On the Role of Evolvability for Architectural Design

Stephan Bode

Faculty of Computer Science and Automation
Technical University Ilmenau
P.O. Box 100565, 98684 Ilmenau
Stephan.Bode@TU-Ilmenau.de

Today a software system usually represents a major investment for an enterprise, which thus has to be profitable. Therefore, if a system is build once, it has to remain usable for a long time. However, there is a permanent demand for software changes, because of the technological evolution, the optimization of processes, or because of the integration of existing systems into the development of new software architectures.

In the scope of software maintenance, changes frequently have to be performed with low effort and in short time frames. Therefore, changes can lead to deterioration in the structure of the software, which in turn hampers or inhibits further changes. This effect is called architectural decay or architectural drift. However, a replacement of affected systems by a completely new development and, hence, the prevention of this effect generally is not an acceptable solution. A replacement potentially comes along with a financial risk for the enterprise and mostly is impossible with regard to deadlines or budgets. That is why the software, and especially its architecture, has to be able to cope with frequent requests for change to permanently stay usable.

Because of the frequent software changes and of the unsuitability to replace existing software systems, a demand beyond software maintenance is arising to keep the software system in a condition that allows quick and easy changes for the long term. Therefore, it is the author's opinion, that today's practice in software maintenance and to achieve software maintainability are not enough for the architectural design of long-living software systems, and we additionally have to consider *evolvability* as an important quality goal for the architectural design phase.

In the full paper we discuss the current situation and practice in software engineering considering maintainability and argue why the efforts made today are not enough. We explain why a distinction between software evolution and software maintenance as well as between evolvability and maintainability is necessary due to their different focus. We discuss, why a consideration of evolvability among other quality attributes as a major goal for architectural design is necessary to strengthen a software system's ability to adapt to frequently changing needs and remain usable. Moreover, we state, that we have to further analyze evolvability as a quality goal and its sub-characteristics for a better understanding, to sharpen developers' awareness and for a better support during the design phase. Beside the argumentation pro evolvability, the paper provides advice for some actions to improve the current situation. It tries to point in a direction for an integrated method for architectural design considering evolvability.