NOTES

WELDING THE HOOD SHUT: THE COPYRIGHTABILITY OF OPERATIONAL OUTPUTS AND THE SOFTWARE AFTERMARKET IN MAINTENANCE AND OPERATIONS

Matthew J. Leary*

INTRODUCTION ............................................................................................. 1390

I. BACKGROUND ON OPERATIONAL OUTPUTS ...................................... 1392
   A. The Importance of the Copyrightability of Operational Outputs to the Support Aftermarket......................... 1394
   1. Large-Scale Software Maintenance and Operations Management – Background........................................ 1394
   2. How ISOs Use Operational Outputs......................................... 1395
   3. Results of Copyright for Operational Outputs............................. 1396
   B. Business and Contractual Considerations................................. 1397

II. ARE OPERATIONAL OUTPUTS COPYRIGHTABLE SUBJECT MATTER? 1399
   A. Outputs as Derivative Works ..................................................... 1401
      1. Arguments Against Characterizing Operational Outputs as Derivative Works ........................................ 1403
      2. Minimal Originality for Derivative Works.......................... 1405
   B. Outputs as Factual Compilations ............................................. 1406
   C. Outputs as User Interface ....................................................... 1409
   D. Ownership of Computer-Generated Works ................................. 1410

III. DOES AFTERMARKET USE OF OPERATIONAL OUTPUTS VIOLATE COPYRIGHT?...................................................... 1412
   A. Infringement of Operational Outputs as Literal Copies of the Source Code .................................................. 1413
   B. Infringement of Operational Outputs as Copies of Non-Literal Elements of the Underlying Source Code ......... 1414
   C. Infringement of Operational Outputs as Derivative Works ...... 1417
   D. Infringement of Operational Outputs as Compilations .......... 1420
   E. Copyright Exceptions and Defenses ............................................ 1421
      1. 17 U.S.C. § 117(a)(1) .......................................................... 1421
      2. 17 U.S.C. § 117(c) ........................................................... 1424
      3. Fair Use ............................................................................. 1425
      4. Merger Doctrine ............................................................... 1428

IV. THE DMCA AND THE AFTERMARKET ........................................... 1429
   A. The 17 U.S.C. § 1201 Anti-Circumvention Provision .............. 1429

V. POSSIBLE REMEDIAL MEASURES................................................. 1434
INTRODUCTION

Imagine if General Motors (GM) forced its customers to service their cars only at official GM dealerships – and that GM would sue any local garage that even attempted to open the hood. This Note argues that copyright law may enable hardware and software vendors to do just that.

Independent service organizations (ISOs) are increasingly being used to provide aftermarket service and support for large-scale software systems and are beginning to displace the software’s original manufacturers in these “care-and-feeding” roles. ISOs are analogous to independent car repair shops, maintaining products initially manufactured by other companies at lower prices than those offered by the original manufacturer. Even as enterprise software vendors search for new growth areas when sales saturate in their primary markets, profit margins are approaching seventy percent in software maintenance contracts. These vendors must be concerned about ISOs encroaching into such a lucrative market.

Hardware manufacturers are already using intellectual property (IP) law to protect themselves from a similar displacement trend in the hardware

* J.D. Candidate, Boston University, 2006.

1 “Large-scale” or “enterprise” systems have varied meanings, but are readily distinguished from smaller programs, e.g., spreadsheets. Large-scale systems are usually comprised of multiple programs and may run the majority of an enterprise’s operations, some supporting thousands of end-users across multiple continents. The most important characteristic for this Note, however, is that large-scale systems are complex enough to produce operational outputs for use by support engineers. Complex hardware often contains software producing similar outputs.

2 David Bank, ‘Rebel’ Customers May Cut into Profits at Big Software Firms, WALL ST. J., Sept. 30, 2004, at B61 (“Corporate and government software customers are rebelling against the rising annual ‘maintenance’ fees that companies such as PeopleSoft, Oracle and SAP AG charge . . . .”); Pimm Fox, Don’t Own Your IT, COMPUTERWORLD, Apr. 19, 2004, at 18 (reporting benefits that IT outsourcing has brought to companies like H&R Block); James Riley & Michael Sainsbury, Jobs Risk as Telstra Quits GSA, THE AUSTRALIAN, Aug. 28, 2003, at 23 (asserting that “basic application software maintenance tends to be the first area outsourced to third parties”).

3 See Nick Langley, Oracle’s Revival Raises Skills’ Value; In the SSL/CW List, Oracle E-Business Suite is Number 4, COMPUTER WEEKLY, Mar. 2, 2004, at 54 (observing that Oracle, like other suppliers, is targeting small and mid-sized customers now that the enterprise market is saturated); Malcolm Wheatley, Decisions; Mid-Market Vendors; The Grass is Greener, FIN. DIRECTOR, Sept. 1, 2003, at 5 (describing the enterprise software market as “a rapidly maturing market that is approaching saturation”). “[J]uicy” margins of seventy percent or more are giving ISOs an opportunity to undercut the software vendors to gain a share of “the software industry’s most profitable revenue stream.” Bank, supra note 2, at B5.
aftermarket. These companies have been largely successful in using IP law to prevent ISOS from providing maintenance services, at least prior to the enactment of 17 U.S.C. § 117(c) (2000) – a legislative effort to protect third-party hardware maintenance. Software vendors may not be far behind.

Operational outputs are message streams that large-scale software systems create to narrate and record their systems’ operations. ISO support engineers use these outputs to analyze and resolve problems in their customer’s software. Operational outputs are critical breadcrumb trails for determining what went wrong when these complex systems fail. Without access to copies of this output, it is often impossible for engineers to provide maintenance and service.

If software providers can copyright operational outputs, ISOS may be unable to compete against software manufacturers because the copyright essentially “welds the hood shut” on software systems, preventing third parties from getting the access they need to provide service. Whether outputs are

---

4 See, e.g., Triad Sys. Corp. v. Se. Express Co., 64 F.3d 1330, 1333 (9th Cir. 1995) (affirming preliminary injunction for automotive store management hardware/software manufacturer against ISO who used the manufacturer’s diagnostic software in violation of the customer’s license); Data Gen. Corp. v. Grumman Sys. Support Corp., 36 F.3d 1147, 1152 (1st Cir. 1994) (affirming $27 million in damages for computer manufacturer against ISO who copied the manufacturer’s diagnostic software over ISO’s antitrust defenses); MAI Sys. Corp. v. Peak Computer, Inc., 991 F.2d 511, 523-24 (9th Cir. 1993) (affirming preliminary injunction for computer and operating-system manufacturer against ISO who used the manufacturer’s diagnostic software); Serv. & Training, Inc. v. Data Gen. Corp., 963 F.2d 680, 691 (4th Cir. 1992) (affirming computer manufacturer’s injunction against ISO who copied the manufacturer’s diagnostic software over ISO’s antitrust defenses); see also 17 U.S.C. § 117(c) (2000) (providing a safe harbor for infringement for purposes of machine maintenance or repair); Storage Tech. Corp. v. Custom Hardware Eng’g & Consulting, Inc., No. 02-12102-RWZ, 2004 WL 1497688 (D. Mass. July 2, 2004) (granting a preliminary injunction to tape drive manufacturer who sued to enjoin ISO’s use of manufacturer’s diagnostic output generation software), vacated, 421 F.3d 1307 (Fed. Cir. 2005) (vacating the district court’s injunction after concluding that Custom Hardware’s copying was permitted by § 117(c)).

Interest in protecting the service market is not surprising given that at least forty percent of the life-cycle cost of software is maintenance, not the original price. See John Kavanagh, Understanding Software Is Vital to Stop Maintenance Costs Soaring, COMPUTER WEEKLY, Feb. 1, 2005, at 38 (lamenting that “[s]oftware maintenance gobbles up 90% of the cost of software over its lifetime”); Pamela Samuelson, Modifying Copyrighted Software: Adjusting Copyright Doctrine to Accommodate a Technology, 28 JURIMETRICS J. 179, 183 n.16 (1988) (estimating that maintenance and enhancement activities range from forty to seventy-five percent of life cycle costs).

5 In order to effectively support these systems, ISOS need access to diagnostic information about where problems have occurred or are likely to occur. See Storage Tech, 2004 WL 1497688, at **2-3; discussion infra Part I.A.2.

6 See Stephen M. McJohn, Fair Use of Copyrighted Software, 28 RUTGERS L.J. 593, 595-96 (1997) (commenting that if ISOS are forced to obtain the manufacturer’s permission to
Copyrightable is also likely to be dispositive for causes of action under the Digital Millennium Copyright Act (DMCA).\(^7\) Despite explicit Congressional intent to protect ISOs from harm due to an overzealous application of copyright law,\(^8\) the current regime appears unable to provide this protection to hardware or software servicers, and some ISOs are already suffering.\(^9\) This Note examines whether operational outputs may be copyrighted and the extent to which copyright law and the DMCA may be used to exclude ISOs from the aftermarket. The Note begins by describing operational outputs and their importance in software operations. It then examines whether outputs are copyrightable and whether copying them constitutes infringement. Finally, this Note discusses possible defenses to infringement and issues related to the DMCA and concludes that copyright law must change if ISOs are to compete in the marketplace.

Because many hardware platforms are controlled by software, most of the arguments in this Note should also be applicable to hardware ISOs who use operational outputs from embedded software in order to service hardware devices.\(^10\) As the line between hardware and software continues to blur, this discussion becomes even more generally applicable.\(^11\)

I. BACKGROUND ON OPERATIONAL OUTPUTS\(^12\)

Systems generate operational outputs for purposes of maintenance, operations support, debugging, and statistical analysis. They provide a use its operational outputs, “then the software maker can either require licensing fees from [an ISO] or have an exclusive market to service such computers”).

\(^7\) 17 U.S.C. §§ 1201-1205 (2000); see also discussion infra Part IV (describing the DMCA’s prohibition on circumventing controls which control access to copyrighted works).


\(^9\) See discussion infra Part I.A.3.


\(^11\) Id. (warning that attempts to grant differential patent treatment based on whether a particular innovation can be implemented in software is becoming increasingly untenable).

\(^12\) The author has worked in software systems engineering for twenty years. The commentary in this section is based on his experience developing and using operational outputs in a variety of different industries. The author has found that the concepts outlined here are ubiquitous in large-scale systems across industries as diverse as manufacturing, travel, telecom, and the military.
narrative of the system’s operations, including errors and warnings, informational messages, system event logs, program audit information, and other data. Unlike outputs such as user interfaces or paychecks printed by an accounting program, generating operational output is not the software’s primary purpose.

Operational messages contain several elements. Each typically has a timestamp to allow correlation with events occurring in other systems and in the real world. Messages also establish context by indicating which subroutine or method\footnote{“Methods” are to object-oriented programming what subroutines are to procedural programming languages; they are both small sub-programs that are the building blocks for the larger program as a whole.} generated them and by displaying the data with which the subroutine or method was working. Some messages simply note the occurrence of an event, while others report errors or warnings. An example of the actual operational output from a large software system is included in the Appendix. A condensed and simplified version illustrating the salient characteristics follows:

```

2003-04-24 – 21:13:23.914 FindCustomer( “Jane”, “Smith”, “80213”) –Could not find associated billing record number 38954. RECOMMENDATION - Record may have been archived - Check using archive utility. The customer database integrity tool may be used to provide a default fix for this problem. (CustomerBilling.C, Line 626)

2003-04-24 – 21:13:24.016 UpdateCustomerAccount( 846728, “80213”, “Comment – Customer deleted after her credit card was lost. She called back about 30 minutes later saying that she found it, and I thought we shouldn’t charge her an additional startup fee again – Could someone on the India team please fix the database for this? - Marjorie” ) – FATAL ERROR 3456 – Database Inconsistency: FindCustomer failed. (CustomerBilling.C, Line 4984)
```

In the third message above, the “UpdateCustomerAccount” subroutine encountered an error when trying to find data on customer number 846728. An engineer assigned to fix the error can see the data that was in use when the problem occurred, including user-created data such as the credit card narrative.

Taken together, these messages essentially describe the structure of the program. The output traces subroutines calling other subroutines from the “top” of the program (the most abstract level) down to the lowest-level
functions. The output describes the program’s internal interfaces by including the parameters that are passed up and down this chain of subroutines. The example above shows that a routine called UpdateCustomerAccount called a subroutine named FindCustomer which in turn encountered a database error at an even lower level. The failure of UpdateCustomerAccount was actually triggered by a failure in FindCustomer.

Most messages are verbatim copies of fragments of the source code, interspersed with parameterized information generated or received as the program is run, for example, customer information. Thus operational outputs mirror the actual source code of the program itself. The program code that generated the second message above (with the parameterized information in boldface) might be:

```java
WRITE_LOG( System.CurrentDateTime, “FindCustomer”,
customer.FirstName, customer.FirstName,
customer.Zipcode, transaction.Comment,
“Could not find associated billing record number”
(* )customer.RecordNum, “. RECOMMENDATION - Record may have been archived - Check using archive utility. The customer database integrity tool may be used to provide a default fix for this problem.”,FILENAME, FILELINE);
```

Although this Note will attempt to avoid making any specific assumptions about how a particular output message will look, the above samples should be sufficient where necessary for this analysis.

A. The Importance of the Copyrightability of Operational Outputs to the Support Aftermarket

Operational outputs are essential to support large-scale software and hardware systems because operators rely on them to understand the systems’ inner workings and failures.

1. Large-Scale Software Maintenance and Operations Management – Background

Like complex hardware, complex software may fail when faced with the occasional confluence of unexpected events. Software “maintenance” involves diagnosing why the software responded inadequately to the conditions it encountered and possibly rewriting source code and/or data to remedy the

---

14 An interface is the information (and its format) that is input to and output from the subroutine or method.

15 “Parameterized” in this context simply means that the source code contains blanks – parameters – that will be filled in with information at run time. These “run-time” parameters in the example include elements like customer name, zip code, end-user comments, and record number.
problem. The goals are to fix the fallout from the earlier failure and to prevent future failures. Operations management is a broader concept, encompassing maintenance and any other responsibilities required to ensure the successful operation of the system in its operating environment.

No amount of pre-deployment testing will completely eliminate software problems in a large system. The possible combinations and permutations of real-world dynamics make testing all scenarios a numeric impossibility. Variations in system load and user volume may create situations that are possible, yet occur rarely if at all. Reproducing or testing a problem might require orchestrating the interactions of hundreds of users. The low frequency of failure and high cost of simulating failure conditions makes testing for such situations (let alone preparing for them) unlikely.

This complexity also means that software problems do not necessarily decrease over time even as more problems are corrected. Individual programs making up large-scale systems may behave differently or present different interface characteristics over time. As these partner programs change, the programs interacting with them must be maintained to adapt to the changes.

Further, even when systems behave correctly, operational outputs are critical because the systems may experience physical failures. Due to finite storage capabilities, high transaction rates, and the need for systems to store transaction information for long periods of time, elements of software systems, like tape drives and hard discs, occasionally fill up. Although these failures are often easier to predict, the human element makes them inevitable.

2. How ISOs Use Operational Outputs

Because engineers cannot predict failures, they find themselves continually picking up the pieces afterward. For this reason, it is crucial that the software logs what it is doing at all times, especially when it experiences problems.

An end-user might see an error revealing no more than “Unexpected failure adding new customer – Notify support staff,” but engineers responsible for fixing this problem may need much more information. The problem might be as simple as a missing data field, or as complex as a failure in the interaction of dozens of systems. Without detailed operational output about each system and the information flowing between them, there are not enough indicators to determine what went wrong.

ISOs may use operational outputs to make repairs themselves or may use them solely to diagnose problems, leaving actual repairs to the licensee. If the problem’s source is a defect in the software, the manufacturer will typically have to make the final repair, often under a maintenance contract. Several courts have decided the issue of whether ISOs can use and make copies of the licensee’s diagnostic software to make these repairs.16 This Note will not

explore the issue of whether ISOs can access or create copies of diagnostic software, but will instead focus on whether ISOs may make copies of operational outputs. As discussed below, it is the copyrightability of outputs that will ultimately decide whether ISOs can operate legally.

