

Contents

1	2018	5
1.1	April	6
	2nd Semester - W1 (2018-04-04 08:27)	7
	W2 - Scope & Risk Management (2018-04-11 09:35)	9
1.2	May	11
	W3 - FUNCTION POINT ESTIMATION (2018-05-02 07:38)	12
	W4 - Unit Testing (2018-05-16 07:41)	17
	W5 - Refactoring (2018-05-22 09:24)	19
	W6 - Design Pattern (2018-05-29 07:27)	21
1.3	June	23
	W7 - Metrics (2018-06-06 07:40)	24
	W8 - Installation Test (2018-06-12 12:16)	27
	W9 - Final Exam (2018-06-20 08:00)	28

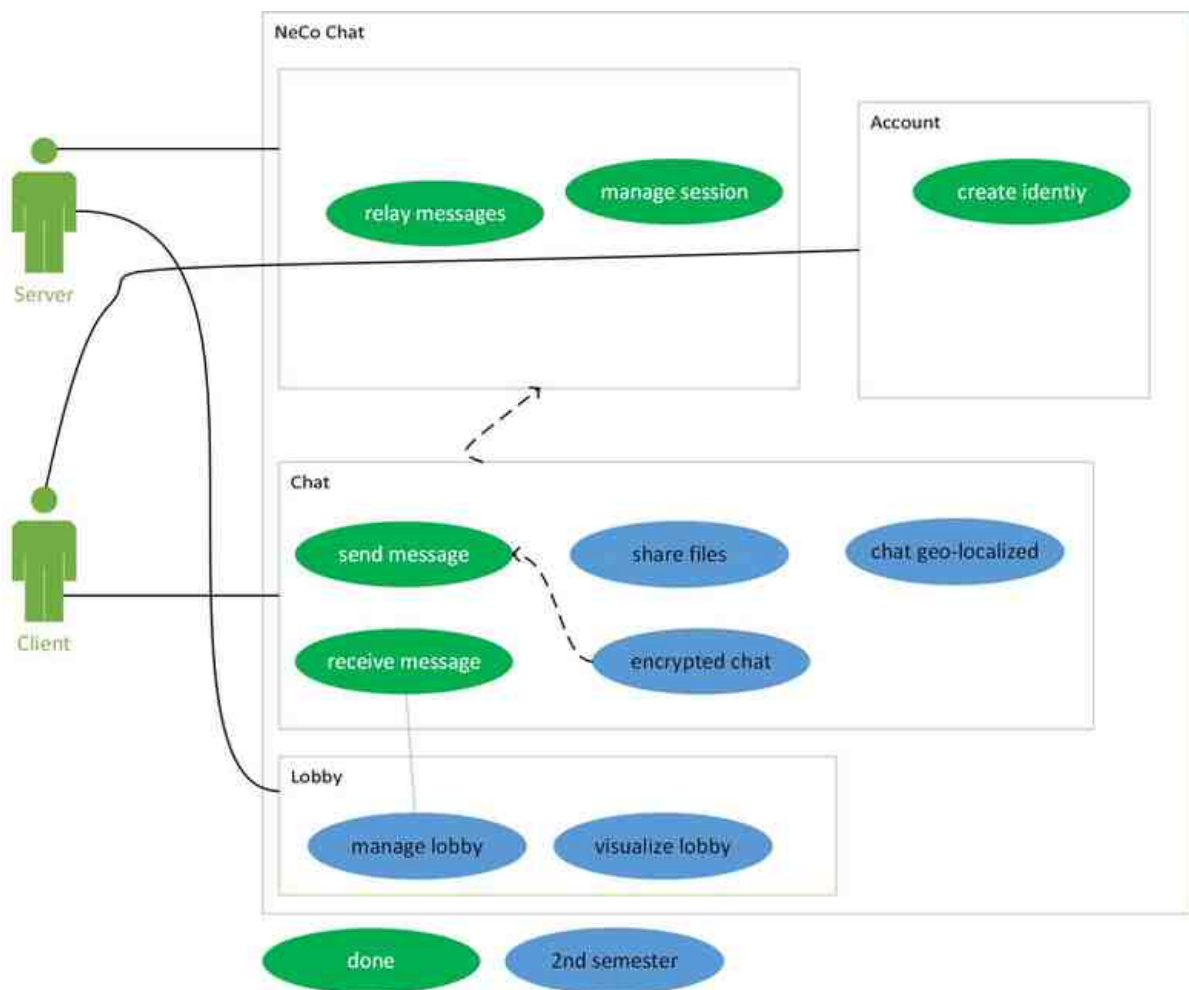
1. 2018

1.1 April

Hi everyone,

for the first week of the new semester we revisited our code and documentation and set all up, to work further on our project.

We changed our scope to new requirements and updated our Overall Use Case Diagramm:



We still rely on our well know technologies.

You can see how many hours we spend on each use case and how many LOC are written per use case, in the following. All values are estimated, using our Jira issues and our GitHub commits:

- Relay messages
 - 8h / 1 PD
 - 250 LOC
- Manage session
 - 16h / 2 PD

- 150 LOC
- Create identity
 - 8h / 1 PD
 - 200 LOC
- Send message
 - 16h / 2 PD
 - 1000 LOC
- Receive message
 - 16h / 2 PD
 - 1000 LOC

Fortunately our team consists of the same team members as last semester :)

W2 – Scope Risk Management – NeCo (2018-04-11 09:35:12)

[...] an overall overview you can visit our first post of this semester, where you find our overall use case [...]

