*Software Quality Journal* **Vol. 13 No. 3, 2005**

# Editorial: Is Anyone Listening?

Software development disasters continue to make the headlines. Generally, the most highly publicized fiascos are in systems developed to support government functions. Such failures can harm the public, and the failures of systems developed with taxpayer money cannot be as readily hidden from public view. What I find appalling is that many if not most of the disaster could have been prevented if good software assurance practices had been applied.

The most recent software disaster to really irk me affects people close to my home. The dysfunctional system is called the Colorado Benefits Management System (CBMS), a system intended to realize the "dream of improving access to public assistance and medical benefits for clients" [1]. This system is supposed to help those in society with the greatest need: the poor, ill, and disabled.

According to the Colorado Department of Human Resources, "the CBMS has three major goals:

1. Replace antiquated, inflexible legacy systems that use outdated programming languages, are difficult to modify, and perform redundant processing in individual 'stovepipe' systems.
2. Enable a universal worker who will not have to memorize rules and look through manuals to determine eligibility and will have single point of data entry for multiple programs.
3. Improve access to public assistance and medical benefits by providing one-stop shopping for clients, permitting faster eligibility determinations, and allowing for higher accuracy and consistency in eligibility determinations statewide." [1]

Yet, when the CBMS system went online, access to public assistance in Colorado virtually disappeared. An "explosion in hunger across the state is the direct result of problems with the state's $200 million Colorado Benefits Management System, a nightmare computer-software program that has thrown food stamps, Medicaid, Old Age Pensions and other relief services into chaos since it was brought online Sept. 1" [2]. By the end of September there were "9,000 backlogged food-stamp applications in five metro-Denver counties" and "16,000 backlogged requests for family medical assistance and Temporary Assistance to Needy Families" [3].

Let's look at some of the numerous root causes of the disaster [4]:

1. A case-worker must open 17 different screens to process just one case. It appears that there was little or no consideration of the system's usability by the developer; usability could not have been tested.
2. The system ran so slowly that it timed out before data entry could be completed. Clearly, there little stress testing was conducted, and functional testing was inadequate.

3. There was no mechanism available to rollback to the prior "obsolete" system that was in use before the CBMS was put on line. A risk assessment is likely to have shown the need to have a rollback mechanism.
4. There was a strong suspicion that the software contractor used inexperience developers who were overworked, and not fully qualified.

These problems could have been prevented, along with a great deal of human suffering, if only professional software assurance practices were employed. But, as in far too many software development projects, software assurance was ignored until the only possible action was disaster recovery.

Failures like this one demonstrate a lack of professionalism as well as a lack of ethics. Shortcuts were taken with little regard for the consequences. Perhaps the development contractor negotiated a contract that allowed the delivery of such a shoddy product. If so, then there was clearly a lack of professional ethics. In this case, the most helpless members of the community are the ones to suffer.

I must remind readers that this is just one example, and disasters of this kind occur in commercial systems. However, when commercial systems are shoddy, the details are usually hidden to protect the guilty. Public systems, like the CBMS are exposed to great publicity, so that I can talk about them.

The bottom line is that professional software development contractors often behave like amateurs. There is often little regard for professional practice, and organizations deliver a system with the minimum level of quality that they may get away with. Such practices are contemptible, and reflect poorly on the entire profession.

The software assurance community has loudly promoted the use of numerous techniques to verify that systems will perform as required, but is anyone listening?

James Bieman
Fort Collins, Colorado
U.S.A

## References

[1] Colorado Benefits Management System (CBMS) Home Page
http://www.cbms.state.co.us/

[2] Diane Carman. "Benefits backlog gets worse." *Denver Post.* Tuesday, February 22, 2005.

[3] Bill Scanlon. "Computer glitch costly: taxpayers stuck with tab as welfare programs stumble." *Rocky Mountain News.* September 29, 2004.

[4] Glen Emerson Morris. "Lessons from the Colorado Benefits Management System Disaster." *Advertising & Marketing Review*. October 2004.
http://www.ad-mkt-review.com/public_html/air/ai200411.html