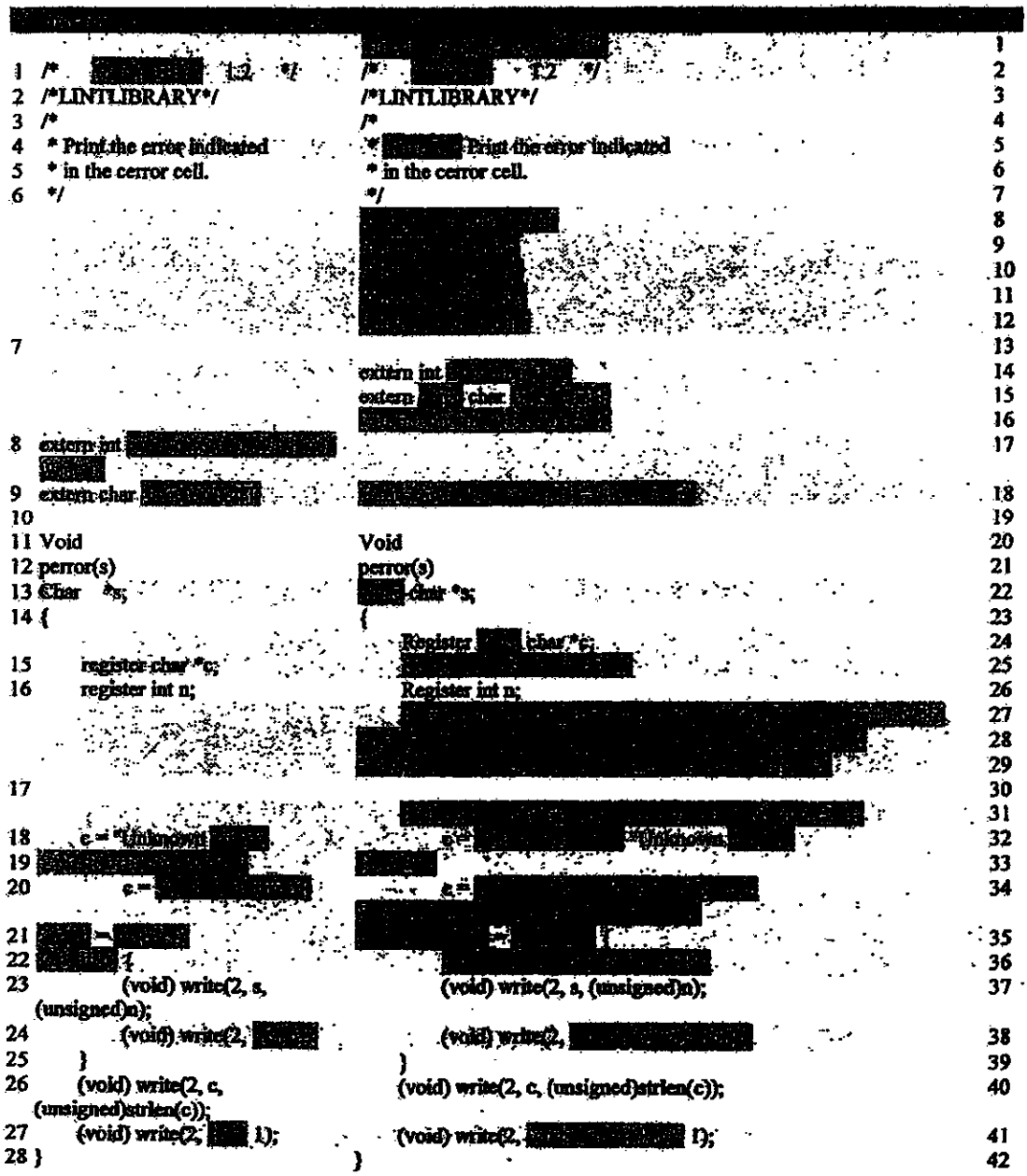
37. Shown and described below is an example of UNIX SVR4 source code *illustrating accumulated modifications over time. This is SCO source code for which SCO has* the versions available. The specific file name is perror.c.[1] Table 1 illustrates perror.c version 1.1, written in 1981 as compared to version 1.17, written in 1992. The shading indicates differences between the two versions. As can be seen from a casual review of the table, over 50 percent of the source code lines changed from the 1981 version to that from 1992. In fact, perror.c version 1.17 is so changed from version 1.1, that even an experienced UNIX programmer would have trouble determining that one was derived from the other.

38. The difference plot shown in the above Table 1 is replicated in color as Exhibit A. In the exhibit, the shaded areas are shown in two colors, pink and yellow. The yellow-shaded areas are lines of code where differences exist. Within each yellow-shaded area are pink-shaded areas that highlight specific differences in the code. Unshaded areas in the exhibit are lines of code were the two versions are identical.

39. Table 1 and Exhibit A show differences between UNIX code versions 1.1 and 1.17, and the differences are significant. However, as the code version numbers imply, there are many versions of the perror.c file from 1981 to 1992. Each of these code versions involves generally small variations from prior versions. It is the accumulation of these changes over time that makes the final version (i.e., version 1.17) look so different from the initial version (i.e., version 1.1). Exhibits B through L are differences plots of selected neighboring perror.c versions

[1] The code file "perror.c" is an unusual example of a UNIX source code file in that the file consists of only one short function (to generate a one-line description of the most recent error code, and provide a diagnostic). perror.c is written in 28 lines (in version 1.1). More typically, UNIX source code files consist of multiple functions and thousands of lines of code. File perror.c was chosen as an example for this Declaration because it illustrates the relevant concepts related to version control in a compact, easy to understand format.

