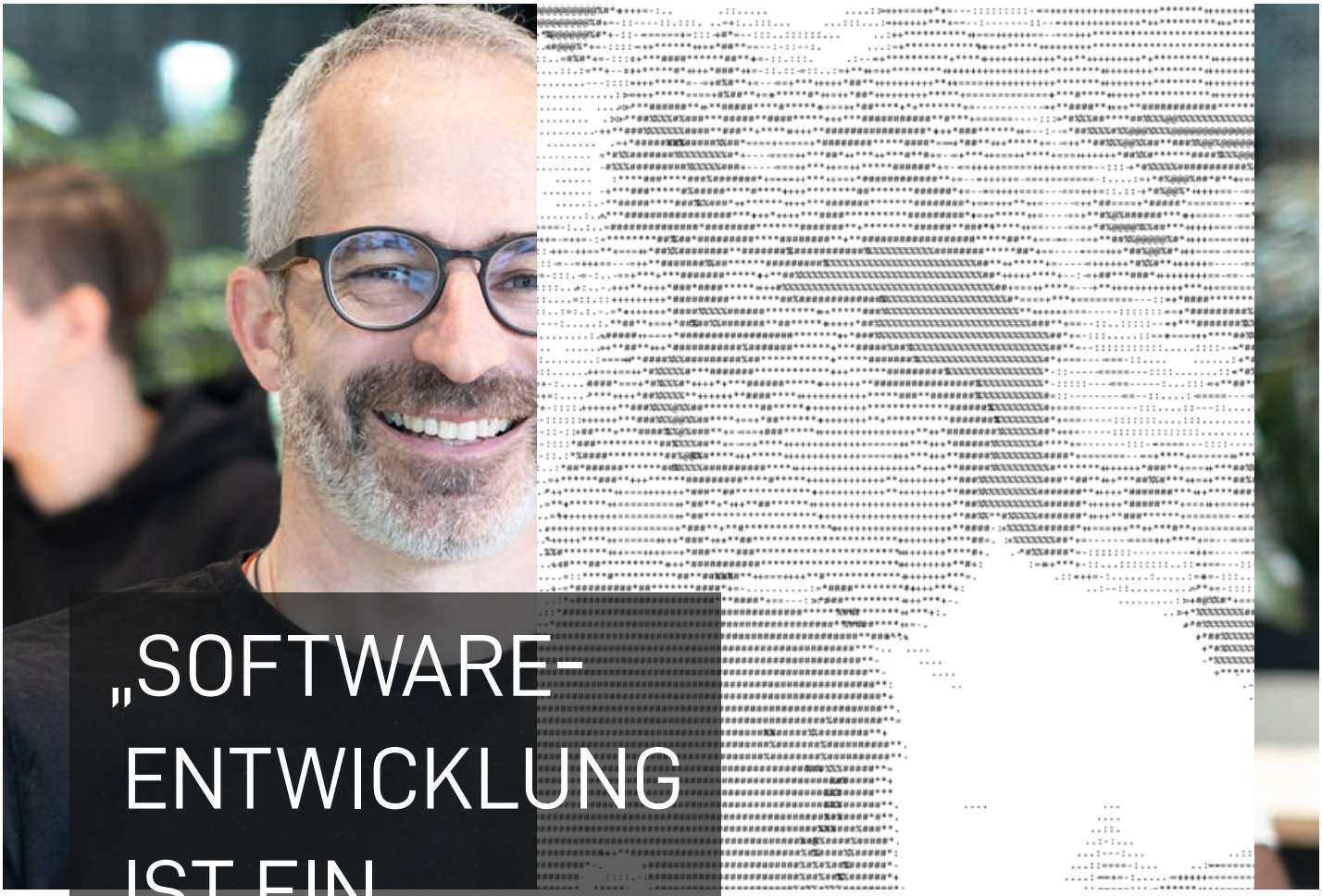

„SOFTWARE-ENTWICKLUNG IST EIN KREATIVER PROZESS."

Ein Gespräch mit Vincent Stüger, Office of the CTO, Dynatrace.

