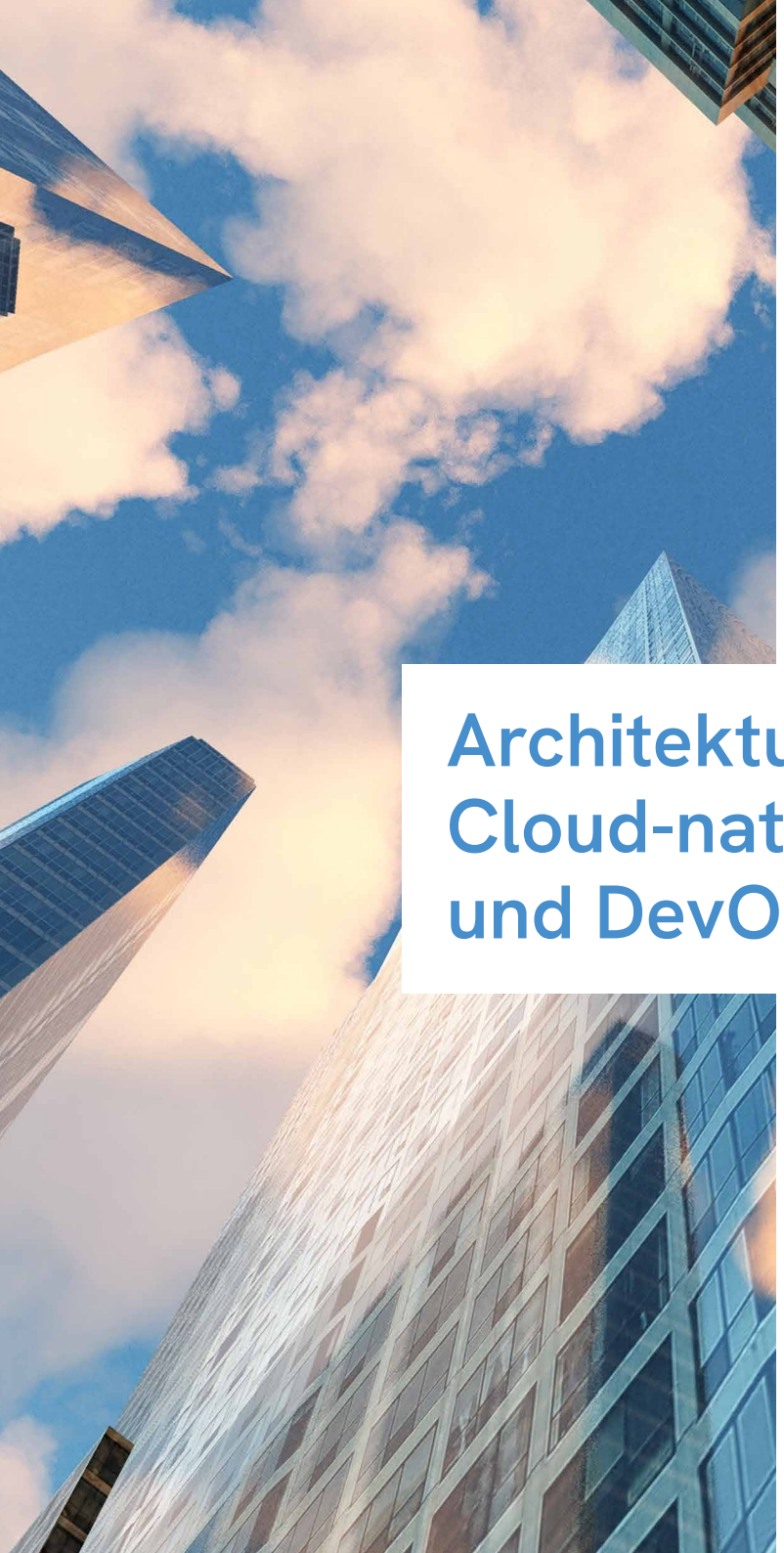


Architekturarbeit im Zeitalter Cloud-nativer Architekturen und DevOps-Teams

Im Jahr 2020 erlebten wir den Beginn einer nicht vorhergesehenen Beschleunigung der weltweiten Digitalisierungsbestrebungen. Die Digitalisierung ist auf dem Vormarsch oder sogar „per Turbo“ beschleunigt worden.
Dabei ändert sich die Erwartungshaltung der Menschen an Software zunehmend. Statt Software „herunterzuladen” wird vorzugsweise eine agile Bereitstellung von (Self-) Services präferiert.