Table 1
perror.c Source Code Difference Plot – Versions 1.1 and 1.17



from the initial version 1.1, through version 1.17. Viewing any of Exhibits B through L, a skilled UNIX programmer can readily see the evolutionary, derivative nature of the perror.c code development from 1981 to 1992. This derivative nature is not, however, readily apparent to the same skilled UNIX programmer based on the difference plot of Exhibit A alone.

40. As noted above, programmers use a VCS to track changes to code. The VCS serves as a road map that tracks all the code changes over time. Exhibit M is a printout of that portion of a VCS related to the perror.c file. As can be seen from Exhibit M, each change in the perror.c file is accompanied by an entry in the VCS that includes the date, identity of the author, and a comments section that lists the nature of and the reason for the change. Each entry in the VCS also references a corresponding entry in a bug tracking log. For example, entry D1.10 in the VCS, which relates to the change in perror.c from version 1.9 to version 1.10, refers to corresponding entry (referred to as UNIX Modification Request # bl86-28117) in the bug tracking log. Exhibit E shows the difference plot for this change. Exhibit N is the corresponding entry bl86-28117 from the bug tracking log. As can be seen from Exhibit N, the bug tracking log entry describes the specific problem that exists with the current version (in Exhibit E, the noted problem is that the existing error code does not check for an error (errno) less than zero) and lists what should be done to the perror.c code sequence to correct this problem. The bug tracking log entry also lists the originator of the log entry (in Exhibit E, D.E. Good), and the individual assigned to correct the problem (mao). The log entry further identifies the individual who approved the proposed problem correction (prb), the reason for change (error in design implementation) and other information that relates to the perror.c code.

41. Viewing Exhibit E, which is the difference plot between perror.c versions 1.9 and 1.10, line 20 of versions 1.9 and 1.10 are highlighted. The actual code change between the versions is shown in Table 2:

Table 2
perror.c errno Statement Versions

perror.c Version	Code Statement	Line Number
1.1	if (errno < _sys_nerr)	19
1.9	if (errno < _sys_nerr)	20
1.10	if (errno < _sys_nerr && errno >= 0)	20
1.17	if (thread_errno < 0 || thread_errno >= _sys_num_err)	32

42. As the example in Table 2 makes clear, even a skilled programmer would likely not be able to determine the derivation of the current perror.c code sequence without a road map that lays out specific changes in detail.

43. In view of the information provided in Paragraphs 4 - 42, SCO requires the following materials to analyze source code so that it can rebut IBM's Tenth Counterclaim:

- all version control system and bug tracking information (including documents, data, logs, files, and so forth) for AIX, Dynix/ptx, ptx, and Dynix from 1984 to the present,
- source code and log information for all interim and released versions of AIX, Dynix, ptx and Dynix/ptx from 1984 to the present, and
- depositions as appropriate of programmers identified from the foregoing.

44. The VCS information is especially important to SCO's opposition of IBM's Tenth Counterclaim. Without VCS information for AIX, Dynix/ptx, ptx, and Dynix, and any related

information such as documents, data, logs, files, SCO will not be able to prioritize its efforts to identify all lines of code in Linux that are derived or copied from UNIX System V. SCO will instead have to rely on luck and happenstance to find derived and copied code, then trace such code back to System V. The VCS information, however, will help SCO streamline its search efforts to find evidence that Linux code was copied or derived from UNIX System V code.

45. The materials in Paragraphs 31 and 32 – both the VCS information and the source code and log information – directly respond to IBM's factual allegations that the Linux code was developed or created by programmers, rather than taken from System V. *See, e.g., IBM's Cross-Motion* ¶¶ 1 ("collaborative development"); 2 (Linus Torvalds created a "new" operating system); 3 ("programmers joined to create code"); 4 (developers "contributed to the further development of Linux"). The VCS information will establish that various versions of AIX and Dynix are in fact derivative works of UNIX System V, and consequently, IBM's contributions to Linux from AIX and Dynix constitute copyright infringement.

46. Furthermore, the VCS information and the source code and log information will allow SCO to rebut IBM's allegations that SCO cannot prove copying (*IBM's Cross-Motion* ¶¶ 46 (SCO cannot show that IBM's activities infringe SCO's copyrights); 48 (SCO cannot establish that material in Linux is covered by SCO's copyrights)), as such information shows the history of development of AIX and Dynix code, the authors of the various versions of those systems, and the sources of the code. In other words, if a portion of Dynix code was obtained from UNIX System V, the VCS information for Dynix would show where that portion of code originally came from, who obtained it and when, and how that code was used in Linux.

47. SCO believes that much of its copyrighted code was copied from AIX and Dynix into Linux. While SCO has some evidence of literal copying between System V on the one hand, and Linux on the other hand, the VCS and source log information will show changes between various versions of AIX and Dynix, and the detailed history of those changes. Thus, SCO will be able to show that Linux code is substantially similar to UNIX code. SCO must have this material to establish what material in Linux is covered by SCO's copyrights, which IBM alleges SCO cannot do. *IBM's Cross-Motion* ¶ 48.

