Our Ageing Computer Programming Workforce

Shôn Ellerton, October 1, 2019

With each passing year working in IT, I've become aware of a growing ageing workforce of computer programmers and coders. Why would this be?

Twenty years ago, I had this vision that most of today's computer programmers and coders would be replaced by young twenty-somethings and the older ones, like me, shelved out of existence.

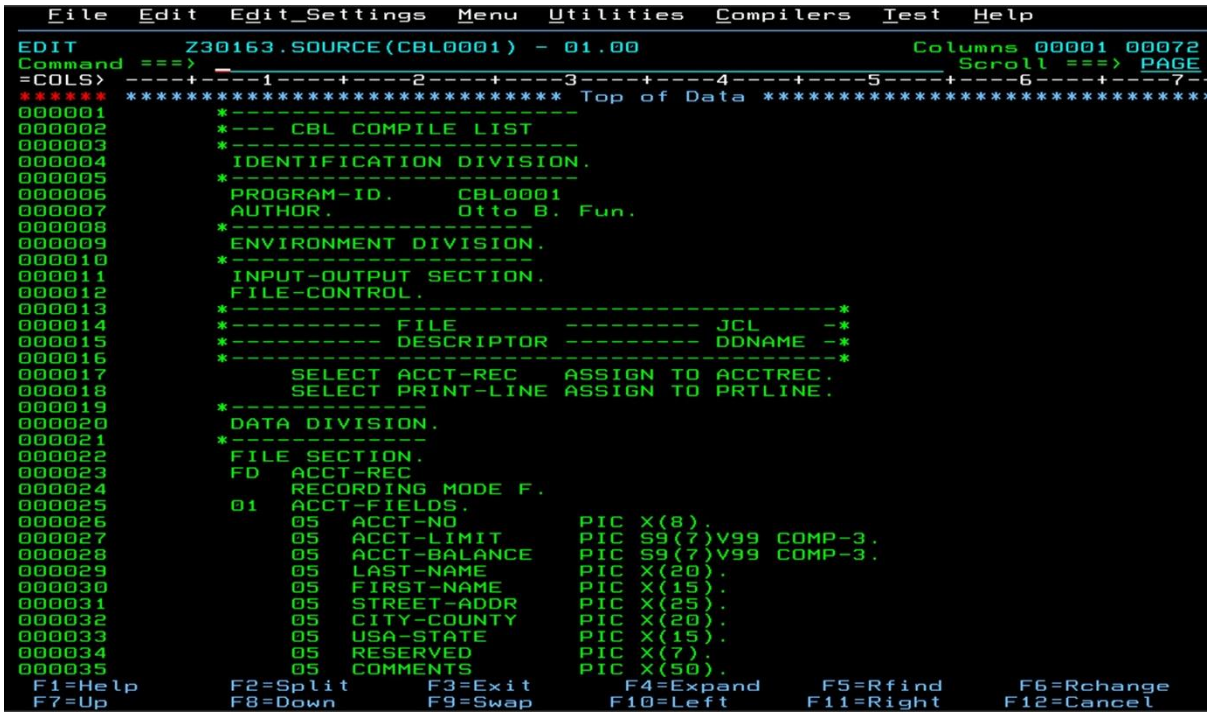
How wrong I was!

I've been contracting for various companies and organisations within the world of database programming, integration and migration during the last two years. What I've noticed is that the average age of those who I have worked with in developing software using popular languages like C#, React, Java and Python has risen. It's as if, at some point in time, there have been no new programmers and coders going into the workforce.

Old programming languages outliving their owners

Recently, I have been working alongside coders well into their late senior years maintaining and programming IDMS (Integrated Database Management System) databases on IBM mainframes. Remember, these are not RDBMS (Relational Database Management System) databases, the type most database programmers are familiar with these days. Programming, maintaining and extracting data from IDMS systems requires an intimate knowledge of mainframe programming with the ability to write COBOL programs and to create journals and jobs to extract data from the mainframe via conversion tools

before being staged into a RDBMS data warehouse. What happens when we have no more COBOL programmers left?



To put this in perspective, as of writing, 70 percent of data stored in Fortune 500 companies are stored on mainframe. Around 90 percent of ATM transactions take place on mainframes using COBOL and CICS. Vast amounts of banking, governmental and big-industry data is stored on IDMS databases on mainframes. Many cities rely on COBOL algorithms to run their traffic light systems. The list seems endless.

There have been a few initiatives put in place in the industry to try to bridge the gap between young and old programmers required to maintain these critical systems. The so-called ‘odd-couple’ system whereby older experienced programmers pass on the necessary skills to younger programmers much like a master craftsman passes on skills to an apprentice or journeyman. However, these initiatives are few and far between. Most universities and other seats of learning in the IT world do not teach these older technologies as they are usually viewed as obsolete, old-fashioned and a dead-end in the IT industry. However, there is a small handful of universities (notably in India) where COBOL is firmly embedded into the syllabus of a computer programming student to take advantage of the skills void left behind when the old COBOL masters end their working lives.