3. Results of Copyright for Operational Outputs

The rise of large-scale enterprise software systems is relatively new, and the use of the DMCA and copyright law with respect to these systems is therefore largely unexplored.17 Recent cases such as Storage Technology Corp. v. Custom Hardware Engineering & Consulting, Inc. (“Storage Tech”),18 however, show that legal actions utilizing copyright law and the DMCA to limit ISOs’ access to operational outputs are still having an effect in the hardware aftermarket.

Storage Tech is one of the first DMCA cases dealing with a combination of these laws and the maintenance of hardware systems. The district court’s decision to issue a preliminary injunction – later vacated – did not reach the issue of whether copying of the plaintiff’s diagnostic information logs could constitute copyright infringement.19 This “present[ed] more difficult questions” that the court did not have to answer because it felt the plaintiff had

that ISO copied the manufacturer’s diagnostic software for MRI equipment); Data Gen. Corp. v. Grumman Sys. Support Corp., 36 F.3d 1147, 1181 (1st Cir. 1994) (finding infringement of service software over antitrust counterclaims in part because of insufficient evidence that the manufacturer’s software was tied to its services).


18 No. 02-12102-RWZ, 2004 WL 1497688 (D. Mass. July 2, 2004) (enjoining defendant from providing service for more than a year before the injunction was overturned), vacated, 421 F.3d 1307 (Fed. Cir. 2005).

19 The Federal Circuit vacated the original injunction on a somewhat technical basis. Because the plaintiff’s “maintenance code [was] so entangled with the functional code that the entire code must be loaded into RAM [random access memory] for the machine to function at all,” the court found the defendant’s affirmative defense against infringement was valid. Storage Tech. Corp. v. Custom Hardware Eng’g & Consulting, Inc, 421 F.3d 1307, 1314 (Fed. Cir. 2005). This is a technicality because some manufacturers may separate their diagnostic software from the startup software, and the Federal Circuit noted that “separate ‘freestanding programs’ that load into (RAM) upon startup clearly may not be accessed under section 117(c)(2).” Id.
If these outputs can be copyrighted by hardware and software providers, ISOs may find it impossible to provide support alternatives to what has become an increasingly smaller set of enterprise software manufacturers.21 Recent DMCA cases indicate this copyright question may control both infringement and DMCA actions brought against ISOs.22

B. Business and Contractual Considerations

The business context in which ISOs, end-users, and software manufacturers interact may play an important role in determining the contours of any infringement action. Software manufacturers probably cannot prohibit licensees from using and copying operational outputs.23 To use outputs when resolving problems themselves, licensees must at least make a copy of the output just to view it on a monitor screen.24 Software manufacturers may, however, seek to disallow ISOs from making copies. After all, it is the market erosion in maintenance caused by firms other than the software licensee to

20 Storage Tech, 2004 WL 1497688, at *3 n.3. The district court enjoined the ISO defendants partially on the basis of the DMCA because the ISO had allegedly circumvented a security measure in order to copy the plaintiff's diagnostic code inside the plaintiff's tape-storage device. Id. at *5.

21 John Edwards, Thumbscrew, 2.0, CFO MAGAZINE, Feb. 2005, at 23: [T]he ongoing consolidation in the software industry is going to make it nearly impossible for finance chiefs to rein in the total costs of enterprise applications. Now that Oracle’s acquisition of PeopleSoft is complete, for instance, large multinational corporations have two – count 'em, two – viable ERP vendors to choose from.

22 See Storage Tech, 421 F.3d at 1314 (concluding that “courts generally have found a violation of the DMCA only when the alleged access was intertwined with a right protected by the Copyright Act”); Chamberlain Group, Inc. v. Skylink Techs., Inc., 381 F.3d 1178, 1204 (Fed. Cir. 2004) (finding unauthorized use of copyrighted software to be crux of DMCA cases), cert. denied, 125 S. Ct. 1669 (2005); see also discussion infra Part IV.

23 See McJohn, supra note 6, at 615 (reasoning that customers presumably have the right to make copies of software into RAM because “there would be no point in licensing software without the ability to use it”); see also Pamela Samuelson, Allocating Ownership Rights in Computer-Generated Works, 47 U. PITT. L. REV. 1185, 1219 (1986) (asserting that Congress has rejected a right of exclusive use for semiconductor chip designs partly because of the objection to a use right being included in a copyright-like statute).

24 Video displays have their own RAM, so in order to view information, a copy of that information must necessarily be copied into video memory. Wiener v. NEC Elecs., Inc., No. C 91-20843 JW, 1995 U.S. Dist. LEXIS 20659, at *2 (N.D. Cal., July 17, 1995) (announcing parties’ stipulation of use of video RAM (VRAM)); In re DRAMs of One Megabit and Above from the Republic of Korea, Investigation No. 731-TA-556, 1993 ITC LEXIS 232 n.16 (ITC 1993) (describing operation of VRAM). Because RAM copies are potentially infringing, just viewing them may be a problem if unauthorized. MAI Sys. Corp. v. Peak Computer, Inc., 991 F.2d 511, 519 (9th Cir. 1993) (“[T]he copy made in RAM is ‘fixed’ and qualifies as a copy under the Copyright Act.”).
which manufacturers such as Storage Technology object. 25

An obvious question is why ISOs would not themselves have the same rights to use and copy the operational outputs as the licensee. 26 Scholars have posed this same question, 27 but courts have often found ISOs liable for running licensed software even on the customer’s own computer for service purposes. 28 These courts typically find copyright infringement because unauthorized “copying” occurred when the software was copied into RAM. 29 Most cases on this point have involved a purportedly separate diagnostic program, and it is running this program, not copying its output, that constitutes infringement. 30 The proper focus should instead have been on preventing copying of the programs’ outputs. After all, if the licensees have the right to load and run the diagnostic programs (as most do), they could avoid potential legal problems involving unauthorized copying of the software by executing

25 See generally Storage Tech, 2004 WL 1497688 (enjoining third-party use of tape driver manufacturer’s diagnostic output and generation software).

26 See McJohn, supra note 6, at 617 (reasoning that in providing servicing, the ISO is doing only what the licensee is presumably entitled to do).

27 See Llewellyn Joseph Gibbons, Entrepreneurial Copyright Fair Use: Let the Independent Contractor Stand in the Shoes of the User, 57 Ark. L. Rev. 539, 546 (2004) (arguing that modern realities militate for either broader fair use or legislative changes to allow independent contractors to “stand in the shoes” of the software licensees who hire them).

28 See Micro-Sparc, Inc. v. Amtype Corp., 592 F. Supp. 33, 36 (D. Mass. 1984) (rejecting derivative rights for independent contractors to whom software owners had purportedly delegated their rights in a § 117 context); Marina Lao, Unilateral Refusals to Sell or License Intellectual Property and the Antitrust Duty to Deal, 9 Cornell J.L. & Pub. Pol’y 193, 205-06 (1999) (citing cases that allowed infringement actions against third-party ISOs in spite of customers’ abilities to run the software at issue). But see Storage Tech, 421 F.3d 1307, at 1315-16 (agreeing that “while the defendants ‘are engaged in consulting services on behalf of [the licensee], Defendants’ activities are sheltered under Norwest’s license rights’” (citing Hogan Sys., Inc. v. Cybersource Int’l, Inc., No. 3:96-CV-2083-H, 1997 WL 311526, at *4 (N.D. Tex. June 2, 1997), aff’d, 158 F.3d 319 (5th Cir. 1998))). The Storage Tech court noted that the plaintiff “could have drafted its license agreement to explicitly disallow copying by third parties through activation of the equipment owners’ machines.” Id. at *9.

29 Lao, supra note 28, at 206. But see Hogan Sys., Inc. v. Cybersource Int’l, Inc., 158 F.3d 319, 324 (5th Cir. 1998) (construing contract such that whatever licensee could legitimately do under contract, it could hire contractor to do).

30 See supra note 4 and accompanying text (providing examples of cases involving ISOs who copy diagnostic software). There are concerns that manufacturers may be strategically trying to frame the output generators as distinct programs that are sold separately in order to strengthen their cases. This may be a subterfuge to ensure that the copying question involves the software, not the output, since the software allegedly needs to be loaded in order to produce the output. Telephone Interview with David York, President, Custom Hardware Eng’g & Consulting, Inc., in Fenton, Mo. (Nov. 2004). Mr. York is a defendant in Storage Tech, 421 F.3d 1307.
the programs themselves and sending the generated outputs to their ISOs.\footnote{See, e.g., NLFC, Inc. v. Devcom Mid-Am., Inc., 45 F.3d 231, 236 (7th Cir. 1995) (deciding that viewing and printing source code to make software repairs over a modem line was not a copy for infringement purposes); Hogan, 158 F.3d at 324 (determining that viewing a program’s output on a remote terminal did not in itself constitute copying).}

Yet even if the licensee creates the output, ISOs must often make their own copies for purposes including the assignment of problem resolution responsibilities within their own organizations. ISOs must achieve some semblance of independence from their customers to support the value proposition of outsourcing the work in the first place. An ability to duplicate and distribute the output within ISO organizations is essential to this independence, and thus the copyrightability of outputs is the true lynchpin for the aftermarket.

Licensing agreements may obviate the issue of output copyrightability in some contexts. Vendors and customers may contractually determine whether outputs may be copied. In some circuits, software purchasers may contract away statutory rights such as the right to reverse-engineer.\footnote{See Bowers v. Baystate Techs., Inc., 320 F.3d 1317, 1325-26 (Fed. Cir. 2003) (finding private parties “free to contractually forego the limited ability to reverse engineer a software product under the exemptions of the Copyright Act”); ProCD, Inc. v. Zeidenberg, 908 F. Supp. 640, 650 (W.D. Wis. 1996) (concluding that shrink wrap licenses generally include reverse-engineering prohibitions which are enforceable), rev’d on other grounds, 86 F.3d 1447 (7th Cir. 1996). But see Vault Corp. v. Quaid Software Ltd., 847 F.2d 255, 268-69 (5th Cir. 1988) (finding licensing agreements including reverse-engineering prohibitions to be enforceable under Louisiana’s License Act).}

Customers might negotiate with vendors to allow ISOs to copy operational outputs. But this negotiation would be meaningless until the question of copyright in outputs is determined. For this reason, this Note will not explore the preemption issues surrounding contractual limitations on software purchaser’s rights.

II. ARE OPERATIONAL OUTPUTS COPYRIGHTABLE SUBJECT MATTER?

Computer outputs are generally considered copyrightable subject matter.\footnote{See, e.g., MiTek Holdings, Inc. v. Arce Eng’g Co., 89 F.3d 1548, 1558 (11th Cir. 1996) (adopting virtual identity infringement standard when comparing architectural program user interfaces); Apple Computer, Inc. v. Microsoft Corp., 35 F.3d 1435, 1446 (9th Cir. 1994) (positing that an infringement action could be mounted if defendant had copied the plaintiff’s unique selection and arrangement of interface features).} Most cases dealing with the question of output copyrightability, however, have been concerned with the primary graphical user interfaces (GUIs), not operational outputs.\footnote{See, e.g., Lotus Dev. Corp. v. Borland Int’l, Inc., 49 F.3d 807, 819 (1st Cir. 1995) (deciding on copyrightability of spreadsheet command menu hierarchy); Apple Computer, Inc. v. Microsoft Corp., 799 F. Supp. 1006, 1017 (N.D. Cal. 1992) (comparing operating system graphical user interfaces), aff’d, 35 F.3d 1435 (9th Cir. 1994); Data East USA, Inc. v. Epyx, Inc., 862 F.2d 204, 209 (9th Cir. 1988) (examining karate video game display).}
The case most closely addressing the question of operational outputs is *Torah Soft Ltd. v. Drosnin*. The Torah Soft program allowed users to enter search information and print biblical text related to the search in a special matrix format. The main issue of the case was whether copying the output of the program constituted copyright infringement. Although Torah Soft’s output was directed at its end-user, not support engineers, it contained the same mixture of factual textual information with software-generated modifications and formatting that characterizes operational outputs. The court found that the software’s output was not protectable expression because the algorithm producing it lacked sufficient creativity. The underlying code that created the output’s formatting and arrangement was too functional and commonplace to warrant copyright protection for the output. Religious rules determined the functional algorithm of the Torah Soft program, and the court acknowledged that, more generally, the “business practices and technical requirements of the end user” may so limit the originality of the software as to prohibit protection.

Although operational outputs also may be driven by business practices and technical requirements, they have several characteristics that distinguish them from Torah Soft’s output. Operational outputs contain significant portions of the underlying (and copyrighted) source code, while Torah Soft’s output was mostly phrases from its public-domain database of the Hebrew bible. Expressive (and arguably copyrightable) elements contributed by the end-user and the software developer may be interspersed within operational outputs. Design decisions embodied in outputs are not necessarily determined solely by functional concerns and may be more novel than the religious rules that controlled Torah Soft. For these reasons, the *Torah Soft* holding may be inapplicable to operational outputs.

Since *Torah Soft* does not appear to give much guidance, the issue of copyrightability of operational outputs requires analysis from first principles.

The GUI interface of a program is typically directed at the end user, not the operational community, and as such is designed to be more “user friendly.”

---

35 136 F. Supp. 2d 276 (S.D.N.Y. 2001) (determining that output of bible software was not copyrightable).
36 Id. at 289.
37 Id. at 282.
38 Id. at 292 (concluding that the Software’s ability to disregard spaces between words was a function “not sufficiently original or creative to be worthy of protection”).
39 Id. at 286-88, 290. The court determined that whether the outputs were copyrightable “turns on whether the Software and the Database contain protectable elements.” Id. at 283-84. The court offered no precedent supporting this assertion.
40 Id. at 285 (quoting *Melville B. Nimmer & David Nimmer, 4 Nimmer on Copyright § 13.03(F)(3)(d) (1998)*).
41 Id. at 286 (“Material found in the public domain . . . is free for the taking and cannot be appropriated by a single author even though it is included in a copyrighted work.” (quoting Computer Assocs. Int’l Inc. v. Altai, Inc., 982 F.2d 693, 710 (2d Cir. 1992))).
“Copyright protection subsists . . . in original works of authorship fixed in any tangible medium of expression . . . from which they can be perceived, reproduced, or otherwise communicated . . . .” 42 Operational outputs are generally “fixed in a tangible medium of expression.” 43 Whether they are “original works of authorship” may depend on whether they are characterized as creative, derivative, or compiled works. 44 Although all these works are entitled to copyright protection, each receives progressively thinner protection. 45 However, regardless of how outputs are framed, they appear to qualify for some copyright protection.

A. Outputs as Derivative Works

The 1976 Copyright Act defines a derivative work as “consisting of editorial revisions, annotations, elaborations, or other modifications which, as a whole,


43 See Richard S. Vermut, File Caching on the Internet: Technical Infringement or Safeguard for Efficient Network Operation?, 4 J. INTELL. PROP. L. 273, 311 (1997) (reasoning that works stored on hard disks of computers are “fixed”). Even if the outputs are viewed remotely, the presence of the output’s embodiment in memory constitutes “fixation.” Williams Elecs., Inc. v. Artic Int’l, Inc., 685 F.2d 870, 874 (3d Cir. 1982) (finding that Defender video game output was “fixed” in Read-Only Memory (ROM)); see also Torah Soft, 136 F. Supp. 2d at 283 (determining that “[a]lthough the [outputs] do not appear either in the Software or the Database, they are ‘fixed’ insofar as the output is repeatable whenever the input is identical”).