48. The evidence SCO currently has – three versions of AIX that IBM selected, Linux code, and System V code – is insufficient to show infringement because IBM could have copied System V code into any number of the multiple versions of AIX and Dynix. To trace SCO-owned code from System V into the code's current form in Linux, SCO must be able to trace every step and change the code underwent through AIX and Dynix. To do so, SCO requires the VCS information and the source code and log information. Without this material, SCO will not be able to prioritize and streamline its search efforts and will have to expend considerable time and resources to find evidence that Linux code is substantially similar to UNIX code.

49. IBM will not bear any significant burden to produce VCS information, IBM stores this information on its Configuration Management Version Control (CMVC) system. *See* IBM's CMVC Introduction (1710058191-92) (Exhibit O hereto).

50. SCO also requires the following materials to oppose IBM's Tenth Counterclaim: All design documents, white papers and programming notes, created from 1984 to the present.

These materials provide a wealth of information related to code development beyond that which can be found in the source code testing, VCS and bug tracking log.

51. White papers are usually generated early in the software code development process, and often discuss reasons for implementing code changes, problems with existing code, and alternative solutions. Thus, white papers serve as an early indication of possible code changes. By setting forth solutions, white papers can be used to look for specific code segments in Linux and thus help SCO prioritize its search.

52. Design documents are often prepared by the group that ultimately authors the changes to the code sequences. Design documents are generally more detailed that white papers. For example, SCO propriety design document "Virtual Memory Design for UNIX System V Release 4.2 Multiprocessor," contains almost 150 pages of detailed description and code requirements to implement virtual memory in a UNIX-based processor. The design document is directed to such implementation on a specific processor family, namely the Sequent Symmetry Model S16. This and other design documents explain the initial code concepts, and how such code will be developed and written. As such, design documents provide an invaluable bridge between existing code sequences, such as in UNIX, and derivative works, such as in AIX and Dynix. Because these design documents describe the basis for code development, they may be useful for pointing to a portion of Linux that contains code substantially similar to UNIX code.

53. Finally, programming notes contain the thought processes of individual programmers as they write and revise code sequences. For example, programming notes might list changes made to code, and might list additional changes to consider. As such, programming notes provide detailed rationale for code changes and an indication of how the code may change

in the future. Programming notes may reflect the purpose for code changes and where in the kernel those changes occur. Thus programming notes are another source SCO can use to streamline its efforts to locate Linux code that is substantially similar to UNIX code.

54. SCO requires these white papers, design documents, and programming notes for all AIX, Dynix, ptx, and Dynix/ptx.

55. To find copying of other operating systems (e.g., AIX, Dynix, ptx, and Dynix/ptx) and features of UNIX System V, SCO must have the discovery related to items listed in Paragraphs 4 - 54. SCO believes that many of these features of its System V were copied, directly and/or indirectly, into AIX and Dynix by IBM. Therefore, SCO requires discovery of materials related to these aspects of AIX and Dynix to establish IBM's copyright infringement and rebut its Cross-Motion. Also, information is needed as to these code sequences to determine if some UNIX code has been copied, and if the Linux version is a substantially similar to UNIX.

56. SCO previously requested the above-listed materials in *SCO's Memorandum Regarding Discovery* submitted to Magistrate Judge Brooke C. Wells on May 28, 2004, pursuant to Magistrate Judge Wells' Order dated March 3, 2004. To date, Magistrate Judge Wells has not ruled on SCO's May 28 memorandum.

III. Discovery Required From Third Parties

57. Unlike AIX and Dynix, Linux code was developed in a somewhat unstructured format. At a minimum, no single version control system, or group of version control systems was implemented to track the derivation and evolution of Linux and to screen out contributions that may constitute copyright infringement. Furthermore, IBM has stated that thousands of programmers contributed code to Linux, and hundreds of Linux versions exist. Simply put, no

road map exists that will allow SCO to trace the migration of UNIX code into Linux completely. Thus, the discovery sought by SCO will permit SCO to identify major contributors to Linux so as to focus and narrow discovery. Clearly, it is impossible to seek discovery from thousands of contributors worldwide, and SCO does not intend to do so. Accordingly, SCO requires discovery relating to third party contributions to Linux – thus demonstrating that IBM's use of Linux constitutes infringement – as follows:

- Determine what third parties IBM has partnered with to develop Linux and what work those groups have done. Many of these arrangements are not in the public domain, particularly as to the details of the partnering, such as which party makes what contribution, the motivation for the contribution, and the starting and ending code versions that resulted from the partnership. This discovery will also help SCO identify specific code authors, who can then be deposed.

- Take discovery on Linus Torvalds, the purported creator of Linux, about the contributors and contributions to Linux since its inception, and the maintenance of any records about the development history of Linux. Mr. Torvalds is expected to have detailed records of these contributors and their contributions, material that is not publicly available. Further, Mr. Torvalds can answer specific questions as to what each contributor intended, and where and how the contributor acquired or developed the derived code.

- Take discovery on maintainers of the kernels. Kernel maintainers take responsibility for approving and including patches for Linux, and should have a wealth of information on who has contributed what code to the various Linux kernels over the years.

- There are many contributors to the kernels, some who have significant contributions to Linux code over the years. Some of these individuals, whose names are publicly available, should be deposed to find out their sources for their contributed code.

