**Function points**

We will now evaluate the parameters that are used to calculate the Function Points.

**Internal logical files**

MTS has to store internal information about:

* Taxi
* Users
* Rides
* City zones
* Taxi queues

These can all be considered simple structures, except the Rides, which are composed by many crucial parameters, and the city zones, which are more complex due to the fact that they represent a geographical locations that can be interconnected. These two structure can be considered to be of a medium complexity.

Thus,

**External interface files**

The only data provided by the external interfaces is:

* Taxi location

The taxi location contains only information about the taxi id and its GPS position, thus it can be considered as a simple structure.

**External inputs**

The following are the functionalities offered by the MTS application which can be considered as inputs.

Simple complexity operations:

* Login
* Logout
* Change the taxi driver’s availability (driver side)
* Accept a ride
* Refuse a ride

Medium complexity operations:

* Registration
* Request a ride
* Create a driver’s account (admin side)
* Delete a ride
* Delete a driver’s account (admin side)
* Change the status of a ride (admin side)

High complexity operations:

* Reserve a ride

Where we have considered as *simple* all the operations that involves basic functionalities and only little data processing of the *Internal Logical Files.* Medium complexity operations are considered to be involving a higher data processing and management, especially regarding the creation or deletion of data. “Reserve a ride” has been considered as the only highly complex operation because it’s a more advanced version of the “Request a ride” functionality.

**External Outputs**

* Emails
* Mobile app notifications
* Web notifications
* Ride requests to taxi drivers

Here we consider all the operations as simple, except for the “Ride request to taxi drivers” because it contains data about an Internal Logical File: the ride.

**External inquiries**

* Visualize the users’ account information
* Visualize the details of a single ride
* Visualize the history of rides of a user

We consider the first two operations as simple, while the “Visualize the history of rides of a user” is a medium complexity operation because it involves the retrieval of multiple data.

**Total number of function points**

The un-adjusted function points (UFP) results:

This value can be further adjusted by applying a final correction which depends by other parameters that can be extracted from the project’s design phase. However, we won’t perform such correction because it usually doesn’t improve the precision of the estimation (in some cases it may even get it worse). Instead, we will use the UFP estimation in combination with COCOMO in order to estimate the project effort.