MIGRATION BLUEPRINT

STRUMENTA

THE MIGRATION BLUEPRINT OVERVIEW

THE CHALLENGE OF LEGACY SYSTEM MIGRATION

Legacy systems are the backbone of many businesses, but they come with increasing costs, security risks, and a lack of flexibility. Companies often face roadblocks such as:

- **High maintenance expenses that drain budgets.**
- **Security vulnerabilities that put critical data at risk.**
- **Compatibility issues that hinder integration with new technologies.**
- **Lost competitive advantage due to outdated architectures.**

Migrating to a modern system is essential, but it's a complex process. That's why we developed the Migration Blueprint—a proven framework to guide your transition efficiently and strategically.

WHAT IS THE MIGRATION BLUEPRINT?

The Migration Blueprint is a structured approach designed to help companies modernize their software with confidence. Instead of diving into migration blindly, we help you:

- **Assess your current system to identify risks and opportunities.**
- **Define a clear migration strategy tailored to your business goals.**
- **Minimize disruption and keep operations running smoothly.**
- **Optimize costs by choosing the right technology and execution plan.**
- **Ensure future scalability so your new system grows with your business.**

Our framework is designed for businesses that want to modernize without unnecessary risks, wasted resources, or lost productivity.

WHY CHOOSE THE MIGRATION BLUEPRINT APPROACH?

1

Clarity Before Action
Many migrations fail due to poor planning. The Migration Blueprint gives you a clear roadmap before any code is touched, reducing risks and ensuring a smooth transition.

2

Tailored for Your Business
We don't believe in one-size-fits-all solutions. Our strategy is customized to your software, industry, and long-term vision.

3

Risk Mitigation
By identifying potential pitfalls early, we help you avoid common migration mistakes that lead to delays and budget overruns.

4

Expert-Led Strategy
Our team has years of experience guiding companies through complex migrations. We bring technical expertise and strategic foresight to every project.

WHO IS THE MIGRATION BLUEPRINT FOR?

This framework is ideal for:

Tech Leaders & CTOs seeking a structured, low-risk migration strategy.

Product Managers who need to maintain software reliability during migration.

Business Owners who looking to future-proof their operations with modern technology.

Development Teams tasked with executing a successful migration without disruption.

HOW IT WORKS

→ **Discovery & Assessment:** We analyze your current system, pain points, and business objectives.

→ **Blueprint Development:** We create a step-by-step migration strategy, addressing risks and best practices.

→ **Implementation Roadmap:** You receive a clear action plan to execute the migration efficiently.

→ **Support & Guidance:** We offer expert advice throughout the process to ensure success.
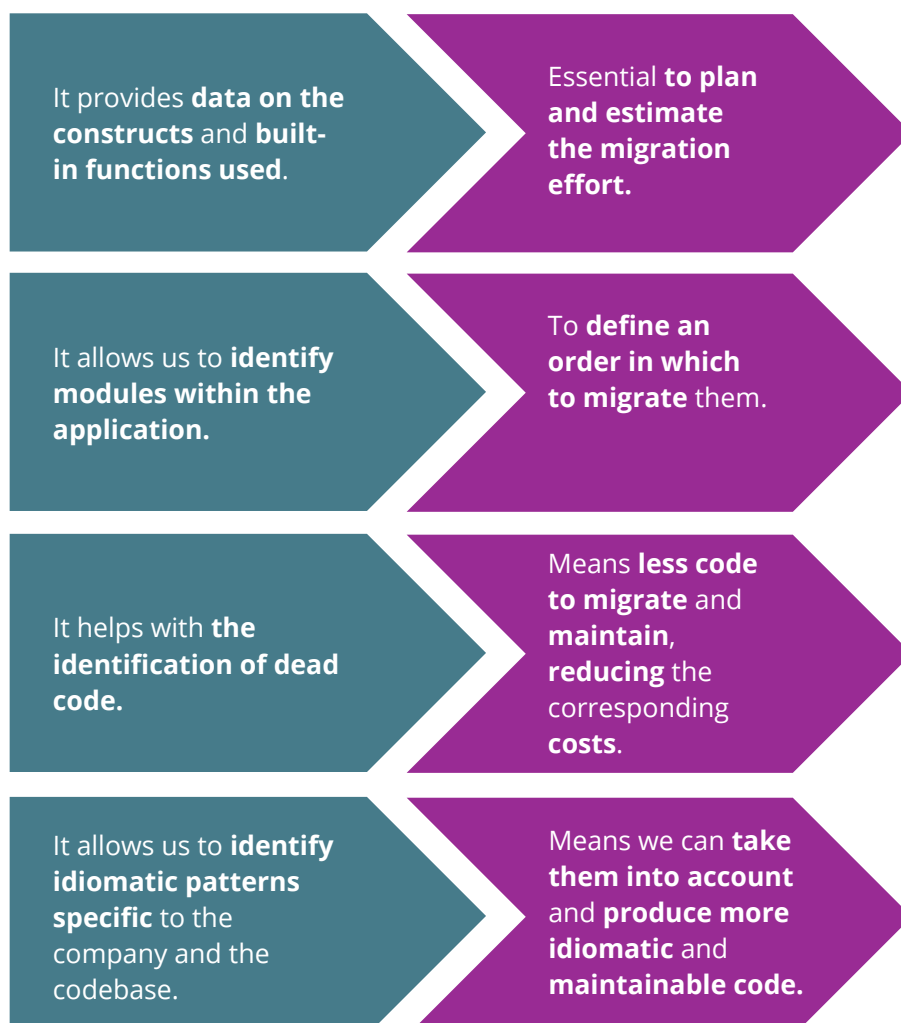
THE MIGRATION BLUEPRINT REPORT