DIALOG: Herr Stüger, welchen Impact hat Software auf Menschen und Organisation?

VS: Je mehr das Produkt Software wird, desto mehr löst es sich von physischen Dingen, die gewisse Prozesse und Arbeitsweisen vorgeben. Und auch die Geschwindigkeit wird eine völlig andere. Physische Produkte haben bestimmte Fertigungszyklen, die Vorlaufzeiten erfordern. Einen Fabrikbau kann man immer noch nicht viel schneller hinstellen als vor ein paar Jahren. In der Softwareentwicklung ist das anders, wobei Plattformen, Technologien und Modulbaukästen den Prozess immer weiter beschleunigen. Der Umsetzungszyklus von einer Idee zu einem Minimum Viable Product wird immer kürzer, bei Services ist die Entwicklung nochmal schneller. Diese Dynamik hat auf die Menschen und die Organisation sehr großen Einfluss.

DIALOG: Eine hohe Geschwindigkeit hat mehrere Implikationen – Passagiere sind ihr ein Stück weit ausgeliefert, Fahrer müssen ab einer gewissen Geschwindigkeit überproportional viel Energie aufwenden

VS: Die entscheidende Frage ist immer: „Was ist wichtig?" Dafür braucht es einen guten Kommunikationsfluss. Bei hoher Geschwindigkeit brauche ich eine Vision und die daraus abgeleitete Strategie, die beim Mitarbeiter ankommt. Jeder muss klar wissen, wo die Reise hingeht, warum sie dorthin geht und was die Rahmenbedingungen sind. Bei hoher Geschwindigkeit muss man dafür sorgen, dass das jeder versteht. Darum sind Softwareorganisationen sehr flach, oft mit nur zwei Hierarchieebenen im klassischen Sinn. Das erfordert neue Kommunikationsformen, Klarheit, eine klare Verbindung zwischen Strategie und Vision, effektives Alignment, ohne das Zielsystem zu eng zu machen.

Dann bekommt man die zum Problemlösen nötige Kreativität. Aber wenn man das Zielsystem zu weit macht, endet man im Chaos. Dafür sind diese „schnellen" Organisationsformen sehr anfällig.

DIALOG: Warum entsteht dieses Chaos?

VS: Alignment passiert ja selten am Frühstückstisch, wie bei einem Start-up. In einer großen Organisation mit Hunderten Entwicklerteams, die an einem Produkt arbeiten, muss ich anders dafür sorgen. Flache Hierarchien generieren keine Struktur, die das Alignment sichert. Ich muss sicherstellen, dass die Leute abgestimmt arbeiten und nicht einer links, weil er nicht weiß, dass der rechts

das schon baut, das noch einmal baut und dann auch noch anders. Es geht darum sicherzustellen, dass die Leute ausgerichtet an einer gemeinsamen Vision arbeiten, die sich auch noch immer wieder ändern kann. In diesem Umfeld ist ein Strategieprozess eigentlich gar kein Prozess mehr, sondern passiert laufend und verändert sich immer wieder mit neuen Kunden und Technologien. Das muss man absichern, dabei aber auch Autonomie erlauben, die für die Kreativität unabdingbar ist. Dabei stehen Autonomie und Steuerung prinzipiell in einem Widerspruch, den ich durch intensive und inklusive Kommunikation auflösen muss.

DIALOG: Wie groß ist die Gefahr bei einer Softwareentwicklung, dass man aufgrund dieser hohen Geschwindigkeit immer wieder zurück muss? Man dreht ein riesiges Rad, hunderttausend Leute, alle schnell, agil – und merkt dann aber nach einer recht langen Zeit, irgendwo ist man falsch abgebogen oder an der Station irgendwie komplett vorbeigefahren.

VS: Da ist jetzt die Frage, ob es die Geschwindigkeit ist oder die notwendige Arbeitsweise. Agile Setups funktionieren bis zu einem gewissen Grad über einen Selbststeuerungsmechanismus, der stark vom Feedback, vor allem Kundenfeedback, lebt. Nur gibt es ab einer gewissen Komplexitätsebene den Kunden, der alles sieht, auch nicht mehr. Also müssen mehrere Feedbackzyklen synchronisiert werden, das ist die eine Herausforderung. Die andere liegt in der Superkomplexität solcher Systeme. Es geht ja nicht mehr darum, dass Hundert Kunden zehn Funktionen einer App testen – nach dem Motto „geht, geht nicht". Das kann man organisieren.

