Analyzing Programming Languages Energy Consumption: An Empirical Study*

Extended Abstract†

Stefanos Georgiou‡
Athens University of Economics and Business
Athens, Greece
sgeorgiou@aueb.gr

Diomidis Spinellis§
Athens University of Economics and Business
Athens, Greece
dds@aueb.gr

Maria Kechagia¶
Institute for Clarity in Documentation
Dublin, Ohio
dds@aueb.gr

ABSTRACT

Motivation:
Goal:
Method:
Results:

CCS CONCEPTS

• **Hardware → Power estimation and optimization**; • **Software and its engineering** → *Software libraries and repositories*; *Software design tradeoffs*;

KEYWORDS

GreenIT, Energy Efficiency, Energy Optimization, Programming Languages

ACM Reference format:
Stefanos Georgiou, Diomidis Spinellis, and Maria Kechagia. 2017. Analyzing Programming Languages Energy Consumption: An Empirical Study. In *Proceedings of Panhellenic Conference on Informatics, Larrisa, Greece, September 2017 (PCI '17),* 6 pages.
https://doi.org/10.475/123_4

1 INTRODUCTION

The *proceedings* are the records of a conference.[1] ACM seeks to give these conference by-products a uniform, high-quality appearance. To do this, ACM has some rigid requirements for the format of the proceedings documents: there is a specified format (balanced double columns), a specified set of fonts (Arial or Helvetica and Times Roman) in certain specified sizes, a specified live area, centered on the page, specified size of margins, specified column width and gutter size.

*Produces the permission block, and copyright information
†The full version of the author's guide is available as `acmart.pdf` document
‡Dr. Trovato insisted his name be first.
§The secretary disavows any knowledge of this author's actions.
¶The secretary disavows any knowledge of this author's actions.
[1]This is a footnote

PCI '17, September 2017, Larrisa, Greece

ACM ISBN 123-4567-24-567/08/06…$15.00
https://doi.org/10.475/123_4

2 RELATED WORK

3 EXPERIMENT SETUP

In this Section, we describe the experimental approach we used in order to perform our research and retrieve our measurements. Initially, we provide information about our dataset and the ways we decided to select our tasks and refine it. Furthermore, we explain the setup up of our experimental platform, the additional hardware tools, and the software tools used to conduct this research. Moreover, we argue on the selected tasks and programming languages we used and the way re refined our dataset.

3.1 Dataset

In the context of study, we used Rosetta Code,[2] a publicly available programming chromatography site that offers 851 tasks, 230 draft tasks, and a collection of 658 different programming languages. In general, not all of (and cannot) tasks are implemented in all languages. We found and downloaded a git hub repository[3] which contains all the currently implemented tasks found on Rosetta Code website.

For selecting the highly used programming languages, we made use of tiobe,[4] a software quality company. By making use of a formula[5], 25 of the highest ranked search engines (according to Alexa),[6] and a number of requirements enlisted for programming languages, tiobe provides a search query for index rating of the most popular programming languages around the web for each month. Initially, we decided of choosing the top 15 programming languages as enlisted for June 2017. From the current list, we excluded programming languages such as Delphi and Assembly. In contrast, we included Rust in our dataset which a memory safe programming language and is gaining popularity in the web. Therefore, we ended up with 14 programming languages as illustrated in Table 2.

In terms of selecting tasks, we developed a shell script (more details in Subsection 3.2.2) to identify which of the 851 tasks offers the most implementations for the programming languages of our selection. To refine further our dataset we used the following steps:

- We excluded implemented tasks that are using external libraries. Since we would like to exploit the energy dissipation

[2]http://rosettacode.org/wiki/Rosetta_Code
[3]https://github.com/acmeism/RosettaCodeData
[4]https://www.tiobe.com/tiobe-index/
[5]https://www.tiobe.com/tiobe-index/programming-languages-definition/
[6]http://www.alexa.com/

of simple every day tasks, we decided to exclude form our dataset the use of tasks requiring external libraries.
- Some of the tasks offers more that one implementation for the same programming languages. Thus, we had to drive manually through each directory and remove them until we have only one that is consistent with the other implementation. For example, palindrome in recursion we removed the iterative implementations.
- The Java file's names were different from the public class names which results to compilation error if not change accordingly.
- Some of the implementations didn't have main classes, nor the same data with other tasks. Therefore, we change the source code to offer consistency.
- Some of the tasks are relatively small and may finish faster than 1 second which makes it impossible for our power analyzer to capture those results. Therefore, we added all the selected tasks in a loop to iterate them a million of times.

