

# MONOLITH MIDDLEWARE MODERNIZATION

From Monolith Middleware to Microservices – A guide.

Oracle SOA • Oracle BPEL • IBM BPM • Oracle ADF • JBOSS Implementation



—ONATE

# ARE YOU EXPERIENCING THESE TOP SIGNALS OF UNCERTAIN FUTURE?

Too much spend on  
Middleware  
licenses and upgrades

Time-consuming  
Middleware  
deployments

Struggle to build new features quickly  
as per changing market demands

Trouble finding  
specialized and skilled  
Middleware consultants



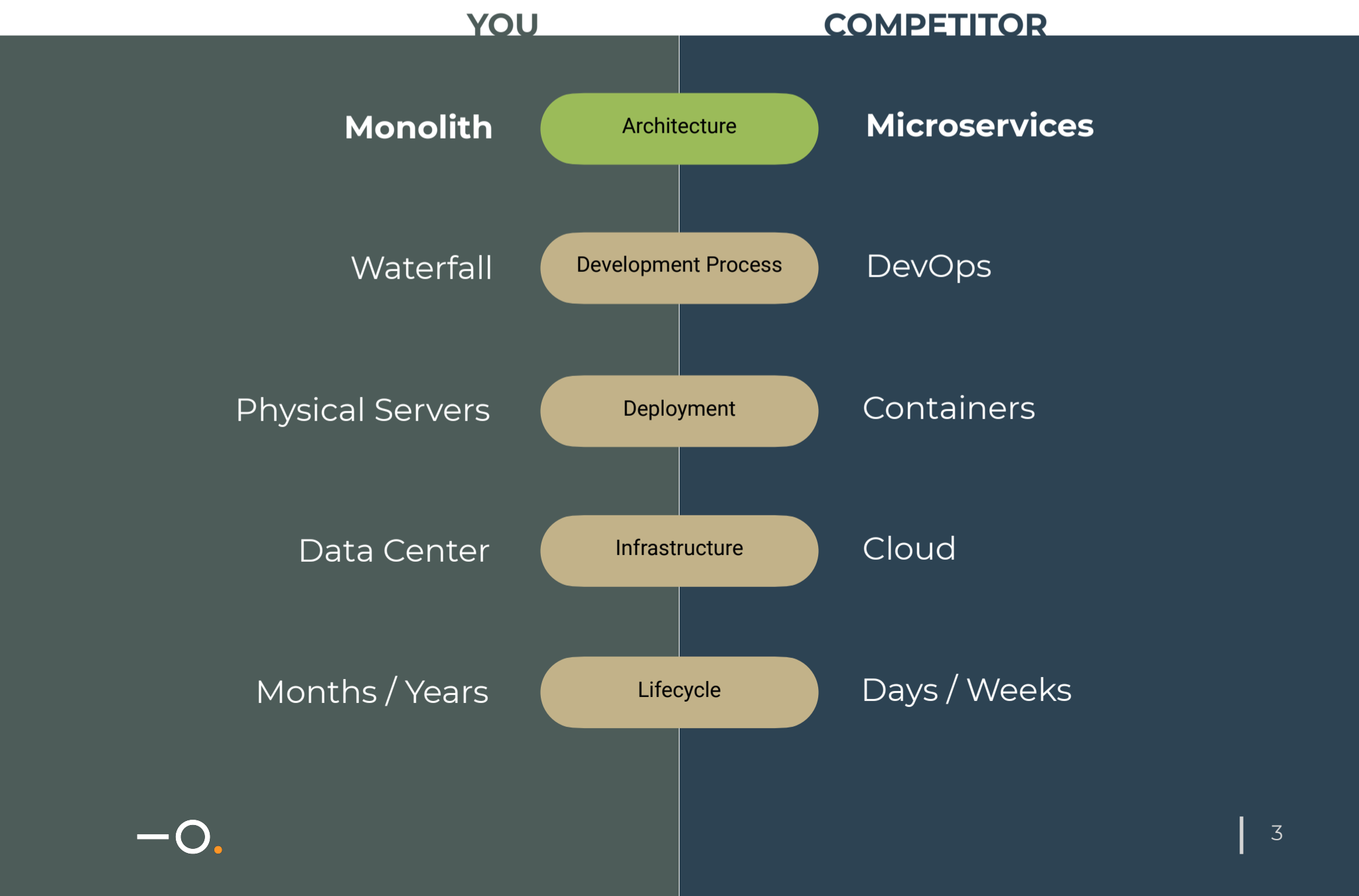
# THE PROBLEM

*You identify a customer pain point. You validate and plan to fix that by adding a new feature to your application. It takes 9 months to develop, build, test, and release the feature. By then the customer has moved on — to your competitor. The competitor