Aber ein Produkt, das global verteilt läuft, ist in Gänze irgendwann nicht mehr beurteilbar. Hochverteilte Systeme sprengen die menschliche Denkmöglichkeit. Das kann man nur noch mit Werkzeugen betrachten. Und darum ist die Komplexität der Systeme verbunden mit der Geschwindigkeit, in der sie sich ständig ändern, und der Virtualität, der Nichtgreifbarkeit, eine Mischung, die uns an Grenzen bringt. Das birgt Risiken und steht oft an der Kante zum Chaos.

DIALOG: Braucht die Softwareentwicklung Menschen, die das alles aushalten können, die auch so schnell surfen können? Es kann ja nicht nur die Frage der fachlichen Ausbildung sein.

VS: In einer Softwareorganisation sind Fünfjahrespläne, in Stein gemeißelte Visionen und langfristig stabile Teamstrukturen vollkommen unmöglich. Man braucht deshalb vor allem Menschen, die sehr gut kommunizieren können, was gerade im technischen Umfeld weder einfach noch selbstverständlich ist. Die in einem dynamischen Setup Teams formen, verändern und führen können und dafür sorgen, dass eine Strategie, die ja in sich ebenfalls dynamisch ist, richtunggebend bleibt.

DIALOG: Wie kann man angesichts dieser Dynamik und Komplexität die Effizienz in der Softwareentwicklung beurteilen und verbessern?

VS: Ich glaube, dass Effizienz in der Softwareentwicklung anders zu betrachten ist als in der klassischen Produktion. Allein schon, weil bei Software Fertigung und Entwicklung quasi immer miteinander verzahnt sind. Das Produkt entsteht, während man es denkt, es ist hundertprozentig „mind-made", der Prozess ist in weiten Teilen nicht automatisierbar. Das heißt, wenn Menschgen motiviert sind, einen guten Informationsfluss und eine gute Ausrichtung haben, trägt das sehr stark zur Effizienz bei. Messen kann ich das vor allem retrospektiv, etwa anhand der Frage, wie schnell man Themen bestimmter Komplexität durchkriegt. Dann kann ich den Output von Teams bewerten.

Die Problematik entsteht häufig durch Redundanzen, weil Teams parallel an den gleichen Themen arbeiten und Dinge immer wieder neu erfunden werden, z.B. weil eine transparente Plattform fehlt, ein Baukasten, auf den alle zugreifen, und weil die Kommunikation nicht stimmt. Das ist der eine Fall. Der andere, deutlich häufigere Fall ist: Man baut einfach etwas Falsches, was durch die hohe Geschwindigkeit noch verstärkt wird.

DIALOG: Dann lassen sich in der Softwareentwicklung Effizienz und Effektivität gar nicht scharf trennen?

VS: Die beiden Themen überlagern sich. Softwareentwicklung ist ein stark kreativer Prozess. Und die Effizienz kreativer Prozesse lässt sich grundsätzlich nicht nur schlecht messen – auch der Wert solcher Messungen ist nicht immer eindeutig. Weil

man kreative Prozesse nicht beliebig beschleunigen und um jeden Holzweg und jede Redundanz bereinigen kann. Letztlich dominiert die Frage, ob man das Richtige macht. Deswegen sind ja die Klärung der Zielrichtung, die Kommunikation, das Alignment so wichtig.