44 There are arguments that operational outputs are either derivative works or compilations. See discussion infra Part II.A-C. Software may be considered a creative work. See Storm Impact, Inc. v. Software of the Month Club, 13 F. Supp. 2d 782, 789 (N.D. Ill. 1998) (regarding computer games as creative works); Advanced Computer Servs. of Mich., Inc. v. MAI Sys. Corp., 845 F. Supp. 356, 365 (E.D. Va. 1994) (describing plaintiff’s software, although used for a functional purpose, as “essentially a creative work”); Philip J. Weiser, The Internet, Innovation, and Intellectual Property Policy, 103 COLUM. L. REV. 534, 539 (2003) (asserting that “computer programs are eligible for [copyright] protection . . . as creative works of authorship”). Thus to the extent that outputs are seen as literal copies of the underlying software code, they might also be considered creative works. See discussion infra Part III.A. Without the benefit of this relationship, however, non-audio-visual program output is not the type work typically labeled “creative.” Melville B. Nimmer & David Nimmer, 1 NIMMER ON COPYRIGHT § 2.11 (2005) [hereinafter NIMMER ON COPYRIGHT] (using a novel as the typical creative work); see also Flack v. Friends of Queen Catherine, Inc., 139 F. Supp. 2d 526, 533 (S.D.N.Y. 2001) (listing movies, newspapers, and magazines as creative works); Nash v. CBS, Inc., 691 F. Supp. 140, 141 (N.D. Ill. 1988) (listing poetry and art as creative works, unlike “pure facts,” which lie at the opposite end of the creative “continuum” and are not eligible for copyright protection).

45 Warren Publ’g, Inc. v. Microdos Data Corp., 115 F.3d 1509, 1515 n.16 (11th Cir. 1997) (en banc) (explaining that derivative works receive thinner protection than creative works, but that compilations receive the “thinnest” protection – below that of derivative works).
represent an original work of authorship.”46 In Stewart v. Abend, the Supreme Court defined a derivative work as one based upon a preexisting work, such as an abridgement or condensation or “any other form in which a work may be recast, transformed, or adapted.”47

It is reasonable to characterize operational outputs as derivative works.48 Outputs are nothing more than a combination of text messages from the program’s source code with the “blanks” in the messages replaced with data that has been received or calculated by the system.49 For example, much of the second output message documented in Part I of this Note is a verbatim copy of the source code modified by inserting the customer record number into the text.

Under Stewart’s § 101 standard, operational output is an “abridgment” (the subset of the parameterized source code that produces the output) “modified” by having the parameters replaced with values derived from input data. No case appears to have directly addressed this question, but some courts have found similarly modified abridgments to be derivative works.50

Alternatively, the program’s copyrighted source code may be considered a theatrical “plot” of the real-life activities (i.e., a customer having her order fulfilled) that occur when the program is run. Operational output could then be analogized as the result of the program’s plot with minor variations in the names and actions of the characters. Object-oriented-programming (OOP) techniques make this analogy quite natural. The “objects” in OOP terms are often called “actors” (i.e., “CreditCardCustomer”), whose basic capabilities and rules are established by the source-code’s design (i.e., customers can create, update, and delete accounts), and whose attributes are set by the user at runtime (such as name and social security number). Viewed as a modified “plot,” output may be copyrightable as a derivative work.51 It is true that a

48 Cf. Micro Star v. Formgen Inc., 154 F.3d 1107, 1110, 1112 (9th Cir. 1998) (finding that user-created group of “levels” in a video game that combines creativity from the user and the underlying software is a derivative work, though the conclusion was “not obvious”); see also Samuelson, supra note 23, at 1211 (likening the relationship between programs and output to that of screenplays and novels). Samuelson also argues that operational outputs are generally not derivative works. Id. at 1212-14. There is a trend toward “increasingly broad interpretation of the definition of derivative works.” Christian H. Nadan, Comment, A Proposal to Recognize Component Works: How a Teddy Bears on the Competing Ends of Copyright Law, 78 CAL. L. REV. 1633, 1640 (1990) (arguing that copyright protection should be extended to “component” works that function only in conjunction with the preexisting work).
49 See supra text accompanying note 15.
50 See Matthew Bender & Co. v. West Publ’g Co., No. 94 Civ. 0589, 1997 U.S. Dist. LEXIS 6915, at *3 (S.D.N.Y. May 19, 1997) (finding that Westlaw’s relatively minor annotations and edits to court opinions constitute derivative works).
51 See, e.g., Suntrust Bank v. Houghton Mifflin Co., 268 F.3d 1257, 1267 (11th Cir.
particular instance of operational output is likely to represent only a fraction of all the software’s possible “plots.” Regardless, if a copy of the entire plot infringes on the original, then a copy of an abridged version of that plot could be a substantially similar copy of the original. Additionally, if the plot changes are considerable, the resulting derivative work may be eligible for its own separate copyright. Although both possibilities indicate that operational outputs are copyrightable, the latter opens up the issue of ownership in the separately-copyrightable work, as later discussed in Part II.D.

1. Arguments Against Characterizing Operational Outputs as Derivative Works

Professor Pamela Samuelson gives five reasons why computer-generated output should not generally be considered a derivative work, but the particular characteristics of operational outputs confound each of them. First, Samuelson argues that Congress did not intend “derivative works” to cover all computer outputs, in part because many could be “entirely separate” from the underlying program. Operational outputs, however, typically are the program (and thus not separate at all). Much output is a verbatim copy of the program’s code, with some added factual parameters. The same reasoning applies to her second objection that “[i]n general, computer-generated works do not incorporate recognizable blocks of expression from the underlying program.”

Third, “useful” articles created from copyrighted plans such as bridges are not considered to be copyrightable derivative works of those plans. Drawing on the same doctrine, Samuelson reasons that computer outputs are useful articles created by the program’s “plans” and thus do not deserve to be derivative works. Yet, by themselves, operational outputs are no more useful

2001) (determining that minor changes in plot and character names in “The Wind Done Done” does not save the work from infringement of “Gone with the Wind,” though fair use may); Williams Elecs., Inc. v. Artic Int’l, Inc. 685 F.2d 870, 877 (3d Cir. 1982) (concluding that a video game display created in part with user interaction was still copyrightable).  

52 See discussion infra Part III.A.  
53 Micro Star, 154 F.3d at 1112 (finding that user-created “levels” in a video game that use the same characters and props as the original are an infringing derivative); Anderson v. Stallone, No. 87-0592 WDK(Gx), 1989 U.S. Dist. LEXIS 11109, at **24-25 (C.D. Cal. April 25, 1989) (determining that a new “Rocky” script that was based on the original movie’s characters was an unauthorized derivative work).  
54 Samuelson, supra note 23, at 1212-21.  
55 Id. at 1212-13.  
56 See discussion supra Part I, particularly note 15.  
57 Samuelson, supra note 23, at 1214-15.  
58 Id. at 1216-18 (observing that 17 U.S.C. § 101 (2000) differentiates between “applied art,” which is copyrightable, and “industrial design,” which is not).  
59 Id. at 1217-18 (reasoning that “to the extent that the output of a computer program is a useful article (such as a computer chip) or other uncopyrightable work (such as a chemical
than pinning up a blueprint of a bridge between two river banks. Samuelson argues that a bridge is merely an “implementation of the ideas that are embodied in the copyrighted work.” Samuelson argues that a bridge is merely an “implementation of the ideas that are embodied in the copyrighted work.”60 Operational outputs are different. They are not implementations of ideas in the source code—they are verbatim copies of segments of the source code itself.

Fourth, Samuelson asks what would happen if the computer was taken out of the equation. If a book expressed an algorithm that taught someone to create operational output by hand, the doctrine of Baker v. Seldon⁶¹ would not allow the resulting output to receive copyright, because it would just be the result of using the book’s idea—the algorithm.⁶² Samuelson reasons that this outcome should not change just because a computer automates the algorithm that implements the idea.⁶³ But the idea “taught” by large-scale software algorithms and thus protected by Baker is the automation of business processes, not the creation of operational outputs, which are at most tertiary results. Also, taking the computer out of the equation does not make sense for outputs. They can often be created only by the related computer program, as it relies upon the exact implementation of the program and may even refer to literal elements such as program modules and source-code line numbers.

Lastly, Samuelson argues that Congress could not have intended to grant copyright owners such as software developers an exclusive right to all of the byproducts of a functional copyrighted work—essentially the right to use a copyrighted work.⁶⁴ If this had been the intent, a novel created using a word processor would be a derivative of that software and could not be “used” or exploited by the novelist, belonging instead to the software manufacturer.⁶⁵ Although this argument works well for word processors, most large-scale software is “used” to implement business processes, not to generate operational outputs. Limiting the end-user’s use of the operational output is less problematic because outputs are only indirectly necessary for supporting the software’s main purpose. When the system is working correctly, prohibiting access to the operational outputs should not have any effect on the system’s usability to its end-user. More importantly, it is third-party use of the outputs

formula), it could not be considered a derivative work within the meaning of the copyright statute even if it incorporates a recognizable block of expression from the underlying program”).

⁶⁰ Id. at 1217.

⁶¹ 101 U.S. 99 (1879) (determining that copyright protects only the expression of a how-to accounting book, not the underlying system of accounting).

⁶² In the same way, Baker held that although a book on accounting was protected against copying, copyright could not be allowed to prevent readers from using the ideas that the book taught. See id. at 105.

⁶³ See Samuelson, supra note 23, at 1218.

⁶⁴ See id. at 1219-21.

⁶⁵ See id. at 1200 n.71.
that troubles manufacturers, not use by their end-users.66

Although no specific cases or doctrines exist to determine whether
operational outputs are derivative works, the Fifth Circuit has suggested that
examples of a plaintiff’s output that were copied into a defendant’s software
product might have produced an infringing derivative/compilation work.67 The
court recognized that the output would be copyrightable only to the extent that
the formats were not scènes à faire dictated by industry demand and practice,
i.e., to the extent they had sufficient originality.68 Given the above analysis, it
seems reasonable to frame outputs as derivative works, which would make
them the proper subject of the Copyright Act.69

2. Minimal Originality for Derivative Works

If operational outputs are derivative works, then we must next ask if they are
sufficiently original to deserve copyright. The Second and Ninth Circuits use
the two-pronged Durham test to determine whether a work deserves protection
as a derivative work.70 The Durham test requires that the original aspects of
the derivative (1) be more than trivial; and (2) reflect a relatively low degree of
reliance on the preexisting material, and must not in any way affect the scope
of any copyright protection in that preexisting material.71

For operational outputs, the only elements added to the underlying work by

---

66 See supra note 25 and accompanying text.
67 See Eng’g Dynamics, Inc. v. Structural Software, Inc., 26 F.3d 1335, 1346 (5th Cir.
1994) (finding that data output examples from the plaintiff’s user manual were found
copyrightable to the extent the formats were not scènes à faire dictated by industry demand
and practice). The plaintiff had also claimed that the resulting software could have been a
derivative work, though the court never reached this question. Id.
68 Id. at 1347 (remanding for a determination of whether industry demand and practice
dictated the software’s input and output formats); see also Lexmark Int’l, Inc. v. Static
Control Components, Inc.:
In the literary context, the [scènes à faire] doctrine means that certain phrases that are
‘standard, stock, . . . or that necessarily follow from a common theme or setting’ may
not obtain copyright protection. In the computer-software context, the doctrine means
that the elements of a program dictated by practical realities – e.g., by hardware
standards and mechanical specifications, software standards and compatibility
requirements, computer manufacturer design standards, target industry practices, and
standard computer programming practices – may not obtain protection.
823, 838 (10th Cir. 1993)).
69 See 17 U.S.C. § 106 (2000) (granting a copyright holder the “exclusive right” to
prepare “derivative works based upon the copyrighted work”).
70 Entm’t Research Group, Inc. v. Genesis Creative Group, Inc., 122 F.3d 1211, 1220
(9th Cir. 1997) (adopting the Durham test for derivative works), cert. denied, 523 U.S. 1021
(1998); Durham Indus. v. Tomy Corp., 630 F.2d 905, 909 (2d Cir. 1980) (describing the
two-prong test for determining protection for derivative works).
71 Durham, 630 F.2d at 909.
the end-user are mostly factual and, as such, are unlikely to meet the first prong. Any non-factual elements are typically terse, functional statements closely constrained by the business context for which they were generated, which would thus seem “trivial.” Many of the parameters included in operational outputs will be computed by the software from end-user and operational data, and as such rely heavily on the copyrighted source code.

*Durham*’s second prong, however, presents no barrier to copyrightability. It is difficult to see why a copyright in such derivative works would affect the “scope” of the preexisting source code. When the Ninth Circuit examined the *Durham* test, the court was primarily concerned with whether the derivative work’s owner could effectively block the underlying copyright holder from licensing the underlying work to others. But even if software consumers own the copyright in derivative outputs, they certainly couldn’t stop software manufacturers from selling their applications.

Thus protection for operational outputs as derivatives may turn on what level of originality is added to the text of the source code by the end-user’s parameters and data. This question will be highly fact-sensitive, and the courts have not settled on the required quantum of originality itself.

**B. Outputs as Factual Compilations**

If operational outputs do not qualify for copyright as derivative works, they might still gain protection as factual compilations. Outputs are analogous to machine-generated compilations of historical facts occurring while the program is running. 17 U.S.C. § 101 defines a compilation as “a work formed by the collection and assembling of preexisting materials or of data that are selected, coordinated, or arranged in such a way that the resulting work as a whole constitutes an original work of authorship.”

If a work’s selection, coordination, or arrangement of facts displays some minimal level of creativity and this selection or arrangement was not itself copied from another work, then the compilation constitutes an original work of authorship within the meaning of § 101. While the selection and

---

72 The type of information that users input into enterprise software is likely to be related to the business, and is typically oriented toward customer and product information.

73 *Entm’t Research Group*, 122 F.3d at 1220 (expressing concern for the rights of the preexisting work’s copyright holder).

74 Steven S. Boyd, *Deriving Originality in Derivative Works: Considering the Quantum of Originality Needed to Attain Copyright Protection in a Derivative Work*, 40 SANTA CLARA L. REV. 325, 369 (2000) (describing the “unpredictable and complex landscape of the law developing in assessing the quantum of originality necessary to create a derivative work”); see also discussion infra Part III.C (discussing the protection afforded operational outputs as derivative works).


76 *Feist*, 499 U.S. at 358 (“Originality requires only that the author make the selection or
arrangement need not be novel to be copyrightable, they cannot be “so mechanical or routine as to require no creativity whatsoever.”77 The *Feist* court outlined this threshold by finding a telephone book’s alphabetical ordering of names and phone numbers so commonplace that it did not meet even this de minimis quantum of originality.78 When testing compilations for minimum originality, some courts focus on *Feist*’s “mechanical” language, and indicate that any non-mechanical (i.e., not alphabetic or numeric) arrangement is sufficient to clear the hurdle.79 Even an author’s “random” arrangement suffices.80

Most operational outputs are arranged by time sequence. This is no more original than the alphabetic sorting of *Feist*’s phonebook, as it too is “commonplace” in the industry.81 Thus operational outputs fail the originality test with respect to arrangement.82

However, § 101 indicates that the arrangement or selection may constitute the required originality,83 and the selection of facts for inclusion in outputs often requires significant creativity. Operational outputs are collections comprised of both facts (such as names and telephone numbers), and expressions (annotations describing the operational significance of these facts) selected and arranged by the software developer. Software engineers choose

77 *Feist*, 499 U.S. at 362.

78 Id. at 362-63 (deciding that alphabetical arrangement of white pages did not qualify for copyright protection).

79 See Lipton v. Nature Co., 71 F.3d 464, 470 (2d Cir. 1995) (viewing minimal, non-mechanical arrangement of venery terms for animal groups as copyrightable); Key Publ’ns, Inc. v. Chinatown Today Publ’g Enters., Inc., 945 F.2d 509, 514 (2d Cir. 1991) (granting copyright protection to author’s selection of which businesses to include in directory).

