"Any clod can have the facts, but having opinions is an art."
Charles McCabe, *San Francisco Chronicle*

Computer Science and the Pygmalion Effect

Joao Carreira and Joao Gabriel Silva, Dependable Systems Group

Several writers of this column have expressed the opinion that computer science can learn from social sciences. We now want to focus these discussions on another undesired phenomenon that commonly affects software developers. Known as the Pygmalion effect, this phenomenon has been studied intensely by social scientists but almost entirely ignored by computer scientists.

Pygmalion was a sculptor who fell in love with his statue, Galatea, a statue that was brought to life for him by Aphrodite. In the '60s, two American psychologists, Robert Rosenthal and Lenore Jacobson, used this myth to name an observation of theirs: Whenever someone evaluates something, the evaluator's expectations concerning the evaluated object influence the evaluation, in a way that tends to prove the evaluator's initial hypothesis.

Intensive studies since the 1960s have confirmed this initial observation: Virtually every evaluation process in which humans intervene is prone to the Pygmalion effect, and computer science is no exception.

THE PYGMALION EFFECT

In *Pygmalion in the Classroom* (reissued by Irvington in 1996), Rosenthal and Jacobson describe the results of several interesting experiments. In their first experiments, Rosenthal and Jacobson recruited two groups of collaborators, group A and group B. The psychologists told both groups that the experiment was aimed at testing the intelligence of mice. They asked the collaborators to administer some tests and record the results.

Editor: Will Tracz, Lockheed Martin Federal Systems, MD 0210, Owego, NY 13827-3998; will.tracz@lmco.com

The expectations of the evaluators can strongly bias the outcome of the evaluation.

Rosenthal randomly assigned a group of mice to each group. However, he told group A that their mice were genetically selected and exceptionally smart (favoring positive expectations). He told group B that their mice suffered from hereditary degeneracy (favoring negative expectations). As surprising as it may seem, the results showed that the mice of group A performed much better than those of group B.

Rosenthal observed a similar result when he repeated the experiment in a classroom with students and professors (omitting the group B's scenario for ethical reasons). When professors *expected* students to be more intelligent, the students' actual achievements corresponded with the higher expectations.

Rosenthal and Jacobson demonstrated that the expectations of the evaluators can strongly bias the outcome of the evaluation. Most of the time, they found, the evaluators perceive neither that they have any such expectation nor its enormous power.

PYGMALION AND COMPUTERS

Social scientists study the Pygmalion effect in order to reduce its negative impact in society. Computer scientists have not studied it in any depth, although we believe the Pygmalion effect is very prevalent in our profession.

Among others, software inspections and dependability and performance evaluation are fields in which the Pygmalion effect can play a significant role. In a code review, for instance, the expectations concerning the design or code of a particular individual can bias the review process. It's true that a typical code review meeting will include staff external to the project, in order to cope with human factors. At a first glance this measure would seem to mitigate the Pygmalion effect, but we wonder if it really does. Instead, we think this practice addresses the fact that we are not impartial judges of ourselves—our own perspectives of ourselves are biased. So the practice of external reviews addresses the problem of handling different perspectives, but not different expectations.

There are some important efforts to mitigate the Pygmalion effect that are worth mentioning. In some fields, the development of benchmarks to evaluate performance do address the Pygmalion effect. Examples include Spec (which measures raw processor performance), TPC (for transactional database systems), and Splash (for shared memory parallel programs). Benchmarks provide a systematic and automated process for evaluation to enable fair, unbiased comparisons.

Unfortunately, the Pygmalion effect can still show up later, in the analysis and interpretation of results—a task left to humans. Systems can also be designed with performance against specific benchmarks in mind, but this is something quite different from the unconscious Pygmalion effect.

But even supposing no maliciousness, can a well-defined process be guaranteed to totally eliminate the Pygmalion effect? The typical engineer would say yes—there's no place for subjectivity in a rigorous process. But is it really that simple?

The fact is that one group of people clearly understand the potential of the Pygmalion effect and use it intensively: the marketers. Good marketing can create positive expectations for a product and thus foster the Pygmalion effect. Microsoft's release of Windows 95 was an unparalleled example.

