TECHNICAL REPORT

COMPUTER SCIENCE

A FIRST PROGRAMMING LANGUAGE FOR INFORMATION SYSTEMS STUDENTS

A.L.J JEFFERIES, C.E. BRITTON

REPORT NO: 312

May 1998

A First Programming Language for Information Systems students.

A.L.J.Jefferies, C.E.Britton

University of Hertfordshire, College Lane, Hatfield, Herts. AL10 9 AB, England

email:- A.L.Jefferies@herts.ac.uk
fax:-01707 284303

ABSTRACT: *Following the decision to move to a modular structure for degree courses at the University of Hertfordshire for September 1997, an Information Systems award was added to the Computer Science scheme. On considering the content of proposed courses, it was decided to undertake a review of the Programming languages taught at first year level. A set of criteria was developed by the authors for comparing the different candidate languages. The reasons for reaching their conclusion , both academic and pragmatic, are discussed along with the particular needs of Information Systems students when compared with the traditional programming courses offered to undergraduate computer scientists.*

KEY WORDS: information systems, programming language, criteria,

---

1. Introduction

At a time when the national trend has been to divide up the teaching of Computer Science into composite areas and review the traditional components of Computing, there has been a growing interest in providing courses in Information Systems as described by Britton and Derrick (1995).

In common with many other universities in recent years, the University of Hertfordshire decided to introduce a new modular structure to its many schemes of study from September 1997. Following a consultation process, the Department of Computer Science took this as an opportunity to review its traditional teaching areas for undergraduates and post-graduates, in order to offer more choice in areas of study to prospective students. One of the most significant changes was the introduction of a new Information Systems degree scheme to run alongside the Computer Science scheme.

Because of financial constraints, a number of the proposed courses for the new schemes were developed using material currently being taught on the existing Computer Science scheme; it was felt, however, that a review was needed of the Programming languages and environments currently taught to all students, both undergraduate and post-graduate. The overall aim was to have a suite of courses which was more suitable to the educational needs of students graduating in the year 2,000 than it was felt that they are currently, (Bornat 96). The status quo, which had existed in the Computer Science degree for some years, was to introduce a small amount of a functional language, such as Miranda, and then teach Ada as the main first year programming language. This was felt to be somewhat outdated, as these languages are difficult for the students to grasp and moreover currently little used in industry. However some of the pragmatic reasons why they had been retained were because of the large amount of staff effort and the cost of funding a replacement necessitated by a change of language. With a new group of students having different requirements from a programming environment, there was now a greater incentive to change for everyone's benefit.

2. Information Systems students

As a first step towards defining the needs of information systems students an ethos was developed by the authors for the team reviewing the degree courses. It was felt that programming was not the cornerstone of an information systems student's knowledge, rather that as graduates they should be able to go out into organisations, assess their information needs at a variety of levels, propose and design systems to meet these needs and be aware of the effects that such systems will have on the organisation as a whole and the individuals that are part of it, (Hinton et al. 1997). Although this ethos was later refined, the main tenet remained that information systems students are unlikely to be doing much serious low level programming during the course of their careers: the focus of their interest typically lies in higher level issues such as the development and management of total systems, rather than in the detailed low level programming of part of a solution. However it was felt strongly by the team considering the new degree streams that prospective information systems graduates need to be able to understand and appreciate the generic principles of programming and they should have some experience of programming as a discipline.

An analogy can be drawn here with the difference between an architect and a builder, the architect needs to know the principles of building and engineering to ensure the building does not collapse but she is not normally involved with the physical tasks of digging the foundations and laying the bricks. Similarly the information systems specialist will understand how the building blocks of a system are put together and should have some practice at doing this but he does not spend much of his time at the programming level.

3. Criteria for a first programming language

Before the department of Computer Science began considering a subset of languages as candidates for teaching programming in the coming year to replace Ada and Miranda, the authors developed a set of criteria which would help them to identify why some languages would be more suitable for teaching *ab initio* to students from a range of backgrounds. These were:-

- to enable a straightforward introduction to the principles of programming in a first semester course
- to facilitate exposure to a variety of programming environments to be encountered throughout their degree courses
- to choose a tried and tested language and not be swayed unduly by the current commercial opportunities for jobs.

We wanted to avoid the situation where information systems students were only able to use commercial applications and where their only programming capability was in writing and developing databases. At the same time it was becoming clear that in order to pool resources and save money, the department was going to require the information systems students to be taught with the other first year computer science students for many of their courses. Could we therefore find a commonality of aim and teach the same material to students of a traditional Computer Science stream alongside the information systems students for part at least of their first year? This led to a review of our first set of criteria and further consultation with colleagues who would be doing the teaching and raised the series of questions given below.

