HM Revenue & Customs

# Re-homing complex tax applications to the cloud.

Supporting HMRC to save millions in operational costs and deploy service improvements in minutes—not months.

EQUAL EXPERTS


## As a forward-thinking and customer-focused UK Government department, Her Majesty's Revenue and Customs (HMRC) is committed to 'making tax digital'.

In short, that means providing easy, efficient and reliable tax services for over 50 million business and individual customers. To deliver on this strategy, HMRC shifted many pre-existing services to their cloud-based **Multi-channel Digital Tax Platform (MDTP).**

But what about complex, high-volume and high-revenue services—like Self-Assessment Online—that had been running for many years? Completely re-writing the apps from scratch would take far too long and cost far too much. But maintaining the applications on expensive physical servers was equally inefficient, especially in the wake of the Global Financial Crisis, when governments were more prudent than ever before.

**Here's how HMRC transitioned complex legacy applications to the cloud, cutting significant infrastructure costs while establishing more control and flexibility for ongoing updates.** And all with minimum impact to service availability for millions of customers.

## This case study will help you to understand:

**Why HMRC re-homed 55 complex tax applications, transitioning from physical servers to the cloud.**

**The cost benefits and operational improvements of re-homing to the cloud via 'lift and shift' migration.**

**How to complete a complex re-homing project with minimal service downtime or loss of potential revenue.**

# Table of contents.

ABOUT HER MAJESTY'S REVENUE AND CUSTOMS.

## Her Majesty's Revenue and Customs (HMRC) is a non-ministerial department of the UK Government.

HMRC is responsible for the collection of taxes, the payment of some forms of state support, and the administration of other regulatory regimes, including the national minimum wage and the issuing of national insurance numbers.

**57,000**
employees.

**45 million**
consumer customers.

**5 million**
business customers.

**£605 billion**
generated in revenue through 2017/2018.

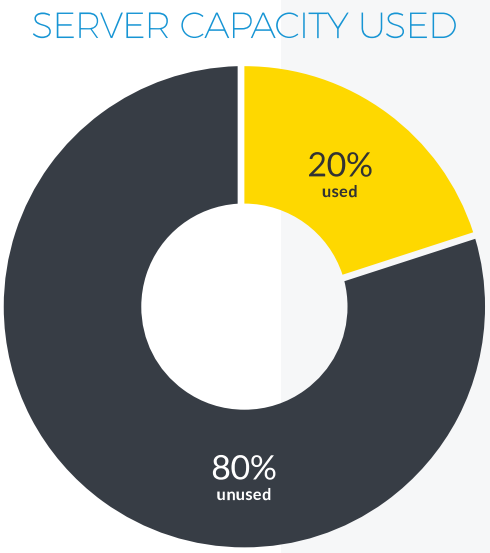
MIGRATING TO THE CLOUD FOR MORE COST-EFFECTIVE INFRASTRUCTURE.

## We all know the feeling: Tax time hits, and there's a rush to make submissions before the cutoff date.

For HMRC, that means an influx of **600 million Self-Assessment transactions within a ~24- hour period.** That's where HMRC's legacy self-service tax applications come into play. Originally built in Java (using Spring), the apps support people to find tax information and file their submissions online.

Before re-homing, the apps were housed on physical servers. These servers were configured to cater for the maximum volume of traffic required—processing 600 million transactions and 3.5 billion metric data points in 24 hours—at all times of the year. Plus, the entire capacity was duplicated to a disaster recovery (or DR) state to allow for fallback in the event of any critical incident.

By re-homing to the cloud, hosted on Amazon Web Services (AWS), HMRC's applications can dynamically scale resources to match demand, rather than continuously operating at—and incurring charges for—peak-period utilisation.




### HMRC's original physical infrastructure consisted of 32 servers in total.

Eight servers were live. Six were staging servers. Four were for a testing environment. The live and staging sites were duplicated for disaster recovery purposes.

Outside of the main peak period, the applications only required about 25% of their total live staging and test capacity servers to support business-as-usual traffic (in other words, 0% of the DR capacity was used). But despite HMRC's annual rush period only lasting between 24 and 48 hours, the servers were set for the highest levels of utilisation, 365 days per year.

