A Graduate Course on Software Reuse, Domain Analysis, and Re-engineering

William B. Frakes

Virginia Tech
Computer Science Department
Falls Church, VA 22042
Tel: (703) 698-4712
Email: frakes@sarvis.cs.vt.edu

Abstract

Reuse education has long been recognized as important for the successful implementation of reuse. Yet reuse has traditionally not been taught in universities. In a recent survey, we found that only 13% of respondents said they had learned about reuse in school [1]. A recent workshop addressed this issue, and published preliminary suggestions for reuse education for various populations (upper managers, line managers, technical staff, etc.), in different areas (industry, government, academia) [2].

I participated in this workshop by leading the group on education for domain analysis. One of the points made by the group was that because of the immaturity of the reuse area, seminar courses were probably most appropriate for university education in reuse. In the Spring semester of 1993, I taught a graduate level seminar course on software reuse, domain analysis, and re-engineering at Virginia Tech. This position paper describes the motivation for the course, its content, and my subjective evaluation of it.

Keywords: reuse education, domain analysis, re-engineering.

Workshop Goals: to explore the state of the art and state of the practice in reuse education, the breadth and depth of the education "problem", and to compare this work with similar efforts.

Working Groups: reuse education, domain analysis.

1 Position

There were fourteen students in the course, all at the Masters level. Some were computer science students and some students in the M.I.S. program. Like most students at the Northern Virginia Campus of Virginia Tech, they were part time students employed by various companies in the area. Most have many years of practical experience in the computer industry.

The grading for the class was as follows: exercises 20%, midterm 40%, class project 35%, class participation 5%.

I selected the Hooper and Chester, *Software Reuse: Guidelines and Methods* text [3] for the course since it was the only book on reuse which was not just a collection of readings, and the book had been positively reviewed in ReNews. I also assigned many papers from the reuse literature. The students gave the book a low rating. In general, they found the assigned papers much more useful.

Below is the topic outline for the course. I invited several expert guest speakers. This was very successful both as motivation for the students and as a source of information.

- Reuse Introduction
- Reuse Management & Measurement
- Design for Reuse – Guest Speaker: C. Braun
- Reuse Libraries – Guest Speaker: C. Lillie
- Domain Analysis - Guest speakers: R. Prieto-Diaz, S. Wartik
- Generative Reuse
- Re-engineering
- Commercial Sources, Asset Quality

I assigned exercises for the above topics in an attempt to make the abstract concepts more tangible. Some of these were:

- Reuse Design and Re-engineering: Each student designed a reusable code component, and a non-code component. They were later asked to re-engineer these to make them more reusable, and to design a quality assurance procedure for them. Many of the students found design for reuse difficult and confusing.
- Reuse Libraries: Students were given accounts on the Asset and Adanet reuse library systems and asked to write an evaluation of them. They were also asked to create an enumerated, faceted, and free text classification of 20 Unix tools. Students did not find either Asset or Adanet very usable. In general they did very well on the classification tasks.
- Domain Analysis: Students were asked to do a domain analysis of sorting routines, and of another domain they were familiar with. The lack of a complete prescriptive method for domain analysis was a problem, as it is in practice.
- Reuse Measurement: Each student was asked to use RL [4] to measure reuse in C code.

Each student was required to do a project, write a report on it, and make a 20 minute presentation to the class. These projects covered various areas of reuse. Some of the more interesting were:

- Design of a reusable code collection for imaging and information retrieval.
- Re-engineering C into C++, a case study.
- A comparison of domain analysis methods (FODA and Prieto-Diaz).
- Legal Issues and reuse.
- Japanese software factories.

In summary, though much of the material in the area is still uncertain, in general the course was successful. The students gave the course an overall rating of 3.5 (3=good, 4=excellent). They did not like the Hooper and Chester text. There is a need for better texts in this area. The guest speakers were considered to be very effective. Students without a strong background in compilers and formal methods found some of the material on reuse design and generative methods hard to follow. The exercises and projects were helpful in making the abstract material more tangible.

References

[1] W. Frakes and C. Fox, "Software reuse survey report," tech. rep., Software Engineering Guild, Sterling, VA, 1993.

[2] C. Lillie, "Asset project," in *Proceedings of Asset Reuse Education Workshop, Morgantown, WV*, 1992.

[3] J. Hooper and R. Chester, *Software Reuse: Guidelines and Methods.* Plenum Press, 1991.

[4] W. Frakes, "An empirical framework for software reuse research," in *Proceedings of the Third Annual Workshop on Methods and Tools for Reuse, Syracuse University CASE Center Report*, 1990.