In5020 - Oblig 1 - Sigurd og Fredrik

Description of problem

The purpose of the delivery is to make a server client combo that transfers information and requests for information via remote invocation. The delivery will also implement caching of information, and searching for missing information.

Compilation

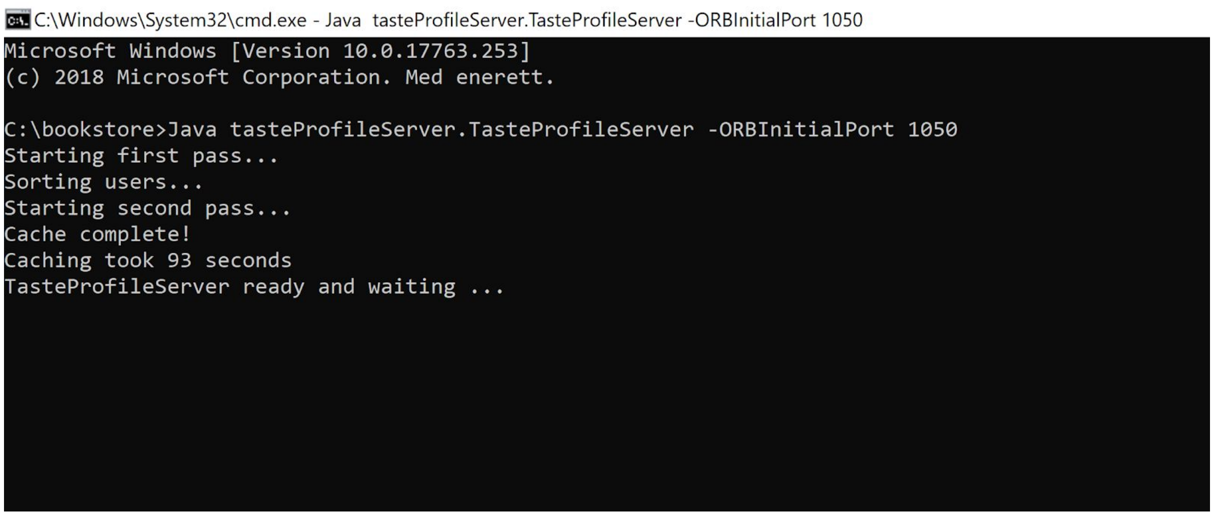
Compilation was done with eclipse.

Running the program

The program consists of two distinct parts; the server and the client.

First, ORBD must be started with the command:
start orbd -ORBInitialPort <port>
Where <port> is substituted for a number indicating the port to run the service. For example:
start orbd -ORBInitialPort 1050

Then the server is started with the command:
java -jar Server.jar -ORBInitialPort <port>
Where <port> is substituted for a number indicating the port to run the service. For example:
java -jar Server.jar -ORBInitialPort 1050