- Many corporations have made contributions to Linux, and SCO needs to take discovery on certain of these companies to determine the sources of their contributions. Also, SCO needs to depose the programmers who work for these companies and made the contributions to determine the sources of those programmers' code contributions. This discovery will show why the contributions were made and what features the contributions relate to, and will allow SCO to trace back from the Linux code to UNIX.

- SCO has identified some, but not all, independent authors of various portions of Linux code. (See partial list at Exhibit P hereto.) Those authors should know the sources of their code and should be able to provide information as to whether the code they contributed to Linux was obtained from SCO copyrighted code.

- Several private groups also made major contributions to Linux, so SCO should also be permitted adequate time to identify and take discovery from these entities.

- Many organizations exist whose purpose is to track and report on changes to Linux, and in many cases to collect documentation on Linux and distribute that information. However, such reporting is generally very summary, and SCO needs access to the more detailed information these organizations maintain. Such organizations are also potential sources of infringement information.

- Licensees and former licensees of UNIX source code to see if these entities, their employees, or former employees are contributing UNIX code to Linux.

58. IBM asserts that SCO has yet to set forth evidence that any Linux code infringes any SCO copyright and that SCO cannot do so. To counter this assertion more fully, SCO needs the discovery requested herein and the time to analyze it to find all instances of substantial similarity.

59. This discovery will provide leads as to which portions of the Linux code have copied portions from UNIX. Linux is so extensive that an evaluation of each of its 8750 files would be an enormous task. SCO needs some initial discovery in the area of third-party contributions to Linux to focus SCO research and further discovery.

60. SCO has not yet undertaken to conduct this far-reaching third party discovery because it is relevant only to IBM's tenth counterclaim which was only recently filed and not to SCO's copyright infringement claim. As a user of Linux (which IBM contends is one of its Linux activities), IBM copies Linux. Therefore, IBM's use and copying of Linux necessarily involves all of the source code contributed by others, as well as its own. Thus, SCO requires a continuance to take this discovery.

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

July ___, 2004

Chris Soutas

CERTIFICATE OF SERVICE

Plaintiff, The SCO Group, hereby certifies that a true and correct copy of **DECLARATION IN SUPPORT OF SCO'S MOTION FOR CONTINUANCE PURSUANT TO RULE 56(f)** was served on Defendant International Business Machines Corporation on the 9th day of July, 2004, as follows:

BY HAND DELIVERY:

Alan L. Sullivan, Esq.
Todd M. Shaughnessy, Esq.
Snell & Wilmer L.L.P.
15 West South Temple, Ste. 1200
Salt Lake City, Utah 84101-1004

Evan R. Chesler, Esq.
Cravath, Swaine & Moore LLP
825 Eighth Avenue
New York, NY 10019

Donald J. Rosenberg, Esq.
1133 Westchester Avenue
White Plains, New York 10604

EXHIBIT 23

SNELL & WILMER L.L.P.
Alan L. Sullivan (3152)
Todd M. Shaughnessy (6651)
Amy F. Sorenson (8947)
15 West South Temple, Suite 1200
Salt Lake City, Utah 84101
Telephone: (801) 257-1900
Facsimile: (801) 257-1800

CRAVATH, SWAINE & MOORE LLP
Evan R. Chesler (admitted pro hac vice)
David R. Marriott (7572)
Worldwide Plaza
825 Eighth Avenue
New York, New York 10019
Telephone: (212) 474-1000
Facsimile: (212) 474-3700

Attorneys for Defendant/Counterclaim-Plaintiff International Business Machines Corporation

IN THE UNITED STATES DISTRICT COURT

FOR THE DISTRICT OF UTAH

THE SCO GROUP, INC.,

Plaintiff/Counterclaim-Defendant,

v.

INTERNATIONAL BUSINESS MACHINES CORPORATION,

Defendant/Counterclaim-Plaintiff.

IBM'S MOTION TO LIMIT SCO'S CLAIMS RELATING TO ALLEGEDLY MISUSED MATERIAL

(ORAL ARGUMENT REQUESTED)

Civil No.: 2:03CV-0294 DAK

Honorable Dale A. Kimball

Magistrate Judge Brooke C. Wells

Defendant/Counterclaim-Plaintiff International Business Machines Corporation ("IBM"), through counsel, respectfully submits this motion, pursuant to Rules 1, 26, 30 and 37 of the Federal Rules of Civil Procedure, to limit the scope of SCO's claims to the Items of allegedly misused material disclosed with sufficient specificity in SCO's December 22, 2005 Disclosure of Material Allegedly Misused by IBM (the "Final Disclosures").

As this Court has recognized, SCO has made a plethora of public statements accusing IBM of misconduct, while offering no support for its allegations. The Court deferred IBM's motions for summary judgment but ordered SCO to particularize its claims, once and for all, in the Final Disclosures. SCO has declined. For 201 of its 294 Items, SCO fails to identify the allegedly misused material with the most basic detail. SCO's failure to provide even the most basic specificity for its claims is extraordinarily prejudicial to IBM and should not be allowed. Thus, IBM respectfully requests that the Court limit SCO's claims to the 93 Items for which SCO provides detail sufficient to identify both the allegedly misused material and the allegedly improper source of that material.

For the foregoing reasons, and as set forth in detail in the accompanying memorandum filed and served herewith, IBM respectfully requests that the Court enter an Order limiting the scope of SCO's claims relating to allegedly misused material to the following Items in SCO's Final Disclosures: Item Nos. 1, 113-142, 150-164, 183-185, 194-203, 205-231, and 272-278.