80 See Lipton, 71 F.3d at 470 (determining that the plaintiff’s random arrangement of terms defining animal groups was sufficient to find it copyrightable).

81 *Feist*, 499 U.S. at 362-63 (deciding that alphabetical phonebook arrangement is unoriginal).

82 This analysis considers the “facts” that the programmer has arranged to be the individual lines of output (each indicating that a particular subroutine was called). If, instead, the data values within each line are the facts, this might be a closer question because their arrangement is arguably more creative.

83 17 U.S.C. § 101 (2000) (“A ‘compilation’ is a work formed by the collection and assembling of preexisting materials or of data that are selected, coordinated, or arranged in such a way that the resulting work as a whole constitutes an original work of authorship. The term ‘compilation’ includes collective works.”) (emphasis added); COHEN ET AL., COPYRIGHT IN A GLOBAL INFORMATION ECONOMY 294-95 (2002).
which factual elements to include in the output from large amounts of operational data. In fact, operators measure the quality of outputs in part on how effective they are at giving problem solvers enough information to fix problems without producing a mind-numbing deluge of data. Many systems provide user-controlled “trace levels” of output to allow the problem solver to fine-tune this selection, giving high-level information at one setting and much more detailed facts at other settings. This trace level helps balance between creating unused mountains of information and providing sufficient detail when needed. More detailed outputs can also be detrimental to a system’s performance. The selection of data to be included at each level may reflect the designer’s understanding of the business and operational communities as well as typical maintenance scenarios.

In addition to creativity in selection, the software developer must also write code, necessarily making non-“mechanical” decisions in selecting and arranging each fact placed in each line of output. Thus, it is reasonable to regard operational outputs as § 101 compilations under Feist.

No court has yet determined whether operational outputs are copyrightable compilations, but some cases have answered this question regarding other types of program output. The court in Engineering Dynamics v. Structural Software found that even output reports that had merely arranged parameters and variables as a series of columns might have been sufficient to meet Feist’s originality threshold, though there was a question of Feist’s applicability in the case. Similarly, in Digital Communications Associates, Inc. v. Softklone Distributing Corp., the court granted copyright protection to a program’s status screen. The screen consisted of a compilation of current program settings, operational output often includes “dumps” of configurable settings.

Bolstering this argument, there may be user-created expressive elements in outputs that are distinct from both the factual portions and the selection and arrangement. These elements may be copyrightable by themselves, and would make outputs more deserving of copyright than the facts-only subject matter of

84 For example, there are over a thousand types of data values in the system that implements wireless portability in the United States. From these values, the developers (which include the author) might choose only three or four as the most useful and interesting to include in the system’s trace output for a particular scenario.


86 See Eng’g Dynamics v. Structural Software, 26 F.3d 1335, 1346 (5th Cir. 1994) (“[T]o the extent that Feist’s definition of originality applies here, it appears that EDI has selected data and arranged their placement in a way that is unique and original to SACS.”).

87 659 F. Supp. 449, 462 (N.D. Ga. 1987) (determining that the “status screen, even if found to be a ‘form,’ clearly expresses and conveys information and, therefore, is copyrightable”).

88 Id. at 452.
Feist’s phonebook. For example, the output may include user-created expressive content like the credit-card narrative in Part I of this Note, or a reproduction of sections of a copyrighted self-help manual to aid the operations community in its work. The blend of uncopyrightable and copyrightable text should at least be protected against wholesale copying.89

C. Outputs as User Interface

Another way to examine operational outputs is to look at them functionally – as merely another program interface aimed at a different type of user – the ‘hidden’ user community responsible for maintenance and operations.

Several circuits hold that because user interfaces such as menu hierarchies are essential to operating the software, they are “methods of operation” or are otherwise excluded from copyright by 17 U.S.C. § 102(b) (2000).90 Yet these circuits may not necessarily withhold copyright from operational outputs. Considered as interfaces, operational outputs are very different from menu hierarchies. Outputs are not a method of operation concerning the “control” of the system, because they are not a mode of input. Using a prior analogy, they are not essential for the normal running of the program any more than a car mechanic is “essential” to the normal functioning of an automobile. Of course, a colorable argument could still be made that these outputs will be practically if not actually necessary for operation.

When determining the copyrightability of user interfaces, other circuits apply the Altai “abstraction-filtration-comparison” test91 in the belief that there may still be copyrightable expression at levels of abstraction below those described as methods of operation.92 These courts understand, however, that

89 See Lexmark Int’l, Inc. v. Static Control Components, Inc., 387 F.3d 522, 559 (6th Cir. 2004) (Feikens, J., concurring in part and dissenting in part) (finding that wholesale copying of the program containing a blend of scènes à faire and material, even if not dictated by norms or constraints, would still constitute infringement); Eng’g Dynamics, 26 F.3d 1335, 1348 (5th Cir. 1994) (remarking that even output formats that are almost uncopyrightable are protected against verbatim copying); see also discussion infra Part III (discussing protection afforded operational outputs).

90 See Lotus Dev. Corp. v. Borland Int’l, Inc., 49 F.3d 807, 815 (1st Cir. 1995) (characterizing a spreadsheet’s menu command hierarchy as a “method of operation” and thus uncopyrightable under 17 U.S.C. § 102(b)), aff’d by an equally divided court, 516 U.S. 233 (1996); see also MiTek Holdings, Inc. v. Arce Eng’g Co., 89 F.3d 1548, 1556-57 (11th Cir. 1996) (affirming district court’s holding that menu structure was an uncopyrightable process under 17 U.S.C. § 102(b), but declining to hold it uncopyrightable as a matter of law as Lotus had done).


92 See Mitel, Inc. v. Iqtel, Inc., 124 F.3d 1366, 1372 (10th Cir. 1997) (affirming denial of injunction for copyright infringement of telecommunication hardware command codes); Incredible Techs., Inc. v. Virtual Techs., Inc., 284 F. Supp. 2d 1069, 1078 (N.D. Ill. 2003)
once they filter out the uncopyrightable elements during the filtration test, there may be nothing copyrightable left to compare.\textsuperscript{93} The result of applying the \textit{Altai} test to operational outputs is inconclusive, as will be discussed in Part III.B.1.

Courts that view interfaces as methods of operation, and thus ineligible for copyright protection, have based their decisions partially on concerns that awarding copyright to user interfaces would increase training costs for software due to a proliferation of different interfaces.\textsuperscript{94} Yet the uniqueness and complexity of large systems already require specialized training for these systems. Outputs are also much more complex and expressive than standard interfaces such as the VCR “Play” button or the “P” command in the Lotus 1-2-3 menus.\textsuperscript{95}

Even if courts were to embrace the analogy between operational outputs and user interfaces, the resulting arguments for copyright protection may be weaker than those supported by framing outputs as compilations or derivative works. Commentators and courts have had difficulty applying copyright rules to user interfaces.\textsuperscript{96} Consequently, courts have been generally reluctant to grant any form of IP protection to user interfaces.\textsuperscript{97} Ultimately, courts may wind up analyzing user interfaces as factual compilations anyway.\textsuperscript{98} Arguments for protecting interfaces often rely on distinctive characteristics, such as visual arrangement, that text-based operational outputs do not have.\textsuperscript{99}

D. \textit{Ownership of Computer-Generated Works}

If courts see outputs as derivative works or factual compilations, then determining who or what created the works becomes critical because copyright (determining that organization of an interface’s buttons and its instructions and graphics are not necessarily uncopyrightable).

\textsuperscript{93} See \textit{Feist Publ’ns v. Rural Tel. Serv. Co.}, 499 U.S. 340, 359 (1991) (“There remains a narrow category of works in which the creative spark is utterly lacking or so trivial as to be virtually non-existent.”).

\textsuperscript{94} See \textit{Lotus}, 49 F.3d at 818 (basing its decision partly on the policy argument that requiring users to learn a new command syntax for every program would be “absurd”).

\textsuperscript{95} The \textit{Lotus} court compared the commands in the program’s menu hierarchy to the ubiquitous “Record, Play, Reverse, Fast Forward, Pause, Stop/Eject” buttons on a VCR. \textit{Id.} at 817.


\textsuperscript{97} \textit{Id.} at 157 (suggesting trade-dress as a possible protection strategy for GUIs).

\textsuperscript{98} Cf. \textit{Eng’g Dynamics, Inc. v. Structural Software, Inc.}, 26 F.3d 1335, 1346 (5th Cir. 1994) (contending that while user interface was likely not a factual compilation, the \textit{Feist} analysis for copyrightability of compilations might still apply).

ownership is a prerequisite for bringing an infringement suit. Several commentators have suggested that computer users, not the software authors, should own computer output. The National Commission on New Technological Uses of Copyrighted Works’ (“CONTU”) Final Report said that the user should “obviously” own any output, at least from the 1976 perspective that likened software to a non-intelligent typewriter or camera. After all, few would argue that a photographer’s expressive contributions should be attributed to and owned by a camera manufacturer. Scholars like Professor Samuelson argue for user ownership even when the software is more sophisticated.

Samuelson makes an exception, however, for instances in which the generated work incorporates a substantial portion of expression from the underlying software code. Because a substantial portion of operational output is really just a verbatim copy of the underlying source code, even Samuelson would likely conclude that the manufacturer is the “owner” of the output. This argument is even stronger given that the elements of operational output not contributed by the software developer are typically uncopyrightable factual data. For the same reason, courts are unlikely to find joint ownership. Most courts use Professor Goldstein’s copyrightability test, requiring that an alleged joint author contribute something that is itself

---


101 See Samuelson, supra note 23, at 1192 (concluding that “in general, the user of a computer generator program should be considered the author of a computer-generated work”); see also Arthur R. Miller, Copyright Protection for Computer Programs, Databases, and Computer-Generated Works: Is Anything New Since CONTU?, 106 HARV. L. REV. 977, 1071 (1993) (suggesting that even for artificial-intelligence programs, perhaps the person activating the program should be the author). Copyright does not expressly require the author to be human, so it is even possible to say the computer created the output. See Urantia Found. v. Maaherra, 114 F.3d 955, 958 (9th Cir. 1997) (examining the unusual question of copyright in a work alleged to be created by a divine alien being).

102 NAT’L COMM’N ON NEW TECHNOLOGICAL USES OF COPYRIGHTED WORKS, FINAL REPORT 45 (1979), available at http://digital-law-online.info/CONTU/PDF/index.html [hereinafter CONTU FINAL REPORT] (asserting that ownership in computer outputs should vest with the computer user, with certain qualifications).

103 See Samuelson, supra note 23, at 1200 n.71 (rejecting the idea that a novel created by a writer using a word processor would somehow be owned by the word processor’s maker).

104 Id. at 1199-1200 (arguing that even if computers could be authors, incentive policies would militate for human authorship).

105 Id. at 1192. Professor Miller has similar sensibilities. Miller, supra note 101, at 1056-57 (questioning whether the end-user of an AI program should be considered the author of its output or if the software writer might be a more appropriate author).

106 There are, however, counterexamples, such as the comment regarding a phone call with a customer provided in Part I of this Note. See discussion supra Part I.
copyrightable.107  

The amount of work done by the software relative to that done by the user may also be important. The Torah Soft court found that the user’s role in creating the output, typically restricted to entering short search phrases, was too small compared to the work performed by the software.108 The amount of work required to create a large-scale software system is likely to be much larger than that done by even a relatively large group of end-users, another reason for finding sole ownership for the software developers.109 Unfortunately for ISOs and those who would use their services, the nature of operational outputs is such that any copyright will be owned by the program developer, not its user.

III. DOES AFTERMARKET USE OF OPERATIONAL OUTPUTS VIOLATE COPYRIGHT?

Assuming the copyrightability of operational outputs, the next question to address is whether copying these outputs amounts to copyright infringement. The basic two-prong test for infringement is the same for any type of work. “To establish infringement, two elements must be proven: (1) ownership of a valid copyright, and (2) copying of constituent elements of the work that are original.”110 The amount and type of copying courts require to find infringement varies by the type of work being copied.111 The analytical framework used to evaluate the second prong for infringement of software depends on the type of copying alleged.112 “‘Literal’ copying is the verbatim copying of original expression, while ‘non-literal’ copying is that which is paraphrased, or loosely paraphrased.”113 “Non-literal” elements of a program

107 Erickson v. Trinity Theatre, Inc., 13 F.3d 1061, 1069-71 (7th Cir. 1994) (contending that the majority of courts use the Goldstein test requiring the copyrightability of each collaborator’s contribution); see also Paul Goldstein, Derivative Rights and Derivative Works in Copyright, 30 J. COPYRIGHT SOC’Y 209, 217 (1983) (analyzing distinctions between the reproduction of copyrighted works and creation of derivative works).


109 The “sweat of the brow” rationale for copyright protection has fallen from favor, but here the respective amount of work by the user and software developer determines ownership, not protection. Eldred v. Ashcroft, 537 U.S. 186, 236 (2003) (observing that the “sweat of the brow” rationale was rejected in 1834).


111 See supra note 45 and accompanying text.

112 See, e.g., ILOG, Inc. v. Bell Logic, LLC, 181 F. Supp. 2d 3, 6 (D. Mass. 2002) (describing logic behind choice of analytical framework for testing copyright infringement of software); 4 NIMMER ON COPYRIGHT, supra note 44, at § 13.03(A) (observing the difficulty in identifying “substantial similarity” between two works).

include its “fundamental essence or structure.” Copying operational outputs could qualify as either. ISOs routinely work with electronic copies of operational outputs, so most copies will be verbatim. Because the output consists of fragments of the underlying code, verbatim copying could be considered literal copying of the source code. However, outputs also represent the “essence or structure” of the source code, so reproducing them may also be considered non-literal copying.

A. Infringement of Operational Outputs as Literal Copies of the Source Code

Developers can typically obtain copyrights for the source code of computer programs. Because operational outputs consist of many copies of small fragments of the underlying copyrighted source code, copying the output may essentially infringe the copyright of the source code itself. “Fragmented” or otherwise partial literal copying is no bar to finding infringement. Courts have found infringement where as little as thirty percent of a program’s source code was copied. Of the many lines of code in large software systems, however, only a tiny percentage may end up as part of the operational output. Even well-written programs that are mission-critical and thus highly dependent on good operational outputs may have output statements that comprise less than five percent of their total code base. There may be thousands of instances of those statements in a typical output stream, but it seems unreasonable to determine whether there is infringement based on how many times a statement was copied.

Ultimately, infringement through verbatim copying is a case-by-case determination. Even small amounts of verbatim copying may be found to

---

115 Summarized or paraphrased outputs are impractical for several reasons. The time required to summarize the output might remove the economic value of having a third party deal with operational issues in the first place. Also, outputs are typically terse to begin with, and any condensation would run the risk of removing some information critical to solving operational issues.
116 Altai, 982 F.2d at 702 (stating that protection for source code is “well settled”).
117 See Best Cellars, Inc. v. Grape Finds at Dupont, Inc., 90 F. Supp. 2d 431, 460 (S.D.N.Y. 2000) (holding that extensive and in many cases word-for-word copying of a wine brochure with interspersed non-copied material is sufficient for injunction). The mix of literal copies of fragments of copyrighted code interspersed with user-input data in operation outputs suggests that such outputs may be eligible for copyright protection under the reasoning of the Best Cellars case.
118 See Altai, 982 F.2d at 700.
119 This is the case with the Wireless National Number Portability applications from which the outputs in the Appendix hale.
120 Roy Export Co. v. Columbia Broad. Sys., Inc., 503 F. Supp. 1137, 1145 (S.D.N.Y. 1980) (commenting that substantiality of use is a classic jury question), aff’d, 672 F.2d 1095 (2d Cir. 1982); 4 NIMMER ON COPYRIGHT, supra note 44, at § 13.03(A)(2) (“[D]etermination
infringe, though there is no bright-line quantitative test.\textsuperscript{121} It seems intuitive, however, that the large-scale copying that ISOs engage in is unlikely to be disregarded as de minimis.

