

Inhalt

Abbildungsverzeichnis	3
Tabellenverzeichnis	5
1 Literatur	17

Abbildungsverzeichnis

Tabellenverzeichnis

- Netzwerkaufbau
 - Edge Computing (zentrale Verwaltung)
 - Fog Computing (lokale Fahrzeuge schließen sich zu Rechenknoten zusammen)
 - “a horizontal system-level architecture that distributes computing, storage, and networking functions closer to the user along a cloud-to-thing continuum”
OpenFog Consortium (Group, O.C.A.W., et al.: Openfog reference architecture for fog computing. OPFRA001 20817, 162 (2017))
 - Mist Computing
- Kommunikation
 - Zertifikate (Public/Private Key)
 - identitätsverschlüsselung
 - Belohnung für bereitgestellte Rechenleistung
- Ressourcenverteilung
 - Bestimmung der verfügbaren Rechenleistung
 - Optimierungsalgorithmen
 - Stackelberg Model
- Publish Subscribe
 -
 - Optimierungsalgorithmen
 - Stackelberg Model
- Softwarearchitektur in Fahrzeugen
 - RTOS
 - Middle Layer (ROS, keine automotive alternative stand 2019)
 - Cloud

- Stand der Technik

- Cloud Computing

- * IT resourcen werden flexibel nach Bedarf zur Verfügung bereitgestellt
 - * Realisiert durch Rechenzentren, Kunden können Ressourcen mieten anstatt eigene Server betreiben
 - * Ressourcen sind von verschiedenen Endgeräten erreichbar [SUN20]
 - * Cloud kann öffentlich oder privat sein, privat für sensible Daten [IBR21]

- Internet of Things (IoT)

- * Internet of Things (Internet der Dinge) bezeichnet eindeutig identifizierbare Objekte und deren virtuelle repräsentation in interent-ähnliche Strukturen[JIE13]
 - * IoT findet mittlerweile in fast allen Bereichen Anwendung
 - * Industrie: Produktionsanlagen, die durch vernetzte Systeme flexibel angepasst werden können (Beispiel: Produktvarianten)
 - * Smart Home: Sensoren und Aktoren im Haus vernetzen [WOR15]

- Edge Computing

- * Daten entstehen in Endgeräten, Applikationen, die die Daten verarbeiten sind zunehmend ebenfalls in Endgeräten
 - * Beispiel Flugzeuge oder autonome Fahrzeuge, generieren Daten in Größe von mehreren Gb pro Sekunde [LIU19a]
 - * Traditionell werden Daten in der Cloud ausgewertet und wieder an den Benutzer zur Verfügung gestellt, diese ist aber mit zusätzlichem Ressourcenaufwand verbunden wegen lange Übertragungsstrecken. [PÉR22]
 - * Übertragung und Verarbeitung in der Cloud langsam/unmöglich wegen Bandbreite und Latenz
 - * Edge Computing ist das Konzept, dass anstatt zentrale Cloud Server, Daten zunehmend auf Endgeräten verarbeitet werden [SHI16]
 - * Edge bezeichnet Geräte zwischen Datenquellen und cloud server

- * “a form of distributed computing in which processing and storage takes place on a set of networked machines which are near the edge, where the nearness is defined by the system’s requirements” (ISO/IEC: Tr 30164:2020 - internet of things (iot) -edge computing. Tech. rep., ISO/IEC (2020))
- * Motivation: Latenzreduzierung, unbenutzte Ressourcen verwenden
- * Anwendungsfälle:
 - Cloud Berechnungen auslagern
 - Smart Home, Daten lokal auswerten statt alles in die Cloud laden
 - Smart City
- Fog Computing
 - * Fog Computing: Eine Form von Edge Computing, zusätzliche Schicht zwischen Endgeräte und Cloud
 - * Edge Geräte kommunizieren mit Fog nodes, die wiederum für die Kommunikation zwischen Edge und Cloud zuständig sind
 - * Fog nodes übernehmen Datenverarbeitung lokal, für nur Lokal benötigte Daten, Leiten nur relevante Daten an Cloud weiter
 - * Fog nodes können PC-s, raspberry PI-s, nano-server, micro-datenzentren sein. [MAH20b]
 - * Beispiele:
 - Iot for All
 - FogFlow
- Mist Computing
 - * Datenverarbeitung direkt im Sensor.
 - * Erlaubt z.b. einfache Monitoringfunktionen direkt im Sensor
 - * Reduktion von benötigte Bandbreite und Rechenleistung in den übergeordneten Geräten
- Herausforderungen:
 - * allgemeine Rechenaufgaben auf spezialisierte Hardware