DATED this 13th day of February, 2006.

Snell & Wilmer L.L.P.

/s/Todd M. Shaughnessy
Alan L. Sullivan
Todd M. Shaughnessy
Amy F. Sorenson

CRAVATH, SWAINE & MOORE LLP
Evan R. Chesler
David R. Marriott

Of Counsel:

INTERNATIONAL BUSINESS MACHINES CORPORATION
Jennifer M. Daniels
Alec S. Berman
1133 Westchester Avenue
White Plains, New York 10604
(914) 642-3000

Attorneys for Defendant/Counterclaim-Plaintiff
International Business Machines Corporation

CERTIFICATE OF SERVICE

I hereby certify that on the 13th day of February, 2006, a true and correct copy of the foregoing was hand-delivered to the following:

Brent O. Hatch
Mark F. James
HATCH, JAMES & DODGE, P.C.
10 West Broadway, Suite 400
Salt Lake City, Utah 84101

and a true and correct copy of the foregoing was sent by U.S. Mail, postage prepaid, to the following:

Robert Silver
Edward Normand
BOIES, SCHILLER & FLEXNER LLP
333 Main Street
Armonk, New York 10504

Stephen N. Zack
Mark J. Heise
BOIES, SCHILLER & FLEXNER LLP
100 Southeast Second Street, Suite 2800
Miami, Florida 33131

/s/ Todd M. Shaughnessy

383461.1

EXHIBIT 24

SNELL & WILMER L.L.P.
Alan L. Sullivan (3152)
Todd M. Shaughnessy (6651)
Amy F. Sorenson (8947)
15 West South Temple
Gateway Tower West
Salt Lake City, Utah 84101-1004
Telephone: (801) 257-1900
Facsimile: (801) 257-1800

CRAVATH, SWAINE & MOORE LLP
Evan R. Chesler (admitted pro hac vice)
David R. Marriott (7572)
Worldwide Plaza
825 Eighth Avenue
New York, New York 10019
Telephone: (212) 474-1000
Facsimile: (212) 474-3700

Attorneys for Defendant/Counterclaim-Plaintiff
International Business Machines Corporation

IN THE UNITED STATES DISTRICT COURT

FOR THE DISTRICT OF UTAH

THE SCO GROUP, INC., Plaintiff/Counterclaim-Defendant, -against- INTERNATIONAL BUSINESS MACHINES CORPORATION, Defendant/Counterclaim-Plaintiff.	**IBM'S MEMORANDUM IN SUPPORT OF MOTION TO LIMIT SCO'S CLAIMS RELATING TO ALLEGEDLY MISUSED MATERIAL** **(ORAL ARGUMENT REQUESTED)** Civil No. 2:03CV-0294 DAK Honorable Dale A. Kimball Magistrate Judge Brooke C. Wells

Defendant/counterclaim-plaintiff International Business Machines Corporation ("IBM") respectfully submits the following memorandum in support of its Motion to Limit SCO's Claims Relating to Allegedly Misused Material. By this motion, IBM seeks to limit the scope of SCO's claims to the Items of allegedly misused material disclosed with sufficient specificity in SCO's December 22, 2005 Disclosure of Material Allegedly Misused by IBM (the "Final Disclosures").

Preliminary Statement

As this Court has recognized, SCO has made a plethora of public statements accusing IBM of misconduct, while offering no support for its allegations. The Court deferred IBM's motions for summary judgment but ordered SCO to particularize its claims, once and for all, in the Final Disclosures. SCO has refused. Although all 294 Items identified in the Final Disclosures fail to provide the level of specificity sought by IBM and required by the Court, the lack of specificity for 201 of the 294 Items renders it impossible as a practical matter for IBM to defend itself. For those 201 Items, SCO fails to identify the allegedly misused material with the most basic detail. SCO's failure to provide even the most basic specificity for its claims is extraordinarily prejudicial to IBM and should not be allowed. Thus, IBM respectfully requests that the Court limit SCO's claims to the 93 Items for which SCO provides detail sufficient to identify the allegedly misused material.

Following SCO's repeated failure to respond to IBM's discovery requests, Magistrate Judge Wells twice ordered SCO to respond to the requests with specificity. In an order dated December 12, 2003, Magistrate Judge Wells ordered SCO to "identify and state with specificity the source code(s) that SCO is claiming form the basis of their action against IBM". (12/12/2003 Order ¶ 4.) Again, in an order dated March 3, 2004, Magistrate Judge Wells ordered SCO "to

provide and identify all specific lines of code that IBM is alleged to have contributed to Linux from either AIX or Dynix" and "to provide and identify all specific lines of code from Unix System V from which IBM's contributions from AIX or Dynix are alleged to be derived". (03/03/04 Order ¶¶ I.1-I.3.) SCO failed to comply, and IBM moved for summary judgment.

After deferring IBM's summary judgment motions, this Court likewise required SCO to particularize its claims. In an order dated July 1, 2005, the Court adopted (over SCO's objection) an IBM proposal to set interim and final deadlines for the disclosure of all allegedly misused material. The Court set October 28, 2005, as the "Interim Deadline for Parties to Disclose with Specificity All Allegedly Misused Material Identified to Date and to Update Interrogatory Responses Accordingly". (07/01/2005 Order ¶ III.) The Court set December 22, 2005, as the "Final Deadline for Parties to Identify with Specificity All Allegedly Misused Material". (Id.)