3.2 Hardware and Software components

3.2.1 Hardware Components. The physical tools composed mainly from a portable personal computer and a real-time electricity usage monitoring tool. Our system's specifications are depicted in Table 1. The real-time power usage tools we used is the Watts Up Pro (WUP).[7]

In general, there are two venues for retrieving energy consumption from a computer-system: on one hand, by indirect energy measurements through estimation models or performance counters, core component of software monitoring tools, on the other hand, via direct measurement, hardware power analyzers and sensors. However, each approach has it own pitfalls such as: coarse-grained measurements for the whole systems energy consumption and low sampling rate for direct measurements case and inaccuracy, lack of interoperability, and additional system overhead while indirect measurements. Therefore, in our research we decided to retrieve our energy consumption measurements using direct approach such as WUP since our tasks are relatively short in terms of source-code lines.

In regards to WUP, it offers accuracy of ±1.5% and as maximum sampling rate of 1 second. In order to retrieve power-related measurements from the WUP we used a software available in a Git repository.[8] This software helped us retrieve measurements such as timestamps, watts, volts, amps, etc. through a mini USB interface after we integrated its code in our script that runs all the tasks.

3.2.2 Software Components. To extract data, manage, and use our Rosetta Code Repository, we developed a number of shell scripts as enlisted below and are publicly available on our Git repository.[9]

- **script.cleanAll**, removes the current instance of Tasks in the current working directory and copy the new one found from in the parent directory.
- **script.fromUpperToLower**, changes the current instance of Tasks directory tree's letters from upper to lower case to offer consistency for our scripts.

Table 1: System hardware and software specifications

	Description
Hardware	HP EliteBook 840 G3, Intel Core i7-6500U (2 physical cores of 2.5 GHz), 8 GB DDR4 memory, 256 GB SSD hard disk
Operating System	Fedora 25 kernel version 4.11.5-200
Software	WUP software (retrieving measurements from the device), bash script, Java, R, feedGnuplot

- **script.findCommonTasksInLanguages**, provides a list of tasks with the amount of programming language implementations.
- **script.createNewDataSet**, filters the Rosetta Code current dataset and removes programming languages and tasks not added as command line arguments.
- **script.compileTasks**, compiles all tasks found under the Tasks' directory and produces error reports in a task fails to compile.
- **script.executeTasks**, executes all the tasks' implementations found in under Tasks directory.
- **script.plotGraphs**, after retrieving our data we use this script to plot our graphs.

Note that most of the scripts offer the *–help* option that shows a list of available command line arguments in order use our scripts. Moreover, users are suggested not to change the folders name or locations since most of our scripts are making use of them.

For plotting our graphs we used FeedGnuplot,[10] an open-source general purpose pipe-oriented plotting tool. To use our script for plotting graphs, someone has to provide a columned file with energy related measurements.

3.3 Retrieving Energy Measurements

As an initial step for our experiment, we shut down background process, as suggested by Hindle[...], found in modern OSs such as disk defragmentation, virus scanning, CRON jobs, automatic updates, disk indexing, document indexing, RSS feed updates, etc. to shorten some noise in our measurements. In addition, we set the network connection to flight mode and also reduced the brightness of the screen. By making the following steps we reduced our platform's idle power usage from 8,6 watts to 5.8 watts. We estimated that after an OS (Operating System) is launched it's necessary to wait for a short time to reach a *stable condition*.

