Managing Writers

Sample Chapter -- Measurement and Metrics

Richard L. Hamilton

XML PRESS

Managing Writers
Copyright © 2009 Richard L. Hamilton

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means without the prior written permission of the copyright holder, except for the inclusion of brief quotations in a review.

This sample ebook may be distributed freely, but may not be sold in any form. Except for brief quotations in a review, any distribution must include the full copyright and all legal notices.

Disclaimer

The information in this book is provided on an "as is" basis, without warranty. While every effort has been taken by the author and XML Press in the preparation of this book, the author and XML Press shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book.

This book contains links to third-party web sites that are not under the control of the author or XML Press. The author and XML Press are not responsible for the content of any linked site. Inclusion of a link in this book does not imply that the author or XML Press endorses or accepts any responsibility for the content of that third-party site.

Trademarks

XML Press and the XML Press logo are trademarks of XML Press.

All terms mentioned in this book that are known to be trademarks or service marks have been capitalized as appropriate. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

XML Press
Fort Collins, Colorado 80528
http://xmlpress.net

First Edition
ISBN: 978-0-9822191-0-2

Printed in the United States of America.

Table of Contents

About the Book

This ebook contains a sample chapter, the glossary, and the bibliography from, *Managing Writers: A Real-World Guide to Managing Technical Documentation [http://xmlpress.net/publications/managing-writers]*, by Richard L. Hamilton. The book is available on Amazon.com [http://www.amazon.com/gp/product/0982219105], and will be available for the Kindle and in ePub and Mobi formats early in 2010.

Please feel free to distribute this ebook to anyone, provided you do not charge for distribution and provided you keep the copyright and legal notices unchanged in all copies you distribute (see the *Legal Notices* for details).

Send feedback to: mailto:publisher@xmlpress.net.

1

Measurement and Metrics

What happens depends on our way of observing it or on the fact that we observe it.
— Werner Heisenberg

I worked as a software developer on an early version of network file sharing, the now common feature that lets you access a folder on another computer as though it were actually on your hard drive. During development, we met with Bill Joy, one of the founders of Sun Microsystems, to share information about the feature, which Sun was also working on.

Bill watched our demo with interest and asked one question, "How fast can you transfer data?" We had barely considered this question. We were consumed with the technical problem of making sure we never lost data and that programs worked the same way over the network as they did locally. Sun was carefully measuring speed, we were carefully measuring compatibility.

When our two products came out, Sun's product was without a doubt faster and ours was without a doubt more compatible.[1] Even though we were, on the surface, building the same capability, the result was significantly different because of what each company valued, and therefore, measured.

The Impact of Measurement

Measurements tell engineers what you value. If you keep track of the number of pages writers produce, they will produce lots of pages. If you keep track of how many errors get reported by customers, they will keep that number low. If you track every milestone, they will do whatever it takes to meet every milestone.

If you measure anything, the odds are overwhelming that that metric will improve over time. However, the odds are also overwhelming that something else will suffer. If you track how many pages get written, that number will go up, but readability will probably go down, since writers will not have an incentive to be concise. If you track errors, that number will go down, but the number of pages written per unit of time will also go down, as writers spend more time on each page looking for errors.

This will happen even if you do not use the metrics in performance evaluations. Writers understand that managers measure what they consider important. And they understand that metrics influence performance evaluations, even when managers claim they will not.

Managing technical writers by measuring output is a bit like controlling the shape of a balloon. If you press one spot, another will pop out. If you measure page count, writers will feel pressure to sacrifice other qualities of the content to increase page count. If you see that, and control some other measure, yet another dimension will pop out.

Management Strategies

Robert D. Austin's *Measuring and Managing Performance in Organizations*[3] formally makes the case that I have outlined above. That is,

[1] For those who care, our product was Remote File System (RFS) and the Sun product was Network File Sharing (NFS).

unless you measure every *critical dimension*[2] of effort, any measurement will result in dysfunction.

If you can measure every critical dimension – unlikely, though maybe just possible in a very simple job classification – then Austin would advocate doing exactly that; he refers to this as *Full Supervision.* In this case, every dimension that matters is measured and the manager has complete control. In addition to finding and measuring every critical dimension of effort, the manager also needs to set the right value for each of these objectives to optimize results.