did not even discover the pain point until a few months after you did. But they won by adding the new feature in 1 Week!*

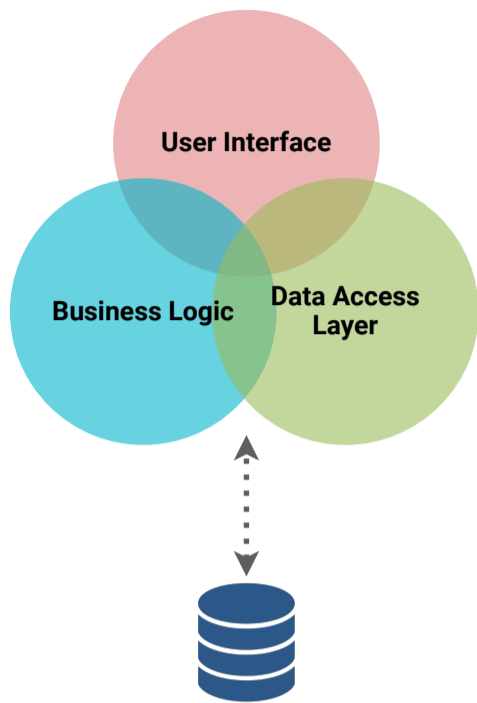
Let us get down to the basics and understand the key elements in the above example. The speed, agility, and responsiveness to the ever-changing market demands will determine how businesses will survive, and maintain, gain, or lose market share.

Traditionally developed software (monoliths) are unable to keep up with today's fast-paced competitive business landscape. Modern microservices on the other hand — promises easily customizable software and faster release cycles — apart from other benefits like scalability and resilience.

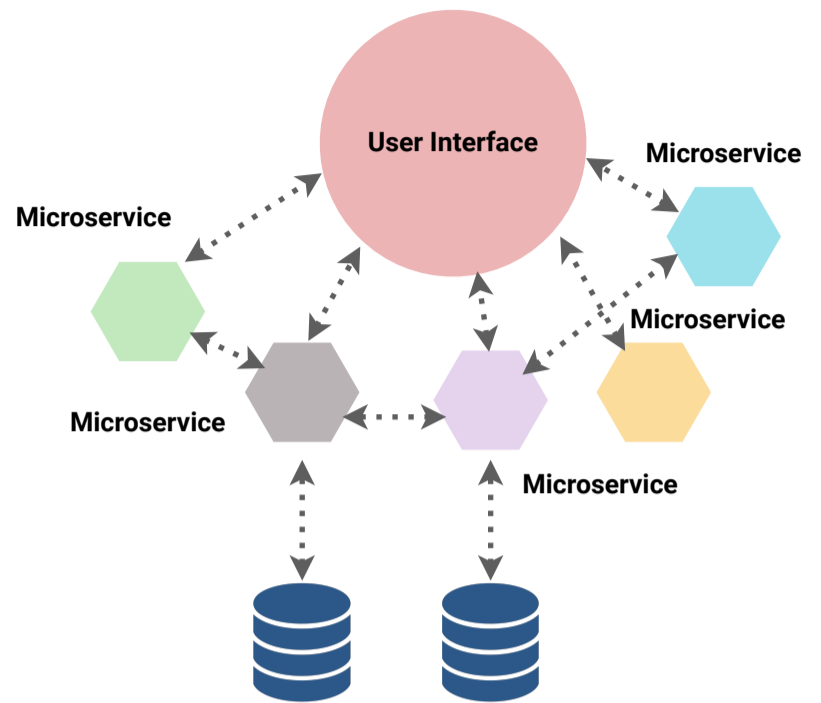
## Bird's Eye — Monolith vs Microservices



