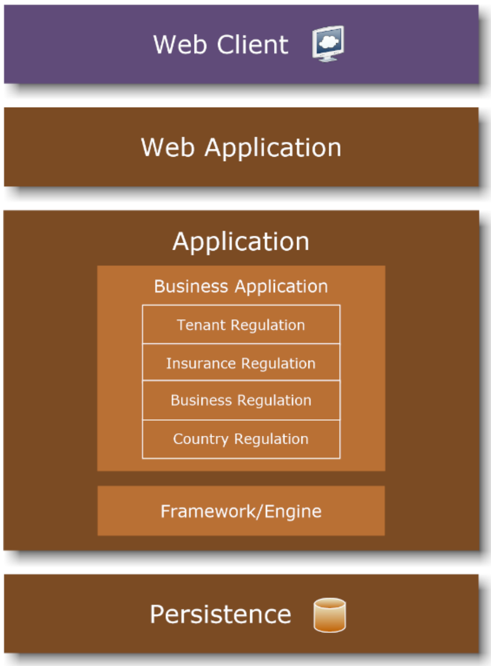
# Payroll Engine - Design of a scalable payroll software



Driven by bureaucratization, digitalization and constant technology changes, the design of modern software is a complex challenge. Based on practical experience, the scalability of business cases is particularly crucial in payroll software.

To reduce complexity, software systems are usually divided into different areas (layers or services). These are usually the frontend/client with the user interface and the backend/server, which consists of the application and the data persistence (database). From the perspective of value creation, the application is the software core because it implements the business processes. The user interface and persistence should ideally be generated from the business model.

If the application is built monolithically, every customization must be realized by highly qualified software engineers who do not always know the relevant business requirements.

When designing the *Payroll Engine*, we followed the concept of Domain-Specific Application Frameworks [1)], which provides for the separation of the application into a framework and the domain/business application. The framework contains the basic functions and is developed by software engineers, which is only affected by fundamental changes. The business application is developed by business specialists who are provided with professional tools for this purpose. A certain technical understanding is required from the business specialist.

This separation reduces the administrative, financial and communicative effort between the business and framework developers. The central design aspect is the correct distribution of knowledge, which enables the agile development of business applications and reduces the transfer of know-how between the parties, for example in the case of a change of personnel, to what is necessary.

In the *Payroll Engine*, the REST API represents the framework and the regulation a component of the business application. The regulation contains the business cases, the wage calculation and the evaluations/reports.

Another challenge in the design of a payroll software is the fact that the business functionality comes from different sources, such as Swissdec, LGAV, insurance, etc. the *Payroll Engine* integrates each business source as a separate regulation and merges them into a business application in a layered model. Analogous to the image layer in an image processing software, all regulation layers result in the overall image of the business application. Adjustments for the payroll client are integrated in the top regulation layer (image layer).

With the *Payroll Engine* ecosystem, regulations are exchanged between payroll service providers. Thus, payroll knowledge is continuously built up and shared, which means enormous potential for the scalability of the payroll software in the long term.

1) Mohamed E. Fayad & Ralph E. Johnson – Domain-Specific Application Frameworks
ISBN: 978-0-471-33280-0, Wiley