If the highest levels of computing resources weren't available during a peak period, every digital application housed on a specific server would fall over. This would prevent HMRC from processing critical revenue streams, such as VAT or PAYG. Further, loss of services would create additional pressure by triggering enormous amounts of customer enquiries and traffic to help desks.

Unfortunately, it wasn't as simple as 'turning down the server capacity' in quiet periods.

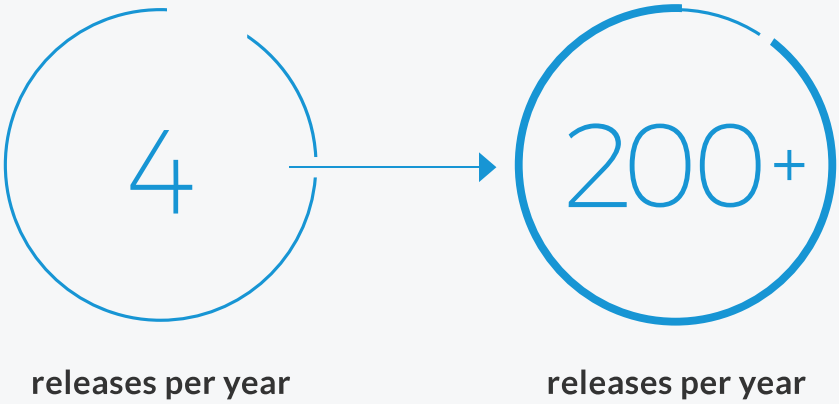
Any change in resources throughout the year would require manual effort from the data centre supplier. They'd have to plan, scale, test, and provision DR testing. **They'd have to manually add or reconfigure computers, storage, and networking, then manually ramp things up for performance and stability when the inevitable tax-time tsunami rolled in.**

All told, any change in server capacity triggered a months-long process. In order to avoid the costs associated with manual reconfiguring, the infrastructure was essentially set to deal with HMRC's busiest hour, of their busiest day, for the entire year.

**After re-homing, and with the services now operating via Amazon Web Services (AWS), things can automatically scale as needed.**

For example, from April 2017 to April 2018, 10.1 million Self-Assessment tax returns were filed. Of those total returns, 4.1 million were submitted in January.

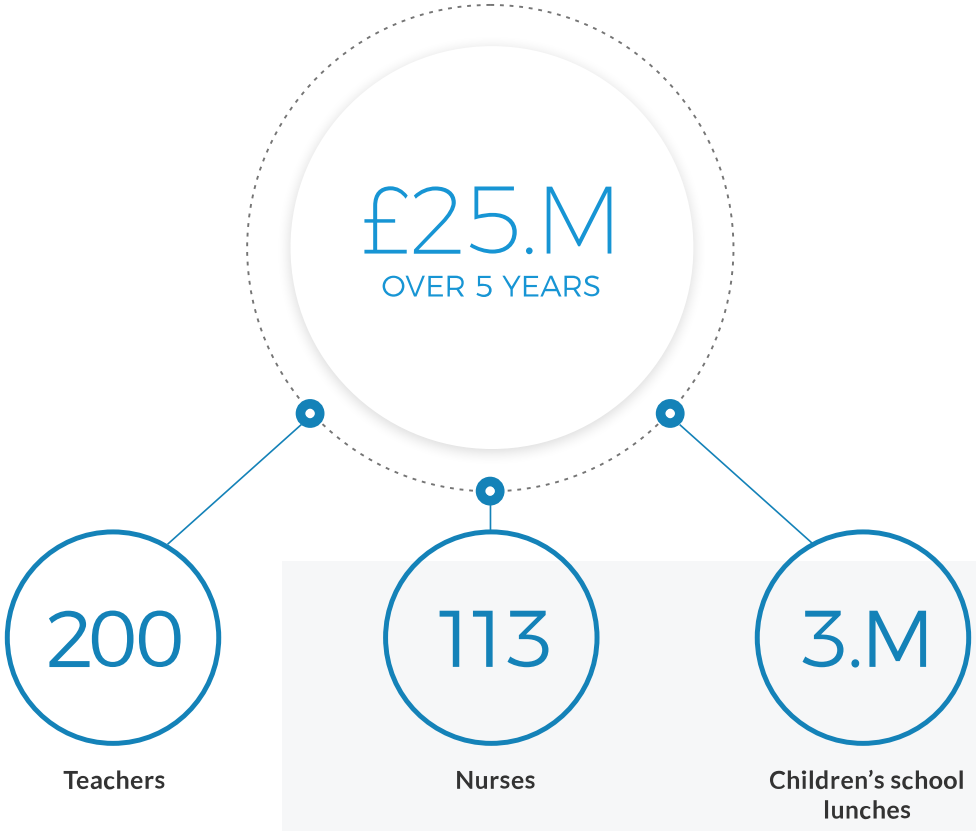
Thanks to the new flexible and automated infrastructure, HMRC were able to scale up from 4 to 32 releases of the Self-Assessment filing application—coping with the rapid demand for additional performance and stability—before scaling back down once the filing deadline had passed.



