Language Tug-Of-War: Industry Demand and Academic Choice

Michael de Raadt **Richard Watson**
Department of Mathematics and Computing
University of Southern Queensland
Toowoomba 4350, Queensland
{deraadt,rwatson}@usq.edu.au

Mark Toleman
Department of Information Systems
University of Southern Queensland
Toowoomba 4350, Queensland
markt@usq.edu.au

Abstract

This paper seeks to inform instructors responsible for designing introductory programming courses within a university setting. In particular, guidelines for choosing programming languages to be taught are presented. Information relevant to instructors of later programming courses is also presented.

We ask the question: "are instructors of introductory programming courses wanting to teach industry demanded languages and if so, are they choosing the correct languages?" The guidelines produced to answer this question are based on a census of introductory programming teaching in Australian universities, coupled with a survey of employer demand based on newspaper job advertisements.

Keywords: programming languages, teaching, industry demand

1 Introduction

This paper reports findings revealed as part of a closely related but more general investigation into the pedagogy of introductory programming languages. Teaching is a concrete activity; development of introductory programming instruction techniques is influenced to some extent by the programming language being taught. Choosing a target language or set of languages is a necessary step toward developing specific instruction techniques. It was decided to carry out a census of introductory programming courses in Australia to inform this choice.

While the census was carried out in the context of a separate project, the results stand alone and have value to those involved in teaching programming courses.

Previous studies have examined the use of programming languages in the setting of an introductory programming course within Australia (Robins 1998; McDonald 1999) and overseas (Levy 1995), but surveyed only language choice. It should be stressed that a *census* (not a survey) has been conducted. All Australian universities were contacted and so a unique and complete picture of aspects of introductory programming language instruction has been captured.

The census revealed that perceived industry demand was the major factor in choice of introductory programming language. In order to determine whether the teaching languages chosen were in fact required by industry, a survey of newspaper job advertisements was performed.

The remainder of the paper is organised in three sections. Section 2 presents the main findings of the census. Section 3 describes the skills demand survey. Section 4 provides a set of recommendations about choice of programming languages in university information technology (IT) degree programmes. Section 5 provides conclusions drawn from the studies presented in this paper.

2 The Census

Choosing a programming language to teach is an ongoing question for academics responsible for introductory programming courses. To discover what was being used in such teaching, a census was conducted in the first half of 2001.

A full participation rate was achieved in the census by contacting each instructor by telephone. Instructors responsible for each introductory programming course taught within all Australian universities participated in the census; this includes courses taught inside and outside computer science schools/departments. Most courses covered were accredited by the Australian Computer Society (Australian Computer Society 2000).

This census covered language choice, paradigm choice, tools used to support teaching and reasons given by academics for making these choices. The results of this census were reported in full in de Raadt, Watson and Toleman (2002).

Courses	57
Universities	37
Students (Approx)	19,900
Average Students per Course	349

Table 1: Brief statistics

Thirty-seven of the thirty-nine Australian universities offer introductory programming courses. Eleven of these offer more than one (one institution offers as many as six), so the total number of introductory programming courses covered was fifty-seven. Based on estimates from instructors responsible for each course there were a total of 19,900 students studying during this period. Therefore the average course has a size of just fewer than 350 students. 19,900 students equates to approximately 4% of the population of Australian undergraduates. Table 1 summarises these results.

Language	Courses	Weighted by Students
Java	23	43.9%
VB	14	18.9%
C++	8	15.2%
Haskell	3	8.8%
C	4	5.5%
Eiffel	2	3.3%
Delphi	1	2.0%
Ada	1	1.7%
jBase	1	0.8%

Table 2: Languages taught

Nine different languages were being taught in Australian universities during the first semester of 2001. The number of courses teaching each of the nine languages and the proportion of the student population taught each language is shown in Table 2. Instructors were also asked to indicate the language taught prior to these current languages. The results showed a reduction in language diversity from 18 languages taught in 1996, 17 in 1997, 16 in 1998, 14 in 1999, 11 in 2000, to 9 in 2001. This trend indicates that choice of language is tending towards a smaller group of languages. The next run of the census will reveal if this trend will continue.

Pascal was no longer taught in any Australian university, nor were its descendants (Modula, Oberon or Component Pascal). One course used Delphi, however the instructor responsible for this course indicated that it would be replaced by C++ at the end of the current run of the course.

Four of the six courses teaching 'non-commercial' languages (Ada, Eiffel and Haskell) were run within the 'Sandstone' universities (Australian universities established before 1950). Haskell is taught only in Sandstone universities.

