6 Tipps für CIOs, die sich das Leben einfacher machen wollen

von Esko Hannula, CEO Qentinel

1. Hören Sie auf, Ihre Systeme zu kontrollieren – und beginnen Sie, **diese zu steuern**
2. Investieren Sie **in Produkte,** nicht Projekte – so machen Sie Ihre **Organisation effektiv**
3. Setzen Sie auf **DevOps –** und lassen Sie sich **nicht entmutigen**
4. Testen Sie – immer dann, wenn es Ihnen **am meisten bringt**
5. **Automatisieren** Sie so viel wie möglich – **aber keinen schlechten Prozess**
6. Messen Sie - **aber mit Köpfchen**

© kras99

Competence Book #NextDigital

1 Hören Sie auf, Ihre Systeme zu kontrollieren – und beginnen Sie, diese zu steuern

Jede anständige Führungskraft will kritische Situationen unter Kontrolle haben. Aber unsere Informationssysteme und insbesondere ihre Interdependenzen sind längst über die gängigen Management-Methoden hinausgewachsen. Die Suche nach vollständiger Kontrolle ist daher eine Illusion geworden. Setzen Sie stattdessen lieber auf Steuerung. Das Prinzip der Steuerung ist einfach, aber nicht immer einfach auszuführen:

- Definieren Sie Ihre kritischen Geschäftsprozesse.
- Identifizieren Sie Ihre Kernsysteme.
- Erkennen Sie deren kritische Abhängigkeiten.
- Frieren Sie ein, was Sie können.
- Fokussieren Sie das Testen auf kritische Integrationen.

Da Sie noch im Geschäft sind, wissen Sie sicher, was Ihre kritischen Geschäftsprozesse sind und wie Ihre Informationssysteme dazu beitragen. Sie haben wahrscheinlich eine Vielzahl von Anwendungen und Systemen, aber nur eine Handvoll davon sind Kernsysteme. Sie sind durch eines oder mehrere der folgenden Merkmale gekennzeichnet:

1. Sie sind betriebsnotwendig für einen kritischen oder zentralen Prozess. Fertigungssysteme (MES) erfüllen typischerweise dieses Kriterium.

2. Sie enthalten große Mengen an kritischen Geschäftsdaten. ERP- und CRM-Systeme erfüllen typischerweise dieses Kriterium.

3. Sie werden von einer großen Anzahl an Menschen aktiv genutzt. Bürosysteme, Auftragsbearbeitungssysteme, Helpdesk-Systeme und Selbstbedienungsportale können dieses Kriterium erfüllen.

4. Sie dienen als Integrationszentrale in der Unternehmensarchitektur.

Das vierte Kriterium ist besonders interessant. Einige Unternehmen haben alles rund um SAP aufgebaut. Einige haben vielleicht einen „versteckten Integrationsknotenpunkt“ in einem sehr alten Informationssystem, auf dessen Datenbank sich alle anderen verlassen. Andere haben möglicherweise einen eigenen Integrationsbus, aus dem sich das komplette System entwickelt.

Da Software immateriell ist, kann eine faktenbasierte analytische Diskussion darüber schwierig sein. Die Grenzen zwischen Kernsystemen sind manchmal schwer zu definieren. Zum Beispiel neigt man dazu, die Benutzeroberfläche zum Namensgeber für alles zu machen, was über sie abgerufen wird – auch wenn diese Oberfläche nur eine dünne Hülle um riesige Informationssysteme herum ist.

Probleme mit Informationssystemen werden häufiger durch Abhängigkeiten und Integrationen zwischen Systemen verursacht, als durch ein einzelnes System allein. Da sich das Unternehmen in Lichtgeschwindigkeit bewegt, muss sich das digitale Rückgrat ebenso schnell entwickeln. Daher gibt es eine Reihe von nicht vorhersehbaren System-Updates, die ständig im Gange sind. Das Problem ist, dass diejenigen, die ein System aktualisieren, sich selten um die anderen Systeme kümmern, mit denen es integriert ist.