W2 - Scope & Risk Management (2018-04-11 09:35)

Hi there,

this week, we digged further into project management, defining our scope more detailed and identifying risks of this project.

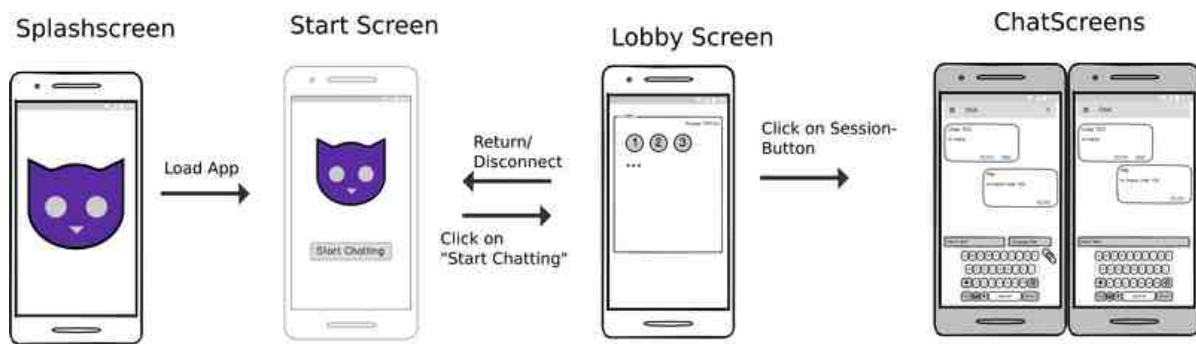
- You can find our risk management table [1]here.

Identified	Description	prob of occurrence (in %)	damage/impact (1-10)	mitigation strategy	person in charge	risk factor (0-10)
showstopper at critical time	changes in code lead to an fatal error, before presentation	0,4		splitting branches in master, develop (Git Workflow); regular testing 7 in the end; feature freeze	Maurice	2,8
server shutdown	shutdown due to failure/updates	0,3		update freeze before endpresentation; backups; 7 docker	Mirko	2,1
underestimated workload	to much workload of this course regarding the workload of other courses	0,4		flexible workload 5 management	Alex	2
loss of data	due to device failure;	0,1		code management using 10 GitHub; track code sharing	Alex	1
illness	fatal	0,3		code sharing: communication via WhatsApp; regular commits 2 to feature branches		0,6
loss of team members	loss of team members due to exmatriculation	0,02		share knowledge via 8 GitHub; document detailed	Alex	0,16
						0
						0
						0

- We made a time estimation for our use cases done in the last semester, which you find here [2]here.