OVERVIEW

The plan is based on the audit of the existing system, the definition of the requirements for the new solution and a plan from moving to the existing system to the new solution. This plan takes into account the specificities of the codebase, and the importance of defining idiomatic code in the new system. It will then examine possible risks and define mitigation strategies for those.

The following report is delivered at the end of the Migration Blueprint engagement. This document would refer to the Code Insights Report, a separate document that would contain all the statistics calculated on the codebase.

WHAT ARE THE BENEFITS?

The Migration Blueprint report provided by Strumenta is a report that defines a clear plan for performing a migration.

It provides **data on the constructs** and **built-in functions used**.

Essential **to plan and estimate the migration effort.**

It allows us to **identify modules within the application.**

To **define an order in which to migrate** them.

It helps with **the identification of dead code.**

Means **less code to migrate** and **maintain**, **reducing** the corresponding **costs**.

It allows us to **identify idiomatic patterns specific** to the company and the codebase.

Means we can **take them into account** and **produce more idiomatic** and **maintainable code.**

STRUCTURE OF THE REPORT

In this section, we describe what should go in the report and how it should be structured.

▶ SECTION 1 - GOALS OF THE MIGRATION

This section should summarize the goals of the migration in half a page or so.

Typically the goals of the migration are: cost reduction, solving developers shortage, maintainability, or scalability.
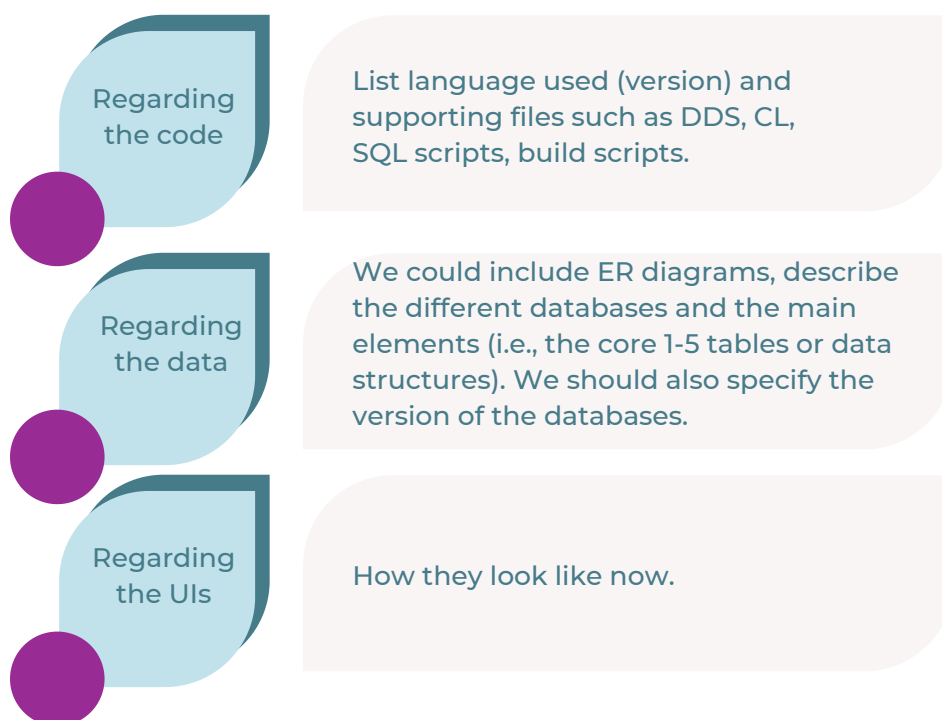
▶ SECTION 2 - THE CURRENT SYSTEM

This section should provide a reasonable overview of the current system. It would typically take a significant part of the report. It could go from 2 to 10 pages.