Sobald Sie die kritischen Abhängigkeiten Ihrer Kernsysteme kennen, können Sie eine strengere Kontrolle und Qualitätssicherung ausüben und allen anderen Systemen mehr Entwicklungsfreiheit lassen. In der Praxis bedeutet dies weniger Änderungen in Kernsystemen und kritischen Abhängigkeiten, eine gründlichere Analyse der Auswirkungen dieser Änderungen und eine gründlichere Qualitätssicherung, bevor die Änderungen in Kraft treten. Mit Sicherheit wird es immer noch Probleme und wütende Anwender geben, aber zumindest Ihre kritischen Geschäftsabläufe werden weniger anfällig sein.

2 Investieren Sie in Produkte, nicht Projekte – so machen Sie Ihre Organisation effektiv

„Project to Product“ ist in letzter Zeit zu einem modischen Slogan geworden – und das aus gutem Grund. In der Vergangenheit war ein Informationssystem eine Projektinvestition. Man hat ein Jahr mit der Definition verbracht, ein paar Jahre mit der Implementierung und ein paar Jahre mit der Nutzung. Dann wurde ein neues Projekt gestartet, um das System zu ersetzen.

In Wirklichkeit hat das Informationssystem während seiner aktiven Lebensdauer eine ganze Menge „Wartung“, „kleine Entwicklungen“ und „Upgrades“ durchlaufen. In vielen Fällen erforderte die „Wartung“ die gleiche Anzahl an Personen wie das ursprüngliche Inplementierungsprojekt.

Ein Projekt ist eine temporäre Organisation mit einem klaren und festen Umfang, Zielen, Budget, Ressourcen und Dauer. Eine Produktorganisation ist vergleichsweise dauerhaft oder zumindest langlebig. Ihre Mission ist es, nachfolgende Versionen des Produktes zu liefern und die Wettbewerbsfähigkeit mit jeder Veröffentlichung zu verbessern.

Der Lebenszyklus eines Informationssystems ähnelt heute eher einem Produkt- als einem Projektlebenszyklus. „Von Projekten zu Produkten“ bedeutet einfach, aus einer Notwendigkeit eine Tugend zu machen.

Beginnen Sie mit der Einrichtung einer Produktmanagement-Funktion. Ein Produktmanager führt das Informationssystem als inkrementelle Investition, bei der jedes Update den Wert des Systems steigert. Er oder sie wird auch ständig über die Anforderungen des Unternehmens informiert sein, sie priorisieren und einen konstanten Fluss von Systemreleases planen, die wertschöpfende neue Funktionen im Produkt liefern. Mit anderen Worten: Der Produktmanager verwaltet den Wert des Informationssystems nicht als einmalige Investition, sondern als ein Kontinuum. Gleiches gilt für das „Produktentwicklungsteam“. Die an der Entwicklung des Systems aktiv beteiligten Personen werden nach der Phase „Projekt ist abgeschlossen“ bleiben, um das System wettbewerbsfähig zu halten. Natürlich kann die Größe des Teams im Laufe der Zeit variieren.

Der Wechsel von Projekten zu Produkten kann nach einem enormen Kostenanstieg klingen, lassen Sie sich daher nicht in die Falle locken, die Produktorganisation neben der alten Projektorganisation aufzubauen. Möglicherweise müssen Sie Rollen und Verantwortlichkeiten neu ordnen - und einige Personen austauschen. Grundsätzlich brauchen Sie jemanden, der das Projekt definiert, die Software liefert und betreibt.

Ein Projektmanager liefert nach einem Plan und Budget und macht dann weiter. Ein Betriebsleiter hält die Dinge am Laufen und minimiert Benutzerbeschwerden. Ein Produktmanager hält das Produkt wettbewerbsfähig, indem er seinen Business Case verwaltet.

Und was hat „Project to Product“ mit dem Testen zu tun? Stellen Sie sich vor, wie unterschiedlich der Projekt- und der Produktverantwortliche über Qualität denken und wie unterschiedlich sie die Aktivitäten priorisieren würden. Wir werden in den nächsten Punkten in diese Unterschiede einsteigen.

