SOFTWAREENTWICKLUNG

Großes Missverständnis

Wie Softwareentwicklung funktioniert, meinen viele zu wissen – oft ein Irrglaube mit katastrophalen Konsequenzen.

Meine treuen Leser wissen längst, dass ich hier immer wieder über Themen schreibe, die mich in meinen Projekten gerade in irgendeiner Art und Weise beschäftigen. Das gilt für viele Dinge im Bereich der Qualität, der Architektur und natürlich auch für den Entwicklungsprozess. Aber die große Klammer, die all diese Begriffe und viele mehr zusammenfasst, hat nur einen Namen: Softwareentwicklung. Wie diese „ordentlich" ablaufen soll, glauben viele zu wissen. Leider jedoch ist dem oft nicht so, und ich sehe Woche für Woche die oft fatalen Konsequenzen aus diesem Irrglauben.

Das typische Projekt

Um das Ganze anschaulich zu machen, skizziere ich hier ein Projekt, das ich in dieser Form allein im ersten Halbjahr 2023 schon dutzendfach gesehen habe. Ein Unternehmen ist in den letzten Jahren stark gewachsen, aus wenigen Entwicklern sind viele Entwickler geworden und mit deren steigender Zahl stellen sich immer mehr Probleme in den Projekten ein. Die Anforderungen sind schlecht gefasst, die Entwickler zu langsam, Deadlines werden nicht eingehalten, der Quellcode wird immer komplexer und schwieriger zu warten und zu erweitern. Ist ein Mitarbeiter im Urlaub, erkrankt oder verlässt er das Unternehmen, ist die Verzweiflung groß: Nahezu niemand kann den Quellcode und damit das Projekt des Kollegen verstehen und es somit auch nicht erweitern und warten. Welche Funktionen in welchem Softwareprojekt implementiert wurden? Weiß auch niemand. Und einen wirklichen Entwicklungsprozess gibt es auch nicht, alles funktioniert irgendwie – hauptsächlich, weil sich bestimmte Mitarbeiter schon seit Langem kennen und daher gut miteinander zusammenarbeiten können. Kommt in diesem Gefüge ein neuer Mitarbeiter dazu, gibt es oft zahlreiche Probleme, die sich erst nach langer Zeit langsam auflösen.

Auch wenn dieses Beispiel hoffentlich nicht zu einhundert Prozent auf Sie zutreffen wird, ist Ihnen bestimmt schon mal ein Projekt begegnet, bei dem zumindest einige dieser Punkte zutrafen. Ich für meinen Teil höre als Berater gefühlt jede Woche eine solche Geschichte. Das eigentliche Problem ist eines, für das es allerdings keine schnelle Lösung gibt: Nicht genau zu wissen, wie Softwareentwicklung funktioniert.

Die Unwissenheit

Die meisten Unternehmen haben wenig bis gar keine Ahnung davon, wie ordentliche Softwareentwicklung funktioniert. Aber das ist nicht schlimm, sondern sogar sehr verständlich. Nicht, weil dieses Wissen so besonders schwer zu erlernen ist oder man dazu über eine hohe Intelligenz verfügen muss, sondern weil das ein ganz normales Aufgabenfeld wie jedes andere ist, bei dem man sich einarbeiten und eben lernen muss, wie es funktioniert.

Aber genau das ist das Problem. Besonders in Zeiten, bei denen Projekte mit immer weniger Personal in immer kürzerer Zeit umgesetzt werden müssen, wird Weiterbildung oder Wissenserwerb allgemein auf die Themen beschränkt, die einen direkten und offensichtlichen Nutzen haben: Programmiersprachen, Frameworks, Plattformen, Technologien. Das sind die Themen, mit denen die funktionalen Anforderungen umgesetzt werden und somit letztlich Geld verdient wird. Weiterbildung, die zeigt, wie das große Ganze funktioniert, wird dabei meist komplett vernachlässigt – oft Wissen, das niemand in Studium oder Ausbildung lernt. Dass zur Entstehung eines ordentlichen Softwareprojekts noch wesentlich mehr gehört, als eine Programmiersprache zu beherrschen, ein paar Frameworks zu kennen und dem Entwickler irgendwie die Anforderungen der Stakeholder klarzumachen, erkennen nur die wenigsten.

