Holden Sandlar

Advanced Computer Architecture

A Plea for Lean Software

The first portion of this paper focuses on the trend seen for the past few decades in software development. The trend is that most software designers have moved away from elegant and optimized software solutions and instead rely on hardware advances to deliver increased performance. As a result of this trend, the majority of software developed today winds up being bloated and unnecessarily complex.

The author, Nikolaus Wirth, details some of the reasons for the inefficiency trend. The first reason mentioned is the "need" for features -- any feature that a customer thinks should be there, companies will try to implement, no matter what the cost (self-inflicted complexity). Often products would be better off leaving out "features" which are unused by the majority of the user-base. Another reason of complexity in software design is simply because the company developing the software wants their users to be dependent on them. "Customer dependence is more profitable than customer education." Another reason for complexity is that software engineers are typically not given enough time to design elegant and flexible solutions but are instead pressured to deliver the minimum viable product in the shortest time possible. While this approach may be satisfactory for the short term, when a new feature needs to be added "quick fixes" are performed which increase overall complexity and inefficiency of the program.

The second portion of the paper goes into good software design practices and features insight about what Wirth learned while developing Project Oberon (barebones operating system implemented by 2 people in 3 years). The primary goal of Oberon was to prove that software can be developed with a fraction of memory capacity and processing power required without sacrificing flexibility, functionality, or user convenience. The main sticking point of Oberon for me is its flexibility by creating a core which can be extended through the use of modules. This is typically what we consider an object oriented approach, however because Wirth was dissatisfied with the non-strict typing of C/C++, he developed his own language to enforce his own set of design rules. Doing this gives his system many advantages in terms of simplicity of implementation such as the ability to load modules on demand and ensuring that every module is only in memory once.

Overall the project was a success at proving that given good design practices, a large software system can be developed quickly and efficiently. In fact, having a large team size in which each member of the team is highly specialized can in some cases be detrimental to progress. By keeping the design team small (2 members) and enforcing good design principles, he was able to design and implement a new programming language and a new operating system in less time than similar operating systems. His system is also more efficient both in program execution and in memory consumption (and thus power consumption) than many similar systems without any sacrifice in flexibility or functionality.

"A system that is not understood in its entirety, or at least to a significant degree of detail by a single individual, should probably not be built."