TITLE: "WHEN YOU HIT A WALL, GO AROUND IT"

by Tim Bryce
Managing Director
M. Bryce & Associates (MBA)
P.O. Box 1637
Palm Harbor, FL 34682-1637
United States
Tel: 727/786-4567
E-Mail: timb001@attglobal.net
WWW: http://www.phmainstreet.com/mba/
Since 1971: *"Software for the finest computer - the Mind"*

Years ago we were hired by a Blue Cross/Blue Shield plan to look over a new Claims Processing system they were building. The focal point of their problems centered on adjudicating claims whereby they wanted to devise an automated way to analyze a claim and determine the amount of money to be paid out. They had spent a lot of time and money analyzing adjudication and were frustrated they couldn't come up with a standard algorithm for computing all claims. We studied the problem and found that 90% of their claims were easy to analyze and calculate adjudication. For example, simple doctor visits, a broken bone, normal childbirths, etc. were easy to analyze and compute. However, unusual medical claims such as complications at childbirth, and accidents from a massive car accident, involved many more variables and, consequently, were difficult to compute based on standard algorithms. After studying the problem carefully, we reached the conclusion that trying to accurately calculate 100% of all claims was an impossibility. It was simply not practical to try to achieve this lofty goal and, as such, was a waste of time pursuing it. Instead, it was our advice they simply automate the 90% claims they could easily perform and segregate the remaining 10% for handling by a human adjustor. To their surprise, this worked remarkably well and saved them considerable money.

Too often in systems and software development we try to do the impossible and often run into a stumbling block when trying to achieve our goal. Do we continue to waste time and money on a problem that cannot be conquered or do we stop, lick our wounds, and move around? The problem is knowing when to stop. As "Dirty Harry" once said, "*A man has got to know his limitations.*"

Let me give you another example. Years ago, we devised our own set of in-house programming standards. These standards were used in Phase 4-II of "PRIDE"-ISEM and allowed us to engineer and review a program before coding. We then took it another step by creating software that would read the program's specifications and generate the initial source code. We called it a "Program Shell Generator" for it generated the lion's share of the code (be it COBOL, C, or any other language). It could generate 100% of the code for simple programs, but we recognized from the outset it couldn't do everything. Instead, it would generate approximately 80% of the code which the programmer would then have to complete. Some would say such a generator would be a colossal waste of time. Far from it, we found it to be a tremendous time saver. Instead of wasting time setting up the initial code, the programmer was free to concentrate on the 20% of the code requiring their attention. Other program generators are faced with the same reality; they can generate a lot of code, but probably not 100% for any major application of any substance.

It is important that Project Managers and senior analysts be wary of such potential roadblocks and not try to conquer the impossible. Instead, look for practical solutions. In other words, don't try to drive into a wall, turn on your turn signals and go around it.

END

"PRIDE" Special Subject Bulletins can be found at the "PRIDE Methodologies for IRM Discussion Group" at:

http://groups.yahoo.com/group/mbapride/