outsystems®

# Why RAD is Right for Mobile and Web Front-End Development

Digitalizing the customer journey requires front-end touchpoints that are easy to use, attractive, and full featured. The explosion of mobility has transformed the way customers are interacting with companies – and to remain competitive, customer-facing applications have to be deployed fast and continuously iterated.

## The customer journey has gone digital

In order to achieve digitalization in today's rapidly changing landscape, IT must fundamentally change the way applications are built and delivered. End users (employees, customers and partners) are expecting an easier way to access and interface with technology. Subsequently, the demand for highly complex, easy to use but secure apps is pressed upon IT teams.

Further, rapid evolution in the web and mobile space has dramatically impacted the delivery time in which applications are expected, which means IT must deliver quickly. Users of these apps also expect an attractive UI, a low learning curve and an intuitive app that performs essential business functions.

Finally, an IT department will have its own requirement – the application lifecycle management (ALM) or the process of development, delivery and management of applications must happen in an agile way. App users will always have a plethora of requests from simple additions, like adding a form field, to a completely reorganized customer engagement process. To avoid complete breakdown in productivity of the IT team, the application management process must accommodate any kind of change, large or small. Even if the department outsources mobile app production and delivery - it's critically important to have a lifecycle process that is agile; completely supporting the organization's databases and other systems.

Organizations today have "...crushing and very heavy demands on application development and delivery professionals to deliver software faster." - John Rymer - Forrester Research

## A need for speed (ease of use, and intelligence)

How fast do applications need to be developed and delivered? The average expected timeframe today is between 10-16 weeks. For some very complex applications that could extend a few weeks, but generally this is the time frame business users are expecting apps to be delivered. Delivering applications six months or a year from the initial request is almost redundant in today's adoption rate

of technology. Can existing IT frameworks deliver applications in weeks that are highly complex, easy to use, quick and easy to change and manageable?

## Existing frameworks don't address new requirements

The short answer is probably not. Here's why: whether IT departments are operating with a legacy stack or have modernized to incorporate an open source framework, they are still not able to meet every request for the revitalized customer journey.

## Legacy stacks

Legacy stacks are, unfortunately, the dinosaurs of the app development world. They represent a thorough production process of applications, solidly built by hand-coding, and cemented with "once-and-for-all" integrations which use monolithic systems. They also make rapid app development and delivery nearly impossible (or possible, but prohibitively expensive) Using agile methodology along with a legacy framework does not typically deliver the speed required in today's app delivery market.

## Open source stacks

On the other side, if IT teams have embraced a more modern open source IT stack, such as MEAN (MongoDB, Express.js and Angular.js, Node.js), it's quite possible to build applications rapidly and in an agile manner. But now instead of attempting to force a slow, legacy framework to produce quickly, users of a MEAN stack have to deal with continual maintenance of applications. Quite often, this iterative process is broken because good developers don't want to get stuck forever updating an already deployed app. If the lifecycle can't be maintained, then the agile methodology is fundamentally broken. What's the use of creating apps quickly if they can't be iterated easily?

All in all, the primary challenges for adopting an open source stack to deliver apps are complex. Some organizations may experience more significant struggles over others, depending on the current level of a company's digitalization of front-end processes. Typically though, the critical pain points in adopting an open source stack revolve around:

**Setting up the initial stack –** It can be difficult to acquire the right team of developers who understand most, if not all, components of the application architecture, along with knowledge of each language that corresponds to the architecture component.

**Integration of the stack within existing systems –** No matter how modern an IT department claims to be, there will always be older systems of record or big monolithic databases that require certain levels of integration. Merging new fluid languages with old static systems can become a complex undertaking, that may prove difficult and time consuming.

**Creating a continual pipeline of app production –** If the entire app lifecycle isn't handled quickly and efficiently from end to end then the stack is not really solving the essential requirements for building applications in today's development landscape. Add that to the often immature agile processes

**Autoliv on building applications with a legacy stack:**

"We've built applications internally and struggled. It just took forever, and became harder and harder to change. The apps didn't scale well - they might be ok for departmental applications, but attempting to scale it to a global operation all over the world never worked well and it was difficult to maintain performance over a broader scale."

Autoliv is the world's largest automotive safety supplier.

**Typical tools used in today's IT**

| | Typical Enterprise Legacy Stack | Typical Modern Stack (e.g. MEAN.io) |
|---|---|---|
| Front-end | Java or .Net | Angular JS |
| Back-end | Java or .Net | Express / Node.js |
| Database | Oracle or SQL Server | Mongo DB |

in an organization, and a completely broken delivery model arises.

