The Kolence Effect
Margaret Greenberg

After leaving “the Boole,” as Kolence and others affectionately referred to Boole and Babbage, a company he co-founded with Dave Katch, he started Kolence Associates. This was a continuation of the consulting company the two started in order to finance their dream. Things don’t always work out as planned. Nevertheless, Kolence has remained a force for better programming.

When I recently read his riff discrediting the value of flowcharting, it seemed as if I had found a friend, a kindred soul that had been missing for years. In the 1960s and early 1970s or so, there was considerable emphasis on creating flowcharts for every program even though it was just created as part of final documentation. The idea was to enable a maintenance programmer or analyst to quickly pick up pieces for improvements or bug analysis. However, that rarely worked as expected. In the event of a bug, say the famous COBOL 0C7, the flowchart was not helpful. One just found the referenced field via the dump and determined why there was alpha data in a numeric formatted field. Often the management wanted every instruction to have its own little box on the flowchart form. By the way, this documentation was a heavy manual process using a template and handwriting. A real mess to clean up and hard to maintain, these charts were often ignored after initial creation.

Then came Autoflow software which automated the “documentational” flowchart. It wasn’t a significant improvement over the hand crafted version because there were not very good graphics on the mainframe therefore the charts were not easy to read. Nevertheless, it generated doc and management was happy. Kolence was still right: Flowcharts were not so useful.

During my stint in operations over ten years ago, he loomed large. There are many processes and activities that could be improved to make operations work better and reduce the operational costs. And he is still on target with complaints about system/design program/coding practices. Those have evolved over the years from one “next greatest thing” to another. There has been much angst felt when these new ideas were introduced to change-averse management. Maybe twenty-five years ago, we started to use prototyping software. This allowed our “customer” departmental management to view mock-ups of screens and reports prior to final design and programming. Well, that went against the usual linear path of requirements, system design, programming, acceptance, etc. and quite a fuss ensued until they were able to accept this as a time saving, cost cutting measure.

Though Kolence spent many years grousing about the state of programming and design, he seems to favor a current hot topic, Agile Programming, Refining his idea of software physics has always been one of his missions. Many of the early CMG conference papers are directed toward that in various forms. My
researching/reviewing the early years has been quite interesting. It is very much so for the more experienced “older” crowd who actually lived through some of the harrowing times way back when mainframes were “tubular” shortly after WWII. Then the mainframes were not so big, e.g., the IBM 1400 series. The Control Data Corporation (CDC), as a brand, is now entirely history though this was the machine for which the earliest performance software was written.

What did he do after all this? For a few years, he both worked at Kolence Associates and retrenched himself by working at a hardware store. He was happily solving problems for homeowners in the same way as he did systems problems. However, he found the hardware store more gratifying because the work was less complicated and he had more “accomplishments” at the end of the day. Due to health issues, he is now fully out of the workforce, but has a sunny disposition and is quite happy.

Next time: more discussion about software physics and the early years of Boole and CMG.