Project Development and Design Tips

Lesson 3

Luca Arrotta and Riccardo Presotto

EWLab – Università degli studi di Milano

Professor: Claudio Bettini

These slides are based on previous versions created by Letizia Berolaja, Sergio Mascetti, Dario Freni, Claudio Bettini and Gabriele Civitarese

Copyright

Some slides for this course are partly adapted from the ones distributed by the publisher of the reference book for this course (Distributed Systems: Principles and Paradigms, A. S. Tanenbaum, M. Van Steen, Prentice Hall, 2007).

Every Ware Lab

Disclaimer

- The content of these slides has to be considered indicative
  - Development and design suggestions will be provided.
  - Each student can make different choices

- The directives on how to carry out the project are reported in the text of the project on the course website.

Every Ware Lab

Recommended Development flow I

- First step (first lesson): REST server and SETA development
  - Design of the *Administrator Server* (resources and methods)
  - Synchronization problems analysis
  - Testing of the REST server with dedicated tools
  - *Administrator Client* Development
  - Implementation of SETA and the MQTT protocol to publish and receive orders

- Second step (second lesson): Development of the taxis' network
  - Architecture and protocols design of the peer-to-peer network of taxis
  - Insertion of a taxi in the peer-to-peer network
  - Rides management via a distributed and decentralized algorithm
  - Removal of a drone from the peer-to-peer network

Recommended Development flow II

- Third Step (Today's lesson): **Sensor data collection and local statistics**
  - Implementation of the sensors data collection.
  - Computation and communication of the local statistics

- It is **crucial** to carefully consider both the internal synchronization and distributed synchronization problems

Every Ware Lab

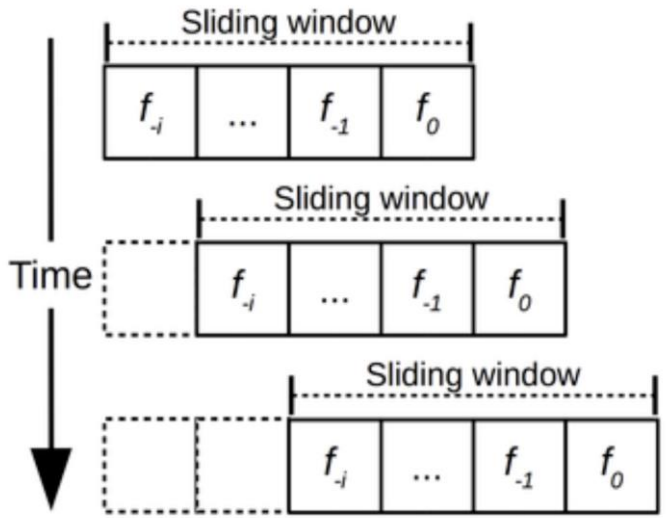
PM10 Sensors

- Each measurement is characterized by:
  - PM10 value
  - Timestamp (in milliseconds)
- Each sensor puts the measurements in a buffer
  - The buffer is used to periodically compute the local statistics about the pollution level
- Do we have to consider synchronization problems?

Every Ware Lab

Sliding Window (1)

- A standard technique to analyze or compress real-time sensors' streams
- Consist of computing statistics considering temporal windows of length *w*
  - Every time that *w* measurements are collected, we compute the statistics
- We can use a certain percentage of overlap *o* to capture the transitions between temporal windows
  - The temporal window w_i considers a certain percentage *o* of the most recent measurements of the window w_{i-1}



Every Ware Lab

Sliding Window (2)

- In the project, we consider a sliding window of 8 measurements with an overlap factor of 50%
- When the sliding window contains 8 measurements, you must compute and store the average of such measurements
  - Then you must discard the oldest 4 measurements of the sliding window to make room for new measurements
- Every 15 seconds, the averages computed in this way will be sent to the Administrator Server with the other local statistics

Local Statistics

- Every 15 seconds, each Taxi computes the following statistics (occurred within this interval of 15 seconds):
  - The number of kilometers traveled to accomplish the rides of the taxi
  - The number of rides accomplished by the taxi
  - The list of the averages of the pollution levels measurements
- This local statistics must be sent to the Administrator Server associated with
  - The ID of the Taxi
  - The timestamp in which the local statistics were computed
  - The current battery level of the Taxi

Every Ware Lab

Administrator Client's Requests (1)

- Considering only the local statistics about the air pollution levels' measurements, a Client can ask to the Administrator Server
  1. The average of the last *n* local statistics of a given Taxi
  2. The average of the local statistics from all the taxis occurred between timestamps T1 and T2

Every Ware Lab

Administrator Client's Requests (2)

- In the first case you must
  - Take the last *n* local statistics of the taxi
  - Compute the average of each list of measurements' averages
  - Compute the average of these averages

- In the second case you must
  - Take all the local statistics of the taxis sent between T1 and T2
  - Compute the average of each list of measurements' averages computed by all the taxis
  - Compute the average of these averages

Good Job!