

Propaganda and Software Development

Warren Harrison

I was raised in the midwestern US in the state of Missouri (natives will know that it's pronounced "Mizzoura"). Besides being the home of Harry Truman, the 33rd President of the US, Missouri is also known as the "Show Me" state. This is said to represent a certain stubbornness and devotion to simple common sense for which the population is known.



Over the past 20 years or so, I've seen countless "movements" in the software development community come and go. Structured programming, object-oriented programming, rapid prototyping, formal methods, process maturity—the list goes on. They're almost always good things in that they generally leave the field better than they found it. However, the one universal truth of these "movements" is that they're almost always not the way people are doing things right now. So, it seems that we spend as much effort trying to get people to change as we do formulating the "movement" to start with.

Convincing someone to change

Given my legacy of the "Show Me" state, it's natural for me to ask, "How can I best convince software developers to change their behavior?" My academic cousins will respond that statistics is the one true path—it should go without saying that if I can demonstrate a level of confidence of 0.01 that only a dunder-

head would seriously doubt the value of my proposal.

I say it should, because it really doesn't. When you're trying to convince someone to change the way they do things, especially if that change will be painful—and which ones aren't?—the decision to change is at least as emotional as it is rational. If it weren't, we'd have a society of nonsmoking, vegetarian joggers who never drink beyond moderation. Since this describes few places I've ever been, I can only conclude that statistics by themselves aren't adequate to effect change. In fact, based on my informal observations of life, statistics might not even be necessary.

Change agents and statistical reasoning

Many professions are built around changing people's behavior: marketing and advertising, politics, the clergy. Even traffic cops are responsible for changing the public's driving behavior. If you study how these groups operate, the first thing you'll notice is the lack of statistics or statistical reasoning. When "statistics" are used, they're little more than window dressing: "75 percent of soda drinkers prefer GinFiz Cola!" Although this might seem anathema to the hard-core scientist, the reality is that statistics appear to have little effect on people in the way they conduct their daily lives.

So what does work when you're trying to convince someone that a new technique or tool, or even a way of thinking about problems, is worth considering? We can take a tip

DEPARTMENT EDITORS

Bookshelf: Warren Keuffel,
wkeuffel@computer.org

Construction: Andy Hunt and Dave Thomas,
{andy, dave}@pragmaticprogrammer.com

Design: Martin Fowler,
fowler@acm.org

Loyal Opposition: Robert Glass,
rglass@indiana.edu

Open Source Software: Christof Ebert,
christof.ebert@alcatel.com

Quality Time: Nancy Eickelmann,
nancy.eickelmann@motorola.com,
and Jane Hayes, hayes@cs.uky.edu

Requirements: Suzanne Robertson,
suzanne@systemsguild.com

STAFF EDITORS

Senior Lead Editor
Dale C. Strok
dstrok@computer.org

Group Managing Editor
Crystal Shif

Senior Editors
Shani Murray and Dennis Taylor

Staff Editor Assistant Editor
Rita Scanlan Rebecca Deuel

Editorial Assistant
Brooke Miner

Magazine Assistant
Hilda Hosillos, software@computer.org

Art Director
Toni Van Buskirk

Cover Illustration Technical Illustrator
Dirk Hagner Alex Torres

Production Editor Production Artist
Monette Velasco Carmen Flores-Garvey

Executive Director
David Hennage

Publisher Assistant Publisher
Angela Burgess Dick Price

Membership/Circulation Marketing Manager
Georgann Carter

Business Development Manager
Sandra Brown

Senior Production Coordinator
Marian Anderson

CONTRIBUTING EDITORS

**Robert Glass, Cheryl Baltes,
Anne Lear, Molly Mraz, Joan Taylor**

Editorial: All submissions are subject to editing for clarity, style, and space. Unless otherwise stated, bylined articles and departments, as well as product and service descriptions, reflect the author's or firm's opinion. Inclusion in *IEEE Software* does not necessarily constitute endorsement by the IEEE or the IEEE Computer Society.

To Submit: Access the IEEE Computer Society's Web-based system, Manuscript Central, at <http://cs-ieee.manuscriptcentral.com/index.html>. Be sure to select the right manuscript type when submitting. Articles must be original and not exceed 5,400 words including figures and tables, which count for 200 words each.

from the military and political parties and explore the idea of propaganda.

