AccountService Research

Purpose: create a service and estimate access time to it depending on input parameters

---

---

1. Create a service with the following interface:

```
public interface AccountService
{
/**
* Retrieves current balance or zero if addAmount() method was not called before for specified id
*
* @param id balance identifier
*/
Long getAmount(Integer id);

/**
* Increases balance or set if addAmount() method was called first time
*
* @param id balance identifier
* @param value positive or negative value, which must be added to current balance
*/
void addAmount(Integer id, Long value);
}
```

The service will work in the high-load fault-tolerant system.

The service must cache data in the memory and store it in DB (*Oracle, PostgreSQL, MySQL*)
or generate Exceptions if operation failed.
You can use any of protocols as a transport layer such as: *RMI, Hessian, HTTP*

---

2. Create a test client

The test client should be able to launch several concurrent streams on a defined subset of identifiers
- *rCount* – number of readers requesting the *getAmount(id)* method
- *wCount* – number of readers requesting the *addAmount(id,value)* method
- *idList* – list or range of keys that will be used for testing

These parameters can be set using the command line or configuration file
Several test clients can be launched simultaneously on one or several computers.

---

3. Receive request processing statistics from the *AccountService* server

---

For each of the two *AccountService* methods (*getAmount, addAmount*) you need to receive
- number of requests processed on the server per unit time (!!! not on the client)
- total number of requests from all clients

Statistics can be obtained from the service on demand by any method
or logged periodically.
It should be possible to reset the statistics to zero for the working service.