Gute und schlechte Software

Stefan Menzl, Dipl. El.-Ing. ETH

Mängel, die erst nach Jahren zum Vorschein kommen, sind auf dem Bau wohl bekannt und dementsprechend gefürchtet. Die SIA-Norm 118 regelt die Verjährung solcher verdeckter Mängel. Bei Software-Entwicklungsprojekten gibt es keine solche Norm, sehr wohl aber verdeckte Mängel.

Folgende Situation ist leider keine Seltenheit: Eine Software-Entwicklung wird in Auftrag gegeben, die Software mit Verspätung geliefert – und nach der Installation funktioniert sie nur leidlich. Es braucht noch einige Anpassungen. Die Anpassungen werden umgesetzt und plötzlich funktioniert dafür etwas anderes nicht mehr. Es braucht noch einmal ein paar Bug Fixes und diese verursachen wiederum neue Fehler. Die Erfahrung mit einer kleinen Änderung „ganz ohne Risiko“ verkommt zum Albtraum. Weitere Änderungen darf es nicht geben – das Risiko ist zu gross.

Was ist passiert? Die Qualitätssicherung wurde während der Entwicklung vernachlässigt. Die Software wurde geschrieben, ohne Rücksicht darauf, dass sie in Zukunft nicht nur von Maschinen, sondern auch von Menschen verstanden werden muss. Der Schlamassel beginnt üblicherweise damit, dass mangels eines griffigen Pflichtenhefts keine saubere Architektur erstellt wird. Die Anforderungen werden im Verlaufe der Entwicklung ad hoc ermittelt, die Architektur jeweils etwas „erweitert“. So entsteht ein unübersichtlicher Flickenteppich, den kurz nach der Fertigstellung – oder schon davor – niemand mehr versteht.

Wie aber schützt man sich davor? Der Bauherr hat dieselben Probleme: Er versteht das Handwerk des Auftragnehmers nicht in der Tiefe, muss sich also auf die Fachkenntnisse des Handwerkers, des Bauführers und des Architekten verlassen. Bei komplexen Vorhaben leistet er sich dann einen zusätzlichen Sachverständigen, einen Bauherrenberater. Bei der Software-Entwicklung kleinerer Applikationen kommt oft alles aus einer Hand – wie also soll der Auftraggeber prüfen, ob die Qualität seiner Software stimmt? Wer sich keinen unabhängigen Berater leisten möchte, hinterfragt Folgendes:

- Gibt es eine saubere Dokumentation? Beginnend beim Lasten- und Pflichtenheft (das man verstehen kann und soll), einer Beschreibung der Architektur, der Schnittstellen, der grafischen Benutzeroberfläche, über einen Projekt- und Testplan bis hin zu den Abnahmekriterien müssen detaillierte und aktuell gehaltene Dokumente vorhanden sein. Die Dokumentation wird üblicherweise vor oder zumindest parallel zur Implementation geschrieben, nie nachher.
- Gibt es einen Entwicklungsprozess? Software-Entwicklung ist zwar unter Umständen schwierig, aber eine Kunst ist sie nicht. Denn das Vorgehen ist prozessorientiert, hat einen Ablauf und definierte Meilensteine. Die verschiedenen Modelle

zur Software-Entwicklung, teilweise mit furchteinflössenden Namen wie Extreme Programming, RUP oder SCRUM, zielen allesamt darauf ab, dem Prozess eine Struktur mit klar definierten Lieferobjekten zu geben. Lassen Sie sich die Prozesse erklären und die Lieferobjekte zeigen. Prüfen Sie zudem, auf welchen bestehenden Bibliotheken die Software aufbaut. Es ist wichtig, dass diese auch in Zukunft gewartet und aktuell gehalten werden. Was genau wird effektiv neu entwickelt? Und: Hat der Lieferant bereits Erfahrung damit und kann Referenzen vorweisen?

- Gibt es eine Prüfung durch Dritte? Lassen Sie die Arbeit der Entwickler zwischendurch von anderen Software-Ingenieuren prüfen. Sei dies in Form eines Sicherheitsaudits, eines Code-Reviews oder eines Q-Reviews. Wenn ein unabhängiger Software-Ingenieur nicht innert weniger Stunden einen groben Überblick bekommt, ist Vorsicht geboten.

„Drum prüfe, wer sich ewig bindet“ hat Friedrich Schiller schon vor 200 Jahren geschrieben – ohne Bezug auf die Probleme bei der Software-Entwicklung. Aber auch dafür gilt dieser Satz, denn wer eine Software programmieren lässt und einführt, kann normalerweise bald nicht mehr ohne diese Software arbeiten.