

ICS 35.210

Unified Application Management Interface for Cloud Application Platforms; only on CD-ROM

Vereinheitlichte Programmierschnittstelle (API) für die Verwaltung von
Anwendungen auf Cloud-Plattformen; nur auf CD-ROM

Interface de programmation unifiée pour les plateformes d'application en nuage; seulement
en CD-ROM

Zur Erstellung einer DIN SPEC können verschiedene Verfahrensweisen herangezogen werden:
Das vorliegende Dokument wurde nach den Verfahrensregeln einer PAS erstellt.

Gesamtumfang 22 Seiten

Dieses Dokument wurde durch die im Vorwort genannten Verfasser erarbeitet und verabschiedet.

Content

	Page
Foreword	3
Introduction.....	4
1 Scope	5
2 Normative references	5
3 Terms and definitions.....	5
4 Symbols and abbreviations.....	5
5 Basic PaaS Entities and specification of a Unified Application Management API	6
5.1 General	6
5.2 Specification of Basic Entities	7
5.3 Unified Cloud API specification	8
5.3.1 Application Management.....	10
5.3.2 Stack Management	12
5.3.3 Key Management.....	13
5.3.4 Service Management	14
5.3.5 ServiceBinding Management.....	15
5.3.6 Domain Management	16
5.3.7 Metric Management.....	17
5.3.8 Measurement Management	18
6 Requirements for Application Portability.....	18
6.1 General	18
6.2 Dynamic Binding of Services	18
6.3 Requirements for portability	20
Bibliography	22

Foreword

This DIN SPEC has been developed according to the PAS procedure. The development of a DIN SPEC according to the PAS procedure is carried out in workshops and does not require the participation of all stakeholders.

This document has been developed and adopted by the initiator and authors named below:

Will be fulfilled after final vote.

At present, there are no standards covering this topic.

DIN SPECs are not part of the body of German Standards.

A draft of this DIN SPEC has not been published.

Despite great efforts to ensure the accuracy, reliability and precision of technical and non-technical information, the Workshop cannot give any explicit or implicit assurance or warranty in respect of the accuracy of the document. Users of this document are hereby made aware that the Workshop cannot be held liable for any damage or loss. The application of this DIN SPEC does not release users from the responsibility for their own actions and is applied at their own risk.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. DIN [and/or DKE] shall not be held responsible for identifying any or all such patent rights.

Introduction

This DIN SPEC aims to create a global standard for managing and controlling the life cycle of an application that is executed in a Platform-as-a-Service (hereinafter PaaS) cloud provider. The term PaaS provider is in line with the definition that is provided by NIST¹⁾ according to which a PaaS provider is a cloud provider that offers the *“capability to the consumer to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly application hosting environment configurations”*.

The term “life cycle” denotes all steps and administration functions that are required in order to manage an application starting from the submission of an application to the PaaS provider to the de-provision of an application. Indicatively, these administrative functions include:

- preparing the proper execution environment for the application;
- managing services that are required for the execution of an application (e.g. databases);
- transferring of the application to the provider;
- managing the application’s execution state (i.e. starting, stopping, pausing);
- monitoring the application during execution;
- managing domains that are associated with an application;
- managing cryptographic keys that are associated with the PaaS provider customer.

The above steps represent core functions/services that have previously been offered to service users by platform-as-a-service providers as non-standardized, heterogeneous programming interfaces. The DIN SPEC is intended to provide a definition of these programming interfaces, thus maximizing the interoperability between platform-as-a-service solutions on the one hand and third-party add-on services on the other. Ultimately, the DIN SPEC will thus form the basis for an open, provider-independent platform-as-a-service ecosystem. Indicative examples of add-on services include cloud brokers, and automatic monitoring systems for contractually assured quality-of-service features.

The basis for the planned DIN SPEC is formed by the “Unified Cloud Platforms Interface Model and API” interface specification developed by a team of experts in the European PaaSPort research project (<http://paasport-project.eu>). The sole initiative with practical relevance in this field is “Cloud Application Management for Platforms” (CAMP)²⁾ by the OASIS standardization consortium, which at present, however, neither represents an officially adopted standard nor offers definition of all of the above-mentioned steps and administration functions.

