Persistent Object Systems: From Technology to Market

Andreas Gawecki[1] Florian Matthes[1,2] Joachim W. Schmidt[1,2]
Sören Stamer[1]

[1] *Higher-Order* Informations- und Kommunikationssysteme GmbH
Burchardstr. 19, D-20095 Hamburg, Germany, www.higher-order.de
[2] Arbeitsbereich Softwaresysteme, TU Hamburg-Harburg
Harburger Schloßstr. 20, D-21071 Hamburg, Germany, www.sts.tu-harburg.de

Abstract. This text describes how persistent object system technology developed in European basic research is being used by a small German startup software company to realize innovative customer-oriented information services on the Internet. The description of this particular "entrepreneurial experiment" is intended to provide some input to the discussion of possibilities to stimulate the uptake of academic research results by industry in Germany.

1 Persistent Object System Technology Development

Between 1991 and 1995 an enthusiastic team of researchers, Ph.D. and master's students at the computer science department of the University of Hamburg created *Tycoon*, a programming environment for the construction of integrated persistent object systems. The vision that unified and shaped this team can be summarized best by the following quote from [Mat93, p. 1–6]:

- Persistent object systems are software systems that provide their users with a flexible and problem-oriented access to large collections of persistent objects with rich semantic relationships.
- The longevity, the user-orientation and the security requirements of persistent object systems lead to a strong need to utilize off-the-shelf services like object stores, information retrieval engines, transaction monitors, database systems, GUI toolkits, and distributed object managers.
- The quality of future persistent object systems will depend more on the flexibility, efficiency and correctness in the interplay between the objects of these heterogeneous *generic* services than on the performance of individual system services.
- The key to quality improvements are therefore
  - expressive naming, binding and typing mechanisms,
  - the integration of system and application programming, and
  - the use of abstract and formally-defined intermediate representations to enable open, scalable, portable, and optimizable system implementations.

Five years later, after the commercial success of Java, Visual Basic and other platform-independent "system integration languages", some of these statements might sound familiar, but even in 1997, these commercial environments fail to support the following important language concepts which are the cornerstones of the Tycoon system and language design.

Higher-order language design ensures that all abstraction principles (naming, typing, static and dynamic binding, parameterization, scoping, etc.) of the language are applicable uniformly and without restriction to all objects of the language (values, functions, types, modules, interfaces, processes, ...).

Polymorphic and strong typing is crucial for precise yet flexible service descriptions in a system integration scenario where generic libraries of multiple languages (C++, CORBA IDL, Java, SAP R/3 function modules, ...) need to be utilized in a consistent and safe manner. In particular, both, parametric polymorphism and (higher-order) subtype polymorphism have to be supported.

Orthogonal persistence makes it possible to support longevity for objects of arbitrary complexity and size including failure recovery and fully automatic memory management. In particular, code and processes are first-class persistent objects.

Orthogonal mobility enables unrestricted migration of data, code and active threads (processes) between multiple, possibly heterogeneous system platforms.

More details on the Tycoon system and its language can be found in two books [Mat93, Atk97] and in several academic papers that present specific innovative aspects of the Tycoon system [MS91, MS92, CMA94, MS94, MMS94, MMS95, GM96, MMS96b, MMS96a]. A number of master's theses demonstrate how the above three language and system design principles enable a seamless interoperation between Tycoon and commercial servers like Oracle, ADABAS-D, ObjectStore, O_2, information retrieval engines (WAIS, INQUERY), SAP R/3, NeWS, StarView, C and C++ libraries, Sun-RPC, DCE-RPC and Kerberos (visit [Tyc92] for more details).

The remainder of this paper reports on the technology transfer process that finally led to the exploitation of the Tycoon system technology for the construction of customer-oriented Internet information services by a small German software startup company.

This particular case study also provides some (subjective) answers to the following questions: What are options to commercialize such innovative system technology? What are promising markets for persistent object system technology? Is it possible for enterprises to gain a strategic advantage through this technology?

2 Technology Transfer in Computer Science: Options and Issues

Similar to the situation in engineering disciplines, the prime motivation of academic researchers and students in applied computer science is to achieve a better understanding of the models, construction principles and architectures for complex systems. However, they are also highly motivated to see an uptake of their newly developed models, techniques and architectures by industry to improve the working practice and to advance the state-of-the-art in their discipline.

At the beginning of 1996, the key members of the Tycoon team were convinced that their language and system platform had reached a level of maturity that made it a competitive candidate for the implementation of commercial persistent object systems. In particular, they strongly believed Tycoon to be technologically superior to established database programming environments like C++/SQL, PLSQL, Smalltalk, Microsoft VisualBasic/Access, ABAP/4, NATURAL for complex, data-intensive applications.

