# Evaluation of the FGCS Project

David H. D. Warren
Department of Computer Science
University of Bristol

## General Impact

The FGCS project had a major political impact from the time it was first announced. The originally described plan was rather broad and fuzzy, with some apparently grandiose objectives, and its announcement generated a lot of hype. It was some time before Fuchi's clear and far-sighted vision of future computer systems, in which logic programming would provide the central link between parallel architectures and knowledge processing applications, became widely understood. Many international developments were stimulated by the perceived "threat" of the FGCS project, including Alvey in the U.K. and MCC in the U.S.A. Other international developments were more directly inspired by the scientific vision of the project, and included the setting up of institutes such as SICS in Sweden and ECRC in Europe which were very much overseas counterparts of ICOT with very similar research directions.

Overall, the project has had a major scientific impact, in furthering knowledge throughout the world of how to build advanced computing systems. It certainly provided a tremendous boost to research in logic programming. In a real sense, FGCS has become an international research effort. This clearly has enhanced Japan's international prestige. The project has also led to Japanese researchers becoming far more "plugged in" to the international research community than they were at the time of the project's announcement. A further general benefit of the project to Japan must surely be the transfer to Japanese industry of research ethos and experience, provided by staff returning to their home companies after their three-year assignments to ICOT.

## Organisational Issues

The project appears to have been handicapped, in tackling its very ambitious research goals, by being set in a framework more suited to an industrial development project. Ten years of basic research cannot be tightly laid down in advance, as much of ICOT's programme seems to have been, with its fixed duration, phases, milestones and hardware deliverables. The inflexibility of ICOT's programme seems to have prevented the possibility of changes of direction and reevaluation that are necessary in an advanced research project.

The research leaders of ICOT are scientists of very high calibre much respected by their international colleagues. Most of them have been with the project for its duration and have provided continuity of direction. Most of the other ICOT staff have been working on three-year assignments from the companies. The resulting lack of long-term continuity of ICOT staff, and the fact, as I understand it, that ICOT could not hand-pick the majority of its staff, are additional handicaps to pursuing advanced research not shared by comparable institutions such as ECRC, MCC and SICS.

## Major Technical Achievements

The FGCS project has produced many significant technical achievements. Some of the major accomplishments which are of particular interest to me and which I would personally highlight are the following.

First, ICOT has achieved its foremost concrete objective of building a parallel inference machine with a performance exceeding 100 megalips. Given the state of the art at the time the project was announced, when Prolog performance was at best 40,000 lips and large-scale parallel machines hardly existed, this achievement is quite remarkable and should not be underestimated. Although KL1 lips are not quite as powerful as Prolog lips, ICOT's achievement still represents a leap forward by more than three orders of magnitude.

On the language side, I consider GHC to be a most significant contribution. It embodies, in my opinion, the most elegant encapsulation of the committed-choice language (CCL) concept, simplifying and clarifying what was introduced by Parlog and Concurrent Prolog.

In its parallel implementations of KL1, ICOT has significantly advanced the implementation technology for CCLs. My own group has drawn on this work, and on the key idea of GHC, in our implementation of Andorra-I.

Although I have some reservations about that ICOT has committed itself entirely to CCLs and the concurrent logic paradigm, it cannot be denied that ICOT's PIM machine and operating system PIMOS are a powerful demonstration of what is possible in terms of building a machine and operating system entirely based on a CCL. It strikes me as something of an heroic feat, akin to climbing Everest or putting a man on the Moon, which opens our minds to future possibilities while perhaps not bringing immediate economic benefit.

As part of its programme for producing demonstrations of KL1 and PIM, ICOT has created a number of innovative parallel symbolic applications, notably in the areas of VLSI CAD, molecular biology, natural language analysis, and theorem proving. For me, they are particularly interesting in showing the potential for parallelism in algorithms very different from the kind of regular and repetitive numeric computations which are typical of parallel computing today.

## Technical Issues

There are some specific technical issues on which I would criticise the approach taken by ICOT. While I can appreciate some of the reasons why ICOT took the path it did, I feel the project might have achieved more, and remained closer to its original vision, if certain key decisions had been made differently.