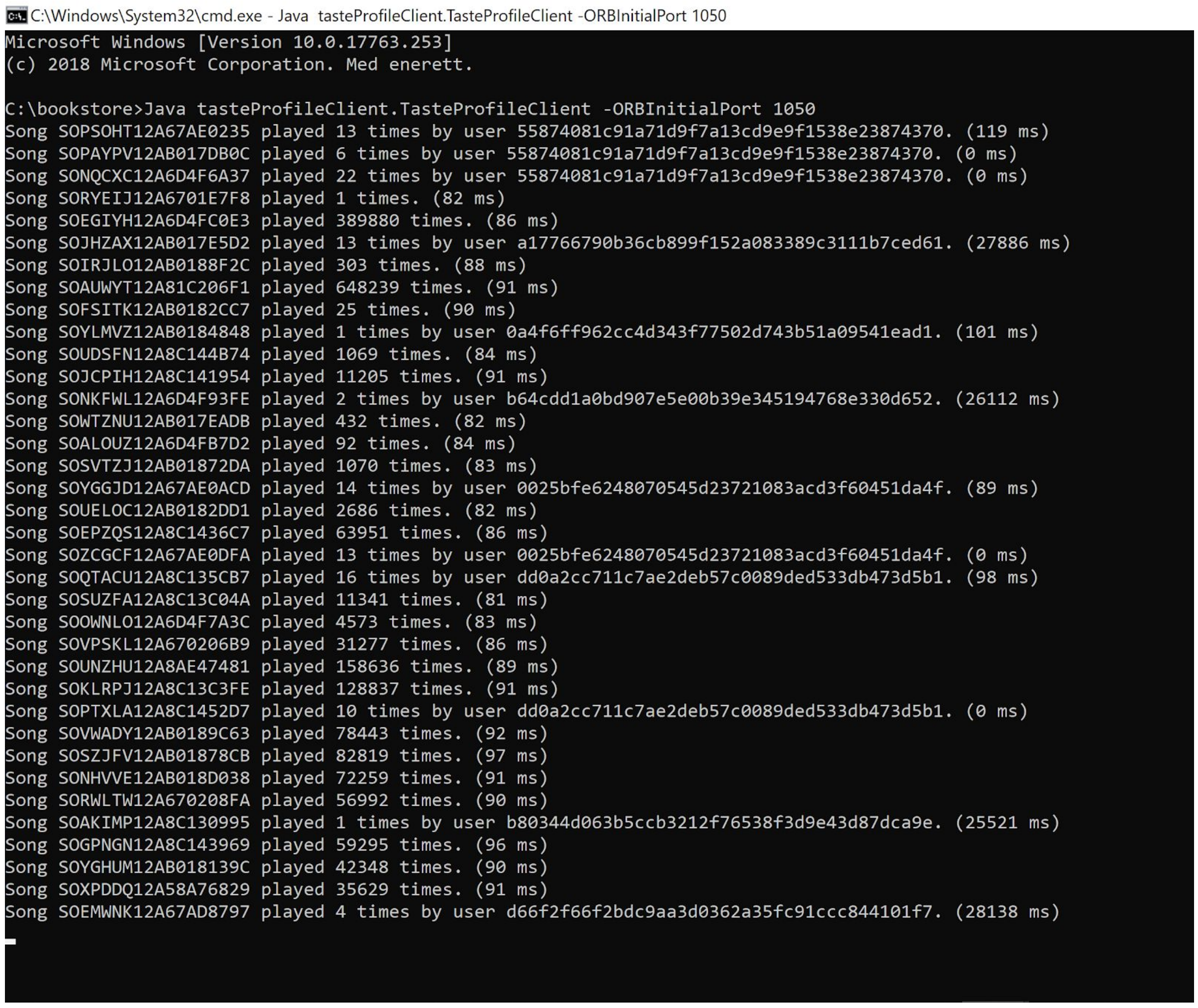


```
C:\Windows\System32\cmd.exe - Java  tasteProfileServer.TasteProfileServer -ORBInitialPort 1050
Microsoft Windows [Version 10.0.17763.253]
(c) 2018 Microsoft Corporation. Med enerett.

C:\bookstore>Java tasteProfileServer.TasteProfileServer -ORBInitialPort 1050
Starting first pass...
Sorting users...
Starting second pass...
Cache complete!
Caching took 93 seconds
TasteProfileServer ready and waiting ...
```

