Integration of Payroll Engine

Success factors for integrating the Payroll Engine service

The *Payroll Engine* extends human resources applications with employee payroll services. The following article shows possible scenarios for integrating the Payroll service into existing software systems.

Prerequisites

The *Payroll Engine* provides the following basic services:

- Multi-client capability
- Task management with employee notifications
- Document management system (DMS)

In order to support the novel *Payroll Engine* model of business cases, the payroll front application must provide a dynamic input form.

Payroll Engine Services

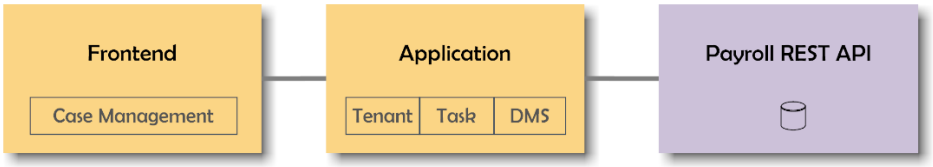
The *Payroll Engine* includes the following services:

- Multi-tenant OpenAPI REST interface
- Client services with Payroll Console and runtime libraries

The Payroll REST interface stores payroll data in a relational database (currently MS SQL Server), which is either run autonomously or integrated with the application database.

Basic connection

The simplest connection of the payroll application to the Payroll API is via REST (HTTP), where the functions are executed via Internet addresses (endpoints). In this constellation, the payroll application is responsible for adapting the payroll model. The case management in the frontend enables the administration of the business cases.



The basic connection is technology neutral and only requires HTTP communication.

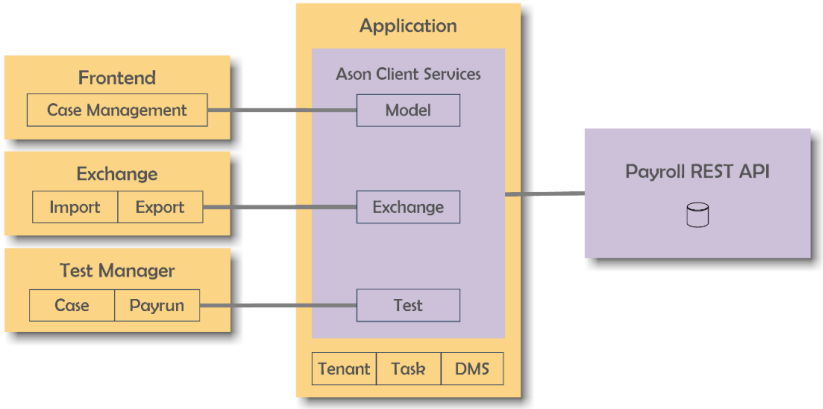
Payroll Console

The Payroll Console Client is a command line application for Windows/Linux/MacOS which is used to control the Payroll API. For data migrations and interface connections, payroll data can be imported and exported in validated JSON files.

The console enables automated testing of cases and payroll runs. It processes input data from JSON files and validates it against the expected output data.

Runtime library

For efficient integration of the Payroll API, the *Payroll Engine* Client Services include a runtime library (NuGet) that provides the available API objects as well as access to the Payroll API. The Client Services are dependent on the development technology and are currently available for the .NET Core Framework:

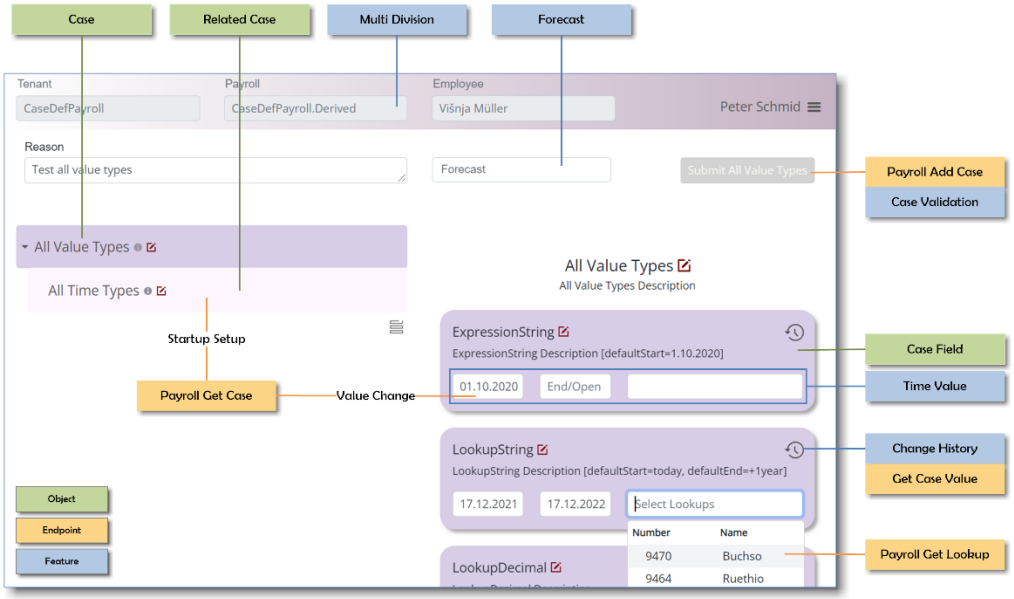


In addition to the payroll model, the runtime library contains components for the exchange of wage data (import and export), as well as for the integration of wage tests (business cases and payruns) in test applications.

From transaction data to business cases

In the *Payroll Engine*, all business and employee data (e.g. employee address) is managed in business cases. The model of business cases is dynamically determined by regulations (see blog Regulation of wage calculation). This requires that the front end supports dynamic input of a business case. Mapping between the dynamic model and a conventional/static model does not make sense.

Centralized input of business cases ensures consistent user guidance and eliminates the need to develop costly special forms. The following example shows a Blazor front-end entering a business case:



Payroll Report Integration

The Payroll Engine Report includes the preparation of the report data as well as the template documents for the conversion and verification of the report. In the report preparation, the data is determined by means of queries on API endpoints and provided as a DataSet.

The report is created by the payroll application:

1. determining the report data through the Payroll API
2. transforming the report data with the conversion document (serial letter, XSLT, RDLC, URL...)
3. optional validation of the report with the validation document
4. providing the report

Report templates of standard regulations (e.g. salary certificate), can be overridden in business or tenant regulations.

Services Integration

To integrate pending services such as legal wage reporting, the Payroll API provides a callback mechanism with Webhooks. For specific API events, a message is sent to a web address.

The following events serve as triggers for Webhooks:

- entry or cancellation of a business case
- change in the payrun status
- invocation in a client script (see blog Scaling of Payroll).

Conclusion

When integrating the *Payroll Engine*, both technical and application-specific criteria must be considered. The technical dimension determines the type of connectivity as well as the availability of additional runtime components. The payroll application must be able to integrate the dynamic model of the *Payroll Engine* business case and provide it in the frontend.