Brady Field

The Re-Engineering and Reuse of Software

**Important points**

“Software re-engineering and reuse are concerned with maximizing software usage for any given development effort.” [335]

“The production of software is expensive, and with the decrease in the cost of hardware and the increase in hardware capability, we have been led to ever more ambitious development projects, while qualified and experienced software development staff are in short supply.” [335]

“One method proposed for making a significant improvement in productivity and quality is software reuse: using a given piece of software to solve more than one problem.” [336]

“Facilities are provided for user created components to be stored and reused, for example through subroutines and macros. This is not the usual way of viewing programming languages and compilers, which are usually seen as ‘tools’ rather than the engineering foundations of software production. This view of programming languages emphasizes the unique nature of software, the great diversity that its components might take, and the very flexible way they might be interconnected.” [337]

“Many tools are really very simple, having been in use for many decades, and it is surprising to see these being offered now as if they were new.” [338]

“We have seen that the important part of the whole reverse engineering process is recognizing the known higher level domain concepts in the code.” [339]

“The general idea of software reuse is that it is a component repository from which reusable components may be extracted.” [342]

“An important issue is the size or granularity of the components – are they small like scientific subroutines, or large like word-processing packages? Clearly there is a place for both, but the requirements for description and storage may be very different.” [343]

“The most recent development in the frameworks arena is the use of design patterns. These are small configurations of components which work together to achieve particular ends, but which could not themselves be encapsulated as a component.” [344]

“In both Reverse Engineering and in Reuse, we have seen that a proper understanding of the application domain is essential. We termed the process of obtaining this understanding domain analysis. To do this we need to identify the major concepts of the area and identify the relationships between these concepts.” [345]

“In order to promote reuse, it is not sufficient to establish the correct technical environment; we must also address other non-technical issues, such as managerial practices and the legal constraints of a particular community.” [346]

“Preparing software for reuse as a component does require extra effort and this extra effort needs tp be rewarded.” [347]

“The technology for these is now well established, and while it is the focus of on-going research, there is really no technical barrier to reuse and re-engineering.” [348]

**Disagreements**

“Often, the maintenance effort is forced to rely on the code itself. This means that 30 to 35 percent of total life cycle costs are consumed in trying to understand the software after it has been delivered in order to make changes.” [336]

This is uncited. The numbers they’ve used here have nothing behind them to give them weight. How do they know it will be 30 to 35 percent? Why didn’t they tell the reader how this was measured? This could be a wildly different number. In fact, this number is probably different depending on many factors which also were not referenced.

**Questions**

I understood everything in the article.