FGCS Project Evaluation Report

Keith L. Clark

Imperial College
London, UK
5th June 1992

This report is an expanded version of the short presentation that I gave at the FGCS Project Evaluation Workshop on June 3rd in Tokyo. It is based on knowledge of the project gained since my first visit to ICOT in 1983, on the ICOT reports and associated presentations of the first two days of the FGCS92 conference, and on the presentations and discussions of the evaluation workshop.

Interest and involvement with the FGCS project

Let me say at the outset that I was very pleased and honoured to be invited to take part in the evaluation workshop. As one of the originators of the concept of committed choice concurrent logic programming I have had a vested interest in the project, which in the latter stages became a huge experimental investigation into the utility of this variant of logic programing. Your adoption of this special form of logic programming has been criticized. It is said to be too low level, too far away from the ideal of high level declarative programming. The retort I have always given is that at the time you made the decision to fix on such a language as your kernel language for parallel machines, no other proposed type of logic programming language, which one could hope to implement efficiently on a multiprocessor machine, enabled one to program the concurrent communicating processes needed in an operating system for a parallel machine. Quite correctly, you had the programming of the operating system, in the LP language of the machine, as a major goal. In addition, and as you have shown, I have argued that more declarative logic program languages can be supported on top of such a language. (I am now inclined to agree with the view expressed at the evaluation workshop by David Warren that the recently proposed Andorra extension of Prolog, or the Pandora extension of Parlog, are higher level languages which today would be a better starting point for a PIM kernel language than FGHC. But that is with hindsight.)

Interestingly, three people (Giles Kahn, Alan Robinson and Tony Hoare), who had an influence on the conception of committed choice concurrent logic programming, are attending FGCS92. Giles Kahn, in his 1977 IFIP paper with Dave McQueen on a stream communication model of parallel programming, set me and Frank McCabe on the track of exploring coroutining and pseudo parallelism, with incremental communication via shared variables, in IC-Prolog. It was our inability to see how we might efficiently implement IC-Prolog on a multi-processor that was the motivation to find an alternative approach. Then, on a semester visit in 1980 to Syracuse University, at the invitation of Alan Robinson, Steve Gregory and I decided to look at Hoare's CSP for inspiration. The concept of guards and committed choice non-determinism of that language (an idea previously proposed by Dijkstra for his guarded command language) seemed to us just the concept needed to allow efficient implementation of and-parallelism with stream communication in a logic programming language. This lead to the so called Relational Language, which merged the committed choice, communication only on committment, concept of CSP with the equally elegant stream communication model of Kahn and McQueen.

Because of our early work on concurrent LP languages, Steve Gregory and I were invited to ICOT in 1983. (I was pleased to hear in his Monday conference presentation, that Koichi

Furukawa had read with interest our 1981 paper on the Relational Language and, even before the start of the FGCS project, had considered using a concurrent logic language rather than an or-parallel Prolog as the PIM kernel language.) Our 1983 visit coincided with the second ICOT visit of Ehud Shapiro, the originator of Concurrent Prolog, which was based on but significantly extended the Relational Language. I believe that between the three of us, we helped convince Koichi Furukawa and his colleagues that adopting a concurrent LP language as the PIM kernel language was a sound approach. During that visit Steve Gregory and I crystalized our views on the essential features of Parlog, our successor to the Relational Language.

Since 1983 I have briefly visited ICOT twice, in 1985 and 1990, and had papers in both the 1984 and 1988 FGCS conferences. Colleagues Ian Foster and Jim Crammond, working on programming environments and implementations for Parlog, have both been invited to ICOT. Over the years there has been much exchange of views between ICOT and the far smaller Parlog Group at Imperial. The meta call of Parlog, introduced into Parlog by Steve Gregory and I on our 1983 visit to ICOT, is very similar to the shoen of KL1. Both are used to support the programming of operating system functions. Hence my vested interest in the project, and my earnest wish that it be perceived to be the great achievement that I believe it is. If some of my following remarks appear to be critical, they are intended as constructive criticism. They represent what I consider needs to be done to convince a skeptical world that there are significant results and achievements in the FGCS project of which the world *had better take note.*

Impact of the FGCS project

Let me begin by saying some positive things about the impact which the project has had outside Japan.

Firstly, it made Japan pacemakers in logic programming research and a country whose research into LP and its AI applications had to be taken seriously by the international AI research community. In addition, by the spin offs and interest in computer science research that it has generated in Japan, it has also made the country a force in CS research. You have also, through rotating industry researchers through the hot house of ICOT, trained a new generation of computer scientists and engineers into techniques of advanced research. I and others have observed with pleasure the maturing of the young scientists that were nurtured by ICOT. They are now well able to hold their own in the international research community and to explain their ideas effectively and clearly. Many have remarked to me at this conference on the quality of the presentations, especially those from ICOT researchers. ICOT staff and associated researches have not only had an impact in the fields of LP language design, programming methodology and implementation, they have made significant contributions in all areas of logic programming.

