My experiment

Part 1 Define Experiment

Experiment overview

According to the global warming problem, my experiment gathering the information of the temperature through last two summer, after what, based on sample of temperature calculate and compare its median parameter, which could somehow show the changing through years. All filtered data from the web service, which were used for calculation, are written in the files with the name of the month and extension ".txt". That experiment was implemented on java language with using already existing project from GitHub repository (https://github.com/dvdme/forecastio-lib-java) and data collection which was taken from http://forecast.io/ source. According to the mathematical calculation I used package math3.stat.

Diagram depicting experiment

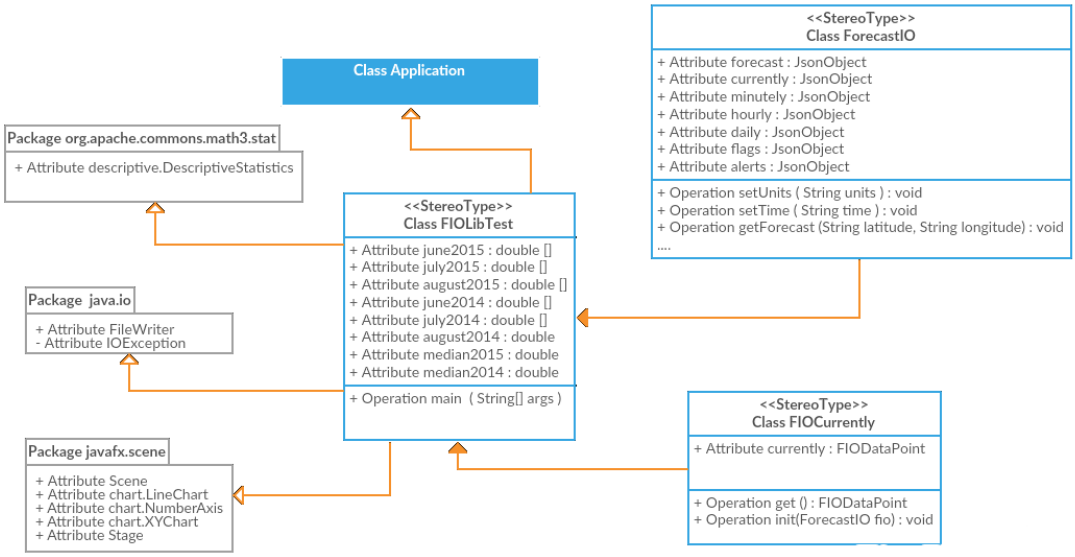

Figure 1UML-diagram of classes/packages dependency

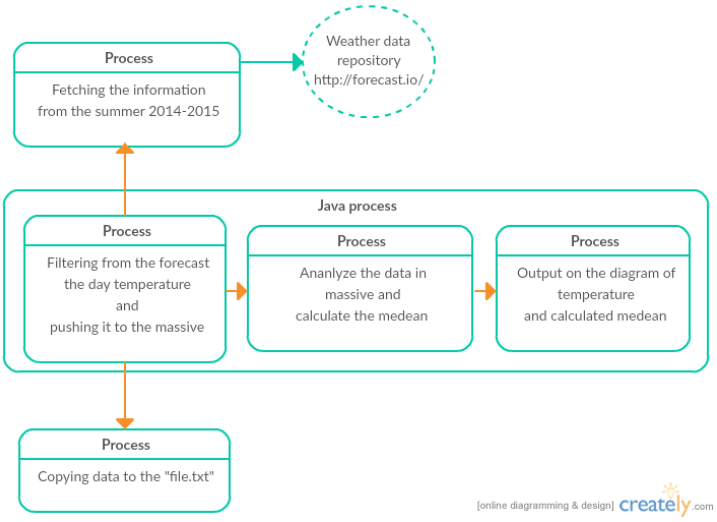

Figure 2 Archi model

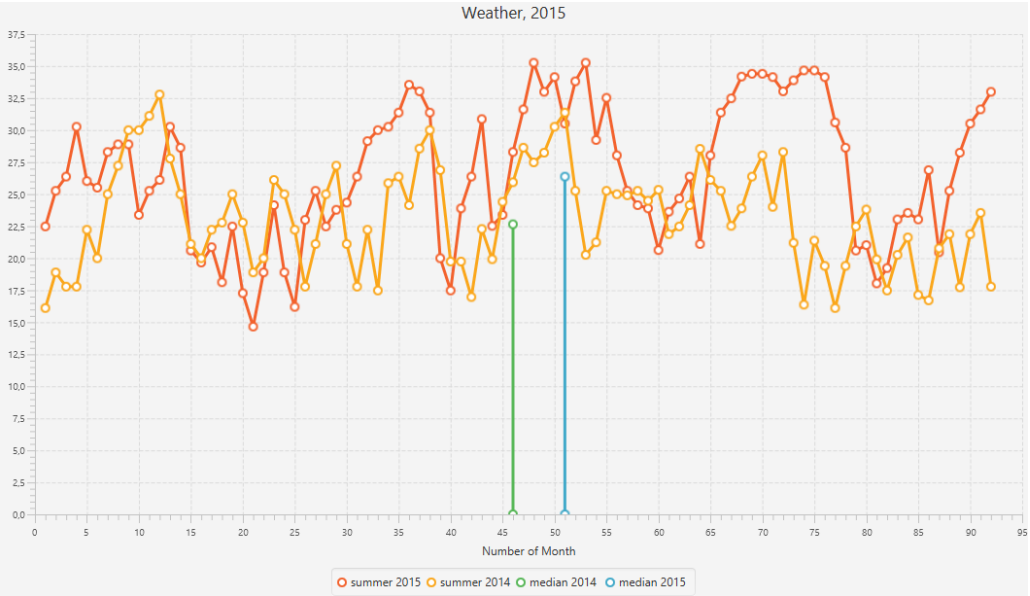

Figure 3 Results of the experiment

Part 2 Execution model analyzes

After the experiment was defined, we need to identify the process of the environment. According to the process migration framework we get the model which represents the process environment through the source system. To automate the process of gathering the information from PMF I used SPARQL Query, I identified parameters which are represented in the table below:

Table 1 environment process identification based on PMF

№	Name of the parameter	Value
0	Display identifier	- local_system
1	User	- uni
2	Version of the OS	- Ubuntu1604LTS
3	List of Debian packages	- libc6 - libtinfo5 - locales - libc6dbg
4	List of data files which read and written by the experiment	- folder “dist”
5	Check communication with external services	-
6	List addresses of external services	-
7	List dependencies that are neither data files, nor Debian packages, but are still used in the experiment.	- /home/uni/Documents/dist/run_process.sh - /usr/lib/locale/locale-archive - /home/uni/Documents/dist/

