Microservices Architecture for Retail and E-commerce

The Backbone for Achieving Speed and Scale

Suman Mukhopadhyay
AVP - Senior Director and Head - Technology Management, EdgeVerve Systems Ltd (An Infosys Company)

Summary

Retail therapy is the therapy Gen X, Y, and Z swear by, and with the rise in e-commerce, this therapy is now available to everyone in their palm. However, for the whole experience to be truly worthwhile, retail and e-commerce brands now must shoulder the responsibility of providing variety as quickly as possible. And that means faster scalable applications. This article takes you through the various pitfalls of a traditional architecture used by retail and e-commerce enterprises and the way forward.

As I am writing this article, it's the festive season in India, which means every e-commerce website has a sale going on. I was buying a gift for someone and as I reached the payment page, the app got stuck. I waited for a few seconds and finally decided not to buy the product at all.

Even in the 5G era, with the internet running at GBPS and not MBPS, websites and apps getting stuck are common occurrences. That almost always results in the customer opting out of buying the product and sometimes never visiting the app/ site at all. While sometimes, the blame lies on internet speed, most of the time, it is the traditional monolithic architecture of the application.

End of the Monolithic Era

In the traditional monolithic architecture, the entire application is a single, self-contained, tightly coupled entity. It is responsible for a particular task and can finish it end-to-end. The components of monolithic applications are built on top of one another and all code exists in a single codebase. It was great for ease of development, but the compromise was speed, scale, and flexibility.

A monolithic application, built on the three-tier architecture, is slow. For every request, the entire application – the presentation layer (UI layer), Business Layer (Logic code layer), and data layer (Layer which connects to database - is deployed at the same time. If one component falls short or fails, everything comes crashing down. Also, each copy of an application instance gets access to all the data. This makes caching less effective, along with large memory consumption. Added to this is the sheer amount of input-output traffic.
A monolithic architecture is also difficult to update and scale as additional nodes are needed for new functionality. If you duct tape new functionality to keep up with your business needs, monolithic architecture can quickly grow into a "big ball of mud"

5 reasons why Monolithic Architecture Cannot Support New Business Needs

- **Architecture gets more complex, and the quality of code declines over time, making it difficult to understand**
- **Technology updates or moves to a new language or framework require redeploying the entire application**
- **Every change opens up the application to more risk**
- **Overall application performance is affected by an error in a functionality**
- **Does not support modularity and scaling will increase the code base affecting performance**

where no one knows the interdependencies anymore. Also, any time you make a change to a monolithic application, you have limited options AND you open yourself up to more risk – even a small bug can bring down the entire system!

The slow, sluggish, and change resistant nature of monolithic applications makes them unsuitable for digital business needs – especially when the ask is complex feature sets across a variety of ever-changing devices.

Microservices: Modularity that Supports Agile and Resilient Businesses

Microservices architecture brings in modularity in application development with each feature developed separately and independently. The application is loosely coupled and structured as a collection of small services which are autonomous. These are modeled around a business domain. While the development complexity is higher, applications built on the microservices architecture are high performing, scalable, reliable, and more secure.

Doing it Right

Designed for rapid deployment, Microservices are a great fit for companies working in Cloud environments or with a DevOps approach. In fact, companies like Google, Amazon, and Netflix have been active proponents of this architecture for its ability to provide a consistent user experience across devices and platforms. For companies in Retail, Supply-Chain, WEB/ E-Commerce, Order Fulfillment systems, and FMCG space, moving to a microservice architecture is the need of the hour.

Companies across the board are waking up to the potential of microservices. Recently we helped one of our clients, a global leader in sports retail, deploy a solution based on the microservices architecture to improve logistics between their distributors / warehouses across countries. The new architecture enabled the functionality and scale that was essential for their business needs and could not have been supported by a monolithic application.

Here's why a microservice architecture is a mandate in these times of rapid technology disruption:

- **Faster Time to Market**

Microservices make it easier to add or modify features faster. Being loosely coupled, makes it possible to change only the specific service and test and deploy it independently. This cuts down time to the market significantly as compared to the monolithic approach.

- **Increased system resiliency**

With microservices there is minimal impact of modifications to one component on the others, reducing the risk of system failure. Even if several of the services are brought down for maintenance, there will be no noticeable changes in the overall services to your clients.

- **Development efficiency**

Small teams working on different components in a continuous delivery model work more efficiently. Microservices help leverage reusable code, reduce deployment time, cut down infrastructure costs, while minimizing downtime. In addition, different components do not share code, so they can be written in different languages making the most of new technology skills.

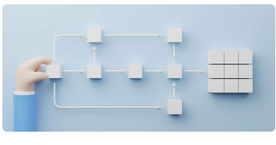
- **Scalability**

Unlike the monolithic architecture the microservice-based application is well suited for scalability. Instead of the whole application, only the service needing to be scaled up can be updated. In addition, business-critical services can be made available on multiple servers. This improves availability and performance without impacting other areas of the app.

- **Performance**

In a microservices architecture, components communicate through APIs instead of sharing a database. Each combination of request and response is an independent transaction creating multiple instances of communication and improving the response time. For example, we observed that while a monolithic application on legacy infrastructure could perform X number of Credit Card checkouts per minute, a microservices-based architecture on a Cloud platform could do a similar exercise at 100X per second. That's a massive difference in response time! Imagine the difference it could make to customer experience during a holiday shopping rush.

Recognizing the importance and relevance of this architecture in supporting the products and services of today and tomorrow, we at EdgeVerve have started to rapidly adopt microservices for all our offerings.



Microservices in Action

Think of any e-commerce application. Such an application would have a couple of services. For example, these services could be 'Customer Service', 'Product Service', and 'Cart Service'. With monolithic architecture, all of these various services are deployed to the customer as a single service. Their codes are the same, they function all together and if anything goes wrong with one unit, all hell breaks loose.

But, not quite so with microservice architecture. Here, each service functions as a separate component that functions on its own. To the customer, the services are deployed in the same place. But, behind the scenes is a whole other story. Since their codes and data are not the same, they perform services without disturbing the other components. If a program developer were to modify a single service, say, the 'Product Service,' she would be able to do so without touching the other components.
In addition, there may be microservices for channel management that enable various channels (FTP, email, etc.) for file exchange, a distributed cache management service for lookups and transformations, and microservices for row level data processing to perform high-level validation and preparatory tasks.

Important Aspects of Building a Microservices Architecture

The core idea of a microservice architecture is to keep things as straightforward as possible to avoid tight coupling of the components.

The first and foremost common mistake developers make when it comes to microservices is the database. It is quite common to have a single database for all the different capabilities in a monolithic architecture. When a user accesses its order, you'll look directly in the user table to display the customer information, and the same table might be used to populate the invoice managed by the billing system. This seems logical and simple. However, with microservices, you will want the services to be decoupled—so that invoices can still be accessed even if the ordering system is down—and because it allows you to optimize or evolve the invoice table independent of others. This means that each service might end up having its own datastore to persist the data that it needs.

Another thing to remember while splitting the monolithic architecture to build a microservices architecture is to keep the communication between services simple with a RESTful API. The communication protocol between services should be as simple as possible.

Let the Future Come Calling

In the coming years, businesses will have to ramp up on innovation and deploy products and services rapidly to meet consumer needs – both within and outside the enterprise. Technology will be the backbone of these goals, and ensure your business is operating from a future-ready technology architecture should be on every business leader's agenda today. Are you ready to welcome the future?