Funding a Revolution: Government Support for Computing Research

Committee on Innovations in Computing and Communications: Lessons from History, National Research Council

ISBN: 0-309-52501-2, 302 pages, 6 x 9, (1999)

Part II

Case Studies in Computing Research

Part II, Chapters 6 through 10, presents five case studies in computing research. These case studies are not meant to be definitive histories of the fields they address; rather, they are intended to illustrate the role the federal government has played in the innovation process by investing in computing research. They contain historical material that is important not only in indicating the government's role per se, but also in characterizing the larger innovation process, including the movement of researchers between universities and industry, the transfer of research results into practice, and the interrelationships among people and research in government, universities, and industry.

Taken together, the cases cover a range of technologies, time periods, and federal investments; individually, they differ considerably in their scope and emphasis. The case studies of relational databases and the Internet, for example, are relatively narrow in the sense that they trace the development of a particular technology or system. The innovations described have a fairly well defined beginning and end, although clearly the systems described in each will continue to evolve over time. The case study of theoretical computer science highlights the development of theory as well as its relationships to computer engineering and the construction of computer systems. It traces the refinement and dissemination of ideas throughout the research community and into educational curricula. The final two case studies, artificial intelligence (AI) and virtual reality (VR), address relatively broad areas of research that are motivated

by a long-term vision: development of systems that display intelligent behavior in the AI case, and development of systems that generate synthetic environments meant to resemble the real world in the VR case. Progress in both these fields is marked by a series of research breakthroughs or technological advances that move ever closer to these objectives.

Each case study concludes with a summary of themes or lessons regarding the innovation process and the government's essential role within it. Because they derive from the individual case studies, these lessons reflect the particular conditions that prevailed at the time described, such as the nascent state of the computing field and dominant styles of federal research management in past decades. The case studies themselves do not attempt to discuss the relevance of these lessons to the current policy environment or to provide guidance regarding future federal support for research in computing. That task is taken up instead in Chapter 5 of this report, which synthesizes the lessons from all the case studies and attempts to consider their more general applicability.

6

The Rise of Relational Databases

Large-scale computer applications require rapid access to large amounts of data. A computerized checkout system in a supermarket must track the entire product line of the market. Airline reservation systems are used at many locations simultaneously to place passengers on numerous flights on different dates. Library computers store millions of entries and access citations from hundreds of publications. Transaction processing systems in banks and brokerage houses keep the accounts that generate international flows of capital. World Wide Web search engines scan thousands of Web pages to produce quantitative responses to queries almost instantly. Thousands of small businesses and organizations use databases to track everything from inventory and personnel to DNA sequences and pottery shards from archaeological digs.

Thus, databases not only represent significant infrastructure for computer applications, but they also process the transactions and exchanges that drive the U.S. economy. A significant and growing segment of the software industry, known as the database industry, generates about $8 billion in annual revenue. U.S. companies—including IBM Corporation, Oracle Corporation, Informix Corporation, Sybase Incorporated, Teradata Corporation (now owned by NCR Corporation), and Microsoft Corporation—dominate the world market. This dominance stems from a serendipitous combination of industrial research, government-funded academic work, and commercial competition.

Much of today's market consists of relational databases based on the model proposed in the late 1960s and early 1970s. This chapter provides

background on early data management systems and then examines the emergence of the relational model and its rise to dominance in the database field, and the translation of this model into successful commercial products. The final section summarizes the lessons to be learned from history. It highlights the critical role of the government in advancing this technology. For instance, although the relational model was originally proposed and developed at IBM, it was a government-funded effort at the University of California at Berkeley (UC-Berkeley) that disseminated the idea widely and gave it the intellectual legitimacy required for broad acceptance and commercialization.

This case study does not address the entire database field (it omits topics such as transaction processing, distributed databases, and multimedia), but rather focuses on events that illustrate the ways in which synergistic interactions of government, universities, and industry built U.S. leadership in a particular subfield, largely through the work of individuals who developed and then transferred technology between firms and laboratories. As James Gray, a senior database researcher, has observed: "A very modest federal research investment, complemented by an also-modest industrial research investment, led directly to U.S. dominance of this market" (CSTB, 1995a).