Diese Dienste sollen selbstverständlich der „Cloud-Philosophie” entsprechen und damit jederzeit erstellbar/buchbar/abonnierbar, von jedem Ort online erreichbar und immer verfügbar sein. Trotz der Erreichbarkeit müssen alle verarbeiteten Daten vor unerlaubtem Zugriff geschützt sein. Die Beseitigung von Fehlern und Störsituationen findet erwartungsgemäß auch rund um die Uhr statt.

Die Erreichung dieser Ziele war bis vor wenigen Jahren nur den IT-Organisationen von großen Konzernen vorbehalten. Die Cloud hat das verändert und die dafür notwendige Technologie zugänglicher und erschwinglicher gemacht.

Beschleunigte Digitalisierung erzwingt IT-Neuausrichtung

Um die oben genannten Cloud-typischen, nicht-funktionalen Anforderungen in einem IT-Service zu erfüllen, gilt es derzeit – auch aufgrund vieler Innovationen der Technologiehersteller – als alternativlos, sich der „Cloud-native-Philosophie“ hinzugeben.

Dabei stellt man fest, dass der Digitalisierungserfolg stark davon abhängt, wie gut es einer Organisation gelingt, sich in den folgenden Disziplinen auszurichten:

- Organisation, Prozesse und Methoden (SAFe®, Scrum, Kanban)
- Organisations- und Teamkultur (DevOps & Continuous Improvement, Lebenslanges Lernen, Fehlerkultur, Lean Culture,...)
- Architektur (Domain Driven Design, Evolutionäre Architekturen, Microservices, Cloud-native Architecture, Event Driven Architecture, APIs, ...)
- Technologie (Cloud-native Technologien, Infrastructure as Code, Container-Orchestrierung/Kubernetes, NoSQL, Messaging/Streaming...)

Welchen Einfluss hat diese Neuausrichtung auf die Aufgabenstellung im Bereich der Softwarearchitektur? Es wirkt logisch, dass es starke Berührungspunkte mit den Disziplinen „Architektur“ und „Technologie“ gibt, aber welchen Einfluss haben Organisation, Prozesse, Methoden und Kulturaspekte auf die Architekturarbeit?

Allgemeine Aufgaben eines Softwarearchitekten

Bevor wir uns mit den Aufgabenstellungen der Softwarearchitekten im Cloudnative-Umfeld auseinandersetzen, lohnt sich ein Blick auf die Lehrmeinung bzgl. der Aufgaben eines Softwarearchitekten.

Softwarearchitekten sind verantwortlich für die Erreichung der Qualität und der gewünschten Funktionalität eines softwarebasierten Systems. Zwar ist diese Rolle in allen Phasen des Software Lifecycles beteiligt, i.d.R. aber nicht für die Umsetzung der Funktionalität verantwortlich.

Ein wichtiger Bestandteil des Aufgabenspektrums befasst sich logischerweise mit dem abstrakten Entwurf der Systemkomponenten und Beziehungen zueinander. Die Architektur eines Systems sollte dabei nachweislich von Qualitätszielen beeinflusst worden (z.B. über den Aufbau von Qualitätsbäumen bzw. CTQ Analysen) und mit Hilfe von messbaren Qualitätsszenarien jederzeit hinsichtlich Tragfähigkeit überprüfbar sein.
Am Ende werden diese Entscheidungen pragmatisch in einer zielgruppengerechten Architekturdokumentation erfasst.

