To Pull In Harness: Industry and Universities face educating the Professional Software Engineer

Rick Duley, D Veal, S P Maj
Edith Cowan University, Perth, Western Australia

Abstract

Computing curricula in Australia have tended to emphasize the scientific and computer engineering side, the hardware side, of computing but the advent of the Professional Software Engineer (PSE) demands new approaches to curricular design. With the Institute of Engineers, Australia, (IEAust) already taking part in establishing the new profession in Australia it is to be expected that educational requirements here for prospective professionals will mirror those for the traditional disciplines. This includes a requirement for a full four-year course including practical work as in traditional engineering courses so that the new discipline can merit the equal status it seeks. Since the graduate PSE will tend to be industrially oriented, a large practical content will be desirable in the course. This, in turn, will not only raise educational issues within academia but also challenge long-held industrial attitudes towards universities, their students and graduates. Employers in the software industry have, historically, been somewhat sceptical of the value of a traditional CS education. Complaints that graduates lack practical competence and that universities pay little attention to the needs of industry have been noted. Indeed, the reputation of academia in industry appears somewhat tarnished, yet only practicing software developers can supply the positions in which potential graduates can undertake their practical training and only experienced software producers can supply that training. If universities are to eschew any ‘ivory tower’ philosophy, design new courses to meet the requirements of engineering registration committees and send undergraduates into industry on practicum, then industrial management must be prepared to undertake the roles this new scenario gives them. This paper looks at the industry/university relationship in Australia and the USA, and the difficulties to be surmounted to produce the necessary atmosphere of trust and teamwork.

1. Introduction

Software Developers and Computing Academics have historically held their ground on opposite sides of an argument. Academics hold to theory and the latest technology and techniques — Developers are output driven and are only interested in getting the job done and out the door. Academics talk of ‘formal methods’ and ‘knowledge engineering’ and ‘state machines’ and
‘high-level supertypes’ and on and on and on. — Developers create software. Academics talk of ‘software best practices’ and ‘personal software processes’ — Developers are too busy to listen. Academics scoff at ‘hackers’ — Developers scoff at ‘creeping buzzwordism’. Academics say things like “The general practice of industrial software engineers are poor by almost any measure. Their projects are typically late and over costs, they cannot predict when they will finish, and the final products frequently have many defects.”1 and then bewail comments like “He was unlikely to hire any of our graduates; he was not looking for 'engineers'; he needed a few 'brilliant hackers'.2 Historically, ‘.ne’er the twain shall meet’. This paper is about the need to change all that.

2. Background

Seventy per cent of software written in Europe is outside the Information Technology sector. In virtually every situation, failure of industrial software means the failure of the machinery it operates and failure can be disastrous. This mission-criticality has led to a push towards the recognition and accreditation of the Professional Software Engineer — and this, in turn, to a call for tertiary curricular support for the new profession. The Institute of Engineers, Australia, (IEAust) already taking part in establishing the new profession in Australia it is to be expected that educational requirements here for prospective professionals will mirror those for the traditional disciplines. This includes a requirement for a full four-year course (standard undergraduate science degrees are of three-year duration) including practical work as in traditional engineering courses so that the new discipline can merit the equal status it seeks. Computing curricula in Australia have tended to emphasize the scientific and computer engineering side, the hardware side, of computing but the advent of the Professional Software Engineer (PSE) demands new approaches to curricular design. Obviously, Universities in Australia are now dealing with a major problem — the requirement for curricular reform — and a recent survey conducted by the authors shows that the eleven Universities accredited by IEAust are meeting that challenge. This paper asks whether a one-sided effort is adequate.

3. An Uneasy Relationship

Employers in the software industry have, historically, been somewhat sceptical of the value of a traditional CS education. Complaints that graduates lack practical competence and that universities pay little attention to the needs of industry have been noted. Indeed, the reputation of academia in industry appears somewhat tarnished. McCracken writes that new hires do not understand what it takes to develop real products. Bach complains that students he has interviewed for employment rarely received significant instruction in testing, requirements analysis, inspections, documentation or software design techniques. Dawson says that graduates are ill-prepared for reality. Parnas (an academic) confesses that he is appalled at what graduates don’t know. On the other hand, from a university viewpoint, relationships between education and industry have tended to be somewhat one-sided. Certainly, there are structures to encourage industry input into course construction — as with most, if not all, Australian universities there is a Consultative Committee at Edith Cowan University (ECU) for that purpose — but students tend to proceed through the system and then disappear into industry leaving the
university with little in return for the effort. Industrial involvement within universities appears to remain at a low level while industry still expects a quality graduate. This industry mistrust is deep rooted — Fisher referred to it in 1974\textsuperscript{12}.

