

80 zu 20 – Boeing und Bombardier ignorieren eine alte Faustregel der Softwareentwicklung!

Boeing und Bombardier bringen es an den Tag: Die Achillesferse heutiger Technologie ist die Software. Und das in einer Zeit euphorischer Versprechungen der boomenden Künstlichen Intelligenz.

Wobei wir auch die vermutlich für den Absturz verantwortliche Software von Boeing zur KI zählen müssen, denn es geht um Entscheidungen aufgrund von externen Sensorsignalen, also dasselbe wie bei neuronalen Netzen von Robotern. Im Grunde genommen ist jede Software Künstliche Intelligenz, den Begriff könnte man sich ebenso gut sparen.

Der Skandal bei Boeing ist wohl der, dass nur ein Sensor angezapft wird, und im Falle, dass dieser Fehlinformationen liefert, natürlich sofort falsche Entscheidungen zustande kommen.

Dass Bombardier es nicht einmal schafft, Türen korrekt zu schliessen, ist noch erstaunlicher. Geschlossen werden sie wahrscheinlich korrekt, das ist schliesslich bewährte Mechanik, nur die Software, die den Status richtig interpretieren und anzeigen sollte, versagt permanent.

Unsere bewährten industriellen Produkte sollen mit den Segnungen der digitalen Revolution veredelt werden, kommen aber gerade dadurch immer mehr in Schieflage und enden massiv verschlechtert beim Kunden. Software wird zunehmend der schwächste Teil nicht nur beim Transport, auch bei Dienstleistungen und selbst der Elektronik, wie das Beispiel Swisscom zeigt, Telefonie und Fernsehen sind in den 70er Jahren nie ausgefallen.

Boeing wird sich nicht herausreden können, die Software wurde ganz sicher mies getestet, wenn es ganz schlimm kommt, wurden Testresultate sogar ignoriert. Noch schlimmer, sie wurden auch von der amerikanischen Luftfahrtbehörde FAA ignoriert.

In der heutigen Softwareentwicklung inklusive Testing nichts aussergewöhnliches, da ist Schlamperei an der Tagesordnung. Ich kenne das bei Banken. Für die Öffentlichkeit neu, dass es selbst in der Luftfahrt und dem Bahnwesen eingerissen hat, wo Menschenleben gefährdet sind.

Die Gründe dafür: Die Systeme sind sehr komplex, das Softwareengineering ist im Grunde genommen aber noch eine junge Disziplin und kann die nötige Kompetenz schlicht nicht erbringen. Und das Testen wird sträflich vernachlässigt. Der Testaufwand würde exponentiell zur Komplexität einer Software steigen, im IT-Alltag ist es umgekehrt, je komplexer ein System, je nachlässiger und schneller wird getestet, oft aus blosser Resignation und Hilflosigkeit.

Wie schlecht heutzutage Software sein kann, zeigt Bombardier, die mit blossem Türöffnen nicht zurande kommen. Dort gibt es den Vorwurf der Entwickler, dass die SBB zu viele Anforderungen hatte. Als ehemaliger Programmierer kann ich mir vorstellen, wie verzweifelt das Engineering über nicht durchdachte Luxusanforderung fluchte und diese gehässig und übereilt einzubauen suchten, gehetzt vom Termindruck und einem nie endenden Anforderungskatalog.

Früher gab es eine Faustregel, sie hiess 80:20 und bedeutete: Im Normalfall lassen sich etwa 80 % des Anforderungskatalogs einer Software relativ sicher und zügig implementieren, bei den letzten 20 % wird es sehr aufwendig und teuer. Eigentlich sollte das ein verantwortungsbewusstes Projektteam dem Auftraggeber entsprechend kommunizieren, dieser wird dann, wenn er vernünftig und realistisch ist, auf die Luxusanforderungen verzichten. Ich

nehme an, im Falle Bombardier/SBB wurde das nicht gemacht, sondern aus falschem Ehrgeiz und Stolz hat man alles brav realisieren wollen. Die jahrelange Verzögerung scheint diesen Verdacht zu bestätigen.

Was mich diese Skandale lehren: Fliegen nur im Notfall, beim öffentlichen Verkehr genügend Zeitreserve einplanen, und bis auf Weiteres: Hände weg vom Steuer eines selbstfahrenden Autos ... sorry, die haben ja keine Steuerräder mehr. Zum Glück hab ich noch ein Velo im Keller - ohne Software!