Einen großen Teil ihrer Arbeitszeit verbringen Softwarearchitekten allerdings in der Funktion als Ansprechpartner oder Advokat für die unterschiedlichsten Zielgruppen. Ein Softwarearchitekt kommuniziert u.a. „das (nicht) Mögliche“, macht versteckte Risiken sichtbar und verhandelt notwendige technische/architektonische Investments in das System mit Sponsoren, Projektleitern, Auftraggebern und Fachbereichen.
Zusätzlich wird durch eine enge Zusammen- und Mitarbeit mit den Entwicklungs-, Qualitätssicherungs- und Betriebsteams dafür gesorgt, dass die Architekturvision in der Realität auch die gewünschten Ziele erfüllt. Dabei wird der Sinn der Architektur vermittelt, gegen die Erosion der Architektur gekämpft, es werden technische Hypotheken dokumentiert, neue und bessere Feedbackmechanismen

(Logging, Monitoring, Alerting) etabliert und vieles mehr. Und manchmal werden auch naturwissenschaftliche Gesetze ausgehebelt, wenn der Auftraggeber sich das so gewünscht hat ☺.

Cloud-native Technologie schafft neue Möglichkeiten

Zur Erreichung der anvisierten Qualitätsziele eines Systems spielt die Wahl der richtigen Technologien eine Schlüsselrolle. Ganz konkret schaffen z.B. Technologien, die unter der Schirmherrschaft der Cloud Native Computing Foundation (kurz CNCF) beherbergt werden, neue Möglichkeiten, wie man flexible Architekturen schaffen kann.
CNCF Technologien sind allesamt auf Extrembedingungen ausgerichtet. Bis vor ein paar Jahren galten verteilte Systeme und dynamische Infrastrukturen oft noch als Spielwiese für Nerds, die auf Herausforderungen stehen. Heute ist dies eine seriöse und notwendige Arbeitsbasis für Softwarearchitekten. Die Ausrichtung des CNCF lässt sich wie folgt zusammenfassen:

- Empowerment von Software Engineers und DevOps-Teams durch Self-Services und einen hohen Standardisierungsgrad
- Ermöglichung von häufigen und kritischen Änderungen an verteilten Systemen
- Verbesserung bezüglich den Aspekten Überwachung, Management und Stabilität von komplexen Systemen
- Lauffähig in modernen dynamischen Umgebungen (Private, Hybrid und Public Clouds)

Bei der Wahl der konkreten Technologie kommen Softwarearchitekten mit einer äußerst undankbaren Aufgabe in Berührung. Entweder man pflückt und genießt die Low-Hanging-Fruits des präferierten Cloud-Anbieters, nimmt den Lock-In in Kauf und freut sich über einen besseren Time-To-Market auf Kosten der Portierbarkeit eines Systems. Oder man setzt auf „Standards“. Bei der Auslegung des Begriffs „Standard“ scheiden sich auch die Geister.
Welche Bedeutung hat der Begriff Standard in diesem Umfeld?

„Innovative Open Source Frameworks und Werkzeuge mit sehr großen Entwickler-Communities, die von den fünf größten Technologieherstellern nur so lange am Leben erhalten werden, wie auch Bedarf dafür existiert“ oder „Langfristig abwärtskompatible Frameworks, die in häufig langwierigen Standardisierungsprozessen von den fünf größten Technologieherstellern geprägt werden“.

Egal wie man das Blatt wendet und dreht, man wird am Ende Technologie verwenden und gegebenenfalls auch bezahlen müssen. Bei der Bewertung der gesuchten Technologie sollte insgesamt der Fokus auf folgenden Fragestellungen liegen:

- Zu den Fähigkeiten und dem Delivery-Prozess des Teams passende Abstraktion der Infrastruktur (Serverless, PaaS, Container as a Service, Infrastructure as Code)
- Offenheit der Technologie und Stärke der Community (Support, Dokumentation, Blueprints/Beispiele, angrenzende Ökosysteme mit Adaptern und Zusatzwerkzeugen)
- Reifegrad der Lösung, Vision und Marktdurchdringung des Herstellers
- Technologiekosten (kostenlos, skalierend auf Basis der Last oder langfristig mit Rahmenvertrag)

Eigenschaften einer Cloud-native Architektur

Cloud-Technik liefert Softwarearchitekten neue Möglichkeiten, Dinge zu tun, die mit bisherigen Mitteln in den eigenen Rechenzentren häufig nicht mit vertretbaren Aufwänden (Dauer, Kosten) umsetzbar bzw. mit zusätzlichen Risiken verbunden waren (z.B. fehlende Erfahrung im Betrieb von kritischer

Infrastruktur, NoSQL Datenbanken, usw.). Zu den neuen Möglichkeiten zählen folgende Aspekte:

- Self-Healing, automatisierte Skalierung und Elastizität
- Reduzierung der Tätigkeiten im Bereich Infrastruktur für DevOps-Teams
- Autonomie und Flexibilität durch Self-Services und Automatisierungslösungen
- Transparenter und kosteneffizienter Betrieb der (geteilten) Infrastruktur

Darüber hinaus bringen Cloud-native Architekturen eine eigene Philosophie mit sich, die sich u.a. durch folgende Zielsetzungen beschreiben lässt.