BACKGROUND

Emergence of Computerized Databases

The U.S. government has always had significant requirements for the collection, sorting, and reporting of large volumes of data. In 1890, the Bureau of the Census encouraged a former employee, Herman Hollerith, to develop the world's first automated information processing equipment. The resulting punched-card machines processed the censuses of 1890 and of 1900. In 1911, Hollerith's company merged with another, also founded with Census support; the resulting company soon became known as International Business Machines (Anderson, 1988), now IBM.

During World War I, the government used new punched-card technology to process the various data sets required to control industrial production, collect the new income tax, and classify draftees. The Social Security Act of 1935 made it necessary to keep continuous records on the employment of 26 million individuals. For this, "the world's biggest bookkeeping job," IBM developed special collating equipment. The Census Bureau purchased the first model of the first digital computer on the commercial market, the UNIVAC I (itself based on the government-funded Electronic Discrete Variable Automatic Computer (EDVAC) project at the University of Pennsylvania). In 1959, the Pentagon alone

had more than 200 computers just for its business needs (e.g., tracking expenses, personnel, spare parts), with annual costs exceeding $70 million. U.S. dominance of the punched-card data processing industry, initially established with government support, was a major factor in U.S. companies' later dominance in electronic computing.

By the early 1960s, substantial progress had been made in removing hardware-specific constraints from the tasks of programmers. The term "database" emerged to capture the sense that the information stored within a computer could be conceptualized, structured, and manipulated independently of the specific machine on which it resided. Most of the earliest database applications were developed in military command and intelligence environments, but the concept was quickly adopted by commercial users (System Development Corporation, 1964; Fry and Sibley, 1974).

Early Efforts at Standardization

As computing entered the mainstream commercial market, a number of techniques emerged to facilitate data access, ensure quality, maintain privacy, and allow for managerial control of data. In 1960, the Conference on Data Systems Languages (Codasyl), set up by the U.S. Department of Defense (DOD) to standardize software applications, established the common business-oriented language (COBOL) for programming (ACM Sigplan, 1978), incorporating a number of prior data-definition languages (Fry and Sibley, 1974). Magnetic disk drives, which could access data at random, began to replace magnetic tape drives, which required serial data access, for online storage. In 1961, Charles Bachman at General Electric Company introduced the integrated data store (IDS) system, a pioneering database management system that took advantage of the new storage technology and included novel schemas and logging, among other features.

During these early years, innovations in the practice-oriented field tended to be made by user groups and industrial researchers, with little academic involvement (CSTB, 1982; Wiederhold, 1984). In the mid-1960s, Bachman and others, largely from industry and manufacturing, set up the Database Task Group (DBTG) under Codasyl to bring some unity to the varied field (Olle, 1978). The group published a set of specifications for how computer languages, COBOL in particular, might navigate databases. In 1971, it published a formal standard, known colloquially in the industry as the Codasyl approach to database management. A number of Codasyl-based products were introduced for mainframe computers by Eckert-Mauchly Computer Corporation (the maker of Univac), Honeywell Incorporated, and Siemens AG, and, for minicomputers, by Digital Equipment Corporation (DEC) and Prime Computer Corporation.[1]

Notably missing from the list of vendors that supported Codasyl products was IBM, which had earlier (in 1968) introduced its own product, IMS, derived in part from a National Aeronautics and Space Administration (NASA) Apollo project. It ran on System/360 equipment. Whereas Codasyl was based on a network model of data, IBM's database used a hierarchical structure. (Cullinet Corporation provided a Codasyl-compatible database for IBM users.) Both the IBM and the Codasyl products were sometimes called navigational databases because they required the user to program or navigate around a data set. Bachman's Turing Award lecture in 1973, in fact, was entitled "The Programmer as Navigator" (Bachman, 1973; Cardenas, 1979).

