Preparing Students for Software Engineering

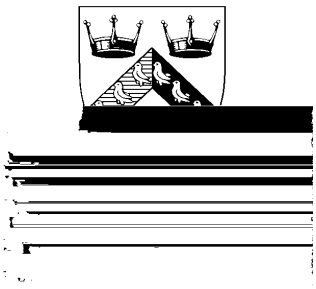
Steve M. Easterbrook
Theodoros N. Arvanitis

CSRP Number 413

March 18, 1996

ISSN 1350-3162

UNIVERSITY OF



Cognitive Science
Research Papers

Preparing Students for Software Engineering

Steve M. Easte b ook
Softwa e Resea ch Lab.
NASA/WVU IV&V Facility
Fai mont, WV 2655
Email: steve@atlantis.ivv.nasa.gov

Theodo os N. A vanitis
School of Cognitive & Computing Sciences
Unive sity of Sussex, Falme , B ighton, BN1 9QH, UK
Email: theoa@cogs.susx.ac.uk

Ma ch 18, 1996

Abstract

This position paper[1] describes our work with a new course at Sussex University, designed to bridge the gap between computer science and software engineering. We argue that the way in which software engineering is introduced in most computer science degrees makes it hard for students to internalise the lessons of good engineering practice. In particular, programming is seen to be divorced from software engineering. We describe a new course taught24.71996he3654(et)999.959ra0T

1 Introduction

There is an important gap in the training of computing professionals, between software engineering and the practical aspects of "computer science", particularly programming. Programming is normally taught right at the start of a computer science degree. A typical introductory programming course will introduce students to a particular programming language, showing them how to construct algorithms to solve simple problems, and how to convert those algorithms into programs, through proper use of the constructs available in the chosen programming language. At some subsequent point in their careers, these students will fets to

2.1

work. At the very least, this allows them to let off steam when they get frustrated with the practical work. More importantly, it provides a route into discussion about process improvement. Whenever they criticize the course, we try to respond positively to their criticism, but also ask them to come up with coping strategies, so that if they can't change the nature of the course, they can at

to have gained less from the course than the others, and we hope to prevent this happening in future years.

Even with these problems, the course has been a great success in preparing students for software engineering. It has given them a wide range of experiences, from teamwork and technical presentation, through to an appreciation of how hard it is to modify other people's undocumented code. Most importantly, it has given them practice in the key software engineering skill of learning from failures. The majority of the students enjoyed the course, and rated it highly in terms of relevance to themselves. This first set of students are currently taking their second year software engineering course. They are proving to be far more motivated to learn about software engineering that any previous cohort of students.

At the current time, we have taught the course once, in Spring 1995, and are running it again in Spring 1996. It is too early to tell whether we have achieved our long term goal: we will not be able to determine whether the students who took the course really have taken the lessons to heart, until we observe whether they willingly adopt some of the techniques in their own software projects.

Acknowledgments

We would like to thank the tudent at Su ex Univer ity who took part in the cour e in 1995, and provided many helpful comment on the cour e. Al o, thank are due to Amer Al-Rawa and Joe Wood, who helped u to de ign and run the cour e.

References

[1] D. L. Parnas, "Education for Computing Professionals" *IEEE Computer*, Pp. 17-22, Jan 1990.

[2] J. J. Horning and D.B. Wortman, P13atare