3 Setzen Sie auf DevOps – und lassen Sie sich nicht entmutigen

Falls Sie *DevOps* noch nicht in Ihre Anwendungsentwicklung integriert haben, sollten Sie jetzt damit anfangen. Einfach ausgedrückt: *DevOps* ist für die Softwarebereitstellung das, was Lean für die Produktionsprozesse ist. *DevOps* optimiert das Verhältnis von Zeit zu Wert – natürlich mit hoher Qualität. Softwareentwickler lieben *DevOps*, weil es als cool angesehen wird. Leider behandelt die Mehrheit der Entwickler *DevOps* so, wie sie zunächst *agile* behandelt haben: indem sie nur die lustigen Teile übernehmen.

Ihre Lieferanten werden Releases häufiger als je zuvor liefern. Sie werden nicht mehr entscheiden können, wann Sie ein Upgrade durchführen möchten. Sie werden gezwungen sein, dem Rhythmus Ihrer Lieferanten zu folgen. Ohne *DevOps* wird das schmerzhaft, oder gar unmöglich.

Machen Sie sich zunächst mit dem Thema vertraut, indem Sie das *DevOps* Handbuch oder das Phoenix Projekt lesen. Beide Bücher sind leicht zu lesen, nicht zu technisch und vermitteln Ihnen, worum es bei *DevOps* wirklich geht. Nach dem Lesen dieser Bücher können Sie Ihre eigenen Urteile fällen, ohne von Ihren Technikern manipuliert zu werden.

Beginnen Sie mit einem kleinen, aber motivierten Team, vorzugsweise kompetent in agilen Methoden. Wählen Sie eine unkritische Zuordnung aus. Erwägen Sie, einen externen *DevOps*-Coach zu engagieren. Ihr Team wird wahrscheinlich viel Zeit damit verbringen, die richtigen Arbeitsweisen herauszufinden, Werkzeuge zu implementieren, untereinander zu kämpfen und unter dem Strich wenig liefern. Eine neue Denkweise zu erlernen kann zunächst schaden. Auch neue Werkzeuge können Schmerzen verursachen. Es gibt eine Fülle von netten Open-Source-Tools, die für *DevOps*-Teams verfügbar sind, aber sie alle zum ersten Mal zusammenzustellen, ist eine Qual.

Beachten Sie, dass Entwickler *DevOps* hin und wieder missverstehen. Die Kernidee besteht darin, dass Entwicklung, Betrieb, Qualitätssicherung und Sicherheit in einem agilen Tempo zusammenarbeiten und durch Software Mehrwert schaffen. Während eigentlich jeder die Idee begrüßt, finden es Entwickler in der Regel praktisch, die anderen drei aus dem Prozess auszuschließen, da es ihr Leben einfacher macht. Betrieb, Qualitätssicherung und Sicherheit sind oft froh, ausgeschlossen zu werden, weil sie wahrscheinlich noch weniger mit dem *DevOps*-Verfahren vertraut sind als die Entwickler selbst.

Um *DevOps* richtig zu machen, sollten Sie Folgendes tun:

- Priorisieren Sie strikt nach dem Business Value.
- Erkennen Sie, dass die Bereitstellung von Business Value auch kontinuierliche Investitionen in die Softwarearchitektur und den Softwareprozess erfordert.
- Ihre Teams für Development, Qualitätssicherung, Security und Operations sind nach den gleichen Prioritäten organisiert und haben das gleiche Arbeitstempo.
- Automatisieren Sie Prozesse, wo immer Sie können.
- Messen Sie alles und nutzen Sie die Messungen, um Entscheidungen über Ihr Unternehmen zu treffen.

Viele betrachten die Einführung von *DevOps* als kulturellen Wandel und das aus gutem Grund. Arbeitskultur ist keine Zauberei: Es geht um Gewohnheiten und Überzeugungen, und sie verändern sich durch Arbeitsstrukturen, Prozesse, Praktiken und Werkzeuge.

