INTEGRATION UND VERWALTUNG VON IOT-GERÄTEN MIT AZURE IOT OPERATIONS

IoT neu eingebunden

Wie sich das neue Azure IoT Operations von bestehenden Azure-Diensten unterscheidet, welche Technologien dabei zum Einsatz kommen und wann sich der Umstieg lohnt.

Das Internet der Dinge (IoT) entwickelt sich zu einem entscheidenden Baustein für datengetriebene Geschäftsmodelle. Immer mehr Unternehmen vernetzen Maschinen, Sensoren und Geräte, um Betriebsdaten in Echtzeit zu erfassen, zu analysieren und daraus direkt Maßnahmen abzuleiten. Mit dem Aufkommen von künstlicher Intelligenz, vor allem generativer KI, wächst der Bedarf an zuverlässiger und flexibler Konnektivität sowie belastbaren Datenquellen. Nur wenn Daten aus der physischen Welt effizient erfasst und dauerhaft gespeichert werden können, kann der Grundstein für moderne AI-Anwendungen mit echtem Mehrwert gelegt werden.

Die bisherigen Azure-IoT-Services wie Azure IoT Hub und Azure IoT Edge bieten eine solide Basis für viele IoT-Szenarien. Sie ermöglichen es, sowohl kleine Edge-Geräte als auch leistungsfähigere Systeme anzubinden und Daten zentral in der Cloud weiterzuverarbeiten. Dabei unterstützen sie gängige Protokolle wie MQTT, allerdings oft nicht in vollem Funktionsumfang oder mit aktuellen Protokollversionen.

Ein weiteres Problem ist die begrenzte Skalierbarkeit. Azure IoT Edge ist nicht darauf ausgelegt, riesige Datenmengen effizient zu verarbeiten oder hochverfügbare Architekturen am Rand des Netzwerks abzubilden. Besonders in Szenarien, in denen Daten lokal kritisch sind und jederzeit verfügbar sein müssen, stößt das klassische Modell an Grenzen.

Auch das zentrale Nachrichtenrouting über den Azure IoT Hub gerät schnell zum Engpass. Das liegt an der Latenz, der Abhängigkeit von Cloud-Konnektivität und den Kosten. Wer etwa Events lieber direkt an Azure Event Hub oder Event Grid senden möchte, steht vor dem Problem, dass es keine direkte Device Identity oder Berechtigungssteuerung gibt. Diese Funktionen sind tief im Azure IoT Hub verankert, fehlen aber außerhalb.

Genau hier setzt Azure IoT Operations an: Statt einen Dienst wie den Azure IoT Hub zu ersetzen, verfolgt Azure IoT Operations einen modularen Ansatz. Es kombiniert bewährte Azure-Services mit offenen Standards und bringt sie über Azure Arc und Kubernetes näher an die Edge. Der Fokus liegt nicht auf einer All-in-one-Lösung, sondern auf Flexibilität, Skalierbarkeit und Zukunftssicherheit.

Was diesen neuen Ansatz ausmacht, welche Dienste involviert sind, wie eine mögliche Migration aussieht und wann diese überhaupt sinnvoll ist, soll nun geklärt werden.

Sie möchten gerne mehr erfahren?

Das freut uns, denn ab Erscheinen des vorliegenden Artikels veröffentlichen wir auf unserer Webseite in unregelmäßigen Abständen zusätzlich Beiträge, die den Artikel fortsetzen und ausschließlich online zur Verfügung stehen. Derzeit geplant sind Beiträge zu Themen wie:

- **Architektur und Inbetriebnahme von Azure IoT Operations mit OPC UA**
  Einrichtung der Plattform, Anbindung industrieller Anlagen und erste Schritte mit dem OPC UA Connector
- **Entwicklung und Deployment eigener Module für Azure IoT Operations**
  Design, Umsetzung und Bereitstellung containerisierter Anwendungen auf Kubernetes für den Einsatz an der Edge
- **Datenintegration in Azure: Von der Edge bis zu Microsoft Fabric**
  Data Flows in Aktion: Aufbau robuster Datenpipelines zur Analyse, Visualisierung und Weiterverarbeitung in der Cloud

Was ist Azure IoT Operations?

Azure IoT Operations ist ein eigenständiger Service, der eine moderne Alternative zu bestehenden IoT-Lösungen wie Azure IoT Hub und Azure IoT Edge bietet [1]. Die Plattform bringt ein eigenes Benutzerinterface mit, das in seiner Bedienung ein wenig an Azure IoT Central erinnert (siehe Bild 1). Darüber lassen sich Geräte registrieren, eine Asset-Hierarchie aufbauen und Datenflüsse konfigurieren, ohne dass man direkt auf die Geräte zugreifen muss.

IoT Operations besitzt eine Architektur, die sowohl Services in Azure als auch auf der Edge anbietet (siehe Bild 2). Ein zentrales Element in der Verwaltung von IoT-Geräten innerhalb von Azure IoT Operations ist die Device Registry in Azure. Jedes registrierte Gerät wird dabei als eigene Azure-Ressource behandelt, was die Anwendung von Azure Resource Based Access Control (RBAC) ermöglicht. Dies erlaubt eine fein granulare Berechtigungssteuerung, sodass beispielsweise ein Gerät gezielt für das Lesen oder Schreiben von Daten in andere Azure-Services autorisiert werden kann.