EMERGENCE OF THE RELATIONAL MODEL

Codd's Vision

At least one researcher at IBM was dissatisfied with both the Codasyl products and IBM's database package. Edgar F. (Ted) Codd, an Oxford-trained mathematician, joined IBM in 1949 and later moved to IBM San Jose. Codd found existing and new database technologies "taking the old-line view that the burden of finding information should be placed on users. . . . [In this view, the database management system] should only recognize simple commands and it would be up to the users to put together appropriate commands for finding what was needed" (Codd, 1982).[2]

In a series of IBM technical reports and then a landmark paper, "A Relational Model of Data for Large Shared Data Banks," Codd laid out a new way to organize and access data. What Codd called the "relational model" rested on two key points:

> It provides a means of describing data with its natural structure only—that is, without superimposing any additional structure for machine representation purposes. Accordingly, it provides a basis for a high level data language which will yield maximal independence between programs on the one hand and machine representation on the other. (Codd, 1970)

In other words, the relational model consisted of (1) data independence from hardware and storage implementation and (2) automatic navigation, or a high-level, nonprocedural language for accessing data. Instead of processing one record at a time, a programmer could use the language to specify single operations that would be performed across the entire data set. Codd's model had an immediate impact on research and, as described below, spawned a number of significant prototyping projects.

Given its eventual commercial success, the relational model might seem bound to emerge and even dominate the field without any government involvement in research. It was formulated, after all, entirely within the walls of an industrial laboratory. But Codd's model was long seen as something of an intellectual curiosity. To gain legitimacy within the field, it had to survive at least two battles—one in the technical community at large, and one within IBM. The relational model might not have survived either battle without government intervention, which, in this case, involved funding of a competing project at another institution.

Within IBM, the trouble was the existing database product, IMS. The company had already invested, both financially and organizationally, in the infrastructure and expertise required to sell and support it. A radical new technology had a great deal to prove before it could displace a successful, reliable, revenue-generating product such as IMS. Initially, the threat was minimal; Codd published his original paper in the open literature because no one at IBM (himself included) recognized its eventual impact. The response to this publication from the outside technical community, however, soon showed the company that the idea had great commercial potential. To head off this eventuality, IBM quickly declared IMS its sole *strategic* product, thus setting up Codd and his work to be criticized as counter to company goals. Internal politics further compounded the situation, as IBM was not accustomed to major software innovations coming from IBM San Jose, which until then had worked primarily on disk storage.

In spite of IBM's reaction, Codd spoke out zealously and promoted the virtues of the relational model to computer scientists. He arranged a public debate between himself and Charles Bachman, at that time the key proponent of the Codasyl-sponsored standard. The debate exposed Codd to criticism from within IBM that he was undermining the company's existing products, but it also achieved his intended effect on the technical community. In the early 1970s, two projects emerged to develop relational technology and prove its utility in practical applications. One, System R, began within IBM, and the other, Ingres, began at UC-Berkeley with military and National Science Foundation (NSF) funding. The synergy between the two projects, which were at once mutually reinforcing and competing, demonstrates the subtle but significant effects that government-supported research can have on computer technology.

System R

In the early 1970s, a group of IBM programmers moved from Yorktown to San Jose.[3] The group designed and built a prototype system to demonstrate relational ideas. Dubbed System R, this prototype was

intended to provide a high-level, nonnavigational, data-independent interface to many users simultaneously, with high integrity and robustness (Astrahan et al., 1976). The first phase of the project, in 1974-1975, produced a quick prototype to demonstrate feasibility, but its code was eventually abandoned. The next phase produced a full-function, multiuser version, which was evaluated in subsequent trials in 1978-1979. Perhaps the most lasting development to come out of the project was the Structured Query Language (SQL), now a U.S. and international standard for database access (Chamberlin et al., 1981; McJones, 1995).[4]

On its own, System R did not convince IBM management to abandon its existing product and replace it with relational databases. IBM and its customers still had strong vested interests in the established IMS technology. It took outside efforts, funded by the government, to prove that relational databases could become viable commercial products.

Ingres

In 1973, about when System R was getting started at IBM, two scientists at UC-Berkeley, Michael Stonebraker and Eugene Wong, became interested in relational databases. Initially, they raised money to design a geographic data system for Berkeley's economics group (the name Ingres, which stood for interactive graphics and retrieval system, reflects this legacy). In search of further support, Stonebraker approached the Defense Advanced Research Projects Agency (DARPA), the obvious funding source for computing research and development. Both DARPA and the Office of Naval Research (ONR) turned Ingres down, however; they were already supporting database research elsewhere.