UC	Documentation	Coding	Testing	Total	Warm-Up Time FP
relay messages	2h	12h	5h	20h	5h
manage session	1h	10h	1h	12h	5h
create identity	1h	6h	1h	8h	3h
send message	2h	10h	3h	15h	8h
receive message	2h	10h	2h	14h	6h
share files	1h				
chat geo-localized	1h				
encrypted chat	1h				
manage lobby	1h				
visualize lobby	1h				

This is the screenflow of our application, showing the flow using our application:



For an overall overview you can visit our [3]first post of this semester, where you find our overall use case diagramm.

Best regards,
Team NeCo

1. https://github.com/Haus4/NeCo/raw/develop/docs/sem_2/risk_management.xlsx
2. https://github.com/Haus4/NeCo/raw/develop/docs/sem_2/time_estimation_uc.xlsx
3. <https://necoproject.wordpress.com/2018/04/04/2nd-semester/>

freshmangoshake (2018-04-25 07:44:42)

Hello NECO, nice to see that you are back in business! I like the screenflow image of your app. It's quite easy to understand and provides a great overview over your app. The table of your Riskmanagement is also easy to understand and I like that you added a description. But I'm not sure, what the 'risk factor' means: Are those values in range of 0 to 10? Best regards, REACT

necoproject (2018-04-25 07:51:40)

Thank you for your comment. The risk factor is the probability times the impact of the risk. Therefore the higher the risk factor, the more we have to be aware of it. But it should range from 0-10. We updated the document to make that visible. Greetings, NeCo

thomaspoetzsch (2018-04-25 08:22:09)

Hey there, Team NeCo, we think your risk management table and the time estimations are fine. You should really look into providing GitHub-flavoured markdown, so that people visiting your blog can access your documents without downloading them and having some office application installed (you can create tables there, too -> <https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet#tables>). We really enjoyed the wireframes/mockups of your application. What tool did you use to create them? Best regards, Team poest

necoproject (2018-04-25 08:28:27)

Hey team poest! We would like to keep working with office products instead of writing tables in markdown simply because of usability reasons. Instead, we will provide images of the tables. Concerning the wireframes we used Balsamiq (the cloud trial: <https://balsamiq.cloud/>). Greetings, NeCo

1.2 May

Hello!

Today we want to show you our function point estimation.

First of all we will give you a little explanation about function points.

Function points are used for estimating the time that will be spent on a certain Use Case. They are calculated in reference on External Inputs, Outputs, Inquiries as well as Internal and External Logical Files from the User's view, thereby they do not depend on the used technology.

To estimate how long it will take to implement a Use Case, function points of UC that have been already implemented are put in relation to the time we spent on it.

We used the calculation for our function points from „[1]TINY TOOL„.

Now to the actual work we done.

The following table shows our completed Use Cases with the time we needed to implement. You can also see the at the beginning explained references we calculated the function points on.

use case	estimation (h)	logged (h)	transaction	DET	RET	FTR	Complexity	function points
relay messages	20h		external input	0			low	
			external output	2			low	
			external inquiries	0			low	
			internal logical files		7		avg	
			external interface files	0			low	81.9
manage session	22h		external input	0			low	
			external output	1			low	
			external inquiries	1			low	
			internal logical files		3		low	
			external interface files		1		low	34.6
create identity	3h		external input	0			low	
			external output		1		low	
			external inquiries	0			low	
			internal logical files		1		low	
			external interface files		1		low	16.8
send message	15h		external input	1	1		low	
			external output		1		low	
			external inquiries	0			low	
			internal logical files			3	low	
			external interface files			3	low	48.3
receive message	14h		external input	0			low	
			external output	1	1		low	
			external inquiries	0			low	
			internal logical files			3	low	
			external interface files			3	low	46.2
share files			external input	1			low	
			external output	1			low	
			external inquiries	0			low	
			internal logical files			1	low	
			external interface files			1	low	19.9
chat geo-localized			external input	1			low	
			external output	1			low	
			external inquiries	0			low	
			internal logical files			1	low	
			external interface files		1		low	27.3
encrypted chat			external input	0			low	
			external output	0			low	
			external inquiries	0			low	
			internal logical files			3	low	
			external interface files		1		low	34.6
manage lobby			external input	1			low	
			external output	0			low	
			external inquiries	0			low	
			internal logical files			3	low	
			external interface files			4	low	46.6
visualize lobby			external input	0			low	
			external output	1			low	
			external inquiries	1			low	
			internal logical files			3	low	
			external interface files			1	low	34.9

