MICRO CONSULT

“In unseren Kursen sitzen die Entwickler, die dazulernen wollen.”

MicroConsult-Trainer und Projektcoach Frank Listing über Clean Code und wie er sich den idealen Ablauf eines Embedded-Projektes vorstellt

Womit hat man als Trainer im Bereich Embedded-Software täglich zu tun?

Frank Listing: Man bekommt sehr schnell mit, dass Software in deutschen Unternehmen des Maschinenbaus und der Automobilindustrie nach wie vor einen niedrigen Stellenwert hat. Die Software ist das, was am Ende dazukommt; das funktioniert auch nebenbei, kann man ja immer noch ändern. Aber so einfach ist es leider nicht.

Der Stand der Dinge in der Softwareentwicklung hat viel mit der erfolgreichen Maschinenbau-Branche hierzulande zu tun. Das spiegelt sich auch in den Ausbildungsstrukturen der Entwickler wider. Programmierer im Embedded-Bereich kommen hauptsächlich aus der Elektrotechnik und dem Maschinenbau. In der Minderzahl findet man hier Leute, die tatsächlich ein bisschen mehr in Richtung Software ausgebildet wurden. Viele lesen sich ihr Know-how an und kombinieren es mit dem, was sie vorher gelernt haben. Doch es reicht nicht aus, allein die Syntax der jeweiligen Programmiersprache zu beherrschen. Wenn ich Verfahren, Algorithmen und Vorgehensweisen nicht kenne, dann kann ich sie auch nicht anwenden. Und sie ergeben sich auch nicht immer intuitiv.

In Deutschland fehlen unterm Strich noch immer feste Ausbildungsstrukturen für Softwareprogrammierung in der Industrie. Selbst an vielen Hochschulen wird beim Programmieren ein Softwareprojekt oft nicht in seinem Aufbau bewertet, sondern nur das Ergebnis.

Softwareentwickler, die vorher reine Mechanik oder Elektrik entwickelt haben, sind für die Embedded-Programmierung nicht nur weniger gut ausgebildet, sie haben ja auch ursprünglich einen anderen Beruf gewählt. Mitunter sind sie gar nicht besonders motiviert, gute Software zu schreiben. Nur die wenigsten finden darin ihre neue Passion. Das Ergebnis kann man sich vorstellen.

Was kann man tun, um diese Leute für die Software-Programmierung zu begeistern?

Frank Listing: Wenn man einmal begriffen hat, dass Software schreiben ein kreativer Prozess ist, der nicht nach einem festen Schema ablaufen muss, wenn man es richtig macht, dann kann man damit wirklich schöne Dinge entwickeln. Und wir brauchen ja heute auch mehr Anwendungen, die verzwickte Aufgaben lösen müssen. Das kann manche motivieren, andere wiederum nicht. Auch in der Softwareentwicklung kann man den stupiden Weg wählen, der wenig Spaß macht. Die Software funktioniert dann vielleicht auch, aber die besseren Lösungen sind meistens auch die kreativen.

Bei einer Wald- und Wiesen-Ausbildung bekommt man diese Lust eben nicht vermittelt. Und dementsprechend wenig vorausschauend wird dann die Software programmiert – was sie mit der Zeit auch nicht besser macht. Das führt auch gleich zum nächsten Problem: Wenn schlecht dokumentierte Software in einem Arbeitsumfeld mit einer hohen Fluktuation entsteht, kann das katastrophale Folgen haben.

Wie würde man sich den idealen Ablauf eines Embedded-Projektes wünschen?

Frank Listing: Erst einmal gilt es, rechtzeitig über die Anforderungen nachzudenken. Oft fangen Teams zu programmieren an, ohne zu wissen, was es werden soll. Und wenn ich nicht genau weiß, was am Ende des Prozesses steht, dann kann es sein, das etwas Großes und Langsames dabei herauskommt.

Bei einem guten Embedded-Projekt steht am Anfang ein Systementwurf, also erst einmal die Idee, wie die Lösung prinzipiell aussehen soll. Dafür braucht man natürlich auch schon die Requirements als Voraussetzung für den Entwurf einer Architektur. Dann wird das Ganze gesplittet in die Hardware und die Software, deren Entwicklung dann weitgehend parallel läuft.

Der größte Fehler wird gemacht, wenn man Hardware und Software erst ganz zum Schluss wieder zusammenbringt, ohne beides vorher hin und wieder zumindest etwas zu synchronisieren und neue Anforderungen weiterzugeben. Sobald die Architektur ein bisschen mit ein paar Details verfeinert wird, kommen wir so langsam in das Programmieren rein. Softwareentwicklung ist an sich der gesamte Prozess vom Entwurf bis hin zum Ende der Tests. Getestet wird je nach Entwicklungsprozess entweder in der Implementierung oder kurz danach.