- * Erkennung von Edge nodes
- * Effiziente Identifikation bei der großen und sich dynamisch ändernden Anzahl an Geräten
- * Task Auslagerung und Verteilung
- * Keine Beeinträchtigung der Funktionalität des Edge Gerätes (z.B. Überlastung)
- * Sicherheit
- Anwendungsfälle für Fahrzeuge:
 - * Unbenutzte Rechenleistung von Hardware nutzen, autonome Fahrzeuge haben leistungsfähige Steuergeräte
 - * Auslagerung von Rechenaufgaben auf Fahrzeuge in der Umgebung mit freier Rechenkapazitäten
 - * Ausnutzung von Rechenleistung parkende Fahrzeuge
 - * Möglichkeit, die unbenutzte Rechenleistung zu Vermieten (wie cloud service)
- Publish/Subscribe Kommunikation:
 - * Modell für Nachrichtenbasierte Kommunikation [MAD18]
 - * Ereignisbasierte Kommunikation
 - * Teilnehmer kommunizieren indem sie Nachrichten mit bestimmten Themen veröffentlichen (Publisher), Empfänger können Themen abonnieren (Subscriber) und bekommen nur die abonnierten Nachrichten
 - * Kommunikation ist anonym
 - * Komponenten: Publisher, Broker, Subscriber
 - * Broker stellt Verbindungen zwischen Publisher und Subscriber her
 - * Broker speichert die Subscriptions (Abos)
 - * Weit verbreitete Implementierung: MQTT
 - * RTPS:
 - Real Time Publish Subscribe

- Kommunikation erfolgt ebenfalls über Themen
- Kein Broker, Teilnehmeridentifizierung erfolgt zur Laufzeit dezentralisiert
- Quality of Service parameter: Reliable writer speichert Sequenznummer der Nachrichten und kann erneut versenden bei Übertragungsfehler, Best effort Writer hat diese Funktionalität nicht
- Heartbeat message: writer gibt die verfügbaren nachrichten ID-s and
- AckNack message: kommunikation von empfangenen und nicht erhaltenen Nachrichten
- Konzept UNICARagil:
 - Modulare Systemarchitektur in allen Domänen
 - Fahrzeuge können flexibel auf verschiedene Anwendungen angepasst werden: Taxi, Privatfahrzeug, Shuttle, Cargo
 - gemeinsamer mechanischer Plattform
 - Sensoren und Aktoren mit definierte Schnittstellen
 - Alle Fahrzeuge vernetzt: untereinander, zur Cloud, Infobiene, Benutzer
- Motivation:
 - Zunehmende digitalisierung in Arbeits- und Privatumfeld
 - Cloud Dienste bereits sehr verbreitet
 - steigende Zahl an vernetzte Endgeräte
 - Daten entstehen an Endgeräten, verarbeitete Daten werden ebenfalls an Endgeräten benötigt
 - Cloud basierte ansätze erfordern immer höhere Bandbreite und zentralisierte Rechenleistung
 - Endgeräte haben immer höhere Rechenleistung, die oft unbenutzt bleibt
 - So auch in Fahrzeugen, die wegen autonome Fahrfunktionen deutlich mehr rechenleistung bekommen
 - Ungenutzte Rechenleistung kann für externe Anwendungen zur verfügung gestellt werden