## Shifting the legacy apps to the cloud means HMRC can automatically scale resources as they're needed throughout the year.

As resource requirements shift, the costs immediately adjust in sync, rather than staying constant despite huge fluctuations in usage.

So, when Self-Assessment time hits each year—and traffic ramps up dramatically—so do the instances of applications. The rest of the year, the capacity can drop back down to meet business-as-usual demand. Or seamlessly automate to meet any other smaller peaks that eventuate throughout the year. The result? **Over 5 million pounds saved each year** in infrastructure costs, the equivalent to 200 teachers, 113 nurses or 3 million school dinners.



ABOUT THE RE-HOMING PROCESS.

## To make the most of any existing code—and save on unnecessarily re-writing applications — HMRC approached the re-homing process through a 'lift and shift'.

While it sounds neat—picking up a suite of pre-existing applications and shifting them to a new home in the cloud, largely as is—the reality of the situation required some technical ingenuity.

**The applications were large, complex, and regularly serving huge numbers of customers: upwards of a million each month.** Plus, many were closely linked with critical business processes and high-earning revenue streams, like Self-Assessment Online.

Any period of unavailability throughout the transition would prevent people from finding and submitting tax information. And that translates to significant loss of revenue for HMRC.

To avoid any significant cost of delay, HMRC needed to move away from their physical servers as quickly as possible. The faster the shift, the less would be spent on legacy server contracts, so the greater the benefit. But the speed of migration couldn't come at the cost of service availability.

## The goal was clear: rapidly shift HMRC's legacy applications from 'A' to 'B' while maintaining service continuity at all times.

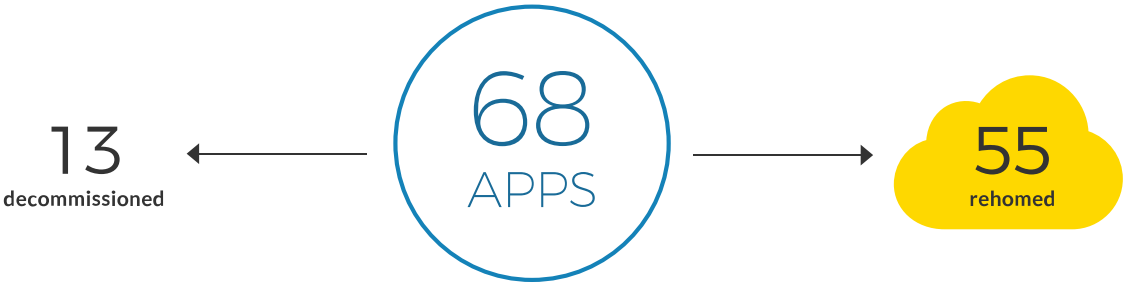
To deliver a speedy migration while still serving huge numbers of people using the applications, the team took a unique approach to re-homing.

The approach began with creating generic services for authentication, auditing, user routing, and filing submissions, which could be used across multiple Portal applications.

Then, the team looked at Portal application architectures.

Each app was a monolithic Java application: a front-end and back-end with business logic built into the application. We decided to decouple the applications, splitting them into a frontend for the presentation layer, and a backend for business logic and data access.

