Container Technology applied in Industry 4.0

Overview

Industrial environments become more and more a target of cybercriminals. IT/Cybersecurity issues can be mitigated by improving their digital communication infrastructure. This document gives a basic understanding of container technology, and shows its power applied on industrial secure gateways, in the context of Industry 4.0.

Running Heterogeneous Software Applications in a Secure Environment

Many industries face the challenge of becoming “Industry 4.0” compliant, but reaching this goal can be hard and often needs a huge effort in terms of expenses and working hours. Secure digital communication is one of the most important challenges a modern industry has to consider. It is the base for access policy management, remote machine monitoring, and many other features. Every industry has its own story and needs, standard software applications often poorly meet them, and deploying a custom extended solution could be so expensive to be impracticable. Container technology gives easy affordable help, and this is what we are going to write about.

Background

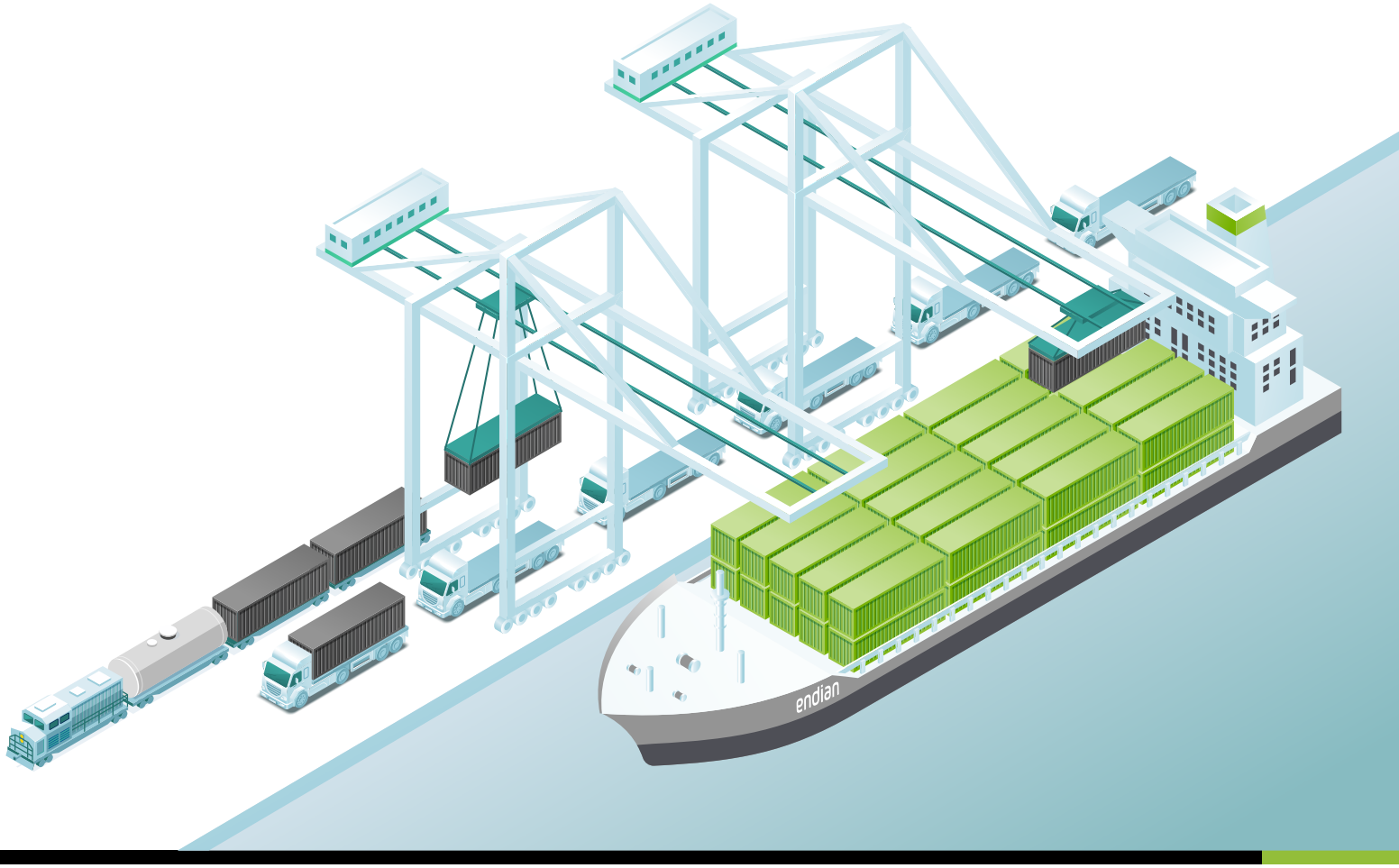
In this section we want to give the reader the basic knowledge in order to understand the topic.

