# Enterprise Grade DevOps with OutSystems

## Intro

In the last recent years, I’ve been seeing **OutSystems** helping large enterprises developing their mobile and web solutions. In several vertical segments, helping their customers from building their public websites to internal mobile apps. For those who have never heard of it, let me briefly try to introduce it:

OutSystems provides an enterprise **Rapid Application Development (RAD)** platform. This platform allows you to create, deploy, change, and manage custom mobile and web applications delivered seamlessly across all devices. Nowadays it’s available as a cloud or on-premises solution with an open architecture.

Now, as a developer, I’ve had several healthy discussions about RAD tools, and I still prefer the broad options choosing technologies, languages, patterns, IDEs, but most importantly this days, the DevOps cycle and respective toolchain. Yes, there are multiple scenarios where RAD tools are the best choice, especially if they deliver as promised.

Large enterprises usually have complex information systems. From legacy technology still running in critical workflows, heterogeneous environments and teams. In the other hand, there are also the innovative teams, with modern development environments and a cool stack running in production, full monitoring and telemetry feeding the modern DevOps cycle put in practice.

So far, I’ve witnessed this large enterprises not being able to have a common DevOps cycle between Outsystems Apps and Non-Outsystems Apps. What happens when you have a large project where in a single release you must sync legacy Line of Business apps (running in your UNIX and windows servers), rest APIs, cloud backends…. AND Outsystems Apps (Mobile and Web)?

Some invest some time and resources, but not all.

I’ve been working with Visual Studio platform for years, and the recent strategies have brought a very interesting value proposition. For me top highlights:

* Cross platform IDE and code editors (love VS Code by the way!)
* A full modern DevOps cycle toolchain – plan work, build, test, release and monitor
* Completely open to Open Source tools (ex: Jenkins or Kubernetes). If you like your tools you can integrate with Visual Studio, and continue to use it.
* An extensible platform, bringing the community together with an awesome marketplace.
* A cloud based version (Visual Studio Team Services) with updates every 3 weeks! – yes, just read about Microsoft DevOps journey.
* An on-promises version (Team Foundation Server), that can have agents both in Linux or Windows.

I decided to contribute creating a Visual Studio extension to integrate with Outsystems platform.

You can read more details in the following sections, or skip everything to the walkthrough.

## Outsystems meets Visual Studio platform

Not being an expert in Outsystems Platform or development, I had to do some research around their current versions. They use model-driven approach to configure the app layers — UIs, data model, web services and/or APIs. Developers can still incorporate their own custom Java or C# code or libraries, and compose them as part of the model, as well as custom JavaScript (and CSS) for front-ends.

One of the Outsystems platform’s module is Lifetime. Lifetime is responsible for managing versioning, deployments and environments, hence managing the releases. And the best news is that, recently a Lifetime API (Swagger defined) was exposed allowing anyone to consume this API.

The Outsystems Integration is now available, and it starts by allowing any enterprise to create, maintain their DevOps cycle, while integrating with Outsystems platform. This extension focuses on:

* A Service endpoint in VSTS/TFS configuring the Outsystems platform;
* An Outsystems Release Task, allowing tag and/or deploy Outsystems Apps;
* Getting the best from VSTS/TFS and Outsystems Lifetime, for release management purposes;
  + Automatically versioning apps;
  + Complete trackability for every release;
  + Release definition cloning for multiple environments;
  + Schedule automatic releases and/or request on demand
  + An approval system for release executions

The extension was completely built using Typescript in Visual Studio Code, and using some ES6 improvements. If you like Promises and Async/Await, this is a good example.

I’ve decided to make it open source and available on Github, so everyone can collaborate, send feedback, or register bugs!

## Conclusion

Hope to see enterprises, small development teams using this extension, so they can have a single common DevOps cycle between Outsystems Apps and non-Outsystems Apps.

It will be easier to build and manage an Enterprise-grade flow with higher developer productivity targeting Multi-channel and heterogeneous developments, while reducing operational costs.

This is one of the biggest advantages of current Visual Studio platform – to easily integrate other tools and make operations/developer life easier.

In the future, we expect that Outsystems platform’s surface of integration will increase, allowing trackability from repositories, work planning (agile or scrum), builds or monitoring. For now release was the most relevant use-case, and it can be done in the cloud (hosted agents) or on-promises agents (linux or windows). It should cover most of the scenarios.

## Walkthrough

Here I will deep dive into Outsystems Extension available for TFS/VSTS, showing how to install, configure and use. I’ll try to list other possible scenarios that can be explored with this extension.

For this walkthrough, I’ll be using a VSTS subscription and my Outsystems platform is running in my Azure subscription.

Installing any Visual studio extension is extremely easy and fast. Just go to the Visual Studio marketplace and search for **Outsystems Integration**, or try this direct link.

Install the extension on you vsts subscription.

<gif de install>

Now let’s configure the Outsystems Platform in your VSTS subscription, so we can act on it. This process is usually done by Service End-points. Let’s create a Service End-point by inserting some information about our OutSystems platform.

Name:

URL:

StrictSSL:

ApiToken:

<gif de Service Endpoint>

We’re ready to start building our Release definition plans, including Outsystems Apps. In your Releases, create a new Release Definition Plan. Here, you can start by either using a template available that may suite one of your apps, or change with an empty release definition.

Add a Outsystems Release Task. Press OK, and let’s configure the task itself.

|  |
| --- |
| Note: In my Outsystems Platform I had three environments configured: Development, Test and Production.  I used Outsystems Studio focused in two applications – X0 Darts(web) and X0M Darts (Mobile). |

<gif de Release Definition Task>

<gif de Release Definition Task – Clone de amb + config>

## Resources

### What is?

* OutSystems provides the most trusted enterprise **Rapid Application Development (RAD)** platform to accelerate digital transformation.
* OutSystems Platform is the fastest and most comprehensive way to **create, deploy, change, and manage custom mobile and web applications** - delivered seamlessly across all devices.
* **Available as a cloud or on-premises solution** with deep integration to all existing systems and an open architecture

### Strengths

* OutSystems Platform uses a model-driven approach to configure the app layers — UIs, data model, business processes, integration workflows, web services and APIs — enabling high-productivity development.
* Developers can incorporate their own custom Java or C# code or libraries, and compose them as part of the model, as well as custom JavaScript (and CSS) for the front end.
* OutSystems supports a citizen developer approach and provides a number of sample apps and industry frameworks that demonstrate how different components can quickly be brought together to build an app.
* OutSystems Platform includes a number of APIs and offers API discovery with usage controlled by organization policies with centralized security governance. It also enables a granular management of user accounts and access roles.

### Cautions

* OutSystems Platform's metadata model to describe the behavior of an app does not give the developer complete control over the generated code, as it limits the developer's ability to define how code is generated from the metadata model.
* While OutSystems' model-driven approach to development accelerates the time to a solution, developers experienced in more traditional app development approaches may struggle. They would need to understand how to balance the use of custom code and libraries against using the model.
* While OutSystems allows app deployment to the OutSystems cloud, on-premises environments, or to other IaaS providers (such as Microsoft Azure or AWS), there is currently no option to directly deploy apps to Cloud Foundry-based environments.