- Dezentralisierte Rechenleistung verringert physikalische Distanz zwischen Entstehung und Verarbeitung der Daten
- Einsparung an Bandbreite und zusätzliche Server Rechenleistung
- Konzept: [MAH20b]
 - Applikation Architektur:
 - * monolithische Architektur
 - Applikation beinhaltet alle operationen
 - die gleiche Applikation kann mehrfach ausgeführt werden für parallele berechnungen
 - * Verteilt:
 - Modulbasiert:
 - Applikation auf Module aufgeteilt, abhängig voneinander
 - Bedienen Daten von einer Quelle
 - partitionierung von Applikationen [ASH22]
 - Micro-Services:
 - Applikation setzt sich aus unabhängigen Prozessen zusammen
 - Ein Microservice erfüllt nur eine Aufgabe
 - Applikation Platzierung: [MAH20b]
 - * Bestimmung verfügbare Rechenressourcen:
 - Profilierung: Applikation wird auf jedem Node ausgeführt, Rechenleistung wird aus Ausführungszeit bestimmt
 - Prädiktiv: Rechenleistung wird aus vergangenen Berechnungen ermittelt
 - Nach Bedarf: Je nach Erwartungen der Benutzer und Anforderungen an die Ausführungszeit wird nach Profil oder Prädiktiv verteilt
 - * Offloading Methoden:
 - Bottom-Up: Edge Geräte verlagern Applikationen in die Node Server

- Top-Down: Applikationen aus der Cloud oder Fog Node werden in Edge Geräte ausgelagert
- Hybrid: Top-Down, Bottom-Up, Edge geräte können Applikationen direkt untereinander austauschen
- * Ressourcenorientierung -Netzwerkstruktur
 - Hierarchie: Fog node server sind in Hierarchieebenen eingeteilt. Anzahl der Nodes auf einer Ebene nimmt nach unten hin zu.
 - Cluster: Fog nodes sind alle untereinander verbunden. Edge geräte werden in jeweilige Cluster gruppiert. Bessere horizontale Skalierung als bei Hierarchie.
 - Client-Server: Einige Fog Nodes funktionieren als Server, die anderen als client. Client nodes leiten ihre daten an den server weiter.
 - Master-Slave: Master node verteilt daten an slave nodes. Verwaltet die slave nodes. Master node kommuniziert die Ergebnisse.
- * Mapping:
 - Prioritätsbasiert: Priorisiert app platzierung auf bestimmte fog nodes, meistens heuristische methoden (best fit, first fit).
 - Optimierung: Kostenfunktion für die optimale Aufteilung erstellen und optimieren
 - multi-objective trade-off: nach mehreren Anforderungen gleichzeitig optimieren wie Energieverbrauch, verfügbarkeit, kosten
- * Platzierung
 - Statisch: Applikation wird einmal für jede Applikation platziert
 - dynamischen: Mehrere Instanzen einer Applikation können ausgeführt werden oder Applikation wird nur für eine Berechnung ausgeführt
 - Ereignisbasiert: Applikation wird zur Laufzeit umgezogen
- * Ausführungsumgebung:
 - Bare metal
 - Virtuelle machine

- container: [COS22]
- weniger overhead als virtuelle maschine
- verschiedene runtimes bereits verfügbar: Docker
- Orchestrierungsmöglichkeiten verfügbar: Kubernetes, Docker Swarm, Nomad, Marathon, Mesos
- Kontainerorchestrator ermöglicht Lebenszyklus, Skalierung, selbstheilung, migrierung,
- Keine Unterscheidung bei Gerätetypen
- Keine Unterscheidung bei Geolocation
- * Platzierungskriterien:
 - Quality of Service
 - Service level Agreement
 - Zeit und deadline
 - Profit
 - User-erfahrung
 - Kosten
 - externe Vorgaben
 - Energie
 - verfügbarkeit
 - Mobilität
- Applikationsverwaltung: [MAH20b]
 - * Fog Netzwerkverwaltung: [COS22]
 - zentralisiert: fog orchestrator an zentraler Stelle. Hat gesamtüberblick über das Netzwerk
 - dezentralisiert:
 - Verteilt:
 - * Safety:

- Datenintegrität
 - Verschlüsselung
 - Authentifizierung
 - * Leistungsüberwachung
 - Implikationbasiert:
 - Grenzwertbasiert:
 - * Finanzielle Unterstützung
 - Kompensation: Bei Serviceausfall werden benutzer kompensiert
 - Anreize: Verfügbare resourcen werden bei Bedarf zur verfügung gestellt, besitzer der Ressourcen werden vergütet
 - Reservierung: Verfügbare rechenleistung kann reserviert werden damit Verfügbarkeit garantiert wird
 - * resillienz:
 - Backup-Restore
 - Verfielfachung
 - Operator Migration
- Security: [MAH20b]
 - Container:
 - * Abstraktionsebene zwischen Prozesse und OS
 - * Packt die Prozesse und deren Abhängigkeiten zusammen so dass sie einfach auf andere Systeme portiert werden können
 - * definiert schnittstelle
 - * Läuft als Teil des Betriebssystems
 - * Software kann auch "bare metalöder in hypervisor virtuelle machine ausgeführt werden
 - Hypervisor:
 - * Virtuelle Machine manager