Container Technology

Let us start with a real-world example about the shipping industry. Many companies have the same problem; how can they ship their own goods around the world keeping the costs low. The idea to solve this problem is to use shipping containers, but what is a shipping container? It is a metal box of fixed standard size, which can be filled with any kind of goods. The concept behind is that a company pays for the box as a unit a fixed fee regardless of content; a shipping container filled with gold could have the same fee as another one half filled with rubbish. There are obviously some restrictions to respect, but the important point we want to show is that the shipping container is content-independent, and considered as a unit.

Shipping industry decided on containers because they meet standard track trailers and special cargo ships designed to accommodate those sizes. Containers can be easily moved from a track trailer to a cargo hold and vice versa as units by cranes on the dock. This technique reduces costs and makes the shipping of goods inside containers more efficient.

Now that we have an understanding of shipping containers let us apply this concept to computer science.



Software Containers

Let us define a software application as composed by the application's core, its dependencies, settings and configuration. A software container, which we shortly call "container" from now on, is a unit comprising not only the application itself, but also its libraries, and everything it needs to work properly. We will discuss the single parts below, but the point we want to underline is that the same way a shipping container can be picked up by a crane and deployed somewhere else as a unit, so a container can be moved or copied from a machine to another as a unit.

Let us consider this example; we have to install the same application with the same settings on a row of machines. The traditional way tells us we need to install, and configure the software on each single machine. This is a very time consuming repetitive job, which could cause malfunctions due to the high risk of making mistakes. We want to do this job in a smarter way using containers; we install and configure the application on one single container, and then we simply deploy it to every machine without additional effort. This procedure saves time, mitigates the risk of making mistakes, and not to be underestimated, it can be fully automated. In this case, thanks to parallelization, it doesn't really matter if we deploy to five machines or to five thousand.

Before we enter into more technical aspects, let us use the following example as a metaphor to explain some concepts. Let us consider a library, in which we define some layers:

- **Customer (Top)** has the need to improve its knowledge about wild animals, and is looking for a book about that topic.
- **Librarian (Middle)** knows the library process very well. It performs following steps:
  - Takes charge of the customer's request.
  - Forwards the request to the library infrastructure.
  - Gets the right book from the library infrastructure.
  - Hands the book out to the right customer.

  Let us focus on this point; the librarian is the connection between customer and library infrastructure. The customer is not allowed to access the library infrastructure, and take a book on its own.
- **Library Infrastructure (Base)** is the engine, storage and management of the library.

Now similarly consider a software divided in following layers:

- **Application (Top)** has to satisfy its own needs to do the job properly; it needs to access resources provided by the Operating System (OS) in the same way a library's customer needs to access a book. An example of application could be the preferred Internet browser.
- **Libraries (Middle)** is a hidden layer of software in the middle. It is comparable to the librarian in the library. It performs following steps:
  - Takes charge of the application's request, and forwards it to the OS.
  - Gets back the requested resource access authorization.
  - Hands the authorization out to the requesting application.

  As the customer in the library does not have direct access to the library infrastructure, so the application does not have direct access to the OS.
- **OS (Base)** is the engine and resources manager of the software system. Some examples of OSs are Windows, Mac OS, Linux, Android, iOS.

