Referenz
Petrik, D.; Mormul, M.; Reimann, P.; Gröger, C.: *Anforderungen für Zeitreihendatenbanken im industriellen IoT*, veröffentlicht in*: IoT - Best Practices*, herausgegeben von Meinhardt, S.; Wortmann, F., S. 339-377, 2021, Springer Vieweg. Vervielfältigt mit Genehmigung von Springer Fachmedien Wiesbaden. Die finale authentifizierte Version ist online verfügbar unter:
https://doi.org/10.1007/978-3-658-32439-1_19

1 Anforderungen für Zeitreihendatenbanken im industriellen IoT[1]

Dimitri Petrik, Mathias Mormul, Peter Reimann, Christoph Gröger

Zusammenfassung

Das industrielle Internet der Dinge (IIoT) integriert Informations- und Kommunikationstechnologien in industrielle Prozesse und erweitert sie durch Echtzeit-Datenanalyse. Hierbei sind sensorbasierte Zeitreihen ein wesentlicher Typ von Daten, die in der industriellen Fertigung generiert werden. Sensorbasierte Zeitreihendaten werden in regelmäßigen Abständen generiert und enthalten zusätzlich zum Sensorwert einen Zeitstempel. Spezielle Zeitreihen-Datenbanken (eng.: Time Series Databases (TSDB) sind dafür ausgelegt, Zeitreihendaten effizient zu speichern. Wenn TSDBs maschinennah, d. h. in der industriellen Edge, eingesetzt werden, sind Maschinendaten zur Überwachung zeitkritischer Prozesse aufgrund der niedrigen Latenz schnell verfügbar, was die erforderliche Zeit für die Datenverarbeitung reduziert. Andererseits können TSDBs auch in den Data Lakes als skalierbaren Datenplattformen zur Speicherung und Analyse von Rohdaten zum Einsatz kommen, um die langfristige Vorhaltung von Zeitreihendaten zu ermöglichen. Bisherige Untersuchungen zu TSDBs sind bei der Auswahl für den Einsatz in der industriellen Edge und im Data Lake nicht vorhanden. Die meisten verfügbaren Benchmarks von TSDBs sind performanceorientiert und berücksichtigen nicht die Randbedingungen einer industriellen Edge oder eines Data Lake. Wir adressieren diese Lücke und identifizieren funktionale Kriterien für den Einsatz von TSDBs in diesen beiden Umgebungen und bilden somit einen qualitativen Kriterienkatalog. Des Weiteren zeigen wir am Beispiel von InfluxDB, wie dieser Katalog verwendet werden kann, mit dem Ziel die systematische Auswahl einer passenden TSDB für den Einsatz in der Edge und im Data Lake zu unterstützen.

Time Series Data, Zeitreihendaten, Zeitreihendatenbanken, Time Series Database, Industrial IoT, Edge Computing, Data Lake, Kriterienkatalog, InfluxDB

1.1 Ausgangssituation und Problemstellung

Im Zuge der Digitalisierung und dem Industriellen Internet der Dinge (IIoT) versuchen Unternehmen moderne IuK- und Software-Technologien in der industriellen Fertigung zu nutzen, um zusätzliche Mehrwerte durch die Erfassung und Verarbeitung von Maschinendaten zu generieren. Begünstigt durch die Reduktion der technologiebezogenen Kosten, werden auch Werkzeugmaschinen und industrielle Systeme zunehmend digitaler. Moderne Werkzeugmaschinen und industrielle Handhabungssysteme verfügen über eine hohe Anzahl von Sensoren zur Überwachung des Maschinenzustands und der industriellen Fertigungsprozesse. Die beispielhafte Spezifikation von DMG Mori listet insgesamt 60

[1] vollständig überarbeiteter und erweiterter Beitrag basierend auf Petrik et al. (2019): Anforderungen für Zeitreihendatenbanken in der industriellen Edge, HMD – Praxis der Wirtschaftsinformatik 56(6), S. 1282-1308.

Sensoren (DMG Mori 2016). Solche Maschinen sind in der Lage konstante Datenströme zu liefern, die z. B. Position der Werkzeugachse, Vorschubgeschwindigkeit, Vibration, Temperatur oder das Geräusch beinhalten (Lenz 2017). Diese für die Überwachung relevanten Daten sind sog. Zeitreihendaten und enthalten gemäß der Definition Sensorwerte (Variable) und äquidistante Zeitstempel (Metcalfe und Cowpertwait 2009). Zusätzliche Domänenspezifika von Zeitreihendaten aus Maschinen in der Fertigung sind häufig hohe Abtastraten zwischen 1 MHz und 1KHz, wodurch insgesamt große Datenmengen erzeugt werden. Prototypische Umsetzungen industrieller Internet of Things (IIoT)-Anwendungen aus der Spritzgussindustrie zeigen, wie 100 überwachte Maschinen eine Gesamtmenge von 8,5 GB Daten pro Stunde erzeugen (Mourtzis 2018). Andererseits hat der Datenalterungsprozess bei zeitkritischen Anwendungsszenarien einen qualitätsmindernden Einfluss auf die erzeugten Daten. Um beispielsweise die erforderliche Präzision bei der Werkzeugverschleißüberwachung und –vorhersage zu gewährleisten, sind kurze Verarbeitungszeiten von unter 100 ms nach der Generierung der Daten erforderlich (Lenz 2017). Diese Einschränkungen werfen die Frage auf, wie Zeitreihendatenmengen effektiv erfasst und verarbeitet werden können.

Typischerweise unterliegt die Datenübertragung aus dem industriellen Shopfloor in zentralisierte (unternehmensinterne oder cloudbasierte) Datenplattformen nach wie vor typischen Bandbreitenbeschränkungen und wird von relativ hohen Latenzzeiten geprägt (Ahmed et al. 2017; Chiang 2016). Zeitkritische Operationen bei der Werkzeugüberwachung wie z. B. das Stoppen der Maschine erfordern eine maschinennahe Speicherung und Verarbeitung der Daten, um die Auswirkungen durch den Datenalterungsprozess zu minimieren. Dabei können Fog- und Edge-Computing-Architekturen helfen, da beide Architekturen ein zentralisiertes System ergänzen und eine maschinennahe Speicherung, Verarbeitung und Steuerung begrenzter Datenmengen ermöglichen (Chiang 2016). Dabei definiert Cisco als Fog Computing die Standards zur Unterstützung von Operationen für Speicher-, Rechen- und Netzwerkdienste zur Umsetzung des Edge Computing Konzepts. Somit ergänzen sich die Edge- und Fog-Computing-Architekturen, wobei die Praxis die beiden Begriffe eher nach dem Ort der Datenverarbeitung unterscheidet. Beim Edge-Computing werden die Daten entweder direkt an der Datenquelle oder von einem speziellen Gerät maschinennah verarbeitet. Beim Fog Computing werden die Daten meist in einer zusätzlichen hierarchischen Ebene zwischen der Datenquelle und einer zentralisierten Datenplattform verarbeitet. So ist die Datenverarbeitung physikalisch weiter von der Datenquelle entfernt als beim Edge Computing (Firouzi et al. 2020). Durch den Einsatz industrieller Edge-Geräte zwischen einem zentralen Datenspeicher und der Maschine können Zeitreihendaten im industriellen Shopfloor zur kurzzeitigen Zwischenspeicherung und Vorverarbeitung vorgehalten werden. Außerdem erhöhen Fog- und Edge-Computing-Architekturen die Skalierbarkeit auf der Shopfloor-Ebene (Corneo und Gunningber 2018). Zusätzlich kann die Analyse der Maschinendaten in einem geschlossenen Netz direkt auf dem Edge-Gerät erfolgen, wenn die Bandbreite der Internetanbindung im Shopfloor nicht ausreicht oder vom Maschinenbetreiber keine Internetanbindung gewünscht ist.
Aktuelle Edge-Geräte sind definiert als High-Level-Mikrocontroller mit geringem Energieverbrauch (Väänänen und Hämäläinen 2018), die auf der x86 oder ARM-Architektur basieren und so konzipiert sind, dass sie genügend Rechenleistung für den Betrieb von Anwendungen für die datenbasierte Prozesssteuerung bereitstellen (Puliafito et al. 2019). Unter einem geeigneten Betriebssystem (OS) wie Linux (Mueller et al. 2017; Chen et al. 2018), kann ein Datenbanksystem zur Speicherung und Verarbeitung der Daten direkt auf dem Edge-Gerät betrieben werden (Martinviita 2018; Oyekanlu 2017). Automatisierungsanbieter wie z. B. Hilscher oder Kunbus bieten bereits marktreife industrielle Edge-Geräte an, die auf modifizierten Versionen des Raspberry Pi basieren (Beck 2019). Somit scheint ein Raspberry Pi ein geeignetes Gerät für eine prototypische Konfiguration eines industriellen Edge-Gerätes für den Betrieb einer Datenbank zu sein. Dennoch ist es fraglich, ob traditionelle relationale Datenbanksysteme geeignet sind, die Geschwindigkeit der erzeugten Zeitreihendaten von Werkzeugmaschinen in der begrenzten Zeit zu verarbeiten (Bader et al. 2017), wenn sie auf Edge-Geräten mit begrenzter Leistung betrieben werden. Zeitreihendatenbanken (TSDBs) und Zeitreihenmanagementsysteme (TSMSs) basieren auf anderen Technologien als relationale Datenbanken, sodass sie speziell für die Speicherung und Abfrage von Zeitreihendaten optimiert sind (Bader et al. 2017; Jensen et al. 2017). In der Forschungsarbeit von Martinviita (2018) werden vier mögliche Anwendungsfälle für die Nutzung von TSDBs im Kontext von IIoT aufgezeigt. Dieser Artikel betrachtet zwei typische Szenarien für den Einsatz von TSDBs im IIoT. Bei dem Edge-Einsatz der TSDB

orientiert sich der Artikel an einem von Martinviita (2018) beschriebenen Anwendungsfall, wobei die TSDB einer hoher parallelen Schreiblast in der Edge ausgesetzt ist. Dieser Anwendungsfall wird in ◉

Abb. 1.1 illustriert und später für die Umsetzung des im ▶ Abschn. 1.3.1 skizzierten Einsatzszenarios verwendet. Dabei wird jeweils ein Edge-Gerät mit Datenbank pro Handhabungssystem betrieben, bevor die Daten in einen zentralen Datenspeicher übermittelt werden. Andere von Martinviita (2018) beschriebene Anwendungsfälle beinhalten zusätzlich ein Supervisory Control and Data Acquisition (SCADA) System zur Steuerung mehrerer Maschinen in einem über eine TSDB verfügenden Produktionscluster, oder berücksichtigen eine Hierarchie aus Datenbanken, die als ein mehrschichtiges System zusammenarbeiten. Aufgrund der Einschränkungen vieler am Markt verfügbaren sowie der für die Evaluation ausgesuchten TSDB in der Open-Source-Version (siehe ▶ Abschn. 1.5.2 – „Verteilter Betrieb") werden diese Anwendungsfälle jedoch in dem vorliegenden Artikel nicht genauer ausgeführt.

Das zweite im Artikel betrachtete Szenario zum TSDB-Einsatz resultiert aus den in der Edge oftmals begrenzten Möglichkeiten für Ansätze zu „Advanced Analytics" (z. B. maschinelles Lernen). Die Grenzen werden durch die verfügbare Rechenleistung eines High-Level-Mikrocontrollers, die Auswahl an verfügbarer Analyse-Software für die Edge sowie durch die fehlenden Integrationsmöglichkeiten für weitere Datenquellen, gesetzt (Patel et al. 2017). Aufgrund dieser Limitierungen können Advanced Analytics nicht in der Edge durchgeführt werden. Stattdessen bietet es sich an, aktuelle Konzepte zur zentralen Speicherung, Verarbeitung und Analyse großer Datenmengen im Rahmen eines Data Lake (siehe ▶ Abschn. 1.2.2) einzusetzen. Ein Data Lake ist eine skalierbare Datenmanagementplattform zur Speicherung, flexibler Verwaltung und Analyse von Daten jeder Art. In einem Data Lake können die Zeitreihendaten in ihrer Rohform vorgehalten werden (Giebler et al. 2020). In der Praxis können TSDBs also auch in der Architektur eines industriell genutzten Data Lake zum Einsatz kommen, wobei sie andere Aufgaben als in der Edge erledigen und somit anderen funktionalen Anforderungen genügen müssen.

Unter allen von db-engines.com gelisteten Datenbankkategorien verzeichnet die Kategorie der TSDBs das stärkste Wachstum zwischen 2017 und 2019 (db-engines.com 2019). Trotz des zunehmenden Interesses in der Praxis gelten TSDBs immer noch als neu und wenig erforscht. Eine initiale Literaturanalyse auf Basis von 40 wissenschaftlichen Veröffentlichungen zu TSDBs (die Literaturliste ist online verfügbar unter: https://bit.ly/3gsehSA) ließ einen Mangel an Wissen über relevante Anforderungen für den Edge-Einsatz erkennen. Es konnten nur wenige Benchmarking-Artikel (Bader et al. 2017; Jensen et al. 2017) identifiziert werden, die Lösungsmerkmale und Features von TSDBs zur Ableitung von Anforderungen enthalten. Diese Artikel berücksichtigen jedoch nicht die spezifischen Bedingungen eines industriellen Einsatzes in der Edge oder im Data Lake und der Mangel an spezifischen Anforderungen für den industriellen Einsatz wird evident. Diese bisher kaum erforschten Anforderungen bestimmen das Ziel dieses Artikels, einen funktionalen Kriterienkatalog für TSDBs im IIoT zu entwickeln. Ausgehend von der Vielfalt der existierenden TSDBs auf dem Markt (Bader et al. 2017; Jensen et al. 2017; Outlyer 2016) soll der Kriterienkatalog für künftige Benchmarks vorhandener TSDBs für den Einsatz in der industriellen Edge, zur Entwicklung der Edge-Komponenten bei der Maschinenüberwachung, sowie zur Integration von Zeitreihendaten in einem industriell genutzten Data Lake, eingesetzt werden können. Funktionale Kriterien werden aus der wiss. Literatur und aus der offiziellen Dokumentation zu einer ausgewählten Datenbank (InfluxDB – siehe dazu ▶ Abschn. 1.2.1) extrahiert. Auf Basis der extrahierten Kriterien werden zusammenhängende Kriterien zu Anforderungen zusammengefasst. Abschließend wird der vollständige Kriterienkatalog am Beispiel von InfluxDB evaluiert, um die Anwendung des Katalogs, sowie die Erfüllung der identifizierten Anforderungen durch InfluxDB zu zeigen und den Auswahlprozess einer geeigneten TSDB zu unterstützen.

1.2 Verwandte Themen

Dieser Abschnitt gibt einen Überblick und dokumentiert die Auswahl einer exemplarischen Datenbank für die Bildung und Anwendung des Kriterienkatalogs und erläutert das Konzept der Data Lakes im IIoT-Kontext.

