

**FACULTY OF AUTOMATION AND COMPUTER SCIENCE COMPUTER SCIENCE DEPARTMENT**

**DISTRIBUTED SYSTEMS**

**Assignment 3**

Remote Procedure Call (RPC)

Smart Home Appliance

**Student**: Cățănaș-Kelemen Kaj Victor  
**Group**: 30244

2021

# Requirements

Suppose that the clients have intelligent home appliances that can be controlled remotely using remote procedure call (RPC). Each such device can communicate with the server that will compute the time when the device will be started for an optimal energy consumption.

Develop a client-side application (either a desktop application or a web application based on a JavaScript framework running from the browser) for the smart appliance associated to a client that:

* 1. gets the client hourly historical energy consumption over d days in the past
  2. gets the averaged energy consumption for the client over the past week (e.g. client

baseline) allows the selection of a program with a duration in hours (select a duration D of a program);

* 1. gets the best time to be started considering the baseline and the program duration to avoid energy peaks from the client (e.g. to minimize the maximum energy consumption for every hour of the day)

𝐶𝑜𝑚𝑝𝑢𝑡𝑒 𝑡𝑠, 𝑡𝑒 𝑠𝑢𝑐ℎ 𝑡ℎ𝑎𝑡 𝑀𝑖𝑛(𝑀𝑎𝑥(𝐵𝑎𝑠𝑒𝑙𝑖𝑛𝑒(ℎ) + 𝐸𝐷𝑒𝑣𝑖𝑐𝑒)), ∀ℎ ∈ [𝑡𝑠, 𝑡𝑒], 𝑡𝑒 = 𝑡𝑠 + 𝐷

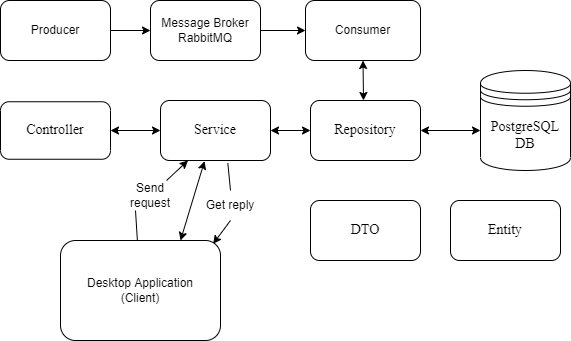
## Functional requirements:

* + - The client application displays a chart with the client historical energy consumption over d days in the past (default d = 7)
    - The client application displays the client baseline as a reference consumption for the next day
    - The client application allows the selection of a program with a duration (either from a list of programs or by entering directly the duration
    - The client application asks the server for the best start time in the next day to minimize the peaks of energy consumption. It displays the new chart of estimated consumption as the baseline summed with the device max consumption.

## Implementation technologies:

* + - Hessian-RPC

**a) Conceptual architecture of the distributed system**



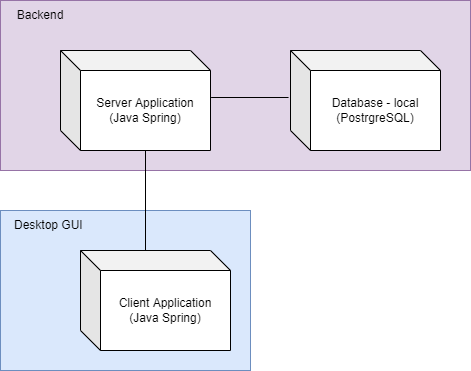
The Controller will receive HTTP requests signaled by the frontend and will send the appropriate reply according to the request. The Service will validate the received data, handle the request (CRUD/login) and access the required data from the Database through the Repository. The data received or sent to the Database by the Repository will be of type Entity and the data sent back or to the Service will be of type DTO (Data Transfer Object).

The Producer will read measurement values from files, send the data through the Message Broker.

The queue Consumer will preprocess the data sent from the broker. If it detects a measurement power peak that exceeds the sensor maximum threshold it notifies asynchronously the client on its web interface. In case the received measurement value does not result in a new power peek, it will be stored in the repository.

Presence of RPC model. Enables a caller hosted by a computer to call to a procedure that might be executed in a process hosted by another computer. Message-passing scheme with only one of the two processes being active at any time. The Client calls a remote method. The request is transmitted containing method name and parameters. After the request is received by the server and the server runs the method, the result of the computation is serialized and then transmitted back to the client. Finally, the result is received and the method returns.

**b) UML Deployment diagram**



**c) Build and execution considerations**

First, we need to run the server application and then start the client application which will display a GUI with a login menu for the client. After authentication, the client can select different features that will display specified charts based on his configurations and past energy consumption.

# Bibliography

1. <https://dsrl.eu/courses/sd/>
2. Lab Book: I. Salomie, T. Cioara, I. Anghel, T.Salomie, *Distributed Computing and Systems: A practical approach*, Albastra, Publish House, 2008, ISBN 978-973-650-234-7
3. Lab Book: M. Antal, C. Pop, D. Moldovan, T. Petrican, C. Stan, I. Salomie, T. Cioara, I. Anghel, Distributed Systems – Laboratory Guide, Editura UTPRESS Cluj-Napoca, 2018 ISBN 978-606-737-329-5, 2018,

<https://biblioteca.utcluj.ro/files/carti-online-cu-coperta/329-5.pdf>

1. <https://grpc.io/docs/tutorials/basic/go/>
2. [https://medium.com/pantomath/how-we-use-grpc-to-build-a-client-server-system-in-go- dd20045fa1c2](https://medium.com/pantomath/how-we-use-grpc-to-build-a-client-server-system-in-go-dd20045fa1c2)