- * Jede Anwendung läuft im eigenen virtuellen OS (guest)
- * Anwendung hat ggf. keine Information darüber dass es in virtuellen Umgebung läuft
- * jede Anwendung läuft getrennt in eigenem Runtime Umgebung
- Anforderungen:
 - Verfügbare Rechenleistung für externe Kunden zugänglich machen
 - Kommunikationsmöglichkeit bieten, auch für parallele Berechnungen
 - Ressourcenmanagement
 - Netzwerküberwachung
 - Quality of Service sicherstellen
 - Lebenszyklus verwalten
 - Sicherheitsmaßnahmen vorsehen
 - Buchhaltung über Angefragte und gelieferte Rechenleistung
 - Verwaltungsdienst:
 - * Kommunikationsschnittstelle für den Kunden
 - * Erkennung von Teilnehmern (edge devices)
 - * Informationen über die Fähigkeiten der Teilnehmer sammeln (Architektur, Rechenleistung, Verfügbarkeit, Latenz)
 - * Ermittlung einer optimalen Verteilung von Kundenaufgaben an Teilnehmer
 - * Verteilung von Kundenapplikationen an geeignete Teilnehmer
 - * Kommunikationsschnittstelle zu den verteilten Kundenapps
 - * Kommunikationsschnittstelle für den Kunden
 - * Datenbank über Angeforderte und geleistete Berechnungen
 - * Vergütungsausschüttung nach Leistung
 - Loader Applikation:
 - * Meldet Verfügbarkeit an Verwaltungsdienst

- * Meldet Architektur, Rechenleistung, Verfügbarkeit an Verwaltungsdienst
- * Bietet Kommunikationsschnittstelle für die Übertragung von kompilierten Kundenapps
- * Bietet Kommunikationsschnittstelle für die Kommunikation mit Kundenapplikationen
- * Bietet Kommunikationsschnittstelle für die Verwaltung durch Verwaltungsdienst
- * Richtet Laufzeitumgebung für Kundenapplikation ein
- * Zugriffsrechte der Kundenapplikation werden eingeschränkt
- * Kann Kundenapplikation starten und beenden
- * Kommuniziert Applikationsstatus an Verwaltung
- Netzwerkstruktur:
 - Autotechagil Struktur gegeben: Cloud: Leitwarte, Edge device: Fahrzeuge, Infobiene
 - Zentraler Struktur bietet sich an: Leitwarte auch Fog Orchestrator
 -

1 Literatur

- [ASH22] ASHRAF , M., SHIRAZ , M., ABBASI , A., ALBAHLI , S.
„Distributed application execution in fog computing: A taxonomy, challenges and future directions“ en
In: *Journal of King Saud University - Computer and Information Sciences* 34.7 (Juli 2022), S. 3887–3909
ISSN: 1319-1578
DOI: 10.1016/j.jksuci.2022.05.002
URL: <https://www.sciencedirect.com/science/article/pii/S1319157822001513> (besucht am 11.01.2023)
- [CHE18] CHENG , B., SOLMAZ , G., CIRILLO , F., KOVACS , E., TERASAWA , K., KITAZAWA , A.
„FogFlow: Easy Programming of IoT Services Over Cloud and Edges for Smart Cities“
In: *IEEE Internet of Things Journal* 5.2 (Apr. 2018), S. 696–707
ISSN: 2327-4662
DOI: 10.1109/JIOT.2017.2747214
- [CHU16] CHUN , S., SHIN , S., SEO , S., EOM , S., JUNG , J., LEE , K.-H.
„A Pub/Sub-Based Fog Computing Architecture for Internet-of-Vehicles“
In: *2016 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*
ISSN: 2330-2186
Dez. 2016,
S. 90–93
DOI: 10.1109/CloudCom.2016.0029
- [COS22] COSTA , B., BACHIEGA , J., CARVALHO , L. R., ARAUJO , A. P. F.
„Orchestration in Fog Computing: A Comprehensive Survey“
In: *ACM Computing Surveys* 55.2 (Jan. 2022), 29:1–29:34
ISSN: 0360-0300
DOI: 10.1145/3486221
URL: <https://doi.org/10.1145/3486221> (besucht am 13.01.2023)
- [DOL04] DOLEJS , O., SMOLIK , P., HANZALEK , Z.
„On the Ethernet use for real-time publish-subscribe based applications“
In: *IEEE International Workshop on Factory Communication Systems, 2004. Proceedings.*
Sep. 2004,
S. 39–44

