The International Journal of Applied Management and Technology, Vol 3, Num 2

Why Managers Should Become Better Acquainted With Programming Issues, Web Source Code, and Technology

Alireza Ebrahimi
State University of New York
College at Old Westbury, USA

Abstract

There has been a controversial debate over how well managers need to acquainted with the area of information technology, especially about what managers need to know of programming and understanding of web source code. Information is an indispensable key to business success nd Information Technology (IT) facilitates the ability of business to use information to its advantage. It is the programs (software) that drive technology, the web page presents the business enterprise to everyone. In the heart of every business there also are managers responsible for keeping the records of performance, and using the records to promote efficiency, discipline, and to get the job done. Today's in-depth involvement of technology in the marketplace has created a strong correlation between IT and managers. Over time, the bond between a manager and IT is becoming stronger, signaling the need for more interaction and understanding of both technical capabilities and business goals on the part of managers. This paper posits that managers need to know more about information technology and ctively participate in the IT decision-making team. Furthermore, it suggests how managers can have incrementally to acquire some elementary knowledge of programming issues, what shows on the web, what is behind the web (source code), and what is placed on the server (database). In a dynamic market where changes are made in nanoseconds. It is rewarding and may soon become almost mandatory that a multi-talented manager will have to deal with change, to add or supervise addition of pragmatic programming, and web update the web in order to be competitive. Key to the process is the increasing communication and interaction between managers and programmers. The benefit of understanding the elementary steps of programming and web technology is that it will allow a manager to play a larger role in communicating and delegating responsibility with confidence, and competence thus leading to cost reduction and better short- and long-term risk management.

Keywords

Technology, Managers, Programming, Web, Source Code, Manager Update, Crisis and Chaos, Virus, Y2K, E-Business, Software Engineering, Risk Management.

Programming Argument

In 1982, Robert Benjamin forecasted the state of IT in the year of 1990 stating that all aspects of software will improve steadily, and the demand for software will be so great as to appear infinite (Benjamin,1992). Now, nearly fifteen years later, we are experiencing the fulfillment of these critical IT predictions, and the far more fundamental knowledge and coordination of managers and the programmers they must direct.

Why is it crucial for a manager to keep up to date with programming issues and web technology? One may assumes it is not part of a manager's job description since programming is associated with complex theories, mathematics, and gibberish code. Furthermore many also assume there are mathematical formulas and theoretical concepts are involved in creating a webpage (front end) or in a web server program (back end)? However the technology and its programming have become less cumbersome than a decade ago, the problems can be easily worked around by some explanation. We do not suggest that managers become programmers who know details of syntax and semantic jargon of each programming language construct. Rather they should become aware of the fundamental changes in programming and be able to identify simple programming concepts such as input/output, decision-making, repetition, and file handling. Managers need to be updated, to identify new simple concept of how to use programming, communicate with programmers employed by their corporations and how to take advantage of web technology. Similarly the programmers and web page specialists need to have a through idea of the purpose production and Modus vivendi of the company. Furthermore, a manager may be able to visualize a concept as to how it is used or should be used. According to the U.S. Bureau of Labor Statistics, there are roughly 100 million information users in the U.S. More interestingly, Sutcliffeis estimates that by 2005 in the U.S. alone, there would be 55 million end-user developers in addition to 2.75 million professional software developers. From the above estimate, one can conclude that one out of five people in the U.S. population (295 million) has to deal with programming issues and write some sort of program. Sutcliffe and Mehandjiev describe that End-user development is about taking control - not only of personalizing computer applications (end-user computing) and writing programs, but of designing new computer-based applications without ever seeing the underlying program code (Sutcliffe, 2004).

Programming Simplicity

A program is a set of instructions telling the computer what to do. Programming is straightforward with three foundations: Sequence, Decision-making, and Repetition, all known as control flow. Each instruction in a control flow interacts with the memory bank for storing, recalling, and modifying one value at a time. The simplicity of a program's control flow can be explained by comparing it to the flow of water cascading down step by step like a waterfall. A waterfall that starts at the top and flows directly to the bottom is sequential control flow, with only one path. If the water flow reaches a point where it can go either one way or the other, this point represents a decision. If the water flows back to a point of origin and cycles, this is repetition. Before each cycle begins, there is a decision on whether to repeat the cycle or to exit. At each step of the flow there is interaction with the memory and, possibly, with the user. With this limited programming knowledge which includes: putting instructions into order; input/output; making decisions; repetition; and file handling, one can accomplish what is necessary you need to do and better communicate (Ebrahimi, 2003). Given a limited knowledge of programming, how have senior managers and executives been able to show competence in understanding the process of programming? What would be the role of the managers in this? Most senior managers and executives don't understand software because they haven't had the experience of direct involvement in a software development project, however when they went through the replication during an experiment they were able to gain a sense and feeling of how a software product is created and how it interacts. Armour has found it "fun and interesting" to see executives code. (Armour, 2004).