Damit *DevOps* in der Praxis funktioniert, stellen Sie sich vor, Sie nehmen einen Haufen freiheitsliebender Handwerker (Entwicklung), ein Regiment aus einer disziplinierten Armee (Betrieb und QS), einen ordnungsversessenen Regierungsbeamten (Sicherheit) und lassen alle in einer schlanken Toyota-Fabrik zusammenarbeiten. Was passiert wohl? Um *DevOps* richtig anzuwenden, sollten Sie bald anfangen, und kleine Schritte machen, um Ihr Ziel zu erreichen.

4 Testen Sie – immer dann, wenn es Ihnen am meisten bringt

In der Entwicklung von Informationssystemen sind Softwaretests ein Problemfall. Das Testen gilt auch als der größte Engpass im Softwarefreigabeprozess.

In traditionellen IT-Projekten starteten die ersten ernsthaften Tests erst kurz vor dem geplanten Abschluss des Projekts. Schließlich war es nicht einfach, das System zu testen, bevor alles zusammengesetzt wurde. Die Tester nennen das den „Big Bang“. Mit dem „Big Bang“-Ansatz kann eine ursprünglich für ein paar Wochen geplante Testaufgabe leicht auf 10 Monate verlängert werden. Dieses Vorgehen ist teuer, unberechenbar und immer noch gefährlich beliebt.

Probleme sind billig und schnell zu beheben, wenn sie frühzeitig erkannt werden. Die Problemursache kann irgendwo tief im Systemdesign liegen. Das Auffinden, Korrigieren und Verifizieren der Korrektur kann sehr aufwendig sein und noch mehr Zeit in Anspruch nehmen. Schlimmer noch, solche Korrekturen werden wahrscheinlich eine Reihe neuer Probleme mit sich bringen.

Ein Programmierer schreibt einen fehlerhaften Code, auf dessen Grundlage fünf andere Programmierer einen weiterführenden Code schreiben. Wird der erste Fehler korrigiert, werden sich die fünf neuen Codeteile wahrscheinlich fehlerhaft verhalten. Passiert das ein paar hundert Mal, ergibt sich ein riesiger und oft panischer Test-Debug-Korrektur-Retest-Kreislauf, der neue Fehler einführt und alte korrigiert. So funktioniert der „Big Bang“.

Die Lösung besteht darin, das Testen nach vorne zu verlagern, also in frühen Phasen der Systementwicklung und näher am Code zu testen. Die Fehler werden dann erkannt, wenn sie noch nicht den Rest des Systems betroffen haben. Sie sind dann schneller zu reparieren und zu überprüfen. Verteilt man den Testaufwand über einen längeren Zeitraum, wird die Gesamtzahl der für den Prozess benötigten Personen geringer.

Agile Entwicklungsmethoden, moderne Tools und billige Rechenleistung haben alte Ausreden beseitigt, um nicht frühzeitig zu testen. Wenn Sie wissen wollen, wie professionell Ihre agilen Teams sind, dann finden Sie heraus, wie sie testen… oder ob sie überhaupt testen.

Das Internet, die Cloud und agile Methoden haben eine ganz neue Art Software zu entwickeln und zu releasen hervorgebracht. Informationssysteme sind keine unabhängigen, sich langsam verändernden Monolithen mehr. Sie werden aus scheinbar eigenständigen Elementen zusammengesetzt und permanent released.

Wahrscheinlich ist Ihr ERP mit dem CRM verbunden. Das ERP kann sich im Netzwerk befinden, aber das CRM kann in der Cloud liegen. Das ERP-System bedient vermutlich mehrere, unabhängig voneinander entwickelte Frontend-Anwendungen und tauscht häufig Informationen mit den Systemen Ihrer Lieferanten und Wiederverkäufer aus. Wann immer sich eines dieser Systeme ändert, können Ihre kritischen Geschäftsprozesse betroffen sein.

