Journal of the American Society for Information Science and Technology
Article No. 8n1098

Copy of e-mail Notification

Your article (JA1827) from Journal of the American Society for Information Science and Technology is available for download
=====
Journal of the American Society for Information Science and Technology Published by John Wiley & Sons, Inc.

Dear Author,

Your page proofs are available in PDF format; please refer to this URL address
http://mothra.cadmus.com/cgi-bin/s-proof/login?825221

Login: your e-mail address
Password: ----

The site contains 1 file. You will need to have Adobe Acrobat Reader software to read these files. This is free software and is available for user downloading at http://www.adobe.com/products/acrobat/readstep.html. If you have the Notes annotation tool (not contained within Acrobat reader), you can make corrections electronically and return them to Wiley as an e-mail attachment (see the Notes tool instruction sheet). Alternatively, if you would prefer to receive a paper proof by regular mail, please contact Staci Westerhoff (e-mail: westerhoffs@cadmus.com, phone: 800-238-3814 x662 or 717-721-2662). Be sure to include your article number.

This file contains:

Author Instructions Checklist
Adobe Acrobat Users - NOTES tool sheet
Reprint Order form
Copyright Transfer Agreement
Return fax form
A copy of your page proofs for your article

After printing the PDF file, please read the page proofs carefully and:

1) indicate changes or corrections in the margin of the page proofs;
2) answer all queries (footnotes A,B,C, etc.) on the last page of the PDF proof;
3) proofread any tables and equations carefully;
4) check that any Greek, especially "mu", has translated correctly.

Special Notes:

1. Figure(s) _____ are unacceptable for publication. Please supply good quality hard copy (and/or TIFF or EPS files) when you return your page proofs.

Within 48 hours, please fax or e-mail the following to the address given below:

1) original PDF set of page proofs,
2) print quality hard copy figures for corrections and/or TIFF or EPS files of figures for correction (if necessary),
3) Signed Copyright Transfer Agreement,
4) Reprint Order form,

Copy of e-mail Notification

5) Return fax form

Return to:

Paul Dlugokencky
Professional/Trade Journals Production
John Wiley & Sons, Inc.
111 River Street
Hoboken, NJ 07030
U.S.A.
(See fax number and e-mail address below.)

If you experience technical problems, please contact Staci Westerhoff (e-mail: WesterhoffS@cadmus.com, phone: 800-238-3814 (X662)or 717-721-2662.

If you have any questions regarding your article, please contact me. PLEASE ALWAYS INCLUDE YOUR ARTICLE NO. (JA1827) WITH ALL CORRESPONDENCE.

Sincerely,

Paul Dlugokencky
Sr. Associate Managing Editor
John Wiley & Sons, Inc.
E-mail: pdlugoke@wiley.com
Tel: 201-748-8893
Fax: 201-748-6052

WILEY
Publishers Since 1807

JOHN WILEY & SONS
605 THIRD AVENUE, NEW YORK, NY 10158

*** * * IMMEDIATE RESPONSE REQUIRED * * ***

Your article will be published online via Wiley's EarlyView service (www.interscience.wiley.com) within a few days of correction receipt. Online publication includes full-text HTML and paginated, fully citable PDF.

☐ **READ PROOFS CAREFULLY**

- This will be your only chance to review these proofs.
- Please note that the volume and page numbers shown on the proofs are for position only.

☐ **ANSWER ALL QUERIES ON PROOFS** (Queries for you to answer are noted on the manuscript.)

- Mark all corrections directly on the proofs, not on the manuscript. Note that excessive author alterations may ultimately result in delay of publication and extra costs may be charged to you.

☐ **CHECK FIGURES AND TABLES CAREFULLY** (Color figures will be sent under separate cover.)

- Check size, numbering, and orientation of figures. Check quality of figures directly from the galley proofs. The reproduction is 1200dpi, and although it is not indicative of final printed quality, it is adequate for checking purposes.
- Review figure legends to ensure that they are complete.
- Check all tables. Review layout, title, and footnotes.

☐ **COMPLETE REPRINT ORDER FORM**

- Fill out the attached reprint order form. It is important to return the form even if you are not ordering reprints. You may, if you wish, pay for the reprints with a credit card. Reprints will be mailed only after your article appears in print. This is the most opportune time to order reprints. If you wait until after your article comes off press, the reprints will be considerably more expensive.

☐ **ADDITIONAL COPIES**

- If you wish to purchase additional copies of the journal in which your article appears, please contact Jill Gottlieb at (212) 850-8839, fax (212) 850-6021, or E-mail at jgottlieb@wiley.com

RETURN

☐ **PROOFS**
☐ **ORIGINAL MANUSCRIPT**
☐ **REPRINT ORDER FORM**
☐ **ORIGINAL FIGURES**
☐ **Copyright Transfer Agreement (If you have not already signed one)**

RETURN WITHIN 48 HOURS OF RECEIPT VIA FAX TO 212-850-6052

QUESTIONS? Paul Dlugokencky, Sr. Associate Managing Editor
Telephone: 212-850-8893
E-mail: pdlugoke@wiley.com
Refer to JASIST article #_____

Softproofing for advanced Adobe Acrobat Users - NOTES tool

NOTE: ACROBAT READER FROM THE INTERNET DOES NOT CONTAIN THE NOTES TOOL USED IN THIS PROCEDURE.

Acrobat annotation tools can be very useful for indicating changes to the PDF proof of your article. By using Acrobat annotation tools, a full digital pathway can be maintained for your page proofs.

The NOTES annotation tool can be used with either Adobe Acrobat 3.0x or Adobe Acrobat 4.0. Other annotation tools are also available in Acrobat 4.0, but this instruction sheet will concentrate on how to use the NOTES tool. Acrobat Reader, the free Internet download software from Adobe, DOES NOT contain the NOTES tool. In order to softproof using the NOTES tool you must have the full software suite Adobe Acrobat Exchange 3.0x or Adobe Acrobat 4.0 installed on your computer.

Steps for Softproofing using Adobe Acrobat NOTES tool:

1. Open the PDF page proof of your article using either Adobe Acrobat Exchange 3.0x or Adobe Acrobat 4.0. Proof your article on-screen or print a copy for markup of changes.

2. Go to File/Preferences/Annotations (in Acrobat 4.0) or File/Preferences/Notes (in Acrobat 3.0) and enter your name into the "default user" or "author" field. Also, set the font size at 9 or 10 point.

3. When you have decided on the corrections to your article, select the NOTES tool from the Acrobat toolbox and click in the margin next to the text to be changed.