- Do we want to use an integrated development environment (IDE), which may weaken understanding and be vendor specific?
- Do we want to be tied to a single supplier when we shall be teaching large numbers of students for some years ahead?
- Are we looking for a popular solution, which will attract students or will this turn out to be the latest programming gizmo which may or may not last in the long term?
- Is the proposed language simple to teach and simple enough for students to achieve something worthwhile in a short time?
- Can students afford their own copies to use at home and is it available on multiple platforms?

It became clear in the ensuing discussion that as far as the information systems students were concerned there was no single language that would fulfil their requirements and prepare them for their second year practical course in the development of an information system. They would need to have an introduction to the principles of programming in the first semester, which could be combined with the computer science students, and then take a separate course for their second semester module which would be more practically oriented. It had already been suggested that Microsoft Access™ would be a suitable vehicle for the second semester, so now the problem shifted back to the best possible environment for the wide variety of students taking the first semester course.

It was decided by members of the department that a course to introduce students to sound principles of programming should include the following elements: data abstraction, information hiding, modularity, component reusability. An object-oriented language was suggested as being the best vehicle for putting these principles across and as more in line with the future direction of programming languages (Heliotis 1996).

Inevitably there was a great deal of discussion before a final shortlist of programming languages was drawn up; this concentrated both on the reasons for change and the different directions in which colleagues felt that programming was moving commercially and academically. The three languages shortlisted for the final choice were Eiffel (Meyer, 1992), Java and Delphi - a surprisingly disparate set. The authors consequently drew up some of the immediate advantages and disadvantages from their point of view for using these as the first programming language for information systems students as well as for the computer science students. The summary below highlights the main points but is certain to lead to further discussion.

- From the academic point of view, Eiffel was the most popular choice. Since Eiffel is a 'pure' object-oriented language, it was regarded as an excellent vehicle for teaching sound principles of programming. In addition, Eiffel was currently being taught to non-specialist students on the M.Sc. Computer Science conversion course. This had the advantage that there was teaching expertise at the right level (i.e. for beginners) and some new support material had already been developed internally. There were copies of Eiffel in the university as well as the fact that it was both cheap and freely available for students to have their own copy. The main disadvantages were felt to be that there was little commercial use of Eiffel and it might not appeal to undergraduates.

- Delphi had already been agreed as the first programming language for the HND course, and thus also gave colleagues the opportunity to share expertise and teaching materials. It offered a Graphical User Interface environment which was seen as attractive to potential students and ideal for job opportunities. However this was also a disadvantage in that it would limit the first year experience of information systems students to purely GUI development, when they need to know rather more about programming commercially than this.

- Java, is definitely 'hot' but apart from the current hype of it being the 'Next Big Thing', can be used for writing straightforward, conventional programs with both text and GUI input and output. Students are reported as being enthusiastic users where it is taught, but as it is still so new, there was at the time of the decision in December 1996 very little current teaching expertise or reliable materials.

The final choice from these three was Eiffel, as the most effective language from an educational point of view, and for the pragmatic reasons that the department already had good experience of teaching it to beginners and the necessary back up materials, therefore the outlay required initially for staff training, new software and books would be small. Java came a close second, but an atmosphere of caution prevented it being chosen for the first year. It is hoped to teach Java on some of the optional second year courses, where it is perceived as being less risky than as an introduction to programming per se.

4. Conclusion

It would be easy to postulate about what the department would have chosen if we were living and teaching in the 'best of possible worlds'. The truth is that we are constrained by particular sets of circumstances. In the worst of situations there would have been either no choice between languages or no possibility of changing the existing language which was chosen several years ago under a different set of criteria. The languages for the new information systems students in their first year introduction to Programming (one semester of Eiffel followed by one semester of Access™) will not appeal to everyone. It will however open up the debate as to whether the department has played too safe and taken an option which is too academic and not useful enough commercially or whether, as the authors hope, the balance will prove to be about right.

References

Britton, C. and Derrick, A. *Introducing Information Systems into Computer Science* 3rd All Ireland Conference on the Teaching of Computing, Dublin City University, September 1995.

Bornat, R. *Is there a future for Computer Science Education?* 4th Annual Conference on the Teaching of Computing, Dublin City University August 1996.

Hinton, P. Jefferies, A.L.J. and Bennett J. *Teaching Information Systems at Undergraduate Level* in "Key Issues in Information Systems", Proceedings of the 2nd UKAIS Conference, McGraw Hill, April 1997

Heliotis, J. *Eiffel in Computer Science Education* JOOP Vol. 9 No.2 May 1996

Meyer, B. *Eiffel The Language* Prentice Hall, Englewood Cliffs, N.J. 1992