De-coupling the monoliths provided HMRC with greater flexibility and the ability to deploy specific front- or back-end changes without changing the regression of the system.




Next, the applications were strategically sliced up so that any common servers or integration points—like authentication or simple content serving—could be shared by the remaining portions of unique code. These could then be easily 'plugged in' to use the shared parts.

Any datastore remained untouched. All the data also resided on separate infrastructures. Data was stored locally until a submission had been lodged, then stored in a data layer that was the existing un-migrated layer. This caused a negligible amount of latency issue, which was mitigated through improving latency of authentication, routing, and more.

The benefit? By migrating incrementally, we could decommission servers gradually and transition spend away from the expensive physical infrastructure, faster.

**Re-routing traffic for service continuity, controlled monitoring and risk mitigation.**

Thanks to an innovative approach to traffic management throughout the migration process, HMRC could dynamically re-route traffic between the legacy infrastructure and the Multi-Channel Digital Tax Platform at a service, user, or page level.

This level of granular control empowered HMRC to send small amounts of traffic to the newly re-homed services as they shifted to the Digital Tax Platform, proactively monitoring the performance of that service at all times.

## In addition to the technical process of re-homing, a project of this scale and complexity warrants an equally sophisticated delivery approach.

We had to ask, and answer, the following questions throughout the Discovery phase.

### How do we prioritise value for HMRC?

- How do we ensure the most valuable applications are moved first?
- How do we establish how long the migration will take?
- Which applications do we migrate? Are there any that would be better to de-scope based on effort vs value?

### How do we maintain usability throughout the migration?

- How do we ensure the applications will serve users continuously throughout the migration, minimising service disruption and loss of potential revenue?
- How do we ensure the applications are meeting user needs and performing as they need to?

### How do we create confidence in the approach?

- How can we prove this will work?
- How do we validate our projected effort, to ensure that the cost of the migration doesn't ultimately outweigh the value associated with re-homing the apps?

## A strategic, staggered approach to the 'lift and shift'.

First, the team migrated a basic application with very few interdependencies. Choosing a simple content application proved the apps would work when re-homed in the cloud. However, it also proved functionally useful, given that nearly every other application needs to serve content as well.

**With a successful proof-of-concept in place, the team developed a roadmap to prioritise re-homing services.** Those with the most potential cost savings were migrated first, along with any services requiring time-critical legislative changes throughout the remaining delivery period.

Right off the bat, the team deliberately chose a larger, more sophisticated (and therefore somewhat risky) second application. This app relies on multiple integration points and serves high volumes of customers at all times.

**The thinking? Solve the most sophisticated technical challenges early and leverage those solutions across the remaining applications.**

Plus, the more functionality the team moved to the cloud, the faster we could switch those costly physical servers off to save more money.

The roadmap was delivered in an agile approach. This ensured the team could collaborate, iterate, and respond to critical technical challenges throughout delivery.

**Learn more about the questions you need to ask and answer before engaging in a lift-and-shift migration.**

### Accommodating a host of existing technical constraints.

As a lift and shift—which is designed to maximise time and cost efficiencies associated with re-purposing existing code—nothing was written from scratch. Throughout the entirety of the re-homing project, all solutions had to conform to and accommodate the requirements of any pre-existing technical platforms, or legacy business applications. This included requirements for the following:

- Adapting applications as Scala microservices
- Using javascript for the front-end
- Building using SB
- Running in docker
- And more...

### Transitioning to microservices for maximum benefits.

The team transitioned the applications to microservices. Or, reimagined singular monolithic applications with high levels of functionality into networks of smaller applications that communicate with each other.

By adapting the legacy services to function as Scala microservices running on docker containers, HMRC were able to benefit from previous investments in a digital platform, named the Multi-Channel Digital Tax Platform (or MDTP). Specifically, by leveraging the efficiencies created by the MDTP's superior deployment pipeline and monitoring capabilities.

To find out more about the potential value of building and maintaining a Digital Platform, **read the Equal Experts Digital Platforms Playbook.**

### Integrating with newer, best-practice services where possible, or leveraging other services built on HMRC's digital tax platform.