If you do not measure every critical dimension, but try to manage as though you do, then you are exercising *Partial Supervision.* According to Austin, partial supervision will *always* lead to dysfunction because whenever you leave a critical dimension unmeasured, that dimension will be neglected in comparison with the measured dimensions, and because that dimension is critical, customer value will, at least in the long run, suffer. Many managers believe they are exercising full supervision when they are really exercising partial supervision, which exacerbates the dysfunction.

There may be workplaces where you can measure every critical dimension, but technical communication is not one of them. There are too many critical dimensions and many of them are too expensive or too difficult to measure or control. So, if full supervision is impossible and partial supervision is dysfunctional, what's a manager to do?

Austin defines one other mode, *Delegatory Management*, which eschews measurement in favor of delegating power to workers. It relies on them to select the amount of effort they will devote to each of the critical dimensions. It depends heavily on the manager establishing an environment of trust and mutual respect, and it also depends on internal motivation as a driving force. As we will see, delegatory management does not completely discard measurement; instead, it uses appropriate measures to provide information about the process and progress. But, it does not use measures to reward or punish individuals or groups.

[2]A critical dimension is one that if neglected will result in the customer receiving no value from the product or service.

Measurement Strategies

About this point you are probably thinking, "but, if I don't measure anything, how do I know how my team is doing?" Clearly some measurements are essential to tracking progress and cannot be ignored. The questions are, what do you measure, who makes the measurements, and how do you use them?

What to measure

I directly measure as little as possible. In fact, I have managed teams where I measured nothing. Instead, I delegated all tasks to team members, including all metrics. However, even if you delegate everything, there are a few things that must be tracked, even if only by an individual for his or her own purposes.

- **Milestones:** You have no real choice but to track milestones. As we have seen, you need to know as early as possible whether there is a problem you must address. However, I almost never personally track fine-grained interim milestones, and I frequently delegate even high level project milestones to the responsible engineer. Depending on the engineer's level of skill and experience, I delegate as much as possible, and stay in the loop only to the extent needed to report progress up the line.
- **Customer satisfaction:** While you may not want to use surveys or other formal means for measuring customer satisfaction, you do need to be responsive to customer concerns. I like to direct customer feedback to the person who is responsible for the content in question and let him or her handle the concern. While I may comment on how well someone handles a customer concern, I do not use the number of complaints or any other numerical measure in an evaluation.

In these two cases, and others like them, I am more interested in how writers handle the situation than I am in any kind of numerical measure. Milestones are met or missed for all sorts of reasons, not all them under the control of the writer. Customer concerns are equally indeterminate, at least with respect to number. There is no way you can learn much about writers by counting complaints, though you can learn something by seeing how they handle the complaints they get.

Generally, I avoid any measurement that has a numerical component. This includes page counts, topic counts, number of engineers per writer, time per page to edit, number of errors discovered per page, and so forth, ad nauseum. As soon as you start using a numerical measure, someone will try to optimize it, resulting in the kinds of dysfunction discussed above.

Who should measure

As much as possible, have writers make and use their own measurements. For example, page count and pages written per day can be useful in estimating the effort required to complete a job. In a delegatory management style, writers would keep track of these measurements and use them to come up with personal metrics they can use to estimate projects.

But, if *managers* measure those things, even if they do not intend to use them for comparison, they risk dysfunction. Therefore, I do not track this information; I let writers do that if it helps them, and I only get involved to help a new writer or when asked.

Once or twice, I have been asked to collect and report some numerical metric like pages per day. If you find yourself in this situation, my best recommendation is to find some analog in that manager's background (for former programmers it might be lines of code, for former marketers it might be number of ads or press releases) and use that to convince them that counting output is a bad idea. Nearly every discipline has some verboten measurement that is analogous to page counts; find it and use it. So far, this technique has helped me dodge that bullet.

How to use measurements

Austin identifies two uses for metrics, motivational and informational. He only advocates using motivational metrics when you can exercise full supervision. The catch is that nearly any metric can be used for either motivational or information purposes.

For that reason, I prefer to measure as little as possible and delegate measurement as far down as possible. In addition, I make it clear I will not use metrics as part of employee evaluation. Of course, I will have a discussion about milestones that are met or not met, that is inevitable. But, if writers have generated those milestones themselves, the discus-

sion can be directed towards making better milestones, rather than towards reward or punishment based on making or not making a milestone.

There is a fine line here between management behavior that distorts and management behavior that enhances. You will not be able to make a perfect call in all situations, but if you have build a trusting, delegatory environment, the odds are that you will avoid the worst dysfunction.

