Project Report

in the module

Architecture & Integration

Mr. Sebastian Speck

Project Group 03

Nour Batniji 1507535

Saida Covrk 1509852

Mariam Zadran 1518535

Introduction

In the digital transformation of universities, the seamless integration of administrative systems is critical for reducing manual work, minimizing errors and increasing process efficiency. At the Frankfurt University of Applied Sciences, student data is centrally managed in the HIS System, which stores information such as student names, IDs, enrolled study programs and earned credits per program. This data is essential for other subsystems such as Peregos, which is used by the Examination Office and WyseFlow, which is used for managing final thesis submissions.

Currently, data entry is performed manually in each of these systems, which leads to increased workload, inconsistencies and errors. In response to this inefficiency, a middleware-based integration solution has been proposed by the head of the Examination Office. This solution uses RabbitMQ as the core middleware, enabling the automated and reliable distribution of student data from HIS to both Peregos and WyseFlow.

This report presents a detailed architectural model of the proposed solution, with a comprehensive explanation of the systems, their connections and the business logic that underlines the entire integration.

System overview and core components

The architecture is composed of the following components:

- HIS – The primary data source for student records.
- Peregos - A subsystem used by the Examination Office to process student requests.
- WyseFlow - A subsystem that handles final thesis applications.
- RabbitMQ – The integration middleware acting as aa message dispatcher.
- Business Processes and Actors: The workflow and stakeholders involved in the data flow.

HIS

At the core of the system lies HIS, which is the central system responsible for managing student and examination-related data. HIS acts as the producer of data for other systems.

In the following, a brief breakdown of the processes related to HIS are described.

- Enables students to register for and withdraw from examinations.
- View information about their registered or deregistered exams.
- Track their accumulated credit points.

All these actions are processed through the “examination management”- function within HIS. Once student related is updated, HIS publishes this information for further administrative use.

The HIS system operates on a dedicated server, which is compatible with Windows, Linux and macOS ensuring flexibility and broad accessibility. All data is stored securely within this environment.

Peregos

The second university system “Peregos” provides the examination office with some help: It supports the work of the examination council. In order for it to be able to do that, it collects data about the university’s students:

- Student name
- Student ID
- Names of enrolled study programs

This ensures that students can create and send requests to the examination council and have them handled or processed in a way that fits their study program. To realize that, Peregos makes use of multiple different functions as well as services, which are described below:

Functions:

(All functions are in an assignment relation to the Consumer Peregos (relation in the model))

- Receive student data: This function is essential for the connection of Peregos to the Middleware RabbitMQ, as it is directly linked to its distribution/routing function.
- Student data management: Direct management of the received student data happens here; the students’ data is grouped and saved.
- Request management: The incoming requests that students make are managed here.

Services:

(All services are in a realizing relation to their respective management function (mentioned in respective descriptions))

- Process student data: This service is directly realized by the student data management and processes the data that enters the system by students submitting their unique or generic requests.

- Creating request & processing request: Both of these services are related to the function for request management. They make it possible for the examination council to handle the requests that they are given.

Peregos is compatible with different Operating Systems, such as Windows, Linux or macOS.

The system is also backed by a database, which it stores all its information in and has direct access to when operating students' applications.

WyseFlow

The university system "WyseFlow" essentially handles the applications for the students' final thesis. To do this, it needs information on the student as well as the credits of the respective study program.

In the architectural model, WyseFlow is, as both HIS and Peregos, a consumer which is modeled as an Application Component in the Application Layer of this model. It includes various functions and services, which are as follows:

Functions:

(All functions are in an assignment relation to the Consumer WyseFlow (relation in the model))

- Receive student data: This function is very vital for this program, as it embodies the connection point to the middleware RabbitMQ.

- Student data management: Here, all student data regarding their personal amount of credit points, specific regulations or restrictions for exams, etc., is handled and checked to see if the requirements for the application are met from the students' side.

- Thesis management: In this function, the topics for each thesis as well as responsible teachers are handled for each thesis application.

Services:

(All services are in a realizing relation to their respective management function (mentioned in respective descriptions))

- Process student data: The processing of the student data comes directly with the function for receiving the data and ensures that the application is possible for a student /is opened up for a student.

- Thesis application: This service makes it possible for students during to submit their application with all their information during the process.

As the other two systems, WyseFlow is compatible with different Operating Systems, such as Windows, Linux or macOS.

The system is also backed by a database, which it stores all its information in and has direct access to when operating students' applications.

RabbitMQ

In the architecture, RabbitMQ serves as the central message broker for student data, specifically mediating the communication between HIS, Peregos and WyseFlow.

From HIS: HIS uses the application service "publish student data" to send student data to RabbitMQ.

To Peregos and WyseFlow: Both Peregos and WyseFlow are Consumers and have the application service "receive student data". RabbitMQ routes the messages received from HIS to both consumers.

RabbitMQ is deployed in this architecture as a robust message broker that has the primary function to facilitate reliable communication of student data from HIS to Peregos and WyseFlow. With that the overall systems resilience and scalability is enhanced by enabling independent scaling of components.

Business Layer

The Business Layer contains the two main actors and a diversity of Business Functions. The first actor, being the student, plays a big role in this model. Students can access the following functions:

- Create request: Here, students can submit their requests to the Examination council.
- Apply for thesis: With this function, students’ applications are brought to the Examination council.

The second and just as important actor is the Examination Office, as it is responsible for a lot of processes inside the university life. It works with the functions listed below:

- Management of deregistered and registered examination
- Process of the visualization of written examinations
- Processing student data
- Request process
- Processing student data

Conclusion

The developed architecture effectively portrays a modern, robust and scalable management system. It addresses the challenges previously caused by manual data entry between HIS, Peregos and WyseFlow. By introducing RabbitMQ as the central messaging middleware, student data can now be transmitted automatically, reliably and fast. Each system processes the received data independently based on its specific requirements, which ensures a clear separation of concerns and improves maintainability.

The modular design allows future extensions such as the integration of additional systems. Overall, the project demonstrates how middleware-based integration can significantly improve data consistency, reduce administrative workload and create a scalable foundation for further digitalization at the university.