Es kommt auch oft vor, dass man zu früh Code schreibt, ohne genügend Anforderungen zu kennen. Es fehlen noch zu viele Requirements sowie Aufbau und Architektur der Applikation, aber man legt einfach schon mal los. An sich ist das der langsamere Weg. Als Antwort bekomme ich dann: Es musste schnell gehen. Wirklich schneller geht es aber, wenn man vorher gut darüber nachdenkt, was man genau umsetzen muss.

Was sollte man vermeiden, wenn ein Projekt mal in Schieflage gerät?

Frank Listing: Was man in diesem Moment nicht braucht, ist ein neues Tool, das einen angeblich aus der Misere rausholt. Mit einem neuen Tool kommt ein neuer Störfaktor hinzu, außer ich habe es vorher in Ruhe evaluiert und einstudiert und man integriert es erst in das Projekt, wenn man es gut kennt. Also: Programmiersprache beherrschen und eingesetzte Tools in- und auswendig kennen, sonst wird es chaotisch.

Welchen Rollen im Entwicklerteam sind sinnvoll?

Frank Listing: Im Idealfall habe ich natürlich vorher die Zuständigkeiten ausreichend definiert. Man braucht einen Softwarearchitekten, von dem auch jeder weiß, dass er diese Rolle innehat, einschließlich er selbst, versteht sich. Das können je nach Größe des Projektes auch mehrere sein. Er alleine oder mit einem kleinen Team ist für den Entwurf zuständig. Das Design verteilt sich dann bereits auf mehrere Köpfe, und die ganze Mannschaft übernimmt die Implementierung. Dieses Vorgehen gibt für das Projekt eine bereits einigermaßen vernünftige Struktur und Richtung vor.

Auf was sollte man sonst noch im Vorfeld achten?

Frank Listing: Ein weitverbreiteter Fehler besteht darin, dass die Tester oft viel zu spät reingeholt werden. Ein Tester sollte schon bei den Requirements darauf schauen können, ob die Software und ihre Architektur überhaupt testbar sind. Wenn das nicht der Fall ist, werde ich sie auch in Zukunft nicht vernünftig weiterentwickeln können. Die Realität sieht aber manchmal leider so aus, dass erst die vermeintlich fertige Software zum Testen freigegeben wird. Dadurch verliert man unheimlich viel Zeit. Wenn man die Tester stattdessen früher miteinbezieht, kann man bereits vorher Probleme ausschließen und frühzeitig beheben. Wenn die Tester das Problem bereits kennen, können sie sich schon vorher damit beschäftigen. So ist der ganze Zeitplan viel einfacher einzuhalten.

Clean Code steht für mich auch für eine gewisse Ästhetik. Es macht nicht nur Spaß, ihn anzuschauen, sondern er folgt auch gewissen Regeln, ist gut lesbar und enthält weniger Fehler. Mit dieser Lesbarkeit gehen auch andere Sachen einher: Man kann ihn einfach erweitern und muss ihn nicht verändern, um neue Dinge hinzuzufügen. Dazu braucht man nicht nur Erfahrung, sondern Freude am Programmieren überhaupt.

Das Schöne dabei ist: Die Lernkurve beim Programmieren von Software ist relativ steil. Und man kann sie anheben, wenn man mal ein Lehrbuch liest oder einen Kurs besucht. Manche Fehler muss man selber machen, andere Lösungswege kann man von anderen lernen. Und meistens macht man dann auch gleich einen großen Schritt nach vorne. In unseren Kursen sitzen zum Glück immer die Entwickler, die dazulernen wollen. Das erleichtert mir die Arbeit enorm und macht auch mehr Spaß.

MICROCONSULT

Autor: Frank Listing

Seit 2002 ist Frank Listing Trainer und Projektcoach bei MicroConsult; seine Schwerpunkte sind Microsoft-Plattformen, Software-Architekturen, objektorientierte Programmierung und Testen von Embedded-Systemen. Außerdem ist er Spezialist für die Themen C++, C#, Finite State Machines, Clean Code und .NET. Sein Wissen gibt er regelmäßig in Publikationen und Flachvorträgen z.B. auf dem Embedded Software Engineering (ESE) Kongress weiter.

Weiterführende Informationen

MicroConsult Training: Clean Code für C-Programme

MicroConsult Training & Coaching zum Thema Softwarequalität

MicroConsult Fachwissen zum Thema Softwarequalität

MicroConsult Training & Coaching zu Embedded-Programmierung

MicroConsult Fachwissen Embedded-Softwareentwicklung