Programming and Language Problems

Although programming languages have changed only slightly over the last fifty years, the textual representation of control flow makes it hard to follow the course of a program and understand what is happening. A program in execution does not necessarily follow the order of the program written by a programmer. Similarly, arcane notations used for language constructs are contributing to programming errors themselves. Some new language constructs are more confusing, unfriendly compared to their predecessors and these new constructs are justified based on personal preference. The bottom-line is that not much has changed with regard to programming and its languages. Is this good or bad news? For those who believe in technology and change, it is bad news. The criticism that programming and programming languages have fallen behind the technology they have created leads one to wonder whether the programming can meet the needs of the unit it is supposed to streamline and improve. The good news is that due to the relative stability of programming and languages, it is possible to encourage managers to take advantage of the situation and deal with it. The problems of arcane notations and programming errors still persist and there is hope that a big change will eventually come (Ebrahimi, 1992). Yet, to be "ironed out" the wrinkles and quirks need to be worked out by the team of managers of businesses and programmers and web workers.

Technology Crisis and Some historical Lessons Software Engineering

After the introduction of Integrated Circuits (IC’s), programmers became free from dealing with the small size memory associated with transistors. Programmers soon could write many large programs without much restriction. With this overabundance of programming, the situation went so out of hand it caused chaos. As a result, there was a call for a NATO conference in Europe in 1968 at which the term Software Engineering was coined. Several resolutions dealt with the software crisis, and on the positive side, the software crisis led to the deployment of Software Engineering paradigms. Y2K Compliance Just a few years ago, everyone can recall that we dealt with another chaos that put many managers under tremendous stress and caused the firing of many. The worldwide problem known as Y2K was both a programming problem and a managerial problem: We could not The International Journal of Applied Management and Technology, Vol 3, Num 2 represent the year 2000 with 00 since 1900 was represented that way. What were we to do? Should we write a new program to change all the data from two digits to four? Should we set a flag for the new data? Yet another problem with Y2K is the year itself, 2000 which was a leap year that was divisible by 400 (not every 4 years is a leap year, e.g. 1900). Y2K made programmers with no managerial training run the show, thus taking over managerial duties. Programmers instructed managers on what to do, often in a compressed period of time, which seemed to challenge the authority and intelligence of managers. Now is not the time to refresh our mind as to whose fault Y2K was, or why managers waited so long to realize that a problem would occur. No one really knows exactly how much time was spent to deal with this crisis, either directly or indirectly. In addition, the Y2K problem and its aftermath was calculated to cost trillions of dollars. We are still paying for it. The trillion plus dollars have been spent by businesses on Y2K compliance, liabilities, and lost productivity due to computer crashes. Managers had to rethink and set survival strategies for the future. What will be the next technology crisis? Should we wait for problems to occur and then tackle them? Why was the Y2K issue not addressed at an earlier stage? Even if Y2K wasn’t preventable, involvement of executives and managers with programmers at a far earlier stage would have reduced the overall cost of Y2K. (Braithwaite, 2000).

Web Chaos, Spam, Spim, Spit

The web has become the platform for the e-market. Every business strives to become web-oriented. What is going to happen if everyone is web-oriented and has a link to everyone else?

Let us examine the worst-case scenario where X e-businesses is connected to many other M businesses and all the M businesses are connected to the X businesses (many-to-many relationship). In addition, each of these businesses is linked to many nodes of their own. When one system sends one request after another to other systems, there will be a point when too many requests to a system cannot be handled. If we are not experiencing these difficulties yet, let us simulate a possible scenario with web growth, a kind of fantasy of denial that one's system is immune and that everything possible and necessary has been done to avoid problems. But smooth internet use requires a sort of defensive driving where laws and regulations need to be agreed and acted upon. For example making massive number of phone phone calls using the internet instead of traditional phone lines can lead to chaos, which would hinder everyone. Less than the potential for chaos, businesses, no less than individuals, need to look at the negative side of e-marketing. Spam (unwanted mail, advertisement and messages) is organized and directed by a program that generates enormous volumes of traffic and often hides its original source. Similarly, unwanted instant messaging can be generated (SPIM) and Spit- Spam over Internet. Obviously, these Internet abuse (Spam, Spim and Spit) leads to waste of business time, waste of space in memory, and more importantly, it aggravates the users and often creates emotional distress in the workplace (Vinton, 2005).