Die Komplexität der Softwareentwicklung

Verstehen Sie mich nicht falsch: Ich sehe nicht Softwareentwickler in der Pflicht, dieses Wissen zu besitzen oder aufzubauen, sondern das gesamte Unternehmen. Ich sehe ständig Unternehmen, die mit einer entwickelten Software ihren primären Umsatz erzielen, jedoch ist das Wissen um Schlüsselthemen wie Entwicklungsprozesse, Anforderungen, Visionen, Architektur, Qualitätssicherung, Deployment, Support und so weiter kaum bis gar nicht vorhanden, und am Ende werden diese Dinge oft irgendwie von den Softwareentwicklern erledigt. Sie wären erstaunt, wie oft ich schon von Projektbeteiligten die Aussage gehört habe: „Wir wundern uns selbst, wie wir überhaupt mit Software Geld verdienen."

Wenn Sie Softwareprojekte ordentlich organisieren wollen, benötigen Sie eine Produktvision, sehr gute Anforderungen, eine passende Architektur, guten und qualitativ hochwertigen Quellcode, irgendeine Art von Teststrategie, die Möglichkeit, die Anwendung schnell und häufig bereitzustellen, und natürlich einen guten Support, der die Kunden zufriedenstellt. Neben diesen ganzen „harten Punkten" sollten dabei aber auch die Mitarbeiter – und damit der Faktor Mensch – nicht zu kurz kommen. Jeder dieser angesprochenen Punkte erfordert einen oder manchmal sogar mehrere Mitarbeiter, die speziell für diese Rolle ausgebildet wurden oder entsprechendes Wissen besitzen.

Die „Kleine Projekte"-Ausrede

Immer wieder sagen mir in Beratungsprojekten Personen, die meist eine Führungsverantwortung haben: „Ja David, in großen Projekten kann ich mir das schon vorstellen, aber wir sind ja nur ein paar Entwickler, da braucht man doch so was nicht." Ich kann aus Erfahrung mit absoluter Überzeugung sagen: Nein! Das stimmt so überhaupt nicht!

Egal wie groß ein Projekt ist, es geht immer darum, die Idee einer Person mithilfe der Softwareentwicklung in ein fertiges Softwareprodukt umzusetzen. Ob dieser Transformationsprozess von hundert Entwicklern geleistet wird oder von nur einem einzigen, spielt keine Rolle. Aktivitäten wie Vision, Anforderungsanalyse, Architektur, Implementierung, Test, Bereitstellung und Betrieb müssen immer ausgeführt werden – notfalls eben nur von einer Person.

Natürlich zeigt die Praxis, dass besonders kleine Teams oder gar Einzelpersonen einzelne oder mehrere dieser Aspekte nicht berücksichtigen, aber diese Projekte enden meist als monolithische Legacy-Anwendung, die dann für viel Geld neu aufgesetzt werden muss, wie in einem großen Team. Der Unterschied hierbei ist lediglich, dass ein Team aufgrund der höheren Entwicklungsgeschwindigkeit den „Todeszeitpunkt" meist wesentlich früher erreicht, als es ein einzelner Entwickler tut.

Fokus auf Umsatz

Das große Problem an dieser falschen Sichtweise ist, dass es ja augenscheinlich erst mal genau so funktioniert: Es entsteht ein Produkt, dieses erzielt Umsatz, der Kunde ist zufrieden, das Unternehmen wächst. Alles klingt nach einer Erfolgsstory, aber kaum jemand im Unternehmen bemerkt, dass dies passiert, ohne dass wirklich das Know-how im Unternehmen vorhanden ist, wie Software „ordentlich" entwickelt werden sollte. So wächst nicht nur der Umsatz und das Unternehmen, sondern gleichzeitig mehren sich auch die Probleme in der Softwareentwicklung.