1.2.1 Überblick über existierende Open-Source-TSDB

Zur Auswahl wird das Ranking des Portals db-engines.com verwendet, da es nicht nur einen landesweiten Suchalgorithmus von Google verwendet, sondern unterschiedliche Suchen kombiniert und Suchergebnisse von mehreren Suchmaschinen, technischen Diskussionen, Jobbörsen und sozialen Netzwerken einschließt (db-engines.com 2019). Basierend auf diesem Ranking belegte im Februar 2019 InfluxDB den ersten Platz unter allen TSDBs (db-engines.com 2019). Weitere in der initial untersuchten Literatur enthaltene Rankings von Bader et al. (2017) und Jensen et al. (2017) werden nicht berücksichtigt. Das von Jensen et al. erstellte Ranking enthält diverse für bestimmte wissenschaftliche Projekte in Publikationen entwickelte Datenbanken, die keine Produktnamen tragen und öffentlich nicht auffindbar sind. Dank der Methodenbeschreibung von Bader et al. (2017) ist es zwar möglich die Methode zu replizieren, allerdings werden dabei nur die Ergebnisse von google.com berücksichtigt. Das resultierende Ranking ist somit ungenauer als von db-engines.com. Basierend auf dem Rankingverfahren von Bader et al. (2017) landet InfluxDB zwar auf dem fünften Platz von insgesamt 33 untersuchten Datenbanken, kann allerdings unabhängig von anderen Datenbanksystemen auf einem einzigen Knoten (Edge-Gerät) betrieben werden und liefert eine native Unterstützung für Überwachungsanwendungen von Drittanbietern wie z. B. Grafana (Grafana 2018). Bei Leistungstests auf einem einzigen Knoten wurde InfluxDB in einigen Studien als führend eingestuft, weil es bei bestimmten Leistungsmetriken eine bessere Performanz als die TSDBs anderer Anbieter aufweisen konnte (Naqvi 2017; Arnst et al. 2019; Martino et al. 2019). Außerdem ist InfluxDB in der untersuchten Version 1.7.4 ein integriertes Modul einer so genannten "TICK-Stack"-Plattform zur Zeitreihenanalyse (InfluxData 2019a). Der TICK-Stack umfasst drei weitere Open-Source-Module und bietet eine Komplettlösung für die Speicherung, Verarbeitung und Analyse von Zeitreihendaten an. Der TICK-Stack ist darauf ausgerichtet die Anforderungen für die Analyse von Sensordaten ganzheitlich zu erfüllen und bietet einen breiteren Funktionsumfang als eine gewöhnliche TSDB. Das Modul Chronograph stellt die administrative Benutzeroberfläche und die Visualisierungs-Engine zur Verfügung. Das Modul Kapacitor ist eine native Datenverarbeitungs-Engine für die Stream- und Batch-Verarbeitung. Telegraf wird für die Datensammlung von Events und Metriken verwendet und bietet des Weiteren Plugins zur Anbindung von diversen Datenquellen an InfluxDB, sowie zur Anbindung von InfluxDB (InfluxData 2019). Beide Konfigurationen (InfluxDB und TICK-Stack) sind im Zusammenhang mit IIoT und der Analyse von Werkzeugmaschinendaten von Nutzen. Deshalb eignet sich InfluxDB zur Extraktion zusätzlicher Kriterien und zur Evaluation des Katalogs. In der aktuellen Version 2 wurden die TICK-Module InfluxDB, Chronograph und Kapacitor in einem einheitlichen Zeitreihenanalysesystem zusammengefasst, das durch die Telegraf-Module erweiterbar ist. Die Version 2 wird jedoch aufgrund des Alpha-Status[2] zum Zeitpunkt der Evaluation nicht betrachtet.

1.2.2 Data Lakes im IIoT

Das Konzept der Data Lakes wurde vom CTO des Business Intelligence Unternehmens Pentaho, James Dixon im Jahr 2010 in seinem eigenen Blog vorgestellt (Jamesdixon 2010). Bislang existiert keine einheitliche Definition eines Data Lake, weshalb der Begriff auch als konzeptioneller Rahmen für bestimmte Technologien wie z. B. verteilte Dateisysteme wie Hadoop Distributed File System (HDFS) verwendet wird (Mathis 2017; Firouzi et al. 2020). Eine umfangreiche und framework-unabhängige Definition wird von Giebler et al. (2020) formuliert: „Ein Data Lake ist allgemein eine skalierbare Datenmanagementplattform für Daten jeder Struktur, die in ihrem Rohformat gespeichert werden, um Analysen ohne vordefinierte Anwendungsfälle zu ermöglichen." Mit dem Data Lake Konzept wird somit angestrebt, massiv skalierbare Systeme zum Management großer Datenmengen in ihrem Originalformat umzusetzen. Demnach ist das Data-Lake-Konzept eine architektonische Lösung zur Verarbeitung strukturierter, unstrukturierter, halbstrukturierter und großer Datensätze (inkl. hochfrequenter Sensordaten) (Mathis 2017; Gröger et al. 2018; Gröger und Hoos 2019). Ohne Beeinträchtigung des Quellsystems sollen Data Lakes unterschiedliche Datenimportverfahren (batch- und stream processing) unterstützen (Gröger und Hoos 2019; Miloslavskaya und Tolstoy, 2016). Data Lakes sollen sowohl strukturierte, als auch semi-strukturierte und unstrukturierte Daten speichern und

[2] Versionshinweise zu InfluxDB: https://www.influxdata.com/blog/category/tech/releases/

verarbeiten können und sind dadurch flexibler als ein Data Warehouse (Miloslavskaya und Tolstoy, 2016; Mathis 2017). Somit kombinieren Data Lakes häufig mehrere Datenbanksysteme, wie z. B. relationale Datenbanksysteme und NoSQL Datenbanksysteme, um jeweils geeignete Speicherlösungen zu bieten (Giebler et al. 2020). So können unter anderem in der Edge gesammelte Zeitreihendaten zur langfristigen Speicherung in einem solchen Data Lake gesammelt, mit anderen Daten (z. B. von anderen Maschinen oder Softwaresystemen) angereichert und beliebigen Stakeholdern zur Verfügung gestellt werden. Im industriellen Kontext bietet ein Data Lake die Möglichkeit ergänzende und kontextgebende Daten z. B. aus Manufacturing Execution Systemen (MES) oder gewöhnlichen Kalkulationstabellen in die Analyse zu integrieren (Gröger und Hoos 2019). Zusätzlich zur Kombination der Zeitreihendaten mit beliebigen anderen Datentypen, bietet ein Data Lake den Vorteil die angereicherten Daten von unterschiedlichen Benutzergruppen wie Domänenexperten oder Datenwissenschaftlern analysieren zu lassen (Gröger et al. 2018; Gröger und Hoos 2019). So ermöglicht ein Data Lake eine ganzheitliche Sicht auf den Fertigungsprozess.

Diese Eigenschaften machen den Data Lake zu einem geeigneten Konzept zur Speicherung hochfrequenter Zeitreihendaten und ihrer Analyse. Zusätzlich zu dem beschriebenen Edge-Einsatz (d.h. als Datenquelle für einen Data Lake), lassen sich TSDBs potentiell in einen Data Lake als Rohdatenspeicher einsetzen (Gröger 2018; Gröger und Hoos 2019). Ein möglicher Einsatz einer TSDB in der konzeptionellen Architektur eines Data Lake auf Grundlage des Lambda-Architekturparadigmas (Gröger und Hoos 2019) ist in Abb. 1.1 skizziert:

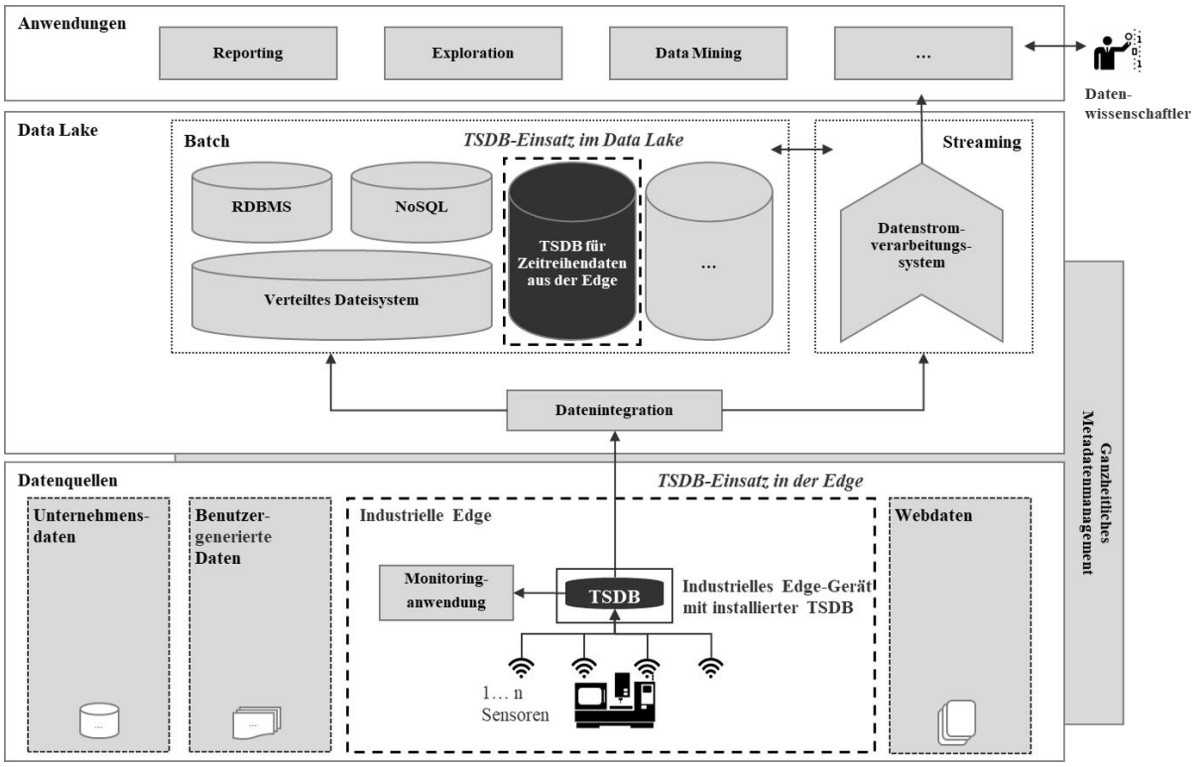

Abb. 1.1 Untersuchte Szenarien für den Betrieb von TSDBs

Aufbauend auf existierenden Forschungsarbeiten zu Data Lakes, stellt sich die Frage, ob sich aktuelle TSDBs wie InfluxDB für den Einsatz im Data Lake eignen und wie gut sie die Anforderungen an einen Data Lake erfüllen. So sind die Datenmengen in einem zentralen Data Lake ohne Mehrwert, wenn sie nicht entsprechend analysiert und genutzt werden können (Naumann und Krestel 2017). Für die Nutzer eines Data Lake ist es z. B. wichtig, zusätzliche Informationen zur Datenherkunft zu haben, um relevanten Edge-Knoten zu identifizieren und den Kontext der Daten zu verstehen. Relevante Metadaten über die Edge können Angaben zum Standort des Edge-Knotens, zur ID des Edge-Gerätes oder des Sensors, sowie die Informationen zum ausgeführten Prozess, oder zur Messeinheit enthalten und helfen den Nutzern eines Date Lake bei der Identifikation und der Wiederauffindung der benötigten Sensordaten. Zur Beurteilung der Herkunft von Zeitreihendaten aus der industriellen Edge sind daher die Möglichkeiten der eingesetzten TSDB solche Informationen zu liefern von Relevanz, weil

sie das einem Data Lake inhärente Metadatenmanagement unterstützen. In einem Data Lake werden für gewöhnlich noch weitere Arten von Metadaten erzeugt und werden im Rahmen eines ganzheitlichen Metadatenmanagements analysiert. Dazu gehörten z. B. die Metadaten über die im Data Lake gespeicherten Daten, sowie Metadaten zu den durchgeführten Analysen, gebraucht werden (Gröger und Hoos 2019).

1.3 Einsatzszenarien für Zeitreihendatenbanken im industriellen IoT

Folgend werden die Einsatzszenarien für TSDBs in der industriellen Edge und im Data Lake beschrieben. Um den Nutzen von TSDBs zu verdeutlichen, werden Einsatzszenarien am Beispiel eines Vakuumhandhabungssystems konkretisiert (Schmalz 2019). Im Abschnitt 3.1 wird die echtzeitnahe Überwachung der Komponenten in Vakuumhandhabungssystemen in der Edge beschrieben. Anschließend wird im Abschnitt 3.2 die weitergehende Nutzung der erzeugten Daten im Data Lake beschrieben.

1.3.1 Beispielhaftes Einsatzszenario für die Edge

Ein typisches Vakuumhandhabungssystem (siehe ◉ Abb. 1.2) besteht aus einem Handhabungsroboter und einem Greifer. Der Greifer eines Vakuumhandhabungssystems besteht aus mehreren Vakuumsauggreifern (Aktoren), die in Kontakt mit dem Werkstück kommen und somit einem mechanischen Verschleiß unterliegen. Mit der Zeit führt der Verschleiß zur Undichtigkeit und zur Entstehung von Leckagen. Wenn die Leckagen zu stark sind und das erforderliche Vakuumniveau nicht mehr stabil gehalten werden kann, kann die sichere Handhabung des Werkstücks nicht mehr garantiert werden. Die verschlissenen Vakuumsauggreifer des Handhabungssystems müssen bei der nächsten Inspektion ausgetauscht werden, da sonst Menschen oder Sachgüter (z. B. anliegende Förderbänder) zu Schaden kommen können.

Bei einer Edge-Speicherung und Analyse der Sensordaten zum vorherrschenden Vakuumniveau können mehrere Vorteile erreicht werden. So kann z. B. der Ausfall des Vakuumhandhabungssystems frühzeitig vorhergesagt werden, wobei eine cloudbasierte Plattform und somit eine Internetanbindung, zur Analyse der Sensordaten, nicht erforderlich ist. Ein Stillstand des Vakuumhandhabungssystems kann durch eine entsprechende Meldung an die Steuerungskomponente auf Basis zusätzlicher Sensorwerte schneller als bei einer zentralisierten Auswertung (z. B. im Data Lake) herbeigeführt werden. So kann zwar die Steuerung der Vakuumerzeuger die Schwellenwerte erkennen, aber die für den Stillstand des Systems verantwortliche Robotersteuerung hat diese Daten nicht, da beide Steuerungen in der Regel getrennt und nicht immer miteinander kompatibel sind. So kann das Gesamtsystem von den Zeitreihendaten aus der Edge profitieren.

Falls das überwachte System über eine veraltete Steuerung verfügt, die nicht mit dem Internet verbunden werden darf, bietet die Edge ebenfalls Vorteile, da die Datenanalyse und Speicherung direkt auf dem Edge-Gerät ohne Internetzugriff erfolgen. Die Überwachung des Vakuumniveaus erfolgt mit einer Abtastrate von 1 Hz. Ein marktübliches Handhabungssystem kann nach Kundenwünschen flexibel mit Vakuumsauggreifern (Aktoren) bestückt werden (Schmalz 2019). Obwohl pro Sensor jeweils eine einstellige Zahl von Aktoren überwacht wird, kann eine Linie mit mehreren Vakuumhandhabungssystemen hunderte von Aktoren haben, die eine hohe parallele Schreiblast auf die TSDB in der Edge generieren. Wie im ▶ Abschn. 1.1 beschrieben, wird zur Überwachung des gesamten Handhabungssystems ein Edge-Gerät mit einer TSDB, angelehnt an Martinviita (2018), eingesetzt.

1.3.2 Beispielhaftes Einsatzszenario für den Data Lake

Wie in Abschnitt 1 erläutert, weist die Edge signifikante Limitationen (1) beim Aufbau einer historischen Zeitreihendatensammlung, (2) bei der Kontextualisierung der Zeitreihendaten und (3) bei unterschiedlichen Analysen auf. So ist bei einer modernen Steuerung des Vakuumhandhabungssystems die Anbindung des Edge-Geräts an das Internet denkbar, damit die kurzzeitig in der Edge gespeicherten Sensordaten zusammen mit den Metadaten zum Vakuumhandhabungssystem (Standort des Systems, ID des Edge-Gerätes und der einzelnen Sensoren für jede erzeugte Zeitreihe) z. B. am Ende der Schicht in einen Data Lake für Langzeitspeicherung und historische Langzeitanalysen

überführt werden. Im Data Lake können die Daten langfristig aufbewahrt werden, um unter Berücksichtigung der Vielfalt an relevanten Umgebungsdaten ein kunden- und domänenübergreifendes Fehlerrepository zu bilden. So können zahlreiche domänenspezifische Datenausschnitte zusammengeführt werden, was die longitudiale Auswertung der aggregierten Datensätze über große Zeiträume ermöglicht. Bei Vakuumhandhabungssystemen bringt die Überwachung des Energieverbrauchs im Data Lake Vorteile, da so der Energieaufwand für ganze Anlagen oder Produktionsstandorte aufgenommen und mit Prozessdaten angereichert werden kann.