Used in industry / Marketable	56.1%
Pedagogical benefits of language	33.3%
Structure of degree/dept politics	26.3%
OO language	26.3%
GUI interface	10.5%
Availability/Cost to students	8.8%
Easy to find appropriate texts	3.5%
OS/Machine limitations of dept	1.8%

Table 3: Reasons for choosing language

Instructors were asked to indicate the reasons for their language choice. They gave up to three reasons which were grouped as shown in Table 3. The most common reason (as indicated by 56% of participants) was the industry relevance of the language and its potential to attract students. The second most common reason was the teaching benefits of the language.

Within Sandstone universities the ordering of the first two reasons for choosing a particular language was reversed, indicating that the pedagogical benefits of a language were more highly valued that the perceived marketability of a language.

Language	Courses Using	Chose For Industrial / Marketing Reasons
Java	23	16 (70%)
VB	14	6 (43%)
C++	8	7 (88%)
C	4	1 (25%)
Haskell	3	0

Table 4: Reasons for choosing particular language

When examined by language as shown in Table 4, C++ and Java stand out as being chosen due to perceived demand in industry and/or how this will attract students. Five of the nine current introductory programming languages (Java, VB, C++, C, Delphi and jbase) are in industry demand (see Section 3). 83% of students are being taught an industry relevant language.

Procedural	51%
Object Early	40%
Functional	9%

Table 5: Paradigm used in teaching

Instructors were also asked to classify their approach to teaching, by paradigm. This is independent of language taught. In other words, it is possible to teach using a procedural approach with an object-oriented language. As can be seen in Table 5, over half of all introductory programming students were taught using a procedural paradigm, even though 81% were taught using an object-oriented language.

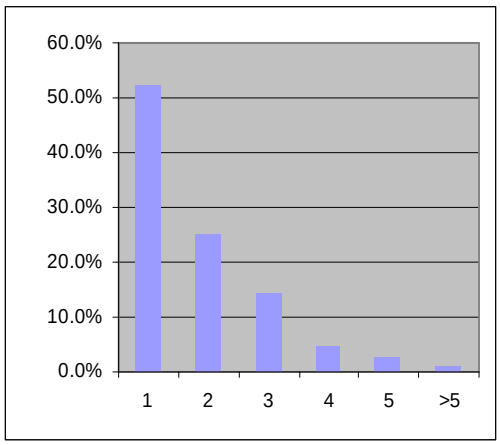

Figure 1: Number of languages per advertisement

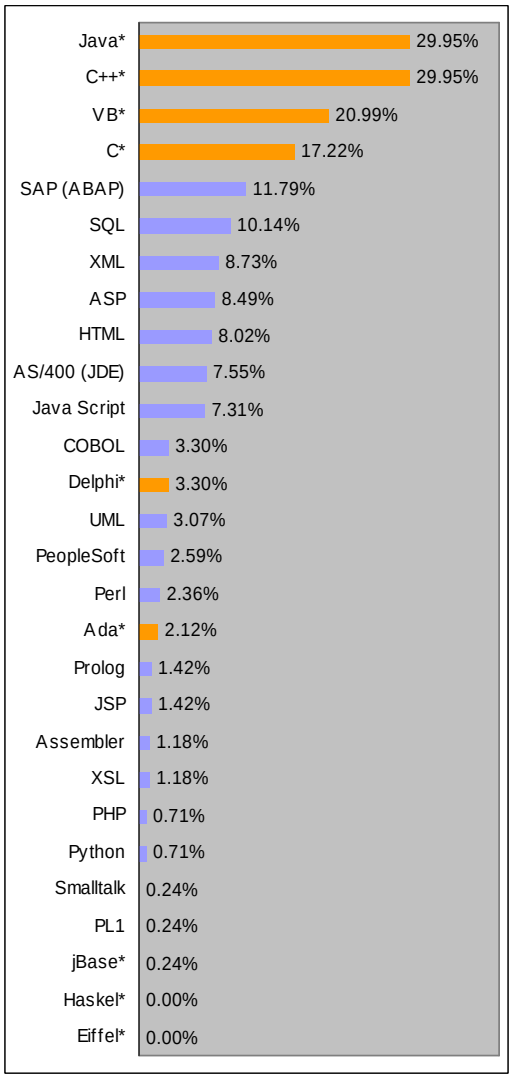

Figure 2: Advertisements by language

Language	Procedural	Object Oriented
Java (23)	30%	70%
Visual Basic (14)	79%	21%
C++ (8)	75%	25%
Eiffel (2)	50%	50%

Table 6: How OO languages are taught

Table 6 shows that while Java was predominantly taught using an *object-first* approach (introducing objects at the outset of a course) Visual Basic and C++ are taught using a procedural approach in most cases.

