

VDI

**REIHE 08**

MESS-,  
STEUERUNGS-  
UND REGELUNGS-  
TECHNIK



# Fortschritt- Berichte VDI

Mahyar Azarmipour, M. Sc.,  
Aachen

**NR. 1275**

## Virtualisierung prozess- naher Steuerungen in der Prozessautomatisierung

**BAND**

**1|1**

**VOLUME**

**1|1**



Lehrstuhl für  
Prozessleittechnik  
der RWTH Aachen

<https://doi.org/10.51202/9733186275080-1>

Generiert durch IP '18.218.160.239', am 03.05.2024, 12:54:59.

Das Erstellen und Weitergeben von Kopien dieses PDFs ist nicht zulässig.



# Virtualisierung prozessnaher Steuerungen in der Prozessautomatisierung

Von der Fakultät für Georessourcen und Materialtechnik der  
Rheinisch-Westfälischen Technischen Hochschule Aachen

zur Erlangung des akademischen Grades eines

**Doktors der Ingenieurwissenschaften**

genehmigte Dissertation

vorgelegt von

**Mahyar Azarmipour, M. Sc.**

Berichter: Univ.-Prof. Dr.-Ing. Ulrich Epple  
Univ.-Prof. Dr.-Ing. Stefan Kowalewski  
Univ.-Prof. Dr.-Ing. Tobias Kleinert

Tag der mündlichen Prüfung: 25.01.2022



VDI

**REIHE 08**  
MESS-,  
STEUERUNGS-  
UND REGELUNGS-  
TECHNIK



# Fortschritt- Berichte VDI

Mahyar Azarmipour, M. Sc.,  
Aachen

**NR. 1275**

## Virtualisierung prozess- naher Steuerungen in der Prozessautomatisierung

BAND  
1|1

VOLUME  
1|1

**ACPLT**  
**AACHENER**  
**PROZESSLEITTECHNIK**

<https://doi.org/10.51202/9783186275080-1>

Lehrstuhl für  
Prozessleittechnik  
der RWTH Aachen

Azarmipour, Mahyar

## **Virtualisierung prozessnaher Steuerungen in der Prozessautomatisierung**

Fortschritt-Berichte VDI, Reihe 08, Nr. 1275. Düsseldorf: VDI Verlag 2022.

126 Seiten, 62 Bilder, 4 Tabellen.

ISBN 978-3-18-527508-1, E-ISBN 978-3-18-627508-0, ISSN 0178-9546

48,00 EUR/VDI-Mitgliederpreis: 43,20 EUR

**Für die Dokumentation:** Virtualisierung – Speicherprogrammierbare Steuerung – Informationsdiode – Industrie 4.0 – Dynamisches Deployment – Automatisierungstechnik – IT/OT-Konvergenz

**Keywords:** Virtualization – Programmable logic controller – Information diode – Industry 4.0 – Dynamic deployment – Automation – IT/OT-Convergence

Die vorliegende Arbeit wendet sich an Ingenieur\*innen und Wissenschaftler\*innen aus der Prozessautomatisierung. Ziel dieser Arbeit ist ein Architektorentwurf für die Steuerungsgeräte der prozessnahen Komponenten, um diese mit einer höheren Vernetzung und Wandelbarkeit auszustatten. Die Architektur setzt Hypervisor-Virtualisierung ein, um eine Trennung der Anwendungen mit unterschiedlichen Anforderungen auf der gleichen Hardware zu ermöglichen. Die Anwendungen werden in vorkonfigurierten Partitionen gekapselt und betrieben. Um die Modularisierung der Anwendungen zu erhöhen, werden Container als zusätzliche Virtualisierungskomponenten eingesetzt. Für die Verwaltung der gesamten Komponentenhierarchie ist ein Verwaltungssystem vorgesehen, das die erforderlichen Dienste zur Komponentenverwaltung zur Verfügung stellt.

### **Bibliographische Information der Deutschen Bibliothek**

Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliographie; detaillierte bibliographische Daten sind im Internet unter [www.dnb.de](http://www.dnb.de) abrufbar.

### **Bibliographic information published by the Deutsche Bibliothek (German National Library)**

The Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliographie (German National Bibliography); detailed bibliographic data is available via Internet at [www.dnb.de](http://www.dnb.de).