Outside Japan the FGCS project stimulated a great deal of research activity by both universities and industry, and it unlocked significant government funds to support this research. The UK Alvey and EC ESPRIT programs almost certainly would not have started, or would have been funded at much lower levels, were it not for the FGCS project. Nor would the industry supported MCC, ECRC and SICS research institutes have been formed. For this stimulus to CS research, thank you. I personally owe my chair at Imperial, certainly the fact that I got it in 1987, to this increased activity and respect for LP research that followed the announcement of the project.

The FGCS project had a significant effect on the amount of research activity and perceived importance of both AI and LP research. The IJCAI 1986 conference in LA and the ICLP 1986 conference in London have not had higher attendance or greater interest from

industry. This interest was a direct result of the excitement and interest that the FGCS project aroused.

Some comments on the "Achievements of the FGCS Project" report

I shall now offer what I hope is some constructive criticism via some comments on this short (two page) report that you gave me to look at as part of the evaluation.

In this report you make several claims which I believe to be true but as yet unproven in the eyes of the world outside Japan, a world that is perhaps uncharitably looking for any excuse to claim that the project was a failure.

"Thus, .. (KL1) makes it possible to quickly develop application programs which make full use of parallel machines with hundreds to thousands of processors."

To convince the rest of the skeptical world you have to properly document KL1 and the KL1 programming methodogies that you have developed. You have an impressive range of applications on display at the exhibition and described in the conference, but the program level anatomy of the applications is not adequately described. Programming a large application in Prolog is not easy for beginners, programming in any parallel language is worse. The LISP or C++ hacker building AI applications will believe that programming in a concurrent LP language must be near to impossible. Of course they are wrong, but convince these AI application developers that you do have an easy to use language and good application development support tools. Describe them much more fully, in clear English. Show step by step how to develop a model application. I know that writing such documentation is an onerous ask, for which researchers have no appetite. But it needs to be done, perhaps by writers skilled in the art of technical documentation who have been shown how to use the software, and who are helped in the task by its developers and expert users. Your experiments in quickly building highly parallel applications need to be repeatable, by people who have not helped develop the technology or been subcontracted by ICOT to do it, if the truth of the above claim is to have the impact that it should.

Doing such documentation is also necessary if the excellent policy of making the software freely available is to have any effect.

Under the time and resource constraints that you had, I do not believe that you could have done such documentation before now. Indeed, many of the tools and methodologies would only recently have been developed. But please seriously consider doing such documentation in the final nine months of the project, or in the first year of a follow up project.

"(Pim) is now providing the most powerful symbol processing capability in the world"

Again, to be really convincing on this claim you should compare the implementations of KL1 and PIMOS on your Pims with an implementation on a standard multiprocessor, ideally one that uses a RISC processor. Many are skeptical about the need for special purpose processors and language dedicated machines. The LISP machines failed because LISP was as fast, or nearly as fast, implemented via a good compiler on a general purpose machine. The PSI machines surely do not have a market because the latest Prolog compilers, compiling down to RISC instructions and using abstract interpretation to help optimize the code, deliver comparable performance. Such compilers run on $5000 workstations that offer all the other UNIX tools on which many have become to depend. Might not clever implementation on standard multiprocessors offer acceptable performance for parallel applications developed in KL1 and its extensions. If that is the case, the major

result and impact of the FGCS project will be its software, and its radically different approach to developing parallel applications.

I believe that this will indeed be the lasting legacy of the project, rather than the features of the PIM machines that have been built. (Did you really need to build five? Or was their construction relatively inexpensive compared to the cost of building PIMOS, KL1 and its extensions, and the applications?) However, computer architecture is not my field. A comparative evaluation against stock hardware might prove the need for the specialized architectural support of KL1. I understand that you intend to do such a comparison as well as a thorough evaluation of the alternative Pims. A good result would be the identification of a few low cost features that significantly boost the performance of KL1 and PIMOS on a multiprocessor machine, as was mentioned by Takashi Chikayama in the workshop. Such features could then be incorporated onto a general purpose commercial machine which offered both UNIX and PIMOS. (Could PIMOS run as a subsystem of UNIX?) UNIX would ensure initial penetration of the market and the ICOT software should then ensure a runaway success for some ICOT licensed Japanese manufacturer.

Conclusions

The FGCS project is something of which Japan can be truly proud. It has had more impact than any other research project in computer science. It was magnificent and bold in conception, and has delivered much more than I expected it could achieve.

PIMOS, the Pim machines, KL1 and its extensions and the impressive range of initial applications are significant achievements that are testimony to the skill, dedication and single mindedness with which the goals of the project were pursued. I suspect that in most other countries such a project would have ended long before the 10 year deadline, either through withdrawal of government support or lack of stamina of the principal investigators. ICOT and its associated researchers have also done excellent research in other areas of LP, as evidenced by the many publications and the high quality ICOT research report series.