- DOI: 10.1109/WFCS.2004.1377674
- [EVE99] EVENSKY , D. A., GENTILE , A. C., WYCKOFF , P., ARMSTRONG , R. C.
 „The Lilith Framework for the Rapid Development of Secure Scalable Tools
 for Distributed Computing“ en
 In: *Distributed Applications and Interoperable Systems II*
 Hrsg. von L. Kutvonen, H. König und M. Tienari
 IFIP — The International Federation for Information Processing
 Springer US, Boston, MA, 1999,
 S. 163–168
 ISBN: 9780387355658
 DOI: 10.1007/978-0-387-35565-8_12
- [HOU16] HOU , X., LI , Y., CHEN , M., WU , D., JIN , D., CHEN , S.
 „Vehicular Fog Computing: A Viewpoint of Vehicles as the Infrastructures“
 In: *IEEE Transactions on Vehicular Technology* 65.6 (Juni 2016), S. 3860–
 3873
 ISSN: 1939-9359
 DOI: 10.1109/TVT.2016.2532863
- [HOU17] HOU , L., LEI , L., ZHENG , K.
 „Design on Publish/Subscribe Message Dissemination for Vehicular Net-
 works with Mobile Edge Computing“
 In: *2017 IEEE Globecom Workshops (GC Wkshps)*
 Dez. 2017,
 S. 1–6
 DOI: 10.1109/GLOCOMW.2017.8269200
- [HUA18] HUANG , X., YU , R., LIU , J., SHU , L.
 „Parked Vehicle Edge Computing: Exploiting Opportunistic Resources for
 Distributed Mobile Applications“
 In: *IEEE Access* 6 (2018), S. 66649–66663
 ISSN: 2169-3536
 DOI: 10.1109/ACCESS.2018.2879578
- [HUA20] HUANG , X., YE , D., YU , R., SHU , L.
 „Securing parked vehicle assisted fog computing with blockchain and optimal
 smart contract design“
 In: *IEEE/CAA Journal of Automatica Sinica* 7.2 (März 2020), S. 426–441
 ISSN: 2329-9274
 DOI: 10.1109/JAS.2020.1003039
- [IBR21] IBRAHIM , I. M., AL , E.

- „Task Scheduling Algorithms in Cloud Computing: A Review“ en
In: *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*
12.4 (Apr. 2021), S. 1041–1053
ISSN: 1309-4653
DOI: 10.17762/turcomat.v12i4.612
URL: <https://turcomat.org/index.php/turkbilmat/article/view/612> (besucht am 24. 10. 2022)
- [JIE13] JIE , Y., PEI , J. Y., JUN , L., YUN , G., WEI , X.
„Smart Home System Based on IOT Technologies“
In: *2013 International Conference on Computational and Information Sciences*
Juni 2013,
S. 1789–1791
DOI: 10.1109/ICCIS.2013.468
- [KAM19a] KAMPMANN , A., ALRIFAE , B., KOHOUT , M., WÜSTENBERG , A.,
WOOPEN , T., NOLTE , M., ECKSTEIN , L., KOWALEWSKI , S.
„A dynamic service-oriented software architecture for highly automated
vehicles“
In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*
IEEE, 2019,
S. 2101–2108
- [KAM19b] KAMPMANN , A., WÜSTENBERG , A., ALRIFAE , B., KOWALEWSKI , S.
„A Portable Implementation of the Real-Time Publish-Subscribe Protocol for
Microcontrollers in Distributed Robotic Applications“
In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*
Okt. 2019,
S. 443–448
DOI: 10.1109/ITSC.2019.8916835
- [KAM22] KAMPMANN , A., LÜER , M., KOWALEWSKI , S., ALRIFAE , B.
„Optimization-based Resource Allocation for an Automotive Service-oriented
Software Architecture“
In: *2022 IEEE Intelligent Vehicles Symposium (IV)*
IEEE, 2022,
S. 678–687
- [LAK22] LAKHAN , A., MEMON , M. S., MASTOI , Q.-A., ELHOSENY , M., MOHAM-
MED , M. A., QABULIO , M., ABDEL-BASSET , M.

