## Letter from the 2016 IEEE TCDE Contributions Award Winner

## **Why Data Engineering** revisited

First of all I can't take all the credit for the establishment of the initial Data Engineering conferences. Richard Shuey from GE and Mas Tsuchiya from TRW were equally effective, and we received organizational support fromChittoor V. Ramamoorthy. These computing experts cannot be here now.Ram died this March; at the time he was working on his Handbook for Software Engineering, leading us to a new term, Data Engineering.

I provided some international and commercial background within this diverse group. I had learned programming in 1958 in the Netherlands, working for NATO. During 1964-1965 I taught at a IIT Kanpur in India. In 1973 I led a too-early startup on remote access to remote business data over 300bps AT&T modems. To save on phone costs we sent an employee weekly between Los Angeles and New York with the updated date files on a 2400 tape.

Several IEEE members had encountered a dissemination problem. In the late seventies there was much industrial innovation in the database arena within companies as GE, who had built a network DBMS following CODASYL standards, and TRW, who had worked with IBM on the database for the Shuttle program, which became IMS. Such work was hard to disseminate directly, in part because these efforts were components of larger applications. Engineering issues became more broadly significant as people were starting to put new theoretical database concepts into practice, as grown from the seeds sown by Ted Codd's 1970 relational model paper.

Getting started was hard and not always clean. ORACLE, one of the earliest commercially viable relational products, was created by providing relational access to a hierarchical database written for the US Air Force, by automatically inserting dummy parent nodes when relational operations required it.

But such technical innovations related to database engineering had to be presented in broad general computer science venues, as AFIPS or IFIP (the American and international Federations of Information Processing Societies) conferences, in their broad application tracks. For instance, during 1967-1968 Gio's ACME project developed a column-store database system for on-line use by medical researchers. The database was built and enabled within a time-sharing system which incorporated an incremental compiler. Such a compiler incorporates the advantages in terms of being able to make changes in the code during execution. There were no hypotheses, only feasibility, being proven, and being first, no comparisons could be made, hence it was not research. So its description was published in the AFIPS proceedings, and hidden as such from the later Internet. Later column stores were again considered novel.

Of current significance is that ACME also checked if references were within array bounds, at about a 10% overhead, well worth it in terms of code reliability.

Other technologies were equally hard to disseminate, as Smalltalk, an object-oriented compiling system for the Alto system. It was used by Allan Borning, an early student of mine, now at the University of Washington, then at XEROX PARC. I learned that it also provided secure execution. I believe that if those user-facing technological concepts had been more widely shared, wed now have safer code and fewer opportunities for hacking. Smalltalk was not made freely available because XEROX lawyers wanted to protect its commercial property value. Now valuing IP and modeling the surrounding Intellectual Capital is my research focus.

Subsequently, free-standing products, as C, and its successor, C++ were made liberally available to a general audience by Bells research arm. That openess enabled further broad development, outside of coherent system environments. Being developed for telecommunication experts, performance was more important than code reliability. Most hacking attacks today are enabled by intruders gaining access to cells by using indexes outside of array bounds.

Sang Cha, another student of mine, here with us at ICDE, was able to use the performance demands of the telecommunication industry to develop in-memory technology, now available as HANA from SAP.

Today, entire capable database systems, as MySQL, are distributed as freeware. They are sufficiently capable that startups, as Twitter, can depend on them for their initial growth. But soon, engineering issues, often associated with reliability, arise. With mechanical storage becoming a backup for semiconductor storage, and, with ubiquitous high-speed communication, moving much data into the cloud, the technological foundations keep changing.

It is great to see so many early colleagues, postdocs, students, and grand-students here in a corner of the world that was not known to the IT community on the seventies and eighties.

I expect that the Data Engineering conference will continue to play a role in this world that now covers all sciences and entrepreneurial fields.

A final anecdote relevant to this ICDE meeting being in the Baltic and my background.

I named an early computer on the ARPAnet, now the Internet, funded for electronic commerce-work, “Haring”. Colleagues would be able to send email to Gio@Haring (that was before 3-field IP-addressing was introduced.)

The reason for the name is a Dutch saying: The fishing of haring is the basis for all commerce.

The Dutch needed salt to preserve the Herring they caught. The sun allowed the Portuguese to make salt, but they were not interested in the Dutch herrings. But they needed wood for ships, they had deforested much of their country in the preceding centuries. So the Dutch had to take their herring in the Baltic regions, trading it for wood to get the salt. Later Dutch went to Indies to get spices and really got rich. Marten Kersten here is well aware of that history.

Gio Wiederhold
Stanford University