3 Industry Demand Survey

The introductory programming census revealed that instructors see the demands of industry as highly important when choosing a language to teach to novice programmers. This finding leads to the question: are instructors choosing industry-relevant languages?

A source of existing statistical information about languages demanded by industry was sought without success. For example, firms such as Olivier record trends in job advertisements by sector. Within the Information Technology & Telecommunications sector the finest grain element relating to Introductory Programming is total jobs in "Software Development and Engineering" (Oliver Recruitment Group 2002).

Each week *The Australian* newspaper publishes an IT section containing employment advertisements for IT professionals. From this, programming jobs were counted together with the languages indicated as required for each position advertised. This survey was conducted from January to June (both inclusive) 2002. There was a six month gap between the capture of industry demand information and the completion of the census of introductory programming courses. However, as only 5 out of 57 of the instructors from the census indicated there were plans to change the language being taught, the census results can be seen as fairly static.

The survey revealed 424 advertisements for programmers. Only programming positions were considered, so while SQL was counted as part of programming positions, advertisements for database administrators were not considered.

Figure 1 shows the distribution of the number of languages required per advertisement. The average advertisement required 1.84 languages. 48% of jobs required more than one language. The most popular languages were C++ and Java. C++ appeared as a requirement in around 30% of advertisements. Java also was a requirement for around 30% of advertisements. Visual Basic was next with 21%, then C with 17%.

Figure 2 shows the percentage of all advertisements that required each particular language. As mentioned, almost half of all advertisements required more than one

language, so the figures sum to more than 100%. Languages listed with an asterisk* are taught in introductory programming courses within Australian universities.

Advertisements were categorised by language as shown in Figure 3. Categories included traditional paradigms of procedural (PL1, Ada, Assembler, C, COBOL and Delphi) and object-oriented (C++, Java, Smalltalk and Visual Basic). No functional language skills were required in advertisements, but 1.4% of advertisements required Prolog. Prolog was categorised as 'Other' together with Assembler and UML. Remaining languages were classified into broader categories by similarity in application. Those concerned primarily with web applications (including ASP, HTML, Java Script, JSP, Perl, PHP, Python, XML and XSL) were categorised together as Scripting/Markup languages. Special purpose languages including SAP, PeopleSoft and AS/400 were categorised together. SQL and jBase were placed together in a database category.

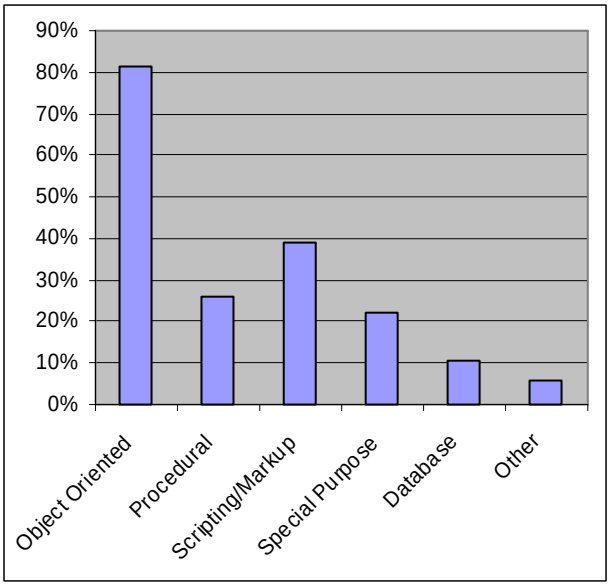

Figure 3: Advertisements by category

Object-oriented languages dominate, being demanded in 81% of all advertisements. Of note was the demand for web related scripting/markup languages, required in 39% of all advertisements. Procedural languages were required in 26% of advertisements. Demand for special purpose languages should not be ignored as they account for 22% of demand. Again these figures are relative and sum to more than 100% as many advertisements required multiple language skills.

C and C++	10.61%
C++ and Java	7.08%
C++ and VB	6.60%
VB and ASP	5.19%
VB and SQL	4.72%
JavaScript and HTML	4.25%
JavaScript and ASP	4.01%
SQL and ASP	3.77%

Table 7: Top matched pairs

The languages appearing together in the greatest number of combinations of two or more languages, as part of the requirements stated in a job advertisement, were firstly C (in 21 combinations), then equally C++ and Java (in 20 each). The most common pairings of languages are shown in Table 7.