B. Infringement of Operational Outputs as Copies of Non-Literal Elements of the Underlying Source Code

Even if a court determines that a program’s operational output is too different from its source code to constitute fragmented literal copying, there is still a strong argument to be made that reproduction of the output constitutes non-literal copying. The Altai test has been widely adopted by courts and praised by commentators for use in determining whether a copy of non-literal elements of a program is infringement.\textsuperscript{122} Because operational output can represent non-literal elements of a program such as its architecture, Altai is an appropriate test for determining whether copying of outputs is tantamount to non-literal copying of source code.\textsuperscript{123}

Altai may also be useful for determining whether the reproduction of outputs, framed as fragmented literal copying, has copied protected elements.\textsuperscript{124} Even if the output is substantially similar to elements of the source code, copying them will not result in infringement if the elements are unprotected \textit{scène à faire}, taken from the public domain or dictated by external factors.\textsuperscript{125} Altai is also relevant for evaluating the application of the merger-of substantial similarity with respect to fragmented literal similarity . . . in the last analysis requires a value judgment.”).

\textsuperscript{121} Educ. Testing Servs. v. Katzman, 793 F.2d 533, 542 (3d Cir. 1986) (rejecting the argument that copying a “handful of questions out of thousands” was de minimis); Chi. Record-Herald Co. v. Tribune Ass’n, 275 F. 797, 799 (7th Cir. 1921) (determining that infringement cannot be measured by “lines or inches”); 4 NIMMER ON COPYRIGHT, supra note 44, at § 13.03(A)(2) (contending that there is “no easy rule of thumb” for the quantum of copying required to constitute infringement). Courts examining ISOs’ use of outputs must also determine how to measure the amount copied. It could be measured with respect to the entire program, relative to the amount of output the program generates in a given period of time, or in many other ways. Though there is no clear doctrinal answer, courts will often look at the ratio of the copy to the component part, not the entire work that has been copied. Markham v. A.E. Borden Co., 206 F.2d 199, 201-02 (1st Cir. 1953) (asserting that the “material and substantial” test is to be applied not to plaintiff’s entire catalog but to each component part which has been infringed).


\textsuperscript{123} See discussion supra Part I (discussing characteristics of operational output).

\textsuperscript{124} After all, Altai is just an implementation of the merger-doctrine. Computer Assocs. Int’l, Inc. v. Altai, Inc., 982 F.2d 693, 706 (2d Cir. 1992) (“This approach . . . draws on such familiar copyright doctrines as merger . . . .”).

\textsuperscript{125} \textit{Id.} at 707. The Second Circuit affirmed the lower court’s holding of noninfringement in spite of the fact that much of the program code was similar. \textit{Id.} at 714-715; see also Folio Impressions, Inc. v. Byer Cal., 937 F.2d 759, 765 (2d Cir. 1991) (requiring that any
doctrine to literal copies of source code in operational output.126

The *Altai* test recognizes that not all elements of a software program are deserving of copyright.127 The *Altai* test may preclude portions of operational output text from being examined for infringement in two ways. At the top of the abstraction spectrum, the “ideas” behind the code that produces output may be unprotectable.128 At the bottom, the details of individual subroutines (each of which accomplishes only one small task) may be uncopyrightable scènes à faire because there is only a limited number of ways to accomplish these low-level functions.129

Operational output is usually the result of “instrumenting”130 subroutine calls that exist at all levels of abstraction, from the fundamental task of the system to its smallest implementation details. Each subroutine is typically named according to its function and included in the operational output. Subroutines from “DetermineBridgeStructuralSafety” down to “GetIndividualBoltLoadLimit” represent descriptions of the program’s entire spectrum of abstraction.131 Some of the subroutines described by the output are likely to be uncopyrightable “ultimate functions of the program” like “DetermineBridgeStructuralSafety.”132 If the output is framed as a non-literal

---

126 See discussion infra Part III.E.4.
127 *Altai*, 982 F.2d at 706 (recognizing the need to filter out uncopyrightable elements of a program before infringement comparison).
128 17 U.S.C. § 102(b) (2000) (“In no case does copyright protection for an original work of authorship extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work.”).
129 *Morrissey v. Procter & Gamble Co.*, 379 F.2d 675, 678 (1st Cir. 1967) (stating that if rules describing a contest can only be expressed in a limited number of ways, the idea of the contest merges with its expression through the rules, which are thus uncopyrightable).
130 “Instrumenting” is the process of inserting output-generating code into elements of a program so as to provide an instrument-like gauge of what the program is doing when these code elements are run by the computer’s CPU. Most, if not all, subroutines are instrumented in large programs.
131 See text *supra* Part I (explaining how operational outputs express the abstraction hierarchy of a program’s architecture).
132 Most programs consist of high-level functions that represent the ultimate purposes of the software. These functions in turn call functions of a progressively more specific and simpler nature. *Altai*, 982 F.2d at 697. Typically, designers will program the software to generate operational output by “instrumenting” the program’s entry and exit points leading to and from each function. Instrumenting means writing the software so that it prints a message indicating the entry to or exit from the function along with the input and output data parameters and any error results. Since at the highest level of abstraction there is nothing but the ideas of functions that the program will ultimately perform, there cannot be
conceptualization of the copyrighted source code, an Altai test might filter out some or all of the output because the output describes the “idea” of what the program is doing. Also, each line of the output describes a subroutine and its parameters or “interface.” Standard programming techniques dictate that these interfaces must be as efficient as possible, and their design is driven in large part by external factors such as the attributes of the real world information that the program manipulates. These are the same types of criteria that Altai is most likely to filter out as undeserving of copyright. Even the names of subroutines are typically functional, e.g., “UpdateCustomer,” and thus likely to be filtered from comparison as simply ideas/expressions dictated by external factors. Many low-level functions that, for example, simply find a record in a database may well be unprotected because they are basic building blocks that are unprotectable due to their simplicity. The simplicity of these functions makes them more likely to be taken from the public domain or labeled by courts as software scènes à faire. Thus, statements in operational outputs describing subroutine calls at high levels of abstraction may be filtered out of comparisons due to the idea-expression dichotomy, and those at lower levels may be filtered out because of the merger doctrine. Yet support for finding infringement using an output-is-source-code argument still exists. Even if the lines of output representing higher-level elements of the program are filtered out as ideas and lower-level elements filtered out as scènes à faire, there is likely some area between these two extremes which will survive the filtering process. Courts comparing any copyright protection at that level, such as for “the efficient management of a dental laboratory.” Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1236 n.28 (3d Cir. 1986).

For example, the parameters that make up the interface of a function called UpdateBankBalance might consist of two numbers – AccountNumber and NewBalance.

Altai, 982 F.2d at 710-11 (examining which elements should be filtered out of the final comparison step of the software copyright infringement test).

See Morrissey v. Procter & Gamble Co., 379 F.2d 675, 678 (1st Cir. 1967) (holding that if an idea can only be expressed in a limited number of ways, then the idea merges with its expression and is thus uncopyrightable). Programmers have a tendency to find programming “patterns” (strategies) that work reliably and prefer to use those patterns whenever possible. These patterns are published and studied. See, e.g., ANDREI ALEXANDRESCU, MODERN C++ DESIGN: GENERIC PROGRAMMING AND DESIGN PATTERNS APPLIED (2001) (providing an in-depth study of patterns in the C++ computer programming language).

This is the converse of Judge Learned Hand’s “Abstraction Test.” See Nichols v. Universal Pictures Corp.:

Upon any work, and especially upon a play, a great number of patterns of increasing generality will fit equally well, as more and more of the incident is left out. The last may perhaps be no more than the most general statement of what the play is about, and at times might consist only of its title; but there is a point in this series of abstractions
verbatim copies in this middle area and using a substantial similarity standard are likely to find infringement.137

“[I]ndividual program elements that are ‘filtered’ out at one level may be copyrightable when viewed as part of an aggregate of elements at another level of abstraction.”138 Operational output may be just such an aggregate; even though many of its individual elements may be unprotectable, when viewed holistically, the output essentially is the program itself, both in its text and in all the creative decisions that go into how it operates. Where the “fundamental essence or structure of one work is duplicated in another,” courts have found copyright infringement under Nimmer’s “comprehensive nonliteral similarity” test.139 As a detailed representation of a work that is undoubtedly protected,140 it seems difficult to say that from a gestalt perspective outputs do not also deserve copyright even if outputs are unprotected by Altai under a line-by-line dissection. Finally, the nature of the alleged copying may affect the outcome of an Altai test for infringement. The literal copying that ISOs are likely to do tips the scale toward courts finding that a work’s expression, not function, has been copied.141

Although the results of evaluating operational outputs under the Altai test are inconclusive, it seems likely that many non-literal aspects of output will survive the filtering process. Alternatively, if considered as fragmented literal copies of the source code, reproductions of operational outputs have an even greater chance of infringement liability.

C. Infringement of Operational Outputs as Derivative Works

Section 106 of the Copyright Act grants the exclusive right to create or
authorize the creation of derivative works to the underlying work's copyright holder. A software manufacturer suing to prohibit a paying customer from using the software because it creates a derivative work is obviously untenable as a business practice. But manufacturers might sue an ISO for creating a derivative work. If operational outputs are considered derivative works, such suits could be successful.

As noted above, some suits may be avoided by having the customer create the operational outputs. Even then, ISOs will likely make internal copies. Thus the critical legal question is whether unauthorized copying, rather than creation, of derivative works violates the Copyright Act. Section 106 grants exclusive rights to copyrighted works without differentiating between "derivative," "compilation," or "expressive" works, and would thus seem to deny unauthorized copying of derivative works on its face.

This logical conclusion is supported by caselaw. Micro Star v. Formgen Inc. involved the software manufacturer of the video game "Duke Nukem" (Formgen) who gave its customers implicit license to create derivative works in the form of audiovisual outputs in new "levels" of player-created game scenarios. Customers could create new levels by adding creative content (parameters and rules) to the copyright holder's existing software. Even though the creation of the resulting derivative works might have been authorized, the court granted a preliminary injunction because the defendant,

---

142 17 U.S.C. § 106(2) (2000); Mirage Editions, Inc. v. Albuquerque A.R.T. Co., 856 F.2d 1341, 1343 (9th Cir. 1988) (finding mounting of art to tiles was a derivative work that violated the rights of the copyright holder of the art).

143 See CONTU FINAL REPORT, supra note 102, at 13 ("Obviously, creators, lessors, licensors, and vendors of copies of programs intend that they be used by their customers, so that rightful users would but rarely need a legal shield against potential copyright problems."). Doing so would essentially be suing the purchaser for using her purchase. Describing the manufacturer's dilemma in one seminal case, a litigation partner said "[f]or obvious reasons, MAI did not want to sue its own customers for breach of the license agreements. On the other hand, something had to be done because Peak and other ISOs were devouring MAI's service business." Kip Schwartz, Stopping the Feeding Frenzy, INTELL. PROP. MAG., July 1997.

144 Presumably, the end-user is authorized to run the program and thus create derivative works. See discussion supra Part I.B.

145 See discussion supra Part I.A. It is possible that the ISO might also cause the output to be made as they did in the Storage Tech case. Storage Tech. Corp. v. Custom Hardware Eng’g & Consulting, Inc., No. 02-12102RWZ, 2004 WL 1497688, at *3 (D. Mass. July 2, 2004), vacated, 421 F.3d 1307 (Fed. Cir. 2005).


147 154 F.3d 1107, 1113 (9th Cir. 1998) ("The only written license FormGen conceivably granted was to players who designed their own new levels . . . .").

148 Micro Star, 154 F.3d at 1110-11 (describing how custom games are created through “.MAP” files).
Micro Star, copied and distributed the works.149

As previously explored in Part II.C, operational outputs may be analogous to the audiovisual output of a video game. Like Duke Nukem’s display, operational outputs are generated when the hardware combines instructions from the underlying program with user interaction.150 Thus, Micro Star indicates that ISOs that copy operational outputs without authorization, even if they are authorized to create the derivative outputs, infringe the software manufacturer’s copyright.

Depending on who owns operational outputs, ISOs would need either the permission of the underlying copyright owner (the manufacturer) or the permission of the copyright owner in the derivative work (either the manufacturer or the end-user). Only where the end-user is deemed the copyright owner of the derivative work do ISOs have some freedom to operate. But as the following discussion demonstrates, even that possibility is unavailing for ISOs because of how ownership is allocated in derivative works.

The Micro Star court focused specifically on who owned the copyright in the underlying work and it held the defendant liable after finding that the copyright belonged to the plaintiff.151 The court’s focus on copyright ownership made sense because that issue was dispositive – without the manufacturer’s permission, the end-user cannot receive copyright ownership even in the information added by end-users.152 Even if the end-user gets permission to create derivative outputs, copyright in all elements other than those added by the end-user remains with the manufacturer.153 Thus, a

149 Id. at 1113-14 (discussing how Micro Star impeded Formgen’s ability to market new forms of the video game).

150 Stern Elecs., Inc. v. Kaufman, 669 F.2d 852, 853 (2d Cir. 1982) (stating that video games are basically “computers programmed to create on a television screen cartoons in which some of the action is controlled by the player”).

151 Micro Star, 154 F.3d at 1109-10 (explaining that the first step in making its decision was determining whether the software-manufacturer plaintiff owned a copyright in the game). The court never discussed the possibility of end-user ownership of the .MAP files that contained the new levels. Id.

152 See 17 U.S.C. § 103(a) (2000) (“[P]rotection for a work employing preexisting material in which copyright subsists does not extend to any part of the work in which such material has been used unlawfully.”); Dam Things from Denmark v. Russ Berrie & Co., 290 F.3d 548, 563 (3d Cir. 2002) (reasoning that if the underlying work is protected by copyright, then the creator of the derivative work receives no protection at all); Pickett v. Prince, 207 F.3d 402, 406-07 (7th Cir. 2000) (stating that one cannot gain rights in a derivative work without the permission of the copyright owner of the underlying work); Gracen v. Bradford Exch., 698 F.2d 300, 302 (7th Cir. 1983) (holding that Gracen still needed authority to use copyrighted materials even if her work had enough originality to be copyrightable as a derivative work).

153 See Stewart v. Abend, 495 U.S. 207, 223 (1990) (“[T]he element drawn from the pre-existing work remains on grant from the owner of the pre-existing work.”); Mulcahy v. Cheetah Learning LLC, 386 F.3d 849, 852 (8th Cir. 2004) (limiting copyright in derivative
verbatim copy of an operational output would necessarily copy at least some elements that were copyrighted by the software manufacturer.\textsuperscript{154} Regardless of who owns the derivative work, ISOs’ verbatim copying of that work will infringe the manufacturer’s copyrights.\textsuperscript{155}

\textbf{D. Infringement of Operational Outputs as Compilations}

If outputs are compilations deserving of copyright, the next question to consider is what “thin” protection exists for them.\textsuperscript{156} Even if a factual compilation “features an original selection or arrangement of facts,” any copyright is limited to the particular selection or arrangement of those facts, not the facts themselves.\textsuperscript{157} Courts apply the “substantial similarity” test to determine infringement of the protectable elements of a compilation.\textsuperscript{158} At the very least, compilations are protected against verbatim or nearly verbatim copying\textsuperscript{159} because verbatim copies of compilations will necessarily duplicate the protectable selection and arrangement of these works. Therefore, such copies would be “substantially similar,” and infringe on any copyright the manufacturer holds in the output.

