Panel: The Impact of Database Research on Industrial Products (Summary)

José A. Blakeley * Dan Fishman † David Lomet ‡ Michael Stonebraker §
Moderator: Daniel Barbará ¶

1 Introduction

We have witnessed the proliferation of articles in the news media addressing the state of research funding in industries and universities. Repeatedly we get news about industries cutting down their research departments. The government is also taking decisions that affect research funding. Recently, Congress gave the National Science Foundation an unequivocal order to spend 60% of its research budget on projects deemed relevant to national needs.

There is an ongoing debate between those who believe that investing in research projects that merely generate knowledge is not enough, and those who maintain that worrying too much about near future returns could curtail the entire innovation process.

All this carries a powerful message to researchers in industry and academia: "get relevant, or else!" As a consequence, researchers continuously find themselves wondering how to have an impact with the results they are producing. For industry researchers, this translates into how to transfer the new technologies to products that ultimately benefit their companies. Academia researchers have do deal with funding agencies that will increasingly demand to see the relevance of the ideas before they fund their projects. And, for both academia and industry researchers, this trend also means being more careful in selecting the topics of research.

This panel, held on February 17th in Houston, Texas, during the 10th Data Engineering Conference, attempted to address the issues of how to efficiently transfer technology and how to make our research more relevant. This paper summarizes the positions taken by each of our panelist members.

2 Blakeley's View

Work on the Open Object-Oriented Database System (Open OODB) in the Central Research Laboratories at Texas Instruments began five years ago with the vision that future enterprise information systems and, in general, large software systems will be built from open object services architectures. An *object services architecture* is a software architecture built from a collection of orthogonal (independent) software services connected via some form of software backplane or message passing bus. For example, some of these services may include persistence, transaction management, version and configuration management, distribution, and name management. One may notice that some of these services are usually found bundled within a DBMS. If the object services architecture is built out of services that can be freely composed, then one could configure many kinds of large software systems including distributed operating systems as well as object-oriented or relational DBMSs. Ultimately, one could configure a data management system that *unifies* RDBMSs, OODBMSs, and file systems thereby making database technology much more pervasive than it is today.

When we approached ARPA with this vision, they suggested that we complement it with a technology transfer (TT) story that would make the resulting technology more widely used. Hence, we developed a proposal that would enable TT not only from our laboratory to a particular user community or product group, but would also enable TT among university R&D and standards organizations (See Figure 1). Therefore, when we talk about TT we see these four communities as potential sources and targets for TT. Notice that the arrows connecting all these communities are bidirectional indicating a flow of TT in both directions.

In the context of the TI Open OODB project, our group participated in the ANSI X3 OODB Task Group whose purpose was to develop a characterization of

*Texas Instruments Inc., P.O. Box 655474, Dallas, Texas 75265
†Hewlett-Packard Laboratories, 1501 Page Mill Rd., Palo Alto, CA 94304
‡Digital Equipment Corp., Cambridge Research Lab, One Kendall Sq., Bldg. 700, Cambridge, MA 02139
§Montage Software Inc. 1111 Broadway, 2000 Oakland, CA 94607
¶Matsushita Information Technology Laboratory, 2 Research Way, 3rd Floor Princeton, N.J. 08540

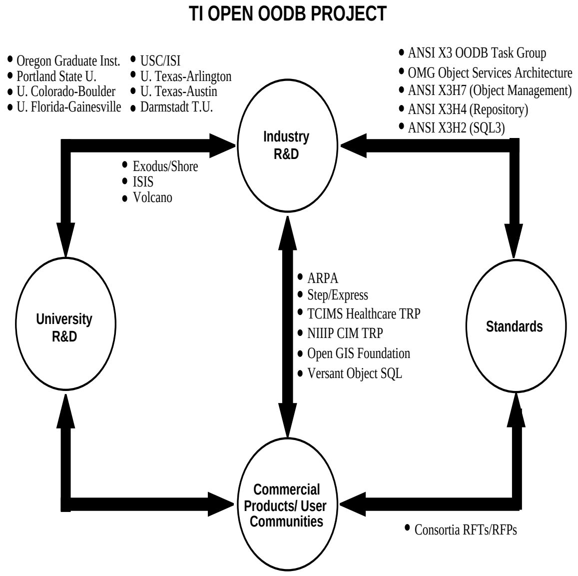

Figure 1: Technology Transfer Pipeline.