Server ready for remote invocations

Then the client is started with the command:
java -jar Client.jar -ORBInitialPort <port>
Where <port> is substituted for a number indicating the port to run the service. For example:
java -jar Client.jar -ORBInitialPort 1050



```
C:\Windows\System32\cmd.exe - Java  tasteProfileClient.TasteProfileClient -ORBInitialPort 1050
Microsoft Windows [Version 10.0.17763.253]
(c) 2018 Microsoft Corporation. Med enerett.

C:\bookstore>Java tasteProfileClient.TasteProfileClient -ORBInitialPort 1050
Song SOPSOHT12A67AE0235 played 13 times by user 55874081c91a71d9f7a13cd9e9f1538e23874370. (119 ms)
Song SOPAYPV12AB017DB0C played 6 times by user 55874081c91a71d9f7a13cd9e9f1538e23874370. (0 ms)
Song SONQCXC12A6D4F6A37 played 22 times by user 55874081c91a71d9f7a13cd9e9f1538e23874370. (0 ms)
Song SORYEIJ12A6701E7F8 played 1 times. (82 ms)
Song SOEGIYH12A6D4FC0E3 played 389880 times. (86 ms)
Song SOJHZAX12AB017E5D2 played 13 times by user a17766790b36cb899f152a083389c3111b7ced61. (27886 ms)
Song SOIRJLO12AB0188F2C played 303 times. (88 ms)
Song SOAUWYT12A81C206F1 played 648239 times. (91 ms)
Song SOFSITK12AB0182CC7 played 25 times. (90 ms)
Song SOYLMVZ12AB0184848 played 1 times by user 0a4f6ff962cc4d343f77502d743b51a09541ead1. (101 ms)
Song SOUDSFN12A8C144B74 played 1069 times. (84 ms)
Song SOJCPIH12A8C141954 played 11205 times. (91 ms)
Song SONKFWL12A6D4F93FE played 2 times by user b64cdd1a0bd907e5e00b39e345194768e330d652. (26112 ms)
Song SOWTZNU12AB017EADB played 432 times. (82 ms)
Song SOALOUZ12A6D4FB7D2 played 92 times. (84 ms)
Song SOSVTZJ12AB01872DA played 1070 times. (83 ms)
Song SOYGGJD12A67AE0ACD played 14 times by user 0025bfe6248070545d23721083acd3f60451da4f. (89 ms)
Song SOUELOC12AB0182DD1 played 2686 times. (82 ms)
Song SOEPZQS12A8C1436C7 played 63951 times. (86 ms)
Song SOZCGCF12A67AE0DFA played 13 times by user 0025bfe6248070545d23721083acd3f60451da4f. (0 ms)
Song SOQTACU12A8C135CB7 played 16 times by user dd0a2cc711c7ae2deb57c0089ded533db473d5b1. (98 ms)
Song SOSUZFA12A8C13C04A played 11341 times. (81 ms)
Song SOOWNLO12A6D4F7A3C played 4573 times. (83 ms)
Song SOVPSKL12A670206B9 played 31277 times. (86 ms)
Song SOUNZHU12A8AE47481 played 158636 times. (89 ms)
Song SOKLRPJ12A8C13C3FE played 128837 times. (91 ms)
Song SOPTXLA12A8C1452D7 played 10 times by user dd0a2cc711c7ae2deb57c0089ded533db473d5b1. (0 ms)
Song SOVWADY12AB0189C63 played 78443 times. (92 ms)
Song SOSZJFV12AB01878CB played 82819 times. (97 ms)
Song SONHVVE12AB018D038 played 72259 times. (91 ms)
Song SORWLTW12A670208FA played 56992 times. (90 ms)
Song SOAKIMP12A8C130995 played 1 times by user b80344d063b5ccb3212f76538f3d9e43d87dca9e. (25521 ms)
Song SOGPNGN12A8C143969 played 59295 times. (96 ms)
Song SOYGHUM12AB018139C played 42348 times. (90 ms)
Song SOXPDDQ12A58A76829 played 35629 times. (91 ms)
Song SOEMWNK12A67AD8797 played 4 times by user d66f2f66f2bdc9aa3d0362a35fc91ccc844101f7. (28138 ms)
```

Client running, processing data

Running options

Both the server and the client can be run without their respective cache by adding the argument "-noMemory", for example:
java -jar Server.jar -ORBInitialPort 1050 -noMemory
java -jar Client.jar -ORBInitialPort 1050 -noMemory

Remote invocation

Remote invocation was mostly implemented in precode, and is based on ORB. The ORBD daemon handles requests and invokes methods in the servant.

Profiles.idl

The idl was used to generate most of the profile files, like the user profile, song profile and more specific details like song counters. Here we inserted provided profiles and value types and compiled to get the necessary classes using the command:
idlj -fall tasteprofile.idl

Based on these we made the impl files, so we could make instances of the classes, but all the impl files we made are empty shells that merely extends the abstract classes, as all necessary methods were in the abstract classes.

Server

To start off the server does startup caching, this is implemented through two searches of the entire database. The methods related to caching are all placed in a separate file named tasteProfileCache, for better readability.
The first pass through it fills the songProfile hashmap with song profiles, and builds a hashmap called userPopularity of the user ids and how many plays they have.
The userPopularity map is converted to an array and sorted to find the top users.
The second pass through is used to find the information related to the top 1000 users, since that can't be found in the first pass through without extremely large memory usage.

To receive requests the server starts a servant, which is invoked by ORB and returns information, either from the cached information or by searching the database for the information in the case of a cache miss, or if server-side caching is turned off.

Client

The client's main purpose is to process the input file and switch between the correct remote calls, as well as printing the returned results to the terminal and to an output file. It also times each call.

The client also does some caching, but only of the most recently relevant user. Otherwise the client simply calls on ORB to get the information from the server.

Results

Here are the time results with client and server caches turned on and off.

Client on, server on 4min
Client off, server on, 9min
Client on, server off 22 min
Client off, server off 37 min

A cache hit on the client side takes about 0-2ms, a cache miss on the client side with a hit on the server side takes about 81-82ms(80ms delay) and a miss on both takes on average 27 to 30 seconds.

Output files are attached to the delivery in devilry.

Conclusion

Both remote invocation and caching was implemented and with acceptable performance given the dataset we handled. Caching demonstrated a significant speedup, and it is very clear that strategic usage and placement of caches can drastically improve performance for a tradeoff in increased memory usage. In this case, it took over 9 times to process with no cache. With selective caching of the most used data it does however not need to be a lot of memory compared to the significantly increased performance.