SAI Final Assignment:

Travel Costs Refund System for Internship Visits

Table of Contents

[1. Introduction 2](#_Toc515874643)

[2. Start-up Code 4](#_Toc515874644)

[3. Assignment 4](#_Toc515874645)

[4. Grading Criteria 5](#_Toc515874646)

[5. Submission and Deadlines 5](#_Toc515874647)

# 1. Introduction

In this assignment you will integrate a system of several applications for requesting of costs foundation of Fontys ICT teachers who visited internship students in their companies. The whole scenario for the Travel Costs Refund system is shown in Figure 1.

**Travel Client application** is a Java desktop application. It is used by Fontys teachers for requesting the refund of travelling costs. Teachers must provide the origin address, destination address, teacher name, student name and travel mode (by car or by public transport). There are two traveling mode types:

* If the mode is car, then the teacher does not provide costs (the system will automatically calculate the costs later). This type of request is illustrated in the blue rectangle in Figure 1.
* If the mode is public transport, the teacher must provide the costs. This type of request is illustrated in the red rectangle in Figure 1. In addition, the teacher is responsible for scanning all bus and train tickets and emailing them to the Financial Department (but this has nothing to do with this integration system).

**Travel Broker application** is a Java desktop application which receives requests from Travel Client application, processes them and sends back the reply (costs are approved or rejected) to the Travel Client Application. Once the Travel Broker receives a new request from the Travel Client, the *Content-Based Router*  inspects the travel model of client request and:

* If the mode is public\_transport (red rectangle in Figure 1), the approval request is created with the costs from the client request.
* If the mode is car (blue rectangle in Figure 1), then the Content-Enricher calculates the costs:
  + Call the google distance service to get the distance in kilometres between the origin and destination,
  + Call the PricePerKilometer service to get the current refund price per kilometer,
  + Calculate the costs=distanceInKm \* pricePerKm.
  + Create the approval request (with calculated costs).

Travel Broker application then sends the approval request via a Recipient List to the two approval applications by these rules:

|  |  |
| --- | --- |
| approval application | processes approval requests |
| Internship Administration | all requests |
| Financial Department | costs >= 50 euro |

After the Travel Broker receives all approval replies, it creates a TravelRefundReply and sends it back to the Travel Client as follows:

* If all approval applications approved the request, then the field reasonRejected is left empty (red rectangle in Figure 1).
* If one or more approval applications rejected the request, then the field reasonRejected contains the names of the applications which rejected (blue rectangle in Figure 1).