It is true that academic forays into industry can be catastrophic. Glass relates that in an effort to retire legacy systems the Westpac Banking Corporation (Australia's second largest bank) launched the CS90 project in which academics, none of whom had worked on industrial software projects or in banking, were hired to drive an innovative project. Terms like 'software component reuse', 'object orientation', 'inheritance', 'state machine models' and 'knowledge engineering' rapidly formed \textit{“a glut of creeping buzzwordism”} which was, apparently, received with less than enthusiasm by veteran software practitioners. One veteran practitioner was quoted as saying that it was like being in Wonderland with the Mad Hatter in control\textsuperscript{13}. However, the authors' reading has yet to discover documentation of industry forays into academia, which tends to support the aforementioned attitude in academia that the relationship is one-sided.

4. Curricular Challenge

Historically, much of the work on projects across the whole spectrum of software development, control systems for aircraft, power plants, petrochemical and water treatment plants and the like — work which is likely to fall to the lot of the PSE — has been done by CS graduates, all based on their education as CSs but after in-service training as SEs. Ratification of the new profession of Software Engineering will change all that. When the numbers of accredited SEs achieve a certain \textit{“critical mass”}, government and big business — probably led by the insurance industry — will demand oversight of major projects to be undertaken by those SEs. The days of the self-taught high-school student making it in Software Engineering without a degree will come to an end as well\textsuperscript{2}. Inevitably accreditation as a PSE will require appropriate academic qualification and, probably, ongoing education to maintain contemporaneous knowledge. Curricular renovation to meet the requirements of the new profession will be far more complex and difficult than a change of mainstream programming language, or even of the manner in which it is taught. Curricular renovation does not mean, however, that it will be sufficient for the university course to provide some facility in Visual Basic, Java or whatever the flavour of the month might be and expect the graduate to be welcomed into anything other than e-Commerce. In fact, familiarity with such non-standard, proprietary tools, while useful in a job market, is often seen (in academia) as peripheral and conducive to a bandwagon mentality\textsuperscript{14}. Universities are faced with a rather different task from that of preparing a graduate for a life of academic research. Graduates must be prepared for a life outside of a university in which they have to deal with real-world problems and real-world people. This is the concept of \textit{“graduateness”} which is the subject of study and debate in the UK where graduates are seen to need \textit{“ancillary skills”} like being able to write grammatically acceptable English (or Welsh), a certain level of numeracy and a range of general knowledge\textsuperscript{15}. Few traditional CS departments would be geared to such a task although, for example, chemistry, physics, English composition and liberal arts (including fine arts, history, philosophy and social sciences) are included in the 1998-99 Bachelor of Software Engineering (BSE) degree curriculum at Rochester Institute of Technology (NY, USA)\textsuperscript{16}. 

\textit{Proceedings of the 2001 American Society for Engineering Education Annual Conference & Exposition} 
Copyright © 2001, American Society for Engineering Education
Further difficulties are faced by the universities stemming from the well-known fact that few CS faculty members have a background in SE and that much of the knowledge in SE resides in industry.

5. Industry’s Challenge

This knowledge/experience shortage is a longstanding problem. In 1989, Gibbs noted that attracting new faculty with industry experience was difficult and that those practitioners who did make the move faced lower salaries, more modest computing facilities and fewer software engineering support tools\textsuperscript{17}. Yet only practicing software developers can supply the positions in which potential graduates can undertake their practical training and only experienced software producers can supply that training. Yourdon refers to a ‘Catch 22’ situation in which everyone bemoans a shortage of trained, competent programmers and systems analysts but that the only way industry can get them is to train ‘relatively useless college graduates’ on the job\textsuperscript{18}. This paper poses the question, “Whose fault is that?” Students regularly enter the tertiary education system, proceed through it and then plunge into a career in a vague and amorphous entity referred to as ‘the industry’ about which they have been told little and know less. (Figure 1) This paper asks, ‘How can industry expect a supply of ‘trained, competent programmers and systems analysts’ when, as mentioned above, CS faculty (through no fault of their own) do not have the experience to teach them?’

Certainly, there are structures to encourage industry input into curriculum construction — as with most, if not all, Australian universities there is a Consultative Committee at ECU for that purpose — but there appears to be a considerable difference between curriculum construction and the provision of adequate, timely, experience-based preparatory education for undergraduate students. [This is not to suggest that the function of a university is solely to prepare students for their first job — it is not, that is the function of a Community College — the function of a university is to educate someone for a career. Students need to be exposed to methodologies, tools etc. (and who better to give the exposure that people experienced in using them in industry?) but even in 1987 Brooks wrote that the object is to teach students to think like software engineers, to shape ways of thinking and to give preliminary experience in the use of tools (which would lead to adaptability in the field)\textsuperscript{19}.] Industry involvement needs to be much more intimate than a triennial gabfest — the challenge to industry, in the view of the authors, is to find ways to make experts available to work with the universities.