After reaching *stable condition*, we launched our main script *i.e.,* **script.executeTasks**, that executes all the tasks implemented in different programming languages. While doing so, it retrieves power consumption and run-time performance measurements from WUP and through command time[11] and stores them in timestamped directories which we are using later to plot our results. Between each executing of a task, we added a sleep[12] period of three minutes. The time gap purpose is to ensure our experimental platform

[7] https://www.wattsupmeters.com/secure/products.php?pn=0
[8] https://github.com/pyrovski/watts-up
[9] https://github.com/stefanos1316/Rosetta-Code-Research
[10] http://search.cpan.org/ dkogan/feedgnuplot-1.44/bin/feedgnuplot
[11] https://linux.die.net/man/1/time
[12] http://man7.org/linux/man-pages/man3/sleep.3.html

Table 2: Programming Languages, Compilers and Interpreters

Programming Languages	Compilers Interpreters
C, C++	gcc version 6.3.1 20161221- (Red Hat 6.3.1-1) (GCC)
C#	mono version 4.4.2.0 (mics)[a]
Go	go version go1.7.5 linux/amd64
Java	javac version 1.8.0_131
JavaScript	node version 6.10.3
Perl	perl version 5.24.1
Php	php version 7.0.19
Python	python version 2.7.13
R	Rscript version 3.3.3
Ruby	ruby version 2.3.3p222
Rust	rustc version 1.18.0
Swift	swift version 3.0.2[b]
VB.NET	vbnc version 0.0.0.5943[c]

[a] https://www.codetuts.tech/compile-c-sharp-command-line/
[b] https://github.com/FedoraSwift/fedora-swift2/releases/tag/v0.0.2
[c] http://www.mono-project.com/docs/about-mono/languages/visualbasic/

reached a stable condition,*e.g.,* the CPU is cooled down and the fan is no longer consuming more power, to avoid unnecessary noise in our measurements.

4 RESULTS AND DISCUSSION

Graphs for programming languages using compilers and interpreters

5 THREATS OF VALIDITY

6 CONCLUSION AND FUTURE WORK

Two Venues, Academic and Industrial. She under the curtain and shed light... Compare PL in different CPU architectures such as ARM and AMD. More tasks and different configuration and optimizaiton flags. Collect resource usage and system calls to identify relationship among them.

Future plans, micro-services proposition of tasks or programming languages after identifying the reasons

6.1 Math Equations

You may want to display math equations in three distinct styles: inline, numbered or non-numbered display. Each of the three are discussed in the next sections.

6.1.1 Inline (In-text) Equations. A formula that appears in the running text is called an inline or in-text formula. It is produced by the **math** environment, which can be invoked with the usual `\begin . . . \end` construction or with the short form `$ . . . $`. You can use any of the symbols and structures, from α to ω, available in LaTeX [26]; this section will simply show a few examples of in-text equations in context. Notice how this equation: $\lim_{n\to\infty} x = 0$, set here in in-line math style, looks slightly different when set in display style. (See next section).

6.1.2 Display Equations. A numbered display equation—one set off by vertical space from the text and centered horizontally—is produced by the **equation** environment. An unnumbered display equation is produced by the **displaymath** environment.

Again, in either environment, you can use any of the symbols and structures available in LaTeX; this section will just give a couple of examples of display equations in context. First, consider the equation, shown as an inline equation above:

$$\lim_{n\to\infty} x = 0 \tag{1}$$

Notice how it is formatted somewhat differently in the **displaymath** environment. Now, we'll enter an unnumbered equation:

$$\sum_{i=0}^{\infty} x + 1$$

and follow it with another numbered equation:

$$\sum_{i=0}^{\infty} x_i = \int_0^{\pi+2} f \tag{2}$$

just to demonstrate LaTeX's able handling of numbering.

6.2 Citations

Citations to articles [6–8, 19], conference proceedings [8] or maybe books [26, 34] listed in the Bibliography section of your article will occur throughout the text of your article. You should use BibTeX to automatically produce this bibliography; you simply need to insert one of several citation commands with a key of the item cited in the proper location in the `.tex` file [26]. The key is a short reference you invent to uniquely identify each work; in this sample document, the key is the first author's surname and a word from the title. This identifying key is included with each item in the `.bib` file for your article.

The details of the construction of the `.bib` file are beyond the scope of this sample document, but more information can be found in the *Author's Guide*, and exhaustive details in the *LaTeX User's Guide* by Lamport [26].