- Arbeitserleichterung durch Nutzung von Services/Abstraktionen des Cloud-Anbieters
- Bessere Systemqualität durch hohen Automatisierungsgrad (Infrastruktur, CI/CD, Elastizität, Monitoring/Alerting)
- Vermeidung von Flaschenhälsen (z.B. durch Einsatz stark skalierender Persistenzsysteme und Implementierung von zustandslosen Diensten)
- Behandlung von Schnittstellen/APIs als Produkte (API Lifecycle Management für REST, aber auch für asynchrone APIs wie z.B. Kafka)
- Kontinuierliche Überprüfung der Sicherheit aller Komponenten und Härtungsmaßnahmen
- Ausrichtung der Architektur auf Veränderung (Evolutionäre Architekturen)

Aus der Definition der CNCF (http://www.cncf.io) für Cloud-native Systeme:

> These techniques enable loosely coupled systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.

Einfluss von Cloud-native auf die Architekturarbeit

Dank dem zwischenzeitlich erreichten Reifegrad der Technologie haben Softwarearchitekten einen wesentlich größeren Werkzeugkoffer, um die Qualität eines Systems beeinflussen zu können. Früher aufgrund der Betriebsaufwände noch undenkbar, werden in diesem Umfeld häufig verteilte Systeme entworfen, um einzelne Systembestandteile hinsichtlich besonderer nicht-funktionaler Anforderungen optimieren zu können.

Neben der klassischen, monolithisch geprägten Architektur werden zunehmend verteilte Architekturen (Microservice Architecture) bzw. ereignisbasierte Architekturen (Event Driven Architecture) gewählt, sowohl um neue Systeme zu implementieren als auch um Legacy-Systeme schrittweise zu modernisieren (z.B. nach Strangler-Fig Pattern). Bei Modernisierung ist der erste Schritt dabei Cloud-tauglich (Cloud-ready) und im zweiten Schritt Cloud-native zu werden.

Der zunehmende Einfluss von Lean Product Management Vorgehensmodellen bringt auch diverse Tücken in der Architekturarbeit mit sich. So macht es keinen Sinn, sich in frühen Phasen (Walking Skeleton, Prototyping) zu intensiv mit Performance und Skalierung zu beschäftigen, wenn damit die funktionale Reifung des Produkts verlangsamt wird.
Nach dem Initial Product Release bzw. in der MVP Phase ergeben sich in der Regel aber viele neue Erkenntnisse, die häufig ein Überdenken der Architektur (und ggf. auch Technologieänderungen) unumgänglich machen.

Früher waren Softwarearchitekten gegebenenfalls. noch vorrangig mit der Modularisierung und Strukturierung monolithischer Codebasen beschäftigt (u.a. durch Einsatz des UML Standards). In einer agilen Umgebung müssen sich Softwarearchitekturen aufgrund der sich stetig verändernden fachlichen Realität weiterentwickeln können. Den idealen Service-Schnitt findet der auf Cloud

ausgerichtete Softwarearchitekt, wenn er mit Hilfe von folgenden Kriterien und Methoden abwägt:

- Domain-Driven-Design und Event-Storming
- (Zielsetzung: Fachlicher Schnitt, IT/Business-Alignment, Gemeinsame Sprache)
- Organisatorischer Schnitt (Servicegröße passend zu Teamgröße)
- Sonstige Rahmenbedingungen (Limitierungen durch Technologiewahl, 3rd Party Vendoren, Outsourcing mit Hilfe von APIs usw.)