Propaganda

The *American Heritage Dictionary* defines propaganda as “the systematic propagation of a doctrine or cause or of information reflecting the views and interests of those advocating such a doctrine or cause.” Although the term has an unnecessarily pejorative tone, it’s what we’re really doing when we’re trying to change someone’s behavior. I need to emphasize that when I point out something is being used as “propaganda,” it shouldn’t be viewed in a pejorative light. Given this definition, virtually every technical advance requires a certain degree of propaganda for its use to move beyond the original inventors.

One of the leaders in propaganda and its dissemination is the US military. They spend significant time studying what works and what doesn’t and have identified a number of effective propaganda techniques (*Psychological Operations Field Manual* No.33-1, US Army, Aug. 1979). While the list is extensive, I’ve selected a few representative examples to discuss in the context of the software development community and its “movements.” Perhaps you’ll find some of these techniques useful to gain acceptance for your favorite “movement.”

The Bandwagon

Bandwagon appeals try to persuade the target audience to take a course of action that “everyone else is taking.” For instance, a 1995 report published by the Software Engineering Institute (*After the Appraisal: A Systematic Survey of Process Improvement, Its Benefits, and Factors That Influence Success* by Dennis R. Goldenson and James D. Herbsleb, tech. report CMU/SEI-95-TR-009) starts out with “The Capability Maturity Model (CMM) ... has had a major impact on software organizations throughout the world. ... [T]he CMM is used as a reference model to guide software process improvement (SPI) efforts in many hundreds of soft-

ware organizations. Initially adopted by defense organizations and their contractors, it is now used in organizations both large and small throughout the software industry.”

This is certainly not untrue. It simply presents something in its best light. However, it’s clearly an example of inviting readers to “jump on the bandwagon.”

Obtain Disapproval

This approach is used to influence the audience to disapprove of an action or idea by suggesting the idea is popular with “groups hated, feared, or held in contempt by the target audience.” We find this most often in our field when new techniques are compared to existing techniques. If we can somehow associate the existing way of doing things with classes or groups that the readers might ridicule or dislike, it’s easier to convince them the new approach is preferable. For example, one statement found in the Agile Manifesto’s history Web page (www.agilemanifesto.org/history.html) is “taken as a whole, the practices define a developer community freed from the baggage of Dilbertesque corporations.” This is perhaps the epitome of the Obtain Disapproval technique within the software development field.

Is such a statement inappropriate? Certainly, organizations have their share of problems in adopting and using heavy process development models—especially for smaller projects where process infrastructure outstrips actual software production effort. While perhaps this is an overly colorful way of putting it, agile methods were developed as an alternative to process-heavy development. And it’s not only perfectly reasonable to point this out but is probably an important observation. Nevertheless, it’s propaganda in its purest form.

Glittering Generalities

This approach involves emotionally appealing words so closely associated with highly valued concepts and beliefs that they carry conviction without supporting information or reason. They

ask for approval without examination of the reason. Often this is achieved through *simplification*. This technique reduces the many facts of a situation to “right or wrong.” By doing this, the propaganda provides simple solutions for complex problems. To achieve this, the technique offers “simplified interpretations of events, ideas, or concepts.” To be effective, “statements [must be] positive and firm; qualifying words are never used.”

To illustrate the use of this type of propaganda, one to-remain-nameless report from a well-known member of the software development “movement” community asserts, “To improve product quality, you must improve process quality ... doing this requires you to measure and track process quality.” Clearly there are many facets to this observation (and to be fair, the report later goes on to explore some of these). For instance, what does product quality and process quality have to do with each other? For many years now, this connection has been taken on faith and “reasoning.” But I’ve seldom seen a detailed explanation of this. Is it true that we can’t have a quality product using a poor-quality process (and what exactly is a poor-quality process)?

Note that positive, firm statement: you must improve process quality to improve product quality. And to improve process quality, you must measure and track the process. Obviously, there’s no room for doubt in this author’s mind. But what if your process is okay right now; will measuring and tracking it improve product quality? What if it’s “almost” okay? Will the gain in product quality offset the cost of the additional measurement and tracking?

The *Army Psychological Operations Field Manual* points out that simplification has the following characteristics:

- *It thinks for others.* Often people will accept information they can’t verify personally as long as they consider the source an authority. Others might be too busy to think

things through or are uneducated and willingly accept convenient simplifications.