As stated in Part II.D, it is likely that the end-user owns these copyrights. The manufacturer has, at least implicitly, authorized the inclusion of fragments of its source code into the operational output.\textsuperscript{160} As the compilation’s owner,

\begin{itemize}
  \item verbatim copies of compilations will necessarily duplicate the protectable selection and arrangement of these works.
  \item At the very least, compilations are protected against verbatim or nearly verbatim copying because verbatim copies of compilations will necessarily duplicate the protectable selection and arrangement of these works.
  \item Courts apply the “substantial similarity” test to determine infringement of the protectable elements of a compilation.
  \item As stated in Part II.D, it is likely that the end-user owns these copyrights.
  \item The manufacturer has, at least implicitly, authorized the inclusion of fragments of its source code into the operational output.
\end{itemize}

\textsuperscript{154} This is because everything except for the user-added parameters comes from the software itself. See discussion supra Part I.

\textsuperscript{155} 4 NIMMER ON COPYRIGHT, supra note 44, at § 13.03(a)(2) (determining that where there is “substantial similarity” between two works, i.e., when there is word for word copying, it “is not necessary to determine the level of abstraction at which similarity ceases to consist of an ‘expression of ideas,’ because literal similarity by definition is always a similarity as to the expression of ideas”) (citations omitted).

\textsuperscript{156} See supra text accompanying note 45.


\textsuperscript{158} Key Publ’ns, Inc. v. Chinatown Today Publ’g Enters., 945 F.2d 509, 514 (2d Cir. 1991) (asserting that the “test for infringement of compilations . . . is one of ‘substantial similarity’”).

\textsuperscript{159} Id. at 514 (stating that although copyright for factual compilations is thin, it is not anorexic (citing Feist, 499 U.S. at 349)); Hoehling v. Universal City Studios, Inc., 618 F.2d 972, 980 (2d Cir. 1980) (describing that in historical works, a second author may “make significant use of prior work, so long as he does not bodily appropriate the expression of another”); 1 NIMMER ON COPYRIGHT, supra note 44, at § 2.11(B) (commenting that “word for word or very closely paraphrased copying” of factual accounts is still prohibited); C. Ginsburg, No “Sweat”? Copyright and Other Protection of Works of Information After Feist v. Rural Telephone, 92 COLUM. L. REV. 338, 349 (1992) (interpreting Feist as suggesting that “extensive verbatim copying” would be disallowed).

\textsuperscript{160} See supra note 23 and accompanying text.
the user would have the same rights to copy and distribute his “work” as any other copyright holder (notwithstanding the presumably lawful inclusion of source-code elements). If so, end-users would be able to authorize ISOs to make whatever copies they need. If courts view operational outputs as compilations, then infringement will ultimately turn on which party the court finds to be the owner. If the court finds the end-user to be the owner, the issue will become whether the software developer implicitly “authorized” the end-user to include whatever material the manufacturer owns in the operational output.

E. Copyright Exceptions and Defenses

Even if copying operational outputs is a per se infringement of the underlying copyrighted work, there are a number of statutory provisions which may exempt ISOs from prosecution.

1. 17 U.S.C. § 117(a)(1)

Section 117 provides two exceptions pertinent to operational outputs. Section 117(a)(1) states that it is not copyright infringement for “the owner of a copy of a computer program to make or authorize the making of another copy or adaptation of that computer program [if] . . . such a new copy or adaptation is created as an essential step in the utilization of the computer program in conjunction with a machine.”ISOs who argue that § 117(a)(1) allows copying of outputs face several hurdles. “Owner” for purposes of § 117 has received differing judicial interpretations and may be factually dependent on the scope of use in the software’s license. The Ninth Circuit has indicated that the most typical software users – licensees – are not “owners.” The Federal Circuit and other

161 17 U.S.C. § 103(b) (2000) (“Copyright in [a compilation] is independent of, and does not affect or enlarge the scope, duration, ownership, or subsistence of, any copyright protection in the preexisting material.”); 17 U.S.C. § 201(c) (2000) (“Copyright in each separate contribution to a collective work is distinct from copyright in the collective work as a whole, and vests initially in the author of the contribution.”); see also N.Y. Times Co. v. Tasini, 533 U.S. 483, 496-97 (2001) (determining that the owner of a compilation is “privileged to reproduce or distribute an article contributed by a freelance author, absent a contract otherwise providing[,] . . . ‘as part of’ . . . ‘that collective work’”). So long, at least, as the original source’s copyright owner gave permission for the compiler to use the original source in the compilation. 1 NIMMER ON COPYRIGHT, supra note 44, at § 3.06 (explaining that “consent of the copyright owner of a still-protected pre-existing work is necessary to render the derivative or collective work non-infringing”).


163 See MAI Sys. Corp. v. Peak Computer, Inc., 991 F.2d 511, 518 n.5 (9th Cir. 1993) (holding that licensees are not owners); see also CMAX/Cleveland, Inc. v. UCR, Inc., 804 F. Supp. 337, 356 (M.D. Ga. 1992) (finding licensee of copyrighted computer software
courts have criticized the Ninth Circuit’s position, and typically find that software licensees are owners.\textsuperscript{164}

Whether copying outputs is “essential” to the utilization of the program may also depend greatly on the facts of the case. Courts tend to read § 117 broadly, and it is difficult to envision companies spending money on ISOs if their services were not truly essential.\textsuperscript{165} Ensuring that important programs continue to operate correctly seems like a modest request compared to other scenarios where courts have sanctioned copying as “essential,” such as saving the time it takes to swap a CD-ROM.\textsuperscript{166}

The highest hurdle ISOs face when seeking an exception under § 117(a)(1) is that the statute contemplates the exception applying to copies of “programs system insufficient to be an “owner” of the code for purposes of § 117). As most software users are licensees, it is difficult to imagine why § 117(a)(1) would have even been enacted if these users were exempt from the privileges of the statute.

\textsuperscript{164} See Krause v. Titleserv, Inc., No. 03-9303, 2005 U.S. App. LEXIS 4570, at *16 (2d Cir. Mar. 21, 2005) (reasoning that an “owner” for § 117 purposes does not require a formal title, and that instead the right “to possess and use a copy indefinitely without material restriction, as well as to discard or destroy it at will, gave [the rightholder] sufficient incidents of ownership to make it the owner of the copy for purposes of applying § 117(a)’); DSC Commc’ns Corp. v. Pulse Commc’ns, Inc., 170 F.3d 1354, 1360 (Fed. Cir. 1999) (disagreeing with MAI, though still finding that the licensing limitations in this particular case made users non-“owners” in spite of the fact that they had a license); Logicom Inclusive, Inc. v. W.P. Stewart & Co., No. 04 Civ. 0604, 2004 U.S. Dist. LEXIS 15668, at *38 (S.D.N.Y. Aug. 9, 2004) (stating in dicta that determining the owner is easy if a consumer has purchased a copy of the program); Telecomm Tech. Servs. v. Siemens Rolm Commc’ns, Inc., No. 1:95-CV-649-WBH, 1999 U.S. Dist. LEXIS 21414, at *12 (D. Ga. July 6, 1999) (finding the owners of phone systems to be “owners” of the embedded software for § 117(c) purposes in an ISO case). The court seemed to find “lessees” to mean licensees. Telecomm, U.S. Dist. LEXIS 21414, at **9-10; see also 2 NIMMER ON COPYRIGHT, supra note 44, at § 8.08(B)(1)(b)-(c) (using terms like “confusion” and “subversion” to describe the MAI decision regarding § 117 ownership).

\textsuperscript{165} See Krause, 2003 U.S. Dist. LEXIS 24456, at **22-23 (E.D.N.Y. 2003) (arguing for a broad reading of § 117 when an owner uses a program “in the pursuit of its own business”), aff’d, No. 03-9303, 2005 U.S. App. LEXIS 4570 (Mar. 21, 2005); DSC Commc’ns Corp. v. Pulse Commc’ns, Inc., 976 F. Supp. 359, 362 (E.D. Va. 1997) (explaining that the “trend is to read Section 117 broadly,” and interpreting that Section to apply to “a legitimate holder of a computer program”), rev’d in part, 170 F.3d 1354 (Fed. Cir. 1999); RAV Commc’ns, Inc. v. Philipp Bros., Inc., No. 87 Civ.3366 (LLS), 1988 U.S. Dist. LEXIS 3048, at *6 (S.D.N.Y. 1988) (reasoning that § 117’s legislative history indicates that it “should be given a broader reading where the owner of a copy of a computer program adapts it for his own internal use”). The end-user of a program creating operation output would almost certainly be using it internally in the pursuit of his own business.

or adaptations” thereof, not to outputs of programs. No court has interpreted § 117 “to encompass anything other than computer programs.”167 On its face, the “right to adapt the program . . . extends no further than what is reasonably necessary to be able to use the program on a certain CPU.”168 Given the “broad reading” of § 117,169 however, one could argue that operational outputs are “adaptations” of the underlying program.

If outputs are derivative works,170 and if derivative works are considered per se adaptations, then § 117 may allow for their copying.171 Although no case has addressed whether derivative works are § 117 adaptations, some guideposts do exist. 17 U.S.C. § 101 defines derivative works as those “based upon one or more preexisting works,” including “any . . . form in which a work may be recast, transformed, or adapted.”172 Therefore, at least some derivative works are adaptations. The CONTU Reports seemed to equate the two, stating that notes in a book’s margin might constitute a derivative work, and then later referring to these annotated books as adaptations.173

If copying operational outputs is not covered by the word of the exemption, it at least appears to be covered by its spirit. The rationale behind § 117(a)(1) is that “[o]ne who rightfully possesses a copy of a program . . . should be provided with a legal right to copy it to that extent which will permit its use by that possessor.”174 A specific purpose of the section was to ensure that hardware ISOs would not be harmed by overzealous copyright protection.175

169 See supra note 165.
170 See discussion Part II.A (discussing whether outputs are derivative works from the source code).
171 Admittedly, there must be some limits to this argument; a licensee of Microsoft Word surely could not create an improved version of the original program and argue that doing so was “essential” for her to make use of the program. At some point the original copyright owner could enjoin the user from making what would in effect be a derivative work. Kemp, supra note 168, at 113 (reasoning that “if the adaptation is a derivative work, copyright law may allow the software license’s prohibition against modification of the program”). Outputs as derivatives would not suffer this problem if they were deemed to be owned by the end-user or licensee that produced the output.
173 CONTU FINAL REPORT, supra note 102, at 13-14 (describing why adaptations were added to § 117).
174 Id. at 13 (explaining why it recommended § 117(a)(1)).
Regardless, courts read “programs or adaptations” narrowly, likely making § 117(a)(2) unavailing for ISOs.

2. 17 U.S.C. § 117(c)

The second exception that might apply to ISOs is § 117(c), which was added by the Computer Maintenance Competition Assurance Act when the DMCA was enacted. Section 117(c) provides that, with certain limitations, it is not copyright infringement for the owner of a machine to authorize copying of a program that the machine contains “solely by virtue of the activation of a machine that lawfully contains an authorized copy of the computer program, for purposes only of maintenance or repair of that machine.”

As with § 117(a)(1), the first difficulty in using § 117(c) is that ISOs do not copy the program per se. Even if that objection can be avoided by considering operational outputs as a fragmented literal copy, there is a greater problem in using § 117(c). The purpose of any copying in the software aftermarket is for maintenance or repair of the software system, not the machine on which it is running. Legislative history indicates Congress intended § 117(c) to permit only those activities strictly relating to hardware. Maintenance and repair do not include “modifications, error corrections or any other changes to any software which may be in the machine being serviced.”

Protecting the ISO industry was clearly a goal of § 117(c), but

question the right of hardware ISOs to provide service under the DMCA).


178 See PracticeWorks, Inc. v. Prof’l Software Solutions of Ill., Inc., No. Civ. JFM-02-1205, 2004 WL 1429955, at *6 (D. Md. June 23, 2004) (explaining that the copying of software is “only permissible under 117(c) if it is for the purposes of repairing the machine into which it is loaded, not the software itself”).

179 2 NIMMER ON COPYRIGHT, supra note 44, at § 8.08(D)(1) (describing the legislative history behind the addition of § 117(c)). Nimmer argues strenuously, however, that licensees should not be liable for copying that is required for maintenance of software, or at least that § 117 does not imply that such copying should be considered ipso facto wrongful. No cases have directly addressed this question. Id. Nimmer also argues that licensees should be able to authorize ISOs to do this copying.

180 S. REP. NO. 105-190, at 59 (1998) (evidencing Congress’ intent in enacting § 117(c)). But see supra note 179 and accompanying text.

181 144 CONG. REC. H10615, 10618 (May 22, 1998) (“ISOs are prevented from reading the diagnostics software and, subsequently, cannot service the computer’s hardware [which means . . . that the multibillion dollar nationwide ISO industry is at risk.”) (statement of Rep. Knollenberg). Congress considered the § 117 amendment to be a clarification of current law. S. REP. NO. 105-190, at 2 (1998). Therefore this amendment was not about placing limits on what protections Congress thought were already supported by the law, but about expanding those protections, which may be an argument for a broader reading of § 117(c).
considering its limitations, it appears to protect only ISOs who service hardware – there is no indication that § 117(c) applies to independent software service providers. One could argue whether software ISOs existed in large enough numbers to warrant explicit legislation when §117 was passed, but given the house of cards on which a § 117 defense might be built, software ISOs should look to other areas of the law for help.

3. Fair Use

Rights to control the reproduction of a work are not absolute. The fair use doctrine requires “courts to avoid rigid application of the copyright statute when it would stifle the very creativity which that law is designed to foster.” The four factors relevant in determining whether a use is fair are: (1) the purpose and character of the use; (2) the nature of the copyrighted work; (3) the substantiality of the portion used in relation to the copyrighted work as a whole; (4) the effect on the potential market for or value of the copyrighted work.

The leading case on fair use of software for computer servicing is Triad Systems Corp. v. Southeastern Express Co. The Triad court decided that loading a copy of a manufacturer’s operating system diagnostic software into RAM to allow an ISO to service the computer’s hardware was not a fair use under § 107.

Given the often blurred line between software and hardware, there is little reason to think software maintenance should be treated differently from hardware maintenance. While this Note concerns fair use of software outputs, not of the software itself, the two § 107 factors the Triad court examined – the effect and purpose of the use – produce the same analytical results when applied to software outputs. Triad analyzed the effect and purpose of ISOs that use software for maintenance, and operational outputs

182 See Storage Tech. Corp. v. Custom Hardware Eng’g & Consulting, 421 F.3d 1307, 1313-14 (Fed. Cir. 2005) (finding that hardware ISO was protected by § 117(c), but focusing on machine repair when commenting that “[a]ccessing software programs, such as freestanding diagnosis and utility programs, that are not needed to boot up the computer and make that determination[] goes too far because access to those programs is not strictly necessary to verify that the computer is ‘working in accordance with its original specifications’” (citing § 117(d))). Section 117(c) had not previously been construed by any court of appeals, so this decision may not be the last word on the subject. Id. at 1311.


185 64 F.3d 1330 (9th Cir. 1995), cert. denied, 516 U.S. 1145 (1996).

186 Id. at 1337 (finding no fair use).

187 See Smith & Mann, supra note 10, at 259.

188 Triad, 64 F.3d at 1336-37 (differentiating Southeastern’s type of use and purpose from an earlier case where fair use had been found); see also 17 U.S.C. § 107(1), (4) (2000) (stating the purpose and nature of both the use and the effect of that use on the marketplace).
presumably have the same purpose and similar effects on the manufacturer’s
service market. Moreover, these were the two most dispositive factors both in
Triad’s arguments and in those of commentators on copyright fair use.189
Thus, both Triad and related commentary provide a cogent framework for
analyzing fair use for software ISOs.

Professor Stephen McJohn admits that mechanically applying a typical fair
use analysis to copying diagnostic software “could militate against fair use”
and that computer maintenance is not one of the categories specifically favored
by § 107.190 Regardless, McJohn argues that Triad was wrongly decided
because the court erred in finding a cognizable harm to the copyright owner
and failed to see the benefits created by Southeastern’s activities.191 These
reasons are applicable when arguing that copying of operational outputs by
ISOs is fair use.

