



# Document Diseases and Software Malpractice

Gregory T. Daich

Software Technology Support Center/SAIC

*This article proposes some names for software documentation diseases based on human diseases that will surely invoke a feeling of alarm to motivate developers to plan immediate treatment to keep project costs and quality under control. To ignore and leave these documentation diseases untreated may one day be considered software malpractice. This article can help to identify serious documentation maladies early when treatment is possible to maintain a lean, healthy, and on-time project.*

Disciplined document reviews are somewhat analogous to medical instrument sterilization. Today, we would not tolerate doctors using unsterilized equipment to conduct surgery. However, it was common practice, for example, during the Civil War to use a surgical instrument such as a saw without proper sanitation between limb amputations. This caused more soldiers to die of infections than from the initial wounds. Once scientists learned about bacteria and other microbes, the medical industry began requiring that equipment be sterilized to avoid infecting patients.

Today, we have plenty of experience data that shows disciplined document review practices are vital to delivering systems on-time and within budget [1, 2]. However, many software development organizations still have no problem using yesterday's practice of whipping together a set of requirements and delivering it to the developers and testers without an adequate review. They do not take the time nor use effective tools to look for document defects. Either they do not see any obvious bugs, or they must figure the bugs can be fixed later. Consequently, they do not seem to mind delivering a document with serious diseases, so to speak. However, it is a well-known fact that defects cost more to fix the longer they remain in the system documentation (and code) [3].

Yesterday's state-of-the-practice is often today's malpractice. Let me go on record as saying that organizations not implementing a disciplined document review program will one day be considered to be conducting software malpractice. Disciplined document reviews are that important!

Disciplined document reviews are also known as peer reviews, inspections, and structured walk-throughs. While there are some differences in definitions for these terms in various standards and guidelines [4] in the industry, the intent is to identify defects to determine if the document is

ready for release to the next phase of development or for delivery to the customer. The author should prepare the document to the point that he or she believes that there are few, if any, defects remaining, and that it is considered ready for release to the next phase of development or delivery. Another major purpose for these reviews is to identify process improvement opportunities to avoid making the same mistakes on future efforts.

## Human Disease Analogies

In any discipline, we provide names to characterize the topics, objects, and projects that we need to discuss. Naming a concept helps us gain an initial understanding of the issue. If naming document maladies after human diseases could help software engineers better understand and deal with the documentation problems confronting them, then I propose doing so. There are several document diseases that I have seen in my research at the Software Technology Support Center (STSC) and during my support of STSC clients. Since many of us are not readily conversant with human diseases, here are a few definitions for review [5]:

- **Rickets.** A deficiency disease resulting from a lack of vitamin D, marked by defective bone growth and occurring chiefly in children.
- **Sclerosis.** A thickening or hardening of a bodily part, especially from disease or excessive growth of tissue.
- **-itis.** Inflammation or disease of [a body part or parts].
- **Scurvy.** A disease resulting from a deficiency of vitamin C and marked by bleeding under the skin.
- **Glaucoma.** A disease of the eye marked by high intraocular pressure and partial or total vision loss.
- **Cancer.** Malignant neoplasms that manifest invasiveness and have a tendency to metastasize to new sites.

After this quick review, surely you can see that relating these names to document defects could invoke a feeling of alarm

that would require immediate treatment to keep project costs and quality under control. The following sections discuss several document diseases to treat immediately on your projects.

## Requirements Rickets

Requirements rickets is a deficiency of important requirements resulting from a lack of exposing requirements to adequate analysis and review. For example, requirements state that a behavior shall occur given a specific condition. But they do not tell you what to do when the condition is not met. This is a common cause of requirements rickets: The symptom is missing requirements. Granted, an iterative or evolutionary development life cycle does not expect all requirements to be defined at the start. But when you define the requirements that you do know about, be sure they are adequately specified.

Another example of requirements rickets occurs when we state that we want a usable system and then do not provide any objective scale of measure for usability. We are missing vital information, without which we can neither build the desired system nor evaluate whether it has been successfully accomplished. Example scales of measure for usability include the average time to learn to use specific product features and the average time to perform specific product features by experienced personnel [1].

## Source Code Sclerosis

Source code sclerosis is a hardening of the source code, making it very difficult to correct or upgrade without breaking some other existing capability. Many old legacy systems are built with breakable design architectures and poor coding practices (for example, inadequate comments, unstructured code, or poorly named variables). These systems require intensive care to revive them to prolong their lives many times. We often spend 70 percent or more of the entire life cycle doing maintenance on many of these systems [6].

Some of these systems with source code sclerosis need to be taken off life support. They need to be redesigned and rebuilt to cost-effectively meet operational requirements. With the support of automated static analysis tools, effective document reviews assures that maintainable code is developed.

### Ambiguositis

Ambiguositis is a disease of ambiguity inherent in all natural languages and is as common as the common cold. Sometimes it provides some variety, mystery, or humor to fictional prose but it always causes literary dizziness and confusion in software documentation. People can legitimately interpret the same specifications in different ways. For example, consider the following statement: "If Transaction\_A is received, and it is the end of the week, or it is the end of the month, produce Summary\_A Report."

