Software Reengineering

Intermediate Report

Ma INF 2020-2021

Igor Schittekat

1 Assignment

The Assignment can be found at
`https://ansymore.uantwerpen.be/2021-reengineering-project`

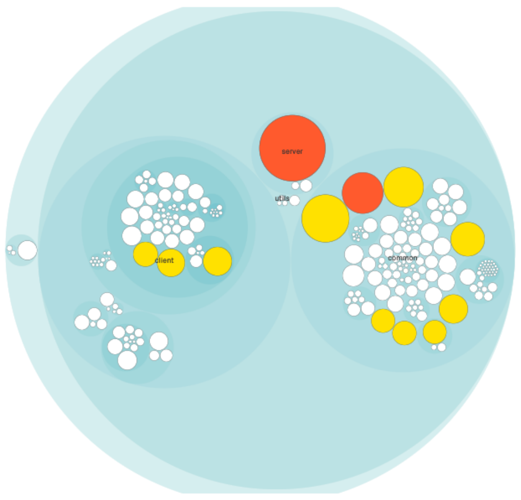
2 Introduction

For the previous milestone, I formed a group with Atisha Ribeiro. Unfortunately he decided to drop out this course, so I'm now my own group. I decided to fork the Megamek project myself, and added the assistents as collaborators again. The link to the new github repository is `https://github.com/IgorSchittekat/megamek`. In this report you can find the tools used so far to identify the parts that need to be re-engineered in the code.

3 Usage of Tools

3.1 CodeScene

The first tool I used was CodeScene, to visualize the files that need refactoring. This tool pointed me to some files that might need a better look.



Here we can see that Server.java in the server and Entity.java in common are marked orange. Those are files that probably need refactoring first. In common there are also 7 files marked yellow: AnnoType.java, MiskType.java, Mech.java, Compute.java, WeaponAttackAction.java, Tank.java and Aero.java, which need a look as well, and in Client.ui.swing there are 3 files that I will look at first: CharLounge.java, MovementDisplay.java and BoardView1.java.

3.2 Dude

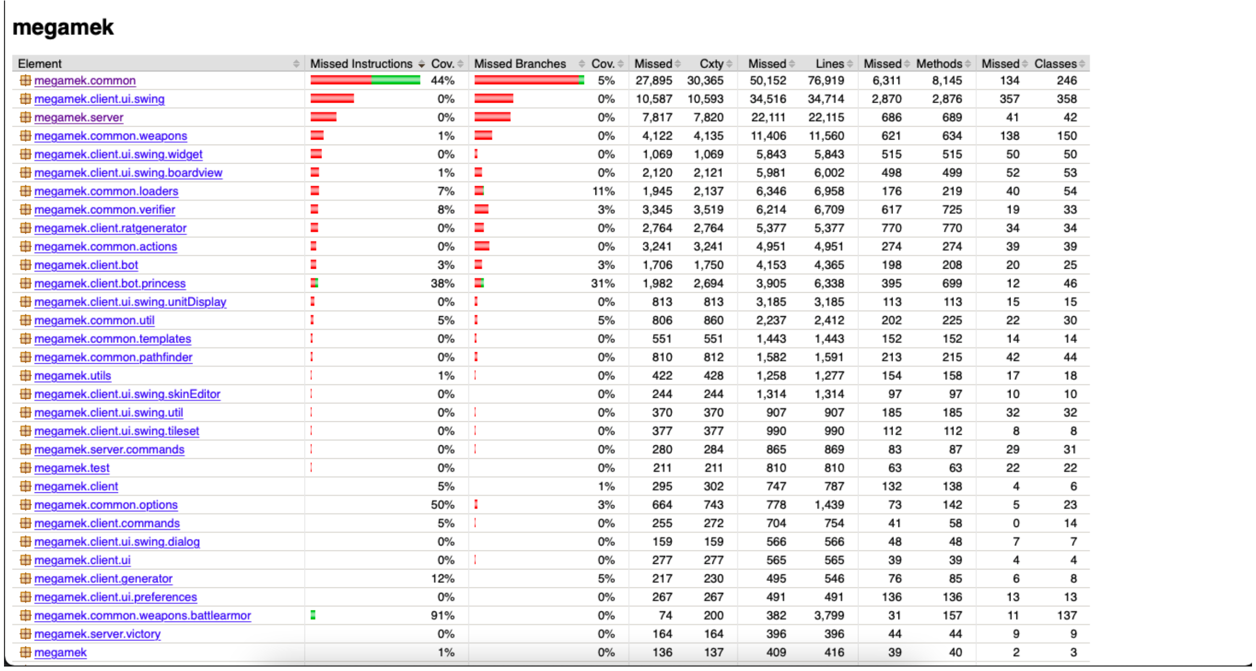
Next, I tried looking if there were code duplicates with Dude. This tool looked at the exact and modified duplicates. As noted by CodeScene, I found many duplicate parts in the Server.java file. When running this on the common folder, I found many more duplicates over the different files.