Man kann und muss natürlich vermeiden, dass man auf einen völlig falschen Weg gerät oder einen kompletten Overlap bekommt. Aber im Detail muss man akzeptieren, dass auch mal zwei Leute die gleiche Idee haben und sie auch umsetzen. Und vor allem muss man die Schwierigkeit meistern, eine oft sehr abstrakte und vage Anforderung so gut zu formulieren, dass die Richtung für die Entwickler klar wird und ein technisches Bild entsteht. Dafür ist ein sehr enger, sehr tiefer Austausch im Team, vor allem aber mit dem Kunden, absolut kritisch. Je besser man die Anforderung versteht, desto besser läuft der Prozess. Was tut der Kunde mit dem Produkt? Welchen Anwendungsfall hat er? Was ist seine Problemstellung? Wie fühlt er sich dabei? Es ist letztlich Storytelling. Und je besser die erzählerische Qualität des Inputs ist, desto höher sind Effizienz und Effektivität in der Softwareentwicklung.

DIALOG: Sie hatten die Herausforderung erwähnt, dynamische und verteilte Softwareprodukte überhaupt als Ganzes betrachten zu können. Wie hoch ist vor diesem Hintergrund die Chance auf einen so guten Input, dass effektiv und effizient entwickelt werden kann?

VS: Das ist nun mal die Komplexität, mit der man umgehen muss. Es geht darum, die große Story, die Eckpunkte, gut zu verstehen und die Menschen zusammenzubringen, die man vermutlich für das Projekt braucht. Das Zielprodukt wird sich im Prozess konkretisieren, ein wenig wie bei einem Buch, an dem mehrere Autoren gemeinsam schreiben. Dabei muss man die Qualität und Effizienz des Prozesses unterstützen, durch Kommunikation, Qualitätssicherung einzelner Module, gezielte laufende Feedbacks. Das schaltet nicht das Risiko aus, dass das Gesamtprodukt nicht den Erwartungen entspricht. Aber dieses Risiko gehört letztlich untrennbar zur Entwicklung komplexer Software.

DIALOG: Die von Ihnen skizzierten Eigenschaften der Softwareentwicklung und der Softwareorganisation beschreiben eine sehr besondere Landschaft. Wie groß sind die Herausforderungen, diese Welt in die klassische Produktionswelt zu integrieren?

VS: Um komplexe Industrieprodukte zu realisieren, die Hardware- und Softwareelemente beinhalten, müssen Entwicklerteams disziplin- und länderübergreifend an einer gemeinsamen Vision arbeiten. Hier kommen Menschen aus diametral entgegengesetzten Welten zusammen: Dabei tun sich Softwareentwickler schwer mit Prozessen, wo es keine Spielräume gibt, die normiert, hochsicherheitsrelevant und massenfertigungstauglich sind, die man nicht agil managen kann. Klassische Ingenieure haben wiederum oft keinen Zugang zur Offenheit und Kreativität, die in der Softwareentwicklung unabdingbar sind, sowie zu dem Phänomen, dass Entwicklung und Produktion eins sind und dass man sehr kurzzyklisch immer wieder neue Elemente aus bestehenden Modulen bauen und in Plattformen integrieren und laufend verbessern kann, während sie schon im Einsatz sind. Zwischen diesen Welten muss man Brücken bauen können. Teilweise muss man die Softwareorganisation vom klassischen Betrieb auch entkoppeln und Schnittstellen definieren, wenn die Unterschiede zu groß sind. Wie schwierig diese Aufgabe ist, sehen wir täglich.

DIALOG: Diese Brücken zu bauen wird sicher nicht dadurch einfacher, dass die Welt zunehmend „software-defined" ist und somit die organisatorische Position der Softwareentwicklung immer stärker wird, oder?

VS: „Software-defined" bedeutet nicht, dass es nur noch um Software geht – auch in Zukunft nicht. Aber man muss dennoch mit der Tatsache umgehen können, dass die mechanische Optimierung und der daraus resultierende Mehrwert in sehr vielen Bereichen an ihre Grenzen kommen. Wenn man bei einer Maschine bei 99,9% physikalisch möglicher Präzision ist, geht es nicht mehr darum, das noch zu verbessern. Dann wird die Möglichkeit, diese Maschine mit anderen zu vernetzen, oder die darauf laufenden Programme innerhalb von Sekunden zu ändern, differenzierend und innovationstreibend. Dann entstehen neue Nutzungsszenarien, neue Märkte. Diesen Paradigmenwechsel kann man nicht ignorieren.