During their search for possible commercialization paths, they looked for advice from others who had past experience in the transformation of system-oriented research results into industrial products (modeling tools, databases, compilers, protocols, etc.).

As a first cut, the advice they got can be summarized by a rule of thumb, which states that the likelihood of a successful commercial uptake seems to be inverse proportional to the degree of innovation of a particular academic research result:

- It is rather simple to evoke commercial interest in incremental improvements of established technologies and of existing solutions (an optimized algorithm, a local add-on to a system, a conservative language extension, a gateway between two commercially relevant systems, a preprocessor).
  The main reason for this situation is the fact that there is already an established market with identified customers and producers which implies a reduced risk, a simplified marketing, a chance for support from existing producers on the market and a clear roadmap for product development.
- It is more difficult to find customers or venture capitalists who are willing to try out or to invest into new technologies which solve *new problems*.
  The main challenges here are to define the product(s), to identify potential customers, to convince potential customers of the benefits achievable via new product qualities, to implement the technology on multiple platforms, to set up distribution channels, etc.
- It seems to be extremely hard (impossible?) to promote technologies which require radical changes in languages, software development practice, or in overall system architectures.
  The past investment in existing data formats, tools, training, etc. and the necessity to touch operational systems make customers very reluctant to replace existing systems or manual business processes with technically superior system solutions. Moreover, the producers, consultants, vendors, ... of the

existing systems will fiercely fight against the potential new competitors, claiming that the features of the new solutions will be covered easily by future releases of their products. The story of relational and object database systems may serve as an excellent example of this kind of difficulties.

A decision for a particular commercialization path should take these basic market rules into account.

The following list is a collection of other advises by academic and industrial player to the Tycoon team on how to promote the commercial update of research results. It might be also of interest to other researchers who find themselves in a similar situation:

Write papers or books targeted at industrial readers. The best academic conferences and journals are not the best targets to influence industry.

Lobby in standardization bodies. There are task forces for everything (database languages, APIs, protocols). Fight to get your ideas in, even if standards alone don't change the world.

Give away your technology to other academics. Remember that Pascal, Modula, the X-Windows System, ... all started this way.

Sell your technology directly from the university. Even if there are only a few specialists who will initially use your system, you may gain a niche market this way.

There is no company which will adopt your technology. There are only very few potent technology providers in Europe who could be interested in your technology (e.g., SAP AG, Software AG, IBM in Germany). However, there you need somebody who will lobby internally for your idea. Application-oriented information and communication enterprises are only consumers of technology (e.g., Siemens AG, CSC Ploenzke AG, Systemhaus DEBIS AG in Germany) and don't run a risk. Small companies are the most innovative, but there you run into the "not invented here" problem.

Send your people to key development labs. However, if they are (PhD) students who just enter the job market, they may not be able to influence significantly the product development policies of their new employer.[3]

Do upper-management consulting. This is an important accompanying strategy. You learn more about the daily needs of potential customers for your technology and you have a chance to gradually influence long-term policies. If there is an additional knowledge transfer (e.g., at the student level), you may be lucky that some of your ideas a picked up.

Establish your own software company. Don't expect others to do the product development, marketing, maintenance, acquisition, etc. Do it yourself with the best people of your team. Find additional people with the necessary business skills and run the risk yourself. Remember that it is much harder to sell technology than selling services.

[3] One of the Tycoon project members is now with Sun Soft, adding persistence to their Java environment, a fact which may also serve as an indication of the relevance of persistent object technology to the future development of the Internet.

3 Higher-Order Services and Products

In the light of these advises, the authors of this paper decided in January 1996 to found their own software company, called *Higher-Order* GmbH, to fully exploit the potential of the Tycoon system technology but also to remain independent of the financial, technical or political constraints of an existing enterprise or technology transfer institution.

One year after its foundation, the work force of Higher-Order consists of five full-time employees, three part-time employees, and around ten students close to their exam who work two to three days per week at Higher-Order.

The business objective of Higher-Order is the design, the implementation and the maintenance of innovative software systems which enable customer-oriented information services on open networks like the Internet. Examples of such services are interactive product catalogues, personalized news feeds, internet shops, logistics information systems, customer help desks and customizable subscription systems.

Currently, the main customers of Higher-Order are large enterprises in the media industry which are the first to offer such demanding customer-oriented information services. However, other lines of business like providers of financial or insurance services as well as mail order and trading companies are already discovering the Internet as an attractive interactive media which makes it possible to add substantial value to their existing customer-oriented services.

The focus of Higher-Order is on IT consulting projects. In each of these projects (3-15 months), a specific customer-oriented information system is designed, implemented and installed in close cooperation with the customer who is an expert in his particular application domain.

