Upskilling the 'Digital' Workforce

This is a popular phrase, used liberally by educators, vendor trainers and other organisations like Microsoft (Global Skills Initiative) and LinkedIn (Skills Path) with their own programmes of numerous courses covering many aspects of IT. This sounds a great idea for bolstering IT skills for the changing world of computing.

I have my reservations though, based on 50+ years of IT experience, both at the coal face across a wide set of industries and latterly as author and researcher. I have examined many, many computing curricula across schools, universities and other organisations offering various forms of "IT" education and, like Sir Thomas Beecham[1], I have 'formed a very poor opinion of it'. Why?

What is the Problem?

That is difficult to put into a few sentences as there are many aspects of this current computing education which make me uncomfortable.

- One is the lack of a common idea of what constitutes IT or 'digital' knowledge, from Mrs Jones using the internet for supermarket shopping to people at the forefront of technology.
- The variation in topic coverage, going from the nitty gritty to the sublime in the same course but for different topics — it feels *bitty* and cobbled together, possibly developed by a lot of people working separately.
- The lack of pragmatic content, particularly in computer science (CS) courses, for example, '*.. this technique is used in the X industry to estimate the depth of a depletion layer in semi-conductors and also in Y industry in..*' never appears. In other words, it lacks real world context.

  Let me give an illustrative example.
- In network courses or modules, one is taken on a journey through the bowels of networks — routers, MAC, 7-layer models, TCP/IP and much more. There is no discussion of what the pragmatic aspects of networks are — congestion, compression, data loss, high availability, cybersecurity forensics, network performance management, network simulation tools — and a host of other day-to-day aspects of designing and running a network.

  In short, much of the computer teaching fails to answer the 'so what?' question.

  - There is a plethora of courses offering a specialism at the end of it and cybersecurity is a case in question. I have looked at many of these and none, apart from a single IBM course, ask for any previous training (pre-requisites). This attempt to develop a specialist from a standing start is akin to trying to become a cardiac or brain specialist without going through general medical school.

---

1 '*I have recently been all round the world and have formed a very poor opinion of it'- Sir Thomas Beecham, English Conductor,*

- The overall impression I was left with was that most courses resembled a fairly random set of specific topics with no feeling a synergy and what IT was all about overall. This left me with the impression that I might do all the topics in a course but still feel uneasy about whether I could give a one-hour presentation on what IT was all about.

I have two analogies which might illustrate what I have said above more clearly.

A. A nautical analogy. I join a course on sailing and am trained on yachts, speedboats and a few larger vessels and graduate feeling very pleased with myself.

However, I meet an old salt who had sailed the seas for decades and he asked me what I'd been taught about navigation -with instruments or by the stars — and about trade winds, dangerous waters, different ports and assessing weather conditions in the absence of a forecast.

I pleaded a headache and politely bid him farewell and a fair wind on the seas of life.

B. Picture a hypothetical course on the motor car which I undertook. It covers the components parts — engine, brakes, transmission etc. — dealing with equations of coefficients of friction, the Carnot Cycle, adiabatic expansion of gases, hydraulic transmission and a host of other topics. Great!

