Docker und Virtuelle Maschine im Vergleich – ein Performanz-Benchmark

Tim Stoffel, :em engineering methods AG

Wir leben in einer Welt in der die Entwicklungszyklen immer kürzer werden und sich die Anforderungen an unsere Systeme und den Betrieb dieser ständig ändern. Um in diesem wachsenden Wettbewerb bestehen zu können, ist es notwendig eine Software schnell auszuliefern und ihre Infrastruktur so flexibel wie möglich zu betreiben. Dies kann kosteneffizient nur mit einem hohen Automatisierungsgrad erreicht werden. Gerade für die Automatisierung werden Container immer wieder als Werkzeug empfohlen. Doch was sind diese Container, welchen Mehrwert bringen sie und was unterscheidet sie von den bekannten Virtuellen Maschinen? Diese Fragen beantwortet der erste Teil dieses Artikels.

Im zweiten Teil wird untersucht, wie sich die Leistung einer Applikation in unterschiedlichen Betriebsumgebungen verhält. Verglichen wird der Betrieb direkt auf der Hardware, in einer Virtuellen Maschine, in einem Container und in einem Container der in einer Virtuellen Maschine läuft. Damit wird die Frage beantwortet ob es für die Performanz kritisch ist, die Containerinfrastruktur auf Virtuellen Maschinen aufzubauen.

Warum Container der nächste Schritt sind

Die moderne Informationstechnik hat die Industrie und die Arbeitsweise in den letzten Jahrzehnten stark beeinflusst. Mit der Einführung der elektronischen Datenverarbeitung konnten viele Aufgaben von Computern übernommen werden, die vorher von Menschen ausgeführt wurden. In der Anfangszeit der Informationstechnik wurden Großrechner in Unternehmen eingesetzt um wiederkehrende Aufgaben abzuarbeiten. Großrechner sind dazu in der Lage, mehrere hundert Benutzer gleichzeitig abzuwickeln.

Um die Ressourcen eines solchen Großrechners besser nutzen zu können und verschiedene Systeme auf einer Hardware auszuführen, entwickelte IBM 1966 das erste Betriebssystem, welches Virtualisierung unterstützte. Diese Technologie veränderte die Arbeit der IT grundlegend: Nun war es möglich, mehrere Anwendungen isoliert voneinander auf einer Hardware auszuführen.

Als 2013 Docker angekündigt wurde, wurde neben der Virtualisierung eine weitere Möglichkeit bekannt, um Anwendungen isoliert zu betreiben. Containervirtualisierung gab es mit LXC schon 2008, Docker machte diese Technologie aber massentauglich. Container virtualisieren Anwendungen und minimieren dabei die Abstraktionsschicht zwischen echter Hardware und Anwendung. Klassische Virtualisierung abstrahiert komplette Computer, auf denen ein Betriebssystem und dann die Anwendungsschicht läuft. Die Abstraktionsschicht ist damit größer als bei Containern.

Die Verringerung der Abstraktionsschicht führt in der Regel dazu, dass auf derselben Hardware mehr Applikationen mit gleichbleibender Performanz betrieben werden können. Die Firma Docker selbst spricht in ihren Beispielrechnungen von Kosteneinsparungen von über 60 %.

Ob so eine hohe Einsparung erreicht wird, ist natürlich stark von den Rahmenbedingungen abhängig. In der Praxis ist eine Reduktion fast immer spürbar.

Die Verwendung von Containern bietet über die Kostenreduktion hinaus noch weitere Vorteile: Container lassen sich sehr gut in eine Continuous Delivery Pipeline integrieren. In der modernen Softwareentwicklung durchläuft die Software nach einer Änderung eine Reihe von Schritten, die sogenannte Pipeline: Sie wird kompiliert, automatische Tests werden durchlaufen und es wird geprüft ob die Schnittstellen der Software noch wie erwartet funktionsfähig sind. Anschließend wird die Software automatisch ausgeliefert. Genau in diesem Schritt haben sich Container in der letzten Zeit immer stärker etabliert. So bieten Container die Möglichkeit Software mit all ihren Abhängigkeiten, also weiteren Programmen, die zur Ausführung der Software notwendig sind, wie Java oder spezielle Bibliotheken, zusammen auszuliefern. Der Betreiber der Software muss dann nur noch den Container in

Betrieb nehmen und braucht die Abhängigkeiten nicht zu beachten.