This text can be interpreted in at least two distinct and valid ways using the parentheses to clarify as follows:

- Interpretation 1: "If (Transaction\_A is received, and it is the end of the week) or it is the end of the month, produce Summary\_A Report."
- Interpretation 2: "If Transaction\_A is received, and (it is the end of the week or it is the end of the month), produce Summary\_A Report."

Making an assumption about what the original text means is dangerous to the health of the project. Interpretation 1 always produces Summary\_A Report at the end of the month. It also produces Summary\_A Report if it is the end of the week and Transaction\_A has been received. Interpretation 2 produces Summary\_A Report only if Transaction\_A is received and either it is the end of the week or it is the end of the month. (Is that clear?) In other words, the results are different depending on the reader's interpretation. This disease has caused entire projects to die a slow and painful death, though the symptoms were often known very early and could have been diagnosed and treated.

### Source Document Scurvy

Source document scurvy is a malicious disease resulting from a deficiency of source document references and is rampant in many project documents. Its effects are often tolerated as part of life when the quality of our projects could be greatly improved with the adoption of a few effective antidotes. If you obtain information from a source document, then provide a useful citation to that document such as

a reference identifier with a page or section number (if it is not obvious where the information came from). It takes only a little more time and saves countless review and rework hours later.

Note that a reference section should always be provided in all documentation, and includes the titles, authors, dates, and version numbers of all sources. No document is an island. Some managers want the documents that they have to review to stand on their own. Thus the authors hide vital reference information by not listing all the applicable references. You can often find inconsistencies between documents when you check it under review against the source document.

### Document Glaucoma

Document glaucoma is a disease caused by unclear or missing document and project objectives. It is a frequent illness in special reports, plans, or guidelines to support software development or maintenance efforts. When I am asked to review these types of documents, I often cannot find a statement of objectives. With no vision of where we are really headed in some of these documents, it is a wonder we make any progress at all.

Software-related documentation written by inexperienced authors often jumps right into the body of the document with no proper introduction. This may have something to do with the mindset of some developers that documentation is bad because it slows us down. That is like saying we do not have time to document our plans or our requirements. We just have time to build. "We'll document later," is a common comment I have heard from a variety of sources. This is a clear example of document glaucoma; unclear project objectives is its main symptom.

### Content Cancers

Content cancers occur whenever an uncorrected, often malignant documentation problem is not corrected prior to delivery to the next phase of development. IBM studied how design defects propagate to multiple design defects, and how design defect propagates to multiple coding defects [6]. Again, documents written in early project phases often transmit these cancerous defects to follow-on phases because they were not diagnosed and the disease treated when authors had a chance – shortly after writing the document. This disease continues to grow maliciously to damage schedules and costs and ultimately kill projects. One particularly malignant strain of this disease manifests itself in "required" but trivial overkill documenta-

tion that does not support key project objectives.

### Disease Cure and Prevention

Unlike some human diseases, these document diseases are completely diagnosable and curable using technologies available today. The technologies are not difficult but are a collection of common-sense activities that require training where effective practices can be experienced. However, effective document reviews require a significant process implementation effort following training. Many organizations do not make this vital investment.

Some of these common-sense activities include the following [1, 7]:

- Checking the document against objective criteria to determine readiness for review, and planning the review by a trained document review leader.
- Conducting a kick-off meeting to introduce the document to the reviewers and to answer questions.
- Checking the document against sources, checklists, standards and checking for ambiguities, incompleteness, inconsistencies, and missing sources.
- Meeting as a team of reviewers to report and check for additional defects.
- Conducting a process brainstorming meeting to begin root cause analysis shortly after the team review meeting.
- Correcting the document by the author or author-representative and addressing all issues and defects.
- Auditing the author's corrections.
- Verifying that objective document readiness criteria have been met by the review leader and recording metrics.

Once the review process is underway, it needs to be constantly monitored to assure that document review leaders are following the process. Without active management support and involvement, developers and support staff often slip back into archaic, skim-review malpractice (one quick gloss-over reading without checking sources and checklists). [1, 7, 8, 9, 10].

Poor document review practices persist in many organizations because there is no documented policy and process for conducting reviews; that is the way it has been done for years. Conducting haphazard ad hoc reviews allows critical project documentation to be transmitted unsterilized to subsequent phases of development.

We all know that "An ounce of prevention is worth a pound of cure." Effective document reviews also feed information back to developers to help improve the authoring process in the future.

Effective document reviews will not prevent all document diseases and will not guarantee success, but they go a long way toward maintaining healthy projects. They require up to 15 percent more time during the document authoring stages of development, but they save time overall on projects through significantly less rework and retesting [1]. It is like taking the time to eat well-balanced meals, participate in an effective exercise program, and get enough rest for your projects. You just cannot expect to live a healthy project lifestyle without effective and efficient document review practices.

