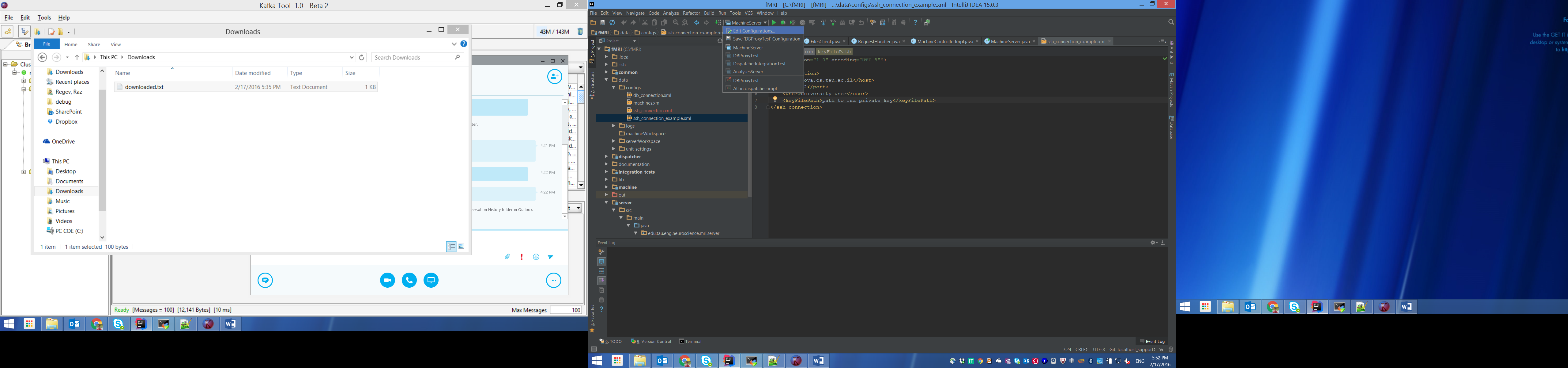
SSH tunneling for DB connection:

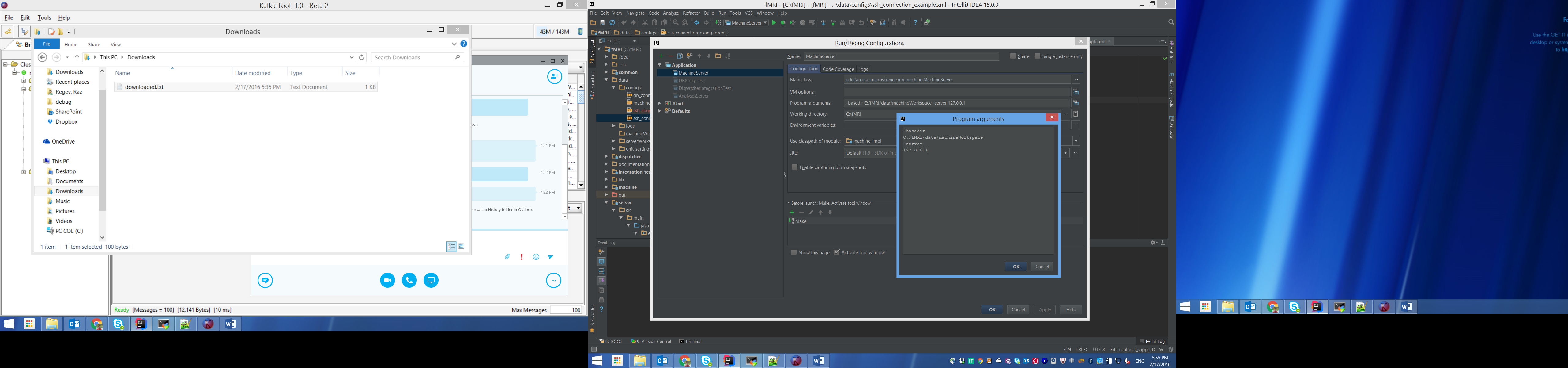
* Create a file data/configs/ssh-connection.xml (make sure not to stage/commit it with Git) based on the example: data/configs/ssh-connection-example.xml
* You can replace <keyFilePath> with <password> and write your password. I preferred not to save my password on the computer so I enabled both options.
* If you want to use a key to authenticate, follow the steps in: <http://www.cs.tau.ac.il/system/servers#servers19> (you only need id\_rsa on your computer so disregard the steps after that).

Networking with localhost:

* MachineServer now receives command line arguments so to run it from IntelliJ you need to configure them in the Run/Debug Configuration window (one way to do it is to run MachineServer and then edit configuration – see below).



* Set Program Arguments to: -basedir <dirPath> -server 127.0.0.1  
  Where <dirPath> should be replaced with the directory path where you want all the files to be saved on the “machine”. Example:



* AnalysesServer configuration example:

-db\_config

C:/fMRI/data/configs/db\_connection.xml

-ssh\_config

C:/fMRI/data/configs/ssh\_connection.xml

-machines

C:/fMRI/data/configs/machines.xml

-unit\_settings

C:/fMRI/data/unit\_settings/

Discussion Agenda:

* Networking:
  + ExecutionProxy should simply send and receive tasks to and from machines (preferably via HTTP post) – we have to send the task anyway in order to have to have the task ID, the Unit parameters and so on. The tasks will include paths and any other information the machine may need to request the files it requires – the idea is that files will be pulled by machines from a fileserver (might be the same physical machine as the program server, but logically separate) and not pushed by the ExecutionProxy. Serialization and deseralization of the tasks should be done by external libraries – it will be very easy for us.
  + TomCat and Server architecture
* Misc:
  + DB structure – tasks, users, subjects? Should we audit?
  + Failures – how do we know if a machine failed to execute a task? What if it got stuck? notifyFatal…