Zusätzlich bietet das Ökosystem rund um Container weiter positive Aspekte: Wenn eine Applikation starken Lastschwankungen ausgesetzt ist, ist es möglich, die Container, in denen die Applikation läuft, automatisch zu skalieren. Somit erhöht sich bei einer starken Beanspruchung die Anzahl der Container mit der Applikation. Ein Load Balancer verteilt die Anfragen der Nutzer dann um, sodass zum Schluss alle Systeme gleich ausgelastet werden.

Diese automatische Skalierung ist gerade bei Containern attraktiv, da nach dem Start die Applikation sofort einsatzfähig ist. Hinzu kommt, dass die Startdauer eines Containers sechs Mal schneller ist als die einer Virtuellen Maschine.

Auch für einen robusten Betrieb werden nützliche Funktionen geboten. So lassen sich über „HealthChecks" Regeln definieren, die Applikationen nach einem Absturz automatisch neu starten oder abhängige Dienste – wie z. B. eine Datenbank -für die eigentliche Applikation starten. Dadurch wird der Betrieb einer komplexen Infrastruktur erleichtert.

Durch die Verwendung von Containern, anstelle Virtueller Maschinen, ändern sich die Prozesse der Software- und Betrieb-Teams, denn für die Arbeit mit Containern ist ein hohes Maß an Automatisierung notwendig. Applikationen lassen sich nur als Container bereitstellen, wenn die Installation und Konfiguration automatisiert wurde.

Diese Automatisierung bringt weitere Vorteile mit sich: So werden die Installations- und Bereitstellungsprozesse reproduzierbar und die Mitarbeiter müssen sich nicht um diese Routineaufgaben kümmern.

Die Technologie im Detail

Im Folgenden werden die angesprochen Technologien im Detail voneinander abgegrenzt:

Virtualisierung ist eine Technologie bei der die Rechenleistung eines Computers oder Servers durch eine Softwareschicht zusammengefasst und weitergegeben wird. Dies ermöglicht eine bessere Ausnutzung der Ressourcen des Computers oder Servers. Virtualisierung ermöglicht so die Ausführung von verschiedenen Betriebssystemen oder Anwendungen auf derselben Hardware. Der Server mit der realen Hardware wird Host genannt, die Systeme die auf ihm laufen Gäste. Die Software die die Virtualisierung übernimmt wird als Hypervisor bezeichnet.

Eine **Virtuelle Maschine** (Bild 1), kurz VM, wird in einem Hypervisor erzeugt. Die VMs auf einem Host sind abgeschottet und wissen nicht voneinander. Das Betriebssystem oder die Anwendungen auf dem System bemerken nicht dass sie virtualisiert sind. Die Formate der VMs unterscheiden sich von Hypervisor zu Hypervisor und sind daher nicht einfach zu migrieren. Die gängigen Hypervisoren sind VMware, Hyper V, KVM und Xen.

In jeder VM ist ein Betriebssystem und alle abhängigen Komponenten – die für die Ausführung der eigentlichen Applikation notwendig sind – installiert. In dieser VM wird dann die Anwendung installiert und ausgeführt.

Container (Bild 1) sind vergleichbar mit virtuellen Maschinen: Auch hier werden Computersysteme gekapselt auf einem System betrieben. Die Softwareschicht zwischen realem Computer und der Anwendung, die isoliert ausgeführt wird, wurde gegenüber einer VM deutlich reduziert. Unter demselben Betriebssystemkern laufen mehrere abgeschottete identische Systemumgebungen. Es wird daher nicht ein Betriebssystem für jeden Container benötigt.

Ein Container wird immer aus einem Containerabbild gestartet das einer Vorlage entspricht. Deshalb ist eine Anwendung in einem Container in wenigen Minuten betriebsbereit. Die Containerabbilder liegen in einem Standardformat vor. Dieses Format ist auf allen Plattformen und Architekturen gleich. Damit werden die Abhängigkeiten zur Ausführung eines Containers minimiert. Wie sein Vorbild in der Realität, lässt sich ein Container einfach umziehen und in jeder Containerumgebung starten. Das Format ist somit portabel.