Summing Up

Here are some thoughts to sum up the use of metrics.

- **Pay attention to what you measure:** People will assume that what you measure is what you value. A trivial example: If you measure pages produced per day and do not measure the quality of those pages, your writers will presume you care more about volume than quality.
- **Never use metrics in a PE:** As soon as you cite a metric in a performance evaluation, that metric will be optimized, not just by that person, but by everyone on your team. The metric will be optimized even if the optimization lowers the value of your product to customers.
- **Never use metrics to compare people or teams:** The same thing will happen in this case that happens if you use a metric in a performance evaluation.
- **Measure for information:** Certain measurements are necessary. For example, you need to keep track of milestones. When maintaining content, you need to keep track of error reports. And, you need to use these measurements for legitimate management purposes like estimating effort, identifying problems, and reporting status to the project team.

  There is always the risk that these measures will be interpreted as evaluation measures, but you can limit this risk by minimizing individually identifiable measures, especially in reports to management, and by using the data strictly for its informational value.

- **Your credibility matters:** The fine line between measuring for information and measuring for evaluation is drawn based on the level of trust between you and your team. If you ever use an informational measurement in an evaluation or if there is a low level of trust for other reasons, you can be sure that your team will perceive any measurement as evaluative.
- **Let people measure themselves:** The most sensitive measures for writers are productivity measurements like page counts. These measurements can be useful for estimating effort and for judging when a project is off track. Let your team members estimate their effort using metrics they collect themselves. Delegate to them the job of measuring and acting on the results.
- **Do not punish *or* reward based on metrics:** It does not matter whether you use a metric as a basis to reward or punish. If you reward someone for producing more pages you will see the same result as if you punish someone for producing fewer pages. Either way, people will perceive page count as something you value and act accordingly.
- **Resist the pressure to measure productivity:** Measures like page counts are insidious to productivity. Your team may use page counts as part of their estimation process, but avoid using page counts as a productivity measure. If you are pressed by management to measure productivity using volume metrics, press back.

The bottom line for me is to measure only what you absolutely must, never use measurement as part of a performance evaluation, resist the pressure to report productivity metrics to management, and let writers manage their own metrics.

Glossary

Agile Methodology

According to Wikipedia, "Agile Software Development is a conceptual framework for software development that promotes development iterations, open collaboration, and adaptability throughout the life-cycle of the project."[39]

Agile methods have been around for many years, but the origin of the term "Agile Methodology" dates from 2001, when a group of software developers met at The Lodge at Snowbird in Utah and penned the *Agile Manifesto*.[4]

See also: Extreme Programming, Scrum.

ASCII

The American Standard Code for Information Interchange (ASCII) is a code set commonly used in the English speaking world. It can represent the 26-character alphabet used for most English language words, upper and lower case, plus the most commonly used other characters (for example, punctuation, numbers, and various other symbols).

Because ASCII cannot represent the vast majority of the world's languages – in fact, it cannot fully support English, either, if you consider words like "résumé," which use characters outside the 26-character

alphabet – it has been replaced in many contexts, including the XML standard, by Unicode.

See also: Code Set, Unicode.

Attribute

In XML, an attribute is a keyword-value pair inside the start tag of an element that provides additional information about the element. For example, in the following element, the `role` attribute says that this content should be given a `strong` emphasis.

```
<emphasis role="strong">
  some important text
</emphasis>
```

See also: Element.

Audience

The audience is the group of people who will be using your product.

See also: Product.

Code Set

A code set (or coded character set) maps each character of a language to a unique number, which a computer can use in its calculations. For example, the character "A" is represented by the number 65 in the ASCII code set.

The most common code set in use today is Unicode, which maps a large portion of the world's characters into a single code set.

See also: ASCII, Unicode.

Controlled Natural Language	Controlled Natural Languages (CNL) are subsets of the grammar and vocabulary of a natural language designed to reduce ambiguity, improve readability, and facilitate translation. Some CNL's are designed to be completely machine parsable, others are designed to be more easily read by humans. The CNL's of interest to technical communicators are in the latter category and include: Plain English, Simplified English, and Special English. See also: Simplified English.
Core Competency	A core competency is an area of expertise, skill, or technology that is fundamental to a company or persons activity. Skill at playing the violin is a core competency for Itzhak Perlman. Product design is a core competency of Apple Computer. A core competency typically gives its possessor a competitive advantage; this is certainly the case with the two examples here.
Cost Center	A cost center is a unit within a corporation that is not expected to generate revenue. While a cost center is not expected to generate revenue, it is expected to provide services or other deliverables that are critical to the corporation's success. Within a cost center, new projects are evaluated based on their ability to reduce costs. See also: profit center.
Critical Dimension	A critical dimension of effort is a part of a job that if neglected will decrease or eliminate customer value. For technical writers, critical dimensions include readability, accuracy, and completeness.