3.3 iClones

I also run iClones on the entire project to find duplicates. First I tried exporting to rcf format, but the provided rcf viewer kept crashing, and because it is a format only created by the creators of the tool to view the output, I couldn't find another tool to open the files. I decided to try the other export formats, but this was not very readable. I decided that it would be better to only focus on the duplicates that were provided by Dude.

3.4 jacoco

When running jacoco to get the test coverage, I got the following report:



megamek

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
megamek.common		44%		5%	27,895	30,365	50,152	76,919	6,311	8,145	134	246
megamek.client.ui.swing		0%		0%	10,587	10,593	34,516	34,714	2,870	2,876	357	358
megamek.server		0%		0%	7,817	7,820	22,111	22,115	686	689	41	42
megamek.common.weapons		1%		0%	4,122	4,135	11,406	11,560	621	634	138	150
megamek.client.ui.swing.widget		0%		0%	1,069	1,069	5,843	5,843	515	515	50	50
megamek.client.ui.swing.boardview		1%		0%	2,120	2,121	5,981	6,002	498	499	52	53
megamek.common.loaders		7%		11%	1,945	2,137	6,346	6,958	176	219	40	54
megamek.common.verifier		8%		3%	3,345	3,519	6,214	6,709	617	725	19	33
megamek.client.ratgenerator		0%		0%	2,764	2,764	5,377	5,377	770	770	34	34
megamek.common.actions		0%		0%	3,241	3,241	4,951	4,951	274	274	39	39
megamek.client.bot		3%		3%	1,706	1,750	4,153	4,365	198	208	20	25
megamek.client.bot.princess		38%		31%	1,982	2,694	3,905	6,338	395	699	12	46
megamek.client.ui.swing.unitDisplay		0%		0%	813	813	3,185	3,185	113	113	15	15
megamek.common.util		5%		5%	806	860	2,237	2,412	202	225	22	30
megamek.common.templates		0%		0%	551	551	1,443	1,443	152	152	14	14
megamek.common.pathfinder		0%		0%	810	812	1,582	1,591	213	215	42	44
megamek.utils		1%		0%	422	428	1,258	1,277	154	158	17	18
megamek.client.ui.swing.skinEditor		0%		0%	244	244	1,314	1,314	97	97	10	10
megamek.client.ui.swing.util		0%		0%	370	370	907	907	185	185	32	32
megamek.client.ui.swing.tileset		0%		0%	377	377	990	990	112	112	8	8
megamek.server.commands		0%		0%	280	284	865	869	83	87	29	31
megamek.test		0%		0%	211	211	810	810	63	63	22	22
megamek.client		5%		1%	295	302	747	787	132	138	4	6
megamek.common.options		50%		3%	664	743	778	1,439	73	142	5	23
megamek.client.commands		5%		0%	255	272	704	754	41	58	0	14
megamek.client.ui.swing.dialog		0%		0%	159	159	566	566	48	48	7	7
megamek.client.ui		0%		0%	277	277	565	565	39	39	4	4
megamek.client.generator		12%		5%	217	230	495	546	76	85	6	8
megamek.client.ui.preferences		0%		0%	267	267	491	491	136	136	13	13
megamek.common.weapons.battlearmor		91%		0%	74	200	382	3,799	31	157	11	137
megamek.server.victory		0%		0%	164	164	396	396	44	44	9	9
megamek		1%		0%	136	137	409	416	39	40	2	3

This report shows that very few parts of the code are covered, and thus many tests have to be written before we can refactor. The initial coverage report is pushed to the Github repo so I can compare this later.

3.5 SonarQube

I managed to run SonarQube, but only on single files at a time, and not even for all files. This was because the memory kept running out, even after increasing the allowed memory. When refactoring a class, this tool might come in handy, but because it runs out of memory so quickly, I decided not to use it to look for files that need refactoring because it would take too long to run everything separately.

4 Reengineering goals

From the gathered information I decided to first take a look at Entity.java. It was marked orange by CodeScene, so I will look why and try to decide what needs to be refactored.
Once I understand the class, tests need to be written, as this class is not really covered. If I'm finished with that, I can refactor and test if nothing breaks.

Next I want to take a look at Server.java. It is a very large class which might need some refactoring. Because it is such a large file, I don't want to start with it immediately.
Again, the first goal is to look deeper at the class and try to understand what the methods do. Because it is a very large class I think it might be a god class and want to split it into different classes where possible.
Once I understand this class and decided it is worth refactoring, tests need to be written, as this class is not covered at all. This needs to be done beforehand to make sure nothing breaks.

After I am done with those 2 classes, I think I might look at other classes and try to remove duplicates if it is necessary.