4. Enter your corrections into the NOTES text box window. Be sure to clearly indicate where the correction is to be placed and what text it will effect. If necessary to avoid confusion, you can use your TEXT SELECTION tool to copy the text to be corrected and paste it into the NOTES text box window. At this point, you can type the corrections directly into the NOTES text box window. **DO NOT correct the text by typing directly on the PDF page.**

5. Go through your entire article using the NOTES tool as described in Step 4.

6. When you have completed the corrections to your article, go to File/Export/Annotations (in Acrobat 4.0) or File/Export/Notes (in Acrobat 3.0). Save your NOTES file to a place on your harddrive where you can easily locate it. **Name your NOTES file with the article number assigned to your article in the original softproofing e-mail message.**

7. **When closing your article PDF be sure NOT to save changes to original file.**

8. To make changes to a NOTES file you have exported, simply re-open the original PDF proof file, go to File/Import/Notes and import the NOTES file you saved. Make changes and re-export NOTES file keeping the same file name.

9. When complete, attach your NOTES file to a reply e-mail message. Be sure to include your name, the date, and the title of the journal your article will be printed in.

John Wiley & Sons, Inc.
Publishers Since 1807

REPRINT BILLING DEPARTMENT • 605 THIRD AVENUE • NEW YORK, NY 10158-0012
PHONE: (212) 850-8789; FAX: (212) 850-6326
INTERNET: reprints @ wiley.com

PREPUBLICATION REPRINT ORDER FORM

Please complete this form even if you are not ordering reprints. This form **MUST** be returned with your corrected proofs and original manuscript. Your reprints will be shipped approximately 4 to 6 weeks after publication. Reprints ordered after printing are substantially more expensive.

JOURNAL: *JRNL OF THE AMERICAN SOC. FOR INFO. SCIENCE* VOLUME____ ISSUE____
TITLE OF MANUSCRIPT_____
MS. NO._____ NO. OF PAGES_____ AUTHOR(S)_____

REPRINTS 8 1/4 X 11

No. of Pages	100 Reprints	200 Reprints	300 Reprints	400 Reprints	500 Reprints
	$	$	$	$	$
1-4	336	501	694	890	1052
5-8	469	703	987	1,251	1,477
9-12	594	923	1,234	1,565	1,850
13-16	714	1,156	1,527	1,901	2,273
17-20	794	1,340	1,775	2,212	2,648
21-24	911	1,529	2,031	2,536	3,037
25-28	1,004	1,707	2,267	2,828	3,388
29-32	1,108	1,894	2,515	3,135	3,755
33-36	1,219	2,092	2,773	3,456	4,143
37-40	1,329	2,290	3,033	3,776	4,528

** REPRINTS ARE ONLY AVAILABLE IN LOTS OF 100. IF YOU WISH TO ORDER MORE THAN 500 REPRINTS, PLEASE CONTACT OUR REPRINTS DEPARTMENT AT (212)850-8789 FOR A PRICE QUOTE.

COVERS

100 Covers - $90 • 200 Covers - $145 • 300 Covers - $200
400 Covers - $255 • 500 Covers - $325 • Additional 100s - $65

¤ Please send me _____ reprints of the above article at..... $_____
¤ Please send me _____ covers of the above journal at..... $_____
Please add appropriate State and Local Tax {Tax Exempt No._____} $_____
Please add 5% Postage and Handling..... $_____
TOTAL AMOUNT OF ORDER** ..... $_____
***International orders must be paid in U.S. currency and drawn on a U.S. bank*

Please check one: ¤ Check enclosed ¤ Bill me ¤ Credit Card
If credit card order, charge to: ¤ American Express ¤ Visa ¤ MasterCard ¤ Discover
Credit Card No._____ Signature_____ Exp. Date_____

Bill To:
Name_____
Address_____
_____
_____
_____
Purchase Order No. _____

Ship To:
Name_____
Address_____
_____
_____
_____
Phone_____ Fax _____
Internet:_____

Open Source Software Development and Lotka's Law: Bibliometric Patterns in Programming

Gregory B. Newby, Jane Greenberg, and Paul Jones
University of North Carolina at Chapel Hill, 100 Manning Hall, CB#3360, Chapel Hill, NC 27599-3360

This research applies Lotka's Law to metadata on open source software development. Lotka's Law predicts the proportion of authors at different levels of productivity. Open source software development harnesses the creativity of thousands of programmers worldwide, is important to the progress of the Internet and many other computing environments, and yet has not been widely researched. We examine metadata from the Linux Software Map (LSM), which documents many open source projects, and Sourceforge, one of the largest resources for open source developers. Authoring patterns found are comparable to prior studies of Lotka's Law for scientific and scholarly publishing. Lotka's Law was found to be effective in understanding software development productivity patterns, and offer promise in predicting aggregate behavior of open source developers.

Introduction

The act of writing software has many features in common with the act of authoring journal articles or scholarly books, although there are also apparent differences. In this article, we investigate whether patterns of productivity in software development are similar to patterns for authoring scientific and scholarly works. This research examines programmer productivity patterns documented in metadata from two leading open source software development resources—the Linux Software Map (LSM) and Sourceforge.

Our goal is to look at patterns of productivity in software development, and evaluate the extent to which these patterns are similar to previously discovered patterns in scholarly writing. In particular, we examine the match of programming productivity patterns with the predictions of Lotka's Law (Lotka, 1926), sometimes called "the inverse square law of scientific productivity" (Lotka, 1926, p. 320).

This article first considers the programmer as an author and the parallel between software development and publishing. Next, Lotka's Law is reviewed followed by a discussion of our methodology and a report of our data analysis. The last two sections of this article present the statistical significance of our results and consider further plans for bibliometric research on programmer productivity and predicting programmer contributions to software projects.

Programmers as Authors

The on-line version of the Oxford English Dictionary (OED) (http://dictionary.oed.com) defines author as a "person who originates or gives existence to anything," an "inventor, constructor, or founder." This is an inclusive notion of the author, one that pertains to people or corporate entities that may create any number of works, including computer programs. We believe that software developers are authors.

Software development (also known as *software engineering*) involves many acts, from defining the initial problem to testing the output. Central to this activity is programming—the act of writing instructions for computers in a coding language (such as C or PERL). Software developers (programmers) might work alone or in groups, as part of a large organization or as an individual hobbyist. In all cases, they are seeking to turn their ideas into instructions for a computer to follow.

In the last quarter of the 20th century, philosophers and literary scholars such as Michel Foucault (1977) have written about and extended the notion of "author" beyond the concept of creating literary texts. Our change in understanding is reflected in the development of metadata schemas that have an "author" element for all types of materials. Below, we will see this realized in the structure of the Linux Software Map (LSM). Other contemporary examples include the revision of library cataloging rules via the Functional Requirements for Bibliographic Record (FRBR, 1997) for national and international bibliographic control and the Dublin Core Metadata Element Set, Version 1.1 (1999). Other recent discussions also address the literary author in the networked world (e.g., Agree, 1994). Programmers clearly fit the extended contemporary definition of

Received October 5, 2001; Revised March 7, 2002; Accepted March 28, 2002

 ● Published online 00 Month 2002 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/asi.10177

authorship. The research presented here examines the extent to which programmer productivity fits a comparable pattern to the productivity of authors who write scientific and scholarly works, which is the principal subject domain Lotka's Law has been applied to previously (see White & McCain 1989).

