

At a Meeting of the Faculty of Arts and Sciences on April 18, 2006 the following Minute was placed upon the records.

THOMAS EDWARD CHEATHAM, JR.

Born: September 20, 1929

Died: June 10, 2001

With the passing of Thomas E. Cheatham, Jr., on June 10, 2001, computer science lost one of those who made it both a science and an engineering discipline. Tom's career began when few general principles governed the design of computer systems. When he taught his last course, his style of thinking had transformed the software business from piece work to mass production. During his professional life Tom inhabited the military, government, corporate, and academic spheres. But he was most at home at Harvard, where new ideas grew quickly, nourished by the plentiful supply of bright students. For years Tom was the only computer scientist at Harvard with a research budget large enough to support a first-rate laboratory, with an ARPAnet host and the staff to keep it running. Throughout, his benevolence and playfulness balanced his serious leadership burdens.

Tom was born on September 20, 1929, in the east Illinois farmland. He received his B.S. and M.S. from Purdue, studying mathematics, and continued towards a Ph.D. In 1954 Tom joined the Army, hoping to be commissioned as an officer because of his excellent mathematical skills and poor eyesight. This plan was a miscalculation. His vision was recorded as 20/20 and his rank as private – on KP duty. But he caught the attention of the officers, and his first mathematical task for the military was keeping the books at the officers' club. Within a year he had become a computer chief at the National Security Agency. Tom's ability to impress military brass, and to move effortlessly around their bureaucracies, paid boundless dividends during his years at Harvard.

In the late 1950s Tom, working in private firms, started his major scientific work, the design of programming languages and software environments. He articulated ideas that became everyday practice only decades later in languages such as C++ and Java. In 1961 he started Computer Associates, a firm that prospered commercially while it made important scientific contributions. By 1966 Tom had written a compendium on programming language design and implementation under the title *The Theory and Construction of Compilers*. He taught a course on the subject at Harvard, and the notes were revised and promulgated many times as the field advanced. Always more interested in getting it right for next time than in archiving an imperfect attempt, Tom never published his compiler notes as a book. Yet for many, they were the bible on the subject.

In 1967 Tom sold Computer Associates – he never liked being a business executive – and his temporary appointments at Harvard evolved into a tenured professorship. Though he never received a Ph.D., his Harvard offer was only one of his academic options, because of his widely recognized software research. He was attracted to Harvard because of the intimate scale of its computing activities.

Tom bridged the divide between software theory and practice. He never denied complexity – he would not simplify a problem so that it admitted an elegant but unrealistic solution. Instead Tom searched for abstractions to isolate complexity and control it. He pioneered the development of

extensible programming languages, whose syntax could be adapted to meet unanticipated needs. He recognized early on that a program might carry information in addition to its algorithm. Documentation of what the program was supposed to do could be embedded in the program text and aid both human understanding and machine analysis. He entered the field of automated theorem-proving because of its promise to improve software reliability. The fundamental idea that a program can be a formal object subject to mechanical analysis underlies all of Tom's research. As a mathematical abstraction the insight went back to Turing, but it had been lost in practice in the furious production of code when computers became commercial.

Tom's animated lectures were down to earth even when the subject became abstract. Tall and bespectacled, a former college saxophonist, he was part performer, part salesman, and part scholar in front of the class. He made programming not just fun, but funny. Tom shamelessly anthropomorphized software while rigid formalists considered this practice sinful. Programs said "mumble mumble" and, for reasons no one knew, always had a variable named WALDO.

Tom advised 36 Ph.D.s, and gave them both his personal support and his intellectual collegueship. He cheerfully acknowledged that 99% of his ideas were wrong. To get better ones he enlisted the smartest students, the "prize pigs" as he called them, whom he treated, Midwestern-style, as equals. Tom so welcomed interruptions for scientific conversations that his colleagues thought it miraculous he got anything done.

Tom showed confidence in Harvard students. He relied on them to figure out the implications of his far-reaching vision. And he let them do what they wanted if they had bright ideas of their own. The "bean counters," as he called them, were not always happy about that, nor were the academic deans tracking students' progress. Nor even, sometimes, was Tom himself when he found out the upshot of the computer privileges he had cheerfully granted. But students who are now professors at computer science departments and founders of software firms realized incalculable dividends because Tom valued reward over risk and was reluctant to say no.

Tom is survived by his four sons, Dan, Steve, Tom III, and Ben, of whose achievements he was unfailingly proud. He shared with them his love of food, wine, classical and jazz music, and travel.

Tom's enduring legacy is that a computer program embodies not just an algorithm, but an abstract model of how some part of the world is structured and operates. As Tom wrote in 1974, "To employ a computer we *require* a theory." Our job was to produce software that usefully reflected a fitting theory. We remember him for always pulling our eyes upwards no matter how messy was the code in which our feet were mired. As Tom's sometime colleague at Harvard, Ivan Sutherland, wrote on learning of Tom's death, "No one will be able to erase the stuff he wrote high up on the blackboard."

Respectfully submitted,

William H. Bossert
Ugo O. Gagliardi
Henry H. Leitner
Anthony G. Oettinger
Harry R. Lewis, Chair