„IN ÖKOSYSTEMEN ZU DENKEN WIRD IMMER WICHTIGER.“

Ein Gespräch mit Dr. Elmar Hubner, Geschäftsführer der ROI Management Consulting GmbH, Wien, über Software Excellence, Projekterfahrungen und Lessons Learned.

DIALOG: Herr Dr. Hubner, was sind die wichtigsten Fragen, die man sich als Industrieunternehmen stellen sollte, wenn man sich erstmals mit dem Thema Softwareentwicklung auseinandersetzt?

EH: Tatsächlich haben viele Industrieunternehmen heute bereits Erfahrung mit der Entwicklung von Software, etwa bei der Steuerung von elektronischen Geräten. Neu ist allerdings die wachsende Anzahl smarter Produkte, die untereinander vernetzt sind, ans Internet angebunden werden und Daten austauschen – sowie die dahinterliegenden Geschäftsmodelle, mit denen sich viele Firmen bisher nicht auseinandersetzen mussten. Zunehmend ist Hardware nicht mehr der bestimmende Faktor. Der technische Lifecycle eines Geräts ändert sich, mit Auswirkungen auf das Engineering und die Organisation. Hier betreten aktuell viele Unternehmen Neuland. Sie müssen das Thema Software neu denken und erstmals eine Softwarestrategie für ihr Unternehmen entwickeln.

Eine zentrale Frage lautet dabei: Wo will ich hin und welche Kompetenzen brauche ich dafür? Was kann inhouse entwickelt werden? Was sollte man extern vergeben? Bei diesen Make-or-Buy-Entscheidungen geht es einerseits um Kosten und Produktivität, denn Entwicklungsressourcen, speziell im Softwarebereich, sind begrenzt und hart umkämpft. Andererseits ist zu überlegen, was im Hinblick auf eine nachhaltige Softwarearchitektur und Sicherheitsaspekte im Haus verbleiben muss und was zugekauft werden kann. Grundsätzlich ist es wichtig, sich bei der Entwicklung auf wettbewerbskritische Kompetenzen zu konzentrieren und – dort, wo es möglich ist – auf vorhandene Lösungen zurückzugreifen. Man muss z.B. kein eigenes Bezahlsystem oder ein User Access Management entwickeln, dafür gibt es passende Standardmodule. Doch auch diese Komponenten können Fehler haben und darauf muss man sich vorbereiten.

Die zweite große Frage ist, wie man die Entwicklung angeht. Hier geht es um Effizienz und die passenden Methoden der Softwareentwicklung. Eine große Rolle spielen dabei die Schnittstellen bzw. Integrationen von Soft- und Hardware-Artefakten innerhalb der Organisation. Dort liegen typische Problemquellen, da beide Entwicklungsströme i.d.R. einer anderen Logik folgen – doch das könnte sich künftig auch ändern.

DIALOG: Was sind die ersten Schritte auf dem Weg von einer reinen Hardware- zu einer Software-getriebenen Company?

EH: Wenn man die Klarheit über die Softwarestrategie hat, muss man den Reifegrad des Unternehmens in Bezug auf die Softwareentwicklungsmethoden und die Organisation kennen. Viele Unternehmen merken erst im Laufe der Zeit, dass es bei Softwareentwicklung um viel mehr geht als die reine Applikation. In unseren Projekten stellen wir häufig fest, dass wichtige technische Kompetenzen gar nicht vorhanden sind, z.B. hinsichtlich Cloud-Technologien, Cyber Security oder der Funktionsweise von „Software as a Service“. Zudem sollte man sich bewusst machen, dass eine Software kontinuierliche Wartung benötigt und potenziell weiterentwickelt wird. Produkte erhalten regelmäßig neue Funktionalitäten, sie müssen auf neuen Plattformen laufen, benötigen neue Kompatibilität. Dieser Prozess ist nie abgeschlossen. Man benötigt eine Produktstrategie und ein Produktmanagement. Zudem braucht man ein Release Management und entsprechend qualifizierte Menschen, u.a. für Wartung, Dokumentation und Software/IT Operations. Man muss also die gesamte Organisation richtig aufstellen, um Software schnell entwickeln und effizient betreiben zu können. Die große Herausforderung liegt dabei oft weniger in den technischen Fragen, als vielmehr darin, aus der heutigen Organisation in die neue Struktur zu kommen. Da geht es dann um einen Transformationsprozess.

DIALOG: Wie kann denn die Organisation dazu beitragen, dass Software-Teams sinnvoll und produktiv arbeiten können?

