

Dynamic Software Productivity is still available

Given that it is estimated, in the total life cycle cost of any software, 80% of that cost is attributed to maintenance and 20% to implementation. It is imperative that "clearly understandable" COBOL coding should have the highest priority during implementation not "ease of writing" (abbreviations) which creates "broken COBOL". The more clearly understandable the code the less time needed for both maintenance tasks and implementation. Ever since COBOL arrived 49 years ago, "ease of writing", abbreviations, destroyed clearly understandable coding, the key to productivity. COBOL was being emasculated before it even got started.

Just imagine the tremendous increase in Software productivity, had all the COBOL programs now residing in company's maintenance library been written in "basic COBOL" as intended and envisioned by the committee that created COBOL and not written in "broken COBOL". The flooding of COBOL coding with abbreviations destroyed COBOL's major reason for being. Those abbreviations reduced it's understandability to the same low level as "machine oriented language's. " Broken COBOL" like broken English can range from poorly understandable to nearly incomprehensible. This assured the COBOL maintenance library's throughout the world, the lowest productivity rating possible. Had those programs been written as envisioned and intended by the prestigious Department of Defense Committee our Software Industry would be at the top of Industry's productivity chart all these years like it's sister Industry, Computer Hardware. Both Industries had strongly benefited from the advancement made by micro chips over the years not just the Computer Hardware Industry. The difference was the micro chip

benefits were negated in Software by poorly readable/understandable coding. Machine oriented languages were never conducive to understandability and COBOL which was conducive to understandability never had that ability utilized. Instead it was written in "broken COBOL". The Committee tried unsuccessfully to warn us about that, by the quotes emanating from their deliberations.

The following are quotes emanating from the Committee during its deliberation in 1959:

1) "Majority of the group favored maximum use of simple English language."

2) "The need is for a programming language that is easier to use even if less powerful."

3) "It certainly was intended (and expected) that the language could be used by novice programmers and read by management. We felt the readability could be achieved because of the intended use of English ... most of our concentration was on making it "easier to read."

4) "The driving force behind consideration of all the "statements" was the concept of readability."

5) "and there was a definite emphasis on ease of reading not on ease of writing."

Special attention should be given to quote 5. From day one of COBOL, I should say, from day one of broken COBOL, the entire Software Industry gave the emphasis to ease of writing not ease of reading. That was the killer of productivity, and it was based on the devastating misconceptions concerning abbreviations by the overwhelming majority of Industry's so called "Experts".

The productivity path reflected in these quotes is still available. All that is needed is for our Industry to go, "back to the future" and use COBOL in implementation as intended and envisioned by the quotes that emanated from the Committee. NO MORE ABBREVIATIONS, Understandable coding given the highest priority. NO MORE BROKEN COBOL. Broken COBOL slows implementation time and devastates maintenance time. True, correcting the damage already done by the broken COBOL in our maintenance library is difficult, but at least we can assure not one more software program will be written using broken COBOL will find it's way into the maintenance library. Once our Software Industry recognizes how much more productive basic COBOL is for implementation and maintenance the machine oriented languages will drop by the wayside. Starting today we can make a 180 degree turn in Software Productivity. COBOL has already proven it can effectively handle most tasks in our Industry by the tremendous number of programs still being maintained, Very few machine oriented languages will be needed once COBOL is used to it's fullest advantage. COBOL is still the only Computer Language conducive to clearly readable/understandable coding. The key to Software Productivity.

There is a drawback. For forty nine years, since COBOL was created, programmers with management's encouragement, have been writing "broken COBOL." Programmers have, in all this time, flooded their COBOL code with abbreviations. Their priority has been, counter productive, "quick coding" not the productive "Clarity Concept" envisioned and intended by COBOL's creators.

Broken COBOL has become an ingrained habit of programmers in our Software Industry. It has become second nature in their doing and their thinking. Management will have difficulty in changing the deeply imbedded habit of programmers using abbreviations. They personally can't check every newly written source code for abbreviations or easily readable coding and still have time to take care of their many other managements responsibility's. Clarity Concept Systems has developed a COBOL Standards Monitor that will do it for them. It is called Enforcer.

Enforcer can read the COBOL source code at any point of implementation. Therefore when a programmer finishes WORKING STORAGE, before he starts the PROCEDURE DIVISION, he can run Enforcer and it will flag every abbreviated word in data names. This gives the programmer the opportunity to change the abbreviations to their full word spelling before it's repeated numerous times in the PROCEDURE DIVISION. The Enforcer is run once more just before the first compile. In that run it will flag all abbreviations found in paragraph names again giving the programmer the opportunity to correct them. These two Enforcer's runs will be a small step toward creating the "Basic COBOL" envisioned and intended by it's creators. It will be a giant leap on the path to Software productivity. Enforcer can monitor 65 other coding standards, none of which will reach their full potential unless abbreviations and broken COBOL are first eliminated, bringing the coding to be much more clearly understandable. For example, how much productivity benefit can be expected from "structuring code" when the COBOL coding ranges from poorly understandable to nearly incomprehensible due to broken COBOL created by abbreviations used in data and paragraph names. It is similar to paving dirt roads to speed the flow of traffic but leaving all road

signs in a foreign language. You will be going faster but you won't know where you're going. When you finally figure out where the program function is you still have to figure out how the function is implemented in spite of the broken COBOL. Any of 65 standards, if desired, can be flagged and corrected during the runs of the Enforcer. The flagging of any standard can be turned "on" or "off" by management.

Our Software Industry has a choice. Continue to write "broken COBOL" as we have for 49 years or starting today, write basic COBOL as the Committee intended and envisioned.

The only thing we have to lose, is our Software Industry's place at the bottom of Industry's Productivity Chart.

Jerry Sitner
Consultant/Productive Software Development
Clarity Concept Systems
212 254-3358