With effective review practices, you can also expect more projects to reach full maturity. You will have fewer incidents of project euthanasia (cancellations) administered (which is probably the best idea for many suffering projects). You will also have more lean and healthy projects delivered on-time and on-budget [2].

Although I do not really expect any project to adopt my document disease naming convention, it brought to light some interesting issues about documentation maladies that are treatable today. One rule in conducting disciplined document reviews is to direct comments at the document, not the author. Naming document diseases this way may suggest that some authors are carriers of certain diseases. We do not really want to label authors this way. However, it may be fair to say that responsible managers who neglect implementing effective reviews of critical documentation may be the real carriers of the above document diseases.

The consequences of poor quality documentation can still result in preventable project fatalities just like individuals practicing poor health habits result in preventable fatalities. Again, to do anything less than implementing disciplined document reviews could eventually be software malpractice. ♦

### Acknowledgements

I would like to thank the following people for their comments regarding this article: Pam Bowers, Ross Collard, Rick Craig, David Dayton, Chelene Fortier, Tom Gilb, Paul Hewitt, Tony Henderson, Cem Kaner, Ed Kit, Bret Pettichord, Ron Radice, Johanna Rothman, Bob Stahl, Beth Starrett, Tracy Stauder, and Karl Wiegers.

### References

1. Gilb, Tom, and Dorothy Graham. Software Inspection. Boston: Addison-Wesley, 1993.
2. Dion, Raymond. "Process Improve-

ment and the Corporate Balance Sheet." CrossTalk Feb. 1994.

3. Boehm, Barry. Software Engineering Economics. Prentice Hall, 1981: 17.
4. Institute of Electrical and Electronics Engineers. "Standard for Software Review Processes". IEEE 1028-1997.
5. Webster's II New Riverside University Dictionary. The Riverside Publishing Company, 1984.
6. Collard, Ross, et. al. System Testing and Quality Assurance Techniques. Collard & Associates, 2001.
8. Ebenau, Robert G., and Susan H. Strauss. Software Inspection Process. McGraw Hill, 1993.
9. Radice, Ronald A. High Quality Low Cost Software Inspections. McGraw-Hill, 1993.
7. Daich, Gregory T. "Disciplined Document Reviews Course." Software Technology Support Center. Mar. 2002, Version 6.
10. Wiegers, Karl E. Peer Reviews in Software: A Practical Guide. Addison-Wesley, 2002.

### About the Author



**Gregory T. Daich** is a senior software engineer with Science Applications International Corporation currently on contract with the Software Technology Support Center (STSC). He supports STSC's Software Quality and Test Group with more than 25 years of experience in developing and testing software. Daich has taught public and on-site seminars involving software testing, document reviews, and process improvement. He consults with government and commercial organizations on improving the effectiveness and efficiency of software quality practices. Daich has developed two Air Force training programs: Software-Oriented Test and Evaluation, and Disciplined Document Reviews. He has a master's degree in computer science from the University of Utah.

Software Technology Support Center  
OO-ALC/MASEA  
7278 4th St.  
Bldg. 100  
Hill AFB, UT 84056-5205  
Phone: (801) 777-7172  
E-mail: greg.daich@hill.af.mil

**CROSSTALK**  
The Journal of Defense Software Engineering

### Get Your Free Subscription

Fill out and send us this form.

OO-ALC/MASE

7278 Fourth Street

Hill AFB, UT 84056-5205

Fax: (801) 777-8069 DSN: 777-8069

Phone: (801) 775-5555 DSN: 775-5555

Or request online at [www.stsc.hill.af.mil](http://www.stsc.hill.af.mil)

NAME: \_\_\_\_\_

RANK/GRADE: \_\_\_\_\_

POSITION/TITLE: \_\_\_\_\_

ORGANIZATION: \_\_\_\_\_

ADDRESS: \_\_\_\_\_

BASE/CITY: \_\_\_\_\_

STATE: \_\_\_\_\_ ZIP: \_\_\_\_\_

PHONE: (\_\_\_\_) \_\_\_\_\_

FAX: (\_\_\_\_) \_\_\_\_\_

E-MAIL: \_\_\_\_\_

CHECK BOX(ES) TO REQUEST BACK ISSUES:

JUL2001  TESTING & CM

AUG2001  SW AROUND THE WORLD

SEP2001  AVIONICS MODERNIZATION

JAN2002  TOP 5 PROJECTS

FEB2002  CMMI

MAR2002  SOFTWARE BY NUMBERS

MAY2002  FORGING THE FUTURE OF DEF

JUN2002  SOFTWARE ESTIMATION

JULY2002  Information Assurance

AUG2001  SOFTWARE ACQUISITION

SEP2002  TEAM SOFTWARE PROCESS

OCT2002  AGILE SW DEVELOPMENT

To Request Back Issues on Topics Not Listed Above, Please Contact Karen Rasmussen at <karen.rasmussen@hill.af.mil>.