## Monolith Architecture





## Microservices Architecture



## THE ANALYSIS

Going back to the example in the problem statement, let us analyze how architecture plays a role in delivering the desired customer feature, on time, at scale.

|                                  | <br><b>Monolith</b>   | <br><b>Microservices</b>  |
|----------------------------------|--|--|
| <b>Development &amp; Testing</b> | <ul style="list-style-type: none"> <li>Can't work with new tech and tools</li> <li>Need specialized skills for old tech</li> <li>Any change requires testing of entire app</li> <li>Low productivity due to interdependencies</li> </ul> | <ul style="list-style-type: none"> <li>Free to use different tech per service</li> <li>Easier to find skilled professionals</li> <li>Any change can be tested within a service</li> <li>High productivity due to modular distribution</li> </ul> |
| <b>Deployment</b>                | <ul style="list-style-type: none"> <li>Must redeploy entire app on each update</li> <li>Any change can take days or weeks to deploy</li> </ul>   | <ul style="list-style-type: none"> <li>Only deploy the updated service</li> <li>Rapid and continuous deployments</li> </ul>  |
| <b>Scaling</b>                   | <ul style="list-style-type: none"> <li>Scaling part of the system is impossible</li> <li>Difficult to scale when app becomes large</li> </ul>  | <ul style="list-style-type: none"> <li>Each service can be scaled independently</li> <li>Can be scaled horizontally and vertically</li> </ul>  |
| <b>Maintenance</b>               | <ul style="list-style-type: none"> <li>Hard to debug and troubleshoot issues</li> <li>Any leak / bug can bring down the entire application</li> <li>Interdependencies make it hard to reuse parts of the application</li> </ul>          | <ul style="list-style-type: none"> <li>Easy to debug due to better fault isolation</li> <li>Reduced downtime due to resilient services</li> <li>Services can be reused in other projects, reducing cost</li> </ul>                               |

# THE STRATEGY

Your reasons to move to microservices could be one or more: reduced costs, operational efficiencies, improved customer experience, continuous innovation, and so on. Whatever be the reasons, the right strategy is to pick an approach that solves the challenges posed by monolith and is long-term.

Let's explore some popular strategies used by the industry.

**Software doesn't age gracefully.**

## Lift and shift

Drop your application into the cloud as-is and hope that it works!

**Core legacy problems remain.**

## Refactor

Restructure and optimize existing code to adapt to cloud environment.

**Skills and time are a challenge.**

## Replatform

Modify your application to take advantage of cloud infrastructure.

**Extremely limited without automation.**

## Rearchitect

Materially alter the code to shift to a new application architecture.

**Time and parity are a challenge.**

## Rebuild

Rewrite the application from scratch, preserving scope and specifications.