The core of these systems (presentation services and business logic) is written in Tycoon-2, a commercial variant of the Tycoon language developed at the university. Moreover, these systems include Tycoon-2 gateways to legacy applications like editorial systems, news feeds, full-text databases or accounting systems written in other languages and running on heterogeneous platforms. Thereby, the newly created Internet services are integrated smoothly into the existing time-critical business processes of the information provider. Tycoon-2 applications act as servers for clients on the Internet and in Intranets.

Based on the feedback of its customers, Higher-Order's core competencies which differentiate the company from its competitors can be described as follows:

- innovation in the design of truly interactive customer-oriented services,
- ability to master complex legacy system integration tasks (databases, information retrieval, back-office systems like SAP R/3),
- expertise in "intelligent" text analysis and transformation (natural language text handling, dynamic document management, personalized presentation), and
- in-depth knowledge of the relevant standards (Internet protocols, document description languages, distribution formats of the media industry, evolving security and payment services)

Where necessary, this competence is complemented through a project-oriented cooperation with partners like Software AG (databases, mainframe connectivity), Sun Microsystems (Java and software development tools) and The Online Project (corporate web site design) to meet the needs of a specific customer.

Moreover, Higher-Order provides strategic consulting services to help enterprises (e.g., Verlagsgruppe Georg von Holtzbrinck, Rowohlt Verlag, Volkswagen Financial Services AG, Springer-Verlag AG) to better understand their role on the evolving Internet, to identify business opportunities and to assess the resources needed for specific Internet-based services.

4 Software Development with Persistent Object Technology

The company name *Higher-Order* GmbH alludes to the notion of "higher-order" models or systems which achieve a very high expressiveness with only a small set of abstraction principles by ensuring that any abstraction step produces a model or system which can in turn be the basis for an iterated "higher-order" abstraction step.

Similarly, a customer should view a software development project not only as the *consumption* of resources to achieve a specific project goal, but should ideally understand it also as an *investment* into his own future where any successful customer-oriented service will require fast, incremental and low-cost changes and additions which have to be supported by the underlying systems.

Despite the strong commitment to its it's innovative persistent object technology, Higher-Order de-emphasizes the importance of Tycoon for its projects and stresses the fact that other languages like C, C++, Java, Java-Script, Perl can be supported where needed. As mentioned already in the previous section, customers are initially rather reluctant to buy into such a new technology and its potential future benefits. On the other hand, customers value the tangible product advantages they gain through this otherwise "invisible" persistent object system (POS) technology:

- Customer-oriented services enabled by Higher-Order products are often the first of breed on the market. This is a crucial benefit on the very dynamic and turbulent Internet market. Clearly, Higher-Order benefits here not only from its POS technology, but also from an experienced development team that shares a common background of ambitious university projects.
- Services enabled by Higher-Order products are fully platform independent. For example, the migration of the operational DPA News-Box service from a Solaris to a HP-UX server only required a simple move of a single persistent object store file (including all structured data, code and processes) from one machine to the other without a need for re-compilation, re-linking or data re-loading.
- Services enabled by Higher-Order products scale well. For example, three newspapers of the Axel Springer Verlag are successfully using the same

Higher-Order search engine for classified ads with three substantially different workloads (number of customers, total number of ads in the database). The use of native threads at the operating system level and of a garbage-collected persistent store are two reasons for this "built-in" system scalability.

– Services enabled by Higher-Order products support smooth service evolution. Contrary to special-purpose languages or tools (e.g. Perl, WebSQL, HexBase, ...), Tycoon-2 is a mature, bootstrapped programming environment with the same expressiveness like Smalltalk or Eiffel. Unforeseen changes of the system requirements at a later point in time, like the need to manipulate new data types, to integrate other applications or to add concurrency to an application can all be accommodated without leaving the scope of the Tycoon-2 programming model and without disruption of existing services. The addition of styled advertisements (formatted text enriched with graphic images) to a text-only classified ad search service may serve as a concrete example for such a successful unforeseen service enhancement
– Services enabled by Higher-Order products can be blended with ease. All Higher-Order products utilize common (multiple inheritance and highly polymorphic) class libraries for bulk data types, Internet protocols, multi-media object management (MIME), Internet protocols, parser and scanner generators, SQL database access, gateways to information retrieval engines etc. General-purpose abstractions developed in individual projects (like electronic payment protocols) are added incrementally to these libraries. As a consequence, new combined or value-added services can often be created avoiding repeated development work.

5 A Cooperation Perspective: Research, Market and Education

It is certainly not possible to draw general conclusions regarding the commercialization of technology developed in academia after only one year of this particular "entrepreneurial experiment" has passed. On the other hand, from the personal perspective of the authors, the positive development of Higher-Order in its first year already compensated for a lot of the financial and personal risks implied by this particular commercialization path.