Perhaps the most important issue is the decision (or assumption?) that parallelism has to be to be expressed explicitly in user programs, rather than designing systems to exploit parallelism automatically (taking advantage of the fact that logic programming, as a declarative formalism, allows parallelism to be expressed implicitly). Requiring the user to take direct responsibility for expressing parallel algorithms adds greatly to the programming burden, especially for the kind of complex knowledge processing applications which are the main target of FGCS. This route is only appropriate for problems which are computationally very intensive and where adequate performance cannot be achieved by other means. But for such problems, the first priority before tackling parallelism is probably to ensure that the sequential algorithm is as fast as it possibly can be, using as low-level a language as is necessary. This tends to argue against using a high-level approach such as logic programming.

On the other hand, there are many problems which may be potentially speeded up by exploiting implicit parallelism automatically, and where logic programming may provide reasonable performance (perhaps via the parallelism) in relation to software development cost. If parallel computers become the norm, as seems technologically inevitable in the near future, software systems which can exploit parallelism automatically will have a major role to play. It is a pity ICOT didn't take the opportunity to pursue this direction, which is being actively explored by other research groups (including my own).

The decision to go for explicit parallelism was linked with the decision to adopt the concurrent logic programming paradigm as central to all aspects of the project. In particular, all user programs in practice have to be expressed in, or implemented via, the concurrent LP paradigm, by means of the kernel language KL1. While the concurrent LP paradigm is of considerable interest in its potential for formalising interactive systems, and may be appropriate for many purposes including implementing operating systems, it is not, in my opinion, suitable for most user programs.

For most user programs, a much more high-level approach is needed, and ideally one would like to use declarative logic programming, i.e. logic programming as it was originally conceived. In declarative logic programming, the program expresses a declarative view of the problem as well as providing an operational solution to the problem. By contrast, the concurrent LP paradigm provides no declarative view of the problem. At best, it can be said to consist of a declarative description of a concurrent algorithm for solving the problem. In practice, users of the paradigm take an exclusively operational view. Without the declarative underpinning, there is no particular reason to maintain the original connection with logic, and every reason to modify the formalism to make it better fit its operational purpose. For these reasons, it is arguable whether concurrent LP is indeed logic programming in its original sense.

Be that as it may, the present situation with ICOT systems is that the main user language, KL1, is considerably lower level than traditional logic programming languages such as Prolog. Other, more high-level, user languages have been provided, but have had to be implemented on top of KL1. Although ICOT believes the use of KL1 as an intermediate language does not entail any unacceptable overhead, there seems good reason to believe that higher level languages and inference systems (including Prolog for example) could be implemented much more efficiently if a lower level implementation language than KL1 were used. In my view, KL1 is too low-level as a user language for most purposes, but too high-level to serve as the lowest level implementation language.

For a kernel language based on logic programming to be acceptable as a general user language it must, in my view, provide at least the basic capabilities of Prolog. This certainly seemed to be the view in the original FGCS proposal and in the early stages of ICOT's work. KL1, however, is considerably weaker than Prolog in that it does not provide a builtin search mechanism for finding at least one (and possibly all) solutions to a problem, although it is more powerful than Prolog in that it provides builtin coroutining (necessary, amongst other things, to support the concurrent LP paradigm).

It should be noted that it would be possible to have a kernel language providing all the capabilities of Prolog together with all the essential features of KL1 (including at least all of flat GHC which is the heart of KL1). Such a language would be quite acceptable as a user language, while providing the necessary basis to implement an operating system according to the ICOT approach. Such a language is provided by the Andorra-I system implemented by my group at Bristol. This language is viewed primarily as a high-level extension of Prolog. However, since it includes flat GHC as a subset, it is capable of supporting the concurrent LP paradigm.

Another most important issue, of a completely different nature, is the question of whether ICOT was wise to concentrate so much effort on building specialised hardware for logic programming, as opposed to building, or using off the shelf, more general purpose hardware not targeted at any particular language or programming paradigm. The problem with designing specialised experimental hardware is that any performance advantage that can be gained is likely to be rapidly overtaken by the ever continuing rapid advance of commercially available machines, both sequential and parallel. ICOT's PSI machines are now equalled if not bettered for Prolog and CCL performance by advanced RISC processors. And it seems very possible that commercial multiprocessors such as Sequent Symmetry, the new Butterfly, and other recent machines could come close to equaling the PIM performance if ICOT's software technology were ported to those machines.