The same could be said for another ancient technology: FORTRAN. FORTRAN programmers are scarce indeed. Whilst I was in university, FORTRAN was taught in our computer programming classes but it was soon phased out a few years later to newer technologies. FORTRAN is still heavily used in defence and in scientific computing, notably using CRAY supercomputers. Predicting weather using computational fluid dynamic models are often calculated using FORTRAN programs for example. Along with C, COBOL and FORTRAN constitute the 'grand-daddies' of programming languages but only C is still widely taught amongst these three.

The big question arises. Why convert all these systems to newer technologies if they are currently working. If they do require converting to newer technologies because of a compelling reason to do so, what's to say that this newer technology will become equally obscure and arcane like COBOL today? Whatever the outcome, we will always need programmers with the skills to be able to maintain our current systems.

It's not just with the old languages either…

Whilst I was expecting the average age of mainframe and COBOL and FORTRAN programmers to rise as the years go by, I did not expect the average age of C#, SQL Server, React and Python programmers to rise as well. Yes, there is a plethora of new technologies and services being developed each year,

many of which, will probably die of death, leaving only the strongest to survive. C#, C, SQL, Python, R, VB, Java and Javascript, to name a few, are certainly not new technologies; however, many of them are part of most computer programming syllabuses. One would expect a healthy crop of young coders to populate our programming workforce; however, this is not the case. Many industries struggle to find the right programmer to fit their needs with many of them resorting to call on the services of the highly competitive job recruitment agency market to fill in the necessary gaps.

Possible reasons

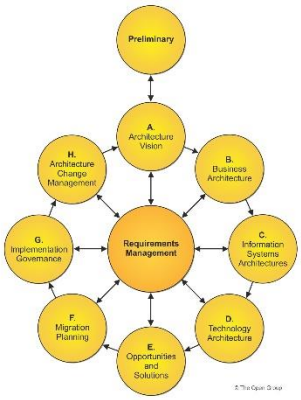
I asked the question to various colleagues and friends if they too have observed the rising average age of programmers in the workforce. One suggested that the tried and trusted technologies aren't 'sexy' or progressive enough to scale into tomorrow's industries. For example, machine-learning and artificial intelligence (AI), virtual and augmented reality (VR/AR), graphics and game design seems far more interesting. Another suggested that the younger generation do not possess the same work ethic and are far more likely to take up another opportunity at a drop of a hat. These reasons are a little over-reaching; however, with the increase of casual jobs over permanent positions and the reduction in job security, the second reason has some merit.

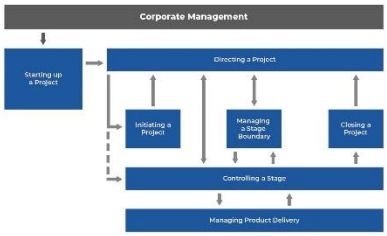
Hardware aside, another theory is that, perhaps, logic and programming, which traditionally took the lion's share of the syllabus in IT educational programmes have been augmented by a swathe of other 'softer' and more organisational skills. For example, learning about software and database deployment in the DevOps environment, understanding Agile methodologies, how to manage IT frameworks, work out infrastructure and data architectural solutions, selecting software-as-a-service (SaaS) products, and so much more detracts time required to learn about the core basics of logic and programming.

Yet another theory states that many of the older programmers never started life as a programmer at all but rather learnt how to program through necessity in their own industries. For example, my grandfather learned FORTRAN because he needed a quicker way to calculate sugar beet trial statistics. I, myself, started off as a civil and structural engineer but taught myself how to program databases through necessity of automating project management tasks. One of my colleagues started off as a biochemist, another as a medical therapist.

Attracting programmers and coders is not terribly easy because there has been an emphasis to learn a vast array of peripheral skills our IT industry seems to require, many of which, require specific certifications. One only need to review

some of the job vacancy postings on offer. Many seek solutions architects, infrastructure managers, bespoke software and reporting specialists, and those armed with a battery of lofty certifications like ITIL, PRINCE2, CCSE and TOGAF. Many of these job briefs are crammed up with requirements that most candidates could not possibly all have expertise in, yet very few ask for the simple requirement of being able to program and deduce logically.





Summing it up

Undeniably, of all the technologies we use today in the world of programming, only the strongest will survive. Built on those technologies will be countless systems, many of which, will be critical to maintain in the future, regardless of what newer technologies may emerge. Perhaps, in the near future, the technologies we use in abundance today like Visual .NET, Python, Java and Javascript will become tomorrow's COBOL and FORTRAN.

Our younger generation of IT professionals may be learning all these new IT skills to keep ahead in the IT race; however, when it comes to the crunch, good programming and logical skills are usually all that is required. Unfortunately, it is becoming more difficult to find these 'back to basic' skills in today's IT industry which is why we have an ageing computer programming workforce.