Japan's Fifth Generation Computers Project — A trip report

Ehud Y. Shapiro
Department of Applied Mathematics
Weizmann Institute of Science
Rehovot 76100, ISRAEL

1. Introduction

Last April Japan's Ministry of International Trade and Administration (MITI), in cooperation with eight leading computer companies, launched a research project to develop computer systems for the 1990's. The project, called the Fifth Generation Computers Project, will span 10 years. Its ultimate goal is to develop integrated systems — both hardware and software — suitable for the major computer application for the next decade, identified by the Japanese as "Knowledge Information Processing". Even though it may ultimately have applicable results, the current focus of the project is basic research, rather than the development of commercial products.

In addition to bringing Japan into a leading position in the computer industry, the project is expected to elevate Japan's prestige in the world. It will refute accusations that Japan is only exploiting knowledge imported from abroad, without contributing any of its own to benefit the rest of the world. Hence the project aims at original research, and plans to make its results available to the international research community.

I was the first non-Japanese researcher invited for a working visit to ICOT, the Institute for New Generation Computer Technology, which conducts the project. Due to the nature of the project I was given explicit permission, even encouragement, to report on everything I saw and heard during my visit; hence this report.

ICOT is located on the 21^{st} floor of an office building in central Tokyo. It currently hosts around 40 researchers, most of them on a 3-year "loan" from their industry-based research laboratories at Fujitsu, Hitachi, NEC, Matsushita, Mitsubishi, Toshiba, Oki, and Sharp.

The institute is divided into three research labs, responsible for research in hardware, basic software and applications software. The leaders of the laboratories are Drs. Munakami of Nippon Telephone and Telegraph, and Furukawa and Yokoi of the Electrotechnical Lab. Both NTT and ETL are government supported research

laboratories. The leader of the whole project is Kazuhiro Fuchi, also originally from ETL.

For me, the most intriguing aspect of the project is its commitment to build the Fifth Generation systems around the concepts of logic programming. In this report I will try to trace the roots and rationale for this commitment, and its possible implications.

Since I was exposed to a rather small circle of people in Japan and did not take any notes, this report may be biased. I regret if I unintentionally offended or misrepresented anyone. For a broader-based survey of the project the reader is referred to Uttal's article in last September's Fortune.

2. History

The logic programming approach was characterized recently by Maarten van Emden [11] — one of its founders — as:

- The use of logic to express information in a computer.
- The use of logic to present problems to a computer.
- The use of logical inference to solve these problems.

More technically, it can be be summed up in two "equations":

program = set of axioms

computation = proof of a statement from the axioms

The axioms typically used are universal axioms of a restricted form, called Horn-clauses or definite-clauses. The statement proved in a computation is an existential statement. The proof is constructive, and provides values for the existentially quantified variables; these values constitute the output of the computation.

The source of these equations can probably be traced back to intuitionistic logic and proof theory, but its first appearance as a practical approach to computing is a sequel to Robinson's unification algorithm and resultion principle, published in 1965 [8]. Several hesitant attempt were made to use the resolution principle as a basis for a computation mechanism, but they did not gain any momentum [3]. The beginning of logic programming can be attributed to Colmerauer, van Emden and Kowalski. Kowalski, in his 1974 IFIP paper [5], formulated the procedural interpretation of Horn clause logic. He showed that an axiom:

A if B1 and B2 and ... and Bn

can also be read as a procedure, where A is the procedure head and the B's are its body. Under this interpretation the axiom reads: to execute A, execute B1 and B2 and ... and Bn; or better: to solve A, solve B1 and B2 ... and Bn. Under this reading, the proof procedure is the interpreter of the language, and the unification algorithm performs the basic operations of variable assignment, data selection and data construction.

At the same time Colmerauer and his group at Marseille developed a specialized theorem prover, written in Fortran, which they used to implement natural language processing systems [9]. The theorem prover, called Prolog (for programming in logic), embodied Kowalski's procedural interpretation. Later van Emden and Kowalski baptized (circumcised?) the language, by providing it with operational, fixpoint, and model-theoretic semantics, and showed them all to be the same [2].

In 1976 Furukawa, today the head of ICOT's second research lab, visited SRI for a year. He got hold of a photocopy of a photocopy of ... of a photocopy of the sources of the Marseille Prolog interpreter. He brought it back with him to Fuchi, his boss at ETL. Fuchi was fond of Planner [4], and later of Kowalski's ideas, but did not believe they would result in a practical system. Nevertheless, he punched in the Prolog interpreter, inductively inferring some missing lines and unreadable characters, and made it work — and, to his surprise, work fast.