EH: Die Organisation sollte der Entwicklung modularer, skalierbarer Softwarestrukturen dienen. Es ist entscheidend, ein ordentliches Produktmanagement auch für die Software zu etablieren. Dies beinhaltet die Anforderungen an die Software und die Architektur des Systems. Damit man ein System warten, updaten und externe Ressourcen einsetzen kann, muss man modular denken. Mitunter findet man in Unternehmen über Jahre gewachsene Software-Monolithen vor, die weder aktualisierbar noch skalierbar sind, weil alle Funktionalitäten zusammenhängen. Da kann es vorkommen, dass Hunderte Mitarbeiter nonstop an der Wartung der Software arbeiten, aber keinen echten Output generieren, weil sie permanent Altlasten – die sogenannte Technical Debt – bereinigen. Wenn man einen Punkt überschreitet, kommt man da kaum noch heraus. Solche ineffizienten Strukturen muss man erkennen, rechtzeitig auseinandernehmen und durch modulare Strukturen

ersetzen. Modulare Architekturen sind auch die Voraussetzung für die Skalierbarkeit einer Organisation. Genauso wird dadurch die Einbindung externer Lösungen und damit die Nutzung von weiteren Innovationsquellen möglich. Doch eine nachhaltige Architektur erfordert eine Menge Vorarbeit, geistige Leistung und kontinuierliche Pflege.

DIALOG: Immer wieder kommt es vor, dass Software beim Release nicht wie geplant funktioniert oder dass Systeme nach einem Update ausfallen. Wo liegen die Gründe dafür?

EH: Dass eine Software beim Launch versagt, kann ganz banale Gründe haben. Vielleicht wurden in der Planung die späteren Zugriffszahlen unterschätzt. Möglicherweise wurde nicht hinreichend getestet. Man darf auch die Variabilität und Komplexität der Produkte im Feld nicht unterschätzen. Zudem läuft Software nicht nur mit Code, sondern mit vielen weiteren Daten, die mitunter nicht berücksichtigt werden. Das Testen von Software ist in der Tat eine große Herausforderung. Automobilhersteller z.B. verlassen sich niemals nur auf reine Softwaretests, sondern prüfen die Software vor einem Release immer im Kontext des realen Fahrzeugs. Aus den unzähligen möglichen Fahr- und Bediensituationen, die sich unter realen Bedingungen ergeben, entstehen zu viele Fehlerquellen. Da kann man sich nicht nur auf eine Simulation verlassen. Zur Qualitätssicherung setzen manche Unternehmen, z.B. in regulierten Branchen, auf das Prinzip des Pair Programming bzw. Pair Fixing, bei dem zwei Entwickler nach dem Vier-Augen Prinzip gemeinsam an einem Code arbeiten. Es gibt jedenfalls Wege, die Risiken zu minimieren.

DIALOG: Welche Synergien oder Skaleneffekte lassen sich denn in der Softwareentwicklung nutzen?

EH: In vielen Fällen können Entwicklerteams auf umfangreiche Funktionsbibliotheken zurückgreifen. Und zunehmend wird der Entwicklungsprozess durch Automatisierung unterstützt. So lässt sich Code in den verschiedenen Phasen des Software-Delivery-Prozesses mittlerweile automatisiert auf formale Fehler oder mögliche Sicherheitslücken überprüfen.

DIALOG: Hardware- und Softwareentwicklung folgen teils unterschiedlichen Prozessen und Methoden. Wenn ein Produkt wie z.B. ein Fahrzeug zu einem bestimmten Zeitpunkt serienreif sein und in die Produktion gehen soll, müssen alle auf diesen Punkt hinarbeiten. Wie bekommt man das harmonisiert?

EH: Das funktioniert über Synchronisationspunkte, die im Entwicklungsprozess in regelmäßigen Abständen positioniert werden. Bei großen Projekten können das im Zeitablauf mehrere Hundert sein. Um bei dem Beispiel zu bleiben: Wenn Sie einen Motor zusammen mit der Abgasnachbehandlung testen wollen und deren Softwareversion nicht mit der des Motors kompatibel ist, kann es sein, dass der Motor beim Teststart nicht anspringt. Insbesondere die Synchronisation der Hardware- und Softwareentwicklung zu managen ist eine hohe Kunst und eine große Herausforderung für die Planung. Wenn die Software- und Hardware-Teams jeweils auf ihrer Seite verharren, funktioniert es nicht. Beide Seiten müssen sich jeweils in die andere Welt hineindenken. Gelingt die Integration, lässt sich großes Potenzial heben.