Although IBM had already produced hundreds of millions of lines of source code (which SCO could have used to comply with the Court's orders), SCO demanded that IBM produce hundreds of millions of lines of additional code, programmers' notes and design documents. IBM produced the equivalent of tens of millions of pages of these materials. As described in the May 3, 2005 Declaration of Todd M. Shaughnessy, the production involved more than 4,700 hours of work from more than 400 IBM employees, not including the time spent by IBM counsel and consultants. (05/03/2005 Shaughnessy Decl. ¶ 5 (attached as Exhibit A).)

SCO's interim disclosures nevertheless fell far short of the specificity required by the Court. SCO failed, for example, to describe all of the allegedly misused material by version, file and line of code. SCO refused to disclose versions, files and/or line numbers for the code at

issue with respect to the 201 Items in dispute on this motion. IBM promptly brought these deficiencies to SCO's attention and asked that SCO correct them in its Final Disclosures. (See 12/5/05 Letter from T. Shaughnessy to T. Normand, a true and correct copy of which is attached as Exhibit B.) Because it is practically impossible to defend against imprecise allegations, IBM advised SCO it would ask the Court to preclude SCO from pursuing any claims regarding allegedly misused material not properly disclosed in the Final Disclosures, which we asked SCO to provide in an electronic format that would allow efficient analysis by IBM. (See id. at p. 2.) SCO did not respond to that letter, or otherwise object to IBM's request.

Rather than correct the shortcomings in SCO's interim disclosures, the Final Disclosures (which SCO declined to provide in an electronic form, hindering IBM's analysis) merely compound them, by challenging even more items without specifically describing them. None of the 294 Items in the Final Disclosures provide the level of detail sought by IBM and required by the Court. Remarkably, for 201 of the 294 Items, SCO does not provide enough particularity even to identify the versions or line numbers for the allegedly misused material. (See Item Nos. 2-112, 143-149, 165-182, 186-193, 204, 232-271, 279-294.)[1] In fact, no versions, files or lines of Unix System V code are identified; no versions, files or lines of Dynix or AIX code are identified as misused; and no specific versions or lines of Linux code are identified.[2] For these

[1] Although SCO does provide versions and line numbers for the files identified in Item No. 204, SCO makes no claim as to any misuse of the code identified in Item No. 204. Under the heading "Improperly Disclosed Code, Method, or Concept", SCO states: "N/A". (See infra note 4.)

[2] Although SCO identifies certain Linux files (but not specific versions or lines of code) as to the 201 Items in dispute, a number of the files are identified unclearly and inconsistently. In some cases, SCO seems simply to refer IBM to a website. (See, e.g., Item Nos. 9, 11, 18, 98, 178.)

201 Items, SCO comes nowhere close to providing the information that IBM needs to defend itself and that the Court ordered SCO to provide.[3]

As is further discussed below, SCO should now finally be precluded from proceeding any further on those 201 Items for which it has not provided even the most basic identifying information.

Argument

I. THE FINAL DISCLOSURES LACK THE REQUISITE SPECIFICITY.

At this point, IBM has been asking SCO for nearly three years to identify with specificity the material that IBM is alleged to have misused. Specifically, IBM has requested SCO to specify the versions, files and line numbers of the allegedly misused material. The Court has ordered SCO no less than three times to do so. Yet, as described above, SCO has refused. While the sheer magnitude of the materials provided with the Final Disclosures gives the false impression of detail, the 201 Items at issue on this motion fail to identify any versions, files or lines of any Unix System V, AIX or Dynix code as being misused. The Items at issue identify Linux files in most cases (albeit inconsistently and confusingly) but nowhere specifically identify any versions or lines of Linux code; in some cases IBM simply is referred to a website. (See, e.g., Item Nos. 9, 11, 18, 98, 178.)

[3] The shortcomings in the Final Disclosures are not limited to failing properly to identify the versions, files and lines of code of the relevant UNIX System V, AIX, Dynix and Linux material. SCO also fails to provide, for example, adequate information as to when and how the allegedly misused material was ever disclosed, by SCO or anyone else; details as to the origin of the material, including when, where and by whom the material was created; and all products in which the material is included, or on which the material is based.

Item Nos. 271 and 294 of the Final Disclosures illustrate the problem. Item No. 271 claims that "AIX and Dynix/ptx patented technologies, based on UNIX System V, were improperly released for the benefit of, and use by, the Linux development community in developing Linux." SCO does not identify a single version, file or line of Unix System V, AIX, Dynix or Linux technology that IBM is alleged to have misused. Instead, SCO merely attaches 34 patents. None of these 34 patents lists any versions, files or lines of code. There is, therefore, no way of telling what, if any, Unix System V, AIX, Dynix or Linux technology SCO contends was misused. Similarly, SCO's Item No. 294 alleges that IBM has engaged in "[e]xtensive use of ptx programming experience (and a fortiori exposure to UNIX System V) in creating numerous Linux kernel patches". In support of this claim, SCO attaches a computer disk containing 33,000 single-spaced pages of proposed code contributions. Nowhere does SCO identify with specificity a single version, file or line of Unix System V, AIX, Dynix or Linux code. Here again, IBM is left to guess as to SCO's claim.