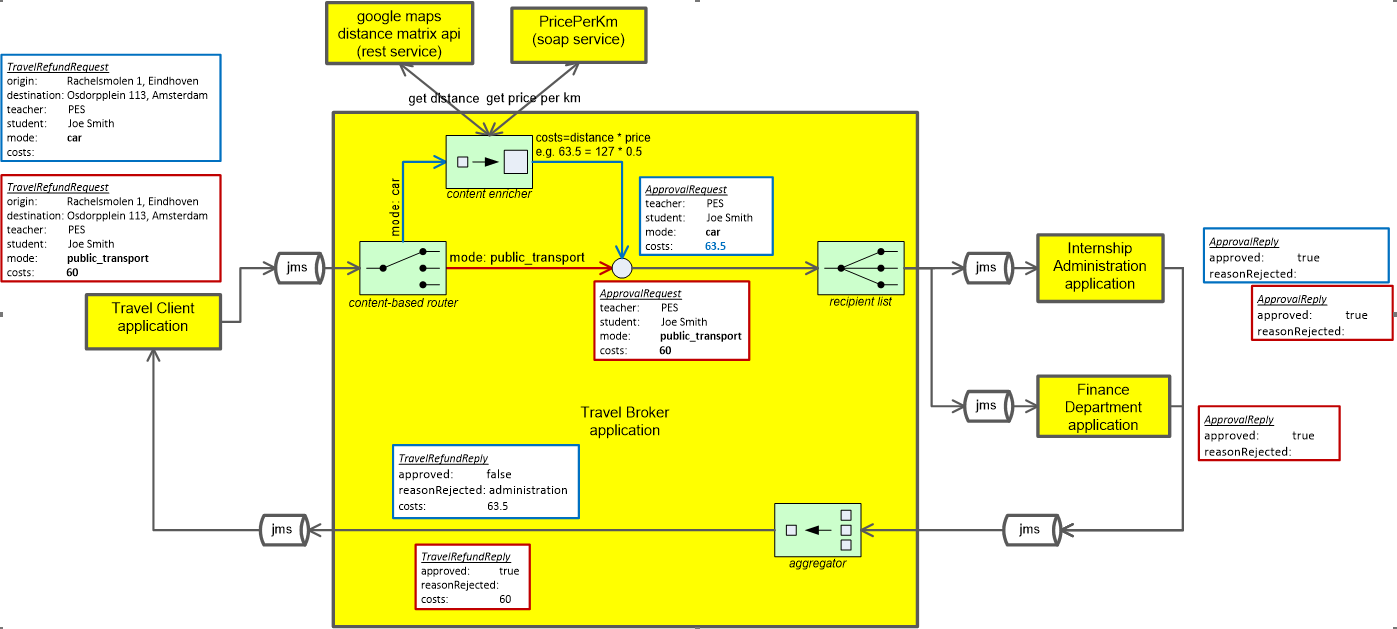


Figure 1. The Travel Costs Refund integration system

# 2. Start-up Code

Download **SAI\_final\_startup\_code.zip** from the SharePoint:

* **“sai-travel-refund”**: IntelliJ IDEA project with client and approval modules/applications.
* **“PricePerKmService”**: is a war file of a RESTfull service which you should deploy on a web Server (e.g., Apache Tomcat). There are two WAR files: one is built with Java 8 and other with Java 9. The service can be accessed at <http://localhost:8080/priceperkm/rest/price>. It has only one method which reads the price (as one double number) from a text file (see your\_tomcat\_installation\webapps\priceperkm\WEB-INF\classes\price.txt).

# 3. Assignment

This assignment is INDIVIDUAL, i.e., it is not allowed to work in groups with other students. The mark you get for this assignment is between 1 and 10, and this will be your mark for the Software Applications Integration (SAI) course.

Implement the Traveling costs Refund integration system as described in this document. You should make use of the following integration patterns:

* Message Broker
* Correlation Identifier (for asynchronous request-reply communication with JMS),
* Return Address (for asynchronous request-reply communication with JMS)
* Messaging Gateway
* Chained Gateways
* Content-Based Router,
* Content Enricher,
* Recipient List,
* Aggregator, and
* Scatter-Gather.

# 4. Grading Criteria

IntelliJ project(s) including full source code and all necessary libraries (gradle, maven or .jar) must be submitted. All submitted projects must compile and run correctly on the computer of the teacher. If the teacher does not have your full source code, or cannot run your project(s) due to compiling errors, missing libraries or exceptions, then your SAI mark will be 1. Otherwise, SAI grades will be determined based on implemented Application Integration Patterns in the following way:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | SAI grades | | | | |
|  | 6 | 7 | 8 | 9 | 10 |
| The system works correctly with one approval application | x | x | x | x | x |
| Message Broker | x | x | x | x | x |
| Correlation Identifier | x | x | x | x | x |
| Return Address | x | x | x | x | x |
| Messaging Gateway | x | x | x | x | x |
| Chained Gateways | x | x | x | x | x |
| Content-Based Router |  | x | x | x | x |
| Content Enricher |  | x | x | x | x |
| The system works correctly with two approval applications |  |  | x | x | x |
| Recipient List |  |  | x | x | x |
| Aggregator |  |  | x | x | x |
| Scatter-Gather |  |  | x | x | x |
| Use of Jeval (or similar) instead of  hard-coded travel approval rules. |  |  |  | x | x |
| Well organized code, with comments, proper variable and method names, no redundant code. |  |  |  |  | x |

# 5. Submission and Deadlines

**Submission of the source code**

The IntelliJ project(s) with full source code and all necessary libraries (gradle, maven or .jar) must be submitted via Canvas. The deadline for submission is Thursday 28 June 2018 at 09:00 am. It is not possible to submit after this deadline. If you do not submit your source code before the deadline, you will not receive a SAI grade in this block.

**Defense of your assignment**

On Friday 29 June 2018 SAI “exam” is scheduled (see class schedules). During this defense you will speak in person to the teacher about your assignment: you will be asked to explain your code, suggest ideas for improvement, etc. If you are not present during this defense, then you will not get a mark for SAI. It is not possible to defend your assignment at another time.

Only students who submitted their source code via Canvas before the deadline specified in the paragraph above will be invited for this defense. You will receive this invitation with your specific time slot from you teacher on 28 June 2018. In this invitation it will be specified at which time you should be present for defense on 29 June 2018. Each student will have his/her own time slot, and you should and can be present only during your own time slot.