



POLITECNICO

MILANO 1863

Software Engineering II

CLup - Customers Line-up

Design Document

Authors:

Cosimo Sguanci

Roberto Spatafora

Andrea Vergani

PAGINE DI COSIMO

PAGINE DI ANDREA

2.F SELECTED ARCHITECTURAL STYLES AND PATTERNS

Three-tier architecture

The client-server architecture follows a three-tier pattern: in particular, we can distinguish a Presentation tier, a Logic/Business tier and a Data tier.

The division into three layers allows to separate tasks and increases the level of reusability and decoupling; in addition, the whole architecture results to be more flexible and maintainable (a single tier can be internally modified, fixed, ... without consequences on other tiers).

More specifically, the tier division is the following:

- Presentation tier is client application's business (mobile application, call center web application); it allows the interaction between user and *CLup*
- Logic/Business tier is present both on the client and server sides: servers' logic layers control application functionalities; client logic is related to tracking mobile app user position, in order to send specific notifications *
- Data tier includes the database and all mechanisms for storing data

** client logic is better described in section named "Thick client" (which immediately follows)*

Thick client

With reference to the mobile application, a thick client configuration is adopted: this implies that, in addition to the presentation layer, the mobile app client also incorporates a part of the business layer.

More in detail, *CLup* specification highlights the need to calculate the user's distance to the grocery shop, in order to send proper notifications for arriving on time and not missing the reservation; furthermore, the application's RASD describes how the customer can insert, in optional fields, his starting position and transportation means, so that the system can more easily deal with its computations (server-side, in general). However, the user may not add this additional information: to solve the situation, the decision is that of making the client itself responsible for sending proper notifications.

The main cause behind this design decision is that server, not having an estimate about when to start checking user's position (because the optional fields were not filled), would have to continuously track him (or something similar, at least for a time period): however, this is costly and would require special permissions for privacy reasons. For this reason, the logic regarding position monitoring for notifications (in case no additional information is provided by the customer) is left to the client, thus leading to a thick client design.

PAGINE DI ROBERTO