In 1979 some MITI officials thought it would be a good idea to start a new, revolutionary research project; they termed it the "Fifth Generation Computers Project". They consulted several people in academics and industry, including Munakami at NTT. Munakami recomended Tohro Moto-oka, from Tokyo University, and Fuchi as people worth listening to. What the MITI people did not know at the time (and perhaps they do not even know today) is that Munakami, Fuchi, and Moto-oka had spent a year together some twenty years ago at the University of Illinois at Urbana-Champagne, working on the ILLIAC-N, for some $N<4$.

Several informal committees gathered to discuss the nature of the Fifth Generation Computers. Simultaneously discussions with industry to solicit cooperation with the project began. Initially, the committees suggested that the Fifth Generation Computers will look like a Xerox's Dorado, and the Fifth Generation Computers network will look like an Ethernet. MITI detested this conservative, unoriginal answer, which was in total disagreement with the goals of originality and innovation.

Even though industry was rather reluctant to cooperate in a more ambitious project, MITI did not give up. It formed a rather large committee, headed by Moto-Oka, with specific instructions to come up with something more revolutionary.

This was the once-in-a-lifetime chance for the dreamers. Fuchi and other people in

ETL who were exposed to logic programming presented their views on how Fifth Generation Computer Systems should look: they should look intelligent, friendly, and — most important — be programmed in predicate logic.

The first step was to determine the target applications of the Fifth Generation Project. After some discussions, the committee agreed on the target of developing "Knowledge Information Processing Systems", which means, roughly, applied Artificial Intelligence. The next step was to choose the tools; at first there were strong Lisp proponents. Lisp is a successful programming language, and has an experienced user community in the US.

On the other hand, Fuchi and his group from ETL were deeply convinced that logic programming is a more promising research direction. They made great efforts to persuade the rest of the committee members to base the Fifth Generation project on logic-programming. They were encouraged by MITI's desire not to follow mainstream computer science research in the US, but difficulties overcoming the fear of their colleagues to make such a bold step into this mostly unexplored territory. What tipped the scale were several students at Tokyo University. They got excited about Prolog, and their enthusiasm swept along Moto-oka — the chairman of the Fifth Generation Committee — as well.

The resulting proposal — to develop knowledge information systems on the basis of logic-programming — was ambitious, went against mainstream computer science research in the US (cf. Ada), and provided a fertile ground for original research.

Around this time an international conference was called, to discuss "Fifth Generation Computer Systems". In this conference, held in October, 1981, the Japanese plan was exposed [7], and feedback from prominent researchers was solicited. The plan presented at the conference was rather baggy, faithful to the Japanese tradition of achieving consensus at almost all costs. As a result it elicited a rather mixed response from the West. Some went as far as saying that it is devoid of any technical content. But even though Ed Feigenbaum told them to forget the logic programming gimmick, the Japanese went on with their plan.

At that time I was aware only of the formal documents distributed by the committee, but I sensed the drama that was occuring behind the scene. I saw that the the official plan talked about "Logic programming machines, Lisp machines, abstract data-type support machine, relational database machine, innovative von-Neumann machines", but that the research papers presented at the conference were showing a slightly different, more focused picture.

The smoke cleared when ICOT was formed, with Fuchi as its director. With the

excuse of budget constraints, all ballast was dropped, and a clear, coherent research project emerged: to build parallel computers, whose machine language is based on Horn-clause predicate logic, and to interface them with database machines, whose data-description and query language is based on Horn-clause logic.

The fancy AI applications of the original proposal remain, serving as the pillar of fire that gives the true justification for building faster and better computers; but no one at ICOT deludes himself that in ten years they will solve all the basic problems of AI, an impression one can get from reading the original Fifth Generation proposal.

3. The Japanese logic programming manifesto

The battle to center the Fifth Generation project on logic programming was mainly political, but it had a clear ideology behind it. The rationale for the Japanese decision was given by Fuchi at the ACM 82 Conference in Dallas last October. The following summarizes the main points of his argument.