Bei einer langfristigen Speicherung im Data Lake können die Energieverbrauchsdaten den Datenwissenschaftlern zur Verfügung gestellt werden. So lassen sich historische Zeitreihendatensammlungen zum Data Mining und zum Antrainieren der Modelle nach den Ansätzen des maschinellen Lernens (ML) verwenden. Der Data Lake ermöglicht es, die Zeitreihendaten aus der Edge mit weiteren Daten anzureichern. Es können bspw. weitere Prozessdaten aus MES-Systemen oder Daten aus einer Produktionsstoppsimulation sein. Damit ließe sich die Überwachung der kritischen Prozessparameter (Condition Monitoring) durch den Einsatz künstlicher Intelligenz (KI) unterstützen. Nach der Anreicherung der Daten im Data Lake und dem Einsatz der Algorithmen zum Trainieren des ML-Modells (Offline-Phase), kann das Modell wieder in der Edge (z. B. auf einem Industrie-PC als Edge-Knoten) zur Vorhersage der Energieverbrauchswerte implementiert werden (Online-Phase). Abb. 1.2 illustriert die beschriebenen Einsatzszenarien:

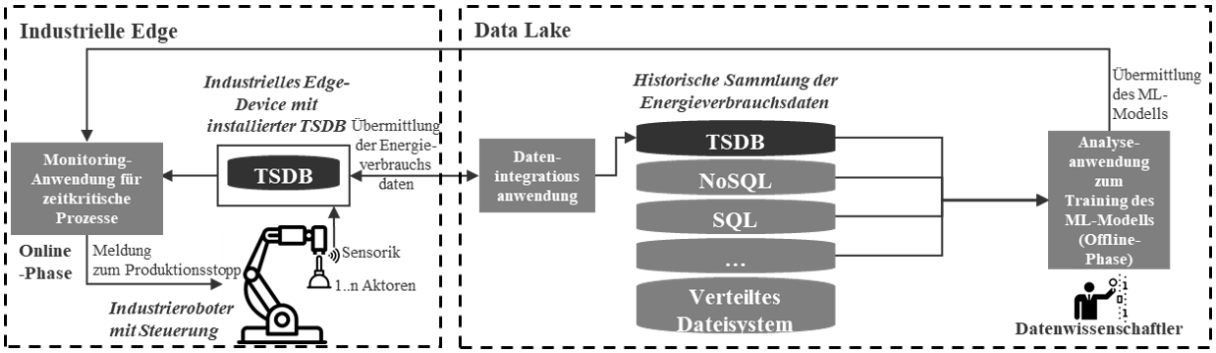

Abb. 1.2 Grafische Darstellung des untersuchten industriellen Einsatzszenarien für TSDBs

1.4 Erstellung des Kriterienkatalogs

1.4.1 Methodisches Vorgehen

Die Interdisziplinarität des untersuchten Einsatzszenarios erfordert, die Berücksichtigung verwandter Forschungsgebiete im Literaturanalyseprozess zur Ableitung von Kriterien:

- IIoT als verwandtes Forschungsgebiet liefert mögliche Eigenschaften vernetzter Werkzeugmaschinen und industrieller Sensordaten, die in TSDBs gespeichert werden.
- Fog- und Edge-Architektur als verwandtes Forschungsgebiet liefert potentielle Kriterien, da eine TSDB auf entsprechenden Edge-Geräten installiert und betrieben werden soll.
- Data Lakes als verwandtes Forschungsgebiet, da TSDBs zusätzlich zur Edge als Speicher für Rohdaten und aggregierte Daten eine Funktion in Data Lakes haben können.
- Big Data Analytics als verwandtes Forschungsgebiet kann potentielle Kriterien zur Analyse der Zeitreihendaten in TSDBs liefern.

Die verwendeten wissenschaftlichen Datenbanken und die Suchbegriffe sind online verfügbar unter: https://bit.ly/3gsehSA. Da einige Suchbegriffe zu sehr hohen Trefferzahlen führten, wurden in diesem Fall nur die ersten 40 Suchtreffer in die weitere Analyse einbezogen. Das Suchergebnis umfasste insgesamt 1003 wissenschaftliche Artikel. Nach der Relevanzbewertung auf der Grundlage der Kurzfassung und dem Ausschluss von Duplikaten ergab sich eine Stichprobe von 191 Artikeln. Diese Artikel wurden nach möglichen funktionalen Kriterien durchsucht. Am Ende wurden in 37 Artikeln 51

relevante funktionale Kriterien identifiziert und zu zwölf funktionalen sowie nicht-funktionalen Anforderungen für die industrielle Verwendung der TSDBs in der Edge verdichtet. Im zweiten Schritt wurde die offizielle Dokumentation für InfluxDB nach weiteren funktionalen Kriterien zur Erfüllung der Sicherheitsanforderung durchsucht, da die untersuchte wissenschaftliche Literatur überraschend nur sehr wenige sicherheitsrelevante Kriterien enthielt. Im letzten Schritt wurden die Kriterien basierend auf den identifizierten Konzepten zu Anforderungen gruppiert, wobei jede Anforderung aus mindestens einem Kriterium besteht.

1.4.2 Kriterienkatalog für Zeitreihendatenbanken

Dieser Abschnitt beschreibt die identifizierten Kriterien und die zusammengefassten funktionalen und nicht-funktionalen Anforderungen. Insgesamt wurden während der Literatur- und der Dokumentationsanalyse **51 funktionale Kriterien** identifiziert. Auf Basis dieser Kriterien werden **zwölf Anforderungen** gebildet, die in ◉ Tab. 1.1 dargestellt werden. Zusätzlich wird im Katalog die Bewertung der Relevanz der Anforderungen zur Umsetzung der in ▶ Abschn. 1.3 skizzierten Einsatzszenarios der TSDB in vier Ausprägungen dargestellt. Obwohl die weniger wichtigen und neutralen Anforderungen eine geringere Bedeutung für die ausschließliche Realisierung der Einsatzszenarien zur Maschinenüberwachung haben, kann die Erfüllung dieser Anforderungen z. B. die Kosten bei der Realisierung oder im Betriebe senken oder die Komplexität im Betrieb reduzieren. Bei der Vielzahl der verfügbaren TSDBs kann die Erfüllung dieser Anforderungen zu einem höheren Begeisterungspotential bei den einsetzenden Industrieunternehmen führen. In diesem Artikel werden nur zentrale Quellen für die jeweiligen Anforderungen angegeben. Das vollständige Quellenverzeichnis inkl. der Zuordnung im Katalog ist online verfügbar unter: https://bit.ly/3gsehSA

Tab. 1.1 Kriterienkatalog für TSDBs in der industriellen Edge

Anforderungen	Kriterien	Relevanz Edge	Relevanz Data Lake
Autarkie im Betrieb	Unabhängigkeit von externen Datenbanksystemen (z. B. Cassandra, HBase)	Sehr wichtig	Weniger wichtig
Modulare Erweiterungs-möglichkeiten	Open-Source-Entwicklungsmodell Premium Support Zustand nach dem Produktlebenszyklus	Weniger wichtig	Weniger wichtig
System-anforderungen	Hardwareanforderungen Datenträgeranforderungen (HDD/SSD) Erforderliches Betriebssystem Unterstützte Cloudinfrastrukturanbieter	Neutral	Weniger wichtig
Industrielles Last-management	Multiwrite-Unterstützung Unabhängigkeit zwischen Lesen und Schreiben Workload Handling Approximation (z. B. AQP) Caching	Sehr wichtig	Wichtig
Interoperabilität	Schnittstellen/API Unterstützte Protokolle Docker Support Verfügbare Konnektoren für Rechencluster	Wichtig	Sehr wichtig

	Verfügbare Konnektoren für Node-Red		
	Verfügbare Konnektoren für Data Analytics Tools		
	Verfügbare Konnektoren für proprietäre IIoT-Plattformen (z. B. ThingWorx, Predix, MindSphere)		
	Unterstützung von NoSQL-Datenbanken		
	Unterstützung von verteilten Dateisystemen		
Abfragesprache	SQL SQL-basierte Abfragesprache Nicht-SQL-basierte Abfragesprache	Weniger wichtig	Weniger wichtig
Speicherplatz-management	Speichergranularität Datenbank Engine Datenkompression Datenaufbewahrungsrichtlinien	Sehr wichtig	Wichtig
Metadaten	Speicherung von Metadaten Möglichkeit zur Abfrage von Metadaten	Sehr wichtig	Sehr wichtig
Verteilter Betrieb	Clusterfähigkeit Multi-Node Unterstützung Unterstützung von Multi-tier Architekturen	Wichtig	Sehr wichtig
Analytics	Grundlegende Analysefunktionen Fortgeschrittene Analytics-Funktionen Machine Learning Stream-Verarbeitung GUI für Analytics GUI für Datenbeziehungen Alarmierung	Sehr wichtig	Neutral
Verfügbarkeit	Backup-Funktionalität Automatische Backups Alarmierung	Wichtig	Weniger wichtig
Sicherheit	Authentifizierung Nutzung digitaler Zertifikate Benutzerverwaltung Verschlüsselung des Kommunikationsprotokolls Datenverschlüsselung Auditing Methoden gegen Angriffe	Sehr wichtig	Sehr wichtig

Autarkie im Betrieb: Eingesetzt in der Edge muss die TSDB autark von anderen Datenbanksystemen arbeiten können. Die Autarkie wird durch das identifizierte Kriterium nach der Unabhängigkeit einer TSDB von anderen Datenbanksystemen erreicht. Die Autarkie im Betrieb ist eine wesentliche Voraussetzung für den Einsatz in der industriellen Edge, um die Bandbreiten- und Latenzprobleme zu umgehen. Sie spart zudem den Aufwand für den Aufbau eines verteilten Computerclusters (z. B. Apache HBase). Bei dem skizzierten Edge-Einsatzszenario spielt die Anforderung eine wichtige Rolle, um ohne die Internetverbindung und ohne ein zusätzliches System die Steuerung des Handhabungssystems mit Daten zu versorgen. Bei dem skizzierten Einsatzszenario im Data Lake hat diese Anforderung eine geringere Bedeutung, weil ein Zugriff der TSDB auf die Funktionalitäten anderer Datenbanken aus der Systemumgebung möglich ist, um bestimmte funktionale Einschränkungen der TSDB auszugleichen.

Modulare Erweiterungsmöglichkeiten: Die TSDB in der Edge muss modular erweiterbar sein, um bspw. bestimmte Anpassungen für die Aufnahme von bestimmten Datentypen oder Anbindung bestimmter Maschinensteuerungen vorzunehmen. Diese Anforderung wird durch das Open-Source-Softwareentwicklungsmodell sowie durch den Support und die Lizenzbedingungen des Datenbankanbieters erfüllt. Das Open-Source-Modell ermöglicht es der Softwareentwicklungsabteilung des Industrieunternehmens und der Open-Source-Community die TSDBs an die vorhandenen Schnittstellen der Maschinen oder der eingesetzten Softwaresysteme, über die Administrationsmöglichkeiten hinaus, anzupassen. Über ein Angebot des kommerziellen (Premium-)Supports durch den TSDB-Anbieter, kann dieser spezifische Funktionalitäten implementieren, wodurch der Entwicklungsaufwand für die Industrieunternehmen sinkt. Weiterhin ist der aktuelle Stand der TSDB innerhalb des Produktlebenszyklus das dritte relevante Kriterium. Steht eine TSDB eher am Anfang im Produktlebenszyklus, wirkt sich die erhöhte Release-Häufigkeit und die Vielzahl von Versionen auf den Wartungsaufwand auf verteilten Edge-Geräten und mögliche Ausfallzeiten durch kritische Updates aus. Andererseits kann eine frühe Phase im Produktlebenszyklus auch auf anstehende Funktionalitäten und die positive Entwicklungsdynamik hinweisen und das Vertrauen in die Langlebigkeit einer TSDB erhöhen. In der Literatur wird zudem die Notwendigkeit einer stabilen Release-Version als Indikator für die Bewertung der Reife einer Datenbank erwähnt. Eine hohe Anzahl von stabilen Release-Versionen lässt auf eine hohe Updatefrequenz schließen und erschwert die Datenbankenauswahl für die lange Lebensdauer von Werkzeugmaschinen, die oft über 15 Jahre hinausgeht. Zur Umsetzung des Einsatzszenarios in der Edge ist die Anforderung nicht direkt erforderlich. Die einzige Ausnahme ist, wenn z. B. die Schnittstelle zur ergänzenden Unternehmenssoftware nicht vorhanden ist oder das Datenformat nicht unterstützt wird und die Funktionalität durch ein Modul bereitgestellt werden muss. Beim Einsatz im Data Lake kann eine ähnliche Einschätzung getroffen werden, wobei die Modularität helfen kann, auf bereits existierende und im Idealfall vom Datenbankanbieter offiziell unterstützte Konnektoren und Clients zuzugreifen.

Systemanforderungen: Die TSDB muss auf gängigen industriellen Edge-Geräten und Industriecomputern (als Einzelkonten) betrieben werden können. Edge-Geräte verfügen nur über eine begrenzte Rechenleistung und sind mit spezifischen Betriebssystemen ausgestattet. Folglich müssen TSDB mit typischen Hardware-Builds der Edge-Geräte (siehe ▶ Abschn. 1.1) stabil laufen und die ARM-, sowie die x86-Hardwarearchitekturen unterstützen. So können TSDBs bestimmte Anforderungen an die Speichermedien haben. Das InfluxDB Beispiel zeigt, dass diese TSDB nicht für den Betrieb auf einer Festplatte (HDD) ausgelegt ist und stattdessen eine Solid State Disk (SSD) benötigt, was den Preis der Edge-Geräte beeinflusst. Weiterhin muss die TSDB mit den auf Edge-Geräten häufig eingesetzten OS kompatibel sein, wie z. B. spezifisch angepassten oder echtzeitfähigen Linux-Distributionen oder Windows IoT. Die niedrigen Systemanforderungen sowie die Unterstützung kompatibler Betriebssysteme sind für die Umsetzung des Einsatzszenarios von untergeordneter Rolle, dennoch kann der Einsatz von wenig leistungsfähigen Edge-Geräten Kostenvorteile bringen, falls die Überwachung in der Edge als digitaler Service verkauft werden soll. Bei einem TSDB-Einsatz im Data Lake sind die möglichst niedrigen Systemanforderungen weniger relevant, weil im Kontext einer zentralen Datenplattform skalierbare Servercluster eingesetzt werden. Somit sind die Restriktionen der Edge, bezogen auf die Datenträger oder das eingesetzte Betriebssystem, nicht mehr präsent. Da ein Data Lake jedoch auch in der Cloud betrieben werden kann, ist zusätzlich zu den aufgeführten

Betriebssystemen, die Unterstützung der wichtigsten Cloudinfrastrukturanbieter von Bedeutung.

Industrielles Lastmanagement: In der Edge muss die TSDB die Last industrieller Daten aushalten und managen können. Unter dem industriellen Lastmanagement werden Kriterien zur Verfügbarkeit und Leistungsfähigkeit einer TSDB im Umgang mit parallelen Sensordatenströmen (sowohl Lese- als auch Schreibvorgänge) zusammenfasst. Die Literatur beschreibt verschiedene Kriterien, um diese Anforderung zu erfüllen. Obwohl die Anforderung in der Literatur eher als Skalierbarkeit definiert wird, stellt die Skalierbarkeit die industrielle Leistungsanforderung nicht angemessen dar. Die Performance sollte auch durch stabiles Arbeiten ohne Leistungseinbußen der TSDB gewährleistet werden, trotz parallel auftretender unregelmäßiger Anfragen. Daher wird diese Anforderung als „industrielles Lastmanagement" genauer definiert. Das erste Kriterium ist die Unterstützung mehrerer paralleler Datenströme (Schreibprozesse), die sich aus einer Vielzahl von Sensoren pro Maschine ergeben können und wird daher als Multiwrite-Unterstützung für industrielle Schreiblasten bezeichnet. So wird die Unabhängigkeit der Lese- und Schreibleistung in einer TSDB erwähnt, damit die Performance von Leseanfragen nicht von der Last durch parallele Schreibanfragen beeinflusst wird und umgekehrt. Die Verletzung der Unabhängigkeit führt zu Engpässen und einer geringeren Performance der TSDB, durch höhere Latenzzeiten bei hoher Schreiblast. Das industrielle Workload-Handling zeigt, ob eine TSDB in der Lage ist hochfrequente Daten einer Werkzeugmaschine zu speichern. So muss eine TSDB in der Lage sein, bis zu einer Million Schreibanfragen für jeden Sensor, ohne Leistungsverluste oder Abstürze, zu unterstützen. Die Unterstützung von Approximationstechniken wie z. B. das Approximate Query Processing (AQP) kann die Abfragezeit für die Datenaggregation verbessern, indem es die Sensordaten in mathematische Modelle oder Stichproben umwandelt und annähernde Ergebnisse liefert (Jensen et al., 2017). Die Unterstützung von Caching ist eine zusätzliche Technik zur Verbesserung der Abfragezeit und der Gesamtleistung einer TSDB. Beide Techniken verbessern die Zeit der Leseanfragen für Analytics-Anwendungen und leisten einen Beitrag zur Erreichung der Echtzeitfähigkeit auf Edge-Geräten. Die Erfüllung der Lastmanagementkriterien durch die TSDB-Funktionen sind aufgrund der Vielzahl von Aktoren in unserem Einsatzszenario von großer Relevanz. Ein marktübliches Handhabungssystem kann nach Kundenwünschen flexibel mit Vakuumsauggreifern (Aktoren) bestückt werden (Schmalz 2019). Obwohl pro Sensor jeweils eine einstellige Zahl von Aktoren überwacht wird, kann bei bis zu 240 Aktoren eine hohe parallele Schreiblast auf die TSDB in der Edge entstehen. Deshalb sind die Multiwrite-Unterstützung sowie die Unabhängigkeit zwischen Schreibzugriffen des Handhabungssystems und Lesezugriffen durch die Mitarbeiter von Bedeutung. Eingesetzt im Data Lake, hat diese Anforderung aufgrund der Skalierbarkeit der IT-Infrastruktur eine leicht geringere Bedeutung, wird aber dennoch als wichtig eingestuft. Zum Beispiel kann die TSDB im Data Lake zur Aufnahme der Zeitreihendaten von einer sehr großen Anzahl an Vakuumhandhabungssystemen genutzt werden (z. B. als unternehmensweiter Rohdatenspeicher für Energieverbrauchsdaten aller vernetzten Vakuumhandhabungssysteme). Gleichzeitig muss die TSDB Lesezugriffe anderer Anwendungen des Data Lake ermöglichen und dabei fehler- und absturzfrei funktionieren. Somit können die Mengen der zu speichernden und abzufragenden Zeitreihendaten, verglichen mit der Edge, vielfach größer sein, wenn die TSDB entsprechend in einer hierarchischen Architektur zur Aggregation von Daten aller verfügbaren Quellsysteme aus der Edge eingesetzt wird.

