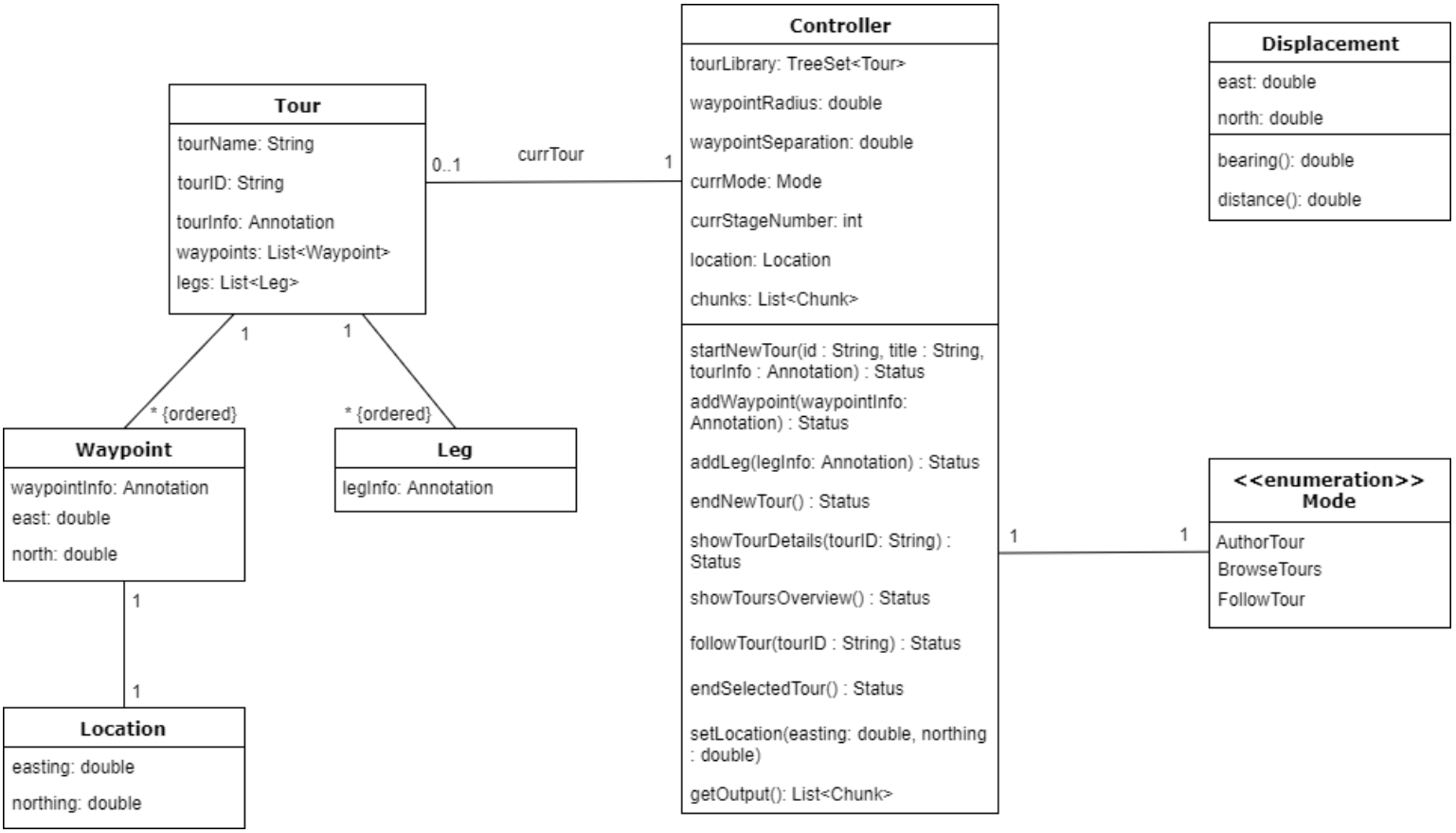
Introduction to Software Engineering

Coursework 3

Nikoleta Kuneva s1643102
Ivan Karaslavov s1632798

November 2017

1.UML class model



2.High-level description

1. Description and implementation decisions

The ControllerImp class serves as the driving force behind our application. It implements the Controller interface. Its function is to take in messages coming from the outside world and pass them on to the internal components of the system. It contains methods related to the three main modes of the app – BROWSE, FOLLOW, AUTHOR, and thus enables the user to choose a tour from a tour library, to start or create a tour and to repeatedly add waypoints and legs. The ControllerImp class also automatically updates the current leg and waypoint and the tourist's location by receiving set location messages. These messages carry as arguments information about the north and east coordinates of the current location. For each tour in the list only its ID and name are displayed. After picking a tour, additional information about the selected tour appears. This information is held in the Tour class and includes the annotation for the chosen tour. Apart from the Tour, the Leg and Waypoint classes also contain Annotation type. Every tour starts with a leg and ends with a waypoint. There is a leg between every pair of consecutive waypoints and if no leg is added, the system adds a default annotation. Using the UML property text "{ordered}" we made sure that the waypoints and legs are kept in the same order as the destinations the tourist wants to visit.

We added several additional classes – Leg, Waypoint, Location, Tour and the enumeration class Mode, to maintain a good object-oriented design. In the previous coursework we assumed that the Leg and Waypoint classes inherit Chunk. However, we realized that this implementation would not be efficient as only the information about the leg and waypoint is presented as a chunk. In the second coursework we had two additional classes – CityLibrary and City. We decided not to include them to keep the design simple. Instead we created a TreeSet tourLibrary in which all the tours are stored (we chose TreeSet in order to make use of the comparison function). In the Tour class we decided to use the Collection class List for the waypoints and legs so as to keep them in the right order.