6. A Possible Industry Response

As quoted above, much of the knowledge of software engineering resides in industry and we believe that industry must be prepared to take that knowledge into the universities. Problems arise from this concept:
(1) Many people in the software industry do not have the advanced educational qualifications which many faculties require. Some flexibility will be required in this matter. At ECU, for example, full-time lecturing positions (other than in exceptional circumstances) require a relevant PhD, but sessional (part-time) positions remain at the discretion of the Head of School.

(2) Formal lecturing requires different pedagogical skills from informal, one-on-one training in the workplace. It might well be that people who would be willing to take on the job would not possess those skills.

(3) It might also be that existing academic staff would find it difficult to relate to sessional staff from industry with limited educational qualifications and pedagogical skills and a totally different background.

(4) Many companies, operating under pressures of development and deadlines would be reluctant to release staff for the requisite time to adequately and effectively carry on a lecturing job.

(5) Finally, with the level of remuneration common in industry, considerable personal financial sacrifice might have to be expected of those who took up such an educational position.

Problems there may be, but the authors believe that the benefits which might accrue from such an exercise would far outweigh the effort required to solve those problems. Industry must be prepared to make the personnel available and academia must be prepared to welcome them. In the end, it appears to come down to a question of willingness on the part of both parties.

7. A Matter of Trust

While a gulf was perceived to exist between the software industry and academia, it was possible — perhaps even comfortable — for each party to take the high moral ground on its side of the gulf and to criticise or ignore the work and words of the other. In Australia, the involvement of IEAust in the education of Software Engineers and the cooperation between IEAust and the ACS, who currently have a working party devising a unified accreditation code for SEs, has changed all that. At least in this field, there is no longer any gulf between the university and the people who produce software for the engineering industry. Neither side can afford itself the luxury of separation.

Management throughout industry is being urged to generate genuine trust and respect as steps...
towards empowerment of staff and the creation of a ‘quality culture’ throughout their organizations. It is widely recognized that this process of trust generation takes real commitment ‘from the top’ and is a long process\textsuperscript{20}. In the same way, universities and industry will now have to build mutual trust and respect to facilitate a smooth change to university educated professionals being accepted in industry and industrial people being accepted in the universities. It is a process which will equally require commitment ‘from the top’ and which is likely to be equally lengthy. Furthermore, universities will have to deserve that trust by producing competent and capable graduates with truly professional skills and attitudes, and industry will have to deserve that trust by their acceptance of, and co-operation in the education of, those graduates. In prospect is a close, if (perhaps, at first) rather uneasy, relationship between academia and industry. A daunting prospect perhaps but one for which a solution appears to be vital. This is a prospect which the authors believe holds out hope that, in the near future cooperation, trust and mutual respect will be the bywords for the relationship between academia and the software development industry in general. The authors believe that that is a situation which can only be of enormous benefit to us all.

Bibliography

14. Feldman, Michael (mfeldman@seas.gwu.edu). (99). First year courses, standards, and Ada. E-mail to Duley, Rick (r.duley@cowan.edu.au).


**Biographical Information**

RICK DULEY graduated with First Class Honours in Computer Science in 1996. He is currently a Doctoral Research Student at Edith Cowan University working in the field of Software Engineering Education. Coming from a Heavy Industry and Mining background he has a special interest in things industrial and in the application of engineering principles to software construction. r.duley@ecu.edu.au

Dr S P MAJ is a recognized authority in the field of industrial and scientific information systems integration and management. He is the author of a text book, *The Use of Computers in Laboratory Automation*, which was commissioned by the Royal Society of Chemistry (UK). His first book, *Language Independent Design Methodology - an introduction*, was commissioned by the National Computing Centre (NCC). Dr S P Maj has organized, chaired and been invited to speak at many international conferences at the highest level. He has also served on many national and international committees and was on the editorial board of two international journals concerned with the advancement of science and technology. As Deputy Chairman and Treasurer of the *Institute of Instrumentation and Control Australia (IICA)* educational sub-committee he was responsible for successfully designing, in less than two years a new, practical degree in *Instrumentation and Control* to meet the needs of the process industries. This is the first degree of its kind in Australia with the first intake in 1996. It should be recognized that this was a major industry driven initiative. p.maj@ecu.edu.au

DAVID VEAL received his honors degree in Theoretical Physics from the University of York in England. After completing a Grad.Dip.Ed. from the University of Keele he lectured in Physics at South Devon College UK for 10 years. He now lives in Western Australia where he has taught Computing and Physics at high school level. He is a Doctoral student in Computing Science at ECU in Perth, Western Australia. He is investigating competency-based techniques in Computing Science as well as the modeling of computers to aid student understanding. d.veal@ecu.edu.au