This article shows only the plainest form of the citation command, using `\cite`.

Some examples. A paginated journal article [2], an enumerated journal article [11], a reference to an entire issue [10], a monograph (whole book) [25], a monograph/whole book in a series (see 2a in spec. document) [18], a divisible-book such as an anthology or compilation [13] followed by the same example, however we only output the series if the volume number is given [14] (so Editor00a's series should NOT be present since it has no vol. no.), a chapter in a divisible book [37], a chapter in a divisible book in a series [12], a multi-volume work as book [24], an article in a proceedings (of a conference, symposium, workshop for example) (paginated proceedings article) [4], a proceedings article with all possible elements [36], an example of an enumerated proceedings article [16], an informally published work [17], a doctoral dissertation [9], a master's thesis: [5], an online document / world wide web resource [1, 30, 38], a video game (Case 1) [29] and (Case 2) [28] and [27] and (Case 3) a patent [35], work accepted for publication [31], 'YYYYb'-test for prolific author [32] and [33]. Other cites might contain

Table 3: Frequency of Special Characters

Non-English or Math	Frequency	Comments
Ø	1 in 1,000	For Swedish names
π	1 in 5	Common in math
\$	4 in 5	Used in business
Ψ_1^2	1 in 40,000	Unexplained usage

Figure 1: A sample black and white graphic.

'duplicate' DOI and URLs (some SIAM articles) [23]. Boris / Barbara Beeton: multi-volume works as books [21] and [20].

A couple of citations with DOIs: [22, 23].

Online citations: [38–40].

6.3 Tables

Because tables cannot be split across pages, the best placement for them is typically the top of the page nearest their initial cite. To ensure this proper "floating" placement of tables, use the environment **table** to enclose the table's contents and the table caption. The contents of the table itself must go in the **tabular** environment, to be aligned properly in rows and columns, with the desired horizontal and vertical rules. Again, detailed instructions on **tabular** material are found in the *LaTeX User's Guide*.

Immediately following this sentence is the point at which Table 3 is included in the input file; compare the placement of the table here with the table in the printed output of this document.

To set a wider table, which takes up the whole width of the page's live area, use the environment **table*** to enclose the table's contents and the table caption. As with a single-column table, this wide table will "float" to a location deemed more desirable. Immediately following this sentence is the point at which Table 4 is included in the input file; again, it is instructive to compare the placement of the table here with the table in the printed output of this document.

It is strongly recommended to use the package booktabs [15] and follow its main principles of typography with respect to tables:

(1) Never, ever use vertical rules.
(2) Never use double rules.

It is also a good idea not to overuse horizontal rules.

6.4 Figures

Like tables, figures cannot be split across pages; the best placement for them is typically the top or the bottom of the page nearest their initial cite. To ensure this proper "floating" placement of figures, use the environment **figure** to enclose the figure and its caption.

This sample document contains examples of `.eps` files to be displayable with LaTeX. If you work with pdfLaTeX, use files in the `.pdf` format. Note that most modern TeX systems will convert `.eps` to `.pdf` for you on the fly. More details on each of these are found in the *Author's Guide*.

Figure 2: A sample black and white graphic that has been resized with the `includegraphics` command.

As was the case with tables, you may want a figure that spans two columns. To do this, and still to ensure proper "floating" placement of tables, use the environment **figure*** to enclose the figure and its caption. And don't forget to end the environment with **figure***, not **figure**!

6.5 Theorem-like Constructs

Other common constructs that may occur in your article are the forms for logical constructs like theorems, axioms, corollaries and proofs. ACM uses two types of these constructs: theorem-like and definition-like.

Here is a theorem:

Theorem 6.1. *Let f be continuous on $[a, b]$. If G is an antiderivative for f on $[a, b]$, then*

$$\int_a^b f(t)\,dt = G(b) - G(a).$$

Here is a definition:

The pre-defined theorem-like constructs are **theorem**, **conjecture**, **proposition**, **lemma** and **corollary**. The pre-defined definition-like constructs are **example** and **definition**. You can add your own constructs using the *amsthm* interface [3]. The styles used in the `\theoremstyle` command are **acmplain** and **acmdefinition**.