Delegatory Management	Delegatory management is a style of management that delegates nearly all responsibility to the worker. In the work of Robert D. Austin[3] it refers to a management style that delegates measurement and interpretation of metrics to workers.
Deliverables	The tangible things that writers deliver to the project. For example, User Guides, Administrator Guides, Manuals, etc.
Developers	The people who design and build a product. See also: Product.
Development Methodology	A methodology for managing a project, including objectives, schedules, milestones, resources, and so forth. Development methodologies typically fall into two categories: Sequential methodologies define a linear sequence of phases, for example, requirements, design, implementation, test, and deployment. Iterative methodologies define multiple short cycles, typically two to four weeks each.
DITA	Darwin Information Typing Architecture (DITA) is an XML-based architecture for authoring, producing, and delivering technical information. Information can be found at: http://dita.xml.org.
DocBook	A widely used XML grammar designed for developing technical documentation. Information can be found at: http://docbook.org.
DTD	Document Type Definition: used to define an SGML or XML grammar.

See also: Schema.

Element

In XML, an element is the basic structural building block. An element comprises a start tag, some content, which may include other elements, followed by an end tag. The start tag of an element may also contain one or more attributes.

Here is an example of an element named "emphasis" that contains some content and an attribute named "role" with the value "strong."

```
<emphasis role="strong">
  some important text
</emphasis>
```

See also: Attribute.

Environment

The environment is the set of tools, processes, and personnel that a writer works with.

Extreme Programming

Extreme Programming is an agile software development methodology. Like most agile methodologies it emphasizes small releases, open communication, and continuous integration. In addition, it uses the concept of "pair programming," where two programmers work together at one computer, one typing code and the other reviewing the code as it is entered.

See also: Agile Methodology, Scrum.

Flexibility Matrix

A flexibility matrix documents the degree of flexibility a project, or sub-project, has in each of the three planning dimensions (content, time, and resources). It is simply

	a matrix that orders the three from most to least flexible. While it is a crude measure, it forces the project to consider and document priorities.
Full Supervision	Full supervision is a mode of management where the manager identifies and measures every *critical dimension* of effort from each member of his or her team, and uses those metrics to evaluate employee performance.
GML	Generalized Markup Language. An early markup language developed at IBM by Charles Goldfarb, Edward Mosher, and Raymond Lorie.
Information Mapping	Information Mapping™ is a specialized methodology, developed and owned by Information Mapping, Inc., for analyzing, organizing, and presenting information. Information about this methodology can be found at Information Mapping, Inc. [http://infomap.com] See also: Specialized Methodology.
Locale	A locale defines a set of user preferences related to location, language, and national conventions. The locale typically defines a language and location (country or territory), plus the character set(s), date and time formats, timezone and daylight savings time conversions, currency formats, and numeric representation.
Mashup	A mashup is a web-based application that takes information from multiple sources and creates a web service that combines the content available from those sources.

Milestone	The term "milestone" originally referred to one of a series of markers along a road, marking each mile along the route. In the project management world, a milestone is an event, either a singular occurrence like "prototype hardware received from Engineering" or the end point of a process like "copy edit completed." In either case, the milestone should be clear and measurable.
Modular Methodology	Modular methodologies for technical communicators decompose content into modules of several different types. Writers develop modules independently, then combine a selected group of modules to create each deliverable. Another common term for modular methodologies is topic-based authoring. Examples include Information Mapping and DITA. While DITA is not itself a methodology, most DITA users follow a topic-based methodology. See also: DITA, Information Mapping, Topic-based Authoring.
Mouseover	Mouseover is a GUI action that occurs when a user moves the mouse cursor over some position in the interface, but does not click a mouse button. On a web page, a mouseover is commonly used on links and interactive elements like buttons. Mouseover actions on a web page typically display information like the URL for a link, a description of a button's action, related text (for example, a glossary definition), or a thumbnail of the site a link points to.
Partial Supervision	Partial supervision is a mode of management where the manager identifies and