1) <https://www.nist.gov/itl/cloud-computing>

2) https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=camp

1 Scope

This DIN SPEC will specify requirements for application programming interfaces (API) for applications on cloud platforms with the aim of optimizing the interoperability between platform-as-a-service solutions, tools and value-added services offered by third party providers.

2 Normative references

There are no normative references in this document.

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1

PaaS Provider

cloud provider that offers the capability to the consumer to deploy onto the offered infrastructure consumer-created or acquired applications created using programming languages and tools supported by the provider

[SOURCE: NIST]

3.2

application

any software that can be potentially hosted in a PaaS provider

3.3

stack

any software that is required by the application prior to its execution and is offered by the PaaS provider

3.4

service

any software that is required by the application upon its execution and is offered by the PaaS provider

3.5

bindedService

any service that is associated by a running application

4 Symbols and abbreviations

API	Application Programming Interface
CAMP	OASIS Cloud Application Management for Platforms
CRUD	Create, Retrieve, Update and Delete
DNS	Domain Name System
HTTP	Hypertext Transfer Protocol
IaaS	Infrastructure as a Service
JSON	JavaScript Object Notation

LDAP	Lightweight Directory Access Protocol
OASIS	Advancing Open Standards for the Information Society
OS	Operating System
PaaS	Platform as a Service
PDP	Platform Deployment Package
QoE	Quality of Experience
QoS	Quality of Service
REST	Representational State Transfer
SLA	Service-level agreement
SME	Small and medium-sized enterprises
SOA	Service-oriented architecture
SPI	Service Provider Interface
SSH	Secure Shell
TOSCA	OASIS Topology and Orchestration Specification for Cloud Applications
UML	Unified Modeling Language
URI	Uniform Resource Identifier
XML	Extensible Markup Language
XSD	XML Schema Definition

5 Basic PaaS Entities and specification of a Unified Application Management API

5.1 General

As already mentioned, the current specification provides requirements targeting the development of a universal application programming interface (API) that can be used to manage applications on PaaS environments. The ultimate goal is to achieve interoperability between PaaS providers and the portability of the applications per se. To do so, the specification will provide specification for the basic entities of a PaaS provider along with API signatures. Therefore, the term *shall* denote mandatory adoption while the term *can* optional adoption.

5.2 Specification of Basic Entities

Prior to the definition of the API the basic entities that *shall* be supported by any PaaS provider are listed on Table 1.

Table 1 — Basic Entities that *shall* be supported by any PaaS

Entity	Description
application	It describes the executable software that is hosted in the PaaS environment. Each application has a specific lifecycle and can be managed through the proposed API. Beyond lifecycle management, each application can expose several monitoring metrics.
stack	The stack refers to all underlying services that <i>shall</i> be available from the PaaS provider prior to its initiation. These services can be an application server, installed runtimes, or OS installations. Several PaaS providers refer to these runtime pre-requisites as stacks [2] yet several PaaS providers refer to stacks with different terms e.g. Buildpacks in case of Cloud Foundry [1], Cartridges in case of OpenShift [3] and stacks of Heroku.
key	It represents an RSA ³⁾ public key that can be associated with a specific user. The key <i>shall</i> be used during the authentication of the user.
service	Services are executable entities that are offered by PaaS providers and associate with an application after its initiation. Common services are relational databases, NoSQL databases, queuing and messaging services. The way the applications bind to the services.
ServiceBinding	A ServiceBinding is the materialization of an association between an application and a service.
domain	It refers to a DNS record that can be associated with a running application
metric	It refers to a variable that characterizes the QoS or QoE of an application and can be quantified.
measurement	The quantification of one metric of a running application on a specific timestamp is a measurement.

The relationship among the basic entities is depicted on Figure 1. As it is depicted, one application *can* require several execution stacks prior to its initialization. Moreover, an application *can* require several services during its execution. If this is the case these services have to be bound i.e. they have to be instantiated. An application *shall* be managed using cryptographic keys and *can* expose several metrics that are quantified in as measurements. Finally, an application *can* be associated with one or more domains.



Figure 1 — Relationship between basic entities

3) <http://theory.lcs.mit.edu/~rivest/Rsapaper.pdf>