McJohn reasons that the harm done to Triad was not in a market that
copyright should protect.192 Copyright protects the initial software sales
market, not the servicing aftermarket.193 The Sixth Circuit DMCA case of
Lexmark Int'l, Inc. v. Static Control Components, Inc. recently emphasized this
distinction in dicta.194

Douglas Rogers has suggested considering a fifth element when examining
fair use—whether the “original work is being used in an attempt to control the
market for another product.”195 Because the Triad court regarded the
diagnostic software as separate from the rest of the system, they were able to
view the software as a product specifically targeted at the service market.
Consequently, Southeastern could not effectively make this argument.196

But ISOs will often be able to show that the developer’s software is

189 See McJohn, supra note 6, at 608 (stating that the Triad court focused on “the harm to
the market for the copyrighted work and the nature of the use”).
190 See id. at 607, 623.
191 Id. at 608-609 (summarizing arguments for a different analysis of fair use than the
Triad court used).
192 Id. at 610-11 ( theorizing that the harm to Triad was in the market for servicing its
computers).
193 Id. ( concluding that “harm to the service market should not be considered in the fair
use balance, because it goes beyond the scope of the copyright”).
194 387 F.3d 522, 544 (6th Cir. 2004) ( finding that although the defendant toner cartridge
aftermarket provider commercially used the code at issue, it was not “seeking to exploit or
unjustly benefit from any creative energy that Lexmark devoted to writing the program
code”).
195 Douglas L. Rogers, Give the Smaller Players a Chance: Shaping the Digital Economy
196 See Triad, 64 F.3d at 1333 ( observing that of the three groups of plaintiff software,
only “[t]he OS software and the service software are involved in this dispute”); id. at 1337
(reasoning that because “[Triad] invented, developed, and marketed its software to enable its
customers and its own technicians to service Triad computers[,] Southeastern is getting a
free ride when it uses that software to perform precisely the same service”).
unrelated to the service market. The source code that produces operational outputs is typically built into large-scale systems. Manufacturers cannot claim that they developed a distinct service and maintenance market because they cannot argue that they developed a separate software product to target that market. It is troubling that this issue might depend merely on whether the manufacturer bundled its diagnostic software with its “primary” software, and perhaps a better focus might be whether it is even possible for a third party to create a separate diagnostic application.

The Ninth Circuit court found that Southeastern harmed Triad’s market for servicing its computers as well as its market for licensing the diagnostic software. Southeastern’s activities were distinguishable from earlier cases finding fair use of software because the Triad court found that Southeastern’s servicing was not transformative in the same way as the uses examined in earlier cases. Southeastern could have written its own diagnostic software, and was thus getting a free ride by using Triad’s. Thus the ISO’s use of the diagnostic software was not a fair use.

Despite the court’s conclusion, Southeastern’s use of the software may not actually have been “pure free-riding” lacking public benefit. Like other ISOs, Southeastern contributed independent resources of talent and time and was not selling copies of software that could be substituted for those sold by the manufacturer. The Triad court found Southeastern’s use to be non-

197 See infra note 220 and accompanying text. In the author’s opinion, this program code must also necessarily be built into the rest of the software – it is not practical to instrument a system from outside of that system, and this also means that ISOs have no real chance of creating their own diagnostic software.

198 Triad, 64 F.3d at 1337 (describing market effects of ISO’s use of diagnostic software).

199 Id. (comparing other cases to Triad). Fair use has been found for transformative purposes like creating competing video game cartridges. See, e.g., Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510, 1527-28 (9th Cir. 1992) (finding that copying done during reverse-engineering and aimed at figuring out software compatibility requirements in order to create new games for the Sega system was a fair use). The Triad court believed that Southeastern had invented nothing of its own to militate for fair use. Triad, 64 F.3d at 1336 (repeating the lower court’s finding that Southeastern’s use was “neither creative nor transformative and does not provide the marketplace with new creative works”).

200 See Triad, 64 F.3d at 1337 (“Southeastern is simply commandeering its customers’ software and using it for the very purpose for which, and in precisely the manner in which, it was designed to be used.”).

201 Id. (“[W]e detect no appreciable public benefit arising from Southeastern’s practice to justify this continuance under the fair use doctrine.”).

202 See McJohn, supra note 6, at 622 (criticizing the court’s reasoning for denying Southeastern’s fair use claim).

203 Id. at 621 (explaining how an ISO is not merely reproducing copies but producing them, and must contribute independent resources to do so).
transformative and thus not the type of use typically protected by fair use.\textsuperscript{204} But because Southeastern produced no “second work,” this distinction is inapposite. There is no other work in which to look for a transformation.\textsuperscript{205} Alternatively, servicing does foster an important feature of transformative use – innovation.\textsuperscript{206} Through continued maintenance, products evolve to better handle customers’ business requirements.

McJohn points out that the Ninth Circuit has found fair use in Sega v. Accolade,\textsuperscript{207} where copying was “the only practical way to ascertain certain functional aspects, such as . . . the compatibility requirements of the Sega video game console program,” whereas the Triad ISO copied programs “not to determine their functional aspects, but to use them.”\textsuperscript{208} Copying operational output is necessary for ISOs and their customers to both use and ascertain functional – and therefore noncopyrightable – aspects of the work.\textsuperscript{209} For this reason, ISOs may receive more favorable treatment than the defendant in Triad.

Although ISOs compete with software manufacturers in the service aftermarket just as Southeastern did, McJohn notes that cases such as Sega have found fair use in spite of the fact that the accused video game manufacturer produced cartridges that directly competed with the video console manufacturer.\textsuperscript{210} For all of these reasons, courts should read Triad critically and support a finding of fair use for operational outputs.

4. Merger Doctrine

In examining a work used for functional purposes (repair) and created by a functional work (the program), it is natural to ask if the merger doctrine would prevent manufacturers from using copyright law to prohibit its copying. If there is only a limited number of ways to express an idea, the idea and expression may effectively merge into one whole.\textsuperscript{211} Courts will not allow copyright of this merged expression because of 17 U.S.C. § 102(b)’s bar against copyrighting ideas.\textsuperscript{212}

\textsuperscript{204} Id. at 623 (opining that a work is transformative if it is “independently creative” and not simply “a copy of the original”).
\textsuperscript{205} Id. at 624 (discussing that searching for transformative use is not the best way to frame the issue).
\textsuperscript{206} Id. at 626 (explaining that while the service industry does not produce “works or authorship,” it is a field that benefits from innovation).
\textsuperscript{207} Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510 (9th Cir. 1992).
\textsuperscript{208} See McJohn, supra note 6, at 593.
\textsuperscript{209} Id. at 621.
\textsuperscript{210} Id. at 613 (supporting the notion that a finding of fair use may foster competition in the market to service the copyrighted work). As is typical currently, the console manufacturer also created games for the console.
\textsuperscript{211} COHEN ET AL., supra note 83, at 94.
\textsuperscript{212} 17 U.S.C. § 102(b) (2000) (“In no case does copyright protection for an original work
The “idea” of a program is thought to be its fundamental task.\footnote{Gates Rubber Co. v. Bando Chem. Indus., 9 F.3d 823, 845 (10th Cir. 1993) ("We normally would associate the term “fundamental tasks” with the highest level of abstraction – that is, the ideas or purposes underlying a program.").} In some regards, operational output is simply a description of the program’s task. If there are only a limited number of ways to express this idea, copyright protection may be withheld from the output. But more than one “idea” may exist in a program and, as a result, each subroutine may have its own “idea.”\footnote{See Computer Assocs. Int’l v. Altai, Inc., 982 F.2d 693, 705-06 (2d Cir. 1992) (rejecting Whelan’s inference that there is only one idea in a program); Whelan Assoc., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1236 (3d Cir. 1986).} It was for this reason that the Altai test was developed to allow courts to filter out any merged expression as they tested for infringement.\footnote{Altai, 982 F.2d at 706 (adopting abstraction-filtration-comparison test).} Because the Altai “Abstraction-Filtration-Comparison” test appears to be the best analytical tool for using a merger doctrine defense, this Note explores this test in its analysis of infringement issues in Part III.B.1.\footnote{The Altai court recognized that its filtration approach was really just an implementation of the merger doctrine. See id. at 706 (“This approach breaks no new ground; rather, it draws on such familiar copyright doctrines as merger . . . .”).} As that section concludes, the results of applying Altai to operational outputs are far from certain. Still, the merger doctrine as implemented through an Altai test is unlikely to completely protect copiers of output from liability.

IV. THE DMCA AND THE AFTERMARKET

A. The 17 U.S.C. § 1201 Anti-Circumvention Provision

Even if ISOs can avoid copyright liability by using defenses such as fair use, these defenses may be unavailable when ISOs face suits under the DMCA.\footnote{Some courts have held that copyright defenses are not available to defendants charged with violating the DMCA. See Universal City Studios v. Reimerdes, 111 F. Supp. 2d 294, 304 (S.D.N.Y. 2000) (“Technological access control measures have the capacity to prevent fair uses of copyrighted works as well as foul.”); 3 Nimmer on Copyright, supra note 44, at § 12A.06(B)(2) (agreeing with commentary stating that fair use does not apply to the DMCA because the DMCA is a “separate” tort). But see Lexmark Int’l, Inc. v. Static Control Components, Inc., 387 F.3d 522, 562 (6th Cir. 2004) (asserting that on remand, if copyright defenses were adequate, there would be no DMCA case); see also Storage Tech. Corp. v. Custom Hardware Eng’g & Consulting, Inc., No. 02-12102-RWZ, 2004 WL 1497688 (D. Mass. July 2, 2004) (relying partially on the DMCA to enjoin ISO use of diagnostic output generation software for over a year before the case was vacated by the Federal Circuit in a 2-1 split), vacated, 421 F.3d 1307 (Fed. Cir. 2005).} The DMCA prohibits the circumvention of any technological measure that
effectively controls access to a work protected by copyright. \textsuperscript{218} Software manufacturers can easily prevent ISOs from using these outputs by augmenting their products with controls that limit access to the outputs and then leveraging the DMCA to enjoin any attempts to circumvent these controls. \textsuperscript{219}

In \textit{Storage Tech}, the plaintiff created a password-protected program that controlled access to the generation of its tape drive’s operational output. \textsuperscript{220} The lower court found the defendants had likely violated the DMCA by circumventing a security measure in order to run (and thus copy) the plaintiff’s diagnostic code inside the plaintiff’s hardware. \textsuperscript{221} The court enjoined the ISO defendants partly on the basis of the DMCA. \textsuperscript{222} The Federal Circuit vacated the district court’s injunction in part because § 117(c)’s safe harbor protected the defendant from a finding of copyright infringement. \textsuperscript{223} The Federal Circuit relied primarily on \textit{Chamberlain Group, Inc. v. Skylink Technologies}’ holding that “[a] copyright owner alleging a violation of section 1201(a) . . . must prove that the circumvention of the technological measure either ‘infringes or facilitates infringing a right protected by the Copyright Act.’” \textsuperscript{224}

Yet if operational outputs are not copyrightable, the DMCA cannot prevent

\begin{itemize}
  \item \textsuperscript{218} 17 U.S.C. § 1201 (2000) (addressing circumvention of copyright protection systems).
  \item \textsuperscript{219} See Myron Hecht, \textit{Reconciling Software Technology and Anti-Circumvention Provisions in the Digital Millennium Copyright Act}, 2004 UCLA J.L. & TECH. 3 (“[C]opyright owners can easily modify existing programs to incorporate copy or access control measures.”); Michael Warnecke, \textit{Appeals Courts’ Limitations on DMCA Prompt Lawyers to Redesign Digital Locks}, 69 BNA PAT. TRADEMARK & COPYRIGHT J. 549, 549 (2005) (describing how to design technological controls that are most likely to survive DMCA defense challenges). Because access controls are built in throughout most large software systems, many manufacturers are already situated to make DMCA arguments.
  \item \textsuperscript{220} \textit{Storage Tech}, 2004 WL 1497688, at *3. In this case, the plaintiff may have helped itself by characterizing the circumvention measure not as a simple switch that turned on the diagnostic outputs, but instead as a control that copied “Maintenance Code” into the machine. The code then generated the messages. Framing the code in this manner meant that a copyrighted work (the code) was copied by the defendant without authorization. Interestingly, it remains to be seen whether the program actually functioned in this manner and whether the copyrighted “Maintenance Code” was actually copied by the activation of the access control. Few if any software systems work in this fashion, as most require the output to be switched on or off dynamically without reloading the software system. The \textit{Lexmark} concurrence warned against just this type of factual “tweaking,” stressing that the crucial question was whether the control’s purpose was “to prohibit the pirating of copyright-protected works.” \textit{Lexmark}, 387 F.3d at 552 (Merritt, J., concurring).
  \item \textsuperscript{221} \textit{Storage Tech}, 2004 WL 1497688 at **4-5.
  \item \textsuperscript{222} \textit{Id.} at **9-10 (holding that the plaintiff had claims for both violation of the DMCA and for misappropriation of its trade secret, though emphasizing the DMCA violation in the opinion).
  \item \textsuperscript{223} See \textit{Storage Tech}, 421 F.3d at 1318 (reasoning that because the defendant’s activities did “not constitute copyright infringement or facilitate copyright infringement, \textit{[Storage Tech]} is foreclosed from maintaining an action under the DMCA”).
  \item \textsuperscript{224} \textit{Id.} at *9 (citing \textit{Chamberlain}, 381 F.3d 1178, 1203 (Fed. Cir. 2004)).
\end{itemize}
access to them. To prove a DMCA violation under § 1201(a)(2), the Federal Circuit requires that the circumvention capability infringe or otherwise facilitate the infringement of a right protected by the Copyright Act. Otherwise, even circumventing a burglar alarm to get access to copyrighted books inside a home would be a violation of the DMCA. In Chamberlain, an original equipment manufacturer of garage door openers sued an aftermarket vendor whose universal door opener circumvented the door opener’s code in order to work with the original equipment. Lacking proof of any copyright infringement, the Chamberlain court refused to find liability under the DMCA. Similarly, if operational outputs are not copyrightable, the DMCA could not be used to prosecute ISOs that circumvent the technological measures used to protect the outputs. If operational outputs are copyrightable, however, even ISOs who create circumvention tools that merely enable the copying of outputs may violate the DMCA.

The Chamberlain court said it did not reach the question of whether § 107 fair use defenses could apply to violations of the DMCA, though as mentioned above, some courts and commentators believe that fair use is inapplicable to DMCA violations. But Chamberlain implies that if the accessed material is uncopyrightable, then there can be no liability. Thus the outcome of these types of cases may rest entirely upon whether the facts or the doctrines applied to them are characterized as defenses or as bars to copyright.

---

225 Chamberlain, 381 F.3d at 1204 (finding copyright infringement to be at the “nexus” of DMCA cases). Without this requirement of infringement under the Copyright Act, the Chamberlain court worried that the DMCA would “allow any manufacturer of any product to add a single copyrighted sentence or software fragment to its product, wrap the copyrighted material in a trivial ‘encryption’ scheme, and thereby gain the right to restrict consumers’ rights to use its products in conjunction with competing products.” Id. at 1201.

226 See id. at 1200-01 (illustrating the nonsensical result that would follow from the plaintiff’s “proposed severance of ‘access’ from ‘protection’” when interpreting § 1201(a)). Instead, the circumvention must be “intertwined with a protected right.” See id. at 1199.

227 See id. at 1183-86 (describing the dispute).

228 See id. at 1204 (“In the absence of allegations of either copyright infringement or § 1201(a)(1) circumvention, Skylink cannot be liable for § 1201(a)(2) trafficking.”).

229 See id. (suggesting that a copyright owner can impose liability by showing that the “accused trafficker’s device enables either copyright infringement or a prohibited circumvention”).