As services transitioned to the MDTP, HMRC integrated with newer, best-practice services to improve the customer experience overall. For example, after re-homing to the MDTP, the team replaced an antiquated service used to validate customer bank details and address information. In other instances, we leveraged services that were already built onto the MDTP, and that no longer needed to reside in the legacy apps. Examples included authentication, bank payments, and address validation.

RELEASING IMPROVEMENTS IN MINUTES, NOT MONTHS.

## As each application was successfully re-homed, HRMC had greater flexibility and control over new feature releases.

**Once an application is re-homed, updates to specific services can be implemented almost immediately—and with less downtime—to ensure the best possible experience for customers and stakeholders alike.**

Before re-homing, updating the applications was a slow and laborious process. The rate of change is much slower for physical infrastructure than when making changes in the cloud.

**Historically, HMRC previously had two major IT releases per year.** This was largely because the release process itself took up to six months: three months to impact, and three months to code and test. Of course, longer lead times typically means a longer list of features being pushed into the one release, which makes each release a riskier proposition.

Now, HMRC can quickly release updates to improve functionality based on customer feedback, respond to new requirements from Product Owners and Stakeholders, and even roll back to previous instances of applications if required.

Enhanced monitoring also means the team can proactively identify potential issues—and fix those more efficiently—to minimise any potential period of downtime.

As part of the 2017 budget announcement, the Chancellor of the Exchequer announced that stamp duty would be abolished for all first-time home buyers, up to **£300,000**.

Within **30 minutes of that announcement**, HMRC's Stamp Duty Land Tax service had been updated to reflect the change.

## Enhanced technical solutions, with improved team performance to match.

Once the applications were re-homed, we worked in close collaboration with HMRC's internal teams to improve the way they interact and operate.

This largely relates to the support of each application in the wake of re-homing. Now, each support request goes directly to the team that migrated the application to the cloud. In the previous structure there were distinct and siloed teams for development, regression, deployment, and post-production support.

By adopting 'you build it, you run it' principles—where the team that migrated the application directly interfaces with the maintenance and operation of that application—we could decrease the size of the overall teams (as there is no longer a requirement for separate build and support teams), decrease month-by-month support costs, and decrease the time involved in resolving incidents from days to hours (and sometimes even minutes).

# £8 million

saved per annum by insourcing production support functions.

While it's a common team structure, operating with siloed, distinct teams for each stage of development, regression testing, deployment, and support is largely inefficient. Let's consider the following team structure, which typically functions in a waterfall structure, and consists of:

- Development team
- Regression testing team
- Deployment team
- Post-production support team

Hypothetically, let's imagine there's an issue with any one of HMRC's applications.

The post-production support team will likely look for issues in deployment as the first port of call. If there's no identifiable problem with deployment—and the team knows the regression test passed, because otherwise it would not have gone through the change process—they'd raise the issue with the development team, assuming an issue within the code itself.

More than likely, the development team has moved on to another project. Halting progress on the current project, they switch context to review the code, trawling back through previous work to identify the potential issue.

Once the problem is identified and a fix is implemented, the fix needs to be reviewed by the regression testing team. They'll perform testing and regression, before the fix is passed to the deployment team to push the change to production.

From there, the support team will identify that the fix has worked. Or, worst-case scenario, the fix hasn't been successful, and the entire process is repeated.

While migrating applications to the cloud, we noticed the opportunity to improve team structure. By ensuring that the support teams also have the capability to deploy and test new fixes and features, we consolidate separate teams into one, and eliminate the inefficiencies created by multiple handovers.

Additionally, this creates:

- Vastly improved collaboration between technical teams and business stakeholders.
- Reduced lead times due to a Continuous Delivery pipeline.
- Removed any external dependencies on third-party suppliers, as internal teams have full ownership of support, maintenance and enhancement of services.
- Less commercial documentation and approval flows to request and deliver work.
- Proactive identification of issues, with no dependence on external suppliers to detect issues.
- More possibility to improve and refactor services through increased speed of delivery. Examples include improving Content Management and improving automated test coverage.

# Want to know more?

Are you interested in this project?
Or do you have one just like it?
Get in touch. We'd love to tell you more about it.