**Time is a huge challenge.**

## Replace

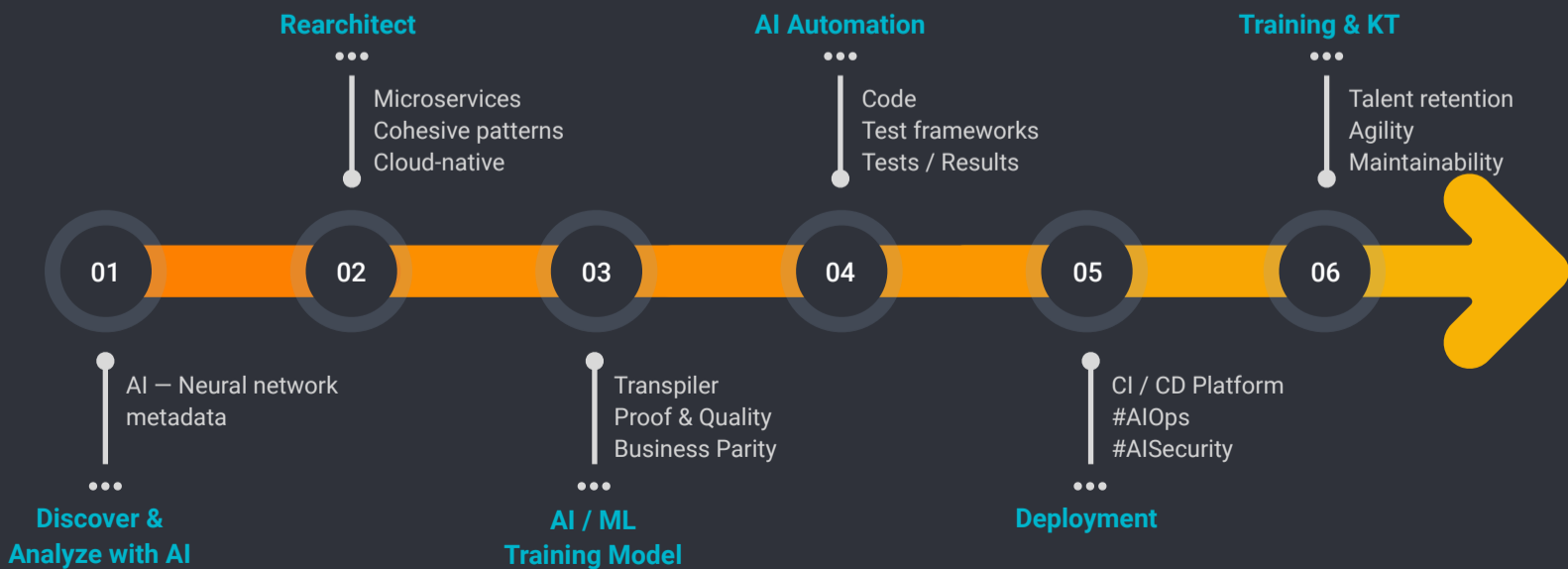
Retire your application and replace it with a new cloud-native application.

**Time and business risk.**

# OUR APPROACH

We've carefully analyzed the long-term pitfalls of some strategies – lift and shift, refactor, and replatform. And we are cognizant of the time and parity challenges of the strategies – rearchitect, rebuild, and replace. As a result, we built a "living" platform that employs a combination of the best long-term strategies, while overcoming the challenges of time and business parity, using AI / ML.