Open Source Software Development

This article is primarily concerned with a particular type of software development, *open source software development*. "Open source" refers to the notion that the encoded instructions (the source code) for the programs produced by the software development team are openly available for scrutiny by potential users, competitors, and others. The authors and their colleagues have an ongoing interest in examining the phenomena associated with open source software development (Dempsey, Weiss, Jones, & Greenberg, 1999, 2001), as do other researchers (e.g., Mockus, Fielding, & Herbsleb, 2000).

Open source software development may be thought of as contrary to *closed source software development*, where the source code is intentionally restricted from everyone but the organization doing the software development and its partners. The primary goal of keeping the source code secret is typically to enhance the profit potential for the software, by minimizing advantages to competition. Note that closed sources does not always mean commercial or for profit. Similarly, open source software does not mean that the software is necessarily free, nor does it mean that all efforts to control distribution or use of the software are abandoned. In fact, many open source programs place restrictions on what the program can be used for, by whom, and with what consequences or expectations.

One of the goals of open source software development is to enhance the quality of the software by increasing scrutiny for errors by more programmers, and by soliciting participation by volunteer programmers. This goal could help increase the market share of companies that choose to engage in open source software projects, despite competition from closed source alternatives (see Raymond, 1999, for more on this perspective).

As mentioned above, open source software is not necessarily free software. We can think of "free" as a spectrum that ranges from public domain at one end to highly restricted at the other. Software that is granted by the author(s) to the public domain has no restrictions on use: the software is truly free for any conceivable use, including copying, modification and redistribution. Under U.S. and international copyright laws, any programs or program segments that are written are copyrighted by the author(s) if they are not explicitly granted to the public domain. An individual, a group or a corporation may own the copyright, depending on who wrote the software.

Within copyright provisions, reuse or redistribution of source code is quite limited. Nevertheless, there is the latitude for open source software developers: they can choose to grant a license for the use of their copyrighted works. Such a license may allow virtually unlimited use of the software, similar to granting to the public domain, or it may allow very few uses. An extensive comparison of different software licenses is provided by the Free Software Foundation (http://www.fsf.org/licenses/license-list.html). For our purposes, the importance of these distinctions is that we will evaluate data from two sources of open source software development where many different software projects are supported. Most of the projects fall into what the Free Software Foundation (FSF) calls "free software" because the source code is available under one of the many licensing schemes that the FSF considers free.

As discussed above, open source software may not necessarily be used freely for any purpose, even if it is "free." Richard M. Stallman of the Free Software Foundation is a vocal proponent of the distinction between open source software and free software (see http://www.fsf.org for his writings on this topic). Free software, he argues, is software that may be used by anyone for any purpose, including modification of the source code. In addition, free software is licensed to stay free: any modifications must also be released under the same free software license. We will examine two core resources for developers of free software: Sourceforge and the Linux Software Map (LSM). These will be discussed further below.

Historically, some of the most important and widely used programs in the world were developed as free open source projects. Examples include Sendmail (Costalles & Allman, 1997) and BIND (Albitz & Liu, 2001), two programs that are central to the operation of the modern Internet. Today, many open source software development projects are related to Linux, an open source kernel for an implementation of the Unix operating system (see DiBona, Ackerman, & Stone, 1999). For the kernel to be useful, a wide range of additional software must be available to provide a user interface, system functionality, and interoperability. Some companies, such as RedHat and IBM, have made commitments to fostering and supporting open source development for their commercial products to be more viable. (Such support is not entirely altruistic, as it is extremely beneficial to the companies to have access to software they do not own or pay for.)

AQ: 3

In open source software development, programmers are often volunteers: people with an interest become participants in projects. The actual timetable for participation, the number of hours spent working on the project, and other details are up to the volunteer, possibly with constraints from the project, the volunteer's work or home environment, and other factors. In this research, we hypothesize that patterns of productivity among open source software developers (including programmers and others involved in a particular open source project) should approximate the patterns of scholarly authors. Our investigation is based on software developer membership in open source software groups documented by metadata from Sourceforge and the

LSM, specifically as compared to predictions from Lotka's Law.

Lotka's Law

Bibliometrics, a term introduced by Pritchard (1969), is research that uses mathematical formulas to quantify the productivity distribution and yield of communication output and allows us to predict and study scientific progress. Information scientists interested in bibliometrics would like to understand and, if possible, predict authoring behavior. Extensive reviews of bibliometric concepts and methods may be found in Narin and Moll (1977), White and McCain (1989), and Sengupta (1992). An important and frequently studied area of bibliometrics is author productivity, which may be measured by the number of publications a particular author has. Lotka's Law is frequently applied to such studies.

Alfred Lotka performed numerical analysis for the Metropolitan Life Insurance Company in New York. In 1926, his article "The frequency distribution of scientific productivity" was published in the *Journal of the Washington Academy of Sciences*. This brief article presented Lotka's analysis of the number of primary (first) authors and their number of publications as listed in two sources, *Chemical Abstracts* and *Geschichtstafeln der Physik* (Auerbach, 1910). *Chemical Abstracts* was limited to authors with last names beginning with A or B from the period 1907–1916 (1543 publications); the Auerbach source included all publications listed up to the year 1900 (1,325 publications). Potter (1981) provides an extensive treatment of Lotka's method, including criticism of his approach and, among other things, a conclusion that Lotka's own data did not necessarily fit what is now known as Lotka's Law (supported by Pao, 1985).

Briefly stated, Lotka's Law stipulates that as the frequency of publications increases, the approximate number of authors with that frequency of publications may be predicted. The equation for this relationship may be stated several ways, depending on whether raw frequency data or percentages are used, and on whether any sort of transformation (usually a log transform) of the data is made. Here, we state Lotka's Law in Equation 1:

$$E_i = E_1 i^n \quad (1)$$

where E is the expected number of persons with i frequency of publications; E_1 is the number of persons at frequency 1 (the number of persons with one publication); i is the number of publications of interest (1, 2, etc.); n is an exponent that is constant for a set of data.

This is a somewhat different formulation than Lotka's original, but we find it is a better equation for explaining what Lotka's Law predicts, and is numerically and conceptually equivalent. Lotka's original formulation was:

$$x^n y = \text{const} \quad (2)$$

where x is the number of publications of interest (1, 2, etc.), n is an exponent, which is constant for a set of data, y is the expected percentage of persons with x frequency of publications, const is a constant value, which holds when y is a percentage (later literature by others often refers to this constant as C).

Other interpretations and formulations for Lotka's Law have appeared in the literature (see Fang and Fang, 1995; Pao, 1985; Nichols, 1986, for examples). AQ: 5

Lotka's Law is often thought to fix the value of the exponent n at 2; this is Lotka's generalized "inverse square law of scientific productivity" (Lotka, 1926, p. 320). In fact, AQ: 6 the *exponent* is one of the items Lotka specifically sought for the data he examined. Some researchers applying Lotka's Law mistakenly thought that their data only fit the Law's prediction if the exponent was exactly 2 (see, e.g., Schorr, 1975a, 1975b). Although Lotka may have been interested in exponents of 2, in fact his own data for the *Chemical Abstracts* demonstrated an exponent of 1.88, not 2.