Interoperabilität: Die TSDB in der Edge muss mit diversen Werkzeugmaschinen und Datenspeichern kommunizieren können. Unterstützte Anwendungsprogrammierschnittstellen (API) werden von anderen Systemen oder Anwendungen verwendet, um sich mit einer TSDB zu verbinden, Abfragen auszuführen und Daten oder Ergebnisse abzurufen. Die Berücksichtigung bestimmter Architekturstile, z. B. Representational State Transfer (REST), bei der Konzeption der APIs, könnte zu einer Vereinfachung der Anbindung der TSDB mit externen Webservices und Anwendungen führen. Aufgrund der Vielfalt der Maschinenprotokolle im industriellen Einsatzszenario wird die Unterstützung bestimmter Protokolle (neben dem HTTP-Protokoll) wie z.B. MQTT oder OPC-UA zu einem wichtigen Kriterium. Native Unterstützung der gängigen Industrieprotokolle erspart den Entwicklungsaufwand für Konnektoren für die Industrieunternehmen. Die Docker-Unterstützung wirkt sich positiv auf die Interoperabilität in der Edge aus, da Docker-Images den Implementierungsaufwand der TSDB auf unterschiedlichen Edge-Geräten verringern und zusätzlich die Bereitstellung im Feld beschleunigen (Martinviita 2018). Standardisierte Konnektoren vereinfachen die Konnektivität der TSDB mit zentralisierten Data Lakes, statistischer oder ML-Software oder sonstigen Drittsystemen zur Datenanalyse. Darüber hinaus wird die

Unterstützung von Node-Red (grafisches Entwicklungswerkzeug) als ein interoperabilitätssteigerndes Feature in den Kriterienkatalog aufgenommen, da es auf Edge-Geräten wie dem Raspberry Pi lauffähig oder gar vorinstalliert ist und als Open-Source-Tool verschiedene sog. „Nodes" enthält, die z. B. als Konnektoren für verschiedene Softwaresysteme und Datenbanken oder als Datenkonverter in andere Formate die Interoperabilität erhöhen und die Anbindung vereinfachen. Zudem bieten bspw. verschiedene industrielle IIoT-Plattformen wie MindSphere, ThingWorx oder Predix eigene Node-Red-basierte Konnektoren an. Die IIoT-Plattformen bieten die Möglichkeit, Maschinen- und Sensordaten in der Cloud langfristig zu speichern und historische Analysen durchzuführen. Durch die Konnektivität der TSDB mittels Konnektoren ohne Entwicklungsaufwand, resultieren für Industrieunternehmen zeitliche und monetäre Vorteile. Da der Zweck von Edge-Geräten darin besteht, die Sensordaten kurzfristig zu speichern, sollte eine in der Edge betriebene TSDB mit verteilten NoSQL-Datenbanken interoperabel sein und verteilte Dateisysteme wie HBase in einem Hadoop-Cluster über Konnektoren nativ unterstützen. Die Anforderung der Interoperabilität ist bei dem Szenario eingeschränkt erforderlich. So sind die unterschiedlichen existierenden Konnektoren für externe Systeme bei einem abgeschlossenen Netzwerk nicht relevant. Dennoch ist die Unterstützung der relevanten Maschinenprotokolle relevant, da die Sensoren in Handhabungssystemen von Schmalz über MQTT die Daten senden. Zudem sind unterstütze Schnittstellen wichtig, um die Daten vom Edge-Gerät zu einem Applikationsserver des betreibenden Unternehmens zu senden. Die Konnektoren für cloudbasierte IIoT-Plattformen hingegen sind bei einem geschlossenen Netzwerk irrelevant. Für das Einsatzszenario im Data Lake ist die Interoperabilität von sehr großer Relevanz, damit die TSDB mit einem möglichst geringen Implementierungsaufwand mit anderen typischen Subsystemen und Datenbanken im Data Lake kommunizieren kann. Diese Anforderung kann insbesondere durch verfügbare und vom Datenbankanbieter offiziell unterstützte Konnektoren für Rechencluster, Analytics Tools, NoSQL-Datenbanken und IIoT-Plattformen unterstützt werden. Wenn ein Data Lake auf verteilte Dateisysteme wie HDFS setzt, ist ein Dateiaustausch mit solchen Dateisystemen hilfreich. Die Unterstützung bestimmter industrieller Maschinenprotokolle wiederum wird im Kontext eines Data Lake unwichtig, da ein Datenaustausch mit anderen Anwendungen eins Data Lake auf Basis des HTTP-Protokolls auf der Anwendungsschicht mit Hilfe der REST-API zur Kommunikation erfolgt. Die Docker-Unterstützung kann im Kontext eines Data Lake ähnlich wie in der Edge helfen, die Skalierung der TSDB-Cluster für zusätzliche Vakuumhandhabungssysteme zu vereinfachen.

Abfragesprache: Die TSDB in der Edge muss einen hohen Funktionsumfang bei möglichst geringer Komplexität über die Abfragesprache liefern können. Komplexere Abfragesprachen können den Funktionsumfang einer TSDB erhöhen, aber gleichzeitig die Benutzung erschweren, da Benutzer zuerst eine zusätzliche Sprache lernen müssen. Bader et al. (2017) und Jensen et al. (2017) erwähnen dieses funktionale Kriterium und unterscheiden drei Typen von Abfragesprachen. Eine TSDB kann demnach SQL als Abfragesprache oder eine eigene Sprache verwenden, die entweder eine SQL-basierte oder eine nicht-SQL-basierte Abfragesprache sein kann. Obwohl spezifische Abfragesprachen die Komplexität bestimmter Abfragen z. B. durch neue Befehle senken können, ist es nicht sichergestellt, dass sie auch den Funktionsumfang von SQL abdecken. Zudem können Datenwissenschaftler in Maschinenbauunternehmen aufgrund der Vertrautheit mit SQL komplexe Abfrageanforderungen in kürzerer Zeit erstellen, als mit nicht-SQL-basierten Abfragesprachen. Bei ausreichendem Funktionsumfang spielt die Abfragesprache bei der Umsetzung des Szenarios aus ▶ Abschn. 1.3.1 eine untergeordnete Rolle (Bedienkomfort des Datenwissenschaftlers). Beim Einsatz im Data Lake sollte die Abfragesprache der TSDB vor dem Hintergrund bewertet werden, ob sie die Standardisierung fördert und den Self-Service aus der Nutzerperspektive unterstützt. Angesichts der Vielzahl unterschiedlicher Systeme im Data Lake kann es sinnvoll sein, die Anzahl an Abfragesprachen im Data Lake gering zu halten. Das hilft bei der systemübergreifenden Föderierung der Daten im Data Lake. Bezogen auf die Energieverbrauchsdaten eines Vakuumhandhabungssystems, wird der Self-Service bei der Analyse der Zeitreihendaten durch die unterschiedlichen Nutzergruppen eines Data Lake unterstützt (Gröger und Hoos 2019). Von daher sollte die verwendete Abfragesprache, falls es keine SQL-Sprache ist, mathematische Berechnungen und die Integration weiterer Quellsysteme zur Kontextualisierung der Zeitreihendaten zum Energieverbrauch bei umfangreichen Datenanalysen innerhalb der Abfragen ermöglichen. Außerdem kann so die vom Data-Lake-Konzept grundsätzlich angestrebte Reduktion der Datentransformationsschritte im Analyseprozess durch zusätzliche Anwendungen unterstützt werden.

Speicherplatzmanagement: Beim Edge-Einsatz muss die TSDB Speichermanagement-Funktionen zur Erreichung besserer Speichereffizienz bereitstellen, da Edge-Geräte nur beschränkten Speicherplatz zur Verfügung stellen und die Frequenz der generierten Daten sehr hoch sein kann. Da einige IIoT-Szenarien Daten in Zeitintervallen von Milli- oder sogar Nanosekunden erzeugen, muss die TSDB diese Speichergranularitäten unterstützen. Um den benötigen Speicherbedarf zu reduzieren, ist eine effektive Datenbank Engine erforderlich. Zeitstempel können komprimiert werden, um den Platzbedarf zu reduzieren. Weiterhin ermöglicht die Definition von Aufbewahrungsrichtlinien, Daten nach definierten Zeiträumen zu löschen, um Speicherplatz auf dem Edge-Gerät, das durch ältere Zeitreihen genutzt wird, zu sparen und reduziert den manuellen Administrationsaufwand zur Bereinigung. Die Überwachung des Vakuumniveaus im Einsatzszenario erfolgt mit einer Abtastrate von 1 Hz und stimmt mit dem beschriebenen Prototyp überein. Das bedeutet wiederum, dass die von der InfluxDB unterstützte Speichergranularität im Millisekunden- und Nanosekundenbereich nicht benötigt wird. Aufgrund der potentiell hohen Anzahl der Aktoren sind eine effiziente Komprimierung und die Möglichkeit, den Wartungsaufwand des Edge-Geräts durch definierte Aufbewahrungsrichtlinien zu reduzieren, wichtig. Eingesetzt im Data Lake ist diese Anforderung von leicht geringerer Bedeutung, da die skalierbare Cluster-Architektur eines Data Lake die Restriktionen eines Edge-Geräts aufhebt. Dennoch ist dieses Kriterium wichtig, da zur Überführung der Zeitreihendaten in einen zentralen Data Lake die gleiche Speichergranularität von der TSDB unterstützt werden sollte wie beim Edge-Einsatz. Zudem ist das Kriterium der Datenkompression wichtig, um eine langfristige Aufbewahrung von Rohdaten zu ermöglichen. Auch das Kriterium der Datenaufbewahrungsrichtlinien ist für die Governance eines Data Lake von Bedeutung. Bereits verarbeitete und aggregierte Zeitreihendaten oder Zeitreihen ohne Anomalien können nach definierten Zeiträumen gelöscht werden. Für einen effizienten technischen Betrieb, können so einerseits die Menge wertloser Daten reduziert werden und andererseits die Kosten zur Datenhaltung unter Kontrolle gehalten werden. Zudem erfordert eine kommerzielle Verwertung der Zeitreihendaten von Kunden fest definierte Datenaufbewahrungsrichtilinien zur Erfüllung der Datenschutz-Grundverordnung (Gröger und Hoos 2019).

Metadaten: Die TSDB in der Edge muss Metadaten speichern können. Da moderne Werkzeugmaschinen über eine Vielzahl an Sensoren verfügen und jeder Sensor eine separate Datenmessung ermöglicht, können Metadaten in der TSDB präzise Beziehungen zwischen den Datenmessungen und den dazugehörigen Sensoren herstellen. Dies kann die Zeit verkürzen, die ein Datenwissenschaftler benötigt, um den Sensor oder die Dateneinheit eindeutig zu identifizieren. Informationen über die Abtastrate oder den Datentyp der Sensormessung sind mögliche Beispiele für Metadaten aus dem IIoT. Bei der Vielzahl der potentiellen Aktoren im Einsatzszenario ist die Unterstützung für Metadaten zur eindeutigen Identifikation defekter Aktoren im System von großer Relevanz im Edge-Einsatzszenario. Beim Betrieb der TSDB im Data Lake haben Metadaten ebenfalls eine hohe Bedeutung, um die Beurteilung der Datenherkunft sowie die Analyse des Datenverlaus über unterschiedliche Datenspeicher zu ermöglichen. In einem Data Lake werden Metadaten unterschiedlicher Datenspeicher des Data Lakes, wie z. B. TSDBs und relationale Datenbanksysteme, meist in einem dedizierten System, einem Datenkatalog, gespeichert (Gröger und Hoos 2019). Datenkataloge bieten Funktionen zur Exploration, Verknüpfung und Anreicherung der Metadaten, z. B. um einzelnen Zeitreihen fachliche Dateneigentümer zuzuordnen. Die Verknüpfung von Metadaten dient der Abbildung der Datenherkunft und des Datenverlaufs, um z. B. aufzuzeigen, dass die Daten einer Zeitreihe aus einer bestimmten TSDB in der Edge stammen und nun in mehreren Datenspeichern des Data Lakes abgelegt sind. Dabei beschreiben die Metadaten ebenso, ob die Originaldaten aus der TSDB in diesen verschiedenen Datenspeichern des Data Lakes als unveränderte, vollständige Kopie der Originaldaten oder z. B. in aggregierter, gefilterter oder anderweitig verarbeiteter Form abgelegt sind. Bei Änderungen in den Quellsystemen können damit die Auswirkungen auf die Speichersysteme des Data Lakes systematisch nachvollzogen und Datentransformationsprozesse gezielt angepasst werden. Im skizzierten Einsatzszenario helfen die Metadaten dabei die aggregierten Energieverbrauchsdaten der Vakuumhandhabungssysteme aus unterschiedlichen Produktionsstätten eindeutig zu identifizieren. Die effiziente und flexible Abfrage von Metadaten einer TSDB spielt im Data Lake folglich eine zentrale Rolle. Im Unterschied zu der Edge befinden sich die Datenkataloge in der Systemumgebung einer TSDB. Sie sind an der Erfüllung der Anforderung beteiligt und können helfen die fehlenden Funktionalitäten der TSDB im Umgang mit Metadaten auszugleichen und die Verwaltung der

extrahierten Metadaten zu ermöglichen (Gröger und Hoos 2019; Giebler et al. 2020).

Verteilter Betrieb: Die TSDB in der Edge muss einen verteilten Betrieb gewährleisten, um eine hohe Verfügbarkeit und Skalierbarkeit zu ermöglichen. Die Clustering-Funktionalität ermöglicht es einer TSDB, verteilt auf mehreren Knoten zu arbeiten. Ein verwandtes Konzept, das die Clustering-Funktionalitäten nutzt, ist z. B. das Load-Balancing zur Verarbeitung großer Datenmengen. Darüber hinaus sollte die Unterstützung von Multi-Tier-Architekturen angeboten werden, um die Hierarchien zwischen mehreren Werkzeugmaschinenclustern im Shopfloor oder unterschiedlichen Datenbanken verschiedener Ebenen in einer TSDB bündeln zu können. Die Clusterfähigkeit im verteilten Betrieb könnte die Unterbrechung der Überwachung, bei potentiellen Ausfällen des Edge-Geräts, kompensieren. Alternativ würde die Clusterfähigkeit bei einer Internetanbindung des Edge-Geräts zusätzliche Stabilität bei Angriffen bringen. Im Vergleich zum Edge-Einsatz, steigt die Bedeutung dieser Anforderung zum verteilten Betrieb bei einem Einsatz im Data Lake signifikant. Zur Gewährleistung der Skalierbarkeit des Rohdatenspeichers, muss die TSDB den verteilten Betrieb auf mehreren Knoten unterstützen können, wobei es immer von der Anzahl der an die TSDB angeschlossenen Datenquellen abhängt. Auch die Unterstützung von Multi-Tier-Architekturen ist im Data Lake von Relevanz. Zum Beispiel hilft die Erfüllung dieses Kriteriums dabei, die Zeitreihendaten im Rohformat und die zugehörigen Metadaten von mehreren Vakuumhandhabungssystemen aus der Edge aggregiert zu speichern.