We start by explaining what are the goals of the system: what does the software do? Who is using it? For which purpose?

We then explain how the system is structured. We should define with diagrams or with words what are the various components of the system: on which hardware do they run, how many different parts there are, where are they deployed, by who, how does this system interface with other systems.

We then list off the different externalities that affected the definition of the current system and explain if any of those changed in the meantime. For example, laws, regulation, corporate rules, security rules, licenses, and any other such element.

We then look at the code, data, and UIs:

Regarding the code

List language used (version) and supporting files such as DDS, CL, SQL scripts, build scripts.

Regarding the data

We could include ER diagrams, describe the different databases and the main elements (i.e., the core 1-5 tables or data structures). We should also specify the version of the databases.

Regarding the UIs

How they look like now.

We discuss the modules in which the application can be divided.

For example, we could have a series of batch scripts, related to processing wire transfers and then a separate set of applications for inserting wire transfers and another one to produce reports.

We then dive into a more analytical part providing:

> Comments about the statistics on the size of the system (number of files, number of lines)

> Comments about the prevalence on the different constructs

▶ SECTION 3 - THE NEW SYSTEM

In this section, we list the requirements for the new system. This section could go from 5 to 15 pages.

We could describe, depending on the case:
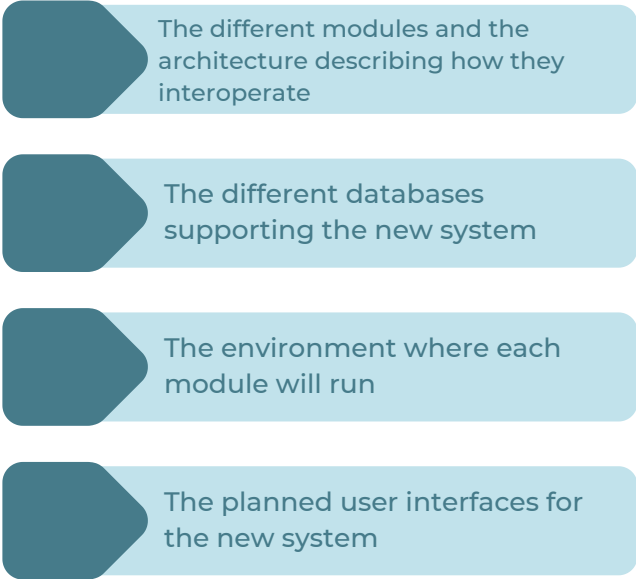
> Which problems from the old system we want to avoid

> What are the requirements in terms of features or performance

> Decisions on the programming languages to be used, their versions, and the frameworks to be adopted

- The different modules and the architecture describing how they interoperate
- The different databases supporting the new system
- The environment where each module will run
- The planned user interfaces for the new system

▶ SECTION 4 - MIGRATION FROM THE CURRENT SYSTEM TO THE NEW SYSTEM

In this section, we define a plan for migrating the existing system (described in Section 2) to the new system (described in Section 3).

This section could go from 5 to 15 pages.

In this section we cover topics such as:

The plan for migration of the UI: which parts of the UI of the new system will be replaced by which parts of the UI of the new system. We will define which parts will be migrated and which ones will be rewritten

The plan for migrating the data, specifying which databases from the old system will be kept as-is and which ones will be migrated and how

Specify how we will translate the different elements. In this section, we describe common code patterns and how they will be translated. This part is key to ensuring idiomatic translation of the code

We discuss risks identified during the workshops and the mitigation strategies we can put in place

We define a testing strategy:

- Provide the Test Specification Template

- Discuss end-to-end testing

▶ SECTION 5 - MIGRATION EXECUTION PLAN

In this section, we provide details on how to implement the plan presented in the previous section.

This section could go from 1 to 3 pages.