DIALOG: Brauchen wir also zunehmend Menschen, die mit der Software- und Hardware-Welt vertraut sind und beide Bereiche wirklich verstehen?

EH: Ja, absolut. Ein integriertes Software Engineering und eine skalierbare Systemarchitektur aufzubauen sind sehr anspruchsvolle Aufgaben. Um übergreifende Systemkompetenz zu entwickeln, braucht man nicht nur die entsprechenden Softwarekenntnisse, sondern auch das individuelle Domänen-Know-how der unternehmensspezifischen Prozesse und Produkte. Die Leute, die sich in beiden Welten bewegen können, kann man nicht einfach schnell einkaufen.

DIALOG: Verschmelzen Hardware und Software zunehmend oder verschiebt sich eher die Rangordnung zwischen den beiden Bereichen?

EH: Das hängt vom Unternehmen und vom Produkt ab – und davon, über welche Merkmale sich das Unternehmen im Markt differenzieren möchte. Am Ende entscheidet darüber die Sicht des Kunden bzw. des Nutzers. Bei manchen Produkten, etwa bei bestimmten Automodellen, standen die mechanische Exzellenz und Aspekte wie das Fahrverhalten im Vordergrund.

Zukünftig werden jedoch softwarebasierte Funktionalitäten, wie z.B. Assistenzsysteme, den Kundennutzen bestimmen. Bei Produkten wie etwa Smartphones leitet sich der Kundennutzen mittlerweile nahezu vollständig aus der Software ab. Klar ist, dass die Bedeutung von Software bei allen technischen Produkten weiter zunehmen wird.

Ein Beispiel ist die sogenannte Funktionsintegration: Wo man früher für zehn Funktionen zehn verschiedene mechanische Knöpfe hatte, kann man heute mit einer Softwareapplikation über einen Touchscreen alle Funktionen bedienen. Damit steigt die Bedeutung der Software, aber auch ihre Komplexität.

DIALOG: Wenn wir den größeren Kontext betrachten, welche Herausforderungen ergeben sich durch Entwicklungen wie das Internet of Things an die wachsenden Software-Ökosysteme?

EH: Die Interoperabilität zwischen Plattformen und in diesem Zusammenhang auch die Fähigkeit, Produkte in Ökosysteme einzubinden, sind entscheidende Punkte. Überhaupt in Ökosystemen zu denken wird immer wichtiger. Wenn Sie heute einen Fernseher kaufen, können Sie diesen selbstverständlich über Android und iOS ansteuern. Vor der gleichen Herausforderung stehen Industrieunternehmen. Es geht dabei nicht nur um die Softwareentwicklung, mitunter kommen von der Softwareseite ganz neue Wettbewerber ins Spiel. Auch hier sind die entscheidenden Fragen: Welche Kompetenzen brauche ich?

Will ich diese intern verankern oder am Markt einkaufen? Diese Ressourcen zu entwickeln, die entsprechende Architektur aufzubauen und die Schnittstellen zu managen, das sind Schlüssel zu zukünftiger Exzellenz in einer immer stärker Software-geprägten Welt.

DIALOG: Kann es unter Qualitäts- und Effizienzgesichtspunkten auch sinnvoll sein, Softwareentwicklung ganz abzugeben? Könnte z.B. ein Unternehmen, dessen Kernkompetenz im mechanischen Bereich liegt, Software nicht komplett extern einkaufen?

EH: Auch die mechanischen Produkte funktionieren heute nicht mehr ohne Software und die Fertigungsprozesse werden immer stärker digitalisiert. Kunden wollen Produkte, die Probleme lösen und funktionieren.

In diesem Zusammenhang spielt Software eine immer größere Rolle. In Branchen, in denen es auf der Hardwareseite wenig Innovation gibt, kann sogar das Umschwenken auf reine Software eine Option sein. Die Software und die Frage, wie man ihre Entwicklung managt, werden zunehmend zum Domänenwissen, zum Differenzierungsmerkmal und zum Wettbewerbsvorteil. Darum ist es wichtig, bei der Entwicklung die Zügel in der Hand zu behalten. Wenn man die Architektur skalierbar und modular aufgebaut hat, kann man problemlos Module extern vergeben oder zukaufen. Aber die Kunden und die Markterkenntnis, die Softwarearchitektur, die Beschreibung der Funktionen und die Spezifikationen, die muss das Unternehmen selbst im Griff haben.