- *It’s concise.* Simplification seems to go directly to the heart of the matter, so the propaganda’s target might not even consider that there could be another answer to the problem.
- *It builds ego.* Some software developers are reluctant to believe that areas outside their own specialty are difficult to understand. For instance, a colleague who specialized in programming languages told me that there was no need to hire software engineering experts to teach software engineering courses because “anyone that has ever written a large piece of software should be qualified to do so ...” Yet this same colleague would probably have a heart attack if I expressed interest in teaching an advanced compilers course. Simplifications reinforce the ego of targets whose expertise is in other areas because they reinforce the idea that these techniques or approaches aren’t actually beyond their understanding.

The tool works

These are just a few examples of well-known propaganda techniques. Innovators who want to get their work accepted by others would be well advised to study propaganda as the military, political parties, and even corporations use it. While we have come to associate propaganda with nefarious mind-control plots and political extremists, it’s really just an approach to convincing others to see things the way you do.

What do you think?

I’d like to hear from readers about your thoughts on this issue. Do you recognize the propaganda aspects of any current or past “movements” within the software development industry? Have you found propaganda important in fostering change in your organization? Let me know. Please write me at warren.harrison@computer.org. ☺

EDITOR IN CHIEF

Warren Harrison

10662 Los Vaqueros Circle
Los Alamitos, CA 90720-1314
warren.harrison@computer.org

EDITOR IN CHIEF EMERITUS:
Steve McConnell, Construx Software
stevemcc@construx.com

ASSOCIATE EDITORS IN CHIEF

Education and Training: Don Bagert, Rose-Hulman Inst. of Technology; don.bagert@rose-hulman.edu
Design: Philippe Kruchten, University of British Columbia; kruchten@ieee.org
Requirements: Christof Ebert, Alcatel; christof.ebert@alcatel.com
Management: Ann Miller, University of Missouri, Rolla; millera@ece.umar.edu
Quality: Stan Rifkin, Master Systems; sr@master-systems.com
Experience Reports: Wolfgang Strigel, QA Labs; strigel@qalabs.com

EDITORIAL BOARD

Nancy Eickelmann, Motorola Labs
Richard Fairley, OGI School of Science & Engineering
Martin Fowler, ThoughtWorks
Jane Hayes, University of Kentucky
Andy Hunt, Pragmatic Programmers
Warren Keuffel, independent consultant
Karen Mackey, Cisco Systems
Deependra Moitra, Infosys Technologies, India
Don Reifer, Reifer Consultants
Suzanne Robertson, Atlantic Systems Guild
Richard H. Thayer, Calif. State Univ. Sacramento
Dave Thomas, Pragmatic Programmers

INDUSTRY ADVISORY BOARD

Stephen Mellor, Mentor Graphics (chair)
Dave Aucsmith, Microsoft
Maarten Boasson, Quaerendo Invenietis
Robert Cochran, Catalyst Software
Annie Kuntzmann-Combelles, Q-Labs
David Dorenbos, Motorola Labs
Enrique Draier, MAPA LatinAmerica
Dehua Ju, ASTI Shanghai
Tomoo Matsubara, Matsubara Consulting
Dorothy McKinney, Lockheed Martin Space Systems
Bret Michael, Naval Postgraduate School
Susan Mickel, Lockheed Martin
Dave Moore, Vulcan Northwest
Melissa Murphy, Sandia National Laboratories
Grant Rule, Software Measurement Services
Girish Seshagiri, Advanced Information Services
Martyn Thomas, Praxis
Laurence Tratt, King’s College London
John Vu, The Boeing Company
Simon Wright, Integrated Chipware
Jeffrey Voas, Cigital

MAGAZINE OPERATIONS COMMITTEE

Bill Schilit (chair), Jean Bacon, Pradip Bose, Doris L. Carver, George Cybenko, John C. Dill, Frank E. Ferrante, Robert E. Filman, Forouzan Golshani, David Alan Grier, Rajesh Gupta, Warren Harrison, Mahadev Satyanarayanan, Nigel Shadbolt, Francis Sullivan

PUBLICATIONS BOARD

Michael R. Williams (chair), Michael Blaha, Mark Christensen, Roger Fujii, Sorel Reisman, John Rokne, Bill Schilit, Linda Shafer, Steven L. Tanimoto, Anand Tripathi