Integration of the Payroll Engine

Success factors for embedding the Payroll Engine

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

Payroll Engine provides the following basic services for payroll applications:

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

To support the new Payroll Engine case model, the front end of the payroll application must provide a dynamic input form.

Payroll Engine Services

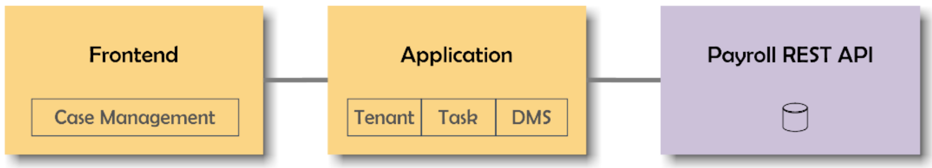
The Payroll Engine includes the following services:

- - 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) that is either operated independently 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 customizing the payroll model. The case management in the front end enables the administration of the business cases.



The basic connection is technology neutral and requires only HTTP communication

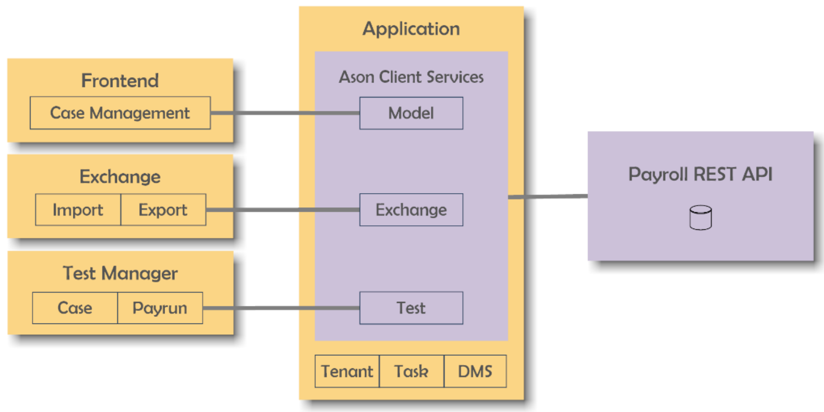
Payroll Console

The payroll command line application for Windows and Linux is used to control the Payroll API. Payroll data can be imported and exported in validated JSON files for data migrations and interface connections.

The console enables automated testing of cases and payroll runs. Input data from JSON files is processed and compared with 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 and access to the Payroll API. The Client Services are dependent on the development technology and are currently available for the .NET Core Framework:

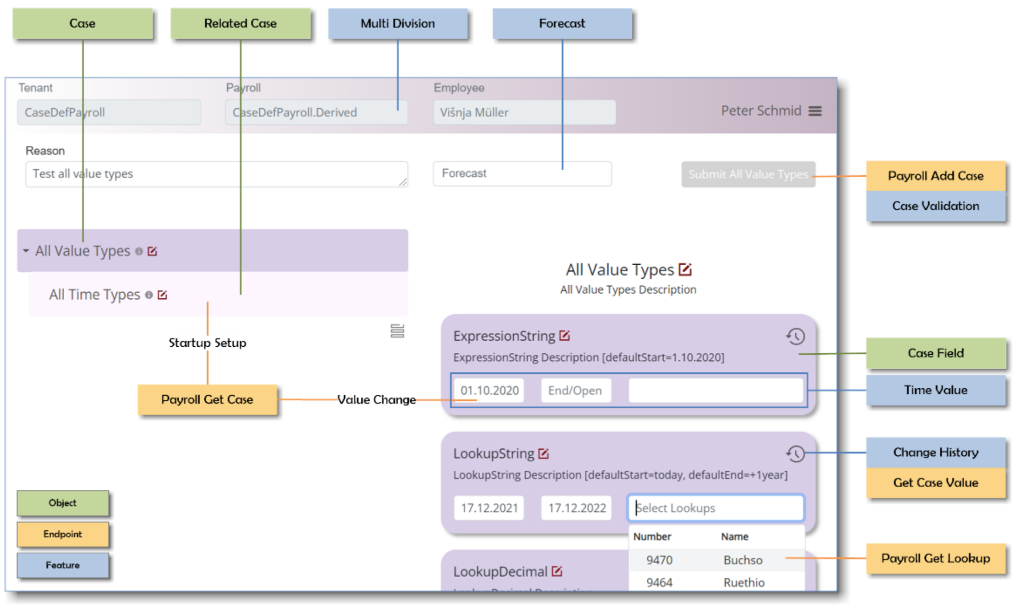


In addition to the payroll model, the runtime library contains components for the exchange of payroll data (import and export) and for the integration of payroll tests (business cases and payroll runs) in test applications.

From transaction data to business cases

In Payroll Engine all business and employee data (e.g. employee address) is managed in business cases. The model of business cases is determined dynamically by rules (see Payroll rules blog). This requires that the frontend supports the dynamic capture of a business case. Mapping between the dynamic model and a conventional/static model is not useful.

Centralized capture of business cases ensures consistent user experience and avoids the need to develop costly special forms. The following example shows a Blazor frontend entering a business case:



Tenant: CaseDefPayroll
Payroll: CaseDefPayroll.Derived
Employee: Višnja Müller
Peter Schmid
Reason: Test all value types
Forecast

All Value Types
All Time Types

All Value Types
All Value Types Description

ExpressionString
ExpressionString Description [defaultStart=1.10.2020]
01.10.2020 | End/Open

LookupString
LookupString Description [defaultStart=today, defaultEnd=+1year]
17.12.2021 | 17.12.2022 | Select Lookups

Number	Name
9470	Buchso
9464	Ruethio

LookupDecimal

Payroll Report Integration

The Payroll Engine report includes the preparation of the report data as well as the template documents for implementing and checking the report. In the report preparation, the data is determined by queries to API endpoints and made available as a DataSet.

The report is created by the payroll application:

1. determination of the report data by the Payroll API.
2. transformation of the report data with the transformation document (serial letter, XSLT, RDLC, URL...)
3. optional validation of the report with the validation document (XSD...)
4. provision of the report

Report templates of standard regulations (e.g., wage statement) can be overridden in industry or client regulations.

Services Integration

For the integration of dependent services, such as statutory wage reporting, the Payroll API provides a callback mechanism with Webhooks. When certain API events occur, a message is sent to a web address.

The following events serve as triggers for WebHooks:

- Entry or cancellation of a business transaction
- Status change of a billing run
- Call in a client script (see The Scaling of Payroll blog).

Conclusion

When integrating the Payroll Engine, both technical and application-specific criteria must be considered. The technical dimension determines the type of connection 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 cases and make them available at the front end.