D82 (Diss. RWTH Aachen University, 2022)

© VDI Verlag GmbH | Düsseldorf 2022

Alle Rechte, auch das des auszugsweisen Nachdruckes, der auszugsweisen oder vollständigen Wiedergabe (Fotokopie, Mikrokopie), der Speicherung in Datenverarbeitungsanlagen, im Internet und das der Übersetzung, vorbehalten. Als Manuskript gedruckt. Printed in Germany.

ISBN 978-3-18-527508-1, E-ISBN 978-3-18-627508-0, ISSN 0178-9546

<https://doi.org/10.51202/9783186275080-1>

Generiert durch IP '18.218.160.239', am 03.05.2024, 12:54:59.

Das Erstellen und Weitergeben von Kopien dieses PDFs ist nicht zulässig.

---

# Vorwort

Die vorliegende Arbeit ist während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Lehrstuhl für Prozessleittechnik der RWTH Aachen University entstanden. An dieser Stelle möchte ich mich herzlich für die Unterstützung und die ermöglichten Chancen bedanken.

An erster Stelle gilt mein großer Dank Herrn Prof. Dr.-Ing. Ulrich Epple für die Unterstützung meines Promotionsvorhabens. Die angenehme und konstruktive Arbeitsatmosphäre sowie der ausgezeichnete fachliche Austausch mit ihm haben zum erfolgreichen Abschluss meiner Arbeit beigetragen.

Bei Herrn Prof. Dr.-Ing. Stefan Kowalewski, Inhaber des Lehrstuhls Informatik 11 – Embedded Software an der RWTH Aachen University, möchte ich mich für die Übernahme der Rolle des Zweitgutachters bedanken.

Ich bedanke mich bei Herrn Prof. Dr.-Ing. Tobias Kleinert für seine Unterstützung und die Fachdiskussionen, welche ebenfalls zum Gelingen der Arbeit beigetragen haben.

Ich danke meinen Kollegen und den studentischen Hilfskräften für die gute Zusammenarbeit und die interessanten Gespräche. Besonders hervorheben möchte ich (in alphabetischer Reihenfolge) Haitham Elfaham, Julian Grothoff, Daniel Jakob, Lars Nothdurft und Christian von Trotha. Bei Frau Milescu bedanke ich mich für die gute Zusammenarbeit und organisatorische Hilfe.

Ein weiterer Dank gilt Afrooz Nazari für die Unterstützung und Motivation in den vergangenen Jahren.

Besonderer Dank gilt auch meinen Eltern und meiner Schwester für die Unterstützung während meiner gesamten Promotionszeit.

Warstein, im März 2022

Mahyar Azarmipour

Wissenschaft: Es ist nicht ihr Ziel, der unendlichen Weisheit eine Tür zu öffnen, sondern eine Grenze zu setzen dem unendlichen Irrtum.

*Bertolt Brecht*



---

# Inhaltsverzeichnis

|  |            |
|--|------------|
| <b>Vorwort</b>   | <b>III</b> |
| <b>List of Symbols</b>   | <b>VII</b> |
| <b>Kurzfassung</b>   | <b>IX</b>  |
| <b>Abstract</b>  | <b>XI</b>  |
| <b>1 Einleitung</b>  | <b>1</b>   |
| 1.1 Motivation   | 1          |
| 1.2 Zielsetzung  | 2          |
| 1.3 Struktur dieser Arbeit   | 3          |
| <b>2 Grundlage und Stand der Technik</b>   | <b>5</b>   |
| 2.1 Virtualisierung  | 5          |
| 2.1.1 Virtualisierungstypen  | 5          |
| 2.1.2 Virtualisierung mit Hypervisoren   | 6          |
| 2.1.3 Virtualisierung mit Mikrokernels   | 7          |
| 2.1.4 Hypervisor und Mikrokern-Technologien  | 8          |
| 2.2 Container-Technologien   | 11         |
| 2.2.1 Virtualisierungsanwendungen in anderen industriellen Domänen                             | 13         |
| 2.2.2 Virtualisierung in der Luftfahrt   | 15         |
| 2.2.3 Industrielle Automatisierung   | 15         |
| 2.3 NAMUR Open Architecture  | 16         |
| 2.4 Speicherprogrammierbare Steuerungen  | 17         |
| 2.4.1 Programmierung   | 17         |
| 2.4.2 Entwicklung der speicherprogrammierbaren Steuerungen                                     | 18         |
| 2.4.3 Neue Architekturen für speicherprogrammierbare Steuerungen                               | 19         |
| 2.5 Digitale Zwillinge und Verwaltungsschalen  | 21         |
| 2.5.1 Digitaler Zwilling als Validierungskomponente  | 22         |
| 2.5.2 Digitaler Zwilling für Beobachtung und Optimierung                                       | 23         |
| 2.6 Laufzeitumgebungen   | 23         |
| 2.6.1 Industrie-PCs und eingebettete Systeme   | 25         |
| 2.6.2 Betriebsmittel und Maßnahmenmodell   | 25         |
| <b>3 Anforderung an zukünftige Automatisierungssysteme</b>                                     | <b>27</b>  |
| 3.1 Anforderungen  | 27         |
| 3.2 Leistungsfähige Übertragung von Feld- und Automatisierungsdaten an überlagerte Anwendungen | 27         |
| 3.3 Prozessbegleitende Optimierung und Überwachung   | 28         |