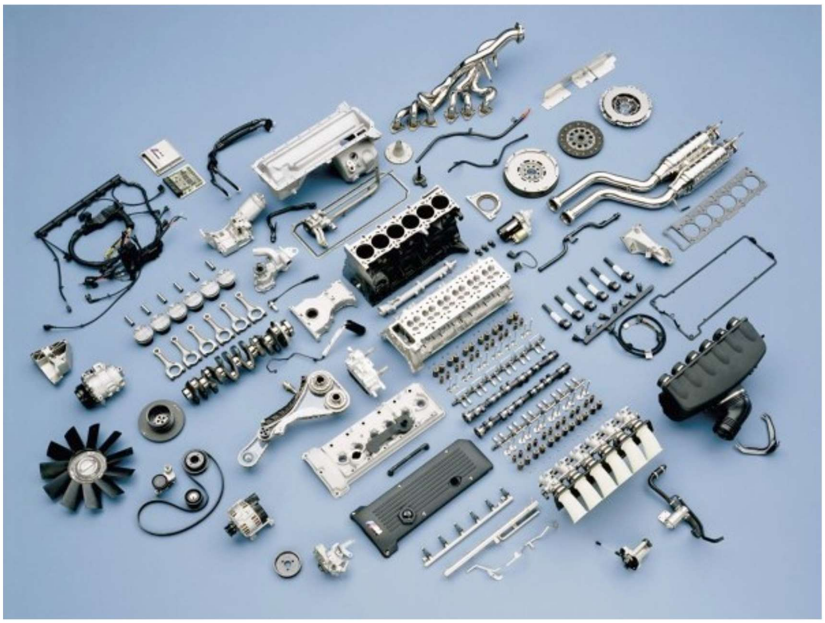

Figure 1: Parts of an Automobile Engine

However, when I re-joined the real world after three year's study, I realised I couldn't drive a car, read a map, plan a journey, assess what kind of vehicle I needed for my purposes, had little idea of service intervals and other day to day things necessary in possessing a car.

Need I go further on this???

What is the/a Solution?

I have thought long and hard about this and came to the conclusion that all IT education should fit into a framework and what takes place within it reflect the activity flow of nearly all IT projects, large and small. When people understand this, they see the value in taking this view from 10,000 feet and then home in on topics, rather like one does with views in Google Earth. It is also key that people recognise that they are moving into IT as a career and not just a specific job paying X £ or $.

Another analogy is apposite here in the career vs. job context.

> 'There's an old story about two men working on a railroad track many years back. As they are laying track in the heat of the day, a person drives by and rolls down the window (not enough to let the air conditioning out, but enough to be heard). He yells, "Tom, is that you?" Tom, one of the men working on the track, replies, "Chris, it's great to see you! It must have been 20 years ... how are you?" They continue the conversation and eventually Chris drives off. When he leaves, another worker turns to Tom and says, "I know that was the owner of the railroad and he's worth nearly a billion dollars. How do you know him?" Tom replies, "Chris and I started working on the railroad, laying track, on the same day 20 years ago. The only difference between Chris and me is that I came to work for $2.25/hour and he came to work for the railroad." '

A key factor which tells why the career mentality is superior to the job mindset is the volatility of change in IT. Many people have written about this and point to the idea that IT jobs, like the Covid virus, mutate over time until its boundaries change or, in the worst case, become a different subject or, even worse still, become obsolete. The half-life of an IT job in a particular form is estimated to about 18-24 months.

This obsolescence does not necessarily occur because its core is no longer current, but often because eventually senior managers will question the investment in 'shiny new' technology with '*What is all this new-fangled stuff doing for my bottom line*'? If the question cannot be answered, the job goes to the wall or becomes very low key. If the incumbent is 'one metre wide and 2 kilometres deep'[2], he/she faces the next 40 years in that job, leaves or has to retrain. See the link below for another perspective on this.

IT leadership: Why job complacency sets in and how to fight it
https://enterprisersproject.com/article/2021/4/it-leadership-how-fight-complacency

Tale of Willy OneSkill

If you only learn one thing in studying IT and that is that signing up to 'lifelong learning', you will have a career and not just a job. If you haven't got a solid underpinning IT knowledge, every new topic will be an uphill struggle to learn and you will have to sprint to keep up with the evolution of IT — technology, tools and techniques, as well as job mutation. Add to this the fact that no computing specialism is an island and needs knowledge of other aspects of computing and the case for solid IT training is made.

---

2 *Peter Cochrane: Where are the generalists we need today?*
https://www.computing.co.uk/opinion/4022661/peter-cochrane-generalists

The FUMPAS Framework

In my years in IT, I tried to encapsulate it in an acronym which evolved with IT progress and the importance of some of its components from FUMP to FUMPAS, which I will explain now.