Neben dem optimalen Service-Schnitt spielt auch die Wahl der passenden Kommunikationsstile (synchron vs. asynchron) und Protokolle eine wichtige Rolle. Um diese Kommunikationsbeziehungen stabil zu halten, ist es sinnvoll, diese APIs und deren Lebenszyklen bewusst zu verwalten (Provider und Consumer Prozess) und Änderungen im Zustand der APIs angemessen zu kommunizieren.
Die Steuerung des API-Lebenszyklus, das Exponieren der Services über Gateways, aber auch das Provisionieren von API Keys sollten ohne manuelle Aufwände innerhalb von CI/CD Toolchains erfolgen. Komplexe Service-Landschaften werden gegebenenfalls sogar mit Hilfe von Services Meshes (wie z.B. Consul, Linkerd oder Istio) verwaltet. Der richtige organisatorische und prozessuale Umgang mit APIs ist das A und O beim Aufbau von verteilten Systemen und ein Garant für ein schnelles Wachstum des sinnvoll genutzten Portfolios an Services.

Die Architektenrolle in DevOps-Teams/Organisationen

Beim Lesen dieses Artikels fällt vermutlich die große Anzahl an Qualitätsattributen, Architekturpatterns, Technologien und Konzepten auf. Ja, die Welt der Cloud-nativen Architekturen ist in jeglicher Hinsicht vielfältig und komplex.

Seit 2014 verbreitet sich eine Philosophie in IT-Bereichen, die Antworten auf die Herausforderungen bzgl. hoher Komplexität liefern soll. Wir hatten bereits erwähnt, dass Cloud-native Technologien (z.B. aus dem CNCF) für DevOps-Teams konzipiert wurden. Man muss aber auch feststellen, dass ohne cross-funktionales Arbeiten und „Ingenieursdenken“ die Komplexität dieser Architekturen und Technologien schwer beherrschbar sind. Die beiden gehören zusammen wie Popcorn und Kino.

Ein nicht zu unterschätzendes Erfolgskriterium für DevOps-Organisationen ist die Übergabe von Verantwortung in die Engineering-Teams. Diese Dezentralisierung ist notwendig, um Silos und Flaschenhälse sukzessive abbauen zu können. Der Software- bzw. Systemarchitekt ist in vielen Organisationen eine Rolle, die häufig zentral bzw. querschnittlich im Organigramm positioniert ist. Wie passt das nun in diese Welt? Gibt es immer noch Bedarf für zentrale Architekten oder sollen cross-funktionale DevOps-Teams diese Rolle nun vollständig selbst übernehmen?

In unserer Praxis haben wir erlebt, dass es weiterhin die Notwendigkeit gibt, bestimmte Architekturaufgaben zentral zu bewältigen. Allerdings sollte dieses zentrale Architekturteam seinen Einfluss nicht vordringlich aus der Hierarchie herleiten, sondern aus dem breit angelegten Respekt gegenüber dem Wissen, dem permanenten Lernen und der sehr guten Kommunikation dieses Teams. Der Fokus sollte hierbei auf folgenden Themen liegen:

- Die Erstellung von Technologieleitfäden (bzw. Leitlinien)
- Die Diskussion und Definition von Blueprints
- Eine zentrale Technologieauswahl, aber stets offen für Verbesserungsvorschläge
- Standardisierung von diversen Security und Delivery-Abläufen
- Aufbau von allgemeinen Dingen wie Service-Meshes, einer Event-Infrastruktur u.v.m.

Diese Aspekte zentral zu harmonisieren ist enorm wichtig, um in der aktuell immer vielfältigeren IT-Welt die Balance zwischen Verkrustung und Wildwuchs zu finden und zu erhalten. Dabei geht es nicht darum, die Autonomie der DevOps-Teams über Gebühr zu beschneiden, sondern sich täglich aufs Neue im Sinne aller Beteiligten auf einen gemeinsamen „Mainstream“ zu fokussieren.

Am Ende muss in einer DevOps-Organisation der gesamte Schwarm der beteiligten Menschen in der Lage sein, mithilfe von internen Communities, Self-Services Dokumentation, Consulting durch interne Service-Provider-Teams und auch durch externe Unterstützung die eingesetzten Technologien zu betreiben und ein effizientes Onboarding neuer Teams zu gewährleisten.