Lotka's Law is not intended to predict a particular author's productivity. Instead, it predicts aggregate behavior across a large number of authors. Simply stated, Lotka's Law explains that at a higher level of productivity, there are fewer authors. The exponent, n, predicts the relative number of authors at each productivity level, and the absolute number is a factor of the total number of authors under study. Based on Equation 2, with n fixed at 2, Lotka calculated that approximately 60.79% of authors would have only one publication.

If, as we speculate, bibliometric models are applicable to patterns of open source development, how will this knowledge benefit the open source community? Benefits may include:

1. Increased understanding of how the apparently chaotic process of open source development actually follows predictable patterns;
2. Explanation of the fact that some packages are widely used and heavily maintained while others languish, based on the probability that a particular package will generate sustained programmer interest;
3. Justification for the benefits of the open source model for predictably and regularly resulting in packages that are widely used and actively supported;
4. Reason for applying other models of human communication to open source development. For example, Roger's (1983) Diffusion of Innovations theory offers a comparable framework for adoption of technologies to Lotka's Law concerning authoring. We can imagine being able to understand the adoption of newly developed open source packages within the Diffusion of Innovations theory, much as we are seeking to understand the authoring process itself in this article.

Bibliometrics has helped to provide a logical framework for understanding human literary output. In this article, we

are seeking to apply well-known bibliographic concepts to open source development. Bibliometrics is not a panacea; it does not provide insight into the motivations of authors, the larger societal framework in which their work fits, or the actual meaning and content of publications. The research reported on in this article applied Lotka's law to LSM and Sourceforge metadata to gain a better understanding of open source development.

Research Design

Historically, the open source community has produced self-collected metadata regarding their projects. Two leading metadata repositories are the Linux Software Map (LSM) and Sourceforge, both of which were used for this research. The Linux Software Map, or LSM, has existed since the early 1990s. The LSM contains metadata describing Linux software (see the metadata template at http://www.ibiblio.org/pub /Linux /docs /linux-software-map /lsm-template). An example of an LSM record is found in Appendix A. LSM records are author-generated, and submitted to several Linux repositories including the University of North Carolina and the Center for the Public Domain's Ibiblio (http://www.ibiblio.org). LSM metadata in these repositories can be searched via customized search engines such as Linsearch (http://www.ibiblio.org/linsearch).

Sourceforge is a centralized resource for open source software development. It is supported by VA Linux, a vendor of software and tools for software development. Sourceforge provided the data for this study from their Web site, http://www.sourceforge.net. This site hosts thousands of different projects, and maintains listings of developers for each project, the project's source code and revision history, download statistics, documentation, and other items of interest. For this article, we examined data listing the number of registered developers for each software project hosted by Sourceforge.

To create a software project at Sourceforge, an individual must obtain a (freely available) Sourceforge login, and then request that the project be set up. Generally, only open source projects are supported by Sourceforge (because the system is explicitly designed to give access to the source code to anyone who wants it). The project, once created, is under the control of the individual who proposed it. The individual may then add any other registered Sourceforge usernames as developers for the project, and may assign particular roles and rights (e.g, while most developers are programmers, some might work with documentation or end-user support).

The data we obtained from the LSM collection were taken mainly from the Author: field of LSM records. The Author: field in LSM records gives us the ability to track the author of record for a software package. LSM metadata also include a list of maintainers, primary software distribution sites, date of update and other items. Because no historical revision information is kept, we must rely on data in the current LSM record, not previous versions. In some cases, ownership of a particular package has changed—this may or may not be represented in a record, depending on the preferences of whoever made the last update. We have also noted that the Author: field, while frequently the same as the Maintained-by: field, sometimes indicates a different person. Because of ambiguity in the end-user instructions for LSM record creation, it seems that sometimes the Author: contains the original author (regardless of who maintains the software currently), while other times it contains the person who created the LSM record (regardless of whether they are the head maintainer for the software). Overall, the pattern of use of the Author: entry indicates that it indicates the current author in charge of development for the software, which is our assumption for the remainder of this work. Not all LSM records are updated (Dempsey et al., 2001, found 6% of records they examined had not been updated since 1993 or 1994).

The data we obtained from Sourceforge consist of a list of developer ID numbers, followed by the number of projects on which the individual is listed as a developer, then the number of projects on which the individual is listed as an administrator. These data were provided for all 33,892 individuals registered to work on projects hosted by Sourceforge in July 2001. Administrator and Developer are two different roles available in the Sourceforge site, and their meaning is, to some extent, open to interpretation by each project. Essentially, Administrators can assign rights and add new Developers or Administrators; Developers can perform the tasks they are assigned, but may not exceed their assigned tasks (e.g., someone assigned to work with documentation might not be allowed to modify the source code held by the Sourceforge system).

Analysis of LSM Data

We examined all of the LSM records from 1993 (the earliest available date) through the third quarter of 2000 from a targeted subset of the Linux directory tree at Ibiblio (http://www.ibiblio.org), which houses one of the largest Linux archives and an immense quantity of other information. Ibiblio is one of the longest running public information services on the Internet, and an authoritative location for

TABLE 1. Distribution of unique authors across LSM entries.

# LSMs	# Authors	% of authors	Predicted with $n = 2$	Predicted %
1	2737	81.92%	2737	65%
2	391	11.70%	684	16%
3	130	3.89%	304	7%
4	44	1.32%	171	4%
5	15	0.45%	109	3%
6	4	0.12%	76	2%
7	3	0.09%	56	1%
8	2	0.06%	43	1%
9	4	0.12%	34	1%
10	2	0.06%	27	1%

TABLE 2. Prediction error for different values of n.

# LSMs	# Authors	Predicted	Exponent with $n = 2$	Prediction error
1	2737	2737	1.00	0.00
2	391	684	4.00	85995.56
3	130	304	9.00	30314.68
4	44	171	16.00	16144.88
5	15	109	25.00	8926.47
6	4	76	36.00	5188.00
7	3	56	49.00	2793.88
8	2	43	64.00	1661.84
9	4	34	81.00	887.45
10	2	27	100.00	643.64
			Total	152556.39

# LSMs	# Authors	Predicted	Exponent with $n = 3$	Prediction error
1	2737	2737	1.00	0.00
2	391	342	8.00	2388.77
3	130	101	27.00	819.66
4	44	43	64.00	1.52
5	15	22	125.00	47.55
6	4	13	216.00	75.19
7	3	8	343.00	24.80
8	2	5	512.00	11.19
9	4	4	729.00	0.06
10	2	3	1000.00	0.54
			Total	3369.28

# LSMs	# Authors	Predicted	Exponent with $n = 2.82$	Prediction error
1	2737	2737	1.00	0.00
2	391	388	7.06	11.64
3	130	124	22.16	41.79
4	44	55	49.87	118.52
5	15	29	93.56	203.17
6	4	17	156.45	182.09
7	3	11	241.64	69.33
8	2	8	352.14	33.32
9	4	6	490.87	2.48
10	2	4	660.69	4.59
			Total	666.92

information resources from Botany to Biblical Greek. The targeted subset consisted of six Linux subdirectories: apps, system, X11, utils, games, and devel. Within these subdirectories, all LSM files were examined, totaling 4,503 separate files. These records identified 3,341 unique author names. Due to ambiguity in some author's names and e-mail addresses, an unknown (but presumed small) number of the unique author names are actually duplicates.