The decision to freely distribute ICOT software is excellent, but this distribution needs to be supported by good documentation of both the software and the methodologies of its use.

You should definitely port KL1 and PIMOS to existing commercial multiprocessor machines. In your achievements report you say "*...the technology of PIMOS as well as the KL1 language is .. applicable to most MIMD .. machines..* ". I agree. By proving this you will increase the impact of the project. It is also necessary if the freely distributed software is to be widely used for developing applications for parallel machines.

You still need to convince a skeptical outside world that KL1 provides "*..much higher productivity and parallel program maintainability than any conventional language*". **Document, perhaps also refine, your application development methodologies. Describe the program level structure of your applications.**

Develop more applications. Develop what Ehud Shapiro in his workshop presentation called a 'killer' application and which I referred to as a 'demonstrator'. At the outset of the FGCS project there was much talk of knowledge information processing as the key application area of FG computers. Why not build a huge information processing application to support management decision making? Such an application must be multi-user, perhaps using KL1 and PIMOS implemented on a distributed loosely coupled

system. There is great potential for your technology to support distributed AI applications and CSCW (Computer Supported Cooperative Work).

Also **look at applications of your technology for numerical applications**. At a workshop in Syracuse in 1990, Geoffrey Fox complained that existing languages for parallel numerical applications only support homogeneous parallelism. Perhaps this is another application area for a suitably extended KL1, heterogenous numerical applications.

In some form or other the FGCS project must continue, or the achievements will have *far less* impact than they should.

The new project, run perhaps by a smaller ICOT, should support, maintain and continue to develop KL1, PIMOS and the application support tools. Its role should be to help others to use this software, by producing excellent documentation and assisting outside groups (in Japan and elsewhere) to develop applications. There is less need, now, for ICOT to develop complete applications, except perhaps the 'killer' application. **In addition, ICOT should be adequately funded to continue fundamental research into LP and its use on parallel machines.**

Finally, thank you for an exciting 10 years of excellent research into concurrent LP and its use. May your good work continue.

Keith Clark
Imperial College, UK

Interest & involvement in FGCS project

1. One of originators of concept of committed choice concurrent logic programming in 1980/81. (debt to Professors Hoare & Robinson)

2. Periodic visits and early involvement

1983	1 month visit Steve Gregory Same time as 2nd visit by Ehud Shapiro
1985/90	Brief visits
1984	Paper on OS in Parlog in FGCS84
1988	Invited lecture at FGCS88
1992	Invited to this workshop

3. Project a partly huge experiment investigating utility of concurrent LP

4. Hence, strong personal interest

Impact of FGCS Project

1. Stimulated much research activity & unlocked considerable government funds for AI & CS research in rest of the world
   - Alvey, Esprit, MCC, ECRC, SICS etc

2. Personal benefit - my chair + Alvey/Esprit projects

3. Significantly raised profile of AI and LP
   - 1985 IJCAI
   - 1986 ICLP conference
   - AI an IKBS become respectable in UK

4. Made Japan pacemakers in AI and LP (maybe even CS) research in world

But

Yesterday Herald Tribune can claim project has been a failure

- A view I do not share
- (Oh..?

Some comments on "Achievements" report

"... Thus (KL1) makes it possible to quickly develop application programs which make full use of parallel machines with hundreds to thousands of processes. ..."

I believe this to be true.

But can the rest of the (skeptical) world?

1. Where is the methodology of KL1 programming?
2. What exactly is KL1 over and above well described FGHC
3. You have impressive range of applications but where is convincing demonstration that they were easier to develop using FGCS project software
4. Need anatomy of applications, not just performance statistics

"(Pim) is now providing the most powerful symbol processing capability in the world ..."

However ...

1. What is the gain as compared with an implementation of KL1 + Pimos on more conventional existing multiprocessors?
2. Compare PSI approach to good compilation technology for Prolog on RISC machines.
3. Is dedicated hardware necessary?
   - expensive for customers
   - lags behind latest technology
   - Why did LISP machines fail?
4. Perhaps it was a mistake to put so much emphasis on specialised machines
   - were 5 Pim's needed to validate approach?
5. Instead, more could have been spent on applications, application support software and investigation of application development methodologies

Concluding remarks

1. FGCS project is something Japan can be proud of.
2. PIMOS, PIMs + impressive range of parallel and complex applications are significant achievements.
3. Free distribution of software a good idea.
4. But, need to port software to existing more widely available multiprocessor computers
   "..technology of PIMOS as well as the KL1 language processors is ..applicable to most MIMD..machines"
5. Need to convince a skeptical world (and HT) that there is "much higher productivity and parallel program maintainability than any conventional language"
   Do this by developing easy to use application development methodologies
6. Need also to develop more applications in new areas. Perhaps a huge knowledge information processing application for management decision making