TITLE: "WHAT PRICE QUALITY?"

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@phmainstreet.com
Yahoo! IM: littleleaguerng
WWW: http://www.phmainstreet.com/mba/
Since 1971: *"Software for the finest computer - the Mind"*

"Quality must be built into the product during design, not inspected in afterwards."
- Bryce's Law

INTRODUCTION

We now live in a fast paced society where we expect products and services to be delivered rapidly (some say "yesterday"), cheaply, and with a high degree of quality. This is particularly true in the systems and software industry. If we lived in a perfect world, systems and software would be developed rapidly and inexpensively, they would effectively satisfy business needs, and would be easy to maintain and modify. There is only one problem with this scenario: it is a fantasy. In reality, we live in a "disposable" world where systems and software are slapped together in the hopes everything will hold together and will pacify the end-user for the moment. Some people believe striving for a Utopian world is an impossibility and, as such, resign themselves to rewriting systems and software time and again as opposed to designing them to be industrial strength.

Improving speed in the development process is relatively simple to accomplish; e.g., the plethora of programming tools available. But adding quality into a product is something entirely different. From the outset we must recognize that quality doesn't come naturally to people anymore. Back when there was a sense of craftsmanship, quality was rarely a problem. This is back when people identified with their work products, and strove to seek perfection as it was a reflection of their character. Corners were not cut and products were made to last. Unfortunately, we no longer live in such times and people tend to disassociate their work from their personal lives. Further, the speed and sophistication of our tools leads us to believe we are producing quality products. The reality is that our tools are only as good as the people using them, not the other way around.

A PERFECT WORLD

How one person perceives quality may be entirely different than another's. This is because we tend to have different perspectives in how to build something, e.g., whereas one person may build a product one way, another may build it using an entirely different approach. This means products are commonly built using inconsistent methods. Let me give you some examples:

- If we lived in a perfect world, we would have a standardized approach for defining requirements, thereby everyone would be operating with a standard approach for scrutinizing requirements. But the reality is our approach to requirements definition is redefined with each development project, thereby making it impossible to validate requirements with any consistency.

- If we lived in a perfect world, developers would be working with standard data definitions that would include validation/editing rules, among other things. This would result in a consistent approach in the use of data (aka "Data Cleanliness") and would promote system integration through data sharing. But the reality is that each programmer specifies the use of data (including its physical characteristics and validation/editing rules) on a program by program basis, thereby defeating the opportunity to share and reuse data in a consistent manner. Even worse, implementing changes on a consistent basis is difficult at best (e.g., the Y2K problem).

- If we lived in a perfect world, programs would be designed in a standardized manner so they may be easily modified or maintained by any other programmer at a later date. But the reality is that programs are written based on the personal nuances of the programmer, making it next to impossible to maintain or modify by another person. Consequently programs are discarded and rewritten.

- If we lived in a perfect world, developers would adhere to a standard and consistent approach (methodology) whereby uniform work products could be produced and reviewed, thereby improving communications among the staff and allowing for the interchangeability of workers in the development process. But the reality is, the development process is defined on a project-by-project basis, thereby uniformity and interchangeability is defeated.

(continued on page 2)

(continued from page 1)

The reality is we live in an imperfect world. What would appear to be obvious approaches to development seldom occurs in most systems and software shops. It is simply unnatural to developers who prefer to operate independently as opposed to adopting a shop standard. This of course means development organizations tend to "reinvent the wheel" with each project.

Because of such inconsistencies, the only option for improving quality is to try to inspect the product after it has been built, not during development. Under this approach, inspection is complicated as each person has designed the product according to their own personal interpretation of development, not as a standard body of work.

BUILDING QUALITY INTO THE PRODUCT

It is obviously cheaper and more sensible to arrest a product defect early during development as opposed to trying to catch it afterwards. To do so, the development process has to be subdivided into defined units of work specifying what is to be produced (work products, aka "deliverables"), how it should be produced (using accepted tools and techniques), and its acceptance criteria (including review points). Such a work environment is in sharp contrast to "The Black Hole" approach used by most organizations today; e.g., requirements are fed into an unknown development environment and the resultant product is inspected afterwards. This approach concentrates only on the final deliverable and not on the overall process by which the product is to be developed. By the time the final product is produced, it may be unrecognizable to the user and the project may have exceeded estimated cost and schedule. Even worse, the product may have to be redesigned and rewritten over and over again. Interestingly, this is the approach advocated by today's "Agile" proponents.