T1 Table 1 presents the distribution of frequency for author count, and the predicted author count for Lotka's Law's special case where the exponent, n, is 2. Of the 3,341 unique author names, 2,737 (82%) are associated with one LSM record. Another 391 (12%) of authors are listed in two LSM records, and so forth. The number of authors who are recorded in 10 or more LSM records is quite small, including only nine authors ranging from 12 to 26 LSMs. Those records were eliminated from further analysis.

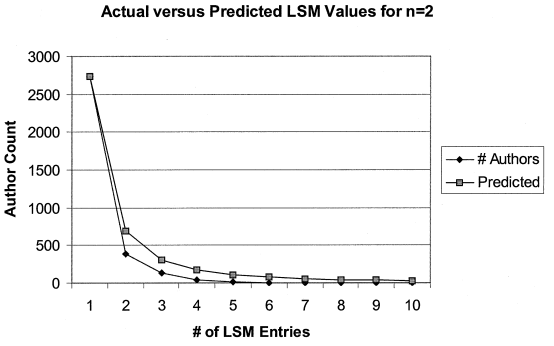

FIG. 1. Acutal versus predicted LSM values for $n = 2$.

Lotka suggested measuring the error in the prediction by squaring the errors. This method is supported by Pao (1985), who provides a modern infrastructure for performing analysis of Lotka's Law, which we generally follow here. By performing simple numerical computation, we can find the value of n that minimizes the error rate, as demonstrated in Table 2. T2

In Table 2, the first column corresponds to the value of i from Equation 1: the number of LSM entries. The second column is the number of authors with i entries in our data. The third column is the expected value for each LSM count value, E_i. The fourth column is the i^n, which is simply the denominator used to produce the value from column 3. The fifth column is the square of the prediction error, column 2 minus column 3. The TOTAL indicates the sum of squared error values, which we are seeking to minimize to find the optimum value for n. Table 2 presents values for $n = 2$, $n = 3$, and $n = 2.82$, which we found to be optimal (i.e., with the smallest aggregate error).

Figures 1 and 2 plot the actual versus predicted numbers F2 of publications for each LSM value from 1–10 for n values of 2 and 2.82. The improved fit is visually apparent for $n = 2.82$ (in fact, the predicted and actual lines can barely be distinguished). For $n = 2$, Lotka's special case for the inverse square law, we see that the LSM data drop off too rapidly to fit well. One interpretation for this is that, compared to data where $n = 2$ is a good fit, it is relatively easy to become an open source software developer (as measured

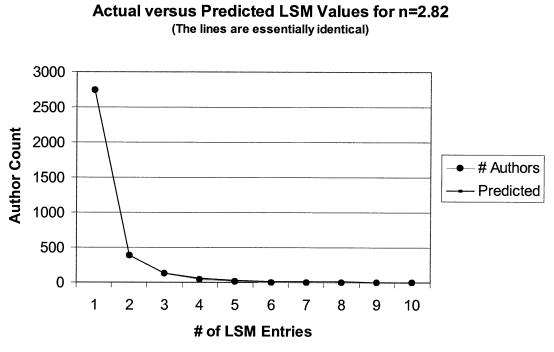

FIG. 2. Actual versus predicted LSM values for $n = 2.82$ (the lines are essentially identical).

TABLE 3. Descriptive statistics for sourceforge data.

	Developer count	Administrator count	Total count
Minimum	0	0	1
Maximum	9	21	23
Average	0.43	0.93	1.36
Median	0	1	1
St. Dev.	0.63	0.94	0.94

by authorship of an LSM record), compared to becoming a scholarly author. Another interpretation, which seems favored by authors such as Egghe and Rosseau (1996), is simply that different literatures or types of literatures have different values of n, and furthermore that equations different from Lotka's might be more applicable.

Analysis of Sourceforge Data

We examined records for all 33,892 individuals working on open source software development projects registered in Sourceforge (http://www.sourceforge.net) as of July 2001. Because Sourceforge has only been in operation since about 1999, these data are somewhat cleaner than the LSM data: there are fewer abandoned projects, and information about which developers are associated with projects is more likely to be correct. However, we did not attempt to verify how many individuals listed as developers for projects are actually actively involved, because project statistics were not available to us. Our goal is to achieve this type of analysis in a future study, when we will use data on actual changes to a Sourceforge-based project to apply other measures of author productivity.

Sourceforge hosts over 26,000 projects, with over 243,000 registered individuals. However, not all individuals are associated with a project (and some persons have more than one UID). Our data are exhaustive for all the UIDs associated with all projects. The raw data from Sourceforge T3 was in three columns, as illustrated in Table 3. The first column is a unique developer ID number for each person registered with Sourceforge. The second column is the number of projects the UID is listed as a Developer for, and the third is the number of projects the UID is listed as Administrator for (these different roles were discussed briefly above).

We performed an analysis similar to that described for the LSM data above to discover the best-fitting value of n for three different variations on the data: (1) total authorship (Developer plus Administrator); (2) Developer only; (3) Administrator only.

Although Developer and Administrator would seem to be descriptive, there is no fixed definition or role for these labels from Sourceforge. Given this limitation, we suggest that the total of all projects on which an individual is involved (the Developer plus Administrator columns) provides the most suitable approach to a bibliometric productivity analysis.

Total authorship ranged from individuals listed as Administrators or Developer on only one project (26,544 persons) to one person listed for 23 projects. Table 4 presents T4 descriptive statistics for the three variations on the data. Note that every project must have at least one Administrator or Developer.

In Table 5, we present the outcome of our analysis for the T5 total count. Based on the data we have, this seems the most

TABLE 4. Total sourceforge data, $n = 2.0$ and 2.55.