|          |   |            |
|----------|---|------------|
| 3.4      | Effiziente interne Kommunikation . . . . .  | 29         |
| 3.5      | Lokale Komponentenverwaltung . . . . .  | 29         |
| 3.6      | Dynamisches Komponentenmanagement . . . . .                                       | 31         |
| 3.7      | Explizite Verwaltung und Sicherstellung von QoS-Eigenschaften . . . . .           | 31         |
| <b>4</b> | <b>Konzept</b>  | <b>33</b>  |
| 4.1      | Allgemeine Architektur . . . . .  | 33         |
| 4.2      | Komponentenhierarchie . . . . .   | 33         |
| 4.2.1    | Kommunikation zwischen den Partitionen . . . . .                                  | 34         |
| 4.3      | Systempartitionen . . . . .   | 36         |
| 4.3.1    | Verwaltungssystem . . . . .   | 37         |
| 4.3.2    | Interface . . . . .   | 38         |
| 4.4      | Verwaltungsdienste . . . . .  | 39         |
| 4.4.1    | Interne Kommunikationsdienste . . . . .   | 39         |
| 4.4.2    | Externe Kommunikationsdienste . . . . .   | 40         |
| 4.4.3    | Konfigurationsdienste . . . . .   | 41         |
| 4.4.4    | Ressourcenverwaltung . . . . .  | 42         |
| 4.4.5    | Komponentenverwaltungsdienste . . . . .   | 44         |
| 4.5      | Anwendungspartitionen . . . . .   | 46         |
| 4.6      | Evaluation anhand der Anforderungen an die Architektur . . . . .                  | 47         |
| <b>5</b> | <b>Anwendungsszenarien in der Automatisierungstechnik</b>                         | <b>50</b>  |
| 5.1      | Architektur der Automatisierungspyramide . . . . .                                | 50         |
| 5.2      | Beispielhafte Anwendungspartitionen . . . . .                                     | 50         |
| 5.2.1    | Control-Partition . . . . .   | 50         |
| 5.2.2    | O&M-Partition . . . . .   | 53         |
| <b>6</b> | <b>Implementierung für eine Kaltwalzanlage</b>                                    | <b>54</b>  |
| 6.1      | Logistik . . . . .  | 54         |
| 6.2      | SMS-Demonstrator . . . . .  | 55         |
| 6.3      | Aufbau . . . . .  | 57         |
| 6.4      | Verification of Request . . . . .   | 57         |
| 6.4.1    | Evaluation des VoR-Konzepts . . . . .   | 60         |
| <b>7</b> | <b>Validierung des Konzepts</b>   | <b>65</b>  |
| 7.1      | Eingesetzte Technologien . . . . .  | 65         |
| 7.1.1    | Portierung von ACPLT/RTE und PikeOS . . . . .                                     | 65         |
| 7.2      | Prozessführung . . . . .  | 66         |
| 7.2.1    | Laufzeitanalyse in virtualisierten und nicht virtualisierten Umgebungen . . . . . | 70         |
| 7.2.2    | Kommunikation . . . . .   | 70         |
| 7.2.3    | Verwaltungssystem . . . . .   | 73         |
| <b>8</b> | <b>Fazit</b>  | <b>77</b>  |
| <b>A</b> | <b>Anhang</b>   | <b>80</b>  |
|          | <b>Literatur</b>  | <b>103</b> |