OODB technology. The final report of this group, characterized OODBs as a collection of independent services [2]. The report was co-authored by several key researchers and developers in industry which represented a form of initial consensus from the small number of companies involved in this effort.

This report was widely circulated in industry and eventually served as one of the inputs to the Object Management Group, influencing the specification of its current Object Services Architecture [3]. It also served as a basis for the creation of a new ANSI committee, X3H7, whose current objective is the formulation of a reference object model that can enable the convergence of many object models currently being developed independently.

At the same time we were carrying out the above TT activities, we were engaged in building an ambitious system which in itself was a challenge. We knew that our R&D group had technical strengths and weaknesses and that trying to build deep expertise in the weak areas was expensive and could take a long time. Hence, we decided to try to leverage as much as possible from research efforts at universities. To date, we have used the Exodus storage manager developed at the University of Wisconsin-Madison, the Volcano optimizer generator developed at University of Colorado-Boulder and now at Portland State University, and the ISIS distributed toolkit developed at Cornell University. These are all examples of TT from university R&D to our project.

We also envisioned that the Open OODB architecture and system could serve as a platform to facilitate database systems research in academia and in the ARPA community. In 1993, we produced two releases of Open OODB and distributed them to thirty selected university and ARPA sites which are providing us feedback in the areas where the system is deficient and on the appropriateness of the internal interfaces among modules or services of the Open OODB. At the same time, these sites are using Open OODB as as infrastructure backplane for their own research and development. Ultimately, we would like to see our system become or be part of a commercial product so that it becomes more widely used. Thus, there is still a significant amount of TT work to be done.

Experiences

Our experience indicates that successful TT cases share some of the following characteristics:

- *Technology vision drives the work.*
- *Technology transfer plan from project start.* TT is not an activity done after the fact as a way to hand off the results of your work or throwing the results over a wall hoping that someone will adopt it. A plan enables the formulation of strategies to *position* the technology in the appropriate communities being influenced. This positioning activity typically involves a lot of "hard work."
- *Champions of the technology at both ends.* Successful technology transfer requires the existence of champions at both the producer and the consumer ends of the technology. These champions may include a combination of technical, standards, or business people depending on the target community being influenced.
- *Close working relationship with customer.* Listening and working closely with the customer is essential to successful technology transfer. This often involves a physical collocation of people. The following are two quotes that appear to be true in TT: (1) "Technology transfer is people transfer"; (2) "Collaboration is technology transfer" [1].
- *Management of customer expectations.* Many TT failures are caused by unfulfilled expectations. Managing customer's expectations represents one of the main challenges for successful TT. Often, what the producer of the technology delivers is not what its consumer expects. The establishment of a *formal agreement* or *contract* up front between the technology transfer partners helps to ameliorate unrealistic expectations early in the process.

In summary, the TI Open OODB project is trying to build an object services architecture to enable the

the configuration of large software systems that could make database technology much more pervasive than it is today. We have had some TT successes, but a significant amount of work still remains to fulfill our vision.

3 Fishman's View

Over the period December 1984 through November 1989, HP Labs engaged in the research and prototyping of a next-generation, object-oriented DBMS, called Iris [1], that was being designed to meet the needs of commercial applications. Iris consisted of (1) an object manager supporting a rich object model, (2) a non-procedural query and object definition language, OSQL, for creating objects and object schemas, and for querying them in a language syntactically similar to SQL, and (3) support for multiple application development languages, such as C and SmallTalk.

