

The RPA system will be used for simple, repetitive tasks (in this case such as invoice processing or payroll management) that could be achieved by a human. However, this system will have the ability to save hours of manual labour, reduce manual errors and derive conclusions from analysing large datasets. The tasks are simple, yet undeniably important for the company to function and for the customers to continue their daily lives. For this is handling their finance. Therefore, we need a development lifecycle that can produce a version of the system in a timely manner, with the opportunity to adapt and scale up the system and act in accordance with all legal requirements. Security of the system must be one of the top priorities when developing, due to the sensitivity of the data it will be handling.

I chose a DevSecOps diagram to represent the development of the RPA (Robotic Process Automation) system for Multiple is due to security vulnerabilities being identified early, rigorous testing and as well as a fast delivery procedure to release new versions as fluently as possible.

Due to this system automating basic functions for a financial service company, it is essential that updates of the system are processed as smoothly and efficiently as possible to avoid hindering current operations. This enables Continuous Delivery by using tools such as Gitlab or Azure DevOps. This is done in the later stage of the development lifecycle (the Deploy phase). By now it would have been thoroughly tested and theoretically should not cause any negative affects on live procedures.

The automated security tools like the SAST (static Application Security Testing) and Code + Infrastructure scanning found in the Build phase, or the DAST (dynamic application security testing) found in the test phase allow consistent security standards across the project.

This methodology also allows firsthand feedback. For it can have a version of the system working live, handling actual customers and actual data in the company. This can be monitored and reviewed, and suggestions can be delivered back to the development team to implement in the next version. Constant feedback loops like this can also mitigate security vulnerabilities and Incidents (like the system going down).