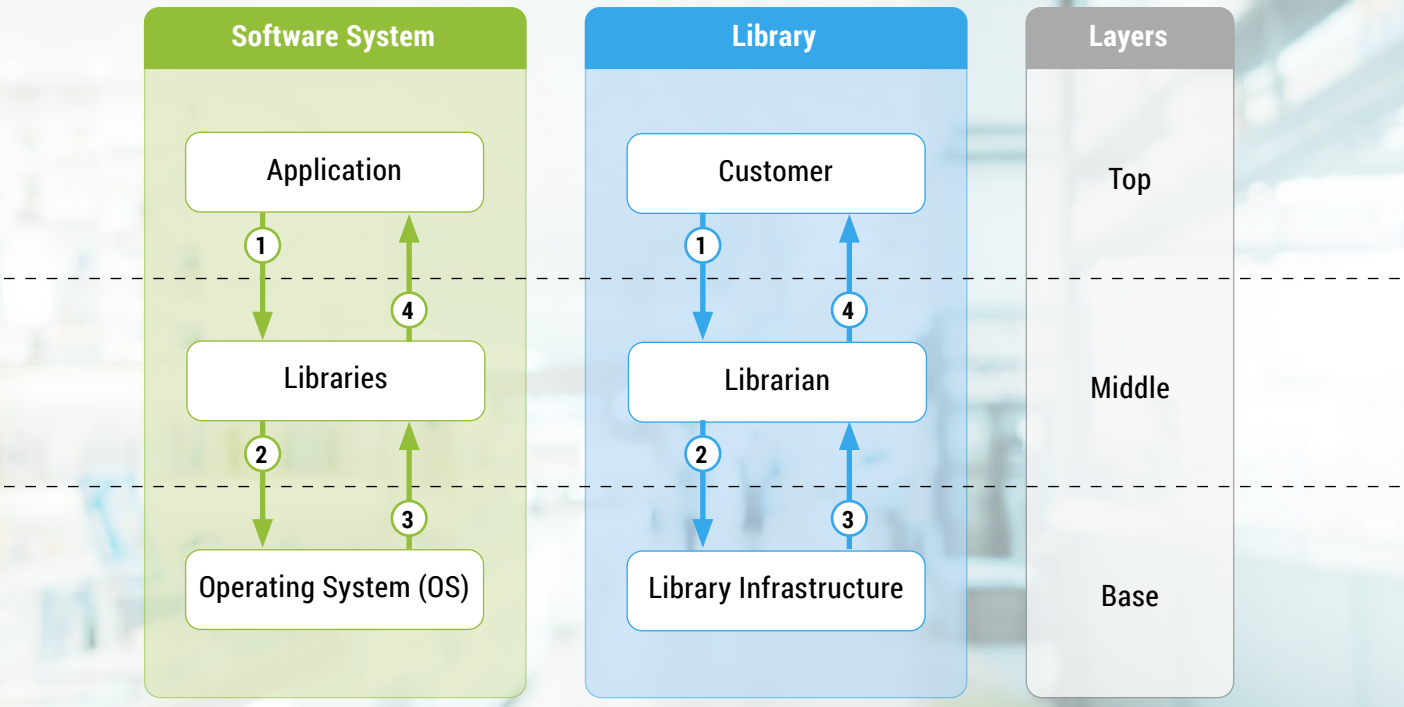


Figure 02 comparing Library with Container Technology

Figure 02 gives a visual understanding of what we already described above. The software layers show in a very simplified way how traditionally an application runs on a standard computer such as a PC.

The step further is to put aside the OS layer for the moment, and pack the application and libraries layers together in one single unit; this is our container.

Let us highlight this important point; a container is composed of an application, and libraries. Since we have removed the OS layer, the container is therefore OS-independent.

The Container Management Engine

Now that we know how a container is composed, let us insert a new layer between the OS and the containers as shown in Figure 03. This is the Container Management Engine (CME) represented by Docker, the most popular CME at the time of writing. This is the most important element of container technology. It is available for almost every OS and is responsible for providing the right needed resources at the right time for each container it has in charge. We can imagine the CME for containers similarly to a ship or a trailer for shipping containers. Both take care about containers they have in charge at that moment.