# Projects	# Authors	Exponent with $n = 2$	Expected	Error
1	26544	1	26544.00	0.00
2	4778	4	6636.00	3452164.00
3	1475	9	2949.33	2173658.78
4	563	16	1659.00	1201216.00
5	244	25	1061.76	668731.42
6	124	36	737.33	376177.78
7	66	49	541.71	226304.08
8	37	64	414.75	142695.06
9	20	81	327.70	94681.57
10	11	100	265.44	64739.71
11	8	121	219.37	44678.08
12	10	144	184.33	30392.11
13	3	169	157.07	23736.05
14	0	196	135.43	18340.90
15	1	225	117.97	13682.76
16	2	256	103.69	10340.35
17	1	289	91.85	8253.31
18	0	324	81.93	6711.86
19	1	361	73.53	5260.47
20	1	400	66.36	4271.93
21	1	441	60.19	3503.51
22	1	484	54.84	2899.07
23	1	529	50.18	2418.45
			Total	8574857.24

# Projects	# Authors	Exponent with $n = 2.55$	Expected	Error
1	26544	1.00	26544.00	0.00
2	4778	5.86	4532.52	60259.66
3	1475	16.47	1611.79	18710.20
4	563	34.30	773.95	44500.27
5	244	60.59	438.12	37682.49
6	124	96.45	275.22	22867.62
7	66	142.89	185.77	14343.91
8	37	200.85	132.16	9054.67
9	20	271.22	97.87	6063.68
10	11	354.81	74.81	4071.86
11	8	452.43	58.67	2567.43
12	10	564.82	47.00	1368.65
13	3	692.72	38.32	1247.41
14	0	836.81	31.72	1006.19
15	1	997.78	26.60	655.52
16	2	1176.27	22.57	422.97
17	1	1372.92	19.33	336.14
18	0	1588.35	16.71	279.28
19	1	1823.15	14.56	183.86
20	1	2077.91	12.77	138.64
21	1	2353.20	11.28	105.68
22	1	2649.58	10.02	81.33
23	1	2967.60	8.94	63.12
			Total	226010.56

TABLE 5. Sourceforge administrator data, $n = 2.0$ and 2.61.

# Projects	# Authors	Exponent with $n = 2$	Expected	Error
1	19008	1	19008.00	0.00
2	3336	4	4752.00	2005056.00
3	910	9	2112.00	1444804.00
4	330	16	1188.00	736164.00
5	119	25	760.32	411291.34
6	85	36	528.00	196249.00
7	32	49	387.92	126677.88
8	16	64	297.00	78961.00
9	8	81	234.67	51377.78
10	9	100	190.08	32789.97
11	2	121	157.09	24053.19
12	4	144	132.00	16384.00
13	1	169	112.47	12426.31
14	2	196	96.98	9021.12
15	0	225	84.48	7136.87
16	1	256	74.25	5365.56
17	1	289	65.77	4195.36
18	0	324	58.67	3441.78
19	1	361	52.65	2668.11
20	1	400	47.52	2164.11
21	1	441	43.10	1772.58
			Total	5171999.97

# Projects	# Authors	Exponent with $n = 2.61$	Expected	Error
1	19008	1.00	19008.00	0.00
2	3336	6.11	3113.49	49508.60
3	910	17.59	1080.56	29091.61
4	330	37.27	509.99	32395.63
5	119	66.73	284.86	27508.07
6	85	107.39	177.00	8463.13
7	32	160.59	118.37	7459.25
8	16	227.54	83.54	4561.06
9	8	309.44	61.43	2854.51
10	9	407.38	46.66	1418.21
11	2	522.44	36.38	1182.21
12	4	655.64	28.99	624.58
13	1	807.96	23.53	507.41
14	2	980.38	19.39	302.36
15	0	1173.81	16.19	262.23
16	1	1389.16	13.68	160.86
17	1	1627.32	11.68	114.07
18	0	1889.13	10.06	101.24
19	1	2175.44	8.74	59.87
20	1	2487.07	7.64	44.13
21	1	2824.83	6.73	32.82
			Total	166651.85

appropriate value to use for patterns of open source authorship, due to the lack of clarity how Developers and Administrators differ. The sum of squares error for the special case of $n = 2$ was 8,574,857. The lowest sum of squares error resulted with $n = 2.55$, with a value of 226,010. The data in this analysis included all cases, ranging from individuals on one project to an individual involved with 23 projects.

T6 In Table 6, we present the outcome of our analysis for Administrator only. In Lotka's data, he only examined the primary (first) author. For this reason, casual interpretation might lead us to believe this subset of the Sourceforge data is closest to Lotka's. However, this is not the case: although it seems likely that the creator of a project would also be listed as an Administrator (and, therefore, would correspond conceptually to the primary author), there is no reason why this must be the case. Furthermore, the Sourceforge data specifically allow more than one Administrator, which is not consistent with Lotka's use of only a single author from his data. For the Administrator data, the total error when $n = 2$ was 5,174,833. Rounded to two decimal places, the least error occurs with $n = 2.61$, with a total error of 166,715.

T7 Table 7 presents Developer data. The maximum project count was nine for this subset. With $n = 2$, the total sum of squared error values was 5,614,255. The best-case n was 3.56, for a sum of squared error values of only 2369.

The Sourceforge data yielded similar optimum values of n for the Total and Administrator (2.55 and 2.61) cases, but a quite different value for Developer (3.56). The higher value for Developer indicates a steeper drop-off than the other cases. We speculate that it may be easier to get involved with a single project than it is to publish a scholarly work (in the nonsoftware world) or to get involved as a project administrator (in Sourceforge projects). The data we examined are insufficient to assert or refute this interpretation.

Because the Total is the sum of the Administrator and Developer, we would expect some degree of agreement among the three different values for optimal n. This makes the value of 3.56 appear to be even more of a chimera. Our best guess is that the Developer status is used differently than Administrator status, or perhaps simply used for different types of projects—perhaps those with a greater total number of Developers and Administrators. All three values are higher than those typically found in other evaluations of Lotka's Law, although Pao (1985) confirms our opinion that considerable variation exists in data gathering, evaluation, and fitting. She wrote that Vlachy (1974) reported values for n ranging from 1.2 to over 3.5.

Goodness of Fit with Lotka's Law

Pao (1985) presents an evaluative framework for comparison of authorship data with Lotka's Law's predictions.

TABLE 6. Sourceforge developer data, $n = 3.56$.

# Projects	# Authors	Exponent with $n = 3.56$	Expected	Error
1	11582	1.00	11582.00	0.00
2	997	11.79	982.01	224.64
3	210	49.95	231.86	478.00
4	46	139.10	83.26	1388.50
5	32	307.85	37.62	31.62
6	7	589.14	19.66	160.25
7	6	1019.88	11.36	28.69
8	0	1640.59	7.06	49.84
9	2	2495.19	4.64	6.98
			Total	2368.52

TABLE 7. LSM Analysis with percentages for K-S test.