	measures some, but not all, *critical dimensions* of effort from each member of his or her team, and uses those metrics to evaluate employee performance. Partial supervision leads to dysfunctions in the team by allowing some critical dimensions of effort to be unmeasured and therefore neglected.
Product	The product is whatever you are writing about, even if it is not a product. It could be a service, software, hardware, an airplane, or a toaster.
Profit Center	A profit center is a unit within a corporation that is expected to generate revenue that exceeds expenses. Within a profit center, new projects are evaluated based on expected Return on Investment (ROI). See also: cost center, ROI.
Repurpose	Content repurposing means that you deliver the same piece of content via different media. For example, if you deliver the same document in print and also on the web, that would be *repurposing*. See also: Reuse.
Reuse	Content reuse means you put the same piece of content in more than one deliverable on the same output medium. For example, if you maintain a single copy of a glossary definition in source control, then include it in the printed versions of your Installation Guide and User's Guide, that would be *reuse*. See also: Repurpose.

ROI	Return on Investment (ROI) is the ratio of profit (or loss) relative to the investment for a project. In finance, ROI is typically expressed as a percentage and is projected for several years into the future. The term is often used less formally to describe the expected monetary gain from taking some course of action.
Schedule	The schedule comprises the timeline and milestones for a project.
Schema	A schema defines the grammar of an XML document. There are several languages used to represent schemas, including: Relax NG [http://relaxng.org], the W3C XML Schema Language [http://www.w3.org/TR/xmlschema-0/], and DTDs. These languages each have their strengths and weaknesses in defining any particular grammar. Therefore, standards bodies typically select one of the schema languages to define the normative (official) version of a particular XML grammar. At this time, the normative schema for DocBook is defined using RelaxNG and the normative schema for DITA is defined using a DTD. See also: DTD.
Scrum	Scrum is an agile software development methodology characterized by multiple short cycles (two to four weeks "sprints"), frequent short communication meetings (called "scrums"), and well defined roles. See http://scrumalliance.org for more information. See also: Agile Methodology, Extreme Programming.

SGML	Standard Generalized Markup Language. A precursor, and origin, of XML. For further information, see http://www.w3.org/MarkUp/SGML
Simplified English	Simplified English is one of several specialized methodologies that attempt to improve readability, reduce ambiguity, and make translation cheaper and easier. Simplified English defines a set of writing rules and a basic vocabulary that can be supplemented with technical terminology specific to the domain being documented. Related methodologies include: Plain English and Special English. All are examples of Controlled Natural Languages. Information about Specialized English can be found at: http://www.asd-ste100.org/. See also: Controlled Natural Language.
Single Sourcing	Single sourcing is a method for reusing or repurposing content to minimize duplication. For example, suppose you have written a procedure for adjusting bicycle chains. If you take the source for that procedure and transform it into a web page and also a printed pamphlet, without altering the original source, you have "single sourced" that content. See also: repurpose, reuse.
Specialized Methodology	A specialized methodology is a methodology for some particular aspect of your work. Single-sourcing, Information Mapping, and Controlled Natural Languages are specialized methodologies used by technical communicators.

See also: Controlled Natural Language, Information Mapping.

Tasks

The tasks are the set of things the audience will be doing with the product.

See also: Audience, Product.

Topic-based Authoring

Topic-based authoring is a specialized methodology for content development. A topic is a self-contained piece of content about a particular subject. Topics are authored independently, then combined to create documentation deliverables. Types of topics include conceptual, procedural, and reference. Users of the DITA XML schema typically use a topic-based methodology.

See also: DITA, Information Mapping, Specialized Methodology.

Troff

A document processing system developed in the late 1960s at Bell Laboratories. It was derived from earlier work at Massachusetts Institute of Technology.

Unicode

An encoding system that maps every character in nearly any language to a unique encoding. By providing a unique encoding, Unicode allows text in multiple languages to sit side by side in a document and be processed by any application that understands Unicode. The Unicode Consortium [http://unicode.org] manages the standard, and their web site provides detailed information about the standard and its use.

See also: UTF-8.

Use Case

A description of a user task, usually cast in the form of a person interacting with a system to reach some objective. Use Cases are often used as part of a Requirements Specification.