Another construct is **proof**, for example,

Proof. Suppose on the contrary there exists a real number L such that

$$\lim_{x\to\infty} \frac{f(x)}{g(x)} = L.$$

Then

$$l = \lim_{x\to c} f(x) = \lim_{x\to c}\left[gx \cdot \frac{f(x)}{g(x)}\right] = \lim_{x\to c} g(x) \cdot \lim_{x\to c} \frac{f(x)}{g(x)} = 0 \cdot L = 0,$$

which contradicts our assumption that $l \neq 0$. □

7 CONCLUSIONS

This paragraph will end the body of this sample document. Remember that you might still have Acknowledgments or Appendices; brief samples of these follow. There is still the Bibliography to deal with; and we will make a disclaimer about that here: with the exception of the reference to the LaTeX book, the citations in this paper are to articles which have nothing to do with the present subject and are used as examples only.

Table 4: Some Typical Commands

Command	A Number	Comments
`\author`	100	Author
`\table`	300	For tables
`\table*`	400	For wider tables

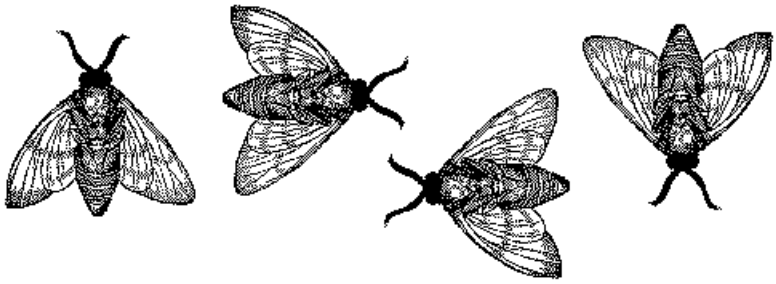

Figure 3: A sample black and white graphic that needs to span two columns of text.

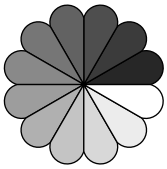

Figure 4: A sample black and white graphic that has been resized with the `includegraphics` command.

A HEADINGS IN APPENDICES

The rules about hierarchical headings discussed above for the body of the article are different in the appendices. In the **appendix** environment, the command **section** is used to indicate the start of each Appendix, with alphabetic order designation (i.e., the first is A, the second B, etc.) and a title (if you include one). So, if you need hierarchical structure *within* an Appendix, start with **subsection** as the highest level. Here is an outline of the body of this document in Appendix-appropriate form:

A.1 Introduction

A.2 The Body of the Paper

A.2.1 Type Changes and Special Characters.

A.2.2 Math Equations.

Inline (In-text) Equations.

Display Equations.

A.2.3 Citations.

A.2.4 Tables.

A.2.5 Figures.

A.2.6 Theorem-like Constructs.

A Caveat for the TEX Expert.

A.3 Conclusions

A.4 References

Generated by bibtex from your `.bib` file. Run latex, then bibtex, then latex twice (to resolve references) to create the `.bbl` file. Insert that `.bbl` file into the `.tex` source file and comment out the command `\thebibliography`.

B MORE HELP FOR THE HARDY

Of course, reading the source code is always useful. The file `acmart.pdf` contains both the user guide and the commented code.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Yuhua Li for providing the matlab code of the *BEPS* method.

The authors would also like to thank the anonymous referees for their valuable comments and helpful suggestions. The work is supported by the National Natural Science Foundation of China under Grant No.: 61273304 and Young Scientsts' Support Program (http://www.nnsf.cn/youngscientsts).

REFERENCES

[1] Rafal Ablamowicz and Bertfried Fauser. 2007. CLIFFORD: a Maple 11 Package for Clifford Algebra Computations, version 11. (2007). Retrieved February 28, 2008 from http://math.tntech.edu/rafal/cliff11/index.html

[2] Patricia S. Abril and Robert Plant. 2007. The patent holder's dilemma: Buy, sell, or troll? *Commun. ACM* 50, 1 (Jan. 2007), 36–44. https://doi.org/10.1145/1188913.1188915

[3] American Mathematical Society 2015. *Using the amsthm Package.* American Mathematical Society. http://www.ctan.org/pkg/amsthm.

