Software Evolution. A Road Map

Harry M. Sneed
Institute for Wirtschaftsinformatik
University of Regensburg, Bavaria
eMail: Harry.Sneed@t-online.de

Abstract: This position paper proposes that the evolution of software systems is determined by the type of application and that one must distinguish between static and dynamic applications. Static evolution can be anticipated and planned. Dynamic evolution is by nature ad hoc and must be reactive. Some degree of stability can be brought to dynamic systems by dividing them into components based on their propensity to change. In the world of electronic commerce evolution has become the standard mode of development.

Evolution depends on System Type
The question of how software should evolve depends on the type of application for which the software is being written. There is no one right answer to this question. Some applications such as embedded real-time software, process control software, or commercial back office software can evolve according to the traditional concepts of software engineering, i.e. their evolution can be anticipated. Other applications such as software tools, websites and commercial front office systems must evolve in a reactive mode, since their evolution can not be anticipated.

All Software is embedded
All software systems are by nature part of another system. This other system can be a device, a machine, a mechanized production process, a communication network or some form of human organization. Devices, machines and mechanized production processes, such as a milling street or a production line, are unlikely to change a lot since the hardware they depend on is relatively expensive and can not be readily replaced. Communication networks evolve with the technology behind them, so that the software supporting them must be frequently adapted. Computer systems, especially those at the PC level are changing every two years as new operating systems, database systems and middleware systems appear on the market. Thus, tools which depend on these basis systems or which are intended to extend them in some way must evolve with them or become obsolete. A good example here are the file names in the MicroSoft environment which have evolved within seven years from eight character restricted strings to 255 character strings of anything including spaces.

Human organizations are changing almost monthly as enterprises are reorganized, merged, dissolved and streamlined to compete in a global market. Hardly anyone is capable of predicting the next six months, let alone the next year. Therefore, the software embedded in such organizations, especially that which is interacting with the outer world, must evolve continually in a non predictable direction. There is no alternative here to a reactive evolution.

Alleviating the Pain of Evolution
There is no way to escape the pain of evolution. Software engineers are paid well to suffer and should not complain. One thing they could do to alleviate the pain and reduce the costs of evolution is to try and anticipate what will be most exposed to change and what not. Software systems should be constructed on the basis of a core, inner layers and outer layers. The core contains that functionality which is least likely to change and is protected from change by wrapping. The inner layers are subjected to periodic change, whereas the outer layers are in a state of continual evolution. This is not always easy to accomplish, but if it can, it leads to better quality and reduced costs. In any case it is worth the effort.

Conclusion
I conclude that for most commercial software systems, the initial development is only the first step in a long journey. Of course, if it is in the wrong direction, there will hardly be any chance to reach the final goal. Even if it is in the right direction, further development can cause it to go off course. Consequently, only a minority of the software development projects started will eventually lead to a successful product. Next to the laws of evolution propagated by Lehman and Belady, there are also the laws of evolution laid down by Darwin. Only the most suitable software will survive and that depends upon many factors both of a technical as well as of a social-economic nature, most of which we have yet to identify.