Writing about CMEs forces us to introduce another concept; the difference between container and instance. We can consider a container as an offline unit; to be clear, it is ready but in stand-by. An instance of a container instead, is a live running container.

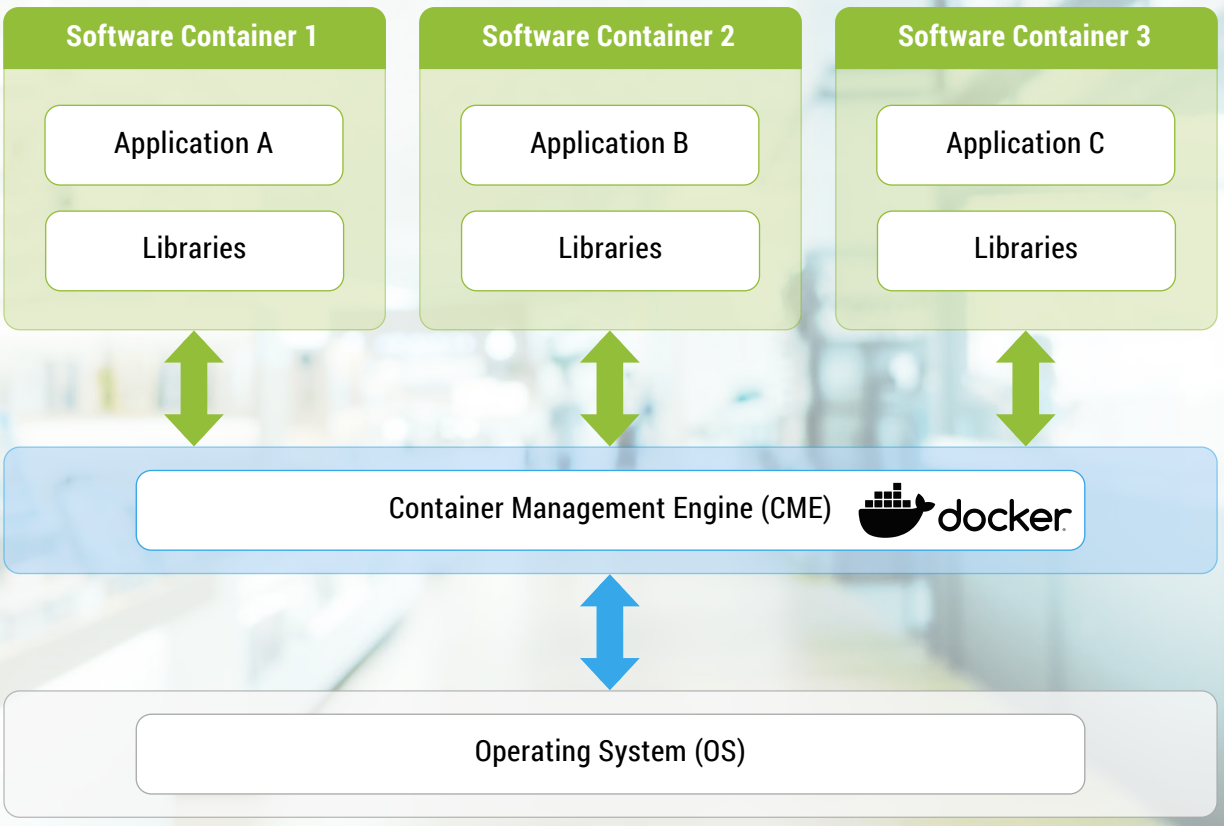


Figure 03 Container Technology with Docker Engine

The power of Docker is that it shares the resources provided by the underlying OS to all its containers. It does not care which machine it is running on, it only needs to know that it has some limited resources at its disposal. Docker engine man- ages those resources to best satisfy the needs of each single container in its charge.

Applications start very fast because they do not waste time in starting the OS before. The CME, provides for everything they need.