Analytics: Die TSDB in der Edge muss Analysefunktionalitäten für Zeitreihen nativ unterstützen, um die Zeit zwischen der Datenerfassung und der Datenanalyse zu verkürzen. Ein Datenwissenschaftler könnte Sensordaten direkt vom Edge-Gerät aus überwachen und analysieren. Das erste Kriterium ist die Bereitstellung grundlegender Analysefunktionalitäten wie z. B. SUM, MEAN. Ein weiteres Kriterium ist die Verfügbarkeit fortgeschrittener, statistischer Analysefunktionalitäten als Zeitreihenklassifizierung, Segmentierung oder Plotten. Neben der implementierten fortgeschrittenen Zeitreihen-Datenanalyse und -vorhersage könnten auch native maschinelle Lernfunktionalitäten wie Long-Short-Term-Memory (LSTM), Hidden Markov oder AutoRegressive Integrated Moving Average (ARIMA) bieten. Ein weiteres wichtiges Kriterium für die Echtzeit-Analyse ist die Unterstützung der Stream-Verarbeitung durch eine TSDB, sodass jeder neue Datenpunkt in Echtzeit analysiert werden kann, entweder durch die nativen Analysefunktionen der TSDB oder durch Datenanalysesoftware von Drittanbietern. Weitere hilfreiche Funktionalitäten sind eine grafische Benutzeroberfläche (GUI) zur Analyse der Daten und die Möglichkeit, die Zusammenhänge zwischen den Datenströmen zu verfolgen. Die GUI kann entweder ein von Datenbankanbieter bereitgestelltes Modul der TSDB sein oder durch eine externe aber offiziell unterstützte Anwendung bereitgestellt werden. Das letzte Kriterium enthält Alarme oder sonstige Ereignisse, die auf Basis der analysierten Daten gesetzt werden können und für die industrielle Prozessüberwachung wichtig sind. Nativ vorhandene Analytics-Funktionen wie Holt-Winters sind für die zeitnahe Vorhersage des Ausfalls in der Edge und ohne Drittsysteme essenziell wichtig. Im Vergleich zur Edge, werden im Data Lake möglichst viele Zeitreihendaten über längere Zeiträume gesammelt, mit dem Ziel, eine historische Datensammlung z. B. zum Energieverbrauch der Vakuumgreifsysteme aufzubauen. Datenwissenschaftler können diese Datensammlung im Data Lake z. B. mit Hilfe des maschinellen Lernens (ML) auswerten und so ein ML-Modell zu trainieren. Dieses ML-Modell beschreibt die in der Datensammlung identifizierten Korrelationen bzw. Muster zwischen einzelnen Messwerten und vorherzusagenden Zielgrößen. Dieser Schritt des Trainierens eines ML-Modells wird im Supervised Learning als Offline-Phase bezeichnet und erfolgt aufgrund der Größe der zu verarbeitenden Datensammlung mit Hilfe der Analyseanwendungen im Data Lake. In der anschließenden Online-Phase wird das antrainierte ML-Modell auf neue Daten angewendet. Dafür wird das Modell wieder in der Edge deployed, um möglichst echtzeitnah Vorhersagen für die Werte der Zielgrößen bei neu generierten Zeitreihendaten zu treffen. Somit hat diese Anforderung insgesamt eine ähnlich hohe Bedeutung in der Edge und im Data Lake. Allerdings müssen die Analysefunktionalitäten und die ML-Anwendung nicht unbedingt eigens durch die TSDB erfolgen. Dafür können im Data Lake spezielle Anwendungen und separate Frameworks zu Big Data verwendet werden, welche lesend auf die Daten in der TSDB zugreifen, diese weiter aufbereiten und analysieren. Deshalb sinkt bei ähnlich hoher Bedeutung dieser Anforderung im Data Lake die Notwendigkeit der nativen Unterstützung von Analytics-Funktionen innerhalb der TSDB. Somit wird die Relevanz der Erfüllung dieser Anforderung im Data Lake als neutral eingestuft.

Verfügbarkeit: Die TSDB in der Edge muss Funktionalitäten zur Verbesserung der Verfügbarkeit bei Störungsfällen bereitstellen. So kann eine TSDB Funktionalitäten zur Durchführung von Backups und Datensicherung bereitstellen. Die Alarmfunktionalität der TSDB, um bei Verhaltensänderungen des Systems zu reagieren und einen Alarm zu senden, trägt ebenfalls zur Verfügbarkeitssteigerung bei. Zusätzlich kann die Möglichkeit der Backup-Automatisierung das Risiko von Datenverlusten durch Abstürze oder Hardwareausfälle des Knotens weiter senken. Da in der Open-Source-Version von InfluxDB nur ein Knoten unterstützt wird und somit nur eine SD-Karte als Datenträger zur Verfügung steht, ist angesichts der Vielzahl der zu speichernden Datenpunkte pro Sekunde das Speicherplatzmanagement der TSDB von Relevanz, damit das Edge-Gerät nicht permanent bereinigt werden muss. Trotzdem erfordert die eingeschränkt erfüllte Anforderung der Verfügbarkeit und die nicht erfüllte Anforderung zum verteilten Betrieb (in der Open-Source-Version) den Einsatz der Enterprise-Version im Produktivbetrieb und den Einsatz von Zusatzsoftware zur Automatisierung der Backups. Die manuelle Backup-Funktionalität erfordert entweder einen stetigen Wartungsaufwand für jedes überwachte Handhabungssystem oder die Anschaffung einer Zusatzsoftware für die fehlende Automatisierung der Backups. Für den Einsatz im Data Lake sind die Backupfunktionalitäten und die Backupautomatisierung weniger wichtig, weil diese Aufgaben durch andere Anwendungen des Data Lake übernommen werden können und somit nicht durch die nativen Funktionen der TSDB abgedeckt werden müssen.

Sicherheit: Da Edge-Geräte über das Internet Zugriffe aus der Ferne erlauben, sind für TSDBs Funktionalitäten zur Verbesserung der Sicherheit relevant. Ein fortschrittliches Authentifizierungssystem ist auch im Edge-Betrieb durch die unmittelbare Nähe zur Maschinensteuerung wichtig. Darüber hinaus kann die Unterstützung digital signierter Zertifikate (z. B. mit Hilfe der Public-Key-Infrastruktur) das Sicherheitsniveau bei der Authentifizierung verbessern. Zudem tragen die Kommunikationsprotokollverschlüsselung und Datenverschlüsselung zur Sicherheit der gespeicherten Daten in einer TSDB während der Kurzzeitspeicherung und der weiteren Netzwerkübertragung in skalierbare Datenspeicher bei. Neben diesen Mechanismen zur Authentifizierung und Verschlüsselung kann zusätzlich Auditing die Sicherheit der Datenbank signifikant erhöhen. Dafür muss eine TSDB sämtliche Informationen über Zugriffe, Fehler, Konfigurationsänderungen und sonstige Ereignisse mit zugehörigen Zeitstempeln protokollieren. Ergänzend können bestimmte sicherheitsrelevante Ereignisse mit zusätzlichem Schreibschutz fälschungssicher gemacht werden. Zuletzt sind native und von der TSDB bereitgestellte Methoden im Umgang mit Angriffen in der Edge relevant. Um die Überwachung der Maschinenkomponenten nicht zu unterbrechen, muss eine TSDB während eines Denial of Service (DoS) Angriffs weiterhin die Aufnahme und Analyse der Sensordaten ermöglichen. Der Funktionsumfang der Datenbank kann insgesamt auf grundlegende Funktionen reduziert werden. Falls ein Angriff zum Absturz der TSDB oder des Edge-Geräts führt, sollte die TSDB über Funktionalitäten zur Wiederherstellung verfügen, um ohne Datenverlust wieder in einen bekannten und stabilen Zustand versetzt zu werden. Dieses Kriterium erfährt eine besondere Relevanz, falls das Edge-Gerät ohne weitere gesicherte Gateways direkt mit dem Internet verbunden wird und externen Angriffen ausgesetzt sein kann. Bei einem Produktiveinsatz des skizzierten Einsatzszenarios ist die Sicherheit von sehr großer Relevanz. Bei der Verbindung des Edge-Geräts mit dem Internet, sollte die TSDB angesichts inhärenter Sicherheitsschwächen der Edge-Geräte und veralteter Steuerungskomponenten keine zusätzlichen Sicherheitslücken bieten und alle relevanten Ereignisse (z. B. Zugriffe durch Mitarbeiter) protokollieren und die Fälschung der Protokolle erschweren. Eine ähnliche Einschätzung zur Wichtigkeit der Erfüllung der Sicherheitskriterien kann auch für den Betrieb der TSDB im Data Lake getroffen werden, allerdings sind im Data Lake andere Angriffsszenarios präsent. Aufgrund der Vielzahl von Benutzerrollen mit unterschiedlichen Berechtigungsstufen, die auf die Systeme im Data Lake zugreifen können, sind eine mächtige Benutzerverwaltung, genau wie das Auditing essenziell zur Unterstützung der Governance-Maßnahmen eines Data Lake. Mit Hilfe von Auditing durch die TSDB können im betrachteten Einsatzszenario die Richtigkeit der Energieverbrauchsdaten sichergestellt und etwaige Verfälschungen leichter entdeckt werden. Zusätzlich kann es im Kontext eines Data Lake wichtig sein, die erfolgten Zugriffe, um die Nachvollziehbarkeit der Datennutzung zu unterstützen. Dies ist bei der Vielzahl der möglichen Stakeholder in einem Data Lake relevant. Im Unterschied zur Edge kann das Auditing in einem Data Lake auch von anderen Teilsystemen übernommen werden. Letztlich ist zur Übertragung der Daten an die lokalen Analyse-Systeme (z. B. für weitergehende Analyse durch die Datenwissenschaftler) oder andere Cluster die Verschlüsselung der Datenübertragung ebenfalls wichtig.

1.5 Evaluation des Katalogs am Beispiel von InfluxDB

Der nächste Schritt ist die Anwendung der identifizierten Kriterien für die Evaluation des Katalogs und für die Messung des Erfüllungsgrades der Anforderungen durch InfluxDB.

1.5.1 Vorgehensbeschreibung zur Evaluation

Als Informationsquellen für die Evaluation werden die offizielle InfluxDB-Dokumentation, die Entwicklerdiskussionen und bereits umgesetzte oder noch offene Feature-Requests für InfluxDB auf Github und Webseiten mit Integrationsbespielen von InfluxDB verwendet. Zusätzlich wird angelehnt an Martinviita (2018) eine prototypische Installation zur Simulation des Edge-Anwendungsfalls aus ◉

Abb. 1.1durchgeführt, um eine valide Bewertung des industriellen Lastmanagements und Evaluation von Funktionen bei Angriffen zu ermöglichen. So wird das tatsächliche Systemverhalten von InfluxDB auf einem Einzelknoten beobachtet, wobei die TSDB auf einem begrenzt leistungsfähigen System, einer hohen Schreiblast bei gleichzeitig lesenden Datenanfragen ausgesetzt ist. Der von Martinviita (2018) beschriebene zusätzliche Lastausgleichsknoten ist nicht enthalten, um eine höhere Schreiblast für das Edge-Gerät zu erzeugen. Auch die Abbildung der Hierarchie mit mehreren Datenbanken unterschiedlicher Ebenen wird nicht berücksichtigt, da die Hierarchie in der Open-Source-Version nicht unterstützt wird (siehe ▶ Abschn. 1.5.2). Die Abb. 1.3 verdeutlicht schematisch den prototypischen Aufbau für den Edge-Einsatz. Der Quellcode des Prototyps ist auf Github verfügbar: https://github.com/Kypez/EdgeTSDB_Prototype_Simulated_Sensor_Ingestion_to_InfluxDB

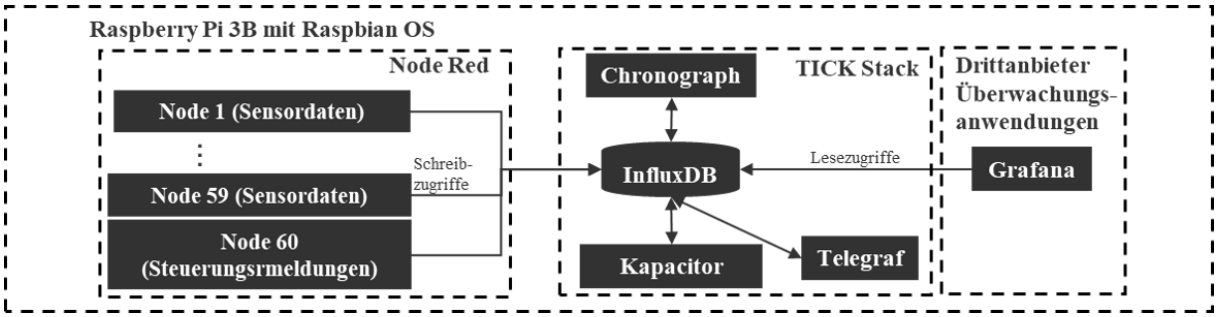

Abb. 1.3 Prototyp des Anwendungsfalls für den Betrieb von TSDB in der industriellen Edge

Der Prototyp basiert auf einem einzelnen Raspberry Pi 3B mit dem Raspbian OS, sowie TICK-Stack mit InfluxDB Version 1.7.4. Der Raspberry Pi wurde als Gerät ausgewählt, da es von einigen Anbietern industrieller Automation bereits verwendet wird (siehe ▶ Abschn. 1.1). Das Node-Red Framework wurde verwendet, um mehrere Datenquellen mit einer InfluxDB-Datenbank zu verbinden und das Schreiben verschiedener Sensordatentypen in InfluxDB durch entsprechende Ausgabenodes zu simulieren. So erlaubte der sog. Sensehat Node kontinuierlich schreibende Sensorströme zu simulieren (Node-Red 2018a), um primär die Multiwrite-Unterstützung, sowie das Workload-Handling mit folgenden Parametern zu testen:

- Es wurde eine Node-Red-Instanz mit 60 Ausgabe-Nodes erstellt, um die Anzahl der Sensoren einer modernen Werkzeugmaschine zu simulieren (DMG Mori 2016)

- Die Sensehat-Nodes wurden verwendet, um das Sensor-Erweiterungsmodul für den Raspberry Pi zu simulieren. Jeder Node kann simulierte Daten verschiedener Sensoren (z. B. Temperatur oder Beschleunigung) mit einer vorgegebenen Abtastrate erzeugen. Jeder datenerzeugende Node ist mit dem aufnehmenden InfluxDB-Node verbunden und ein simultanes Schreiben in die vorkonfigurierte InfluxDB-Datenbank wird ermöglicht.

- Von den 60 Ausgabe-Nodes wurden 54 Nodes so konfiguriert, um Temperaturdaten als exemplarische Sensormessung in industriellen Prozessen zu simulieren. Fünf Nodes lieferten die Daten zur Rotationsintensität eines Werkzeugs (z. B. einer Spindel in einer Werkzeugmaschine). Zusätzlich wurde ein Node konfiguriert, um Twitter-Nachrichten zu erfassen und in InfluxDB auf dem Edge-Gerät zu schreiben. Dadurch wurden Fehlermeldungen vom Überwachungssystem einer Werkzeugmaschine

simuliert (Martinviita 2018). Alle Sensordaten wurden mit einer Abtastrate von 1 Sekunde konfiguriert. Obwohl diese Abtastrate nicht als hochfrequent eingestuft werden kann, sind schnellere Abtastraten bei den verwendeten Sensehat-Nodes (Node-Red 2018a) nicht möglich. Sie entsprechen aber der Frequenz, die im Einsatzszenario vom Anbieter für Vakuumhandhabungssysteme verwendet wird.

- Die in InfluxDB gespeicherten Daten werden sporadisch durch Abfragen ganzer Zeitreihen von der Analyseanwendung Grafana abgefragt, um die Lese-/Schreibunabhängigkeit von InfluxDB zu simulieren und bei aktiven Schreibvorgängen gleichzeitige Abfragen zu testen.

- Mit der Lasttestanwendung Low Orbit Ion Cannon konnte ein DoS-Angriff auf den Prototypen über den von der TSDB verwendeten Port simuliert werden. Zusätzlich wurde die Stromzufuhr des Prototyps während des Betriebs mehrmals unterbrochen, um die Wiederherstellungsmöglichkeiten zu untersuchen.