Die Device Registry dient als zentrale Anlaufstelle für die Verwaltung von Geräten und Assets, sowohl in der Cloud als auch auf der Edge. Sie speichert Informationen über Assets, einschließlich ihrer Metadaten und Verbindungsinformatio-

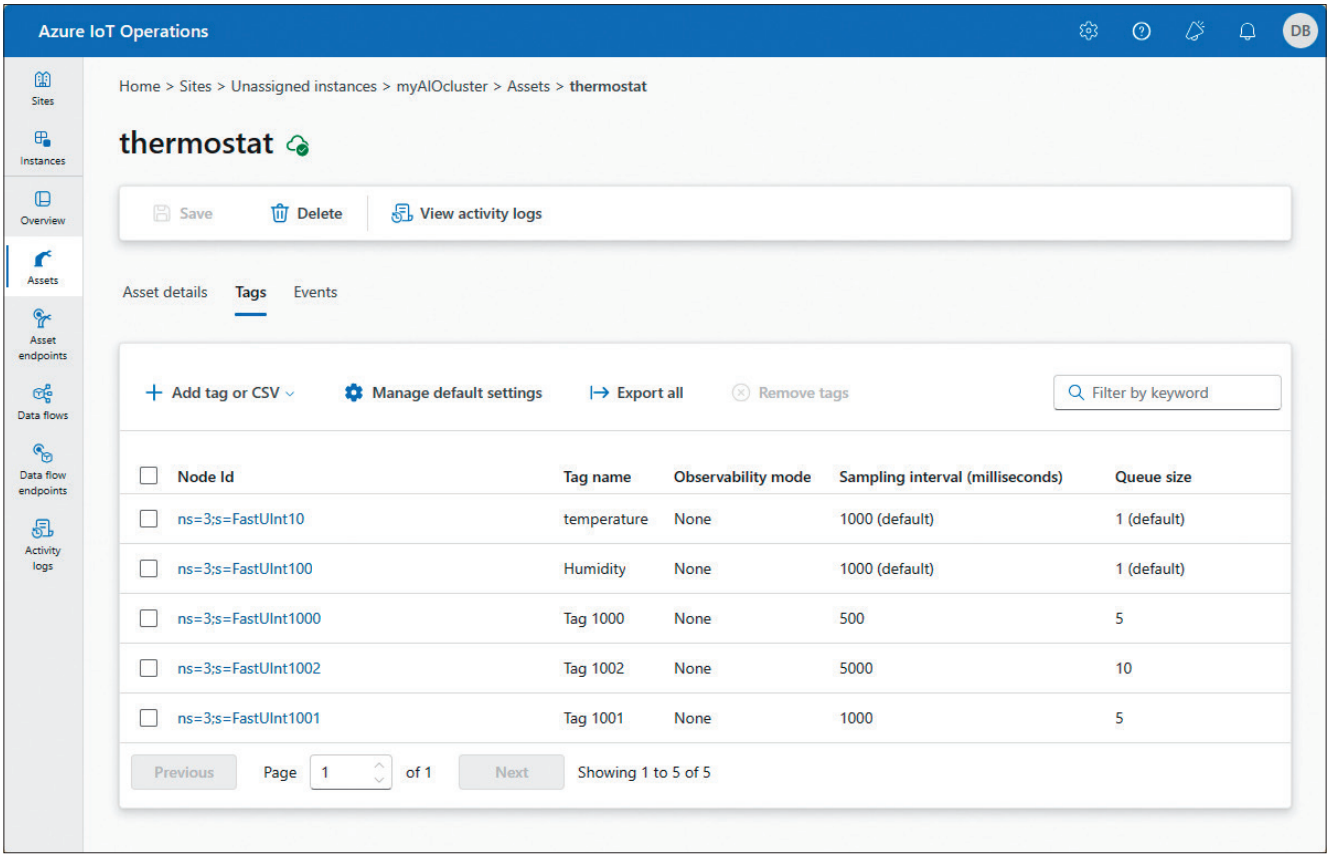


Bild: Microsoft [2]

Die Portalseite von IoT Operations (Bild 1)

nen, und ermöglicht die Nutzung von Tools wie dem Azure Resource Manager zur Verwaltung dieser Assets [2].

Für spezifischere Anforderungen können auch benutzerdefinierte Rollen erstellt werden, die genau die benötigten Berechtigungen enthalten. Durch die Integration von Microsoft Entra ID können zudem Identitäten zentral verwaltet und Zugriffsrichtlinien konsistent durchgesetzt werden. Dies erhöht die Sicherheit, da keine sensiblen Daten im Code oder in Konfigurationsdateien gespeichert werden müssen.

Azure IoT Operations nutzt Kubernetes-Cluster als technische Grundlage, wobei diese Cluster sowohl auf einzelnen Edge-Geräten als auch auf umfassenden Installationen in lokalen Rechenzentren betrieben werden können. Aktuell werden hier Kubernetes-Implementierungen wie K3s oder AKS Edge Essentials unterstützt [3].