- We define a list of activities to be performed

- We discuss responsibilities to be handled for performing the migration: if the Client decides to move forward with the migration, who will perform each of the activities identified at the previous point? Strumenta, the Client, or a third party?

☐ Deploy the new system, when it is ready for production

☐ Deploy the new system continuously during development, for manual testing

☐ Manual testing (User Acceptance Tests) on the new system, during development

☐ Migration of data

☐ Verify the new system behaves exactly as the old one

☐ Verify the new system have certain performance

☐ Re-train developers

➤ FINAL CONSIDERATIONS

In this section we include final considerations. Based on our experience with previous migrations we provide comments on the challenges facing this project and the aspects to be kept under control during the execution of the migration.

This section is typically half a page long.

TAILORED SOLUTIONS

We study your infrastructure and your architecture to propose to you the best-customised solution: customise the framework or the readability of the code that best suit your needs.

ABOUT US

Strumenta offers Language Engineering services: we design new languages or provide tools to extract more values from existing languages.

We work on Language Engineering projects; typical Language Engineering projects include building parsers, interpreters, transpilers, compilers, editors, and designing and implementing Domain-Specific Languages (DSL).

We create language and tools that make organisations more productive. Our deployed instruments make work more efficient and less error-prone, reducing costs and making progress faster.

We help corporations make strategic choices related to Language Engineering, build smooth processes and be more productive.
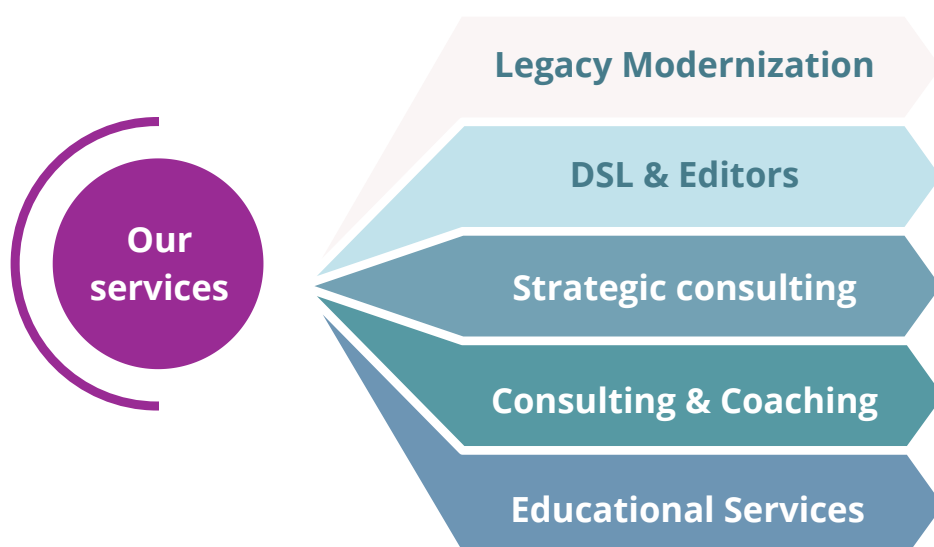
We are specialized in Language Engineering and fully dedicated to providing the most advanced solutions available in this field.

We have worked on Language Engineering projects with many medium and large companies (including Fortune 500) from Europe, the USA, Asia, and South Africa.

As Strumenta, we have written tutorials and articles that have been read millions of times and over 15,000 people subscribed to our newsletter on Language Engineering.

We have also created the Strumenta Community, which includes hundreds of members and many of our field's most well-known researchers and practitioners worldwide.

Hundreds of developers on every continent have bought our books and video courses on Language Engineering, and we regularly present our work at international conferences and universities.

Legacy Modernization

DSL & Editors

Our services

Strategic consulting

Consulting & Coaching

Educational Services

STRUMENTA

CAN YOU BETTER LEVERAGE THE COMPETENCIES YOU HAVE, AND THE PROCESSES YOU DEVELOPED BY USING MORE APPROPRIATE LANGUAGE AND TOOLS?

In this era the main asset most companies have is knowledge.
We produce knowledge, verify it, analyze it, elaborate it, put it into action. We do that by means of languages and tools.
The services we offer aim to put you in the position of getting the most value from the people you have.
Professionals use the best tools and the most important tools used by knowledge workers are languages.

Please get in touch for any further information or visit our website.

✉ info@strumenta.com

🌐 www.strumenta.com