Stonebraker then introduced his idea to other agencies, and, with help from Wong and Berkeley colleague Lotfi Zadeh, he eventually obtained modest support from the NSF and three military agencies: the Air Force Office of Scientific Research, the Army Research Office, and the Navy Electronic Systems Command. The experience of acquiring support for Ingres illustrates the importance of maintaining diverse funding sources within the government. When a researcher can propose a new idea to several potential supporters, it not only increases the chances of funding a good idea but also provides a crucial learning process as proposals are rewritten and resubmitted.

Thus funded, Ingres was developed, during the mid-1970s, into a prototype relational database system that was similar to IBM's System R but based on different hardware and a different operating system. Ingres went through an evolution similar to that of System R, with an early phase demonstrating an initial solution in 1974 followed by significant revisions to make the code maintainable. Ingres was then disseminated

to a small user community, both inside and outside academia, which provided feedback to the development group. The dissemination process was advanced by the proliferation of inexpensive DEC machines in universities. Members of the project team rewrote the Ingres prototype repeatedly during these years to incorporate accumulated experience, feedback from users, and new ideas. Ingres also included its own query language, QUEL, which was similar to, but still distinct from, IBM's SQL (Stonebraker, 1976, 1980).[5]

DIFFUSION AND COMMERCIALIZATION OF RELATIONAL DATABASES

Ingres technology diffused into the commercial sector through three major channels: code, people, and publications. Unlike the technical details of the IBM project, Ingres source code was publicly available, and about 1,000 copies were distributed around the world so that computer scientists and programmers could experiment with the system and adjust it to their own needs. Michael Stonebraker founded Ingres Corporation (purchased by Computer Associates in 1994) to commercialize the Berkeley code directly. Robert Epstein, the chief programmer at Ingres in the 1970s, went on to co-found Britton-Lee Incorporated and then Sybase. Both Britton-Lee and Sybase used ideas and experience from the original Ingres, and government agencies were early customers of both companies. Computer Associates released a commercial version of the Ingres code in the 1980s.

Continued movement of Ingres researchers throughout the database community spread the technology even farther. Jerry Held and Carol Youseffi moved from UC-Berkeley to Tandem Computers Incorporated, where they built a relational system, the predecessor to NonStop SQL. Until joining Kleiner, Perkins, Caufield & Byers in 1998, Held was senior vice-president of engineering at Oracle, where he headed that company's database efforts. Paula Hawthorn moved from Ingres to Britton-Lee (as did Michael Ubell) and eventually became a co-founder of Illustra Information Technologies Incorporated, now part of Informix. Stonebraker himself worked with Ingres Corporation, Illustra, and Informix. Other Ingres alumni went to AT&T, Hewlett-Packard Company (HP), IBM, and Oracle, bringing with them the lessons learned from Ingres. As Robert Epstein observed, "What came from Ingres was the experience of having built a prototype . . . to say what parts need to be done differently."[6]

The Ingres and System R development groups had a complex relationship that fostered a spirit of competition, as both groups worked on similar new technology. Both groups were relatively small and close-knit. Between 1973 and 1979, approximately 30 individuals cycled

through the Ingres group, which never contained more than five or six programmers. The System R group included roughly 15 persons who wrote code and papers on System R and later worked with IBM's product development groups to commercialize the technology. Several members of the IBM group had Berkeley connections, and UC-Berkeley sent summer students to IBM. Timely publication and proper allocation of credit for new ideas became paramount concerns (McJones, 1995). Ingres's QUEL was in competition with IBM's SQL as a query language; the latter eventually won out and became the industry standard.

The success of SQL transpired almost in spite of IBM, which could have taken advantage of its query language several years sooner than it did. Oracle, founded by Larry Ellison, developed and began selling an SQL-compatible product even before IBM had an SQL product in the market. Ellison had learned of SQL through publications by the System R project team. IBM was compelled to develop its SQL/DS system by the threat of competing products from other established database companies, such as Software AG, a German company. Other fledgling companies, such as Informix and Ingres, also introduced relational systems, with Informix embracing the SQL model. System R programmers influenced the industry personally as well as by their writing.[7]

System R and Ingres were not the only relational database efforts to spring from Codd's work. Other research at the University of Toronto, IBM in the United Kingdom, the University of Utah, and the University of Wisconsin made contributions as well. It also became clear that the relational model has limitations, particularly in handling complex data. In 1982, the Ingres project ended, and in 1985 it was transformed into Postgres at UC-Berkeley, which sought to extend the relational model to objects. This change coincided with DARPA hiring its first program manager for databases, who funded Postgres. This project became a component of the digital library and scientific database efforts within the University of California system.