Transfer of the Iris technology from HPL to the recipient HP product division began in 1989 and resulted in the product release of OpenODB [2] in November, 1991. We will refer to this transfer division as the Database Division (DBD). In reality, the transfer division went through a number of organization changes, and each time was identified by a different name. Up until the time of the transfer, DBD had two database product lines, a network DBMS called Image, and a relational DBMS called Allbase. OpenODB is HP's third database product.

The process of transferring the technology involved two main phases. The first phase entailed laying the groundwork for the transfer. This included establishing and maintaining good relationships with the transfer division, both at the management and technical levels. Divisional personnel were kept current on the status of our work, our technology vision, the associated technical challenges, and the potential for future products. Occasionally we organized HP-wide database forums to understand technology requirements from advanced users, to advertise our work, and to obtain feedback. In addition, we established a measure trust in our relationship with DBD with smaller technology transfers that addressed current product needs. At the same time, we attempted to create scientific credibility for our work through publication, conference participation, and university relationships. Finally, and very importantly, we tried to create market pull for an Iris-like product by making presentations and demos at trade shows, and in front of major HP customers or would-be customers. As a result, HP received many calls from companies expressing interest in HP's product plans for this technology. As we came close to the transfer, we and DBD installed and monitored three pilot applications with major accounts and listened very carefully to their assessment, which in the main was quite positive.

The second phase of the transfer was the actual technology transfer itself. Our goal was to do a good job at knowledge transfer. By mutual agreement, we first brought a core technical team from DBD to HPL to work on completing the final research prototype with the HPL team. This went on for 6-8 months. During this time we also developed comprehensive documentation and training on the system internals and externals, including twenty hours of videotaped courseware. The latter proved useful to DBD in training new members of the development team. Before and during the technology transfer, we participated in business planning with the DBD management team in preparation for an OODBMS product. When the development effort moved from HPL to DBD premises, the development team was seeded with four key Iris project members on a 1-2 year loan. In addition, HPL provided ongoing consulting as needed during the productization phase. As noted above, the initial productization phase ended in November, 1991 when OpenODB was released as an HP product.

Our experience with the Iris technology transfer contains many of the ingredients identified in a recent survey of HPL managers of those things they have found to help technology transfer. These include:

- Establishing trust and honoring commitments
- Exchanging people
- Holding project reviews with divisional partners
- Holding celebrations with combined teams
- Having a champion at both ends
- Communicating at all technical and management levels
- Providing thorough documentation
- Creating a visible, irresistible demo

In the same survey, the HPL managers also identified things they felt hindered transfer. These included:

- Transferring the technology with some hard problems still unsolved
- Harboring a not invented here (NIH) attitude
- Not having enough division people involved early on at HPL
- Not having enough HPL people involved later on at the division

- Having a long physical distance between HPL and the division
- Making a weak technical contribution
- Not gaining the division's confidence in the technology
- Not aligning the project strategically with the division

We feel that the transfer of Iris benefited from many of the positive factors noted above, while at the same time, we managed to avoid negative factors.

4 Lomet's View

Good industrial research focuses on marketplace and existing products. Academic research usually does not, and hence it rarely has impact on industry. However, much of industrial research is no more relevant than academic research. So the problem of having impact is not confined to academia.

Much is made of the NIH ("not invented here") syndrome, hence blaming potential research recipients for the problem. But this misses the main problem and is too pat in absolving researchers from responsibility. The problem is deeper than NIH.

4.1 Conflicting Goals

The root of the difficulty is that researchers and product engineers find themselves in dramatically different environments that enforce dramatically different goals.

4.1.1 Researchers

4.1.1.1 What They Do

What researchers do, judged objectively, is produce papers. Our literature bulges with them. Some contain real advances, but many are irrelevant or worse, actually impeding progress. They lead others astray with techniques that are worse than current practice or that are not complete solutions. Many researchers produce **techno-nibbles**. They begin with a small idea, which is then diced into several papers. Referees frequently fail to weed them out, giving excessive weight to novelty and having too much tolerance of complexity.