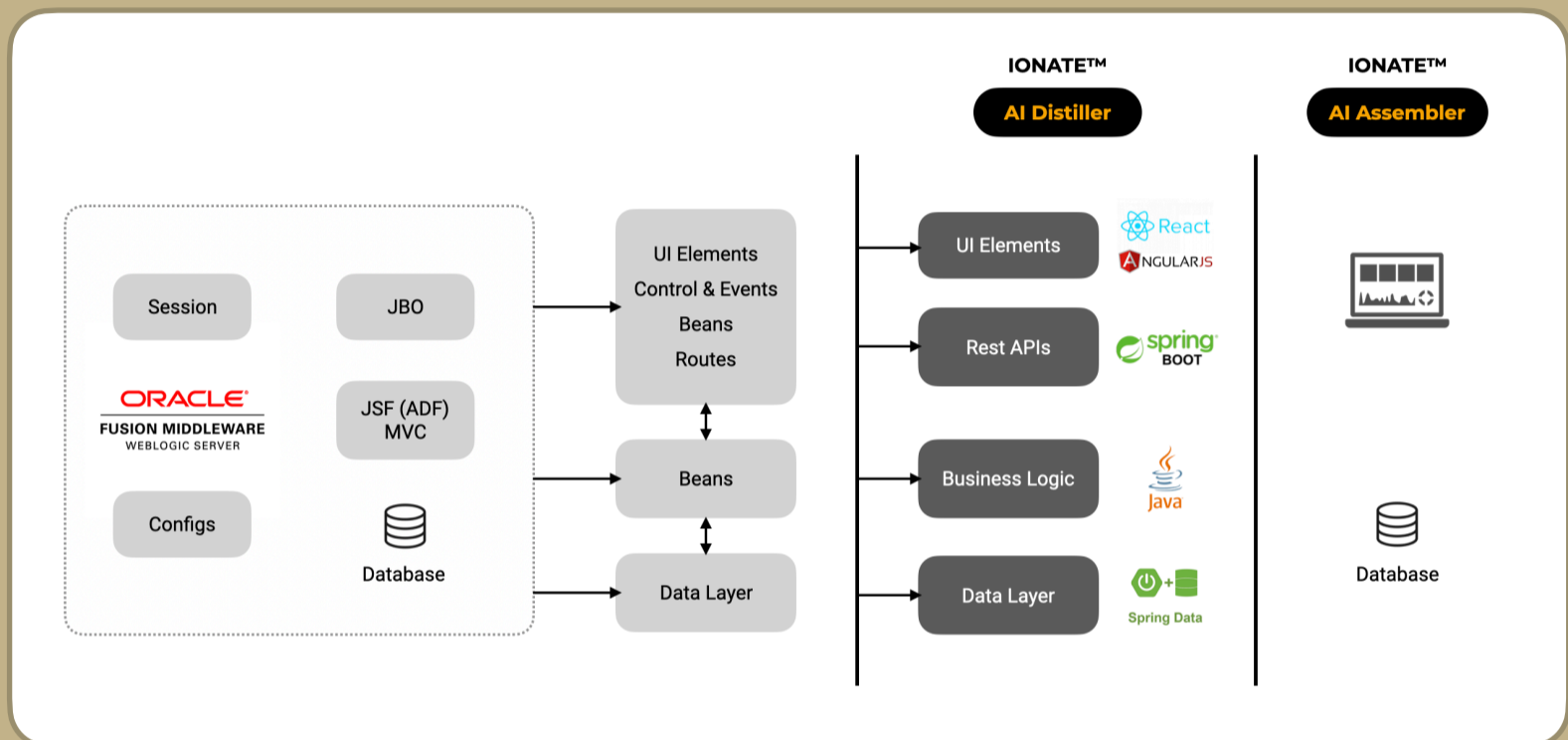
**Since software doesn't age well, we've created a repeatable proprietary process that delivers the goods in a short time (in most cases, under 6 months!)**