The acronyms DET, RET and FTR mean Data Element Type, Record Element Type and File Type Reference. Through these we identified the complexity which can be low, average or complex.

[2] Here you can find the whole document which also includes the data for the new Use Cases. In this document you also see the estimation for the new Use Cases these get explained later in this blog entry.

For the calculation of function points with TINY TOOL you need some extra information about the whole application.

Complexity Adjustment Table

ITEM	COMPLEXITY ADJUSTMENT QUESTIONS	SCALE					
		No Influence 0	1	2	3	4	Essential 5
1	Does the system require reliable backup and recovery?	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
2	Are data communications required?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
3	Are there distributed processing functions?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
4	Is performance critical?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
5	Will the system run in an existing, heavily utilized operational environment?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
6	Does the system require on-line data entry?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
7	Does the on-line data entry require the input transaction to be built over multiple screens or operations?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
8	Are the master files updated on-line?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
9	Are the inputs, outputs, files or inquiries complex?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
10	Is the internal processing complex?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
11	Is the code to be designed reusable?	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
12	Are conversion and installation included in the design?	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
13	Is the system designed for multiple installations in different organizations?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
14	Is the application designed to facilitate change and ease of use by the user?	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

[Domain Characteristic Table](#) | [FP Calculation](#)

In this screenshot you can see what we choose for our project.

We calculated the function points for all our Use Cases. For example in the next picture you can see the table for the Use Case „Relay Message“.