In other manufacturing practices, the definition of the work environment is the responsibility of an Industrial Engineer who defines the units of work in the development of a product (assembly line), the standard tools and techniques to be used, the work products, and the acceptance criteria. Although the concept of Industrial Engineering is applicable to systems and software, few development organizations are familiar with the concept.

THE PRICE OF QUALITY

Regardless of what you call it, Industrial Engineering or Quality Assurance, quality requires a dedicated group of people to define the overall development process, monitor progress, and constantly research new ways to improve it (tools and techniques). This does not mean quality is the sole responsibility of such a group. It is not. Quality is the responsibility of every person involved in the development process. The group simply provides leadership in this regards.

In terms of costs, the truth is that quality is free (as the likes of Philip Crosby have pointed out to us). True, it requires an outlay of money upfront to embark on a quality assurance program, but this will be offset by reduced costs later on in terms of reduced development time and fewer defects requiring rework. By having everyone working according to defined processes and work products, errors are caught and corrected early in the development process. Further, work products are easier to maintain and modify later on, this specifically includes systems and software. Such a program, therefore, does not add overhead to the development process, it reduces it.

To make this work though requires commitment from management and herein lies the rub. As I mentioned earlier, we live in fast-paced times. Implementing an effective quality assurance program takes time to cultivate, it cannot be installed overnight. There is more to it than mechanics; standards have to be devised, attitudes have to be adjusted, consciousness' raised, etc. In other words, it is the people-side of quality that takes time to mature and become ingrained in the corporate culture. As such, a quality assurance program requires management vision and long-term commitment to see it come to fruition. This is difficult to sell to managers who have trouble thinking past the next financial statement. But if executives understand that a company truly runs on systems and software, then they will be more amenable to investing in industrial strength applications.

CONCLUSION

Its interesting, the systems and software industry is one of the few industries that resists standardization as opposed to embracing it. Standardization is an inherent part of any quality program. It means devising and applying craftsman-like rules in the development of a product or service. Such rules substantiates completion of work in a prescribed sequence and is measurable. And maybe it is this kind of accountability that developers resist.

Some developers even go so far as to question the necessity of a quality assurance program since many companies rewrite their systems and software year after year. Maybe they are right, but I tend to see this as a defeatist

(continued on page 3)

(continued from page 2)

attitude, that we can do nothing more than produce mass mediocrity. I believe we can do better. But to do so, we need to invest in ourselves and our future. Remember, you must first plant the seeds in order to harvest the crop. Unfortunately, most companies tend to eat the seeds and then there is no crop to harvest. Somehow I am reminded of the old expression, *"You can pay me now or pay me later, but you're going to pay me."*

For additional information on "PRIDE" Quality Assurance, see:

http://www.phmainstreet.com/mba/pride/spqa.htm

END

About the Author

Tim Bryce is the Managing Director of M. Bryce & Associates (MBA) of Palm Harbor, Florida and has 30 years of experience in the field of Information Resource Management (IRM). He is available for lecturing, training and consulting on an international basis.

"PRIDE" Special Subject Bulletins can be found at:

http://www.phmainstreet.com/mba/mbass.htm

They are also available through the "PRIDE Methodologies for IRM Discussion Group" at:

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

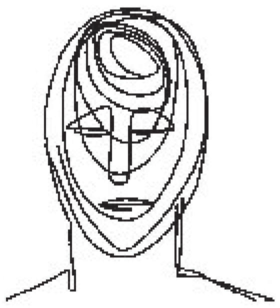
You are welcome to join this group if you are so inclined.

The "Management Visions" Internet audio broadcast is available at:

http://www.phmainstreet.com/mba/mv.htm

Also, be sure to read Tim's Blog at:

http://blogs.ittoolbox.com/pm/irm/



Since 1971: *"Software for the finest computer - the Mind"*