The Shipping Company

Będkowski Patryk, Jaremek Łukasz, Skarzyński Szymon

This project was created as part of the Object-Oriented Programming class at Warsaw University of technology
May, 2021

I. INTRODUCTION

This article describes the use, functionality, creation and architecture of a software. Presented program simulates shipping company which performs operations on shipping centers (hubs) and its shipments (packages). It provides the guidance for users that are unfamiliar with the topic. We will also address the topic of software installation, its most important elements and a description of how to test the software.

II. INSTRUCTIONS ON HOW TO USE THE SOFTWARE

To use the software, user has to compile the program and run it. After that, the console window with the interface will open and the software will be ready to use. The main menu will be first thing that user will see. Menu options can be selected by entering corresponding numbers in the terminal (every input should be confirmed by ***Enter*** key). In case of entry menu number that differ from ones specified, software display adequate message to try again. User can load session data (hubs and packages list) from a txt file or save it to one.

III. MOST IMPORTANT ELEMENTS OF THE SOFTWARE

Software is divided into four main components:

A. App class

The App class is built on the singletone pattern, meaning that only one object of this class can be created during the runtime of the application. This class acts as a bridge between interface and the rest of software logic. Every user interaction through interface is send to the app class, where it calls for adequate methods that send back information to the interface. This class also stores data of the current session, that is: created hubs.

B. Hub class

This class represents a shipping center (hub) that stores packages. Each hub is identified by its name. User can transfer packages between hubs and release them (simulating package delivery to the recipient). The class includes vector with Packages that can be increased by a new package using the += operator.

C. Package class

The Package Class includes general information about a parcel such as sender or recipient. However, the inheriting classes contain more specific information as size or priority. The price of the package depends on it: size, priority and fragility. Each package supports the following operators: comparison operator, display operator. Every package is created dynamically on the stack using *new* operator and on the memory that is assigned to it is deallocated from the memory block using *delete* operator. In addition class Package implements virtual functions that change according to package category.

D. Interface class

This class represents a terminal interface that allows user to use the software. It communicates with the App class.

E. readHubs and writeHubs functions

Functions are used to reading and writing .txt files with Hubs and Packages data. Every line represents Hub or the Package in the Hub.

IV. SOFTWARE TESTING

The following paragraph describe how the software was tested. Tests are implemented using Microsoft Unit Test Framework. The name of the file containing software assessment is *ShippingCompanyTests.cpp.* The structure of this file is divided into four sections:

- App class tests
- Hub class tests
- Package class tests

All sections contain comprehensive list of test cases which should be analyzed, its purpose is to find bugs in later development of the program. Each section act as a TestFixture class that contains number of test methods regarding this class. These methods test small units of the application. Each method is described with multiple comments.

V. TEAM RESPONSIBILITIES

We divided into teams of one to develop each part of the program separately.

Patryk took a role in developing structure of the app class, which connects every part of the software together. Thanks to his contribution in testing the application, code imperfections could be detected. Patryk also provided support for the other members of a team, in terms of how the given software must behave.

Łukasz was responsible for creating the Package class, upgrading Hub class In addition, he made reading and writing database files which contains information about Hubs and Packages. He also added part of comments in some files and tested the software in the process.

Szymon was responsible for creating the interface, extending App class functionalities, creating the initial version of the Hub class and documenting the code.

In later parts of the development, each member of a team contributed to others' code selflessly.