A landmark year for the relational model was 1980, when IBM's SQL/DS product hit the market for mainframes, smaller vendors began selling second-generation relational systems with great commercial success, and Codd was awarded the Association for Computing Machinery's (ACM) Turing Award. The relational model had come of age.

Today, relational databases are but one way of accessing the multiple types of information computers can handle. Related research in information retrieval, multimedia, scientific databases, and digital libraries is under way, supported by DARPA, NSF, and the National Library of Medicine, among others. Still, the history of the emergence of relational database technologies, products, and companies reveals a good deal about innovation in computing and communications.

LESSONS FROM HISTORY

The federal government had important effects on the development of relational databases. The earliest days of this subfield suggest that government missions can create new markets for technology, providing incentives for innovation. The Census Bureau's need to conduct a decadal census supported the information processing industry before computers were created to automate it.

Later on, government funding hastened commercialization. An example is the case of System R and Ingres. The critical issue is not which one was more successful or influential in the long run, but rather, to paraphrase a System R team member, whether either project would have succeeded in the absence of the other. The academic interest legitimized System R within IBM, and Ingres was bootstrapped off IBM's commercial influence.[8] Competitive pressure, combined with the legitimacy bestowed on the relational model by government funding and academic interest, finally convinced IBM to sell relational database products. Were it not for the government-funded effort at UC-Berkeley, such databases probably would have been commercialized anyway, but later—and time-to-market is, of course, a critical factor with new technology.

That same example shows that the commercial interests of firms such as IBM can impede the continued development and commercialization of technologies that compete with existing product lines. IBM and its customers had vested interests in the established IMS technology and resisted change until external events proved that relational databases could become viable commercial products.

This case history also suggests that the large numbers of researchers passing through university laboratories, their willingness to share data and code, and their publication imperatives make university researchers ideal sources of technology transfer to the broader technical community. Industrial laboratories, by comparison, rarely place significant technologies directly into the public domain and have lower rates of personnel turnover, although they often benefit from greater and more stable supplies of resources. Especially in the computing industry, employees may take ideas into the marketplace on their own, but industrial laboratories are likely to publish only information that concerns completed projects or is not deemed critical to the company's vital interests.

Academic research is important for other reasons as well. Because it can push the cutting edge of technology and produce results that may evolve into commercially viable products, existing commercial suppliers never have a lock on advanced technology and are forced to respond to the marketplace of ideas.

Finally, in pursuing new ideas and new areas of technology, aca-

demic research projects can benefit from access to multiple funding sources within the government, as any individual sponsor may assess the value of a new idea from a limited perspective. Although DARPA and ONR declined to support Ingres, for example, the NSF and three other military agencies agreed to do so.

NOTES

1. For his work with IDS and the Codasyl group, Bachman was awarded the Association for Computing Machinery's A.M. Turing Award in 1973 (Bachman, 1973; King, 1983).

2. Edgar F. Codd, in an interview with a representative of the Committee on Innovation in Computing and Communications, February 7, 1997.

3. The group included Mike Blasgen, Ray Boyce, Donald Chamberlain, James Gray, Frank King, Leonard Liu, Raymond Lorie, and Franco Putzolu.

4. Donald Chamberlin, in an interview with a representative of the Committee on Innovation in Computing and Communications, February 4, 1997.

5. M. Stonebraker, in interviews with a representative of the Committee on Innovation in Computing and Communications, December 27, 1996, and February 26, 1997.

6. Robert Epstein, in an interview with a representative of the Committee on Innovation in Computing and Communications, March 19, 1997.

7. Kapali Eswaran left IBM in the late 1970s to form his own company, and its code eventually became part of HP and Cullinet products. Jim Gray moved from IBM to Tandem, where he worked on NonStop SQL, and he is now the senior database researcher at Microsoft. Franco Putzolu also went from IBM to Tandem, where he was a principal designer of NonStop SQL, and later went to Oracle as a senior database architect.

8. Donald Chamberlin, in an interview with a representative of the Committee on Innovation in Computing and Communications, February 4, 1997.