BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2026
http://excel.fit.vutbr.cz

Information System for Logistics Process Management in a 4PL Model

Tomáš Bordák*

Abstract
The aim of this work is to design, implement, and deploy an information system that centralizes logistics data and automates key operational processes. It addresses the problem of inefficient and fragmented management of logistics processes in companies operating in the 4PL (Fourth Party Logistics) model.
The solution is implemented as a web-based information system using a 3-tier architecture that utilizes modern web technologies and containerization to ensure portability and ease of deployment. The system offers a user interface and a REST API, allowing customers to integrate their existing systems and support automation of their processes. Key functionalities include management of shipments, customers, and carriers, as well as automated processing of orders, transport documents, and periodic tariff-based customer billing.
The implemented system was deployed in a real company production environment, where it reduced manual workload and improved access to logistics information. Automation of core processes led to shorter processing times for orders and invoices. The system also enabled customers to track shipments and access relevant data in real time. The proposed approach is applicable to all logistics models and can be used as-is or serve as a basis for further development.

*xborda01@vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Introduction

[Motivation] Companies operating in the 4PL (Fourth Party Logistics) model coordinate multiple independent actors, including customers, carriers, and warehouses, without owning transport fleet or warehouse themselves. In practice, these interactions are often handled using emails, spreadsheets, and partially integrated systems, which leads to inefficient workflows and limited visibility of logistics processes. A key issue is the lack of direct access to operational data for customers, increasing communication overhead and reducing transparency.

[Problem definition] The core problem is the absence of a centralized system that integrates logistics data and automates key processes such as a transport order processing, document handling, and customer billing. Current workflows rely heavily on manual communication and data processing, which is not scalable. A suitable solution should centralize data, enable integration with external systems, support automation, and provide accessible real-time information for both operators and customers.

[Existing solutions] Enterprise systems such as Oracle Transportation Management, SAP Logistics, and Infor LX offer comprehensive functionality and scalability for large organizations. However, they typically require significant investment, complex customization, and long implementation cycles, making them less suitable for small to medium-sized 4PL providers. As a result, such companies often rely on simpler or manual tools, which lack automation and integration capabilities.

[Our solution] This work proposes a web-based information system that centralizes logistics data and supports key processes including shipment management, document processing, and billing. The system provides both a user interface and a REST API, enabling direct user interaction as well as integration with external systems and customer automation. It is designed to support gradual adoption and integration into existing workflows without disrupting ongoing operations.

[Contributions] The main contribution is the design, implementation, and deployment of a functional system in a real-world production environment. The solution demonstrates how automation and integration

can reduce manual workload, improve data availability, and increase process efficiency. It also provides a scalable and extensible foundation applicable to various logistics scenarios.

2. Transport Order Processing

This module focuses on automating the processing of transport orders, replacing a previously manual workflow. Orders were originally received via email or FTP, requiring manual checking, downloading, completion of shipment identification data, and import into an ERP system, where data validity could only be verified after processing. This approach was time-consuming and error-prone.

The system introduces two input methods: UI upload (XLSX) and a REST API (JSON) for integration. Both share a unified validation layer that checks data before processing and returns structured error messages with precise field-level information.

As a result, the shipments creation from orders is significantly streamlined, reducing manual intervention on the provider side and enabling customers to automate order submission according to their technical capabilities.

3. Customer Billing

The implemented system automates billing by integrating tariff management directly into the application. Tariffs are defined using postal code zones with corresponding rates and applied automatically to completed shipments. The tariff format itself was not simplified and remains based on structured XLSX files, that can be downloaded, edited, and re-uploaded, or used as a base for new tariffs. This replaces the previous old desktop application with legacy and error-prone user interface.

Billing is generated based on configurable customer-specific billing periods without the need for intermediate files. This approach centralizes billing, improves usability of tariff management, reduces manual work, and increases data consistency across the system.

4. Transport Documents Processing

Transport document handling was previously manual and fragmented. Documents such as Bills of Lading were collected via email or FTP, manually downloaded, and assigned to shipments in the ERP system. Customers did not have direct access and requested documents via email or messaging.

The system automates processing of two types of Bills of Lading. Documents from carriers and key customers are automatically retrieved, stored, and linked to shipments. Documents can now be centrally managed and immediately available within the system, improving traceability and reducing manual processing effort.

5. Architecture and Technology Stack

The system is implemented using a three-tier architecture consisting of a presentation, application, and data layer. The frontend is built using Vue.js 3 with TypeScript, Pinia for state management, Axios for API communication, and Tailwind CSS combined with PrimeVue for UI components. It is served through an Nginx container, which acts as a reverse proxy to the backend and is integrated with a Certbot container for automated TLS certificate management, enabling the system to communicate securely with a web browser (client) via HTTPS.

The backend is implemented as a Node.js application using Express.js and TypeScript, exposing a REST API. It uses Prisma ORM for database access and migrations, Zod for validation, Multer for file handling, and Swagger for API documentation. Background processing is handled by BullMQ with Redis as the storage engine that powers the job queues. The data layer is represented by a PostgreSQL database, deployed as a separate container.

The entire system is containerized using Docker and orchestrated via Docker Compose, enabling consistent deployment across environments.

6. User Interface and Features

The user interface is implemented as a modern web application with a clean design based on a blue primary color scheme and white background. The system uses role-based access control (RBAC), allowing different views and permissions for customer and provider users. The frontend provides separate functional areas depending on user roles while maintaining a unified interface.

Customers can access shipment information, transport documents, a dashboard with key performance indicators, and submit transport orders via both the UI and REST API.

Provider users can manage shipments, customers, carriers, tariffs, billing settings, API keys, users, and permissions. The system supports automated processes such as tariff validity updates, document assignment for Bills of Lading, and order processing. It also provides a simplified order format and is designed for extensibility and integration into larger logistics environments through its API-first approach.