UTF-8

UTF-8 is a character encoding for Unicode. The Unicode standard identifies characters through an abstract coding that can be implemented in computer systems in many different ways. UTF-8 is the most common character encoding implementation for Unicode.

UTF-8 is the default coding for XML documents, and all XML parsers must support it. It is backwards compatible with ASCII, which makes it easy to use in English only environments. Unless you have unusual needs, UTF-8 is your best choice for character encoding in XML.

See also: Unicode.

Wiki

A wiki is a web site that allows users to edit pages on the site. The best known wiki, Wikipedia [http://wikipedia.org], is an encyclopedia that allows anyone to create or modify entries.

The term Wiki means "quick" in the Hawaiian language. The term WikiWiki, or "very quick" is also used, and is the part of the name of the first Wiki application, "WikiWikiWeb." Additional information can be found at: Wiki (Wikipedia) [http://wikipedia.org/wiki/wiki].

XInclude

The XML-based XInclude standard is a generalized inclusion mechanism that allows you to create a document from other

	XML documents or fragments of XML documents. Additional information can be found at: http://www.w3.org/TR/xinclude.
XML	Extensible Markup Language. XML is a specification for defining customized markup languages. Common XML markup languages for technical documentation include: DITA, DocBook, and S1000D. Additional information can be found at: http://www.w3.org/XML.
XSL	Extensible Stylesheet Language. A set of transformation languages that are used to transform XML instances in various ways. XSL can be used to format XML instances into output formats such as HTML or PDF. It can also be used to transform XML instances in other ways, such as generating tables of contents, extracting data, or re-structuring content. Additional information can be found at: http://www.w3.org/Style/XSL.

Bibliography

[1] Scott W. Ambler, August 2007, *IT Project Success Rates Survey:2007*, http://www.ambysoft.com/surveys/success2007.html.

[2] Kurt Ament, 2002, *Single Sourcing:*, Building Modular Documentation, Noyes, ISBN: 0-8155-1491-3.

[3] Robert D. Austin, 1996, *Measuring and Managing Performance in Organizations*, Dorset House, ISBN: 0-932633-36-6.

[4] Kent Beck, et al, 2001, *The Agile Manifesto*, http://agilemanifesto.org.

[5] Matthew Berk, 2003, *Website content management: covering the essentials, avoiding overspending*, Jupiter Research, http://jupiterresearch.com, Summary at: http://www.atnewyork.com/news/article.php/1690881.

[6] Jon Bosak, 2006, *Closing Keynote, XML 2006*, XML 2006 Conference, December 5-1, 2006, Boston, MA, Idealliance, http://2006.xmlconference.org/proceedings/162/presentation.html.

[7] Frederick P. Brooks, 1995, *The Mythical Man-Month*, Essays on Software Engineering, 20th Anniversary Edition, Addison-Wesley, ISBN: 0-201-83595-9.

[8] Marcus Buckingham and Donald O. Clifton, 2001, *Now, Discover Your Strengths*, The Free Press, ISBN: 0-7432-0114-0.

[9] Pip Coburn, 2006, *The Change Function*, Why some technologies take off and others crash and burn, Portfolio, ISBN: 1-59184-132-1.

[10] Alistair Cockburn, 2000, *Writing Effective Use Cases*, Addison-Wesley Professional, ISBN: 0-201-70225-8.

[11] Alistair Cockburn, *Alistair Cockburn's Web Page/Wiki*, http://alistair.cockburn.us.

[12] W. Edwards Deming, 1982, *Out of the Crisis*, Massachusetts Institute of Technology, Center for Advanced Engineering Study, ISBN: 0-911379-01-0.

[13] Donald A. DePalma, 2003, *Rage Against the Content Management Machine*, Common Sense Advisory, http://www.commonsenseadvisory.com/news/pr_view.php?pre_id=4.

[14] R. Stanley Dicks, 2003, *Management Principles and Practices for Technical Communicators*, Longman, ISBN: 0-321-16523-3.

[15] Edsger W. Dijkstra, 1968, *Go To Statement Considered Harmful*, *Communications of the ACM*, Association for Computing Machinery, Inc., March, 1968, vol. 11, no. 3, 147-148, http://www.cs.utexas.edu/users/EWD/transcriptions/EWD02xx/EWD215.html.

[16] Charles F. Goldfarb, 1996, *The Roots of SGML*, A Personal Recollection, http://www.sgmlsource.com/history/roots.htm.