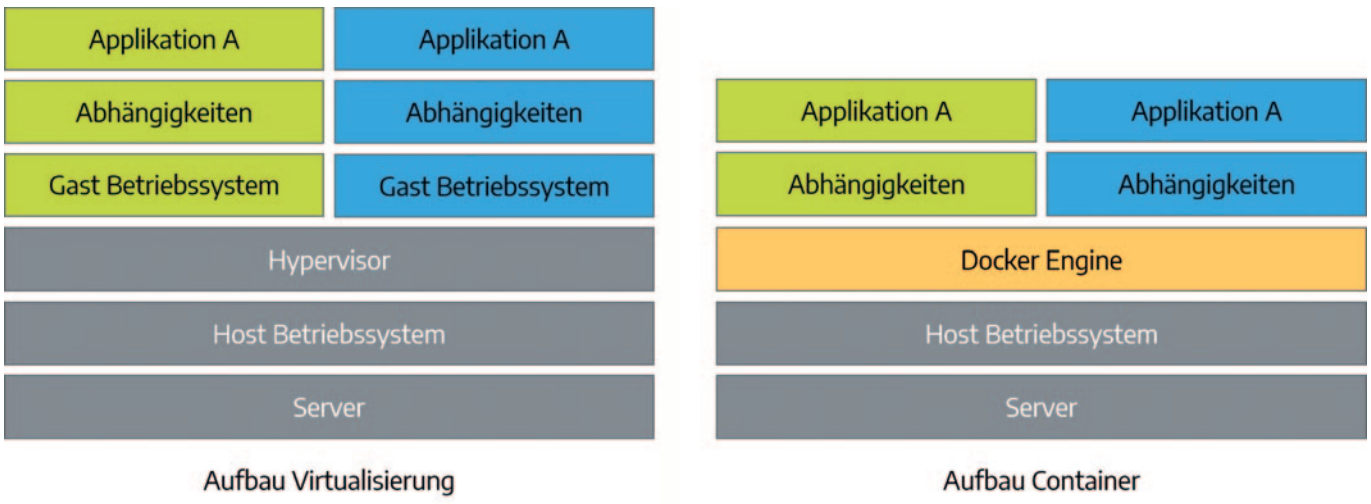

Bild 1: *Virtuelle Maschinen und Container im Vergleich*

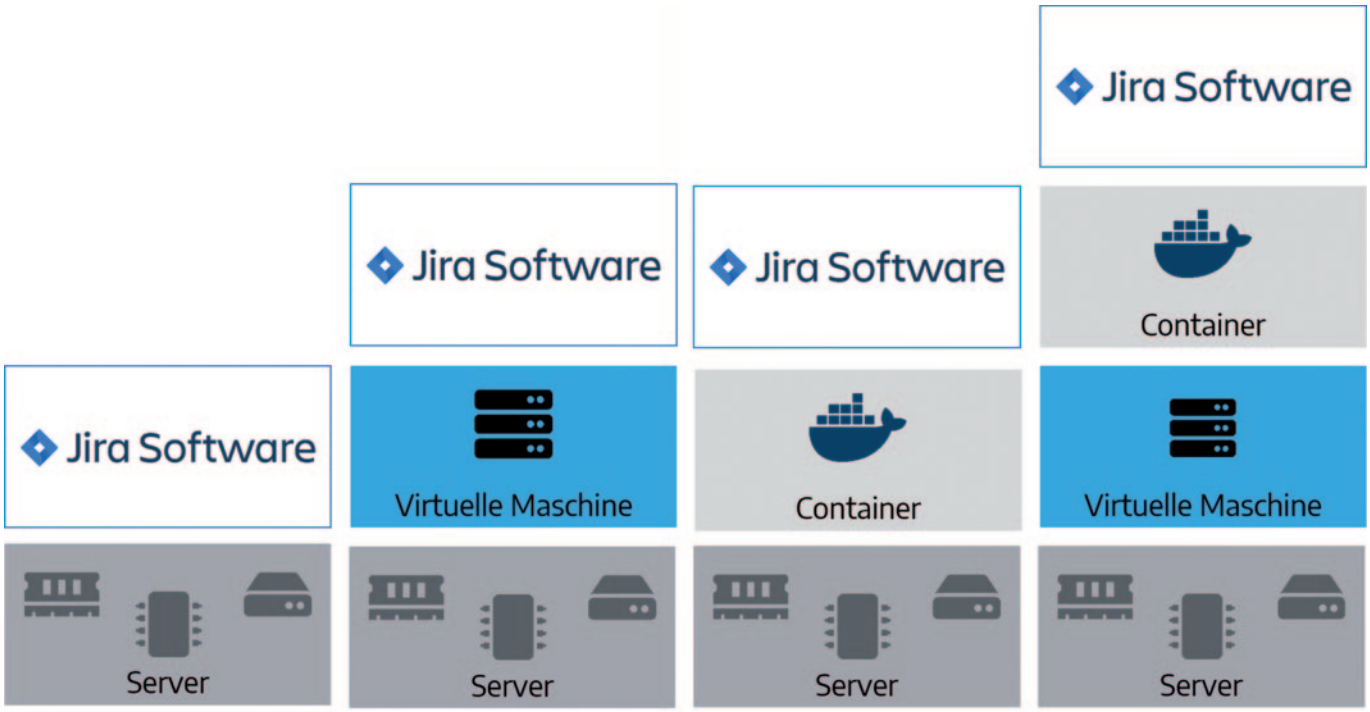

Bild 2: *Übersicht der Untersuchten Systeme*

Docker ist eine Entwicklerplattform für Containervirtualisierung und wurde 2013 veröffentlicht. Der Fokus von Docker liegt auf der Bereitstellung von Applikationen.

Die drei Technologien natives System (z. B.: Server), Virtualisierung und Containervirtualisierung haben in ihren Anwendungsbereichen Überschneidungen. So ist bei dem Betrieb einer Anwendung die Entscheidung zu treffen welche Technologie zu verwenden ist.

Aller Anfang ist schwer

Um jetzt mit den Containern im Unternehmen zu starten, ist es ratsam, mit kleinen Umgebungen zu experimentieren. Dadurch können die Teams in der neuen Technologie geschult und „Best Practises" entwickelt werden. Häufig bietet es sich an bestehende Continuous Integration Pipelines um das „Delivery" zu erweitern. Dies kann erreicht werden indem die Software nach der „Integration" als Container verpackt und ausgeliefert wird. So lassen sich schnell eigene Test- oder Demostellungen updaten und betreiben.

Ein häufiges Problem in größeren Unternehmen ist das für eine Containerumgebung auch für einen langfristigen Betrieb von der IT keine eigenen Hardware Server bereitgestellt werden. Es läuft fast immer darauf hinaus, dass eine virtuelle Maschine zur Verfügung gestellt wird.

Dabei kommt die Frage auf, ob der Betrieb von Containern innerhalb von Virtuellen Maschinen – sprich mit zwei Virtualisierungsschichten – überhaupt performant sein kann? Dieser Frage ist der Autor des Artikels mit einer Messung nach wissenschaftlichen Methoden nachgegangen:

Leistungsmessung der Systeme

Die Leistung der Systeme wurde anhand von Jira verglichen. Damit die Umgebung reproduzierbar ist, wurden die

	Natives System	VM	Container	VM-Container
Summe [min]	18,06	21,00	18,40	18,40
Abweichung zum nativen System	0,00%	16,28%	1,91%	16,20%

