PaaS: Definition

In this lesson, we'll look at a definition of PaaS.

WE'LL COVER THE FOLLOWING

In the cloud, there are three fundamentally different services offered.

IaaS

An **IaaS (Infrastructure as a Service)** offers virtual computers on which software has to be installed.

IaaS is a simple solution that corresponds to **classical virtualization**.

The decisive difference is the billing model, which for **IaaS bills only the actually used resource per hour or per minute**.

SaaS

SaaS (Software as a Service) denotes a cloud offer where **software can be rented** for word processing or financial accounting.

For software development, version controls or continuous integration servers

can be purchased as SaaS.

PaaS

PaaS stands for **Platform as a Service**. PaaS offers a platform on which custom software can be installed and run. The developer only provides the application to the PaaS, the PaaS makes the application executable.

Unlike Docker and Kubernetes (see chapter 13), **the operating system and the software installed on it is not under the developer's control**.

For the microservices examples, the `Dockerfile` specifies that an Alpine Linux distribution and a specific version of the Java Virtual Machine (JVM) should be used. This is no longer necessary with a PaaS. The JAR file contains the executable Java application and everything the PaaS needs.

To start the application in the runtime environment, the PaaS can create a Docker container, but the decision which JVM and Linux distribution to use is up to the PaaS.

The PaaS must be prepared to run different types of applications. .NET applications and Java applications require their own virtual machine, while Go applications do not.

The **appropriate environment must be created by the PaaS**. Nowadays PaaS are flexible enough to support different environments, even your own environment. Therefore, you can usually define which JDK should be used.

	Your own data center	IaaS	PaaS	SaaS
Data	✓	✓	✓	✓
Application	✓	✓	✓	
Databases	✓	✓		
Operating				

System	✓	✓		
Virtualization	✓			
Physical Servers	✓			
Network & Storage	✓			
Data Center	✓			

PaaS restricts flexibility and control

Developers have less control over the Docker images.

However, the question is whether a developer should spend time with the selection of the JVM and the Linux distribution in the first place. Often, these issues lie with operations anyway.

Some PaaS offer the possibility to configure the Linux distribution or JVM. Often, even complete runtime environments can be defined by the user. Nevertheless, **flexibility is limited**.

To run existing applications, the PaaS might not be flexible enough. For example, an application might need a specific JVM version that the PaaS does not support.

Microservices are usually newly developed, so **this isn't a big disadvantage**.

Routing and scaling

The PaaS must forward requests from the user to the application, so **routing is a feature of a PaaS**.

Similarly, PaaS can usually scale applications individually, ensuring scalability.

Additional services

Additional services

Many PaaS can provide the application with additional services such as databases.

Typical features for operation, such as support for analyzing log data or monitoring, are often part of a PaaS.

Public cloud

PaaS are offered in the public cloud where developers only have to deploy their application into the PaaS and then the application runs on the Internet.

This is a simple way to provide Internet applications. In the public cloud, there are further advantages; if the application is under high load, it can scale automatically.

Scalability is virtually unlimited as the application's public cloud environment can provide lots of resources.

PaaS in your own data center

The situation is somewhat different when the applications run in your own data center.

The use of PaaS is very easy for the developers, but the PaaS must first be installed. This can be a complicated process, which outweighs the advantages somewhat.

However, the PaaS **only needs to be installed once**. After that, developers can use the PaaS to install a variety of applications and bring them into production.

Operations only need to ensure that the PaaS functions reliably.

Particularly in the case of operations departments that still have manual processes and where the provision of resources takes a long time, **PaaS can considerably accelerate the rollout of applications** without the need for major changes to the organization or processes.

Macro architecture

As already shown in chapter 12, microservices platforms have an impact on the macro architecture.

While a system like Kubernetes (see chapter 13) can run any kind of Docker container, a PaaS works at the level of applications. A PaaS is, therefore, **more restrictive**. For example:

- All Java applications will be standardized to one or a few Java versions and Linux distributions.
- Programming languages that are not supported by the PaaS cannot be used to implement microservices.
- Monitoring and deployment are determined by the selection of the PaaS.

Therefore, a PaaS creates an **even higher standardization of the macro architecture** than is the case in a Kubernetes environment.

Modern **PaaS can be customized** and are quite flexible. In that case, the way the PaaS is customized, and which technology options are supported, can serve as a way to define the macro architecture.

Then the macro architecture is not defined by the PaaS supplier but by whoever customizes the PaaS.

QUIZ

1 Suppose your company is looking for an IaaS. Which profile best matches that of an IaaS?

In the next lesson, we'll study the PaaS technology, Cloud Foundry.