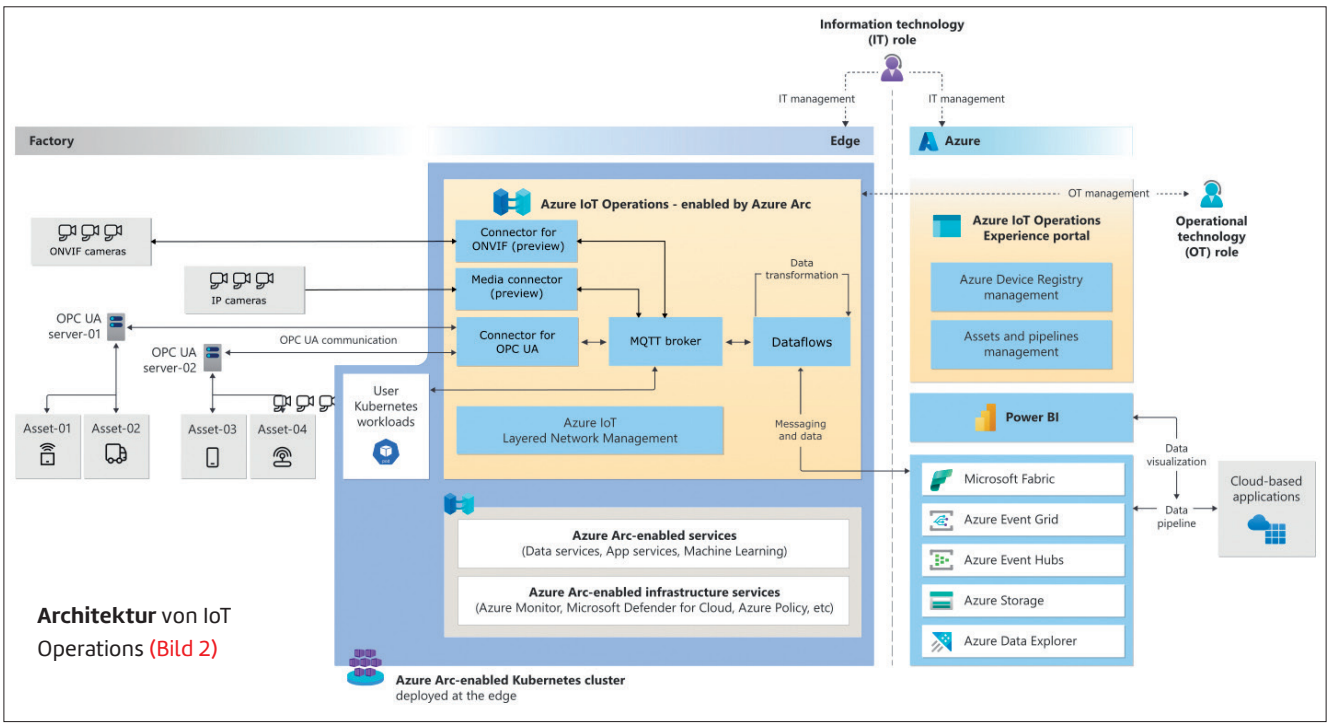
Die Hardwareanforderungen sind höher als bei IoT Edge. Mindestens vier virtuelle CPUs und 16 GB RAM sind nötig. Das macht IoT Operations aktuell ungeeignet für kleine Geräte mit begrenzten Ressourcen. Für einfache oder ressourcenarme Szenarien sind IoT Hub und IoT Edge nach wie vor eine gute Wahl und ein solider Einstieg. Beide bleiben weiterhin verfügbar und werden aktiv weiterentwickelt. Die Verwaltung dieser Cluster erfolgt über Azure Arc, wodurch lokale oder Cloud-fremde Ressourcen wie virtuelle Maschinen oder Datenbanken in Azure integriert und zentral verwaltet werden können.

Anstatt eine eigene Runtime wie bei IoT Edge zu entwickeln, setzt Microsoft in Azure IoT Operations auf offene, Cloud-native-Standards. Kubernetes übernimmt die Orchestrierung der Workloads und ermöglicht so eine flexible und skalierbare Bereitstellung von Anwendungen und Diensten. Durch die Integration von Azure Arc können Unternehmen ihre Edge- und Cloud-Ressourcen nahtlos verbinden und von einer einheitlichen Verwaltungsoberfläche profitieren.

Diese Architektur ermöglicht es, Daten direkt auf der Edge zu erfassen und in Echtzeit zu verarbeiten, was für Anwendungen mit niedriger Latenz und hoher Datenintensität von entscheidender Bedeutung ist. Gleichzeitig können die gesammelten Daten in die Cloud übertragen werden, um von dort aus weiter analysiert und genutzt zu werden. Durch die Nutzung von Kubernetes und Azure Arc bietet Azure IoT Operations eine robuste und flexible Lösung für die Anforderungen moderner IoT-Umgebungen.

Ein lokaler MQTT-Broker, ebenfalls von Microsoft bereitgestellt, übernimmt die Kommunikation in Azure IoT Operations. Dieser Broker ist skalierbar, hochverfügbar und Kubernetes-nativ, was ihn ideal für ereignisgesteuerte Architekturen macht. Durch die Unterstützung der MQTT-Versionen 3.1.1 und 5 ermöglicht er eine flexible und effiziente Kommunikation zwischen Geräten und der Cloud [4].

Die Integration dieses standardisierten Brokers in bestehende Architekturen erleichtert die Implementierung von IoT-Lösungen erheblich. Unternehmen können so ihre vorhandenen Systeme nahtlos mit Azure IoT Operations verbinden und von den Vorteilen einer offenen und skalierbaren Plattform profitieren. Dies fördert die Interoperabilität und reduziert den Integrationsaufwand. ▶



Architektur von IoT Operations (Bild 2)

Bild: Microsoft [1]

Azure IoT Operations nutzt bewährte Komponenten für die Anbindung industrieller Systeme. Der OPC UA Connector, basierend auf dem bekannten OPC Publisher, wurde optimiert, um eine höhere Robustheit und Skalierbarkeit zu gewährleisten [5]. Er stellt sicher, dass Änderungen in OPC-UA-Daten zuverlässig erfasst werden und keine Informationen verloren gehen. Zusätzlich ermöglicht das Modul Akri die automatische Erkennung neuer Assets, Sensoren und Server im Netzwerk, was die Integration und Verwaltung von Geräten erleichtert.