Referees are overly impressed with **syntactically correct papers**. These follow something close to the format: (i) introduction, (ii) background, (iii) main idea, (iv) analysis- sprinkled with equations, (v) performance results- sprinkled with graphs and tables, and (vi) a discussion explaining why the new technique crushes the prior methods. A bibliography cites the work of all likely referees. Some very good papers are syntactically correct. But syntax is not a substitute for real understanding. Papers should be judged by quality of ideas. That is hard, which is why few referees do it.

4.1.1.2 Why They Do It

Academic researchers produce techno-nibbles to build a tenure record. They continue this to be promoted or to move to a high status university. In the research community, too often stature is measured by number of papers. So, to impress friends requires a long vitae. This is very sad.

Some serious re-thinking is needed. University promotion committees need to look for impact instead of vitae length. Most database research is engineering and should further the engineering art. A good conference paper should count more than a journal paper, which all too frequently either was not accepted at a good conference or was.

4.1.2 Industry

4.1.2.1 What They Do

In industry, papers are rarely rewarded, so few are written. Engineers are expected to build and improve products. Improvements can be in functionality, robustness, performance, i.e., attributes important to real customers.

Industry wants the best practice, not the latest paper. Novelty is unimportant. A good twenty year old idea, like B-trees or two phase locking, is just fine. It wants simple ideas that fit with the current system. The closer the fit and the simpler, the better. Even simple ideas are complex to implement. Complex ideas may be impossible.

Industry wants high leverage ideas with great cost/benefit ratios. These ideas need not be publishable. For example, a great way to boost TPC-A performance is to support multi-statement procedures, hence reducing the number of application/system boundary crossings. The improvement is dramatic when code paths are short elsewhere, as with DEC Rdb.

4.1.2.2 Why They Do It

Engineers are mostly rewarded for product marketplace success. If the product makes money, raises and career advancement follow. This is how industry folks impress their friends.

4.1.2.3 Resulting Impediments

Since researchers are not measured on industrial impact, they frequently don't know the state of industrial art, which can lead the research literature. A goal of mine, as editor of the Data Engineering Bulletin, is to disseminate information about the state of the industrial art. Our December, 1993 issue on commercial query processing is an example.

Few engineers in industry really know the research literature, Also, groups with existing products want incremental improvements with small costs. They resist revolutionary technology, as network and hierarchical database groups resisted relational technology.

4.2 How Does Research Have Impact

The clash of environments is a serious barrier to the flow of good research into industrial practice, but the situation is not hopeless. Research has had and will have impact, but how is usually not very direct and takes too long for tenure seekers. Unfortunately, much research will be irrelevant to industry. Below are some ways in which research can have impact.

4.2.1 Conceptual Understanding

This is a unique opportunity for researchers. Product engineers are unlikely to view technology sufficiently broadly and abstractly to produce conceptual understanding. Theory, again from research, can be used to validate understanding and determine its limits. Historical examples are the relational model and transactions, and more recently, multi-level transactions.

4.2.2 Revolutionary Changes

Revolutionary changes are rare but have incredible impact. Existing product groups focus on improving their products, so revolutionary changes usually comes from elsewhere, e.g., researchers or startup companies. The shift to relational databases was revolutionary. Current opportunities are in multi-media data and the information utility. The bad news (for researchers) is that industry sees this. So researchers must move fast.

4.2.3 New Algorithms

Algorithms should be easy to transfer. But, a new algorithm for a previously solved problem must be clearly better. A successful (but ancient) example is B-trees supplanting ISAM. Unfortunately, many "new" algorithms are complex tweaks, worse than the current practice, or not complete and robust solutions. Impact is more likely in a new area, e.g., multi-attribute indexing, but the need for complete and robust solutions persists. And patience is required. Even extensible hashing has yet to appear in commercial systems.

4.2.4 Evolutionary Changes

Evolutionary changes are distinguished from the preceding by the need to consider the specifics of some system. Without this, the cost/benefit ratio cannot be computed. I've worked closely with DEC Rdb. Rdb is a data sharing system with distributed locking and specialized recovery. Few researchers understand data sharing systems and so cannot help.