A subsidiary issue is whether it was necessary to target KL1 so much at distributed memory hardware, with all the attendant problems of achieving good locality of communication and good load balancing, rather than adopting a virtual shared memory approach, for which scalable solutions are becoming increasingly well developed, including ones supporting a quasi-UMA (uniform memory access) model (c.f. KSR-1 and the closely similar work on DDM that I have been involved in). In general, I feel that ICOT perhaps devoted too great a proportion of its effort to developing hardware and operating systems, and could perhaps have focussed its efforts more on the knowledge processing software and applications which were central to the original conception of the project.

This section of my report is rather long! Its length should be interpreted not so much as a measure of criticism of ICOT's approach, which given the many constraints they were operating under has been highly commendable I believe, but rather as a measure of the complexity of the issues that I felt needed to be mentioned.

## Overall Evaluation

The nature of the original FGCS announcement raised a lot of expectations that the project could never have been satisfied and certainly have not been satisfied. Unfortunately, this makes it difficult for the project to be judged a success by the world at large, which includes most of the media. However, I strongly believe that overall the project has been a considerable success, and I think most fair-minded and properly informed observers will share my view.

The project was a major success in galvanising worldwide activity and more importantly for its scientific impact in stimulating worldwide research in new directions inspired directly by the FGCS vision and ICOT's work. The project has also succeeded in achieving its main concrete target of 100 megalips plus, an outstanding accomplishment that shouldn't be diminished with the benefit of hindsight.

But above all, any research project such as FGCS should be judged in comparison with comparable efforts by comparable institutions elsewhere. I believe the specific research and development achievements of ICOT are certainly on a par with the three institutions, MCC, ECRC and SICS, which are most comparable with ICOT and which are representative of the very highest level of computing research in the world. Moreover it should not be forgotten that those three institutions came into being largely following in the footsteps of ICOT and the FGCS project.

## Recommended Future Steps

I strongly recommend that ICOT's work should be continued in some form beyond the 1993 official end date of the FGCS project. The nucleus of highly gifted people and expertise built up at ICOT should not be allowed to evaporate, but should be continued within a smaller and more flexible framework. The KL1 software should be made available on widely available standard hardware, including Unix uniprocessors and multiprocessors such as Sequent Symmetry and perhaps BBN Butterfly. The PIM hardware should be examined to see whether it might potentially form the basis for commercial products if standard languages and operating system were supported. More effort should be put into evaluating the FGCS results, and especially in comparing the performance and usability with the best conventional alternatives. Speedups and good load balancing are not enough by themselves; one needs to show that applications perform better than they would by other approaches with comparable implementation effort. There should also be continuing research, especially in the areas of knowledge processing and applications. I would suggest that all this would best be done within a much smaller research institute, with selected long-term staff, and a focussed but flexible ongoing research programme (c.f. for example SICS).

It is understood that MITI is anxious to have official overseas collaboration in any extension of the FGCS work. My own group would be interested in collaborating with ICOT (or its successor) in evaluating ICOT's parallel applications developed in KL1, to see to what extent the same problems can be solved through more directly declarative logic programs, and whether comparable performance and parallelism can be obtained from logic programming implementations supporting implicit parallelism (such as Andorra-I). Unfortunately, DTI (the UK counterpart of MITI) requires 50% funding from UK industry for any research it supports. So long as the ICOT work is only available on custom hardware, it is unlikely that UK industry would be interested. And even if the ICOT software were ported to standard hardware, the likely payoff from such research is too long-term for most UK industry (with its rather short-term horizons). Therefore, I an afraid the chances of official UK involvement, through DTI, in continuations of the FGCS work seem poor, for the near term at least.

David H.D. Warren
Department of Computer Science
University of Bristol, Bristol BS8 1TR

Phone: +44 272 303322
Fax: +44 272 251154

---

David H.D. Warren is a Professor of Computer Science at the University of Bristol. His current research interests are in parallel logic programming systems and parallel computer architecture. He is well known for his work on Prolog (including the original DEC-10 compiler, the Edinburgh syntax, the Warren Abstract Machine, and Quintus Prolog). More recently, his research has centered on the development of the parallel Prolog systems, Aurora and Andorra-I, and the scalable multiprocessor architecture, the Data Diffusion Machine. He has a BA in Mathematics from the University of Cambridge, and a PhD in Artificial Intelligence from the University of Edinburgh.

---