



CTT

Correios de Portugal
Enterprise Architecture & Software Engineering

CASE STUDY

Modernising a decades-old application with no documentation — without simply migrating the past

CTT chose one of the hardest applications in its legacy estate to test a different approach to modernisation. With AMA — Link's Application Modernization Approach — CTT recovered what the application actually needed to do, decided what was still worth keeping, and generated a new application for its target architecture and engineering framework.

28

macro requirements
recovered

21

retained

7

deliberately removed

1

modern application
generated and validated in
CTT's engineering
environment

The problem was not just old technology. It was lost knowledge.

CTT has applications that have supported critical business processes for decades. Some were built on technologies that are now obsolete. Documentation is incomplete or no longer exists. The people who originally developed them have moved on. And many business rules live inside screens, workflows and code rather than in a reliable specification.

The application chosen for this proof of concept was built in Oracle Forms and is business-critical. CTT picked it because it represents the difficult end of the estate, not the easy one.

Instead of asking “how do we convert this code?”, CTT and Link started with a more important question — and that distinction changed the whole modernisation.

What should this application actually do today?



THE STARTING POINT

- A decades-old Oracle Forms application;
- No original documentation available;
- The original development team no longer present;
- Business rules distributed across screens and code;
- Functionality that no longer belonged in the application;
- A completely different target architecture and technology stack.

ABOUT THE CLIENT

CTT — Correios de Portugal

With more than five centuries of history, CTT operates Portugal's universal postal service alongside parcel and express logistics, retail, financial and digital services. That history also means a complex application landscape built across multiple generations of technology.

A familiar legacy challenge: the application still matters to the business, but the knowledge required to change it safely has progressively disappeared.

“The differentiator was being able to start not only from the code or the documentation that existed, but from the behaviour of the application as a whole.”

Felipe Rodrigues

CTT · Architecture & Software
Engineering

Don't migrate the past. Build what the business needs next.

Traditional migration asks how to move an application. AMA asks first what is actually worth moving — and what architecture should support it next. Only then does the engineering begin.

Legacy
application

— recover

— validate

— generate

— prove

— Evolve

ONE CYCLE, BOTH
PATHS

01

Migrate only what still creates value

Retain, retire, modernise, replatform or consolidate before migration effort begins.

02

Know what you are rebuilding before you rebuild it

Recover the requirements from real usage, documentation and code — and validate them with the business.

03

Build for the architecture you actually want

Validated requirements drive architecture, code and tests for the organisation's target technology stack.

04

Validate before you cut over

Prove the modern application behaves as expected before production. In full programmes, AMA can also run legacy and target in parallel before cut-over.

And after go-live?

The requirements remain — a permanent asset linking requirements, tests and code, so the next change starts from business intent rather than another reverse-engineering exercise.

How it ran at CTT

The project did not start with the source code. Link first recorded a business user working with the application and captured its main use cases. Agents turned that usage into an initial requirements structure; the source code was analysed afterwards to complete the picture.

It identified 28 macro requirements: CTT approved 21 and removed seven that no longer belonged in the target architecture.

Built for what CTT wanted next

Instead of staying coupled to database tables, the new application consumes CTT's existing REST APIs and fits its engineering framework, technology choices, container environment and automated deployment process. It was deployed to the quality environment through CTT's own pipeline and validated there by the technical lead who knew the legacy application best.

"If you go directly from code to code, the complexity remains. Going through requirements is what gives you the opportunity to change it."

Pedro Sousa

Head of Enterprise
Architecture & AMA · Link
Consulting

What the proof of concept demonstrated

01

Lost application knowledge became structured requirements

Real usage and source-code analysis reconstructed what the application needed to do — turning undocumented knowledge into an asset the organisation can validate and maintain.

02

25% of identified macro requirements did not need to be rebuilt

CTT removed 7 of 28 requirements because they no longer made sense in the target architecture. Modernisation reduced the scope before code generation even began.

03

Built for the target architecture — not translated from the legacy one

Built around CTT's target front-end and back-end technologies, REST APIs and software engineering framework. The target architecture defined the new application.

04

Modernisation without creating a separate engineering world

Containerised and deployed through CTT's existing automated pipeline, so the modern application fits the same engineering process used across its current estate.

05

A model CTT can evaluate at portfolio scale

The POC created a tested path for a broader portfolio decision: what to modernise, retire or consolidate. Instead of converting applications one by one and preserving the same complexity, a requirements-first approach creates the opportunity to simplify the estate as it modernises.



READY TO START?

Still running an application **nobody** wants to change?

When an application is too critical to ignore but too risky to change, modernisation should not start with code conversion. Start by understanding what it actually does, what the business still needs and what should exist next.

Start with one legacy application.

Talk to our Application Modernization team →

LINK | **LINKROAD**

Av. Duque de Ávila, 23
1000-138 Lisboa, Portugal
+351 213 100 031
info@linkconsulting.com
linkconsulting.com