In manchen Fällen ist es notwendig, die Tests nach hinten zu verschieben. Das bedeutet, man führt Tests in der Produktionsumgebung durch. Um sicherzustellen, dass alles noch funktioniert, ist der schnellste und kostengünstigste Weg, die Weiterführung von Tests in der Produktion und die Überwachung des Verhaltens der Systeme in Bezug auf die Geschäftsprozesse, für die sie bestimmt sind.

Das Verschieben von Tests nach vorne und nach hinten verbessert die Geschwindigkeit, Qualität und Produktivität der Softwareentwicklung und steigert das Vertrauen in die Leistung von Geschäftsprozessen, die auf verschiedenen integrierten Informationssystemen beruhen.

5 Automatisieren Sie so viel wie möglich – aber keinen schlechten Prozess

Es ist praktisch unmöglich, einen schnellen Software-Release-Zyklus ohne automatisiertes Testen zu betreiben. Manuelle Tests sind zu langsam, um mit dem Tempo agiler Entwicklungsteams Schritt zu halten. Spätestens, während des Systemtests und der Validierung von Integrationen mit anderen Systemen, werden sowohl der Zeitaufwand als auch die Gesamtauslastung untragbar.

Testautomatisierung soll teuer in der Durchführung und noch teurer in der Wartung sein. Daher sollte die Automatisierung zunächst bei Tests angewandt werden, die häufig wiederholt und selten geändert werden. Wenn die Testautomatisierung gut funktioniert, kann man einiges an menschlichem Aufwand sparen. Größere Gewinne ergeben sich jedoch aus der Zeitersparnis. Im besten Fall werden Ihre Tests bis zu 90% beschleunigt, indem Sie Wartezeiten im Prozess eliminieren.

Einfaches Beispiel: Ein Software-Team veröffentlicht eine neue Systemversion zum Testen. Es ist Donnerstag. Die Tester sind noch damit beschäftigt, einige Korrekturen aus dem vorherigen Testrelease zu überprüfen, aber sie können am Montag damit beginnen. Bis jetzt sind drei Tage verloren. Die Tester werden die Testrunde in drei Tagen, d. h. nächsten Mittwoch, abschließen. Dann kann das Entwicklungsteam mit der Korrektur der in den Tests gefundenen Fehler beginnen. Einige Korrekturen sind einfach. Einige erfordern so viel Arbeit, dass sie erst in drei Wochen getestet werden können. Nach dem Testzyklus beginnt es wieder von vorne. Jetzt werden die Tester neue Funktionen sowie Korrekturen von Fehlern testen, die sie vor drei Wochen gefunden haben. Und so weiter.

Ein Testfall, der so mehr als sechs Kalendertage in Anspruch nimmt, kann durch automatisierte Tests in wenigen Stunden abgeschlossen werden. Die Korrekturen im Code können durch erneutes Ausführen der Tests überprüft werden. Die Zeitersparnis im Feedbackzyklus ist enorm.

Aber Testautomatisierung ist kein Kinderspiel. Sie können zwar die Anzahl der Tester reduzieren, aber Sie benötigen Mitarbeiter, die Tests manuell ausführen und automatisierte Tests entwerfen und pflegen. Ein kreativer Mensch ist einer Maschine noch immer überlegen, wenn es darum geht, neue Funktionen zu testen und herauszufinden, was schief gehen könnte. Die Maschine ist überlegen, wenn es darum geht, die gleiche Routine wiederholt durchzuführen. In einer Welt der schnellen Zyklen und mehrfachen Integrationen „wiederholt“ sich immer mehr immer öfter.

Testautomatisierung wird in der Regel als die Automatisierung der Durchführung von Tests wahrgenommen. Das ist ein etwas enger Blick. Im Großen und Ganzen bedeutet Testautomatisierung:

1. Automatisierte Durchführung der Tests
2. Automatisierte Aufbereitung von Testdaten
3. Automatisierte Erstellung von Testfällen
4. Automatisierung der Einrichtung und Konfiguration von Testtools und -umgebungen
5. Automatisierung der Installation und Konfiguration des zu testenden Systems
6. Automatisierung der Berichterstattung über die Testergebnisse
7. Automatisierte Analyse der Testergebnisse und
8. Automatisierung des Reportings von Teststatistiken und Qualitätskennzahlen.