One thing we should not underestimate is Docker's power of starting multiple instances of the same container on a single machine. Let us consider the following example; imagine a web portal of an Internet shop. When there is normal activity, we imagine that one instance of the shop application's container would be enough to satisfy customers requests. One day we advertise a huge discount on the shop, we would expect a flood of requests that the single application's instance may not be able to face. The CME can monitor the instance's load and start one or more new instances of the same container to face the huge increase of requests. The shop can therefore resist the flood of requests satisfying every single happy customer.

Generally speaking, an application could stop working properly due to wrong settings or any other reason. Imagine a user working with an application, setting something wrong, and at the end the application does not start anymore. This is the kind of situation that no user would want to face. In a traditional environment this issue could be severe. Since we are smart using containers, we have a simple way to solve this kind of issue; we simply stop the application's instance, and start it again with clean settings.

Let us emphasize this point; we start the application's instance again without stopping the CME and anything else.

Mastering Container Technology in Industrial Environments

Endian's Secure Digital Platform comes into play not only as an engine for digital transformation, but as a means of adding business value. It integrates container technology on its "4i Edge X" intelligent secure gateways, and therefore supports container-based custom software applications.

As the gateway's name "Edge" suggests, these kinds of devices fully meet the concept of "Edge Computing", which brings services close to users thanks to container technology while greatly reducing the network data communications overhead.

Industrial companies can use Endian's powerful ecosystem to install their own software on their products. A machine builder, for instance, has the option of equipping its products with process data acquisition software, which can be used, among others, to perform predictive maintenance. And not only that; at the same time, the machine builder can offer its customers the option of installing their individual software. This option adds value to the machine.

The customers, in turn, can install their individual software on the machines. This could be, for example, software for data preprocessing or machine status monitoring.

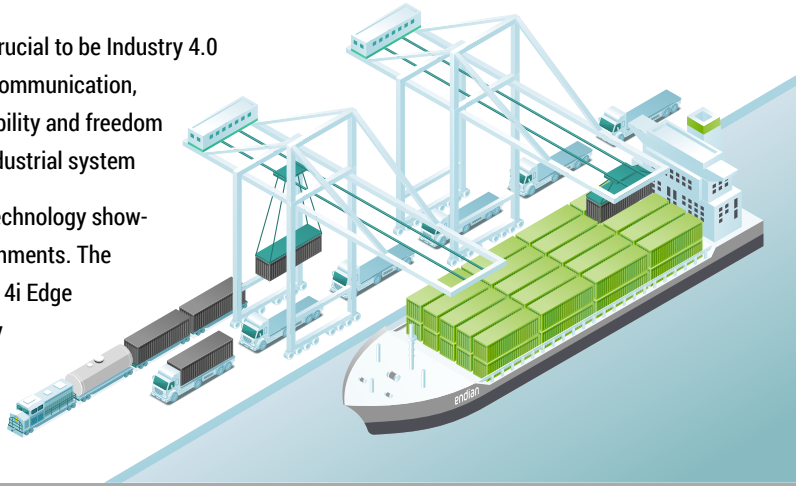
Neither the machine manufacturers nor the customers need to worry about data security, as this is guaranteed by Endian's underlying Secure Digital Platform.

Generally speaking, industry can benefit from Endian' Ecosystem power by fully supporting its custom applications, and considering it as its own system.

Conclusion

Secure digital communication in industrial environments is crucial to be Industry 4.0 compliant. Endian's Secure Digital Platform enables secure communication, and the integration with a CME gives the users so much flexibility and freedom that they can consider it as an integral part of the internal industrial system

In this document we explained in easy words the container technology showing advantages and possible applications in industrial environments. The power of container management engines applied on Endian's 4i Edge X secure gateways, that can be managed in a centralized way over the Switchboard, is in our opinion very promising and open to many improvements.



References

- **Endian "4i Edge X":** https://www.endian.com/products/secure-digital-platform/endian-4i-edge-x/
- **Endian "Switchboard"**: https://www.endian.com/products/secure-digital-platform/switchboard/
- **Docker**: https://docker.com
- **Gartner Research - Edge Computing**: https://www.gartner.com/en/doc/3889058-the-edge-completes-the-cloud-a-gartner-trend-insight-report