# LSMs	# Authors	# Authors %	Predicted	Predicted %	Exponent with n = 2.82	Prediction error	Prediction error %
1	2737	0.82	2737	0.81	1.00	0.00	0.000131
2	391	0.12	388	0.11	7.06	11.64	0.000007
3	130	0.04	124	0.04	22.16	41.79	0.000006
4	44	0.01	55	0.02	49.87	118.52	0.000009
5	15	0.00	29	0.01	93.56	203.17	0.000017
6	4	0.00	17	0.01	156.45	182.09	0.000016
7	3	0.00	11	0.00	241.64	69.33	0.000006
8	2	0.00	8	0.00	352.14	33.32	0.000003
9	4	0.00	6	0.00	490.87	2.48	0.000000
10	2	0.00	4	0.00	660.69	4.59	0.000000
SUM	3332	1.00	3379	1.00			0.000195
						Max	0.000131

Our analysis is approximately consistent with her recommendations. Pao suggests the Kolmolgorov-Smirnov (K-S) one-sample goodness of fit test (rather than the chi-square test) for evaluate the statistical significance of results. This test compares expected to actual values for the cumulative frequency distributions. Pao, like other researchers, deals with percentages of data rather than raw numbers. For the K-S test, this is a necessity, as working with percentages essentially normalizes the data. The question with the K-S test is whether the maximum deviation between expected and actual values exceeds a critical value.

To perform the K-S test, we must change our raw numbers to percentages. Table 8 presents a modification of Table 2, in which additional columns have been added. Column 3 presents the number of authors expressed as a percentage of the total number of authors (3,332). Column 5 presents the predicted number of authors expressed as a percentage of the total number of predicted authors (3,379). The last column contains the squared difference between the values for column 3 and 5.

for the K-S test, we are interested in the maximum percentage deviation, which in this case corresponds to the row where the LSM count is 1, 0.000131. This value is well below the critical value for the K-S test at an alpha value of 0.01, which is 1.63/sqrt(N) or 1.63/sqrt(3332) or 0.03. Based on the K-S test, we conclude that with $n = 2.82$, the data fit the expected values, and thus fail to reject the null hypothesis that the expected values and the actual values are different.

Table 9 present results for the K-S test for the Sourceforge data. The only subset of data that achieved significance at alpha = 0.01 is the Development only subset.

Our analysis shows that Pao's recommendation fails for a large N, because the critical value becomes unreasonably small when N is high. For our Total subset for the Sourceforge data, we had scores from 33890 individuals. To pass the test of significance, the maximum deviation from expected to actual percentages cannot exceed .885%. Yet, in absolute terms, a maximum error of 3.54%, as found for the Total subset of the Sourceforge data, seems to indicate a reasonable fit between our model with $n = 2.61$ and the actual data.

Perhaps more importantly, the K-S test is conservative in the opposite direction from the one appropriate for testing our hypothesis. For a test of Lotka's Law, we are interested in testing the hypothesis that the data fit the model. In other words, our null hypothesis is not that the data are the same, but that they are different. If the data are similar enough, we would like to say the data match the model. If this explanation is unclear, simply look at the table of critical K-S values in any statistics text: as the level of significance (alpha level) decreases, the critical value increases.

For example, the critical value for the K-S test for the Total subset of the Sourceforge data is 0.00885 for alpha = 0.01, and 0.006627 for alpha = 0.10. For our purposes, the higher alpha level gives a more permissive test (we are allowed to have larger errors, and the test is still significant). Perhaps at an alpha of 0.00000001, our data will be significant. This is the opposite of how most statistical tests work, because our model-fitting hypothesis is the opposite of what statistical tests are designed to evaluate.

A further problem with Pao's methodology is that the best significance on K-S does not correspond to the lowest squared errors (regardless of whether raw scores or percent-

TABLE 8. K-S outcomes.

	Max error	Critical
Total	0.03545459	0.00885425
Administrator	0.03937917	0.01055088
Developer	0.00699454	0.01436137

TABLE 9. K-S Outcomes

	Max error	Critical
Total	0.03545459	0.00885425
Administrator	0.03937917	0.01055088
Developer	0.00699454	0.01436137

ages are used). For our Total subset of the Sourceforge data, we found we could improve our maximum error to 0.0194 with an n of 2.7, but this increased our sum of squared error values from 225,866 for our best-case n of 2.55 to 530,409 for n of 2.7. In other words, in striving to achieve significance on the K-S test, we increased the error rate of our model.

The desire for a goodness of fit test for testing Lotka's law is clear. However, based on our findings and discussion above, we fail to see the appeal of the K-S test proposed by Pao. Strict adherence to Pao's methodology leads us to say that at the 0.01 alpha level, we reject the null hypothesis for the Total and Administrator subsets of the Sourceforge data. That is, we cannot say that expected values based on Lotka's Law and the actual experimental values from Sourceforge are the same. For the Development only subset and the LSM data, we fail to reject the null hypothesis: the data were not at significant disagreement with the model.

Conclusions: Lotka and the Real World

We found n values of 2.82 for the LSM data and 2.55, 2.61, and 3.56 for the Sourceforge Total, Administrator, and Developer data subsets. Analysis of the pattern of actual values compared to values predicted by Equation 1 indicated a good fit between our data and values predicted by Lotka's Law. Pao's (1985) criteria for statistical significance of the fit between our data and values that was predicted by Lotka's Law was met only for the LSM data and for the Developer Sourceforge subset. However, we argued above that the Kolmolgorov-Smirnov test suggested by Pao is problematic in this context.

Lotka only looked at primary authorship. For the LSM records, we eliminated all but the first listed author—but fewer than 5% had more than one author listed. For the Sourceforge data, we took a substantially different approach from Lotka (although it is comparable to the approaches of others seeking to apply his Law). We treated all participating developers as authors, and in fact did not have any data about the "primary" author (i.e., the person who first proposed and created the project).

Studies subsequent to Lotka's also used different types of data (Pao, 1985). Based on these other studies and the literature cited in this article, we believe our work represents a fair and accurate assessment of the fit between Lotka's Law and authorship patterns in open source software development. We believe that the analysis presented here demonstrates that the patterns of productivity in the open source software development data are comparable to patterns of productivity found in studies of scholarly or scientific productivity which applied Lotka's Law. These data and analysis provide substance to the notion that software development is a form of authorship comparable to scientific and scholarly writing. Like these other forms of writing, not all authors will create multiple works. For available creative outlets for work, many authors will contribute mostly to a single outlet.

The predictive power of Lotka's Law is limited, but does have a practical application. Because it operates at an aggregate level, we cannot claim any predictive power over such factors as the productivity of an individual software developer, nor can we assess the viability or success of a particular software project. The prediction we can make is: given some number of developers working on open source software development, there will be a distribution of effort on the software development projects that approximately follows an inverse square distribution (in our case, the most compelling data had exponents ranging from 2.55 to 2.82). The actual value of n should not be of primary interest (certainly not to two decimal places). Rather, the approximate range of n gives an idea as to the likely productivity to be gained (in terms of additional projects) by attracting new authors (programmers).

Understanding the similarity between print-based activities and "new" electronic authoring activities, especially programming, has practical implications for expectations by software project managers and companies seeking to engage in open source software development, as well as for the programmers themselves. Employers, universities or others with substantial influence over personnel may decide to allocate numbers of software developers to open source projects based on these findings, in anticipation that the number of projects each developer will engage in is, at the aggregate level, predictable. For an individual programmer, the odds are that he or she will only engage in one project, with relatively few programmers engaging in three or more. Expectations of employers considering investment of resources in open source development should be aware of this likelihood, and not expect every programmer to become significantly engaged in many projects (nor for every project to attract many programmers).