A programmer with skills in more programming languages is qualified for more positions. Indeed the four most demanded languages do not overlap greatly in the positions for which they are required. While a person with skills in C++ is qualified for 30% of positions, Java qualifies for 30% of positions, Visual Basic qualifies for 21% of positions and C for 17%, knowing C++, Java, Visual Basic and C qualifies a programmer for over 70% of all advertised positions. Single language demand and combined qualification percentages are shown in Table 8.

C++	30%
Java	30%
Visual Basic	21%
C	17%
C++ and Java	53%
C, C++ and Java	58%
C++, Java and Visual Basic	66%
C++, Java, Visual Basic and C	71%

Table 8: Top Combinations

4 Discussion

There are many considerations when teaching an introductory programming course; choosing a language and paradigm are two of many decisions that need to be made to improve student outcomes such as better results in a course, deeper learning by students and higher employability of graduates.

The census of introductory programming courses emphasised the importance of employability as a consideration for instructors: will the student have the skills to gain employment at the end of their study? This

outcome is also one that can be judged by the student, based on university advertising material, before they begin their study. Also, students are more enthusiastic when studying a language they feel will make them more employable. It is a very important aspect of a course as it can be easily judged and compared.

Choosing the 'best' language is not a simple decision. The following questions aim to address the key issues being considered when choosing languages to teach.

4.1 What Language Should I Teach In My Introductory Course?

The languages most demanded by industry are C++ and Java. To choose between these two, decisions must be made on other related factors such as pedagogical benefits, compiler and textbook availability, and the skills of teaching staff. Switching from C++ to Java or vice versa for reasons of perceived industry demand is not justified by the demand measured in this study.

Demand measured correlates well with the language choices made by instructors of introductory programming courses as revealed by the census. The top four languages demanded by industry are in the top five languages taught in Australian universities (the odd-one-out being Haskell which has no measured industry demand). Java is taught more widely than industry demand would require; this may be a product of the perceived pedagogical benefits of the language. C++ is equally the most industry demanded language, but is less represented in introductory programming courses at universities.

4.2 What about Haskell and Eiffel?

Pedagogically sound languages such as Haskell and Eiffel can provide a strong foundation for later programming study. Learning such languages may benefit the student in development of programming concepts, problem solving skills and learning later languages. It is not, however, recommended that teaching of these languages be done at the expense of industrially demanded languages.

Notably, Haskell and Eiffel are taught almost exclusively in sandstone universities which are identified by Ashenden and Milligan (1999) as having the highest student demand for courses. This is reinforced by the choices indicated by instructors from Sandstone universities, which showed that the pedagogical benefits of a language are considered to be more important than the perceived marketability of a language.

4.3 What Paradigm Should I Teach?

With current industry demand, it is apparent that students graduating from a study of programming should have some skills in programming in object-oriented programming languages. This does not necessarily mean that their study must begin in an object-oriented paradigm. The teaching of procedural concepts cannot be ignored. Teaching of C++ and Visual Basic, which are both classified as object-oriented languages, need not start with an object-first approach, as is the case in over three quarters of the introductory courses that teach these languages. However from the industry demand recorded in this study, a firm grounding in object-oriented programming, possibly later in a degree, is recommended.

The presence of demand for scripting/markup languages in advertisements indicates that the modern programming degree should include some courses teaching these languages and the web related concepts that surround them.

Some Australian universities have already signed agreements with industrial partners in order to teach special purpose languages for systems such as SAP, PeopleSoft (Hawking, Foster and Bassett 2002; Hawking, McCarthy and Foster 2002; McCarthy and Hawking 2002). With growth in the uptake of such systems, and demand measured in this study, including teaching of such languages may make graduates more employable.

4.4 What Other Languages Should We Teach?

The current work has focused on the issue of programming language choice for introductory programming courses. As an IT-based degree will typically expose a student to a range of programming languages, the choice of subsequent languages is also an important issue.

The census did not capture detailed information about current patterns of language use within non-introductory courses. Indeed such information is likely to be an order of magnitude more difficult to capture with the same accuracy due to the vast number of courses that employ some form of programming language, and the fact that many are non-compulsory. The following observations, based on the job advertisement survey, provide a guide to inform language choice where the employability of graduates is an important consideration.

Universities must consider a progression of programming language teaching that will take a novice through to graduation. Using a single language in this progression allows more time to focus on broader concepts of programming and related topics. However, sending a student out into the workforce with only a single language in their repertoire means ruling them out of almost half of all jobs available. Exposing students to multiple languages is therefore a must. But which languages and what combination of languages will benefit students the most? The employability aspect of such outcomes can be measured.