„Cost-efficient mobility offloading and task scheduling for microservices IoT applications in container-based fog cloud network“ en

In: *Cluster Computing* 25.3 (Juni 2022), S. 2061–2083

ISSN: 1573-7543

DOI: 10.1007/s10586-021-03333-0

URL: <https://doi.org/10.1007/s10586-021-03333-0> (besucht am 10.01.2023)

- [LEW96] LEWIS , M., GRIMSHAW , A.
 „The core Legion object model“
 In: *Proceedings of 5th IEEE International Symposium on High Performance Distributed Computing*
 ISSN: 1082-8907
 Aug. 1996,
 S. 551–561
 DOI: 10.1109/HPDC.1996.546226
- [LIU19a] LIU , S., LIU , L., TANG , J., YU , B., WANG , Y., SHI , W.
 „Edge Computing for Autonomous Driving: Opportunities and Challenges“
 In: *Proceedings of the IEEE* 107.8 (Aug. 2019), S. 1697–1716
 ISSN: 1558-2256
 DOI: 10.1109/JPROC.2019.2915983
- [LIU19b] LIU , Y., YU , H., XIE , S., ZHANG , Y.
 „Deep Reinforcement Learning for Offloading and Resource Allocation in Vehicle Edge Computing and Networks“
 In: *IEEE Transactions on Vehicular Technology* 68.11 (Nov. 2019), S. 11158–11168
 ISSN: 1939-9359
 DOI: 10.1109/TVT.2019.2935450
- [LIU20] LIU , L., LU , S., ZHONG , R., WU , B., YAO , Y., ZHANG , Q., SHI , W.
Computing Systems for Autonomous Driving: State-of-the-Art and Challenges
 Techn. Ber.
 arXiv:2009.14349 [cs] type: article
 arXiv, Dez. 2020
 URL: <http://arxiv.org/abs/2009.14349> (besucht am 19.10.2022)
- [MAD18] MADE WIRAWAN , I., DWI WAHYONO , I., IDFI , G., RADITYO KUSUMO , G.
 „IoT Communication System Using Publish-Subscribe“

- In: *2018 International Seminar on Application for Technology of Information and Communication*
Sep. 2018,
S. 61–65
DOI: 10.1109/ISEMANTIC.2018.8549814
- [MAH20a] MAHARJAN , A. M. S., ELCHOUEMI , A.
„Smart Parking Utilizing IoT Embedding Fog Computing Based on Smart Parking Architecture“
In: *2020 5th International Conference on Innovative Technologies in Intelligent Systems and Industrial Applications (CITISIA)*
Nov. 2020,
S. 1–9
DOI: 10.1109/CITISIA50690.2020.9371848
- [MAH20b] MAHMUD , R., RAMAMOHANARAO , K., BUYYA , R.
„Application Management in Fog Computing Environments: A Taxonomy, Review and Future Directions“
In: *ACM Computing Surveys* 53.4 (Juli 2020), 88:1–88:43
ISSN: 0360-0300
DOI: 10.1145/3403955
URL: <https://doi.org/10.1145/3403955> (besucht am 15. 12. 2022)
- [MOK20] MOKHTARIAN , A., KAMPMANN , A., ALRIFAEI , B., KOWALEWSKI , S.
The Dynamic Service-oriented Software Architecture for the UNICARagil Project de
Techn. Ber.
Institute for Automotive Engineering, RWTH Aachen University ; Aachen :
Institute for Combustion Engines, RWTH Aachen University, 2020
DOI: 10.18154/RWTH-2020-11256
URL: <https://publications.rwth-aachen.de/record/807282> (besucht am 25. 10. 2022)
- [MOK21] MOKHTARIAN , A., KAMPMANN , A., ALRIFAEI , B., LÜER , M., KOWALEWSKI , S.
A Cloud Architecture for Networked and Autonomous Vehicles de
Techn. Ber. 2
Elsevier, 2021,
S. 233
DOI: 10.1016/j.ifacol.2021.06.028