[17] JoAnn T. Hackos, 1994, *Managing Your Documentation Projects*, John Wiley & Sons, Inc., ISBN: 0-471-59099-1.

[18] JoAnn T. Hackos, 2007, *Information Development: Managing Your Documentation Projects, Portfolio, and People*, John Wiley & Sons, Inc., ISBN: 0-471-77711-0.

[19] Kathy Haramundanis and Larry Rowland, 2007, *Experience Paper – A Content Reuse Documentation Design Experience*, SIGDOC 2007, October 22-24, 2007, El Paso, TX, Association for Computing Machinery, http://sigdoc2007.org .

[20] , 2008, *Rewarding India*, Tradition Meets Transformation, http://www.haygroup.com/Downloads/sg/misc/Rewarding_India_web.pdf.

[21] , 1986, *ISO 8879:1986 Information processing – Text and office systems – Standard Generalized Markup Language (SGML)*, http://www.iso.org.

[22] Neon Kelly, 2007, *High failure rate hits IT projects*, Computing, http://www.computing.co.uk/computing/news/2197021/failed-projects-hit-half-uk.

[23] John P. Kotter, 1996, *Leading Change*, Harvard Business School Press, ISBN: 0-87584-747-1.

[24] John P. Kotter, 1985, *Power and Influence*, Beyond Formal Authority, The Free Press, ISBN: 0-02-918330-8.

[25] Jakob Nielsen, *Concise, SCANNABLE, and Objective: How to Write for the Web*, http://www.useit.com/papers/webwriting/writing.html.

[26] Jakob Nielsen, May 6, 2008, *How Little Do Users Read?*, http://www.useit.com/alertbox/percent-text-read.html.

[27] Jakob Nielsen, *Writing for the Web*, http://www.useit.com/papers/webwriting.

[28] Janice (Ginny) Redish, 2007, *Letting Go of the Words: Writing Web Content that Works*, Morgan Kaufmann, ISBN: 0-12-369486-8.

[29] William P. Rogers, 1986, *Report of the Presidential Commission on the Space Shuttle Challenger Accident*, http://history.nasa.gov/rogersrep/genindex.htm.

[30] Louis Rosenfeld, Peter Morville, 2006, *Information Architecture for the World Wide Web (3rd edition)*, O'Reilly, ISBN: 0-596-52734-9.

[31] , *Scrum Alliance*, http://www.scrumalliance.org.

[32] C. M. Sperberg-McQueen and Lou Burnard, 1994, *A Gentle Introduction to SGML*, http://www.isgmlug.org/sgmlhelp/g-index.htm.

[33] Joel Spolsky, 2004, *Joel on Software*, Apress, ISBN: 1-59059-389-8.

[34] Alexandra L. Bartell, Laura D. Schultz, and Jan H. Spyridakis, *The Effect of Heading Frequency on Comprehension of Print versus Online Information*, *Technical Communication*, November, 2006, vol. 53, no. 4, 416-426,

[35] William Strunk, Jr. and E. B. White, 1979, *The Elements of Style*, MacMillan Publishing Company, ISBN: 0-02-418220-6.

[36] Bob Stayton, 2007, *DocBook XSL: The Complete Guide*, Sagehill Enterprises [http://sagehill.net], ISBN: 0-9741521-3-7.

[37] Norman Walsh, Leonard Muellner, 1999, *DocBook: The Definitive Guide*, O'Reilly & Associates, Inc., ISBN: 1-56592-580-7.

[38] Harald Weinreich, February 2008, *Not Quite the Average: An Empirical Study of Web Use*, *ACM Transactions on the Web*, Association for Computing Machinery, Inc., March, 1968, vol. 2, no. 1, article #5, http://doi.acm.org/10.1145/1326561.1326566.

[39] , *Agile Software Development*, http://en.wikipedia.org/wiki/Agile_Project_Management.

[40] , 16 August 2006, *Extensible Markup Language (XML) 1.0 (Fourth Edition)*, http://www.w3.org/TR/xml.

[41] Ray W. Wolverton, 1974, *The Cost of Developing Large-Scale Software*, *IEEE Transactions on Computers*, IEEE, June 1974, vol. c-23, no. 6, 615-636, http://csdl.computer.org/comp/trans/tc/1974/06/01672595.pdf.

[42] Richard Saul Wurman, 2000, *Information Anxiety 2*, Que, ISBN: 0-7897-2410-3.