[4] Sten Andler. 1979. Predicate Path expressions. In *Proceedings of the 6th. ACM SIGACT-SIGPLAN symposium on Principles of Programming Languages (POPL '79).* ACM Press, New York, NY, 226–236. https://doi.org/10.1145/567752.567774

[5] David A. Anisi. 2003. *Optimal Motion Control of a Ground Vehicle.* Master's thesis. Royal Institute of Technology (KTH), Stockholm, Sweden.

[6] Mic Bowman, Saumya K. Debray, and Larry L. Peterson. 1993. Reasoning About Naming Systems. *ACM Trans. Program. Lang. Syst.* 15, 5 (November 1993), 795–825. https://doi.org/10.1145/161468.161471

[7] Johannes Braams. 1991. Babel, a Multilingual Style-Option System for Use with LaTeX's Standard Document Styles. *TUGboat* 12, 2 (June 1991), 291–301.

[8] Malcolm Clark. 1991. Post Congress Tristesse. In *TeX90 Conference Proceedings*. TeX Users Group, 84–89.

[9] Kenneth L. Clarkson. 1985. *Algorithms for Closest-Point Problems (Computational Geometry)*. Ph.D. Dissertation. Stanford University, Palo Alto, CA. UMI Order Number: AAT 8506171.

[10] Jacques Cohen (Ed.). 1996. Special issue: Digital Libraries. *Commun. ACM* 39, 11 (Nov. 1996).

[11] Sarah Cohen, Werner Nutt, and Yehoshua Sagic. 2007. Deciding equivalances among conjunctive aggregate queries. *J. ACM* 54, 2, Article 5 (April 2007), 50 pages. https://doi.org/10.1145/1219092.1219093

[12] Bruce P. Douglass, David Harel, and Mark B. Trakhtenbrot. 1998. Statecarts in use: structured analysis and object-orientation. In *Lectures on Embedded Systems*, Grzegorz Rozenberg and Frits W. Vaandrager (Eds.). Lecture Notes in Computer Science, Vol. 1494. Springer-Verlag, London, 368–394. https://doi.org/10.1007/3-540-65193-4_29

[13] Ian Editor (Ed.). 2007. *The title of book one* (1st. ed.). The name of the series one, Vol. 9. University of Chicago Press, Chicago. https://doi.org/10.1007/3-540-09237-4

[14] Ian Editor (Ed.). 2008. *The title of book two* (2nd. ed.). University of Chicago Press, Chicago, Chapter 100. https://doi.org/10.1007/3-540-09237-4

[15] Simon Fear. 2005. *Publication quality tables in LaTeX*. http://www.ctan.org/pkg/booktabs.

[16] Matthew Van Gundy, Davide Balzarotti, and Giovanni Vigna. 2007. Catch me, if you can: Evading network signatures with web-based polymorphic worms. In *Proceedings of the first USENIX workshop on Offensive Technologies (WOOT '07)*. USENIX Association, Berkley, CA, Article 7, 9 pages.

[17] David Harel. 1978. *LOGICS of Programs: AXIOMATICS and DESCRIPTIVE POWER*. MIT Research Lab Technical Report TR-200. Massachusetts Institute of Technology, Cambridge, MA.

[18] David Harel. 1979. *First-Order Dynamic Logic*. Lecture Notes in Computer Science, Vol. 68. Springer-Verlag, New York, NY. https://doi.org/10.1007/3-540-09237-4

[19] Maurice Herlihy. 1993. A Methodology for Implementing Highly Concurrent Data Objects. *ACM Trans. Program. Lang. Syst.* 15, 5 (November 1993), 745–770. https://doi.org/10.1145/161468.161469

[20] Lars Hörmander. 1985. *The analysis of linear partial differential operators. III*. Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], Vol. 275. Springer-Verlag, Berlin, Germany. viii+525 pages. Pseudodifferential operators.