Sobald die Daten im System erfasst sind, können sie entweder lokal verarbeitet oder weitergeleitet werden. Über verschiedene Schnittstellen zur Weiterleitung von Daten in nachgelagerte Systeme, sogenannte Northbound-Connectoren, lassen sich Informationen beispielsweise direkt in Azure Data Explorer speichern oder an Event Grid senden. Event Grid ist ein vollständig verwalteter MQTT-Broker in der Cloud, der Nachrichten effizient weiterverarbeiten kann. Im Vergleich zum IoT Hub bietet Event Grid zudem kosteneffizientere Nachrichtenverarbeitungsoptionen.

Ein weiterer Vorteil von Azure IoT Operations sind die sogenannten Data Flows [6]. Diese ermöglichen es, Daten bereits vor der Übertragung in die Cloud zu filtern, zu transformieren oder zu aggregieren. So können beispielsweise nur vorverarbeitete Events an Event Grid gesendet werden, während Rohdaten in einen Data Lake geschrieben werden. Dies reduziert die Datenmenge, die in die Cloud übertragen wird, und optimiert die Nutzung von Cloud-Ressourcen.

Die Integration in Microsoft Fabric und OneLake ist ebenfalls möglich, wodurch Analyse- und Visualisierungspfade direkt eingebunden werden können. OneLake fungiert dabei als einheitlicher Data Lake, der Daten aus verschiedenen Quellen zusammenführt und eine zentrale Anlaufstelle für Datenanalysen bietet. Durch die Kombination von Azure IoT Operations mit Microsoft Fabric können Unternehmen Echtzeit-Intelligenz nutzen und von erweiterten Analysefunktionen profitieren.

Eigene Module lassen sich, wie bei IoT Edge, in Containern kapseln und über den Kubernetes-Cluster bereitstellen. Die Kommunikation mit dem lokalen MQTT-Broker erfolgt direkt, Authentifizierung und Autorisierung werden über die Kubernetes-Deployment-Konfiguration geregelt.

Wer IoT Operations testen möchte, findet passende Anleitungen auf Microsoft Learn. Der erforderliche Kubernetes-Cluster kann lokal oder über Azure Codespaces in der Cloud bereitgestellt werden. Zwar ist IoT Operations inzwischen offiziell verfügbar und produktiv einsetzbar, aber die Plattform befindet sich weiterhin in aktiver Entwicklung. Änderungen an der Dokumentation sind daher nicht ungewöhnlich.

Vor- und Nachteile gegenüber Azure IoT Hub und IoT Edge

Azure IoT Operations bietet im Vergleich zu IoT Hub und IoT Edge erhebliche Verbesserungen in Bezug auf die Unterstützung offener Standards und die Skalierbarkeit. Während MQTT vom IoT Hub nur in älteren Versionen mit eingeschränktem Funktionsumfang unterstützt wird [8], ermöglicht IoT Operations die Nutzung aktueller MQTT-Versionen mit vollständiger Funktionalität.

Zudem adressiert IoT Operations die Skalierbarkeits- und Ausfallsicherheitsherausforderungen, die bei der Verwendung von IoT Edge auftreten können. Durch die Implementierung auf einem Kubernetes-Cluster können Ressourcen effizienter skaliert und besser verwaltet werden. Dies führt zu einer höheren Flexibilität und Zuverlässigkeit in der Gerätekommunikation und -verwaltung.

Azure IoT Operations bietet eine moderne, modulare und skalierbare Architektur, die auf Kubernetes basiert und in Azure Arc für die Verwaltung von Edge-Umgebungen integriert ist. Durch die Nutzung offener Standards wie MQTT und OPC UA ermöglicht die Plattform eine flexible und transparente Integration, auch mit Modulen von Drittanbietern. Dies erleichtert den Einstieg, da man mit einem lokalen Server beginnen und später horizontal auf mehrere Server skalieren kann. Die Orchestrierung der Container über Kubernetes sorgt für eine erhöhte Resilienz der Umgebung.

Daten lassen sich lokal vorverarbeiten, etwa durch Filtern oder Aggregieren, und anschließend direkt in Azure Data Explorer, Microsoft Fabric oder OneLake übertragen, ohne den Umweg über den IoT Hub. Dies spart Kosten und reduziert die Komplexität der Architektur (siehe Bild 3).

Der neue OPC-UA-Connector und der lokale MQTT-Broker sind skalierbar sowie fehlertolerant und gewährleisten, dass keine Nachrichten verloren gehen. Zusätzlich ermöglicht das Akri-Modul die automatische Erkennung neuer Assets, Sensoren und Server im Netzwerk. Diese Funktionen erweitern die Möglichkeiten von Azure IoT Operations und bieten Vorteile, die über die bisherigen Funktionen von IoT Edge hinausgehen. Die stärkere Öffnung hin zu Open Source und Standardisierung schützt langfristig vor einem Lock-in und erleichtert die Integration in bestehende Systeme.

Die Einführung von Azure IoT Operations kann mit erhöhten Hardwareanforderungen verbunden sein. Im Vergleich zu Azure IoT Edge, das auf Geräten wie dem Raspberry Pi 3 betrieben werden kann, stellt IoT Operations höhere Anforderungen an die Hardware. Diese Anforderungen machen IoT Operations weniger geeignet für kleine Embedded Devices und erfordern aktuell noch leistungsfähigere Serverhardware.