URL: <https://publications.rwth-aachen.de/record/828696> (besucht am 02. 11. 2022)

[PÉR22] PÉREZ , J., DÍAZ , J., BERROCAL , J., LÓPEZ-VIANA , R., GONZÁLEZ-PRIETO , Á.

„Edge computing“ en

In: *Computing* (Juli 2022)

ISSN: 1436-5057

DOI: 10.1007/s00607-022-01104-2

URL: <https://doi.org/10.1007/s00607-022-01104-2> (besucht am 11. 10. 2022)

[QIA09] QIAN , L., LUO , Z., DU , Y., GUO , L.

„Cloud Computing: An Overview“ en

In: *Cloud Computing*

Hrsg. von M. G. Jaatun, G. Zhao und C. Rong

Lecture Notes in Computer Science

Springer, Berlin, Heidelberg, 2009,

S. 626–631

ISBN: 9783642106651

DOI: 10.1007/978-3-642-10665-1_63

[SAI21] SAITO , T., NAKAMURA , S., ENOKIDO , T., TAKIZAWA , M.

„A Topic-Based Publish/Subscribe System in a Fog Computing Model for the IoT“ en

In: *Complex, Intelligent and Software Intensive Systems*

Hrsg. von L. Barolli, A. Poniszewska-Maranda und T. Enokido

Advances in Intelligent Systems and Computing

Springer International Publishing, Cham, 2021,

S. 12–21

ISBN: 9783030504540

DOI: 10.1007/978-3-030-50454-0_2

[SHI16] SHI , W., CAO , J., ZHANG , Q., LI , Y., XU , L.

„Edge Computing: Vision and Challenges“

In: *IEEE Internet of Things Journal* 3.5 (Okt. 2016), S. 637–646

ISSN: 2327-4662

DOI: 10.1109/JIOT.2016.2579198

[SUN20] SUNYAEV , A.

„Cloud Computing“ en

In:

Hrsg. von A. Sunyaev

Springer International Publishing, Cham, 2020,

S. 195–236

ISBN: 9783030349578

DOI: 10.1007/978-3-030-34957-8_7

URL: https://doi.org/10.1007/978-3-030-34957-8_7 (besucht am 24.10.2022)

- [TUL19] TULI , S., MAHMUD , R., TULI , S., BUYYA , R.
„FogBus: A Blockchain-based Lightweight Framework for Edge and Fog Computing“ en
In: *Journal of Systems and Software* 154 (Aug. 2019), S. 22–36
ISSN: 0164-1212
DOI: 10.1016/j.jss.2019.04.050
URL: <https://www.sciencedirect.com/science/article/pii/S0164121219300822> (besucht am 06.01.2023)
- [VAR16] VARGHESE , B., WANG , N., BARBHUIYA , S., KILPATRICK , P., NIKOLOPOULOS , D. S.
„Challenges and Opportunities in Edge Computing“
In: *2016 IEEE International Conference on Smart Cloud (SmartCloud)*
Nov. 2016,
S. 20–26
DOI: 10.1109/SmartCloud.2016.18
- [WOR15] WORTMANN , F., FLÜCHTER , K.
„Internet of Things“ en
In: *Business & Information Systems Engineering* 57.3 (Juni 2015), S. 221–224
ISSN: 1867-0202
DOI: 10.1007/s12599-015-0383-3
URL: <https://doi.org/10.1007/s12599-015-0383-3> (besucht am 26.10.2022)
- [WU21] WU , Y., WU , J., CHEN , L., ZHOU , G., YAN , J.
„Fog Computing Model and Efficient Algorithms for Directional Vehicle Mobility in Vehicular Network“
In: *IEEE Transactions on Intelligent Transportation Systems* 22.5 (Mai 2021),
S. 2599–2614
ISSN: 1558-0016
DOI: 10.1109/TITS.2020.2971343