1.5.2 Anwendung des Kriterienkatalogs

Zur korrekten Evaluation des Kriterienkatalogs ist es zunächst erforderlich, zwischen InfluxDB und dem TICK-Stack zu unterscheiden. Der TICK-Stack ist ein hochintegriertes modulares Paket für die Zeitreihenanalyse auf Basis von InfluxDB bis zur Softwareversion 2.0, das den Administrations-, Visualisierungs-, Analyse- und Verbindungsaufwand im praktischen Einsatz reduziert (InfluxData 2019). Somit erfüllt der integrierte Ansatz des Stacks zusätzliche Kriterien aus dem Katalog. Andererseits bieten andere TSDB-Anbieter keine Komplettlösungen mit vergleichbaren Funktionalitäten an und können daher nur eingeschränkt mit dem TICK-Stack verglichen werden. Folglich unterscheidet die Evaluation zwischen InfluxDB und dem TICK-Stack, um die Anwendbarkeit des Katalogs für andere TSDB zu ermöglichen. Die Evaluationsergebnisse sind in ⊙ Tab. 1.2 zusammengefasst. Die Kriterien werden durch zwei Typen von Kontrollkästchen (✓- ja/X - nein) vermerkt und die Klammern symbolisieren entweder eine begrenzte Unterstützung der Funktionalität oder wenn die Funktionalität nur in der Premiumversion enthalten ist. Falls ein Kriterium von InfluxDB unterstützt wird, wird es in der Regel auch vom TICK-Stack unterstützt. Dies wird mit dem Symbol ≡ dargestellt. Reicht ein Symbol nicht aus (z. B. Abfragesprache) wird die Ausprägung des Kriteriums als Text dargestellt. Das vollständige Quellenverzeichnis inkl. der Zuordnung im Katalog ist online verfügbar unter: https://bit.ly/3gsehSA

Tab. 1.2 Evaluation des Kriterienkatalogs

Anforderungen	Kriterien		InfluxDB	TICK Stack
Autarkie im Betrieb	Unabhängigkeit von externen Datenbanksystemen (z. B. Cassandra, HBase)		✓	✓
Modulare Erweiterungs-möglichkeiten	Open-Source-Entwicklungsmodell		✓	✓
	Premium Support		✓	✓
	Zustand nach dem Produktlebenszyklus		13 Hauptversionen	N.u.
Systemanforderungen	Hardwareanforderungen	CPU	2-4 Kerne	≡
		RAM	1 GB	
		IOPS	500	
	Datenträgeranforderungen (HDD/SSD)	HDD	X	X
		SSD	✓	✓
		SD-Card	✓	✓
	Erforderliches Betriebssystem	Debian Linux	✓	✓
		Ubuntu Linux	✓	✓
		RedHat Linux	✓	✓

	Unterstützte Cloudinfrastrukturanbieter	CentOS	✓	✓
		macOS	✓	✓
		Raspbian	✓	✓
		Windows	(✓)	(✓)
		Amazon AWS	(✓)	(✓)
		Google Cloud	(✓)	(✓)
		Microsoft Azure	(X)	(X)
Industrielles Last-management	Multiwrite-Unterstützung		✓	≡
	Unabhängigkeit zwischen Lesen und Schreiben		✓	≡
	Workload Handling		✓	≡
	Approximation (z. B. AQP)		X	X
	Caching		✓	≡
Interoperabilität	Schnittstellen/API	HTTP API	✓	≡
		JSON UDP	✓	≡
	Unterstützte Protokolle	HTTP Protokoll	✓	≡
		UDP Protokoll	✓	≡
		MQTT Protokoll	X	✓
		OPC UA Protokoll	X	(✓)
		AMQP Protokoll	X	✓
	Docker Unterstützung		✓	✓
	Konnektoren für Rechencluster		X	✓
	Konnektoren für Node-Red		✓	≡
	Konnektoren für Data Analytics Tools		X	✓
	Konnektoren für proprietäre IIoT-Plattformen		ThingWorx	≡
	Unterstützung von NoSQL Datenbanken		X	✓
	Unterstützung von verteilten Dateisystemen		(✓)	(✓)
Abfragesprache	SQL		X	X
	SQL-basierte Abfragesprache		InfluxQL	≡
	Nicht-SQL-basierte Abfragesprache		Flux	≡
Speicherplatz-management	Speichergranularität		ms, ns	
	Datenbank Engine		TSM Tree	
	Datenkompression		✓	≡
	Datenaufbewahrungsrichtlinien		✓	≡
Metadaten	Speicherung von Metadaten		(✓)	≡

	Möglichkeit zur Abfrage von Metadaten	✓	≡
Verteilter Betrieb	Clusterfähigkeit	(X)	(X)
	Multi-Node Unterstützung	(X)	(X)
	Unterstützung von Multi-tier Architekturen	X	X
Analytics	Grundlegende Analysefunktionen	✓	≡
	Fortgeschrittene Analytics Funktionen	✓	✓
	Machine Learning	X	✓
	Stream-Verarbeitung	✓	≡
	GUI für Analytics	X	✓
	GUI für Datenbeziehungen	X	X
	Alarmierung	X	✓
	Apache Kafka	X	✓
	Apache Zeppelin	X	(✓)
Verfügbarkeit	Backup-Funktionalität	✓	≡
	Automatische Backup	X	X
	Alarmierung	X	✓
Sicherheit	Authentifizierung	(✓)	≡
	Nutzung digitale Zertifikate	X	X
	Benutzerverwaltung	(✓)	≡
	Verschlüsselung des Kommunikationsprotokolls	✓	≡
	Datenverschlüsselung	X	X
	Auditing	✓	✓
	Methoden gegen Angriffe	✓	✓

Autarkie im Betrieb: InfluxDB wurde entwickelt, um Zeitreihendaten ohne Drittsysteme zu speichern, sodass der Konfigurationsaufwand von verteilten Rechenclustern wie Hadoop vermieden werden kann (Bader et al. 2017). Damit erfüllt InfluxDB die Anforderung der Autarkie im Betrieb, da Sensordaten zusammen mit Metadaten und Nicht-Zeitreihendaten (z. B. Textnachrichten als Zeichenketten für Fehlermeldungen) maschinennah in einer gekapselten Edge-Umgebung vollständig gespeichert werden können. All diese Datentypen wurden in dem beschriebenen Prototyp in InfluxDB gespeichert.

Modulare Erweiterungsmöglichkeiten: Da InfluxDB als Open-Source-Lösung gekennzeichnet ist und durch die Community mit spezifischen Erweiterungen und Plugins versorgt werden kann, sind modulare Erweiterungsmöglichkeiten grundsätzlich vorhanden. Für bestimmte Funktionalitäten ist es möglich, kommerziellen Premium-Support vom Datenbankanbieter zu erwerben. Andererseits werden einige Kriterien wie z. B. die Multi-Node Unterstützung nur in der Premiumversion angeboten. InfluxDB wurde ursprünglich 2013 veröffentlicht und steht damit am Anfang des Lebenszyklus. Infolgedessen ist die Häufigkeit von Neuerscheinungen hoch. Im Jahr 2018 gab es 13 Major Releases, die in der offiziellen Dokumentation erwähnt werden. Aus industrieller Sicht erhöht die hohe Anzahl an Hauptversionen den Wartungsaufwand und kann sich negativ auf das Vertrauen in Stabilität und Reife von InfluxDB auswirken.

Systemanforderungen: Um auf einem einzelnen Knoten mit weniger als 5000 Schreibvorgängen pro

Sekunde betrieben zu werden, benötigt InfluxDB offiziell 2-4 Kerne und 2-4 GB RAM oder eine 500 IOPS Hardwareleistung auf einer ARM- oder x86-Architektur. Unsere prototypische Implementierung mit dem Raspberry Pi 3B stellt 4 Kerne bereit, verfügt jedoch nur über 1GB RAM. Zudem benötigt InfluxDB offiziell SSD-Speicher. Trotzdem hat unser Prototyp die Sensordaten für 60 Sensoren im Sekundentakt problemlos auf eine langsamere SD-Karte bei nur 1 GB RAM schreiben können. Das bedeutet, dass InfluxDB keine In-Memory-Datenbank ist und die Leistung wesentlich von der Festplattenleistung des Edge-Geräts abhängt. Für die Integration in einem Data Lake Cluster stellt InfluxDB ähnliche Systemanforderungen an die Hardware, Datenträger und das Betriebssystem. Unter Berücksichtigung des Betriebssystems stellt InfluxData Binärdateien für unterschiedliche Linux-Distributionen (Ubuntu, Debian, RedHat, CentOS) und MacOS zur Verfügung. Die Unterstützung für eine Cloud-Installation auf AWS und der Google Cloud ist gegeben, beschränkt sich allerdings nur auf die Premiumversion. Die Unterstützung von Windows durch InfluxDB und TICK Stack ist bis auf Telegraf lediglich experimentell und ohne offiziellen Support. Die Installation in der Azure-Cloud ist noch nicht möglich und kann nur über die Einrichtung einer virtuellen Maschine (VM) als Behelfslösung implementiert werden. Aus industrieller Sicht ist die Linux-Unterstützung aufgrund ihres Open-Source-Modells, der Vielzahl der unterstützten Treiber und der Verfügbarkeit verschiedener Echtzeit-Distributionen, die sie auf Edge-Geräten zur Steuerung industrieller Prozesse einsetzen, unerlässlich (Mueller et al. 2017). Folglich erhöht die Unterstützung von Linux-Distributionen durch InfluxDB die Kompatibilität der Datenbank mit verbreiteten Edge-Technologien und Komponenten eines Data Lake. Da es auch viele Windows-basierte Edge-Geräte und Industriecomputer gibt, wäre eine verbesserte Unterstützung von Windows ein nützliches Feature für den zukünftigen industriellen Einsatz von InfluxDB. Die Systemanforderungen sind für InfluxDB und den TICK-Stack identisch. Für die Integration von InfluxDB in einen cloudbasierten Data Lake wäre die Unterstützung der Azure Cloud zusätzlich zur Unterstützung der anderen beiden Anbieter wünschenswert.

Industrielles Lastmanagement: Während des einstündigen Lastmanagementtests lief InfluxDB stabil und problemlos – es gab keine Datenverluste oder andere Probleme in einem der ausgewerteten Kriterien. Das parallele Schreiben von 59 simulierten Sensor- und einem Textdatenstrom führte zu einer Prozessorlast von ca. 30%. InfluxDB und TICK-Stack erfüllen beide die industriellen Anforderungen des industriellen Lastmanagements. Aus der Dokumentation ist nicht zu entnehmen, dass InfluxDB die AQP-Technik unterstützt, um ungefähre Abfrageergebnisse in gegebener Zeit zu liefern. Die Implementierung dieser Funktion könnte für den industriellen Einsatz mit eingeschränkter Leistung und die Implementierung von Echtzeitanalysen nützlich sein. Andererseits wird die Caching-Funktionalität von der Datenbank Engine der InfluxDB unterstützt. Obwohl InfluxDB keine In-Memory-Datenbank ist, ermöglicht die Caching-Technik die Speicherung von Teilen der Daten im Arbeitsspeicher. Insgesamt kann die Implementierung der In-Memory-Funktionsweise eine mögliche Empfehlung für die zukünftige Entwicklung von InfluxDB zur Verbesserung der Abfragezeiten sein, da InfluxDB in bestimmten Tests hinter den Ergebnissen anderer TSDBs liegt (Burdack et al. 2018). Ein prototypisches Deployment von InfluxDB in einem Data Lake, um eine vielfach höhere Anzahl paralleler Schreib- und Lesezugriffe zu simulieren, haben wir nicht durchgeführt.

Interoperabilität: InfluxDB stellt APIs zum Lesen und Schreiben von Daten, basierend auf den Protokollen HTTP und UDP, zur Verfügung. Es existieren zahlreiche durch die Open-Source-Community entwickelte Client-Bibliotheken für R, Python, Java, Node.JS und weitere Sprachen. Wichtige Maschinenkommunikationsprotokolle wie MQTT oder AMQP werden von Telegraf unterstützt. Das industrielle Kommunikationsprotokoll OPC UA wird nicht nativ unterstützt und muss für entsprechende Maschinen selbst implementiert oder von externen Anbietern hinzugekauft werden. InfluxDB unterstützt verschiedene Client-Bibliotheken für Software wie Matlab oder R. Durch einen offiziellen InfluxDB Node wird die Unterstützung von Node-Red angeboten. Der Node ermöglicht es, verschiedene Systeme und Datenquellen von Drittanbietern über eine grafische Verknüpfung mit InfluxDB zu verbinden. Die Unterstützung für das Distributed Stream Processing Framework Kafka erfolgt über Telegraf. Offizielle Docker-Builds für InfluxDB sind ebenfalls in der Bibliothek für Docker-Builds aufgeführt. Seit der Version 8.4 wird InfluxDB zudem von der IIoT-Plattform ThingWorx als zusätzlicher Persistenz-Provider für Zeitreihendaten unterstützt. Für andere IIoT-Plattformen wie Predix oder MindSphere werden keine nativen Konnektoren angeboten, aber es gibt entsprechende Node-Red-Konnektoren, welche die Anbindung mit diesen IIoT-Plattformen teilweise kompensieren. Insgesamt fehlt InfluxDB für den Edge-

Einsatz die Unterstützung einiger Interoperabilitätskriterien, aber der TICK-Stack vervollständigt sie, mit Ausnahme der OPC-UA-Unterstützung. Ebenso ist die Erfüllung der Interoperabilitätskriterien im Kontext eines Data Lake eingeschränkt. Entsprechende Konnektoren für verteilte NoSQL-Cluster werden vom Datenbankanbieter zwar angeboten, dennoch fehlen Werkzeuge für verteilte Dateisysteme wie HDFS. Da InfluxData keine angepassten Werkzeuge für die Konvertierung der Daten anbietet, müssen Nutzer eine Konvertierung der Daten in eine Textdatei selbst implementieren.

Abfragesprache: InfluxDB und TICK-Stack verwenden eigene SQL-basierte Abfragesprache InfluxQL und Skriptsprache Flux. Beide Sprachen enthalten nützliche Abfragebefehle zur Analyse von Zeitreihenmessungen, wobei Flux speziell zur Vereinfachung von Self-Service entwickelt wurde und die Möglichkeit bietet Join-Operationen durchzuführen und mathematische sowie statistische Berechnungen (z. B. Kovarianz) über mehrere Zeitreihen durchzuführen.

Speicherplatzmanagement: InfluxDB verfügt über einige Speicherplatzmanagement-Features und erfüllt die Anforderung vollständig. Die unterstützte Speichergranularität ermöglicht es, Sensordaten in Millisekunden, Mikrosekunden oder Nanosekunden als Zeitintervalle zu speichern. InfluxDB verwendet eine Engine, die Komprimierung unterstützt und dem Log-structured merge-tree ähnlich ist. Ein weiteres Kriterium zur Umsetzung von Datenaufbewahrungsrichtlinien, die es einer TSDB ermöglichen, Daten nach definierten Zeiträumen automatisch zu löschen, um den Administrationsaufwand zu reduzieren, ist in InfluxDB implementiert. Der Prototyp konnte die Effizienz der TSDB beim Speicherplatzmanagement zeigen. Nach Ablauf des einstündigen Versuchs mit 60 Datenströmen mit einer Frequenz von 1 Hz betrug das Datenvolumen 10,7 MB. Wenn die Frequenz auf KHz skaliert wird, beträgt die Größe der in der Datenbank gespeicherten Daten 1070 MB/h oder 8560 MB/Arbeitsschicht. Diese Datenmengen können auf einem Edge-Gerät mit gewöhnlichen Datenträgern kurzzeitig gespeichert und an zentrale Datenplattformen übertragen werden, wenn die Maschine z. B. nicht läuft. Im Kontext eines Data Lake haben die verwendete Datenbank-Engine und die Komprimierung der Zeitreihen einen großen Einfluss auf eine kosteneffiziente Langzeitspeicherung der historischen Rohdaten.

Metadaten: InfluxDB unterstützt die Metadatenspeicherung durch Tags und Felder und kann so zusätzliche Informationen z. B. über den Sensor oder über die Dateneinheit liefern. Im Vergleich miteinander sind die Felder ein Pflichtbestandteil der Datenstruktur von InfluxDB und stellen die Beschreibung für eine Reihe von Werten dar. Felder sind Zeichenketten und werden nicht indiziert, was sich negativ auf die Performance der Abfragen auswirkt. Tags sind optional, allerdings werden sie indiziert, weshalb sie performantere Abfragen liefern und zur Verwendung bei häufigen Abfragen von Metadaten empfohlen werden. Tags und Felder sind jedoch dazu gedacht, Metadaten für jeweils nur eine Zeitreihe bereitzustellen. Damit ist die Speicherung aggregierter Metadaten ohne ein externes Datenbanksystem, sowohl für die Edge als auch für den Data Lake als limitiert zu bewerten. Besonders im Data Lake, wenn eine Vielzahl von Zeitreihen in einem Rohdatenspeicher zusammenlaufen, ist es ohne einen externen Datenkatalog nicht möglich die zusammengefassten Informationen zu den unterschiedlichen Edge-Geräten in InfluxDB sinnvoll zu speichern. Allerdings können die in InfluxDB gespeicherten Metadaten ohne Einschränkungen über vorhandene Schnittstellen ausgelesen werden. Damit können die Metadaten im Datenkatalog gespeichert und dort durch weitere Metadaten (siehe ▶ Abschn. 1.4.2 – Metadaten) ergänzt werden.