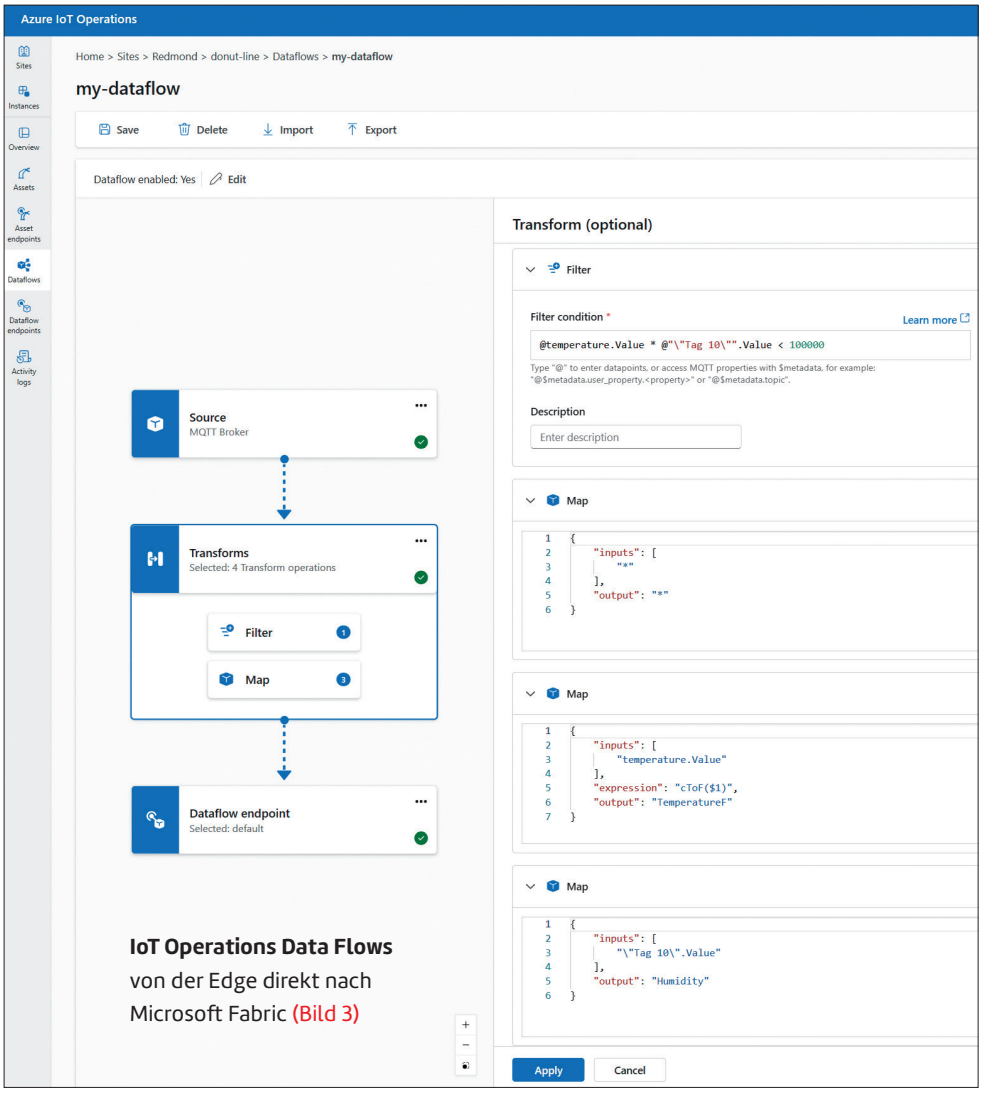
Die Einrichtung eines Kubernetes-Clusters ist nicht trivial und verlangt fundierte Kenntnisse in der Containerorchestrierung. Obwohl Azure Arc viele Verwaltungsaufgaben vereinfacht, bleibt der Betrieb eines solchen Clusters anspruchsvoll und setzt entsprechende Fachkenntnisse voraus.

Das Preismodell von Azure IoT Operations basiert auf einem nutzungsabhängigen Ansatz, bei dem pro Kubernetes-Node abgerechnet wird. Dies kann insbesondere bei der Skalierung auf mehrere Nodes zu höheren Kosten führen. Zudem sind derzeit noch keine Informationen zu möglichen Begrenzungen bei der Anzahl angebundener OPC-UA-Server oder Data Flows verfügbar, was zukünftige Planungen erschwert.

Die Integration von IoT Operations erfordert oft eine Anpassung oder Neuentwicklung bestehender Anwendungen, da die Architektur und Funktionsweise von IoT Operations von derjenigen von IoT Edge abweicht. Dieser Migrationsaufwand sollte bei der Planung berücksichtigt werden.

Wenn Unternehmen derzeit erfolgreich mit Azure IoT Hub und IoT Edge arbeiten, kleine Geräte einsetzen oder keine neuen Funktionen benötigen, ist eine sofortige Migration zu Azure IoT Operations nicht zwingend erforderlich. Beide Plattformen gelten laut Microsoft als „feature complete“ und erhalten weiterhin Support sowie Sicherheitsupdates.

Werden jedoch zukünftig Anforderungen an Skalierbarkeit, Ausfallsicherheit, eine modulare Cloud-native Architektur, offene Standards oder eine Erweiterung in Richtung Data Analytics oder Microsoft Fabric gestellt, erscheint eine Migration zu Azure IoT Operations mittelfristig sinnvoll. ▶



IoT Operations Data Flows von der Edge direkt nach Microsoft Fabric (Bild 3)

Bild: Microsoft [9]

Für neue Projekte sollte Azure IoT Operations direkt in Betracht gezogen werden, um von den modernen Architekturansätzen und der Integration mit aktuellen Azure-Diensten zu profitieren. Bei Bestandsprojekten können ein schrittweiser Umstieg oder ein Hybridansatz sinnvolle Vorgehensweisen sein. Dabei ist zu beachten, dass es derzeit keinen direkten Migrationspfad von IoT Edge zu IoT Operations gibt, da die beiden Plattformen unterschiedliche Architekturen aufweisen.