The table below explains briefly what the initials stand for and why the factors they represent are of importance. This is not a wish list; the items represent the key factors which need consideration in any computing application project

Functionality	Does the system do what the user/business asked for? This functionality MUST be agreed with the users (business) before taking one step forward in the project. Get this wrong and you might as well all go home. This applies to all aspects visible to the end user, including the business software.
Usability	Is it usable — clear and ergonomically easy to use? Does a user have to enter 10 screens of data to add a customer to the customer data base? The UI (user interface) is a key factor here.
Manageability	It must be manageable in all aspects, especially in monitoring, measurement, analysis and actions. *If you can't measure it, you can't manage it and if you can't manage a system, you are in trouble*. This 'M' covers many aspects of running and maintaining business systems and its components.
Performance	It must deliver acceptable response times, usually via perhaps 90 percentile measurement for the application. Consistent, fast response times are vital for businesses using web sites for sales and marketing.
Availability	For some applications. 100% availability of the application, not just the hardware and software is required. This needs careful hardware and software thinking up front. It will be difficult to retrofit it after installation.
Security	Security is now the flavour of the month although the other factors listed here must not be forgotten, as they seem to have been in UK banks and the UK NHS. Apart from the usual impact of data theft, ransomware and data destruction, it has an impact on the availability of applications and user satisfaction. For public web sites, this aspect is vital.

Table 1 Key Factors in a Successful IT Project

These are the areas for consideration when designing, proposing or evaluating a computer project and is a valuable tool for executives presented with a geek-style proposal for a new system. Their relative importance will vary depending on the application and type of organisation.

For example, a military application will have a heavy focus on security and availability, a marketing application on usability, availability and performance; this does not mean the other factors are ignored as they will have an importance which varies with business need.

The Flow of IT

The steps in this flow are not necessarily all technical for various reasons. Today, IT is a crucial, often critical part of the business process and, as a result, IT and business are tightly entwined. To understand this flow requires some IT knowledge on the part of business people and vice-versa for IT people.

My view is that today (mid-2021), neither of these requirements are fully met, hence the statistic that three quarters of IT projects fail at levels ranging from 'not quite what we wanted' to 'total failure and abandonment' support my thesis.

One thing to learn upfront about the flow of an IT project is that each step relies to a great extent on the previous one and is the entry point to the following step.