4.3 How to Do Research with Impact

So how can you do research with impact?

- Impact must be important to you. You must work at it. So, **after tenure, think impact**.

- Others need a reason to read your papers. So, don't publish a bad paper. My favorite authors rarely publish bad papers. To be one, you need to provide a **techno-reward** in every paper.

- New functionality, whether revolutionary or major extension, requires credible demonstration, not only of technical feasibility but also of customer need. System R and Ingres did this for relational databases. This is lacking for recursive queries.

- Incremental technology must be clearly better, low in cost, simple, and complete. It helps to either know or to be a product engineer. Think industry sabbaticals. And be patient. Product priorities are largely driven by business needs, not technology.

The technology enterprise is a vast genetic algorithm. Like mutations, most ideas are either irrelevant or bad. Even good ideas have uncertain prospects. The standards are higher than for even the most selective conference. To succeed requires focus, effort, inspiration, patience and luck.

5 Stonebraker's View

Thoughts on Venture Capital Backing for a Company

Besides getting a large company interested in a potential product idea, one can alternately start a new company to exploit the new concept. There are two ways to finance such an enterprise:

- get a second mortgage

- get venture capital backing

Financing a software startup with personal money has two serious disadvantages. First, the feasible amount of money is limited; hence, inevitably progress will be slowed by having a smaller staff than desirable. Second, if one's personal resources are at risk, there is a tendency on the part of some people to lose sleep. On the other hand, this financing alternative is the only way to retain control of the company. On balance, I would recommend venture capital funding. This has the advantage that sufficient capital will be available to seize the market opportunity. Moreover, it brings into the company one or more backers who have years of entrepreneurial experience and connections to influential people. On the other hand, obtaining venture capital will almost assuredly leave control of the company in the hands of the venture firm(s).

To find venture capital it is highly desirable to have a prototype of the proposed product (i.e. demoware) and an engineering team in place who can build it. It is helpful but not always necessary to have other members of the management team identified. Although there are exceptions to the following rule, the lead technical person will typically be expected to be Vice President of Engineering, requiring the recruitment of a President, and Vice Presidents of Marketing, Sales, and Finance. In addition, the entrepreneur must write a business plan, detailing the market opportunity with sales projections and expected company expenses. These are constructed using "the wet finger in the air" technique, i.e. a best guess based on plausible assumptions.

The business plan should be presented to a small collection of potential investors (say 5). Generally speaking they will take a couple of months to make a decision. If feasible, it is sometimes advantageous to get the team moving ahead on the product in the meantime. Moreover, expect a "lemming effect", i.e. if one venture capital firm decides to finance the enterprise, then all four others will want to do so also.

The goal of any company backed by venture capital is to produce a saleable product as quickly as possible. Basically the fraction of the company stock that will end up in the hands of the backers is monotonic in the amount of capital required to get the company into a profitable business position. The general wisdom is to think incrementally, i.e. produce an initial product quickly and with low technical risk, and then plan to extend it based on customer feedback in subsequent releases. During this process the company should "kiss every nickel", i.e spend as little money as possible.

The biggest hurdle faced my many startup companies is the recruitment of a President. This one person will typically make the greatest difference between the success and failure of the enterprise. Hence, this decision should be very carefully thought out.

References

[1] BABCOCK, J.D., BELADY, L.A., AND GORE, N.C. The Evolution of Technology Transfer at MCC's Software Technology Program: From Didactic to Dialectic. *Proceedings of 12th International Conference on Software Engineering (ICSE)*, Nice, France, March 1990.

[2] FONG, E., KENT, W., MOORE, K., THOMPSON, C. X3/ SPARC/ DBSSG/ OODBTG Final Report. Sept. 17, 1991, available from NIST, email:fong@ise.ncsl.nist.gov.

[3] OMG 1992. *OMG Object Services Architecture.* C. Thompson (ed.) Version 6.0, published by Object Management Group, 1992.