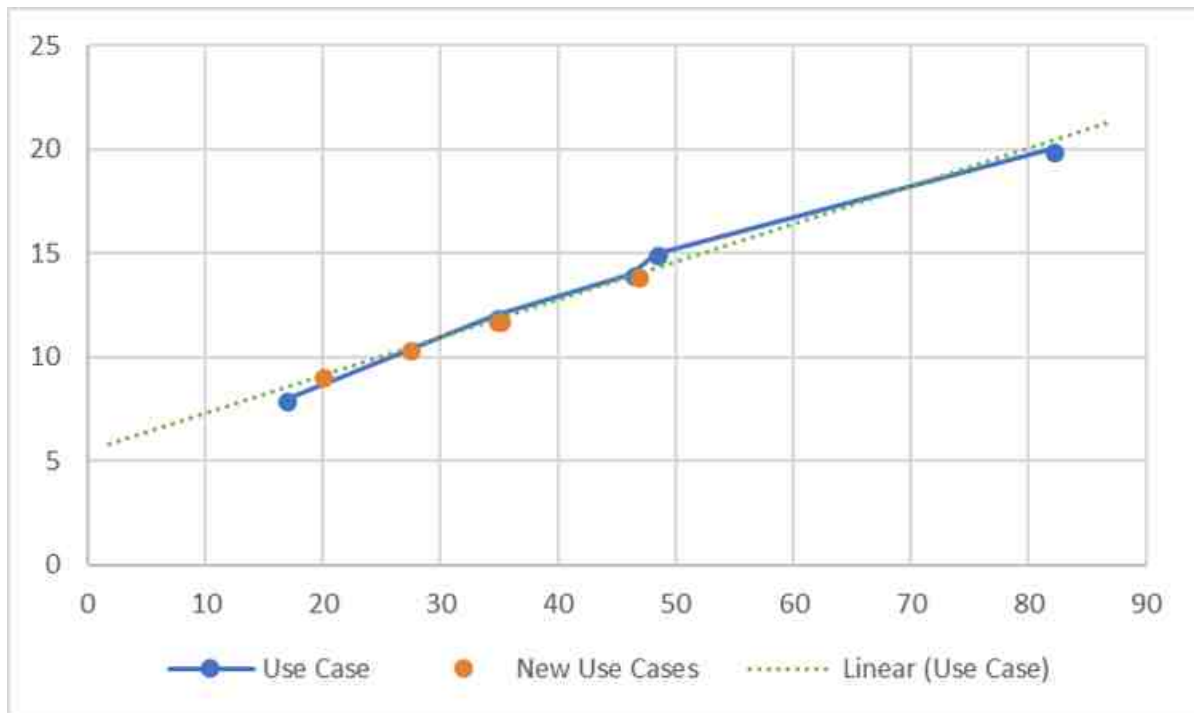
Domain Characteristic Table

MEASUREMENT PARAMETER	COUNT (value >= 0)	WEIGHTING FACTOR		
		Simple	Average	Complex
Number of User Input	<input type="text"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
Number of User Outputs	<input type="text" value="2"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
Number of User Inquiries	<input type="text" value="0"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
Number of Files	<input type="text" value="7"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
Number of External Interfaces	<input type="text" value="0"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>

This Use Case has 81,9 function points. [3]Here you can see the whole document.

With the calculated function points we generated a diagram which shows the interact of the function points and the person hours.

FP calculation and time estimation diagram



We used this graph to estimate the time for our remaining Use Cases. In the graph you can see these as orange dots. You can only see four dots, because two Use Cases („encrypted chat“ and „visualize lobby“) have the same amount of function points. Below you can see our time estimation for our new use cases.

New Use Cases		
share files	19,9	9,11845
chat geo-localized	27,3	10,46895
encrypted chat	34,6	11,8012
manage lobby	46,6	13,9912
visualize lobby	34,9	11,85595

I hope you liked it and if you want give us some feedback.

Greetings,

NeCo

1. http://groups.umd.umich.edu/cis/course.des/cis525/js/f00/harvey/FP_Calc.html?tCountVal=0#FPCalc
2. https://github.com/Haus4/NeCo/raw/develop/docs/sem_2/time_estimation_uc.xlsx
3. https://github.com/Haus4/NeCo/blob/develop/docs/sem_1/UC5_RelayMessage.md

VSS (2018-05-02 09:41:22)

Hey Team Neco, it is crazy how well your use cases are lined up. Maybe you could add labels to the axes and the points... Your estimation seems like you will finish all use cases this semester, nice! Maybe you could add the estimated time to the big excel table. I think all lines in the big excel table should have a DETs entry... So you can get the correct complexity :D Best regards, Team VSS

necoproject (2018-05-02 09:53:37)

Hi team vss, thanks for your compliment. We plan to add some more information to our tables soon. We got our complexity on summing the values of all DET, RET, FTR cells in one row together. Thanks for your feedback. Kind regards, Team Neco

dffcblog (2018-05-02 10:18:10)

Hello Team Neco, I like your work very much. Your time estimation last semester looks quite accurate. Maybe you could name the axes on your time estimation diagram. I see that you didn't use the Null estimation but I think that it's not necessary in your case because without you get that nice ideal line in your time estimation diagram. Best regards, Florian from WakeMeInTime

necoproject (2018-05-02 10:24:29)

Hi Team DFFC, thanks for your good idea, we will add this information. We didn't think of that, but we will add this, when we change the other information. Best regards, Team Neco

W4 - Unit Testing (2018-05-16 07:41)