---

# List of Symbols

- ASIL** Automotive Safety Integrity Level
- AUTOSAR** AUTomotive Open System ARchitecture
- CFC** Continuous Function Charts
- CPC** Core Process Control
- CPU** Central Processing Unit
- CPS** Cyber Physical Systems
- SFC** Sequential Function Chart
- DT** Digital Twin
- ESE** Einzelsteuereinheiten
- ECU** Electronic Control Units
- ERP** Enterprise Resource Planning
- EAL** Evaluation Assurance Levels
- FS** File System
- FB** Funktionsbaustein
- FIFO** First In First Out
- GSE** Gruppensteuereinheiten
- HMI** Human Machine Interface
- I40** Industrie 4.0
- IIC** Industrial Internet Consortium
- IoT** Internet of Things
- IP** Internet Protocol
- IT** Informationstechnik
- IACS** Industrial Automation and Control System
- IPC** Industrie-PC

|               |   |
|---------------|---|
| <b>KAS</b>    | komponentenbasierte Architektur für Automatisierungssysteme |
| <b>LWIP</b>   | Light Weight Internet Protocol                              |
| <b>M+O</b>    | Monitoring and Optimization                                 |
| <b>MMU</b>    | Memory Management Unit                                      |
| <b>MES</b>    | Manufacturing Execution System                              |
| <b>NOA</b>    | NAMUR Open Architecture                                     |
| <b>OT</b>     | Operational Technology                                      |
| <b>OPC UA</b> | Open Platform Communications Unified Architecture           |
| <b>OO</b>     | Object Oriented   |
| <b>OS</b>     | Betriebssystem  |
| <b>POSIX</b>  | Portable Operating System Interface                         |
| <b>PNK</b>    | Prozessnahe Komponente                                      |
| <b>QoS</b>    | Quality of Service  |
| <b>RTOS</b>   | Real Time Operating System                                  |
| <b>ROM</b>    | Read Only Memory  |
| <b>SSC</b>    | Sequential State Charts                                     |
| <b>SFC</b>    | Sequential Function Charts                                  |
| <b>SAP</b>    | Service Access Points                                       |
| <b>SIL</b>    | Safety Integrity Level                                      |
| <b>SPS</b>    | Speicherprogrammierbare Steuerung                           |
| <b>SOA</b>    | Service Oriented Architecture                               |
| <b>SCADA</b>  | Supervisory control and data acquisition                    |
| <b>TCP</b>    | Transmission Control Protocol                               |
| <b>UDP</b>    | User Datagram Protocol                                      |
| <b>VM</b>     | Virtuelle Maschine  |
| <b>VMM</b>    | Virtual Machine Monitor                                     |
| <b>VoR</b>    | Verification of Request                                     |

---

# Kurzfassung

Industrie 4.0 ist ein neues Paradigma, das eine zentrale Rolle in der Entwicklung der zukünftigen Automatisierungssysteme spielt. Die neue Generation der industriellen Automatisierungstechnik zielt auf die Erhöhung der Wandelbarkeit der Automatisierungssysteme ab. Dabei ist die Vernetzung und die Kooperation mit der IT-Welt eine wichtige Voraussetzung, um die angeforderte Wandelbarkeit zu erreichen. Daher müssen neue Architekturen und Lösungen eingesetzt werden, um eine Kooperation zwischen den Automatisierungssystemen und der IT zu realisieren. Ziel dieser Arbeit ist ein Architekturentwurf für die Steuerungsgeräte der prozessnahen Komponenten, um diese mit einer höheren Vernetzung und Wandelbarkeit auszustatten. Die Hauptanforderungen, welche von der Architektur erfüllt werden, sind:

- Der parallele Betrieb von Anwendungen unterschiedlicher Kritikalität
- Das dynamische Deployment von neuen Anwendungen zur Laufzeit
- Die Realisierung eines sicheren Gateways für die Kommunikation zwischen Systemen der Automatisierungsebene und übergeordneten IT-Systemen
- Die offene Kommunikation mit der IT-Welt
- Die Realisierung eines lokalen Software- und Zugriffsverwaltungssystems