Verteilter Betrieb: InfluxDB kann mehr als eine Instanz verwenden, aber dieses Kriterium ist der Premiumversion (InfluxDB Enterprise) vorbehalten (Bader et al. 2017). Dasselbe gilt für die Multi-Node-Unterstützung. Der Mangel an Multi-Node-Unterstützung schränkt auch die Skalierbarkeit von InfluxDB und die Verwendung auf Edge-Geräte-Clustern ein. Vom Datenbankanbieter wird lediglich das Modul Relay bereitgestellt, das eine Multi-Node-Architektur zur Verbesserung der Verfügbarkeit ermöglicht. Allerdings bietet das Modul keine Möglichkeit zum verteilten Betrieb der TSDB, sondern ermöglicht nur deren Replikation. Zudem wurde das Modul seit 2016 nicht mehr aktualisiert. Darüber hinaus fehlt InfluxDB auch die Unterstützung von mehrschichtigen Hierarchien, die von Martinviita (2018) als eines der industriellen Szenarien für TSDB im IIoT erwähnt wird. InfluxDB kann daher nicht zwischen datengenerierenden Maschinen in der Edge priorisieren und in mehrschichtige Architekturen mit aggregierten Daten integriert werden. Beim Einsatz im Data Lake kann InfluxDB aufgrund der erforderlichen Skalierbarkeit und der Clusterfähigkeit nur in der Premiumversion verwendet werden. Der

Einsatz der nicht-kommerziellen Version von InfluxDB ist mit Relay auf mehreren Einzelinstanzen zwar möglich, widerspricht allerdings dem Zweck eines skalierbaren Data Lake. InfluxDB Enterprise erfüllt hingegen die Kriterien zum verteilten Betrieb im Data Lake.

Analytics: InfluxDB unterstützt die grundlegenden Analysefunktionen wie z. B. SUM, MEAN und ähnliche. Die erweiterten Analysefunktionalitäten werden durch die native Unterstützung von Holt-Winters für Zeitreihenprognosen von InfluxDB abgedeckt. Hier ist eine zukünftige Unterstützung weiterer Zeitreiheanalysemodelle wie ARIMA empfehlenswert. Um maschinelle Lernmodelle für Zeitreihenvorhersagen anwenden zu können, ist der TICK-Stack erforderlich, da Kapacitor Konnektoren für maschinelle Lernsoftware oder proprietäre maschinelle Lerndienste wie Azure ML von Microsoft bereitstellt. Die Echtzeit-Stream-Verarbeitungsmaschine wird von Kapacitor bereitgestellt. Somit ist es möglich, eine Sensordatenmessung in InfluxDB zu abonnieren, um jeden neuen Datenpunkt in Echtzeit für die Überwachung und Analyse zur Verfügung zu stellen. Der TICK-Stack und die nativ unterstützte Integration von Grafana bieten eine GUI für Analysen und eine Möglichkeit, datenbasierte Warnmeldungen zu erstellen (InfluxData 2019a). Die Funktionalität für Zusammenhänge zwischen Datenströmen wird von InfluxDB oder TICK-Stack Modul aktuell nicht bereitgestellt. Da bei einem Einsatz im Data Lake die nativen Analysewerkzeuge durch externe Anwendungen mit größerem Analysefunktionsumfang ersetzt werden können (Mathis 2017; Gröger 2018), ist die Unterstützung entsprechender Konnektoren und Schnittstellen für Analysewerkzeuge relevant. Zur Konnektivität mit den Clusteranwendungen des Data Lake bietet Telegraf Module für den Datenaustausch mit Streaming-Komponenten eines Clusters wie Apache Kafka. Zudem fanden im April 2020 Abnahmetests des InfluxDB-Integrationsmoduls für Apache Zeppelin statt. Mit Hilfe der Unterstützung von Apache Zeppelin wird es für Datenwissenschaftler möglich sein, mittels einer Notizbuch-Anwendung Abfragen zur Datenanalyse direkt an InfluxDB als Rohdatenquelle zu senden.

Verfügbarkeit: Die Backup- und Wiederherstellungsfunktionalität wird durch die nicht-kommerzielle Lizenz von InfluxDB unterstützt, aber automatische Backups sind nicht über InfluxDB möglich und erfordern Drittanbietersoftware oder Backups auf Betriebssystemebene des Edge-Geräts oder durch externe Anwendungen des Data Lake. Die Systemwarnungen werden nicht von InfluxDB bereitgestellt, aber Kapacitor ermöglicht es systembezogene Warnungen zu senden. Durch das vom Datenbankanbieter bereitgestelltes Modul Relay kann auch mit der nicht-kommerziellen Lizenz von InfluxDB eine hochverfügbare Architektur geschaffen werden, indem mit Hilfe von Relay ausgefallene Datenbankinstanzen ermittelt und mit Backups der funktionierenden Datenbankinstanzen ersetzt werden können.

Sicherheit: InfluxDB unterstützt eine passwortgeschützte Authentifizierung, ebenso wie die Benutzerverwaltung, die unter Berücksichtigung der unterschiedlichen Benutzerrollen in industriellen Anwendungsfällen wichtig ist. Allerdings erlaubt das Authentifizierungskonzept es nicht unterschiedliche Rollenkonzepte anzulegen. Die Berechtigungsstrukturen werden für einzelne Benutzer definiert. Zudem existiert bislang keine offizielle Unterstützung für digitale Zertifikate (z. B. innerhalb der Public-Key-Infrastructure) bei der Authentifizierung. Weiterhin ist die Authentifizierung im Auslieferungszustand abgeschaltet und muss zuerst eingerichtet werden. Zusätzliche Möglichkeiten der Benutzerverwaltung sind nur in der kommerziellen Version des Kapacitor enthalten. Die grundlegende Unterscheidung zwischen Admin- und Non-Admin-Rollen wird von der nicht-kommerziellen Version unterstützt. Diese Unterscheidung erlaubt es nicht die Berechtigung für einzelne Datenbanken oder unterschiedliche Zeitreihen pro Datenbank einzuschränken, was beim Zugriff von externen Dienstleistern von Vorteil wäre. Verschlüsselte Kommunikationsprotokolle (HTTPS) werden von InfluxDB unterstützt. Die Möglichkeit zur Verschlüsselung der Daten besteht allerdings nicht. Zur Verschlüsselung von Sensordaten muss somit die Datenträgerverschlüsselung durch Fremdsoftware oder das Betriebssystem des Edge-Gerätes verwendet werden. Die Protokollierung diverser Ereignisse (Auditing) wird durch InfluxDB vollständig unterstützt und der Detailgrad kann über die Konfiguration eingestellt werden. Am Prototyp wurde allerdings festgestellt, dass das Auditing mehr Speicherplatz verbraucht als die Zeitreihen und so die Sicherheit gegen den Wartungsaufwand des Edge-Geräts entgegengesetzt wird. Insgesamt bietet InfluxDB keinen besonderen Schutz für die Protokolldateien und die Konfiguration der Protokollierung, sodass ein Benutzer mit Administratorrechten auf dem Edge-Gerät in der Lage ist diese zu verfälschen. Deshalb wird das Sicherheitskriterium des Auditings von InfluxDB eingeschränkt erfüllt. Während eines simulierten DoS-Angriffs auf den Prototypen blieb InfluxDB

allerdings funktionsfähig, solange das Edge-Gerät stabil lief. Lediglich die lesenden Abfragen der Zeitreihen haben länger gedauert. Zudem bietet InfluxDB nach einem Absturz die Möglichkeit die die Daten wiederherzustellen um die Zeitreihen fortzusetzen. Nach einem absichtlich herbeigeführten Absturz des Edge-Geräts waren die Zeitreihen trotzdem lesbar und konnten durch neue weitere Sensordaten fortgesetzt werden. Insgesamt verhält sich InfluxDB bei Angriffen relativ robust, bietet allerdings keine besonderen Funktionen, um z. B. das Sicherheitsniveau des Edge-Knotens zusätzlich zu erhöhen. Sollte das Edge-Gerät direkt mit dem Internet (z. B. durch einen böswilligen Mitarbeitereingriff) verbunden werden, erlauben die Werkseinstellungen von InfluxDB keinen sicheren Produktivbetrieb. Für den Produktivbetrieb wird empfohlen, die TSDB nur in einem internen Netzwerk zu betreiben. Auch die Konfiguration des Edge-Geräts spielt bei der Sicherheit eine große Rolle, da die Ereignisprotokolle für das Auditing bei falsch konfigurierten Edge-Geräten verfälscht werden können. Außerdem nehmen die Ereignisprotokolle zusätzlichen Speicherplatz auf dem Edge-Gerät ein. Der Umfang der durch das Handhabungssystem auf dem Edge-Speicher generierten Sensordaten ist geringer als der Datenumfang der für das Auditing erforderlichen datenbankinternen Ereignisprotokolle. Somit führt das Auditing zum erhöhten Wartungsaufwand des Edge-Geräts. Im Data Lake sind die Speicherplatzrestriktionen nicht so kritisch. Somit können die Auditingprotokolle von InfluxDB aufgezeichnet werden und helfen die Sicherheit des Data Lake zu erhöhen. Da die Zugriffe auf die TSDB von unterschiedlichsten Nutzern und unterschiedlichen Anwendungen stammen können, sind potentielle Sicherheitsrisiken wie z. B. Datenmanipulation oder Datendiebstahl bei einem Einsatz im Data Lake möglich.

Zusammenfassend lässt sich feststellen, dass InfluxDB in der Edge für den Produktiveinsatz bei IoT-Systemen mit vielen Sensoren durchaus taugt. Dennoch sollte die Praxis den Konfigurationsaufwand nicht außer Acht lassen. Bei dem aktuellen Sicherheitsniveau lassen sich die Vorteile der Interoperabilität nur im Zusammenhang mit unternehmensinternen Systemen verwenden. Bei einem Produktiveinsatz mit Internetanbindung des Edge-Geräts sollte die Konfiguration des Betriebssystems und der TSDB genauestens geprüft werden und weitere Softwaresysteme für die automatisierte Sicherung der Ereignisprotokolle für das Auditing erwogen werden. Für einen sinnvollen Einsatz im Data Lake, kann InfluxDB erst in der Premiumversion verwendet werden, da erst diese Version der TSDB Funktionen für einen verteilten Betrieb bietet.

1.6 Diskussion und Ausblick

1.6.1 Fazit

Das Hauptziel des vorliegenden Artikels ist es, Praktikern eine Liste relevanter Anforderungen auf Basis von funktionalen Kriterien für TSDBs zu präsentieren, die Spezifika des industriellen TSDB-Einsatzes in der Edge und im Data Lake berücksichtigen. Erstens umfassen die Ergebnisse einen Kriterienkatalog als Unterstützung zur Umsetzung von neuartigen Anwendungsfällen und zum Start von IIoT-Projekten. Der Katalog bietet die Grundlage für weitere Evaluationen von TSDBs und vereinfacht den Auswahlprozess, sowie das Benchmarking für zahlreiche auf dem Markt verfügbare TSDB-Technologien für unterschiedliche Einsatzszenarien. Produzierende Unternehmen können die Kriterien und das Einsatzszenario für den Aufbau einer Edge-Überwachung ihrer eigenen Produktion oder den Aufbau eines eigenen Data Lake nutzen. So können z. B. die Unternehmen aus dem Maschinen- und Komponentenbau auf Basis des beschriebenen Prototyps und des Kriterienkatalogs eigene digitale Lösungen zur Speicherung und Analyse der Sensordaten in der Edge bei zeitkritischen Prozessen entwickeln. Zweitens bietet die Anwendung des Kriterienkatalogs auf InfluxDB, sowie die Verknüpfung mit dem Einsatzszenario eine Entscheidungsunterstützung beim Einsatz von InfluxDB. Die identifizierten Limitationen dieser TSDB, zeigen implementierungsbezogene Handlungsfelder bei der Umsetzung industrieller datengetriebener Ende-zu-Ende Lösungen. Drittens bietet die Anwendung des Kriterienkatalogs einen Mehrwert für Softwareunternehmen, die TSDB-Lösungen entwickeln. Die identifizierten Kriterien können bei zukünftigen Updates von TSDBs berücksichtigt werden, wenn spezifische Releases oder Forks für IIoT und industrielle Edge-Anwendungen oder für industriell genutzte Cluster für die Analyse von Big Data, entwickelt werden.

Die Ergebnisse des Artikels tragen auch zur Ermittlung des aktuellen Stands der Technik bei

Zeitreihendatenbanken bei. Aufgrund des Neuheitsgrads des Einsatzszenarios stellt sich heraus, dass die aktuellen TSDBs sich wegen ihrer Leistungsfähigkeit sehr gut für zur Kurzzeitspeicherung und Echtzeitanalyse von Werkzeugmaschinendaten eignen. Bei spezifischen Anwendungsfällen (Martinviita 2018) kann insbesondere vorgeschlagen werden, die derzeit noch nicht entwickelten Funktionalitäten für die Verbesserung der Abfragezeit (z.B. AQP), die Unterstützung von Hierarchien und die native (durch den Datenbankanbieter entwickelte und gewartete) Unterstützung von Industrieprotokollen wie OPC UA insbesondere für die Edge zu implementieren. Für den Data Lake, sind die Interoperabilität und die Metadatenspeicherung besonders relevant. Für die Unterstützung der Interoperabilität mit zahlreichen Anwendungen eines Data Lake ist eine Vielzahl an bereitgestellten APIs erforderlich, um die Anzahl der zu entwickelnden Schnittstellen zu reduzieren. Das Metadatenmanagement ist maßgeblich für die Auffindbarkeit und Verwendung der Zeitreihendaten im Data Lake relevant und kann die Implementierung einer Metadatenanalyse unterstützen.

Zusätzlich gibt die Anwendung des Katalogs auf das Referenzeispiel InfluxDB einen Überblick über die fehlenden Funktionen dieser Referenzlösung für den Produktiveinsatz. Die nicht oder nur zum Teil umgesetzten Anforderungen bieten Praktikern eine Orientierung worauf bei dem Einsatz der industriellen Edge zur Maschinenüberwachung besonders zu achten ist. Die Vielzahl der industriellen Anwendungsfälle und die damit verbundene Vielzahl der geeigneten Maschinenprotokolle (z. B. OPC UA, Modbus oder MQTT) erfordert allerdings einen Entwicklungs- und Integrationsaufwand bei dem industriellen Edge-Einsatz von aktuellen TSDBs. Auch der Konfigurationsaufwand von TSDBs auf Edge-Geräten und die Kombination mit zusätzlicher Anwendungssoftware zum Ausgleich von nicht erfüllten Kriterien bilden eine Empfehlung für die Praxis. Dennoch sollte InfluxDB in der eingesetzten Version 1.7.4 nicht für sensible Anwendungen vor allem nicht bei direkter Internetanbindung des Edge-Gerätes eingesetzt werden. Vor allem die Standardeinstellungen von InfluxDB (keine Authentifizierung eingerichtet, diverse offene Ports vorhanden, keine Ereignisprotokollierung aktiviert) bieten keinen ausreichenden Schutz vor externen Angriffen. Falls die Konfiguration der TSDB korrekt durchgeführt wird, hängt die Sicherheit maßgeblich von den Einstellungen des Edge-Geräts ab. Als Referenztechnologie unter TSDBs bessert InfluxDB die inhärenten Schwächen der Edge-Geräte nicht aus. Obwohl InfluxDB relativ einfach zu bedienen ist und von der Leistungsfähigkeit sich hervorragend für die Maschinenüberwachung eignet, bleiben die umfassende Konfiguration und Kombination mit Zusatzsoftware kritisch für den Erfolg des Einsatzes in Edge-Lösungen. Die besondere Berücksichtigung dieser beiden Punkte stellt eine wichtige Erkenntnis dar.

Eine ähnliche Einschätzung zum Reifegrad von InfluxDB kann auch für den Einsatz im Data Lake getroffen werden. Der vom Datenbankanbieter für die Open-Source-Version der TSDB zugelassene Funktionsumfang macht trotz der umfangreichen Interoperabilität und effizienter Speichermöglichkeiten für Zeitreihendaten im Rohformat eine sinnvolle Anwendung im Data Lake aufgrund der nicht vorhandenen Unterstützung für den verteilten Betrieb, nicht sinnvoll. Die nativen Funktionen zur Metadatenspeicherung sind bei InfluxDB etwas eingeschränkt, wobei die Metadaten problemlos über die implementierten Schnittstellen abgefragt und im Datenkatalog gespeichert werden können. Für ein umfangreiches Metadatenmanagement sowie umfassende Verfügbarkeits- und Sicherheitsfeatures sind externe Anwendungen aus der Systemumgebung des Data Lake erforderlich. Ergänzt um die Funktionalitäten zum verteilten Betrieb, stellt die Premiumversion von InfluxDB aufgrund der hohen Speichereffizienz trotzdem eine relevante Technologie dar, um die historische Rohdatenspeicherung für Zeitreihen kosteneffektiv umzusetzen.