Before starting the project, the committee surveyed and studied various branches in computer science; among these are software engineering, databases, computer architecture, and artificial intelligence. The studies tried to find the direction in which the research in these fields was pointing; or, in Alan Perlis's words: "where do the gradients of computer science lead?". The results of these studies revealed — says Fuchi — that the gradients point to one direction: logic programming; that the concepts of logic programming can serve to unify recent innovations in all these fields; that logic programming is "the missing link between knowledge engineering and parallel computer architectures". The findings of the committee were as follows:

- In the field of software engineering, recent work places increasing emphasis on the importance of specifications and specification languages. Symbolic logic is almost universally accepted as the most suitable formalism for problem specification. The task of specification and verification is easier if the specification and the program use the same formalism.

  Program transformation work starts with a clear but possibly inefficient program and transforms it, preserving correctness, into a more efficient program expressed in the same formalism. Most research on program transformation was done in functional languages, but it extends quite naturally to logic programs.

- In the field of databases, logic programs can be regarded as a generalization of relational databases. The data-description language of a relational database is equivalent to logic programs without conditional axioms. Horn clause logic was shown to be adequate both as a query language and as a

language for describing integrity constraints.

One may view logic programs as the least general generalization of functional programming languages and relational databases.

- In the field of computer architecture, researchers have been investigating languages that will enable us to overcome the von Neumann bottleneck. These languages are called "single-assignment", since each variable in a program written in this language can be assigned a value only once during its execution. Logic programming provides a clean framework for designing a single assignment language.
- In Artificial Intelligence, one of the currently active areas is expert systems. These systems, so far mostly programmed in Lisp, contain a rule-base and an inference engine. The rules in the rule base are typically of the form

  A if B1 and... and Bn

  and the inference engine implements a deduction mechanism similar to that of Prolog.

  Another important area of AI research is natural language processing. Both extended attribute grammars invented for writing compilers and the generalized phrase structure grammars invented for parsing natural language can be viewed as notational variants of Horn clause logic[1].

There is another important argument, not mentioned in Fuchi's talk. Most problems of software engineering involve meta-programming: the development of programs that manipulate, analyze, and modify other programs. The effort in implementing meta-programming systems is in direct proportion to the complexity of the syntax and semantics of the object programming language. In the long run, the programming languages that will survive are the ones with a better programming environment. Logic programming languages have simple syntax, simple semantics, and are expressive enough to implement their own programming environment. An extreme example of the opposite approach is Ada.

---

[1]Indeed, the Prolog-10 system includes a macro-expansion facility that translates a context-free grammar into a logic program that functions as a recursive-descent parser. Hence the task of implementing a parser in Prolog does not involve much more than the debugging of the grammar of the language to be parsed.

4. Present

ICOT was formed last April. After the researchers assembled, the first thing they did was to take a crash course in Prolog programming. The goal of the course was to

help create a coherent research team. Even so, I still found some die-hard Lisp hackers in the project. We may argue whether it is better to think and program in Lisp, or think and program in Prolog; but both are certainly superior to thinking in Lisp and programming in Prolog.

Each researcher and research group in ICOT has a long range research theme and a short range one. The short range tasks are usually related to the Personal Sequential Inference machine (PSI). The development of the PSI machine is the immediate goal of ICOT. PSI is the Prolog counterpart of the Lisp machines. It is expected to provide user interface facilities comparable with the Lisp machines, have 1 or 2 megabytes of real memory, and perform around 20K LIPS[2].

The three groups divide the work as follows. The first group, headed by Munakami, is designing the PSI's hardware. The second group, headed by Furukawa, is providing the functional specification of PSI's machine language (Fifth Generation kernel language); the kernel language employs ideas of Warren's abstract Prolog machine [12]. The second group is also developing the Prolog to kernel language compiler. The third group, headed by Yokoi, is designing and implementing PSI's operating system, window system and programming environment.

The development of PSI is viewed mostly as a warm-up exercise, to get the different researchers familiar with the logic programming concepts via first-hand experience. It will also serve as a research tool on which Prolog applications will be implemented and upgrade, or replace, the DEC 2060 which is currently used by the institute.

---

[2]Logical Inferences Per Second, a fancy way of saying Prolog procedure calls per second. For example, micro-Prolog [6] for the Z-80 performs around 120 LIPS; Prolog interpreters written in C for the VAX are estimated to run 1 to 3K LIPS; Waterloo Prolog runs around 25K LIPS on large IBM machines; compiled Prolog-10 [1] runs 30K LIPS on a DECSYSTEM 2060, and is the best Prolog available today.

5. The visit

In an attempt to increase international cooperation and interest in the Fifth Generation project, ICOT will invite five researchers who work in related areas to its project every year. My visit was part of this program. I spent five weeks in Japan, four of them at ICOT. During the visit I gave talks on my research, heard presentations of ICOT research, and had discussions with ICOT members.