**In the end, creating a continual app pipeline is possible with an open system stack –** but it's not easy, and it could end up being quite costly in terms of time and management. Open source stacks do provide the opportunity to be more flexible than a traditional stack, but the application lifecycle suffers because it's still not fast or flexible enough.

## The conundrum

This situation puts IT leaders in a very difficult position. They are challenged with delivering the "holy grail" of applications that must be:

- Highly usable
- Attractive with an intuitive user interface
- Agile, able to be rapidly changed
- Delivered fast (avg 10-16 weeks)
- Complex, handling in-depth business processes

Every time an app needs to be created, there is the common expectation that the final deliverable will contain most, if not all of these requests. How do IT architects utilize their current resources to offer continual delivery of highly functional apps that meet all these criteria?

That's where rapid application delivery platforms come in.

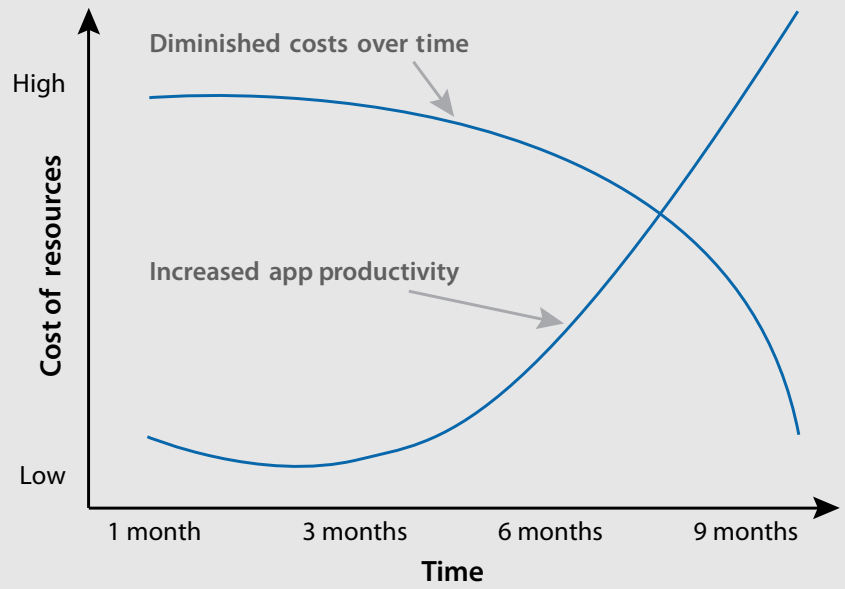
## So what to do? Transform IT with a RAD platform

Rapid application delivery (or RAD) platforms offer what's coined by Forrester as a "low-code" method of creating applications in an agile manner. RAD platforms give IT leaders the opportunity to use current skill sets and resources as well as integrate their existing systems. Developers can not only design whole portfolios of apps – but can develop, test, deploy, review, and manage them all in a fraction of the time of traditional approaches.

Since the full app lifecycle is accounted for with RAD platforms – the need for highly skilled resources from the IT department diminishes while the production and regular management of apps goes up.

Going back to the "holy grail" of requirements, a quality RAD platform must meet each one. The question then turns from "DOES the platform meet each requirement?" to "HOW does the platform meet each one?"

**The value of a RAD platform**




## A robust RAD platform offers the ability to optimize digital touchpoints

By utilizing a robust RAD platform, workflows can be optimized to simplify the mashup of multiple systems in rich unified user interfaces. User adoption can be also optimized because the platform provides the dynamic ability to rapidly iterate on requests, rather than slowly hacking through underlying code or messing with the framework of an application.

## Deeply complex applications are simplified when developing with a RAD platform

Enterprise level applications often require multiple and deep integrations with various databases or systems of engagement. A good platform handles the long tail of integrations – even if the system in question changes or updates its structure.

## Developers get to spend time developing and taking pride in their work

A RAD platform frees up the IT teams' time. Because there is minimal hand-coding involved and the drag and drop interface is easy to use, the team can spend more time creating applications that are not just functional but beautiful and intuitive to use.

## Long delivery times become a thing of the past

Because a RAD platform offers automated deployment and continuous integration, app production and delivery become nearly immediate making delivery within a short time frame a reality.

When examining a platform as a comprehensive toolset to provide an organization with a method of rapidly producing quality applications, it's easy to see how RAD platforms excel for supporting development from front-end through the back-end while integrating with the database/integrations layer.