1.6.2 Ausblick und Limitationen

Die Ergebnisse dieses Artikels bilden die Grundlage für weitere Forschung, um weitere Erkenntnisse im Hinblick auf Anforderungen an Zeitreihendatenbanken in der industriellen Edge zu gewinnen. Die Ableitung relevanter Anforderungen aus der wissenschaftlichen Literatur und der Dokumentation kann in weiteren empirisch Untersuchungen erweitert, bewertet und priorisiert werden. Empirisch fundierte Priorisierung der identifizierten Kriterien ermöglicht ein zukünftiges Gesamtranking der identifizierten Anforderungen. Für die zukünftige Priorisierung können Priorisierungsmethoden wie der Analytical Hierarchy Process (AHP) verwendet werden. Zur weiteren Evaluation sollten Maschinenbediener, die direkt mit Werkzeugmaschinen und Edge-Geräten im Shopfloor arbeiten, sowie Netzwerkspezialisten aus dem industriellen Umfeld befragt werden. Eine interessante Erkenntnis liefert zudem die geringe

Anzahl der identifizierten Sicherheitskriterien in der wissenschaftlichen Literatur, obwohl im IIoT die Sicherheit eine sehr große Relevanz hat. Der Mangel an spezifischen Sicherheitskriterien für TSDBs bietet Potential für weitere Untersuchungen. Unter der Annahme, dass InfluxDB eine technisch fortgeschrittene Referenzlösung für Zeitreihendaten ist und trotzdem die Sicherheitsanforderung, Die Metadatenanforderung oder die Verfügbarkeitsanforderung nur teilweise erfüllt, ist die Analyse von sinnvollen Kombinationen aus TSDB und zusätzlichen Unternehmenssoftwaresystemen zur Kompensation der nicht vollständig erfüllten Anforderungen und Entwicklung neuer Komplettlösungen für die Industrie interessant. In diesem Zusammenhang sind weitere Untersuchungen sinnvoll, um Leitfäden zur Umsetzung einer Gesamtlösung für eine möglichst effiziente und echtzeitnahe Maschinenüberwachung zu entwickeln. Die Entwicklung weiterer Referenzarchitekturen könnte der Praxis eine Entscheidungsunterstützung bieten, welche Kriterien eine TSDB idealerweise in der Edge und im Data Lake erfüllen soll und welche Kriterien durch zusätzliche Anwendungssoftware erfüllt werden können. Der Artikel berücksichtigt nur rudimentär die Kombination der TSDB in der Edge mit verteilten Systemen zur Verarbeitung von Datenströmen wie z. B. Apache Flink und Apache Spark (Gehring et al. 2019). Eine aufbauende Evaluation zum Potential einer sinnvollen Kombination aus marktreifen TSDBs und Systemen zur Verarbeitung von Datenströmen im Kontext einer echtzeitnahen Analyse von Maschinendaten wäre für die Praxis relevant und könnte auf den Ergebnissen dieses Artikels aufbauen. Die aktuelle Evaluation umfasst nur InfluxDB als exemplarische TSDB und ein Benchmark zwischen mehreren TSDBs wäre daher eine wertvolle Erweiterung. Ein auf dem Katalog des Artikels basierender Vergleich über die Erfüllung funktionaler Kriterien mit weniger bekannten aber vom Funktionsumfang vergleichbaren TSDB-Technologien wie z. B. HarperDB oder crateDB könnte nützlich sein. Obwohl HarperDB in den aktuellen Rankings nicht berücksichtigt wird, gilt diese Datenbank als spezifische IoT-Lösung für den Edge-Einsatz (HarperDB 2019) und bildet somit eine Alternative zur Umsetzung der Maschinenüberwachung in der Edge. Die nicht-kommerzielle Lizenz von crateDB (crate.io 2020) ist clusterfähig und scheint somit eine geeignete Alternative für den Einsatz im skalierbaren und clusterfähigen Data Lake zu sein. Darüber hinaus sollte der Kriterienkatalog auf andere gängige TSDBs aus dem Ranking von db-endinges.com (2019) angewendet werden.

Weiterhin kann die zukünftige Forschung auf dem konzeptionellen Ergebnis zum sinnvollen TSDB-Einsatz in industriell genutzten Data Lake aufbauen und weitere Möglichkeiten für TSDBs bei der Umsetzung des Data Lake Konzepts evaluieren. Wir empfehlen, die Potentiale von TSDBs zur Unterstützung eines ganzheitlichen Metadatenmanagements im Data Lake praktisch zu prüfen. Zudem kann das Kosten-Nutzen-Verhältnis unterschiedlicher TSDB-Technologien für die Speicherung großer Mengen an Zeitreihendaten im Rohformat weiter evaluiert werden. Da die existierenden und vom Datenbankanbieter offiziell unterstützen Konnektoren im Zusammenspiel mit den typischen Anwendungen eines Data Lake nicht betrachtet wurden, ist dazu ebenfalls eine vertiefende Analyse auf Basis der durchgeführten Untersuchung denkbar. Einzelne Anforderungen wie die Abfragesprache erfordern eine detailliertere Analyse, um die Komplexität und die Umsetzbarkeit der Abfragen im Vergleich mit der etablierten und mächtigen SQL-Abfragesprache in typischen industriellen Anwendungsfällen zu messen. Eine Untersuchung ob der Funktionsumfang von InfluxDB durch die neuen Abfragesprachen gegenüber konkurrierenden TSDBs auf SQL-Basis eingeschränkt ist oder tatsächlich Vorteile bringt, sollte zukünftig durchgeführt werden.

Obwohl unser Benchmark eine prototypische Implementierung zum Schreiben von Sensordaten in InfluxDB auf einem Raspberry Pi 3B beinhaltet, sollten die gleichen Tests auf industriell modifizierten Edge-Geräten repliziert werden, da industrielle Modifikationen von Raspberry Pi z. B. die Prozessorgeschwindigkeit dauerhaft verlangsamen (Beck 2019). Die verwendete Testumgebung ist insgesamt relativ spezifisch und limitiert die Übertragbarkeit der Ergebnisse zur Erfüllung diverser Anforderungen (z. B. hinsichtlich des industriellen Lastmanagements oder des Verhaltens während eines Angriffs). Andere Einsatzszenarien können ganz andere Spezifika mit sich bringen und die am Einsatzszenario gemessene Relevanz der identifizierten Anforderungen (z. B. durch die erforderliche direkte Anbindung der TSDB an eine IIoT-Plattform oder Berücksichtigung der Kosten für die Datenspeicherung in der Edge) verändern. Außerdem sind weitere Lasttests beim Einsatz TSDBs in skalierbaren Clustern erforderlich, da diese von den Autoren prototypisch nicht getestet wurden. Aufgrund der Größe der historischen Datensätze, welche in Data Lakes zusammengeführt und im Rahmen von Advanced Analytics abgefragt werden, können unvorhergesehene Performanzprobleme

auftreten. Zudem ist es möglich, dass sich die Performanz verschiedener TSDBs unterschiedlich verhält, in Abhängigkeit davon ob die TSDB in einer sog. „kalten" oder „heißen" Speicherebene eines Data Lake befindet und entsprechend die Häufigkeit der Lesezugriffe variiert (Firouzi et al. 2020). Durch die häufigen Updates von InfluxDB und der konkurrierenden Lösungen kann die vollständige Validität der Evaluationsergebnisse nur von kurzer Dauer sein. Für die sich im Alphastatus befindende Version 2.0 von InfluxDB[3] wurden bereits weitere Rollenkonzepte und die Zusammenführung der Module aus dem TICK-Stack zu einer ganzheitlichen Lösung vorangetrieben. Durch zukünftige Hardwareplattformen wie den Raspberry Pi 4 könnte die Performanz der TSDB dank einer leistungsfähigeren Hardwareausstattung noch besser ausfallen. So könnte das Edge-Gerät über die vorhandenen USB3-Schnittstellen mit einem zusätzlichen Datenträger ausgestattet werden und die vom Datenbankanbieter empfohlene Trennung der Zeitreihen in unterschiedliche Dateien auf getrennten Datenträgern ermöglichen. Diese für hohe Schreiblasten und hochfrequente Daten empfohlene Optimierungsmaßnahme wurde im Prototyp nicht berücksichtigt. Außerdem kann der prototypische Versuchsaufbau auch über eine längere Testdauer wiederholt werden, um die Ausfallsicherheit im Dauereinsatz (z. B. eine Woche oder ein Monat) zu testen.

1.7 Literatur

Ahmed E, Yaqoob I, Hashem IAT, Khan I, Ahmed AIAA, Imran M, Vasilakos AV (2017) The role of big data analytics in Internet of things. Comput Networks 129:459–471. https://doi.org/10.1016/j.comnet.2017.06.013

Arnst D, Plenk V, Wöltche A (2018) Comparative Evaluation of Database Performance in an Internet of Things Context. In: 13th International Conference on Systems and Networks Communications, S. 45-50

Bader A, Kopp O, Falkenthal M (2017) Survey and comparison of open source time series databases. In: Mitschang B, Nicklas D, Leymann F, Schöning H, Herschel M, Teubner J, Härder T, Kopp O, Wieland M (Hrsg) Datenbanksysteme für Business, Technologie und Web (BTW 2017) – Workshopband. Gesellschaft für Informatik e.V, Bonn, S 249–268

Beck A (2017) Raspberry Pi in der der Industrie: 10 Fallstricke beseitigt. In: https://www.industr.com/de/raspberry-pi-industrie-2297374. Zugegriffen: 22. Apr. 2019

Burdack M, Rössle M, Kübler R (2018) A Concept of an In-Memory Database for IoT Sensor Data. In: 14. Annual International Conference on Information Technology & Computer Science

Chen CH, Lin MY, Liu CC (2018) Edge computing gateway of the industrial internet of things using multiple collaborative microcontrollers. IEEE Network 32(1):24-32. https://doi.org/10.1109/MNET.2018.1700146

Chiang M (2016) Fog and IoT: An Overview of Research Opportunities. IEEE Internet of Things Journal 3(6):854-864. https://doi.org/10.1109/JIOT.2016.2584538

Corneo L, Gunningber P (2018) Scheduling at the edge of assisting cloud real-time systems. In: Proceedings of the 2018 workshop on theory and practice for integrated cloud, fog and edge computing paradigms, S 9–14

Crate (2020), CrateDB Edition, https://crate.io/products/cratedb-editions/. Zugegriffen am 20.05.2020

db-engines.com. Trends of the last 24 Months https://db-engines.com/en/ranking_categories. Zugegriffen: 14. Apr. 2019

DMG Mori (2016) Maschine 4.0, https://de.dmgmori.com/resource/blob/44728/3f3488f0b873549c9ce5fc74df181305/ps1de16-industrie4-0-pdf-data.pdf. Zugegriffen: 18. Apr. 2019

Firouzi F, Charkabarty K, Nassif S (2020) Intelligent Internet of Things. From Device to Fog and Cloud. Cham. https://doi.org/10.1007/978-3-030-30367-9

[3] Versionshinweise zu InfluxDB: https://www.influxdata.com/blog/category/tech/releases/

Gehring M, Charfuelan M, Markl V (2019) A comparison of distributed stream processing systems for time series analysis. In: Meyer H, Ritter N, Thor A, Nicklas D, Heuer A, Klettke M (Hrsg) BTW 2019 – Workshopband. Gesellschaft für Informatik, Bonn, S 205–214 https://doi.org/10.18420/btw2019- ws-21

Giebler C, Gröger C, Hoos E, Eichler R, Schwarz H, Mitschang B (2020) Data Lakes auf den Grund gegangen. Datenbank Spektrum 20:57-69. https://doi.org/10.1007/s13222-020-00332-0

Grafana Labs (2018) Using InfluxDB in Grafana http://docs.grafana.org/features/datasources/influxdb/. Zugegriffen: 10. Apr. 2019

Gröger C (2018) Building an Industry 4.0 Analytics Platform. Datenbank Spektrum 18:5-14. https://doi.org/10.1007/s13222-018-0273-1

Gröger C, Hoos E (2019) Ganzheitliches Metadatenmanagement im Data Lake: Anforderungen, IT-Werkzeuge und Herausforderungen in der Praxis. In: Grust, T., Naumann, F., Böhm, A., Lehner, W., Härder, T., Rahm, E., Heuer, A., Klettke, M. & Meyer, H. (Hrsg.), BTW 2019, S. 435-452. https://dx.doi.org/10.18420/btw2019-26

HarperDB (2019) The IoT Solution from Edge to Cloud, https://www.harperdb.io/. Zugegriffen: 25. Jun. 2019

InfluxData (2019) Open Source Time Series Platform https://www.influxdata.com/time-series-platform/. Zugegriffen am 02.05.2019

Jamesdixon (2010) Pentaho, Hadoop, and Data Lakes https://jamesdixon.wordpress.com/2010/10/14/pentaho-hadoop-and-data-lakes/. Zugegriffen: 04. Mai 2020

Jensen SK, Pedersen TB, Thomsen C (2017) Time Series Management Systems: A Survey. IEEE Transactions on Knowledge and Data Engineering 29(11):2581-2599. https://doi.org/10.1109/TKDE.2017.2740932

Lenz J, Westkaemper E (2017) Wear Prediction of Woodworking Cutting Tools based on History Data. Procedia CIRP 63:675-679. https://doi.org/10.1016/j.procir.2017.03.098

Martinviita M (2018) Time series database in Industrial IoT and is testing tool. University of Oulu, Degree Programme in Computer Science and Engineering

Martino DI S, Fiadone L, Person A, Vitale VN (2019) Industrial Internet of Things: Persistence for Time Series with NoSQL Databases. In: IEEE 28th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises, S. 340-345.

Mathis C (2017) Data Lakes. Datenbank Spektrum 17:289-293. https://doi.org/10.1007/s13222-017-0272-7

Metcalfe AV und Cowpertwait PSP (2009) Introductory Time Series with R, Springer, New York.

Miloslavskaya N, Tolstoy A (2016) Big Data, Fast Data and Data Lake Concepts. Procedia Computer Science 88:300-305. https://doi.org/10.1016/j.procs.2016.07.439

Mourtzis D, Vlachou E, Milas N (2016) Industrial Big Data as a result of IoT adoption in Manufacturing. Procedia CIRP 55;290-295. https://doi.org/10.1016/j.procir.2016.07.038

Mueller H, Gogouvitis SV, Seitz A, Bruegge B (2017) Seamless computing for industrial systems spanning cloud and edge. In: International conference on high performance computing & simulation (HPCS), S 209–216

Naqvi SNZ, Yfanzidou S, Zimanyi E (2017) Time Series Databases and InfluxDB. Studienarbeit, Université Libre de Bruxelles

Naumann F, Krestel R (2017) Das Fachgebiet „Informationssysteme" am Hasso-Plattner-Institut. Datenbank Spektrum 17:69-76.

Node-Red (2018a) A Node-RED node to interact with a Raspberry Pi Sense HAT, https://flows.nodered.org/node/node-red-node-pi-sense-hat. Zugegriffen: 17. Mai 2019

Outlyer (2016) Top 10 Time Series Databases https://outlyer.com/blog/top10-open-source-time-series-databases/. Zugegriffen: 10. Apr. 2019

Oyekanlu E (2017) Predictive Edge Computing for Time Series of Industrial IoT and Large Scale Critical Infrastructure based on Open-source Software Analytic of Big Data. In: Proceedings of the IEEE International Conference on Big Data, S. 1663-1669

Patel P, Ali MA, Sheth A (2017) On Using the Intelligent Edge for IoT Analytics. IEEE Intelligent Systems 32(5):64-69

Puliafito C, Mingozzi E, Longo F, Puliafito A, Rana O (2019) Fog computing for the internet of things: A Survey. ACM Transactions on Internet Technology 19(2):1-41. https://doi.org/10.1145/3301443

Schmalz (2019) Vakuumtechnik für die Automobilindustrie https://www.schmalz.com/de/anwendungen/branchen/vakuumtechnik-fuer-die-automobilindustrie/. Zugegriffen: 23. Sept. 2019

Väänänen O, Hämäläinen T (2018) Requirements for energy efficient edge computing: a survey. In: Galinina O, Andreev S, Balandin S, Koucheryavy Y (Hrsg) Internet of things, smart spaces, and next generation networks and systems 18th International Conference, NEW2AN 2018, and 11th Conference, ruSMART 2018, St. Petersburg, Russia, August 27–29, 2018 Springer, Cham, S 3–15. https://doi.org/10.1007/978-3-030-01168-0_1 (Lecture Notes in Computer Science, 11118)