Migrationspfade und Szenarien

Die Migration von IoT-Anwendungen von IoT Hub und IoT Edge zu IoT Operations kann in mehreren Schritten erfolgen, abhängig von den genutzten Funktionen und Modulen. Wenn der IoT Hub hauptsächlich als MQTT-Broker verwendet wird, kann die Migration relativ einfach auf Event Grid umgestellt werden. Event Grid bietet eine nahtlose Integration mit anderen Azure-Diensten und übernimmt das Nachrichtenrouting. Allerdings müssen Geräte neu konfiguriert werden, und es ist ein anderer Authentifizierungsmechanismus erforderlich.

Die Migration von IoT-Anwendungen von IoT Hub und IoT Edge zu Azure IoT Operations erfordert eine sorgfältige Planung, insbesondere hinsichtlich Funktionen wie Device Twins, Device Streams und Device Updates.

Die Device-Twin-Funktionalität, die in IoT Hub als JSON-Dokumente vorliegt und Gerätedaten wie Metadaten, Konfigurationen und Zustände speichert, ist in IoT Operations nicht direkt integriert. Daher ist es notwendig, entweder eine eigene Datenbank zur Verwaltung und Synchronisation von Device Twins aufzubauen, einen Caching-Layer wie Redis zu implementieren oder auf MQTT Retained Messages zurückzugreifen. Dies erfordert zusätzliche Entwicklungsarbeit, um die Funktionalitäten, die zuvor über den IoT Hub bereitgestellt wurden, in IoT Operations nachzubilden. Für Methodenaufrufe ist MQTT Version 5 zu empfehlen, da hier Remote Procedure Calls (RPCs) unterstützt werden.

Ein weiterer Aspekt ist die Funktion der Device Streams, die in IoT Hub die Erstellung sicherer bidirektionaler TCP-Tunnel für verschiedene Cloud-zu-Gerät-Kommunikationsszenarien ermöglichen. Diese Funktion ist in IoT Operations derzeit nicht verfügbar. Für ähnliche Anforderungen müssen alternative Lösungen entwickelt oder bestehende Kommunikationsprotokolle angepasst werden.

Die Device-Update-Funktion von IoT Hub ermöglicht das Bereitstellen von Over-the-Air-Updates für IoT-Geräte. In IoT Operations ist diese Funktionalität nicht direkt integriert. Für die Verwaltung von Geräteaktualisierungen müssen entweder eigene Mechanismen entwickelt werden, oder man greift

Listing 1: Abstrahieren des Azure IoT Hub Device SDK

```
public class ModuleClient : IModuleClient
{
  private Microsoft.Azure.Devices.Client.ModuleClient
    _moduleClient;

  public ModuleClient(Microsoft.Azure.Devices.Client.
    ModuleClient moduleClient) => _moduleClient =
    moduleClient;

  public System.Threading.Tasks.Task OpenAsync() =>
    _moduleClient.OpenAsync();
  public System.Threading.Tasks.Task OpenAsync(
    System.Threading.CancellationToken
    cancellationToken) =>
    _moduleClient.OpenAsync(cancellationToken);
  public System.Threading.Tasks.Task CloseAsync() =>
    _moduleClient.CloseAsync();
  public System.Threading.Tasks.Task CloseAsync(
    System.Threading.CancellationToken
    cancellationToken) =>
    _moduleClient.CloseAsync(cancellationToken);
  public System.Threading.Tasks.Task SendEventAsync(
    System.String outputName, Microsoft.Azure.Devices.
    Client.Message message) =>
    _moduleClient.SendEventAsync(outputName, message);
  ...
}
```

```
public class MqttModuleClient : IModuleClient
{
  ...
  public async Task SendEventAsync(
      string outputName, Message message,
      CancellationToken cancellationToken)
  {
    var topic = string.IsNullOrEmpty(outputName) ?
      _messagesTopic : $"{_messagesTopic}/{outputName}";
    var mqttMessage =
        new MqttApplicationMessageBuilder()
      .WithTopic(topic)
      .WithPayload(message.GetBytes())
      .WithContentType(message.ContentType)
      .WithQualityOfServiceLevel(
        MqttQualityOfServiceLevel.AtLeastOnce)
      .Build();

    foreach (var messageProperty in message.Properties)
    {
      mqttMessage.UserProperties.Add(new MqttUserProperty(
        messageProperty.Key, messageProperty.Value));
    }
    await _mqttClient.EnqueueAsync(mqttMessage);
  }
  ...
}
```

Tabelle 1: Funktionalität von Azure IoT Hub und Azure IoT Edge innerhalb von Azure IoT Operations

Azure IoT Hub und Azure IoT Edge	Azure IoT Operations
Messages (Cloud to Device und Device to Cloud)	Azure Event Grid MQTT
Device Twins	Azure Event Grid MQTT: Retained Messages
Direct Methods	Azure Event Grid MQTT: Remote Procedure Calls
Upload Files	Azure Storage Account (Device Authentication)
Message Routing	Azure Event Grid MQTT: Routing
Device and Module Monitoring	Azure Arc, Azure Monitor und Azure Log Analytics
Device Streams	Azure Arc SSH oder PowerShell
Device Updates	Azure Arc und Azure Update Management
Device Provisioning Service	Aktuell noch keine Alternative
Custom IoT Edge Modules: Deployments	Container Deployments über Kubernetes in Azure Arc
Custom IoT Edge Modules: Messaging und Direct Methods	Lokaler MQTT-Broker
IoT Edge for Linux on Windows (EFLOW)	IoT Operations mit AKS Edge Essentials auf Windows 11 IoT Enterprise

auf Azure Arc in Kombination mit Azure Update Management zurück. Da Azure Arc hinter IoT Operations steckt, kann man jedoch davon ausgehen, dass in Zukunft auch weiteres Management der Server hinter einem Cluster in IoT Operations integriert wird.