Neben den in der Tendenz zentralen Architekturaufgaben fallen auch innerhalb der DevOps-Teams viele gewichtige Architekturentscheidungen an. Dedizierte Rollen gibt es selten in diesem Team, das bedeutet, dass Softwarearchitekten dort mehr Praxisbezug (Entwicklungstätigkeiten, Automatisierung, Testen) haben und stärker im Entwicklungsalltag eingebunden sind als in einer zentralen Architekturrolle.
Es macht Sinn, ein Team-Mitglied im DevOps-Team zu haben, dessen Steckenpferd die Softwarearchitektur ist (inkl. der oben genannten Aufgaben). Allerdings liegt die Qualität in der Verantwortung des ganzen Teams, weshalb auch die Architekturarbeit auf möglichst viele Schultern verteilt werden sollte.

In dieser Organisationsform wirken daher häufig zentrale Architekten als Agenten in neuen Produktteams für einen temporären Zeitraum mit, um methodisches Wissen zu übergeben.
Oder die Softwarearchitekten aller DevOps-Teams treffen sich regelmäßig z.B. in der Organisationsform einer Gilde, um gemeinsame Erkenntnisse in Leitlinien und Grundsatzentscheidungen zu gießen und auch hier den horizontalen Wissensfluss in der Organisation zu fördern und Entscheidungen am Ende gegebenenfalls auch gemeinsam zurück in die Organisation zu evangelisieren.

Fazit
Das Wesen der Architekturarbeit hat sich im Cloud-native Umfeld nicht verändert. Allerdings ist der Werkzeugkoffer größer geworden. Verteilte Architekturen bzw. dezentrale Organisationsformen und Prozesse führen dazu, dass neben den zentralen Architekturaufgaben auch wesentlich mehr Architekturarbeit innerhalb der DevOps-Teams stattfinden muss. Diese Teams müssen für diese neue Rolle und Aufgabenstellung vorbereitet und langfristig begleitet werden. Dies ist eine Kernaufgabe zentraler Architekturteams, deren Alltag bei einer Ausrichtung auf Cloud-native durch eine erheblich veränderte Kultur nicht nur in der IT-, sondern der Gesamtorganisation neu geprägt wird.

ARS
Passion for your digital future

Über ARS

Wir sind ARS, und wir leben für die digitale Zukunft unserer Kunden – das ist mehr als nur ein Claim für uns, es ist unser Versprechen. In einer Welt, die zunehmend digitalisiert wird, sind wir vertrauenswürdiger Partner, der Unternehmen durch den gesamten Lebenszyklus ihrer Softwareprojekte begleitet. Von der strategischen Planung über die effiziente Umsetzung bis hin zur Transformation der Arbeitsweisen, Prozesse und Infrastruktur – wir begleiten unsere Kunden mit maßgeschneiderten Lösungen und echter Begeisterung für Technologie.