Most of the time, however, I did my own research using ICOT's facilities. I completed a design for a self-contained subset of the programming language Concurrent Prolog, implemented an interpreter for it in Prolog, and tested several Concurrent Prolog programs, which I had written in Israel previously as a thought experiment. These results are reported in [10]. I have also collaborated with A. Takeuchi, of the second research lab, and together we have implemented a simple window-system in Concurrent Prolog.

I felt very welcome at ICOT, and, as far as I know, had free access to all research carried on in the institute. My main contribution to the project during the visit relates to implementing multi-tasking on the PSI machine. In the initial planning, multi-tasking was done in a conventional way. As a result of the research I have done in ICOT, and discussions I have had with ICOT members, there is a possibility that the PSI machine will implement process creation, synchronization and communication in a way more suitable for logic programming, as done in Concurrent Prolog.

6. Future

Following the warm-up exercise — implementing the PSI — the experience gained from the first implementation will be applied to the design of the super-PSI, which will be a more powerful and, presumably, a better designed personal Prolog machine. In parallel, a logic-based database machine is to be built and interfaced to the PSI machines, probably using specially designed slots in the PSI's backplane.

The more distant goals and plans of the Fifth Generation project are described in their report "Outline of research and development plans for Fifth Generation computer systems", dated May 1982. They include the development of a system with the following functions:

- Problem solving and inference
- Knowledge-base management
- Intelligent interface

In the Fifth Generation systems the problem solving and inference mechanism corresponds to the CPU and main memory of today's computers; the knowledge-base management system corresponds to today's disk and file system; and intelligent interface corresponds to external I/O functions.

The systems programming, data-description, and query languages will be based on predicate logic. The target machine will perform 100M to 1G LIPS, and will comprise up to 1000 processing elements. The knowledge-base management system will have a

capacity of 100 to 1000GB. The intelligent interface system aims at a vocabulary of up to 10,000 words, 2000 grammar rules, and 99% accuracy in syntactic analysis of written natural language; a speech recognition system, capable of recognizing up to 50,000 words with 95% recognition; and a graphics system capable of storing and utilizing up to 10,000 pieces of graphics and image information.

These figures should be taken with a grain of salt, since they represent targets, which can be attained only by means not known today. I suspect that we will not find out the feasibility of these research targets before the project comes close to its conclusion.

As part of the attempt to create a more cooperative atmosphere around the project, and to make computer science research in Japan more accessible to the scientific community world-wide, an establishment of an international academic journal is planned. One of its proposed titles is "Journal of New Generation Computer Technology". The journal will invite contributions from outside Japan, and will probably publish research on all topics relating to non-von Neumann computer architectures, applicative and logic programming languages, and artificial intelligence.

7. The Japanese way

It is difficult to understand and evaluate the Japanese project without some understanding of the Japanese culture. I am certainly not an authority on this subject, but I have had a glimpse of it during my five weeks in Japan.

A key factor influencing the Fifth Generation project is the Japanese striving for consensus, and their tendency towards cooperation and community effort. Hence deliberating on the form of the project was a rather long and tedious process, but once a consensus was reached it was a rather stable one, and its continuation does not depend so much on a particular individual. The tendency to stick with the community may come as a hindrance when individual originality and creativity are concerned, but is of great value when research goals are clearly set, and attaining them requires cooperation and coordination. Hence I think that a group of Japanese "super-hackers", put to work together, is much more likely to achieve effective cooperation and communication then a similar American group.

Another important aspect is the deep identification of Japanese employees with their company and job. This identification is even stronger in a high-energy workplace like ICOT. At some alcoholically-rich occasion people mentioned, only half joking, certain traditional Japanese rituals they might perform if the project fails.

This devotion is reflected also in the less heroic daily routine: It is not uncommon to find people working at ICOT until 8 or 9 p.m. One is not terribly impressed seeing graduate students work like this; but these people are employees with families, who may have to commute one hour or so after work.

An American company that attempts to establish a high-quality research group faces every day the danger that a better paying or a better located company will attract its researchers. As a result, long-range personnel planning is impossible, and preference is given to short-term projects, which can keep the interest of the brighter researchers at least for a little while. On the other hand, MITI can trust that the human infrastructure being developed at ICOT — a team of researchers, trained in a new technology, that can address all aspects of designing, building and programming a new computer — is as solid as the building ICOT occupies.