Der OPC Publisher, der in IoT Edge als Modul läuft, ermöglicht die Kommunikation mit OPC-UA-Servern und das Publizieren von Telemetriedaten an Azure IoT Hub. In IoT Operations übernimmt der Connector für OPC UA diese Funktionalität. Er verbindet sich mit OPC-UA-Servern, überwacht Datenänderungen und Ereignisse und publiziert die Daten im MQTT-PubSub-Format. Allerdings fehlen derzeit Funktionen wie automatische Geräteerkennung (Discovery) und Methodenaufrufe, die für eine vollständige Migration erforderlich wären. Es ist jedoch zu erwarten, dass diese Funktionen in zukünftigen Versionen nachgerüstet werden.

Für die Migration eigener Module von IoT Edge zu IoT Operations ist es wichtig, die Kommunikation von der bisherigen Implementierung über das IoT Edge Client SDK auf eine direkte MQTT-Kommunikation umzustellen. Dabei können MQTT Version 5 und Remote Procedure Calls (RPC) genutzt werden, um Funktionen wie direktionales Messaging, Device-Twin-Aktionen und Methodenaufrufe abzubilden. Microsoft bietet hierfür SDKs an, die die Kommunikation über MQTT 5 mit RPC-Unterstützung ermöglichen.

Ein sinnvoller Ansatz besteht darin, die bestehende SDK-Implementierung so zu abstrahieren, dass ein späterer Austausch gegen eine MQTT-basierte Implementierung erleichtert wird (siehe Listing 1). Dies ermöglicht eine schrittweise Migration und reduziert den Aufwand für die Anpassung bestehender Anwendungen.

Die Migration von IoT Edge zu IoT Operations ist ein komplexer Prozess, der sorgfältige Planung und Anpassung erfordert. Ein zentraler Aspekt ist die Skalierbarkeit eigener Module auf Kubernetes. Anwendungen sollten zustandslos sein oder ihren Zustand extern, beispielsweise in einer Datenbank, speichern. Feste Hardwarebindungen, spezifische Annahmen über Netzwerkadressen oder blockierende Hintergrundprozesse können die Skalierbarkeit und Flexibilität beeinträchtigen. Kubernetes ermöglicht die horizontale Skalierung von Modulen, was die Ausfallsicherheit erhöht, vorausgesetzt, die Anwendungen sind entsprechend konzipiert.

Ein weiterer zu berücksichtigender Faktor ist die zusätzliche Komplexität durch Kubernetes. Obwohl Kubernetes viele Vorteile bietet, bringt es eine steile Lernkurve mit sich, die Entwicklerinnen und Entwickler vor Herausforderungen stellen kann. Dennoch kann das erworbene Wissen über Kubernetes langfristig von Vorteil sein, insbesondere da Azure in Diensten wie AKS ebenfalls Kubernetes nutzt.

Es ist wichtig zu beachten, dass IoT Edge und IoT Operations nicht parallel auf derselben Infrastruktur betrieben werden können. Ein Übergang erfordert daher entweder den Einsatz unterschiedlicher Server oder eine sorgfältige Planung der Migration. Um den Übergang schrittweise vorzubereiten, empfiehlt es sich, bereits vorab die Module auf MQTT umzustellen und hinter einer Abstraktionsebene zu betreiben. Auch ein vollständiger Umstieg auf MQTT mit einem eigenen Broker auf der Edge, etwaMosquitto, kann in Betracht gezogen werden. Innerhalb der Cloud können Nachrichten vom IoT Hub in Richtung Event Grid geroutet werden, um den Weg für eine spätere vollständige Migration zu ebnen.

Bei der Migration bestehender Konnektoren, wie der Konfiguration von OPC UA, ist Vorsicht geboten. Die Konfiguration von OPC-UA-Endpoints, -Nodes und -Zertifikaten wird häufig vom Container in das Host-Dateisystem gemappt und dort in spezifischen Formaten gespeichert. Dieses Mapping im neuen OPC-UA-Connector korrekt umzusetzen und gleichzeitig ein kompatibles Format der Konfiguration ▶

bereitzustellen, kann herausfordernd sein und erfordert sorgfältige Planung.

Eine kleine Zusammenfassung der Funktionalitäten von Azure IoT Hub und Azure IoT Edge, die sich in Azure IoT Operations wiederfinden, zeigt Tabelle 1.

Die Entscheidung, von Azure IoT Hub und IoT Edge auf Azure IoT Operations zu migrieren, sollte wohlüberlegt sein. IoT Operations bietet Vorteile wie Skalierbarkeit, Resilienz und die Nutzung offener Standards. Wer jedoch mit den aktuellen Funktionen von IoT Hub und IoT Edge zufrieden ist und keine unmittelbaren Anforderungen an neue Features hat, kann die Migration auf IoT Operations noch hinauszögern.