SCO's failure to specify its claims is especially egregious because it has had the information necessary to do so since nearly the beginning of this lawsuit. SCO was founded as a Linux company, and Linux source code has been available for download from the internet since the inception of Linux—long before the commencement of this lawsuit. SCO purports to own all Unix System V code and, thus, has ready access to all of the System V code. Further, IBM produced millions of lines of AIX and Dynix source code to SCO almost two years ago and supplemented the production nearly nine months ago with hundreds of millions of additional lines of code, including all iterations and versions of such code maintained by IBM, and thousands of programmers' notes and design documents. Despite requiring IBM to devote

considerable resources to providing SCO with this information, SCO identifies lines of AIX or Dynix code for only one of the 201 Items at issue and fails to make any allegation of misuse in relation to that code.[4]

II. SCO'S GAMESMANSHIP IS EXTREMELY PREJUDICIAL TO IBM.

The shortcomings in the Final Disclosures are not only pervasive, but they also result in extraordinary prejudice to IBM. The lack of particularity in the Final Disclosures cloaks SCO's claims in uncertainty and makes it practically impossible for IBM to defend itself.

SCO contends generally that IBM misused the Unix System V code (which SCO purports to own) and the AIX and Dynix code (which IBM owns, but SCO purports to control). According to SCO, IBM improperly "dumped" Unix System V, AIX and Dynix into Linux. Given the scope of the code implicated by SCO's claims, however, it is practically impossible to assess and defend against them without knowing exactly which versions, files and lines of code SCO contends are at issue. As the Court will recall, there are numerous versions of Unix System V, AIX, Dynix and Linux, and each version consists of thousands of files and millions of lines of code. For example, Unix System V R4.2 ES/MP consists of 22,222 files and 7,339,157 lines of code; AIX 4.3.3 for Power consists of 111,964 files and 138,420,329 lines of code; and Linux 2.6.15 consists of 18,811 files and 7,290,070 lines of code.

[4] In Item No. 204, SCO provides a comparison of System V source code and Dynix source code to support the unremarkable, and uncontested, proposition that the Dynix operating system contains certain code modified or derived from System V source code; neither party contests the fact that IBM (through Sequent) had a valid license to include System V source code in Dynix. In fact, as noted above, SCO makes no claim of misuse of the material identified in Item No. 204. (See supra note 1.)

SCO's failure to specify its claims leaves IBM no way to defend itself except by undertaking a massive analysis, potentially of every single version, file and line of Unix System V code, every single version, file and line of code in AIX and Dynix, and every single version, file and line of code in Linux.[5] As SCO well knows, there is no way IBM could conduct this analysis in several years, let alone in the several months afforded by the scheduling order. Unlike SCO, IBM does not know what SCO claims. If tolerated, SCO's gamesmanship would give IBM and its experts no meaningful opportunity to evaluate in advance the claims SCO may choose to trot out in its expert reports, in opposition to IBM's summary judgment motions and/or at trial.

III. THE ONLY APPROPRIATE REMEDY FOR SCO'S GAMESMANSHIP IS TO LIMIT ITS CLAIMS TO THE DISCLOSED ITEMS FOR WHICH SCO PROVIDED SUFFICIENT SPECIFICITY.

SCO's failings regarding the Final Disclosures do not occur on an empty set. They come following repeated discovery requests by IBM and three separate orders of this Court. Indeed, they come in derogation of this Court's orders. The appropriate remedy for a party's failure to comply with an order requiring the disclosure of the party's claim is an order precluding the party from pursuing undisclosed elements of the claim.

Many courts have held that a party's claim must be limited to exclude elements of the claim for which the party has failed to provide appropriate, court-ordered discovery. See Imax

[5] Based on SCO's claims, the investigation would have to include, among other things, an inquiry into the origin of the code, the value of the code, whether SCO distributed the code under the terms of the General Public License, whether the code was developed to comply with publicly known standards, whether the code is dictated by externalities, whether the code is merely an unprotectable idea, whether the code ever shipped without a required copyright notice, and whether the code is otherwise in the public domain.

Corp. v. Cinema Tech., Inc., 152 F.3d 1161, 1167 (9th Cir. 1998) (affirming district court's decision "refusing [under Fed. R. Civ. P. 26(e)] to consider any trade secret material that was not specifically listed in haec verba in [plaintiff's] Fourth Supplemental Responses", because defendant "could not be expected to prepare its rebuttal to [plaintiff's] trade secrets claim without some concrete identification of exactly which [elements] alleged were incorporated into [defendant's] own projector system"); Kang v. Lee, No. 96 Civ. 1145, 1997 WL 669787, at *3 (S.D.N.Y. Oct. 27, 1997) (ruling that "[a]s a result of Defendant's failure to comply with Plaintiff's discovery demands, even after this Court directed him to do so, he has been precluded from offering any evidence at trial relating to matters raised in Plaintiff's unanswered interrogatories and unsatisfied document requests").

Modifying the Scheduling Order either to afford IBM more time to evaluate SCO's claims or to provide SCO an opportunity to amend its disclosures would not be an adequate solution to the lack of specificity in the Final Disclosures. It would require years for IBM to chase all of the facts relating to the hundreds of millions of lines of code implicated by SCO's claims. As described above, in spite of the benefit of almost three years time and numerous requests from IBM and instructions from the Court, SCO has repeatedly refused to identify with specificity the basis of its claims. The resolution of this case should not be delayed further to provide SCO yet another opportunity. It has had more than enough opportunity to comply with the Court's orders. As IBM has previously advised the Court, we believe it is in IBM's interest and in the public interest to bring this case to a close as soon as possible.