As an academic spin-off, Higher-Order views technology transfer not as a single event but rather as a long-term cooperative process between research, market and education. Through personal links to several academic institutions in Hamburg and in Germany, Higher-Order intends to foster a constant influx of creative new ideas, technology and well-trained people to sustain Higher-Order's core competencies as outlined in Section 3. For example, business process modeling tools and digital library technology will certainly be of relevance to next-generation Intranet and Internet services as requested by Higher-Order customers in the future. In particular, the globalization of its markets calls for a good working relationship to European and international research and development institutions. Higher-Order therefore closely follows the activities of the Pastel Working Group on Persistent Application Systems and Tools that brings together European, Canadian and Australian research groups and of the European Canadian working group on "Cooperative Information Systems".

Vice versa, the use of academic research results like Tycoon to satisfy concrete customer needs provides an additional feedback loop for academic research. This loop may help in identifying long-term research goals which have the potential to lead to technology contributions for relevant market sectors. As mentioned already at the beginning of the second section of this paper, "market relevance" is a strong incentive for researchers at all levels in applied computer science.

Finally, computer science and MBA students appreciate part-time jobs at high-tech startups companies like Higher-Order as a valuable complement to their more theoretical studies. They are thus able to develop practical skills using state-of-the-art tools and project management techniques and to get into contact with companies of their future job market.

Last but not least, Higher-Order is interested to learn from academic teams that master new, exciting technology which may give rise to new products or services. On the other hand, Higher-Order is prepared to make its technology available to academic institutions for specific research and development projects.

Acknowledgements

The academic development of Tycoon between 1991 and 1995 would not have been possible without the generous funding of the ESPRIT projects FIDE and FIDE$_2$. Being a member of the ESPRIT IDOMENEUS Network of Excellence in databases, multi-media and information retrieval contributed significantly to our understanding of the relevance of POS technology to multi-media information systems.

References

[Atk97] M.P. Atkinson, editor. *Fully Integrated Data Environments.* Springer-Verlag (to appear), 1997. This collection contains nine papers on the Tycoon environment.

[CMA94] L. Cardelli, F. Matthes, and M. Abadi. Extensible grammars for language specialization. In C. Beeri, A. Ohori, and D.E. Shasha, editors, *Proceedings of the Fourth International Workshop on Database Programming Languages, Manhatten, New York*, Workshops in Computing, pages 11–31. Springer-Verlag, February 1994.

[GM96] A. Gawecki and F. Matthes. Exploiting persistent intermediate code representations in open database environments. In *Proceedings of the Fifth Conference on Extending Database Technology, EDBT'96*, volume 1057 of *Lecture Notes in Computer Science*, Avignon, France, March 1996. Springer-Verlag.

[Mat93] F. Matthes. *Persistente Objektsysteme: Integrierte Datenbankentwicklung und Programmerstellung.* Springer-Verlag, 1993.

[MMS94] F. Matthes, S. Müßig, and J.W Schmidt. Persistent polymorphic programming in Tycoon: An introduction. FIDE Technical Report FIDE/94/106, Fachbereich Informatik, Universität Hamburg, Germany, August 1994.

[MMS95] B. Mathiske, F. Matthes, and J.W. Schmidt. Scaling database languages to higher-order distributed programming. In *Proceedings of the Fifth International Workshop on Database Programming Languages, Gubbio, Italy.* Springer-Verlag, September 1995. (Also appeared as TR FIDE/95/137).

[MMS96a] B. Mathiske, F. Matthes, and J.W. Schmidt. On migrating threads. *Journal of Intelligent Information Systems*, 8(2), 1996.

[MMS96b] F. Matthes, R. Müller, and J.W. Schmidt. Towards a unified model of untyped object stores: Experience with the Tycoon store protocol. In *Advances in Databases and Information Systems (ADBIS'96), Proceedings of the Third International Workshop of the Moscow ACM SIGMOD Chapter*, 1996.

[MS91] F. Matthes and J.W. Schmidt. Bulk types: Built-in or add-on? In *Database Programming Languages: Bulk Types and Persistent Data.* Morgan Kaufmann Publishers, September 1991.

[MS92] F. Matthes and J.W. Schmidt. Definition of the Tycoon language TL – a preliminary report. Informatik Fachbericht FBI-HH-B-160/92, Fachbereich Informatik, Universität Hamburg, Germany, November 1992.

[MS94] F. Matthes and J.W. Schmidt. Persistent threads. In *Proceedings of the Twentieth International Conference on Very Large Data Bases, VLDB*, pages 403–414, Santiago, Chile, September 1994.

[Tyc92] WWW home page for the Tycoon project. http://www.sts.tu-harburg.de/-projects/Tycoon/entry.html, 1992.