Eine schrittweise Umstellung und die Vorbereitung auf die neuen Standards sind jedoch ratsam, um langfristig von den Vorteilen der Plattform zu profitieren. Bevor man den Schritt zur Migration geht, ist es wichtig, sich intensiv mit dem aktuellen Funktionsstand, dem Pricing und einer passenden Migrationsstrategie auseinanderzusetzen. Da IoT Operations aktuell noch intensiv weiterentwickelt wird, lohnt es sich außerdem, die Entwicklung fortlaufend im Blick zu behalten. Langfristig kann die Plattform durch ihre Architektur und Integration in die Azure-Welt eine solide Grundlage für weiterführende Anwendungsfälle wie KI und Analytics schaffen.

Die Preisgestaltung für Azure IoT Operations basiert auf einem nutzungsbasierten Modell. Abgerechnet wird nach der Anzahl der Kubernetes-Knoten in einem Azure-Arc-fähigen Kubernetes-Cluster, auf dem Azure IoT Operations installiert ist. Die Abrechnung erfolgt stündlich, wobei die ersten 30 Tage der Nutzung als Testphase gelten und nicht berechnet werden. Zusätzlich ist die Azure Device Registry erforderlich, um Geräte zu verwalten, was ebenfalls kostenpflichtig ist. Die genauen Preise können auf der offiziellen Azure-Website eingesehen werden.

Für eine detaillierte Einschätzung der Kosten und eine fundierte Entscheidungsfindung empfiehlt es sich, den Azure-Preisrechner unter [10] zu nutzen. Dieser ermöglicht eine individuelle Kalkulation basierend auf den spezifischen Anforderungen und Nutzungsszenarien.

Fazit

IoT Operations bietet einen vielversprechenden Ausblick auf die zukünftige Entwicklung von IoT unter Azure. Durch den konsequenten Einsatz offener Standards im Cloud-native- und Industrieumfeld zeigt Microsoft, dass bestehende Lösungen unterstützt und in ein größeres Ökosystem eingebunden werden sollen. Anstatt IoT als eigenständigen Azure Service zu betrachten, wird es als Teil eines umfassenderen Ansatzes zur Bewältigung moderner IoT-Anwendungsfälle positioniert. Die Plattform ermöglicht eine flexible und skalierbare Integration von Azure-Diensten und bringt dank Azure Arc und Kubernetes wichtige Funktionen näher an die Edge, sodass Unternehmen ihre IoT-Architekturen zukunftssicher gestalten können.

Für Unternehmen, die auf Skalierbarkeit, Ausfallsicherheit und Robustheit angewiesen sind, bietet IoT Operations konkrete Vorteile. Gleichzeitig ist zu bedenken, dass jene, die aktuell ohne Einschränkungen mit IoT Hub und IoT Edge arbeiten oder deren Anwendungen primär auf kleinere Geräte ausgelegt sind, den Wechsel noch hinauszögern können. Die kontinuierliche Weiterentwicklung von IoT Operations ermöglicht es jedoch, bei steigenden Anforderungen an Datenanalysen und Künstliche Intelligenz von zusätzlichen Funktionen zu profitieren. Daher sollte insbesondere in neuen Projekten IoT Operations als Option in Betracht gezogen werden, während Bestandsprojekte durch einen schrittweisen Umstieg oder einen Hybridansatz unterstützt werden können.

Es ist jedoch wichtig, dass Unternehmen sich im Vorfeld umfassend mit Themen wie Preisgestaltung, Skalierbarkeit und einer durchdachten Migrationsstrategie auseinandersetzen. Eine Migration zu IoT Operations kann mit zusätzlichen Kosten und Herausforderungen verbunden sein, da bestehende Systeme angepasst werden müssen, um eine nahtlose Integration in die neue Architektur zu ermöglichen. Langfristig kann IoT Operations jedoch eine wertvolle Ergänzung in der Azure-Landschaft darstellen, indem es die Grundlage für fortschrittliche Anwendungsfälle, insbesondere im Bereich Künstliche Intelligenz und Datenanalysen, bildet. Wer diese Entwicklungen frühzeitig in den Blick nimmt und seine Systeme darauf vorbereitet, legt den Grundstein für Innovation und nachhaltigen Geschäftserfolg. ■

[1] Microsoft Learn, What is Azure IoT Operations?, www.dotnetpro.de/SL2506-07AzureIOT1

[2] Microsoft Learn, What is asset management in Azure IoT Operations, www.dotnetpro.de/SL2506-07AzureIOT2

[3] Microsoft Learn, Azure IoT Operations, Deployment details, www.dotnetpro.de/SL2506-07AzureIOT3

[4] Microsoft Learn, Azure IoT Operations built-in local MQTT broker, www.dotnetpro.de/SL2506-07AzureIOT4

[5] Microsoft Learn, Azure IoT Operations, What is the connector for OPC UA?, www.dotnetpro.de/SL2506-07AzureIOT5

[6] Microsoft Learn, Process and route data with data flows, www.dotnetpro.de/SL2506-07AzureIOT6

[7] Microsoft Learn, Quickstart: Run Azure IoT Operations in GitHub Codespaces with K3s, www.dotnetpro.de/SL2506-07AzureIOT7

[8] Microsoft Learn, IoT Hub, Choose a device communication protocol, www.dotnetpro.de/SL2506-07AzureIOT8

[9] Microsoft Learn, Create data flows in Azure IoT Operations, www.dotnetpro.de/SL2506-07AzureIOT9

[10] Azure-Preisrechner, www.dotnetpro.de/SL2506-07AzureIOT10

Florian Bader
ist Solution Architect bei Lunaris Digital Solutions und Microsoft MVP für Azure IoT. Mit langjähriger Erfahrung im Bereich DevOps, Cloud und Digitalisierung unterstützt er Unternehmen bei ihren digitalen Projekten.
florian.bader@lunaris.digital

dnpCode A2506-07AzureIOT