In short: enough is enough. SCO should now finally be precluded from proceeding any further on those 201 Items for which it has not provided even the most basic identifying information. (See Item Nos. 2-112, 143-149, 165-182, 186-193, 204, 232-271, 279-294.)

Conclusion

For the foregoing reasons, IBM respectfully requests that the Court enter an order limiting the scope of SCO's claims relating to allegedly misused material to the following Items in SCO's Final Disclosures: Item Nos. 1, 113-142, 150-164, 183-185, 194-203, 205-231, and 272-278.

DATED this 13th day of February, 2006

SNELL & WILMER L.L.P.

/s/ Todd M. Shaughnessy
Alan L. Sullivan
Todd M. Shaughnessy
Amy F. Sorenson

CRAVATH, SWAINE & MOORE LLP
Evan R. Chesler
David R. Marriott

Attorneys for Defendant/Counterclaim-Plaintiff International Business Machines Corporation

Of counsel:

INTERNATIONAL BUSINESS MACHINES CORPORATION
Jennifer M. Daniels
Alec S. Berman
1133 Westchester Avenue
White Plains, New York 10604
(914) 642-3000

Attorneys for Defendant/Counterclaim-Plaintiff International Business Machines Corporation

CERTIFICATE OF SERVICE

I hereby certify that on the 13th day of February, 2006, a true and correct copy of the foregoing was hand-delivered to the following:

Brent O. Hatch
Mark F. James
HATCH, JAMES & DODGE, P.C.
10 West Broadway, Suite 400
Salt Lake City, Utah 84101

and a true and correct copy of the foregoing was sent by U.S. Mail, postage prepaid, to the following:

Robert Silver
Edward Normand
BOIES, SCHILLER & FLEXNER LLP
333 Main Street
Armonk, New York 10504

Stephen N. Zack
Mark J. Heise
BOIES, SCHILLER & FLEXNER LLP
100 Southeast Second Street, Suite 2800
Miami, Florida 33131

/s/ Todd M. Shaughnessy

EXHIBIT 25

The Salt Lake Tribune

Click here to go back to search results.

Click here for a printer friendly version of this article.

Salt Lake Tribune, The (UT)

Date: July 13, 2006
Section: Business

SCO will appeal the gutting of its lawsuit against IBM

Bob Mims The Salt Lake Tribune

Utah's SCO Group is appealing a federal magistrate's gutting of its $5 billion lawsuit against IBM, hoping to salvage the tens of millions of dollars it has spent litigating the case over the past three years. The Lindon software company asked U.S. District Judge Dale Kimball to reverse a June 28 ruling by U.S. Magistrate Brooke Wells that struck down two thirds of SCO's allegations connected to Big Blue's purported leaking of SCO's Unix code into the freely distributed Linux operating system.

The appeal, technically an "objection" seeking Kimball's review, argues that Wells' reasoning confused SCO's contractual allegations over IBM's alleged export of the code with SCO's Unix copyright claims.

SCO attorney Brent Hatch said the appeal, which was filing electronically with the court clerk's office Thursday night, also contends Wells took out of context some of the case law she cited to support her ruling.

Wells had chastised SCO for willful failure to comply with repeated court orders to provide IBM details supporting 187 of its claims. The magistrate dismissed SCO's contentions that it needed access to IBM engineers to flesh out its suspicions that IBM had illegally contributed its intellectual property to Linux.

At the time, SCO spokesman Blake Stowell allowed that Wells had dealt his company's case scheduled for trial next February a heavy blow: "If two thirds of your case is stricken, then it is a pretty serious matter."

Still, he argued the remaining one third of the nearly 300 claims made by SCO still left a viable case, though details of many of those allegations remain under seal to protect proprietary data.

Rob Enderle, chief analyst for the Enderle Group, said odds were against Kimball supporting SCO's challenge to Wells' ruling and the Utah company's previous publicity campaigns trumpeting its allegations would not help it now.

"Given SCO's public behavior it is hard to imagine a judge anywhere on the planet

hasn't already formed a somewhat negative opinion on them, which clearly is problematic now," he said.

Still, Enderle speculated that Kimball could reinstate some of the dismissed allegations, making the appeal worth the effort.

Yankee Group analyst Laura DiDio said that appeal or not, "SCO has failed to provide specific evidence

to support its claims that IBM purloined its Unix code [and] SCO is under increasing pressure to produce concrete evidence and do it quickly.

"Should the appeal fail, it will be a devastating blow to SCO," she added.

In the wake of Wells' decision, SCO stock has lost 37 percent of its value. Shares closed at $2.58 a copy Thursday, down 2 cents.

bmims@sltrib.com

SCO versus IBM

* SCO sued IBM in March 2003, claiming its Unix code illegally showed up in IBM-distributed Linux applications. SCO wants $5 billion in damages.

* SCO is appealing a federal magistrate's decision striking two-thirds of its specific allegations in the case.

* If the appeal to U.S. District Judge Dale Kimball fails, the Lindon company will be left with a shell of the case it has spent tens of millions of dollars to bring.