Tabelle 1: *Summe der Ergebnisse des Benchmark Skripts*

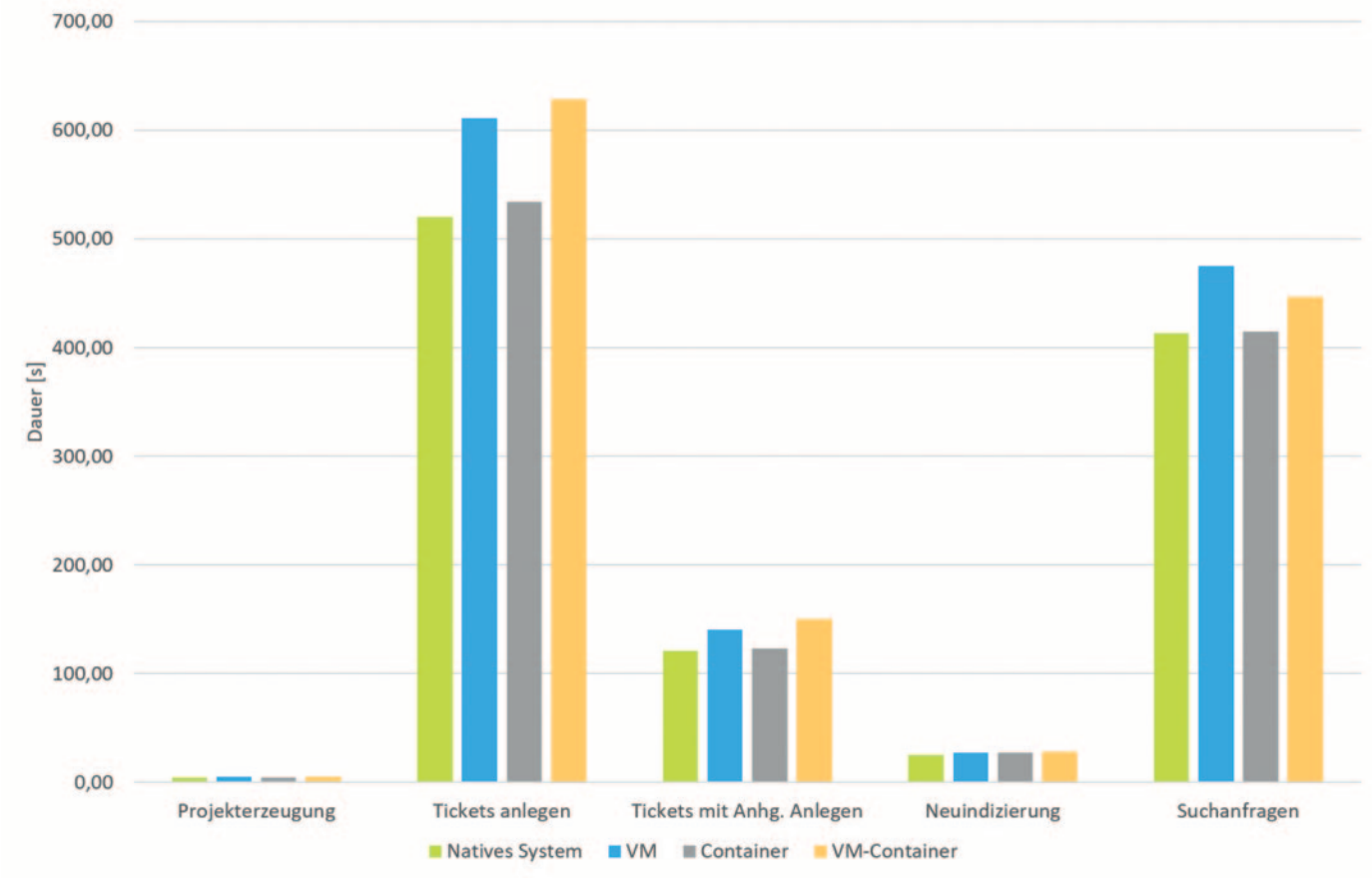

Bild 3: *Ergebnisse des Benchmark Skripts*

Systeme auf identischer Hardware in einem eigenen Netzwerk automatisiert installiert.

Jira ist ein Softwareprodukt des australischen Softwareherstellers Atlassian. Diese Software ist ein Ticketsystem, das in der Software- oder Produktentwicklung eingesetzt wird. Es unterstützt agile Methoden wie Scrum oder Kanban, aber auch klassische Methoden wie das Wasserfallmodell.

Untersucht wurde die Leistung des nativen Systems, der Virtuellen Maschine des Containers und des Containers in einer VM (siehe Bild 2). Als Virtualisierungsumgebung oder Hypervisor für die Virtuelle Maschine wurde KVM verwendet. Diese Umgebung ist stabil und besitzt eine breite Nutzerbasis da sie einfach auf jedem Linux verwendet werden kann. Für die Container Virtualisierung war Docker im Einsatz, weil es die aktuell verbreitetste Lösung ist.

Für den Benchmark wurden mithilfe eines Scripts verschiedene Messungen von Jira durchgeführt:

- Projekterzeugung
- Erstellung von Tickets mit und ohne Anhang
- Neugenerierung des Suchindexes
- Stellen von Suchanfragen

Der Benchmark erzeugte dabei jeweils 5.000 Tickets sowie 500 Tickets mit Anhängen und 500 Suchanfragen. Die Messung wurde von einem zweiten unabhängigen System vorgenommen und mehrfach wiederholt. Die Rahmenbedingungen waren so gewählt, dass möglichst keine externen Einflüsse die Messungen verfälschen konnten.

Messergebnisse

In der Summe lag der Container (Docker) mit dem nativen System fast gleichauf, die Virtuelle Maschine und der Container (Docker) in der VM waren etwa 16 % langsamer (siehe Tabelle 1).