Die Testautomatisierung wird bei richtiger Anwendung enorme Vorteile bringen. Aber auch hier beginnt alles mit dem richtigen Verständnis der Qualitätsrisiken, der Abhängigkeiten des Softwareprozesses. Die Automatisierung eines schlechten Prozesses führt immer noch zu einer automatischen Mittelmäßigkeit.

6 Messen Sie - aber mit Köpfchen

Ein Softwareprozess ist in der Regel schwieriger zu verwalten und zu optimieren als ein industrieller Fertigungsprozess. Dies hat wenig mit der „intellektuellen Herausforderung“ oder der „inhärenten Komplexität“ von Software zu tun. Die Geschichte der industriellen Fertigung ist viel länger. Die zugrunde liegenden Praktiken, Prozesse und Werkzeuge sind viel standardisierter und es gibt weniger Unterschiede zu verwalten als bei Software.

Nur ein Unternehmen, welches so gut ist wie die antreibende Software, kann wettbewerbsfähig bleiben. Indem es seinen Softwareprozess kontinuierlich verbessert und optimiert. Das hat nichts mit der Zerstörung von Kreativität und Innovationskraft zu tun. Je besser, schneller und automatisierter Ihr Prozess ist, desto mehr können Sie sich Innovationen leisten.

Ein guter Softwareprozess erfordert eine rigorose Messung. Wir müssen erkennen, ob sich die Dinge zum Guten oder Schlechten entwickeln und uns anpassen, wenn Verbesserungen oder Korrekturen erforderlich sind. Glücklicherweise ist der Softwareprozess einfach zu messen. Fast alles – vom Code bis zum Helpdesk – umfasst Tools, die eine Fülle von Daten produzieren, die zum Verständnis, zur Kontrolle, Verbesserung und Vorhersage der Qualität der Software sowie des Prozesses, der sie erstellt und veröffentlicht, verwendet werden können.

Menschen in verschiedenen Rollen benötigen unterschiedliche Ansichten derselben Informationen. Den Softwareprozess durchzuführen, indem man sich ausschließlich auf Fehlerberichte konzentriert, ist vergleichbar damit, ein Unternehmen zu führen, bei dem man nur Forderungen und Verbindlichkeiten betrachtet. Einen Betrieb durch das Lesen von Jahresabschlüssen zu führen und zu entwickeln, ist zum Scheitern verurteilt. Doch die Analyse der Trends wichtiger Finanzindikatoren, die sich aus den Jahresabschlüssen ergeben, ist äußerst hilfreich.

Der Zweck von Softwaremetriken ist es, die bestmöglichen Entscheidungen zum richtigen Zeitpunkt zu treffen. Die aktuelle Literatur bietet Hunderte von praktischen Kennzahlen zur Softwarequalität. Die Kennzahlen, die mir im Software-Qualitätsmanagement am besten geholfen haben, sind folgende:

- Pass/Fail-Statistik.
- Fehlerakkumulation
- Prozentsatz der Fehlererkennung
- Zeit-/Phasenverteilung von Fehlern
- Architektonische Verteilung der Fehler
- Akkumulation der Nacharbeit
- Wartezeitakkumulation
- Codeänderungen pro Tag/Woche

Aber das ist nur die Spitze des Eisbergs. Eine umfassende Beschreibung, wie Sie Metriken zu Ihrem Vorteil einsetzen können, finden Sie hier. LINK

Ein kleiner Warnhinweis: Die meisten Menschen hassen es, gemessen zu werden. Sie glauben nicht, dass das Management gute Absichten hat, wenn es um Kennzahlen geht. Das bedeutet, dass jeder verlangte Messaufwand, Dinge manuell zu melden, zum Scheitern verurteilt ist. Automatisieren Sie daher Ihre Kennzahlen. Die Einstellung zur Messung wird sich im Laufe der Zeit ändern, wenn Menschen lernen, Metriken zuerst zu verstehen und erst danach zu kontrollieren.