Die vorgeschlagene Architektur besteht aus einem Mehrebenen-Komponentenmodell und wird als komponentenbasierte Architektur für Automatisierungssysteme (KAS) bezeichnet. Die unterste Ebene der KAS-Architektur ist die Ebene der Partitionen. Die KAS-Architektur setzt Hypervisor-Virtualisierung ein, um eine Trennung der Anwendungen mit unterschiedlichen Anforderungen auf der gleichen Hardware zu ermöglichen. Die Anwendungen werden in vorkonfigurierten Partitionen gekapselt und betrieben. Um die Modularisierung der Anwendungen zu erhöhen werden Container als zusätzliche Virtualisierungskomponenten eingesetzt. Containertechnologien ermöglichen die Kapselung und Verwaltung der Anwendungen in unterschiedlichen Containern innerhalb einer Partition. Dadurch können beispielsweise unterschiedliche Versionen der Anwendungen in einer Partition verwaltet werden. Die Container bilden die zweite Komponentenebene in der KAS-Architektur. Die letzte Komponentenebene stellt die Kapselung in die Funktionsbausteine dar. Für die Verwaltung der gesamten Komponentenhierarchie ist in der KAS-Architektur ein Verwaltungssystem vorgesehen, das die erforderlichen Dienste zur Komponentenverwaltung zur Verfügung stellt. Das Verwaltungssystem ist eine Systemfunktionalität der KAS-Architektur und in einer eigenen Partition gekapselt. Eine weitere Systemfunktion der KAS-Architektur ist das Interface. Dieses wird ebenfalls in einer eigenen Partition gekapselt. Diese Interface-Partition ist die einzige Partition, die mit externen Komponenten außerhalb der Kernautomatisierung kommunizieren darf. In der Arbeit werden für die Validierung der KAS-Architektur beispielhaft Anwendungspartitionen für die Prozessführung

und die Simulation entwickelt. Mit Hilfe dieser Anwendungen können realistische Szenarien der Automatisierungsebene prototypisch implementiert und getestet werden. Die Ergebnisse zeigen, dass die KAS-Architektur eine leistungsfähige und übersichtlich verwaltbare Systemumgebung darstellt, um für neue Anforderungen eine hohe Flexibilität zu bieten, sowie der durchgängigen Interoperabilität der Automatisierungsebene zu genügen, ohne die Integrität der Kernautomation zu gefährden.

---

# Abstract

Industry 4.0 is a new paradigm that plays a central role in the development of future automation systems. The new generation of industrial automation aims to increase the agility of the automation system. In this context, cooperation with the IT world is an important prerequisite to achieve the requested agility. Therefore, new architectures and solutions have to be developed to realize a cooperation between the automation systems and the IT. The goal of this work is an architecture design for the control devices in order to provide them with a higher level of connectivity and agility. The main features, which are fulfilled by the architecture, are:

- The parallel operation of applications of different criticality
- The dynamic deployment of new applications at runtime
- The realization of a secure gateway for communication between automation level systems and higher level IT systems
- The open communication with the IT world
- The realization of a local software and access management system

The proposed architecture consists of a multi-level component model and is referred to as a component-based architecture for automation systems (KAS). The lowest level of the KAS architecture is the partition level. The KAS architecture employs hypervisor virtualization to enable separation of applications with different requirements on the same hardware. Applications are encapsulated in preconfigured partitions. Containers are used as additional virtualization components to increase modularization of applications. Container technologies enable the encapsulation and management of applications in different containers within a partition. This means, for example, that different versions of the applications can be managed in one partition. The containers are the second component level in the KAS architecture. The last component level represents the encapsulation in the function blocks. For the management of the entire component hierarchy in the KAS architecture a management system is developed. The management system is a system functionality of the KAS architecture and is encapsulated in its own partition. Another system function of the KAS architecture is the interface. This is also encapsulated in its own partition. The interface partition is the only partition that is allowed to communicate with outside of the core automation domain. In this work, application partitions for process control and simulation are developed as examples for the validation of the KAS architecture. These applications can be used for a prototype implementation of automation level scenarios. The results show that the KAS architecture provides a powerful and clearly manageable system environment to meet the new requirements for agility as well as continuous interoperability of the automation level without compromising the integrity of the core automation.