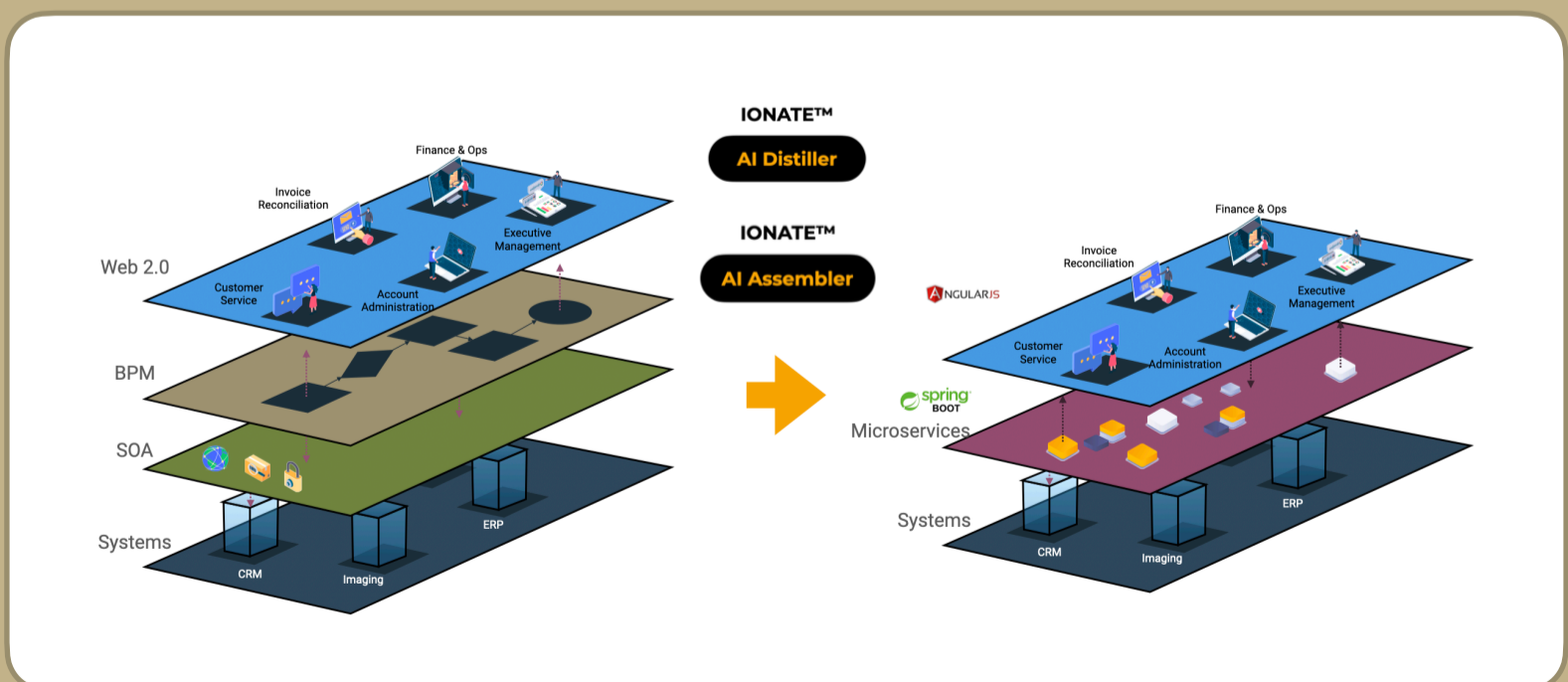
# APPENDIX: CASES

We have helped transform monolith to microservices for multiple clients. The transformation using our proprietary AI / ML SaaS product, IONATE™ APPDATE – typically happens within 6 months. Our engine is mature and ready to convert most of the popular monolith technologies – Oracle SOA, Oracle BPEL, IBM BPM, Oracle ADF, JBOSS implementations (JSF), etc.

## Case | Oracle Middleware to Microservices



## Case | IBM BPM, Oracle SOA to Microservices



# OUR ADVANTAGE



## Time to Market

Our automation ensures agility and a faster go-live!



## Future-proof

Our automation provides scalability, fault-tolerance, resilience, and performance



## Productivity

Automation increases developer velocity and consequently productivity



## Cost-friendly

Our platform eliminates the need for recurring licensing and maintenance costs

And we maintain 100% business parity, which means we absolutely preserve:

100% of Data

100% of Business Logic

100% of Business Processes

## Other Legacy in our repertoire

Thick clients – Oracle Forms, Visual Basic, Delphi, Powerbuilder, Centura, Clarion

COBOL / CICS / RPG – Mainframe, Mid-range, Fujitsu, Unisys, HP, CA



**IONATE**

Global HQ  
San Francisco

[sales@ionate.io](mailto:sales@ionate.io)  
[www.ionate.io](http://www.ionate.io)

### About Ionate

Ionate provides an endgame (turnkey solution) for Digital Transformation with its proprietary AI / ML platform and products. Enterprises can not only modernize their legacy systems, but also operate on high-performing infrastructure and scale their business, with security, predictive forecasting, and data protection.

Ionate is a registered trademark of Ionate, Inc. All other trademarks or service marks are the property of their respective holders and are hereby acknowledged. ©2021 Ionate, Inc. All rights reserved.