8. Conclusion

People who believe in the unpredictability of scientific progress and revolutions find a planned revolutionary project to be almost a contradiction in terms. But sometimes ideology has to give way to reality: the Japanese project is both well planned and revolutionary. It did not invent the concepts of logic programming, but it is certainly the first, and perhaps today the only one, which grasped the immense potential of this approach, and gathered the critical mass of resources necessary to utilize it on a large scale.

There are thoughts and attempts throughout the world at responding to the Fifth Generation project, but I suspect that this battle is already won. The eventual success of the project will follow not from the amount of money invested in it, nor from the number of people working on it, nor even from the individual excellence of these people. It will follow from the coherent vision of its leaders, the genuine enthusiasm that they generate, and from the promising path of research they chose.

Any response to the project may match the amount of money or other resources invested in it, but will fail to to come up with the same sense of direction and devotion that holds the Fifth Generation project together. One such example is the British response, which basically says: Let's keep doing what we do today, but with more money. Money will increase the progress of research, but by itself will not result in a new generation of computers.

The Fifth Generation project faces two dangers: one is that it will succeed too late; the other is that it will succeed too early. If several years pass before any applicable

results come out of the project, it may be decided to dissolve ICOT in its current form, and to continue Fifth Generation research in the industry-based research labs, from which the ICOT researchers came. Because of the crucial role of its leadership, this will effectively mean the death of the project.

If the project succeeds too early, industry may realize that what is going on in ICOT is "hot stuff", and pressures to convert the Fifth Generation project into a classified one will mount. This will undermine MITI's original goal of increasing Japan's prestige in the world and rebuilding its image. Since the project is currently declared as open and solicits international cooperation, such a step may be considered a betrayal of confidence, an exploitation of foreign knowledge of a kind even worse than what Japan has been accused of in the past.

References

[1] D. L. Bowen, L. Byrd, L. M. Pereira, F. C. N. Pereira and D. H. D. Warren.
PROLOG on the DECSystem—10 User's Manual.
Technical Report , Department of Artificial Intelligence, University of Edinburgh, October, 1981.

[2] M. H. van Emden and R. A. Kowalski.
The semantics of predicate logic as a programming language.
Journal of the ACM 23:733-742, October, 1976.

[3] C. Cordell Green.
Theorem proving by resolution as a basis for question answering.
In B. Meltzer and D. Michie (editors), *Machine Intelligence* 4, pages 183-205. Edinburgh University Press, Edinburgh, 1969.

[4] Carl Hewitt.
Description and Theoretical Analysis (Using Schemata) of Planner: a Language for Proving Theorems and Manipulating Models in a Robot.
Technical Report TR-258, MIT Artificial Intelligence Lab, 1972.

[5] Robert A. Kowalski.
Predicate logic as a programming language.
In *Information Processing* 74, pages 569-574. North-Holland, Amsterdam, 1974.

[6] Frank G. McCabe.
Micro–PROLOG programmer's reference manual.
Logic Programming Associates Ltd., London, 1981.

[7] T. Moto-oka et al.
Challenge for knowledge information processing systems (preliminary report on fifth generation computer systems).
In *Proceedings of Internation Conference on Fifth Generation Computer Systems*, pages 1-85. JIPDEC, 1981.

[8] J. A. Robinson.
A machine oriented logic based on the resolution principle.
Journal of the ACM 12:23-41, January, 1965.

[9] P. Roussel.
Prolog: Manuel Reference et d'Utilisation.
Technical Report, Groupe d'Intelligence Artificielle, Marseille-Luminy, September, 1975.

[10] Ehud Y. Shapiro.
A subset of Concurrent Prolog and its interpreter.
Technical Report TR-003, ICOT — Institue for New Generation Computer Technology, January, 1983.

[11] M. H. van Emden.
Community News and Events section.
Logic Programming Newsletter (4):11, 1982.

[12] David H. D. Warren.
Implementing Prolog — Compiling Predicate Logic Programs.
Technical Report 39 & 40, Department of Artificial Intelligence, University of Edinburgh, 1977.

Shapiro 博士略歴

・氏名: Ehud Y. Shapiro

・所属: Department of Applied Mathematics
Weizmann Institute of Science
Rehovot 76000
Israel

・履歴:
the Faculty of the Graduate School, Yale University,
May 1982, Ph. D.

・研究歴: 研究発表; 論文, コンサルタント, 訪問歴