Data characterization

My experiment fetches data from the web-sit and filters them during that process. In the end of the experiment it produces the .txt files with all data which was analyzed during the experiment. Their parameters are represented in the table below:

Table 2 output files information

Name	Size (bytes)	extension	Format	Mime type	PUID
August2014	213	txt	Plain Text File	Text/plain	x-fmt/111
August2015	213	txt	Plain Text File	Text/plain	x-fmt/111
July2014	214	txt	Plain Text File	Text/plain	x-fmt/111
July2015	215	txt	Plain Text File	Text/plain	x-fmt/111
June2014	200	txt	Plain Text File	Text/plain	x-fmt/111
June2015	208	txt	Plain Text File	Text/plain	x-fmt/111

Validation requirements

Table 3 validation requirements

ID	Requirement	Sub-requirement	Metric	Target Value	Measurement point	Tool
R1	Access to the server should be available	Availability of the web service	Available Not available	Get any information from the web	Browser	https://api.forecast.io/forecast/ccf52e30eccc661da66988626e44ef56/48.2081740,16.3738190
		The get information format is not changed, so the data could be reached	Reachable Unreachable	true	Getting temperature from the service http://forecast.io	Java program
R2	Software	Installed Java virtual machine higher than 1.6 version	Installed Not installed	true	Getting information from command line. Command "java –version"	Command line
		Availability list of debian packages	exist not exist	true	The list is in table 1 (query 3)	-
		Software for reading the ".txt" extension	Exist Not exist	true	open ".txt" file	Notepad++
R3	Access rights on the computer	Access to write and read files	Access available Access unavailable	True	create files ".txt" in the directory	Notepad++

Run the experiment

To run the experiment, you should satisfy all requirements in the table 3, after type in command line "java –jar myexperiment_0.0.1.jar"