Im Detail (siehe Bild 3) lässt sich kein eindeutiger Trend feststellen. So war das Anlegen von Tickets in der Virtuellen Maschine ca. 10 % langsamer als im Container, den Suchindex erzeugte die Virtuelle Maschine aber schneller als der Container. Bei der Suche lag die Virtuelle Maschine jedoch 15 % hinter Docker. Virtuelle Maschine-Docker war in fast allen Schritten langsamer als VM, nur bei den Suchanfragen war es ca. 7 % schneller.

Es stellt sich die Frage, warum der Container in einer Virtuellen Maschine bei der Neugenerierung des Suchindex so viel schneller ist, als die Virtuelle Maschine. Die Messergebnisse decken sich mit den Erwartungen. Neu ist das Ergebnis, dass Container in einer Virtuellen Maschine fast genauso schnell sind wie die Virtuelle Maschine.

Interpretation der Messergebnisse

Um der Frage nachzugehen warum die Neugenerierung des Suchindex mit einer zusätzlichen Virtualisierungsschicht schneller ist, wurde eine weitere Messung vorgenommen.

Die Generierung des Suchindex ist eine festplattenintensive Arbeit. Daher wurde die Schreibleistung der Systeme mit

	Natives System	VM	Container	VM-Container
Schreiben von 1GB	241,00 MB/s	248,75 MB/s	242,75 MB/s	261,75 MB/s
	0,00%	3,22%	0,73%	8,61%
Schreiben von 1.000 mal 512 Byte	14,40 MB/s	14,18 MB/s	5,40 MB/s	5,38 MB/s
	0,00%	-1,53%	-62,50%	-62,67%

Tabelle 2: *Vergleich der Festplattenleistung*

dem Linux Tool dd verglichen. Dazu wurde eine Datei mit 1 Gigabyte und 1.000 Dateien mit 512 Byte geschrieben. Die Messungen wurden mehrfach durchgeführt und die Ergebnisse gemittelt.

Die Resultate (siehe Tabelle 2) zeigen, dass Docker beim Lesen und Schreiben von einem Gigabyte immer schneller ist. Gerade Docker in der VM ist deutlich schneller, als das native System.

Die Ursache dafür ist, dass Docker einen effizienteren Zwischenspeicher hat, als alle anderen betrachteten Systeme. Das könnte der Grund sein, warum die Suchanfragen im Container in einer Virtuellen Maschine so viel schneller waren, als die VM allein. Die Index-Datei wird scheinbar besser zwischengespeichert. Bei der Latenzmessung ist Docker etwas langsamer. Das lässt sich dadurch erklären, dass bei der Virtualisierung jedes Lesen und Schreiben durch die Abstraktionsschichten abgehandelt werden muss. Werden viele kleine Dateien gelesen und geschrieben, wächst dieser Overhead und bremst die gesamte Operation aus. Dies kann dann auch nicht durch effizientes Zwischenspeichern ausgeglichen werden, dennoch ist Docker hier schneller als die VM.

Fazit

Sowohl für den Betrieb von Containern als auch von Virtuellen Maschinen gibt es gute Gründe. Letztendlich muss in jedem Anwendungsfall entschieden werden, welche der beiden Technologien besser geeignet ist.

Die Ergebnisse der Messung decken sich in weiten Teilen mit der Literatur und sind somit belastbar. Hinzu gekommen ist die Betrachtung der Leistung von Containern in einer Virtuellen Maschine. Dass die Anwendung in einem Container in einer VM genauso schnell ist, wie die Anwendung in einer Virtuellen Maschine, war nicht zu erwarten, da eine Abstraktionsschicht hinzukommt.

Die Virtualisierung mit Container ist fast genauso schnell wie die Ausführung auf dem nativen System und deutlich schneller als eine Lösung mit Virtuellen Maschinen. In der Praxis macht es also fast keinen Unterscheid ob eine Applikation in einer VM oder in einem Container in einer VM betrieben wird. Ist aber die Möglichkeit gegeben, Container statt VMs zu verwenden, sind Container aus Performanzgründen vorzuziehen. ◀

Kontakt

Tim Stoffel
Fachinformatiker
im Bereich
Anwendungsentwicklung
:em engineering methods AG
E-Mail: tim.stoffel@em.ag