A progression from C to C++ to Java will qualify a graduate for 58% of advertised positions. Teaching Visual Basic provides a graduate with qualifications for 21% of advertised positions. A course consisting of the related languages HTML and XML followed by a second course teaching the related languages JavaScript and how it can be applied in ASP (Active Server Pages) would qualify a graduate for 19% of advertised positions. Including a course involving an industrial partnership with SAP and teaching ABAP programming would qualify a graduate for 12% of advertised positions. An IT

degree course including all these languages would qualify a graduate for 86% of positions.

4.5 What Will Happen In The Future?

Java has grown in the eight years since its introduction to become equal with C++ as the most demanded language in industry. Java has proliferated on the back of Internet growth and the uptake of browsers such as Microsoft Explorer and Netscape Navigator that support the Java Virtual Machine. However, some caution is recommended to any that would alter a degree to include Java for these reasons. Use of the Java Virtual Machine on the Internet has dwindled as the use of server side document generation increases in popularity. Microsoft has also developed the .Net runtime system, which is a competing technology to the Java Virtual Machine as a platform independent system. The standard installation of Internet Explorer does not include the Java Virtual Machine and it is no longer available as an optional plug-in from Microsoft.

As Internet growth continues, demand for skills in scripting/markup languages is also anticipated to continue growing. Compilable languages such as C and C++, associated with the high-speed delivery of information required for server-side web applications, would likewise be anticipated to become more in demand.

5 Conclusions

The results reported in this paper should be of some comfort to both the academic and industrial communities. The majority of academics participating in the introductory programming census felt that industry relevance was highly important in choosing a programming language to teach. The subsequent newspaper survey indicated a strong correlation between language demand and language teaching. Academics are mostly succeeding on choosing an industrially relevant language and industry is benefiting from appropriately trained graduates.

The question of selecting a group of languages to teach is an interesting one. While more extensive information on usage/demand by industry is needed before making a stronger statement, it is quite clear from the current study that experience in a relatively small number of well selected languages can prepare a student very well for a wide range of employment opportunities.

The pair of studies reported here give a snapshot of the language teaching and demand situation in 2001/2002. To discover longitudinal trends in the teaching of languages and related topics in introductory programming courses around Australia, a repeat census is planned for early 2003. Continued monitoring of employer demand should enable even better informed recommendations to be developed in the future.

6 References

ASHENDEN, D. and MILLIGAN, S. (1999). *The good universities guide: Universities, TAFE and private colleges in 2000.* Australia, Hobsons.

AUSTRALIAN COMPUTER SOCIETY (2000). Accredited courses., Retrieved August 29, 2001 on the World Wide Web http://203.58.197.209/acs/events_admin/course20_6.htm.

DE RAADT, M., WATSON, R. and TOLEMAN, M. (2002). *Language Trends in Introductory Programming Courses*. The Proceedings of Informing Science and IT Education Conference, Cork, Ireland, InformingScience.org.

HAWKING, P., FOSTER, S. and BASSETT, P. (2002). *An Applied Approach to Teaching HR Concepts Using an ERP System*. The Proceedings of Informing Science and IT Education Conference, Cork, Ireland, InformingScience.org.

HAWKING, P., MCCARTHY, B. and FOSTER, S. (2002). *Teaching E-Business Concepts Using SAP's OnLine Store*. The Proceedings of Informing Science and IT Education Conference, Cork, Ireland, InformingScience.org.

LEVY, S. P. (1995). "Computer language usage in CS1: Survey results." *SIGCSE Bulletin* **27**: 21-26.

MCCARTHY, B. and HAWKING, P. (2002). *Teaching SAP's ABAP Programming Language to IS Students: Adopting and Adapting Web-based Technologies*. The Proceedings of Informing Science and IT Education Conference, Cork, Ireland, InformingScience.org.

MCDONALD, C. (1999). 1st year programming languages in Australian and New Zealand universities., Retrieved June 26, 2001 on the World Wide Web http://www.cs.uwa.edu.au/~chris/java-in-cs1/anzacs.html.

OLIVER RECRUITMENT GROUP (2002). Internet Job Index, Retrieved 7th November, 2002 on the World Wide Web http://www.olivier.com.au/jobindex.php.

ROBINS, A. (1998). First language survey, Retrieved June 15, 2001 on the World Wide Web http://www.cs.otago.ac.nz/survey/surveyhome.html.