Rocket

LEGACY POWERS LEGENDARY

Rocket® Uniface

Microservices: Everything you need to know

There's no better way of getting a fresh start with the year than with a focus on interesting trends and technologies. We start by having a better look at microservices.

In what ways can microservices help your customers, and how can you benefit from microservices during application development?

It's time for a chat with **Michael Taylor.**

What do microservices mean to you?

"Microservices offer an effective way to break up your applications in order to make them independent and easier to scale. This way, your applications can meet the big-scale needs of modern businesses these days.

Microservices enable you to deliver applications that are not only scalable but are also stable. They offer the infrastructure in which you can deliver applications that last.

Microservices really are popular with containerization because you don't need big virtual machines or bare metal to deploy your applications. Now you can put them into orchestrated environments and get started right away."

About Michael Taylor

Michael passionately believes in the Rocket® Uniface product and what it enables customers to achieve. During his long tenure at Rocket Uniface, Michael has gained experience in support, pre-sales, professional services and engineering, before starting his current role of product manager.

Before we dive deeper into the architecture, in what ways can microservices help the end customer?

"If we look back in the history of application development, we are used to working on monolithic applications—applications that consisted of one block of code sitting on a platform. In these cases, it's possible to scale by adding other platforms or by using a load balancer, but microservices offer a more sustainable way of working. With microservices you are breaking up your application in common parts. Developers can identify the different areas of concern and make sure these are developed, tested, and deployed independently. In a sense, microservices give you more control.

You can have a part of an application that has a high workload and that has to be scaled accordingly, while having another component that is used infrequently and that as a result has different scalability requirements. With microservices, it is possible to break up the application into several areas of concern, put clearly defined APIs around the individual components, and scale them to meet the business needs. When you look at technologies in the cloud and the orchestration, such as Kubernetes (an open-source system for automating deployment, scaling, and management of containerized applications), you really can benefit from the flexibility microservices offer."

> **"Microservices really are popular with containerization because you don't need big virtual machines or bare metal to deploy your applications."**

Why do microservices fit well within the Rocket® Uniface paradigm?

"With microservices, you end up with loosely coupled areas within your application, connected by a lightweight protocol in between them for communication between the individual services. Within the Rocket® Uniface development framework, we already work with loosely coupled components. This means that you can define your services via configuration. The backbone-the communication path between the different services—is already there and is built into the genetics of the Rocket® Uniface applications. As a result, microservices really sit well within this environment."

In what ways does Rocket® Uniface support the use of microservices?

"First, carefully consider your application. You want to create areas of your application where one service is responsible for maintenance of that information. All of this is possible within the Rocket® Uniface environment. At the same time, you can share components between subsystems. With Rocket® Uniface applications, developers are able to define the architecture at deployment rather than in development time. It optimizes your ability to change and makes it easier to keep up with your business needs."

As a Rocket® Uniface developer, what skills and concepts can I use to fully benefit from microservices?

"You need to be able to model your information and to maintain your applications. Next, you can break your application database up into separate databases so they get along with the microservices, etc. All these steps are basically the standard Rocket® Uniface development paradigm. It's just a matter of breaking your application up into several components.

We have long promoted a service-oriented architecture in which you move your business logic out of the end component into a back-end service, and the same way of thinking applies to microservices. You know how to create a subsystem in Rocket® Uniface, you are able to create a clear API on it, and you can then have that API talk into a specific database. Also, you know how to move it and to deploy it across multiple platforms. Rocket® Uniface has always offered the Rocket® Uniface Request Broker (URB) in the background, which provides communication between the different services. So in that sense, the way of thinking is there already. Furthermore, containerization platforms such as Docker mean that you can encapsulate the application subsystem in a lightweight fashion. Before you might have had it on multiple virtual machines, but they were much heavier."

To what extent can I reuse my existing code in order to use microservices?

"You probably don't need to redevelop your whole application. The logic already is available; the same goes for the business knowledge. You can remove that and locate it in common areas rather than rewrite it. Of course, there may be some re-architecting and refactoring necessary. But the logic is still there and is still accessible."

// With microservices, you end up with loosely coupled areas within your application, connected by a lightweight protocol in between them for communication between the individual services. Within the Rocket® Uniface development framework, we already work with loosely coupled components. //

What do I need in order to get started?

“If you already are a Rocket® Uniface developer, you have everything in place to get started. You need a development license, which you can get through your normal development license or via the Community Edition, and you can start creating your Rocket® Uniface applications. Think about how you want to structure your application. When starting with microservices, you want to have shared services. Example: You don’t want to have every single service do its own authentication. Instead, build a service that handles all authentication and use something like JSON web tokens to pass between the different services to show that you have been authenticated. Another aspect you might want to consider is logging. If you have a request coming in, it might be directed across multiple web services. You want to collect that into a single environment in which you can see how that behaves across all the different services, rather than digging into each one.

Your application might need to be augmented with some extras in order to be able to understand what’s going on and to get the full benefit out of it. Another great benefit of breaking it up is that you end up with domain experts in particular areas of your application. So you can progress one subsystem and the others can use it when ready, rather than having to do everything at the same time. This way, you are dealing in a much smaller area and you are developing in that small area. This means that the changes are probably smaller; it’s more testable; and you end up with an agile way of working.”

Now it’s your turn to get started

Rocket® Uniface has a proven track record and leads the way for model-driven, low-code application development and deployment software for enterprise businesses, software integrators, and ISVs.

Accelerate innovation with the Rocket® Uniface low-code platform and find out what’s new in Rocket® Uniface 10 Community Edition.

To get a head start, visit the free eLearning academy and have a look at the repositories on GitHub and library code in the Rocket® Uniface community.

ASK AN EXPERT

Rocket_Uniface_Microservices_v4