230 Id. at 1200.

231 See supra note 217 and accompanying text (discussing applicability of fair use to the DMCA).

232 The Federal Circuit recently hinted strongly, however, that a valid defense to copyright infringement would also be a defense to a §1201 DMCA claim. See Storage Tech, Corp. v. Custom Hardware Eng’g & Consulting, Inc., 421 F.3d 1307, 1318 (Fed. Cir. 2005) (indicating that the defendant’s 17 U.S.C. § 117 maintenance/repair defense was precluded, and commenting that “[t]o the extent that CHE’s activities do not constitute
There is already a circuit split over whether the merger doctrine acts as a bar to copyrightability or as a defense to some types of infringement.\(^{233}\) If the merger doctrine prevents copyrightability, the DMCA would not apply, but if it is only a defense to infringement, it will provide no defense whatsoever against the DMCA. That the outcome of a case against an ISO might turn on such a fine point seems inconsistent with Congress’s stated intentions during the adoption of the DMCA. For example, the antitrust underpinnings of copyright misuse were of considerable concern to the DMCA’s framers, who did not intend the DMCA to be used offensively to create monopolies.\(^{234}\) Yet copyright misuse is characterized as a defense and thus presumably does not prevent software manufacturers from using the DMCA to establish a monopoly in software maintenance.\(^{235}\)

For these reasons, the efficacy of the DMCA in preventing ISOs from gaining access to operational outputs is both unclear and at apparent odds with the anti-monopoly concerns that prompted enactment of the Act. Regardless, the copyrightability of these outputs will play a major role in this issue’s outcome.


At first glance, § 1201(f) seems like a promising defense for ISOs against the DMCA’s anticircumvention bans. Section 1201(f) allows legitimate users of programs to circumvent technological measures “for the sole purpose of identifying and analyzing those elements of the program that are necessary to achieve interoperability of an independently created computer program with other programs.”\(^{236}\) “Interoperability” means the ability of computer programs to exchange information and to make mutual use of the exchanged information.\(^{237}\)

---

\(^{233}\) Lexmark Int’l, Inc. v. Static Control Components, Inc., 387 F.3d 522, 557 (6th Cir. 2004) (Feikens, J., concurring in part and dissenting in part) (“The Sixth Circuit has not previously taken a position on this question. The Second and Ninth Circuits have taken the position that merger operates only as a defense to infringement. The Fifth Circuit holds that merger determines copyrightability.”) (citations omitted).

\(^{234}\) See H.R. REP. NO. 105-551(II), at 26 (1998) (reporting that the Committee on Commerce felt compelled to address the risk of the DMCA placing a “monopoly stranglehold on information”); Lexmark, 387 F.3d at 553 (Feikens, J., concurring in part and dissenting in part) (agreeing with majority that the DMCA “was not intended by Congress to be used to create a monopoly in the secondary markets for parts or components of products that consumers have already purchased”).

\(^{235}\) See 4 NIMMER ON COPYRIGHT, supra note 44, at § 13.09(A)(2) (indicating that many courts deny outright the antitrust violation defense to copyright infringement actions).


Defendants have successfully used § 1201(f)’s safe harbor to allow
circumvention for reverse-engineering of toner cartridges,\(^\text{238}\) and have argued
it as a defense for reverse-engineering garage-door openers.\(^\text{239}\) Yet few cases
have addressed this section of the DMCA.\(^\text{240}\)

Interoperability concerns occur frequently in large software systems. Few
programs exist in a vacuum, and many interact with dozens of other programs.
The problems ISOs resolve likely affect or are affected by other programs.
Thus ISOs can argue that programs whose operational outputs are protected by
anti-circumvention controls must be “analyzed” to “achieve interoperability”
between these programs.

This interpretation of § 1201(f)’s terms, though somewhat stretched, might
provide ISOs with a safe harbor, but courts appear reluctant to read the
DMCA’s exceptions broadly.\(^\text{241}\) Also, § 1201(f) only provides safe harbor
when no copyright infringement has occurred.\(^\text{242}\) If an ISO infringes upon a
copyrighted operational output, the ISO has no defense to the DMCA for any
circumvention the ISO uses to access the outputs. Thus the copyrightability of
operational outputs will likely be dispositive for DMCA actions against
aftermarket service providers. Copyright simultaneously enables a DMCA
offense\(^\text{243}\) and disables any DMCA defense.

who had created a “back door” access to the plaintiff’s online game had not developed an
“independently created computer program”).

\(^\text{238}\) *Lexmark*, 387 F.3d at 550-51 (6th Cir. 2004) (holding that the plaintiff failed to
disprove the defendant’s evidence that the SMARTEK chip contained independently created
computer programs).

\(^\text{239}\) Chamberlain Group, Inc. v. Skylink Techs., Inc., 381 F.3d 1178, 1201 n.15 (Fed. Cir.
2004) (determining that § 1201(f) was irrelevant because it was an affirmative defense and
concluding that the plaintiff failed to establish a prima facie violation of the DMCA).

\(^\text{240}\) As of March 19, 2005, only three cases have even mentioned the section.

that the result of this broad reading is that “[a]lthough the [§ 1201] statutory exemptions
were intended to provide some breathing space for legitimate activities, they have in
practice provided less breathing space than initially envisioned”). However, the *Lexmark*
court seemed to give broader allowances for reverse-engineering, questioning whether the
“technological means” must be necessary at all for interoperability. *Lexmark*, 387 F.3d at
550-51. Furthermore, the court did not require the program with which the interoperability
was to occur to have existed prior to the reverse-engineering activity. *Id.* Most of this was
dicta because the toner cartridge would not have worked at all without the circumvention
involved in the defendant’s reverse-engineering.

\(^\text{242}\) *Lexmark*, 387 F.3d 522 at 546 (commenting that all three reverse-engineering
defenses apply “when traditional copyright infringement does not occur and only when the
challenged actions (in the case of the third provision) would not violate other ‘applicable
law[s]’”) (alteration in original); *Davidson & Assocs.*, 334 F. Supp. 2d at 1187 (holding that
the defendants who violated DMCA could not benefit from the 1201(f) exemption because
they had violated the plaintiff’s copyright in their game).

\(^\text{243}\) See discussion supra Part IV.
V. POSSIBLE REMEDIAL MEASURES

Consumers suffer when decreased competition in the software industry results in higher prices, less innovation, and lower standards of service. As examined above, current statutes and judicial interpretation give manufacturers potent tools to prevent ISOs from providing consumers with competitive alternatives for software and hardware servicing. There are many possible approaches to remedying this problem, though only a few appear to provide meaningful solutions in the near term.

A fair use defense, even if it can be made, will generally be a poor solution for ISOs. Fair use is a “fact-specific matter of balancing” that “introduces an uncertainty that commercial actors dislike.” Most manufacturers are likely to be better financed than the ISOs, so any solutions requiring the considerable legal expenses associated with discovery-intensive court proceedings may be almost as bad as no solution at all. Even a temporary injunction could be enough to severely injure an ISO’s business.

Antitrust and the related doctrine of copyright misuse would be an obvious way of fighting any alleged monopolization of the maintenance and operations market. However, these causes of action would typically require market analysis, and like fair-use defenses, would produce the specter of protracted discovery and litigation, with few opportunities for early summary judgment.

Software consumers can participate in preventing copyright from unnecessarily reducing competition by contract. Both individually and through user groups, consumers could attempt to force manufacturers to allow the use of operational outputs by the customer’s agents and vendors. Many large-scale software contracts are highly customized, which should provide self-help opportunities for at least those purchasers with large buying power who are in a position to make a first purchase or to renegotiate upgrades and license continuations.

A similar result could be affected judicially if courts allow independent contractors to “stand in the shoes of the user” so that ISOs get the same § 107 fair use and § 117 rights afforded to software purchasers. In so doing, courts would effectively force manufacturers who still persisted in filing a suit to sue “through” the ISOs to their own customers by operation of the doctrine of respondeat superior. Even if these defendants are theoretically likely to lose

244 David A. Vise, Editorial, Lawsuit Forces Oracle to Change Tactics, WASH. POST, Feb. 28, 2004, at E1 (reporting that Oracle had to change tactics in its hostile takeover bid for PeopleSoft when the Justice Department filed its antitrust lawsuit).

245 See, e.g., LAWRENCE LESSIG, FREE CULTURE 187 (2004) (worrying that “fair use in America simply means the right to hire a lawyer to defend your right to create”).

246 McJohn, supra note 6, at 635.

247 Interview with David York, supra note 28.

248 See MERGES ET AL., supra note 122, at 1046 (stating that the first step in any antitrust action is defining the relevant market).

249 Gibbons, supra note 27, at 546.
(because § 117 and § 107 do not technically allow copying of operational outputs), business realities (and perhaps some corporate sense of shame) are likely to prevent lawsuits appearing to sue customers for using the software they purchased. Such a “judicial gloss” could better enable software consumers to fully and efficiently exercise both their contract and copyright rights. This could also be accomplished legislatively.

ISOs would be better served by fixing the problem at its heart – by modifying the IP rights that create the plaintiff’s cause of action itself. Better legal approaches for ISOs would state that this use of operational outputs is *per se* not a violation of copyright. This could be accomplished in a number of ways.

Lawmakers can prevent the DMCA from continuing to cause problems in the software aftermarket by extending the ambit of 17 U.S.C. § 1201 to allow circumvention for service and maintenance in addition to the existing reverse-engineering exception. The DMCA’s bite on ISOs might also be defanged by using the existing exception ruling process established by § 1201(a)(1). This section mandates that the Librarian of Congress periodically determine and publish a list of classes of copyrighted works which will be exempted from certain prohibitions of the DMCA for the next three year period.

The most recent anticircumvention ruling regarding exempted classes became final on October 28, 2003, and the next ruling must occur before the same date in 2006. Four new classes were added in the 2003 ruling, but two proposals related to the copying of software programs were not allowed. Although operational outputs have not yet been proposed, failed proposals have previously included embedded computer programs in both toner cartridges and in products where the program “cannot be copied during the ordinary operation or use of the machine.”

Proponents of exemptions have the burden of proving that the DMCA will have, or is likely to have, a substantial adverse effect on non-infringing uses of

---

250 Id.


252 Exemption to Prohibition Against Circumvention, 37 C.F.R. § 201.40 (2003) (providing for categories of works to be exempted from DCMA protection for the period of October 28, 2003 through October 27, 2006). The four classes of works are “[c]ompilations consisting of lists of Internet locations blocked by commercially marketed filtering software,” “[c]omputer programs protected by dongles that prevent access due to malfunction or damage,” “[c]omputer programs and video games distributed in formats that have become obsolete,” and “[l]iterary works distributed in ebook format when all existing ebook editions of the work . . . contain access controls that prevent the enabling of the ebook’s read-aloud function.” Id.

the proposed class of works. Although current DMCA lawsuits may be evidence of harm, the decision on whether the use of operational outputs is non-infringing is far from clear. This uncertainty may create a serious impediment for ISOs using § 1201(a)(1). The current exemptions are only in effect until 2006, leaving little time to build a case for a new proposal. For these reasons, a DMCA exemption ruling appears to be a poor solution.

Lawmakers could also amend 17 U.S.C. § 117 in a manner similar to the Computer Maintenance Competition Assurance Act’s amendment to § 117(c). The purpose of § 117(c) was to allow hardware servicing, and this amendment would simply recognize the importance of operational outputs for servicing of both hardware and software. Congress could determine that it is not infringement for software and hardware owners (or licensees) to authorize copies of any software-created output. To limit this grant of power, Congress could require that such copies be used for hardware/software maintenance and be destroyed after use.

CONCLUSION

There are strong arguments that the current copyright and DMCA regimes allow software manufacturers to inhibit ISOs from copying operational outputs. The defenses provided by the DMCA and the Copyright Act appear to be of little use in avoiding or defending legal battles over the use of operational outputs.

Given the possible remedial measures, an amendment to 17 U.S.C. § 117 allowing copying of operational outputs appears to be the most permanent and expedient solution for ISOs and their customers. Customers who are positioned to negotiate contractual allowances for ISO use of operational outputs should obviously attempt to do so, though this solution, like all negotiations, is likely to come at some cost. Through these measures, the software aftermarket can remain a vibrant and innovative part of the world economy.


255 See supra note 252 and accompanying text.

256 See supra note 9.
APPENDIX: OPERATIONAL OUTPUT EXAMPLE

APE_TRACELOG_MSG : 10/22/04 16:46:56.084 : ni00NiGuiSms1 - The Original Report is NOT Split!!! (FCCReportEvent.C, 589)
APE_TRACELOG_MSG : 10/22/04 16:46:57.412 : ni00NiGuiSms2 - Turned trace on at level [2013265920x78000000]
APE_TRACELOG_MSG : 10/22/04 16:46:57.426 : ni00NiGuiSms2 - !MEMORY[2440340]INITIAL (CPACProgramControl.C, 109)
APE_TRACELOG_MSG : 10/22/04 16:46:59.157 : ni00NiGuiSms2 - !PERF|142881238|-1|-1|-1|DispRcv| (DBEvent.C, 10624)
APE_TRACELOG_MSG : 10/22/04 16:46:59.180 : ni00NiGuiSms2 - MgrDispatch(1) ID: 142881238
APE_TRACELOG_MSG : 10/22/04 16:46:59.181 : ni00NiGuiSms2 - EVENT_NIThousandsBlockRequestEvent/EVST_FCC_REPORT_USER_MODIFY STATE_PROCE
SSING/EVSS_NI_USER_MODIFY_FCC_REPORT IntfId: 0
(EventsMgr.C, 421)
APE_TRACELOG_MSG : 10/22/04 16:46:59.181 : ni00NiGuiSms2 - RGN_PK=0 TN= SVID=0 osrv=NiGuiServer nsrv=NiGuiSms
ISPID= _lockStatus=0
APE_TRACELOG_MSG : 10/22/04 16:46:59.181 : ni00NiGuiSms2 - STATUS-MSG=0-
APE_TRACELOG_MSG : 10/22/04 16:46:59.205 : ni00NiGuiSms2 - Fetch logical query: Select xml_pk from ni_fcc_report_xml
where report_fk = 332 ORDER BY xml_pk (DBNiFCCReport.C, 3098)
APE_TRACELOG_MSG : 10/28/04 23:57:58.865 : ni00NiClient - THRD-001 - CONNECTING 'UtilizationNotification'
(NiCorbaObject.C, 461)

257 The included output is from one of the systems that allows national number portability for cell phones. The system is produced by Evolving Systems, which is located in Englewood, Colorado. Evolving Systems Home Page, http://www.evolving.com (last visited Nov. 18, 2005). The system processes thousands of transactions every day to enable cell phone users to keep their phone numbers even when switching communications providers.
APE_TRACELOG_MSG : 10/28/04 23:57:58.865 : ni00NiClient -
Trying to connect to Local Naming Service
(CorbaNamingService.C, 707)
APE_TRACELOG_MSG : 10/28/04 23:57:58.868 : ni00NiClient -
Throw LnpXInternalError("Cannot resolve name:
pto/UtilizationNotification") (CorbaNamingService.C, 626)
APE_TRACELOG_MSG : 10/28/04 23:57:58.868 : ni00NiClient -
THRD-001 - NiCorbaObject::Resolve Failure to resolve
'UtilizationNotification' (NiCorbaObject.C, 778)
APE_SYSLOG_MSG : 10/26/04 02:46:12.627 : ni00NiGuiSms1
[60901] - 60901 - (60010,yogi,142881789) DN [2077764000]
already exists for add 2077764000, Orig Server:NiGuiServer
Next Server:NiGuiSms
APE_SYSLOG_MSG : 10/26/04 02:48:03.125 : ni00NiGuiServer
Cannot load inventory if the OCN is not the Code Holder
unless Ownership Validation is turned OFF., Orig
Server:NiGuiServer Next Server:NiGuiServer