Virus Catastrophe, Phishing, and Trojan Horses

How would many managers react if an employee says, “MY MOUSE HAS A VIRUS!”? Several probably would look at the person and politely ask, “Are you for real? A mouse cannot get a virus.” Then s/he may look at the mouse on the desk, click it or roll it to see if the ball is stuck. One common problem with the mouse in the good old days was the malfunctioning of the driver program due to interference by some other programs. However, how many managers are savvy enough to realize a virus could also corrupt a mouse drive. How many know that a mouse becomes intelligent and sophisticated with its own programming? A virus catastrophe requires a manager to understand what a virus is and how to handle it. A virus is a program that can infect other programs or data stored in a file and it can knock down your IT (Cohen, 1994). Just one virus “LOVE BUG” inflicted an estimated 10 billion dollars damage in only a few days. Today, a malicious program can act as a real web site and collect information such as passwords, and ordinary computer user cannot distinguish between the fake web site and a real web site. Similarly, through Trojan horse, which has been placed in the user’s computer, the passwords of other systems can be accessed. Managers should be able to observe the size of programs and the data their companies have stored away for use. There is no doubt that a manager’s knowledge of programming will lead to better decision-making that will prevent errors. In case errors occur, managers will be able to understand the error, reports, recover lost files and be able to communicate with programmers in order to resolve failure and defects, such as viruses (Highland, 1997). A little understanding of programming can enable managers to take proper protection measures for programs and data, such as writing to files for back up and reading from files for recovery.

The Benefit and Impact

Because a manager deals with people, quality, and planning, the decisions managers make with regard to information technology can play a crucial role in the day-to-day affairs and also have a consequential impact on the business. It is not enough for a manager to be an end user or a computer operator. Managers also should actively participate in the information technology (IT) decision-making team with programmers and other technical personnel to decide how IT is configured and customized to its organization's business needs. This requires managers to have knowledge of programming foundations, and requiring programmers to have a general idea of company operations, in order to efficiently for each application. With a little routine update, from both a regularized task should become hassle-free. Nowadays, up-to-date knowledge of computer technology can give a manager the skills to use the competitive advantage gained by looking at other organizations' levels of information flow, marketing strategies and more. Thus a well-prepared manager could see the code of other business websites by right-clicking the mouse and selecting View Source. With the increasing demand of the e-market, firms will need managers who understand the technology, and are able to deal with the possibility of chaos and uncertainty without interrupting business progress. Programming and its operating issues may determine the success for some and root of failure for others. Managers will need skills to be able to avoid crisis, to manage several kind of crisis, recognize a crisis, contain and resolve it, possibly even profit from the crisis (Augustine, 1995). A manager who is able to handle a simple programming task can better identify the real cause of problems rather than speculate about it. We conclude that motivating managers to understand and to participate in programming issues will result in most, if not all, of the following: higher quality of performance, better time management, reductions in cost and risk, improved morale, greater respect from subordinate employees, and, finally, becoming a marketable manager.

1. **References**

Armour, P. (2004). When Executives Code, Communications of ACM, 47,1, 19-22.
Augustine, N. (1995). Managing the crisis you tried to prevent, Harvard Business Review, 73, 147-159.
2. Benjamin, R. & Blunt, J. (1992). Critical IT Issues: The next ten years, Sloan Management Review; Summer, 33, 4.
3. Braithwaite, T. (2000). Y2K Lessons Learned: A Guide to Better Information Technology Management. New York: John Wiley.
4. Cerf, V (2005). Spam, Spim, and Spit, Communications of ACM, 48, 4, 39-43
Cohen F. (1994). A Short Course on Computer Viruses, New York: John Wiley.
5. Ebrahimi, A. (2003). C++ Programming Easy Ways (Volume I & II). Boston: American Press.
6. Ebrahimi, A. (1992). VPCL: A Visual Language for Teaching and Learning Programming, Journal of Visual Languages and Computing, 3, 299-317
7. Highland, H. (1997). Procedures To Reduce The Computer Virus Threat, Computers and Security, 16,5, 439-449.
8. Sutcliffe, A. Mehandjiev, N. (2004). End-user development: Introduction, Communications of
the ACM, 47,9, 31-32.