[21] Lars Hörmander. 1985. *The analysis of linear partial differential operators. IV*. Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], Vol. 275. Springer-Verlag, Berlin, Germany. vii+352 pages. Fourier integral operators.

[22] IEEE 2004. IEEE TCSC Executive Committee. In *Proceedings of the IEEE International Conference on Web Services (ICWS '04)*. IEEE Computer Society, Washington, DC, USA, 21–22. https://doi.org/10.1109/ICWS.2004.64

[23] Markus Kirschmer and John Voight. 2010. Algorithmic Enumeration of Ideal Classes for Quaternion Orders. *SIAM J. Comput.* 39, 5 (Jan. 2010), 1714–1747. https://doi.org/10.1137/080734467

[24] Donald E. Knuth. 1997. *The Art of Computer Programming, Vol. 1: Fundamental Algorithms (3rd. ed.)*. Addison Wesley Longman Publishing Co., Inc.

[25] David Kosiur. 2001. *Understanding Policy-Based Networking* (2nd. ed.). Wiley, New York, NY.

[26] Leslie Lamport. 1986. *LaTeX: A Document Preparation System*. Addison-Wesley, Reading, MA.

[27] Newton Lee. 2005. Interview with Bill Kinder: January 13, 2005. Video. *Comput. Entertain.* 3, 1, Article 4 (Jan.-March 2005). https://doi.org/10.1145/1057270.1057278

[28] Dave Novak. 2003. Solder man. Video. In *ACM SIGGRAPH 2003 Video Review on Animation theater Program: Part I - Vol. 145 (July 27–27, 2003)*. ACM Press, New York, NY, 4. https://doi.org/99.9999/woot07-S422

[29] Barack Obama. 2008. A more perfect union. Video. (5 March 2008). Retrieved March 21, 2008 from http://video.google.com/videoplay?docid=6528042696351994555

[30] Poker-Edge.Com. 2006. Stats and Analysis. (March 2006). Retrieved June 7, 2006 from http://www.poker-edge.com/stats.php

[31] Bernard Rous. 2008. The Enabling of Digital Libraries. *Digital Libraries* 12, 3, Article 5 (July 2008). To appear.

[32] Mehdi Saeedi, Morteza Saheb Zamani, and Mehdi Sedighi. 2010. A library-based synthesis methodology for reversible logic. *Microelectron. J.* 41, 4 (April 2010), 185–194.

[33] Mehdi Saeedi, Morteza Saheb Zamani, Mehdi Sedighi, and Zahra Sasanian. 2010. Synthesis of Reversible Circuit Using Cycle-Based Approach. *J. Emerg. Technol. Comput. Syst.* 6, 4 (Dec. 2010).

[34] S.L. Salas and Einar Hille. 1978. *Calculus: One and Several Variable*. John Wiley and Sons, New York.

[35] Joseph Scientist. 2009. The fountain of youth. (Aug. 2009). Patent No. 12345, Filed July 1st., 2008, Issued Aug. 9th., 2009.

[36] Stan W. Smith. 2010. An experiment in bibliographic mark-up: Parsing metadata for XML export. In *Proceedings of the 3rd. annual workshop on Librarians and Computers (LAC '10)*, Reginald N. Smythe and Alexander Noble (Eds.), Vol. 3. Paparazzi Press, Milan Italy, 422–431. https://doi.org/99.9999/woot07-S422

[37] Asad Z. Spector. 1990. Achieving application requirements. In *Distributed Systems* (2nd. ed.), Sape Mullender (Ed.). ACM Press, New York, NY, 19–33. https://doi.org/10.1145/90417.90738

[38] Harry Thornburg. 2001. Introduction to Bayesian Statistics. (March 2001). Retrieved March 2, 2005 from http://ccrma.stanford.edu/~jos/bayes/bayes.html

[39] TUG 2017. Institutional members of the TeX Users Group. (2017). Retrieved May 27, 2017 from http://wwtug.org/instmem.html

[40] Boris Veytsman. [n. d.]. acmart—Class for typesetting publications of ACM. ([n. d.]). Retrieved May 27, 2017 from http://www.ctan.org/pkg/acmart