However, not all RAD tools are created equal. For starters, there are low-code tools that allow IT to develop applications – but not deliver them. A great app tool is one that can rapidly develop and deliver and provide full application lifecycle management (ALM) within an organization's existing set of people and resources.

**NES Financial on speeding app delivery with a RAD platform:**

"We've had on average, a 5x reduction in time to market for our applications. We've been able to simplify our development and deployment processes, as well significantly improve our overall operations."

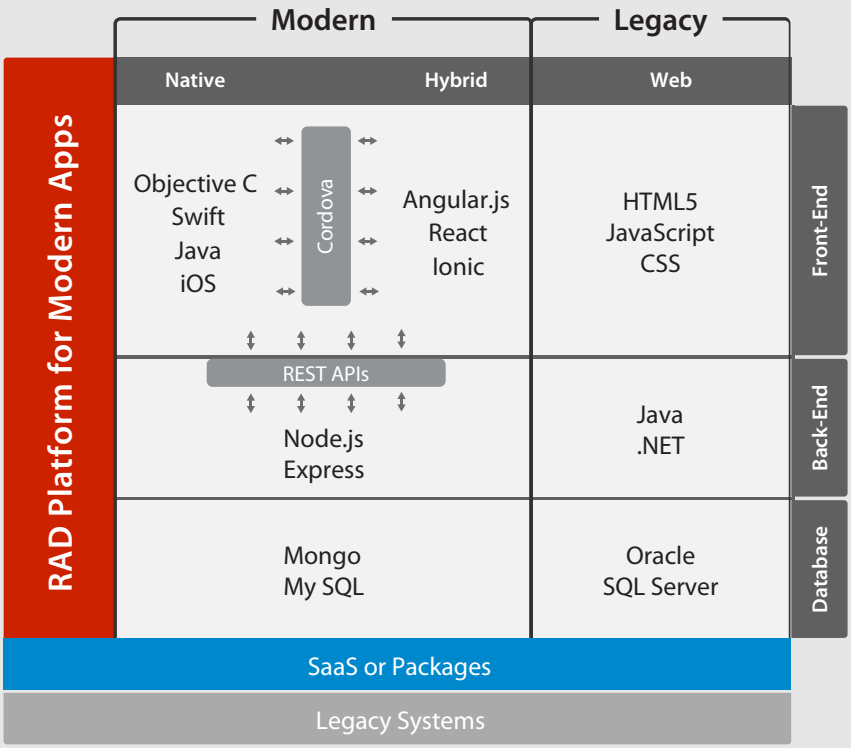
NES Financial is a leading financial technology organization.

## But front-ends aren't everything – back-ends are equally important

It's commonly thought that "front-end development is the most important" when it comes to application development. However, when the continual application lifecycle is examined, the back-end processes need equal amounts of attention. Often, back-ends get redesigned to accommodate for the new customer journey and intuitive front-end applications. The business logic for an organization's portfolio of apps must be fully examined to ensure that appropriate data is being served to the app user while also saving and storing incoming data for other uses within the application.

Rapid application delivery platforms provide a multi-layered approach where back-end processes are just as important to address when redesigning the customer journey. They help redesign current back-end operational processes, actually shifting digital transformation from "just a new set of apps" to a renewed end-to-end customer centric operation and optimal user experience. Learn more about back-end optimization and why BPM's are missing the mark.

**Where a great RAD tool fits - support for the full stack**




## The right choice for IT

Whether IT is ready for it or not, the customer journey is changing and users are demanding more from applications in less time. As your organization puts consistent pressure on IT departments to meet these needs without increasing costs – it's up to the leader to stay informed of the right methods and tools to help his team perform efficiently. A quality rapid application delivery tool can help address this set of issues facing IT today.

**Why is OutSystems Platform the right choice for front-end development?**

- Fast delivery - deliver enterprise-grade applications, six times faster than hand coding.
- Fast change - rapidly iterate, delivering new features across the full application lifecycle.
- Skills reuse - use existing resources, no new specialized skill sets required.
- Powerful user interface - build beautiful, user-friendly applications that run across all devices and screens.
- Powerful RAD - deliver advanced, end-to-end applications with full-stack support for successful front-end development.
- Integration - easily integrate complex existing systems.
- Open platform - extend standards-based code and stacks with your own for development without boundaries.
- Open architecture - add and update your apps easily in a completely open framework.
- Enterprise-grade architecture - deliver apps built for the enterprise with scalability, security, reliability, and governance.

**Download more resources or schedule a demo of OutSystems Platform at** www.outsystems.com

**Share this article**



outsystems®