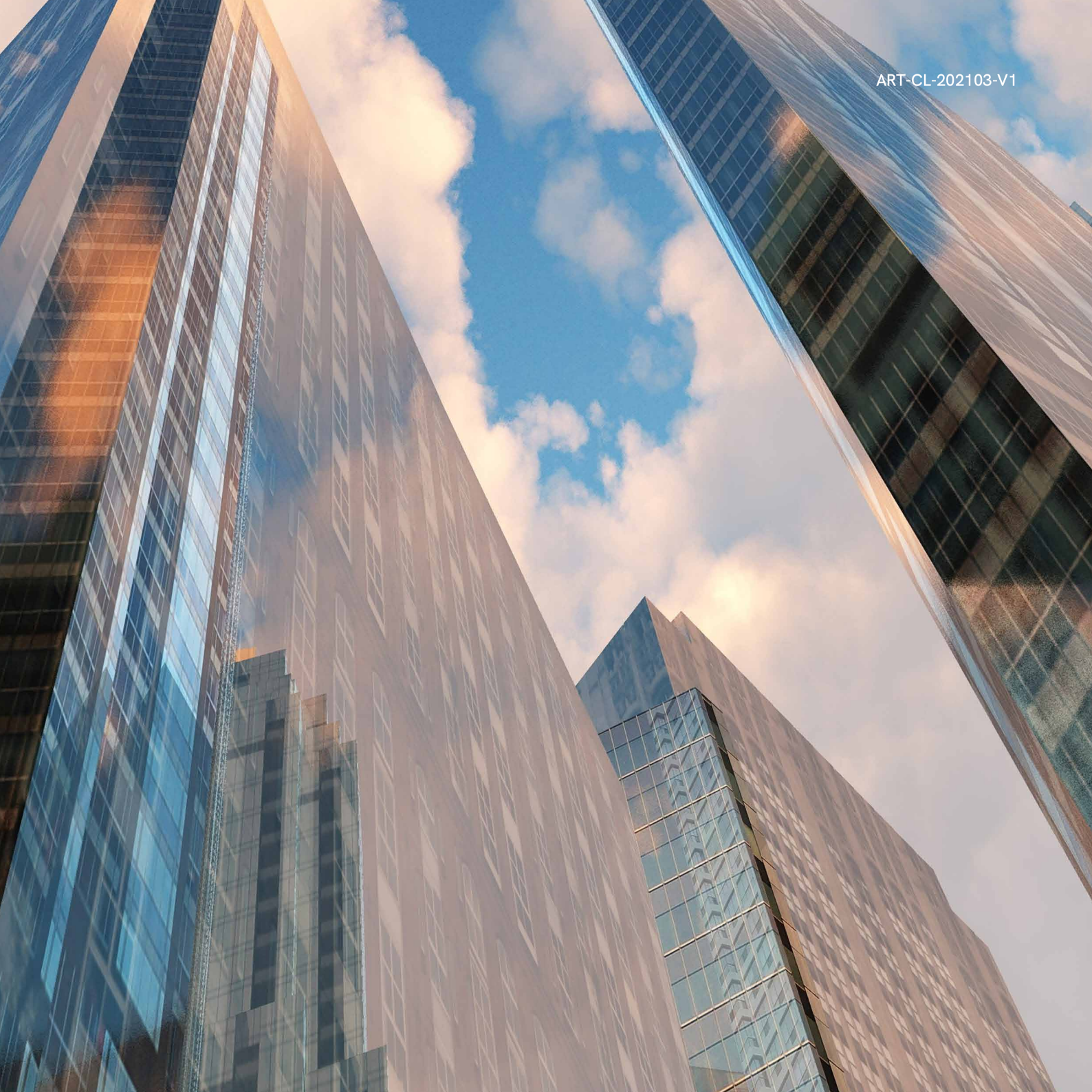
Unser vielseitiges Angebot umfasst Beratung in der Architektur, Anwendungsentwicklung, Qualitätssicherung, DevOps-Betriebsmodelle, API-Management bis hin zu Cloud-Technologien und künstlicher Intelligenz. Doch wir gehen weiter als nur die Bereitstellung technischer Expertise. Wir stellen sicher, dass Agilität in den Unternehmen nicht nur ein Schlagwort bleibt, sondern zur gelebten Realität wird – und das von der Softwareentwicklung bis zum operativen Betrieb.

Unser Herz schlägt für die DevOps-Kultur, die darauf abzielt, technische und organisatorische Hindernisse aus dem Weg zu räumen. Wir machen relevante Informationen verfügbar, die zur Wertschöpfung beitragen, und stärken die kontinuierliche Weiterentwicklung der Teams. Unsere Arbeit ist geprägt von einem interdisziplinären Ansatz, bei dem wir die Erfahrungen und Perspektiven aller Beteiligten berücksichtigen. In gemeinsamen, iterativen Schritten entwickeln wir eine DevOps-Kultur, die perfekt zum individuellen Unternehmensumfeld passt.

So unterstützen wir unsere Kunden nicht nur dabei, sich den Herausforderungen der digitalen Transformation zu stellen, sondern vor allem darin, sie aktiv zu gestalten. Wir sind überzeugt, dass unsere Leidenschaft für Technologie sie voranbringen wird.
Lassen Sie uns gemeinsam eine Zukunft gestalten, die so dynamisch und innovativ ist wie die digitale Welt, in der wir leben.

Mit Leidenschaft, mit Expertise – mit ARS.

ARS Computer und Consulting GmbH
Garmischer Str. 7, 80339 München ▪ T +49 89 324 68-0 ▪ modernize@ars.de ▪ www.ars.de


ART-CL-202103-V1