The nature of authorship, as applied here, is necessarily ambiguous. Further study is needed to identify the different patterns in different roles in software development, which are conflated in the data we analyzed. How do patterns of authorship differ between the programmers, the designers, the documentation specialists, etc.?

The open source research team at UNC is engaged in other studies, or plans to engage in other studies that will examine additional data and metadata on software productivity, and seeks to provide better understanding of open source software development. Data we are interested in working with will address productivity within projects: how can we use measures of software productivity, such as lines of code or developer editing sessions, to further our interpretations? We also desire to make better use of the electronic and Web-based data sources for assessing authorship patterns. This is an important area for bibliometrics (see Cronin, 2001), but one that bypasses many of the traditional barriers to publication.

Our experience with the data presented here is that bibliometrics provides powerful tools for understanding author productivity. With only minor modifications to the procedures or equations from bibliometrics, we have been able to

apply the techniques of scientific and scholarly publications to open source software development. Open source software development is an important phenomenon for the future of computing. We encourage others to gather data and perform analysis to maximize the positive impact that it may have on society. In today's world, where people can create digital artifacts of many types (programs, documentation, e-mail, Web pages, etc.), we think that reconsideration of bibliometric techniques for modern communication modalities is of key interest to our discipline.

Acknowledgments

The authors wish to thank Dr. Bert Dempsey for his work on data collection, Dr. Robert Losee for his comments of our work, and Sourceforge for providing data.

APPENDIX A: LSM RECORD EXAMPLE

```
Begin3
Title:          kdeadmin
Version:        1.1
Entered-date:   6/2/1999
                Admin Tools Source Package for the K Desktop
Description:    Environment
Keywords:       KDE Desktop Environment, KDE, GUI, Qt, X11,
Author:         The KDE Project Team
Maintainer:     The KDE Core Team
Primary-site:   http://sunsite.unc.edu/pub/Linux/X11/kde/1.1/tgz/source/k
Original-site:  ftp://ftp.kde.org/pub/kde/stable/1.1/distribution/tar/ge
Alternate-site: ftp://ftp.de.kde.org/pub/kde/stable/1.1/distribution/tar
Platform: Unix with X11, tested with Linux (Intel, Alpha), FreeBSD
Copying-policy: GPL
```

References

Agree, P. (1994). Net presence. Computer-Mediated Communication Magazine, 1(4), 6.

Auerbach, Felix. (1910). Geschichtstafeln der Physik. Leipzig: J.A. Barth.

Costalles, B., & Allman, E. (1997). Sendmail (2nd ed.). Sebastopol, CA: O'Reilly and Associates.

Cronin, B. (2001). Bibliometrics and beyond: Some thoughts on Web-based citation analysis. Journal of Information Science, 27(1), 1–7.

Dempsey, B.J., Weiss, D., Jones, P., & Greenberg, J. (1999). A quantitative profile of a community of open source Linux developers. UNC SILS Technical Report TR-1999-05. Available: http://ils.unc.edu/ils/research/reports/TR-1999-05.pdf.

Dempsey, B., Weiss, D., Jones, P., & Greenberg, J. (in press). Who is an open source developer? A quantitative profile of a community of open source Linux developers. Communications of the ACM. AQ: 7

DiBona, C., Ockman, S., & Stone, M. (1999). Open sources: Voices from the open source revolution. Sebastopol, CA: O'Reilly and Associates.

Dublin Core Metadata Element Set, Version 1.1: Reference Description (1999). http://dublincore.org/documents/1999/07/02/dces/.

Egghe, L., & Rosseau, R. (1996). Modelling [sic] multi-relational data with special attention to the average number of collaborators as a variable in informetric distributions. Information Processing & Management, 32(5), 563–571.

Foucault, M. (1977). What is an author? In D. Bouchard (Ed.), Language, counter-memory, practice (pp. 113–138). Ithaca, NY: Cornell University Press.

Functional Requirements for Bibliographic Records [final report]. (1997). Recommended by the IFLA Study Group on the Functional Requirements for Bibliographic Records; International Federation of Library Associations and Institutions, IFLA Universal Bibliographic Control and International MARC Programme. Frankfurt Am Main: IFLA UBCIM.

Mockus, A., Fielding, R.T., & Herbsleb, J. (2000). A case study of open source software development: The Apache server. ICSE 2000, Proceedings of the 22nd International Conference on Software Engineering (pp. 263–272), June 4–11, 2000, Limerick Ireland.

Narin, F., & Moll, J.K. (1977). Bibliometrics. In M.E. Williams (Ed.), Annual review of information science and technology, 12, 35–58. AQ: 8

Nichols, P.T. (1986). Empirical validation of Lotka's Law. Information Processing & Management 22(5), 417–419.

Pao, M.L. (1985). Lotka's Law: A testing procedure. Information Processing & Management 21(4), 305–320.

Pritchard, A. (1969). Statistical bibliography or bibliometrics. Journal of Documentation, 25(4), 348–349.

Raymond, E.S. (2001). The cathedral and the bazaar, 2nd ed. Sebastopol, CA: O'Reilly and Associates.

Rogers, E.M. (1983). Diffusion of innovations. New York: Free Press.

Rose, M. (1993). Authors and owners: The invention of copyright. Cambridge, MA: Harvard University Press. AQ: 9

Schorr, A.E. (1975a). Lotka's Law and map librarianship. Journal of the American Society for Information Science, 189–190. AQ: 10

Schorr, A.E. (1975b). Lotka's Law and the history of legal medicine. Research Librarianship, 30, 205–209.

Sengupta, I.N. (1992). Bibliometrics, informetrics, scientometrics and librametrics: An overview. Libri, 42(2), 75–98.

Vlachy, J. (1974). Distribution patterns in creative communities. Toronto: World Congress of Sociology.

White, H.D., & McCain, K.W. (1989). Bibliometrics. In M.E. Williams (Ed.), Annual review of information science and technology, 24, 119–186. AQ: 11

AQ1: not in ref list; please place in list or delete from text

AQ2: not in ref list; please place in list or delete from text

AQ3: not in ref list; please place in list or delete from text

AQ4: not in ref list; please place in list or delete from text

AQ5: not in ref list; please place in list or delete from text

AQ6: not in ref list; please place in list or delete from text

AQ7: please update (accepted 2000?)

AQ8: is this a journal or a book? If a book, please provide publisher & city of publication; if a journal, is this a special issue? Why an edited by?

AQ9: please cite in text or delete from list

AQ10: please provide volume number

AQ11: is this a journal or a book? If a book, please provide publisher & city of publication; if a journal, is this a special issue? Why an edited by?

AQ12: there was no Table 9 with manuscript; please supply a copy or delete from text