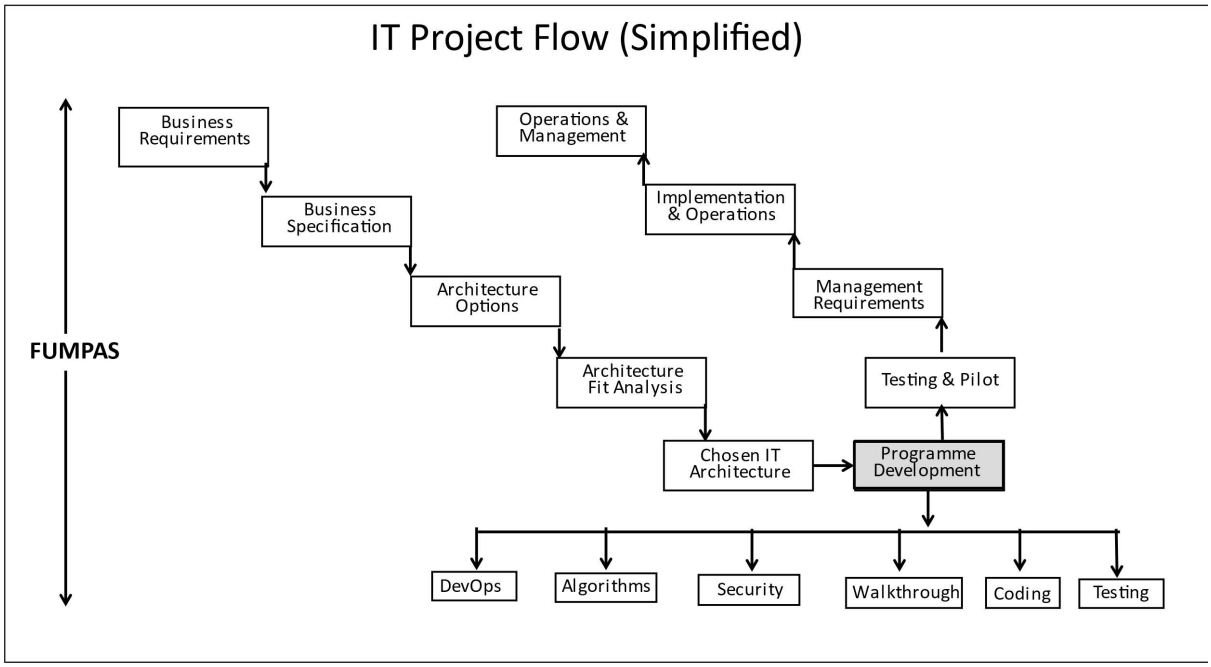

Figure 2: The Flow of an IT Project

The old adage that a chain is as strong as its weakest link is not quite true if you examine Lusser's Law, which states that the availability of a serial chain of components is not the availability of the least reliable but the product of the reliability of each component. See the link below for a discussion of this law;

Lusser's Law and Applicability
https://www.apmdigest.com/lussers-law-and-applicability

What this means in this case is that getting a perfect 'score' for one component, for example, coding or project management, counts for little if the rest of the components have poor scores; the overall 'success rating' will follow Lusser's Law to some extent,

IT Sub-flows

Note, that even within the software development area, there is a similar breakdown of the overall flow of an IT project, wherein Lusser rules too.

Software development activities

- Identification of need — this may have partly done at initial 'need' time.
- Planning process
- Designing
- Implementation, test and document
- Deployment and maintenance.'

Software development
https://en.wikipedia.org/wiki/Software_development

And remember, the above activities are part of the subset of overall activity called DevOps and DevSecOps, thereby blowing out of the water the myth that software development is simply coding. This dangerous myth is often propagated in schools and often at University.

This 'subterranean' flow of sub-activity can apply to other areas in the flow too.

Summary

- Current IT education does not match up to what the workplace needs and uses in tools and techniques. It does not have the scope of ability to match the speed of evolution of IT.
- The field of education is like a patchwork quilt of difference shades and depths and often comprises a set of topics with little connectivity or 'where used' context. It is also limited in scope and ignores major topics such as *mainframes, enterprise computing, cloud, service levels and high performance computing (HPC)*.

  These omissions limit the students' career scope considerably and it is disingenuous to hide them totally.
- Much IT education is based on computer science (CS) which is very limited in its scope and rarely offer s workplace context
- There is a prevailing idea that an IT specialisation can be taught from scratch without solid prerequisite knowledge
- There is also the mistaken notion that 'coding and algorithms' are the 'be all and end' all of IT work. This activity is just one in a chain of varied activities.
- An IT project is a chain of activities which depend on each other for the success of the project.

Books by the author

High Availability IT Services by Dr. Terry Critchley
https://www.crcpress.com/High-Availability-IT-Services/Critchley/9781482255904

High Performance IT Services by Dr. Terry Critchley
https://www.crcpress.com/High-Performance-IT-Services/Critchley/9781498769198

Making It in IT by Dr. Terry Critchley
https://www.crcpress.com/Making-It-in-IT/Critchley/9781498782760

Modern IT Concepts and Technology: An IT Study Guide for Beginners and Practitioners
Kindle Edition
https://www.amazon.co.uk/Modern-Concepts-Technology-Beginners-Practitioners-ebook/dp/B0826XN9L2