Eine moderne Softwareentwicklung braucht eine Rolle, die den Entwicklungsprozess ständig optimiert – wie in Scrum der Scrum Master (egal ob Sie Scrum machen oder nicht). Alle im Unternehmen beteiligten Personen sollten gemeinsam eine Vision des Produkts entwerfen, damit jeder ein glasklares Verständnis davon hat, wo es mit dem Produkt hingehen soll, und auch, wo es nicht hingehen soll. Anforderungen sollten ordentlich aufgenommen, dokumentiert und dann an die Entwickler übergeben werden. Ohne eine passende Architektur werden Sie Ihr Projekt irgendwann neu entwickeln müssen, und das wird richtig teuer; ohne Tests werden Sie irgendwann Probleme mit der Stabilität bekommen. Auch ohne all diese Dinge läuft das Projekt erst mal, am Anfang funktioniert die Softwareentwicklung immer irgendwie. Langfristig jedoch, das garantiere ich Ihnen, wird das nicht gut gehen, oft mit katastrophalen Folgen.

All diese Dinge, die ich oben genannt habe, haben eines gemeinsam: Vision, Prozesse, Architekturen, Tests und vieles Weitere bringen keinen direkten Umsatz, sondern kosten enorm viel Geld, um diese Rollen zu besetzen; und trotzdem werden Sie am Ende des Monats keine Lizenz mehr verkaufen und somit auch nicht mehr Umsatz generieren. Folglich werden diese „unnützen" Kosten oft gescheut. Die Menge an Projekten, bei denen ich mitgewirkt habe, die sich den Scrum Master gespart haben oder ihn halbherzig einem Entwickler „nebenbei aufs Auge gedrückt" haben, kann ich schon lange nicht mehr zählen; Entwickler, die ihre eigenen Anforderungen aufnehmen, ebenfalls nicht; Projekte ohne Architektur übrigens am meisten. Zusammengefasst: ein Trauerspiel!

Der wirtschaftliche Schaden

„Über was regt der David sich hier überhaupt auf?", mögen Sie sich fragen, schließlich verdiene ich doch mein Geld mit genau solchen Beratungen. Das ist auch vollkommen richtig, ich sehe aber immer wieder, welche wirtschaftlichen Schäden in den Unternehmen dadurch entstehen. Weil Entwicklungsteams langsam sind, weil Dinge mehrfach implementiert werden, weil weniger Projekte gemacht werden können, weil Entwickler unglücklich sind, und leider auch sehr oft: Weil Unternehmen wegen ihren Fehlern irgendwann die Türen schließen müssen – für immer. Weil das einzige Produkt neu entwickelt werden müsste, für viel Geld, das aber nicht da ist. Weil irgendwann ein früher kleinerer Mittbewerber den ganzen Markt erobert hat, weil er wusste, wie Softwareentwicklung funktioniert. Weil irgendwann ganze Entwicklungsteams wegen Unzufriedenheit das Unternehmen wechseln.

So leider auch in dieser Woche, in der ich dies hier schreibe; bei einem Kunden, der sich einfach nicht ändern wollte; bei dem die Entscheider so viele Chancen hatten, aber immer den falschen Weg gewählt haben – aus Mangel an Wissen zur richtigen Softwareentwicklung. So viel Arbeit und Engagement über viele Jahre von so vielen guten Softwareentwicklern – umsonst! Die Situation wäre absolut vermeidbar gewesen, aber leider war besonders im oberen und mittleren Management genau das Wissen zu „ordentlicher" Softwareentwicklung nicht vorhanden. Ein Trauerspiel.

Machen Sie es besser, falls noch nicht geschehen. Erkennen Sie, was „ordentliche" Softwareentwicklung ist, was Ihnen noch fehlt, und handeln Sie dementsprechend. Was Sie dazu wissen müssen, habe ich in all den Artikeln an dieser Stelle zu vermitteln versucht; Sie finden die Ratschläge auch in meinem YouTube-Kanal [1]. Sie werden es garantiert nicht bereuen – viel Erfolg dabei! ■

[1] David Tielke bei YouTube,
https://www.youtube.com/@DavidTielke

David Tielke
ist freiberuflicher Berater und von Microsoft zertifizierter Trainer für die Anwendungsentwicklung auf der .NET-Plattform. Darüber hinaus hat er sich auf die Bereich Softwarearchitektur, Softwarequalität und ALM spezialisiert.
mail@david-tielke.de

dnpCode A2309DDD