Unlike other aspects of information technology, which have advanced dramatically in recent years, human factors issues have not kept abreast of software development practices. Although we all seem to accept the fact that people are the most important resource in software development—and although we all seem to know that a team's success is closely linked to an ability to cope with differences in skills, attitudes, backgrounds, and education—we today still use essentially the same development methods we used 20 years ago. The Pygmalion effect is present in many computer engineering practices. It is our responsibility to find it, uncover how it is influencing us negatively, and do something about it. Any clues? ❖

Joao Carreira is a researcher and PhD candidate in the Dependable Systems Group in the Dept. of Informatics Engineering at the University of Coimbra, Portugal. Contact him at jcar@dei.uc.pt; http://dsg.dei.uc.pt/~jcar.

Joao Gabriel Silva is an assistant professor in the Dependable Systems Group in the Dept. of Informatics Engineering at the University of Coimbra, Portugal. Contact him at jgabriel@dei.uc.pt; http://dsg.dei.uc.pt/~jgabriel.

Internet Watch

Continued from page 115

When I checked the cited URLs, 40 percent were erroneous, and one bounced me to three further addresses, each time saying "Sorry, not here now." In many cases, I couldn't find the other URLs at the specified addresses and had to look elsewhere. When I did find them, information regarding the subject was indeed on the site, but I asked myself whether this was the information I was supposed to read.

I understand—and believe in—the use and importance of the Web. In fact, when doing research, the first place I search is the Web. I use the Web, as many people do, as a worldwide encyclopedia. But I believe that I am only receiving about 30 percent of the total amount of information on the subject, because I feel that people are not adding their latest research to the Web.

The Web is a useful resource, but not a complete or accurate library. Thus, I cite URLs only tentatively. It is a question of amount: the more URLs in the reference list the less I trust the paper and the lower its archival relevance. It is useful (and unavoidable at times) to cite some URLs. I know, because I do cite them. But there needs to be a limit on the amount of changeable—or in-the-future unavailable—citations.

Referencing appropriate information is important when writing a scientific paper, as references support, develop, and validate the written argument. On my bookshelf I have a copy of David Lindsay's *A Guide to Scientific Writing* (Longman, 1995), which rightly warns against referencing too many university publications, personal communications, and sources that you cannot easily locate. URLs can be easily located, but, as Tim Wooller pointed out, the information may be easily altered or removed completely from the site.

Computers and the Web are rapidly changing. The printing press moved us from hand-copying manuscripts to mass-copying them. As publishing companies devote more resources to publishing on the Web, we will see (and need) more URL citations in our journals. Our confidence in citing URLs will increase. But for the moment we need to hold back on using too many URL citations.

Jonathan C. Roberts
Research Fellow
University of Kent
j.c.roberts@ukc.ac.uk

GRATUITOUS URLS

One part of current publication practice that I find particularly annoying is the use of "gratuitous" URLs. I call them gratuitous because they serve no purpose except perhaps to indicate the writer knows what a URL is.

Gratuitous URLs provide no useful information. They should not be used as references.

Two examples can be found on page 13 of *Computer*'s December 1997 issue, in the paragraph beneath the heading "Pen technology": "When Apple Computer (http://www.apple.com)..." and "Newer PDAs, such as...3Com's (http://www.3com.com) PalmPilot..."

In cases like these, if the reader can't figure out what URL to try, providing that URL probably won't help. Besides, such a high-level URL probably dooms the user to crawling through several pages of junk to find the appropriate information anyway. This is like allowing a printed reference to a journal article to give the journal title alone, not the volume, number, and page information.

An example of this is the reference in the December issue on page 134, column 3, in parentheses: ("Composite Arithmetic: Proposal for a New Standard," Mar. 1997, pp. 65-73). Is the reader supposed to assume from the context that this is a reference to an earlier issue of *Computer*? So now we have relative links as well as absolute links for text? What happens in another context, like a reprint?

I suggest that such gratuitous URLs provide no useful information. They should not be used as references. ❖

Andy Huber
Sr. Soft. Eng. Staff
Data General
huber@dg-rtp.dg.com