Hi everyone,

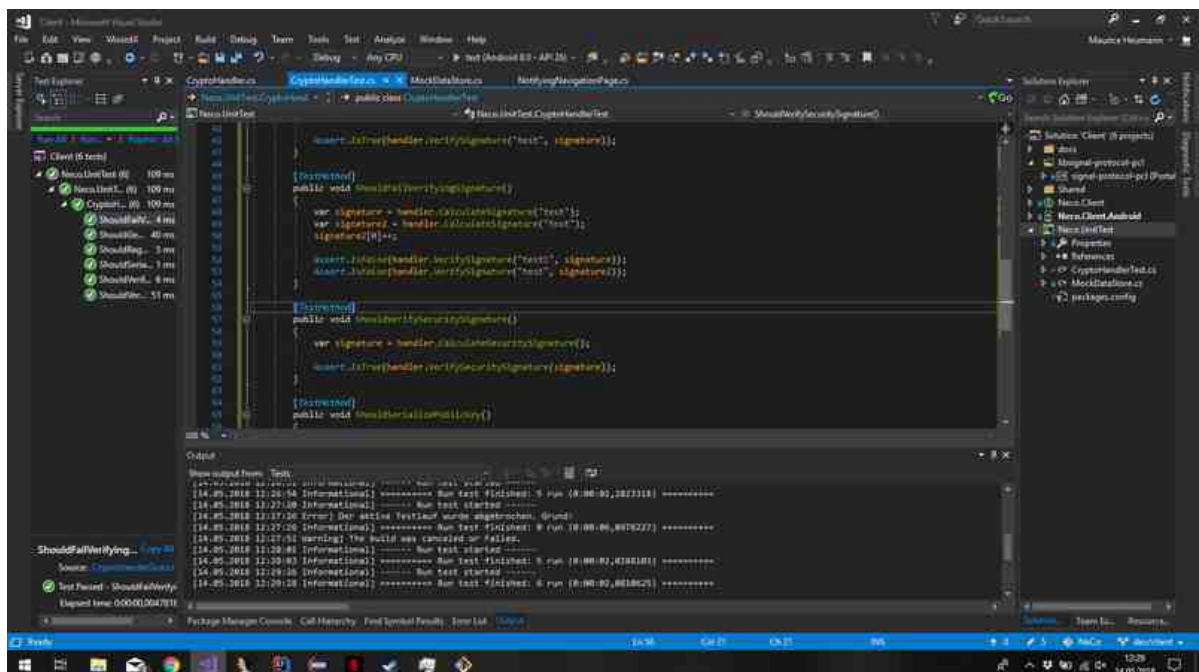
this week, we created our test plan

this week's topic was testing our project. So we started writing tests for our application improving our applications by finding bugs in early stage of development. Therefore we first wrote a Testplan which you can find [1]here. It's going to be completed with all information needed in the future, while we work more on testing our project.

We use the MSTest framework for C # and it has been added as dependency to our project [2]here.

A link to our tests can be found [3]here.

For your overview, in the screenshot below, you can see the tests running in our IDE.



We're looking forward to your feedback

Best regards,

your Team Neco

1. https://github.com/Haus4/NeCo/blob/develop/docs/sem_2/test_plan.pdf
2. <https://github.com/Haus4/NeCo/blob/dev/client/Client/Neco.UnitTest/packages.config>
3. <https://github.com/Haus4/NeCo/tree/dev/client/Client/Neco.UnitTest>

superwomantinf16b4 (2018-05-16 08:01:37)

Hello Team Neco, it is nice to see that you are also doing tests for your project. Can it be that you have accidentally uploaded the original test plan without your changes? Best regards, Team SuperWoman

necoproject (2018-05-16 08:05:21)

Thank you for your reply. The correct document has not been pushed yet, but should be available soon. Best regards, Team NeCo

thomaspoetzsch (2018-05-16 09:33:30)

Hey Team NeCo, your testing documents seem sufficient. I think it is good to have a working IDE integration for Unit Tests, just as you do. It would be nice for us if you could put a link to the Unit Tests into your blog post. Best regards, Team PPR

necoproject (2018-05-16 09:35:03)

Hi, thanks for your reply, we added the link to our blog post. Kind regards, Team Neco

W5 - Refactoring (2018-05-22 09:24)

This week we learned more about refactoring. We studied the introduction to refactoring from Fowler, you can learn about it [1]here. For learning purposes each of us trained the idea of refactoring on some exercise project.

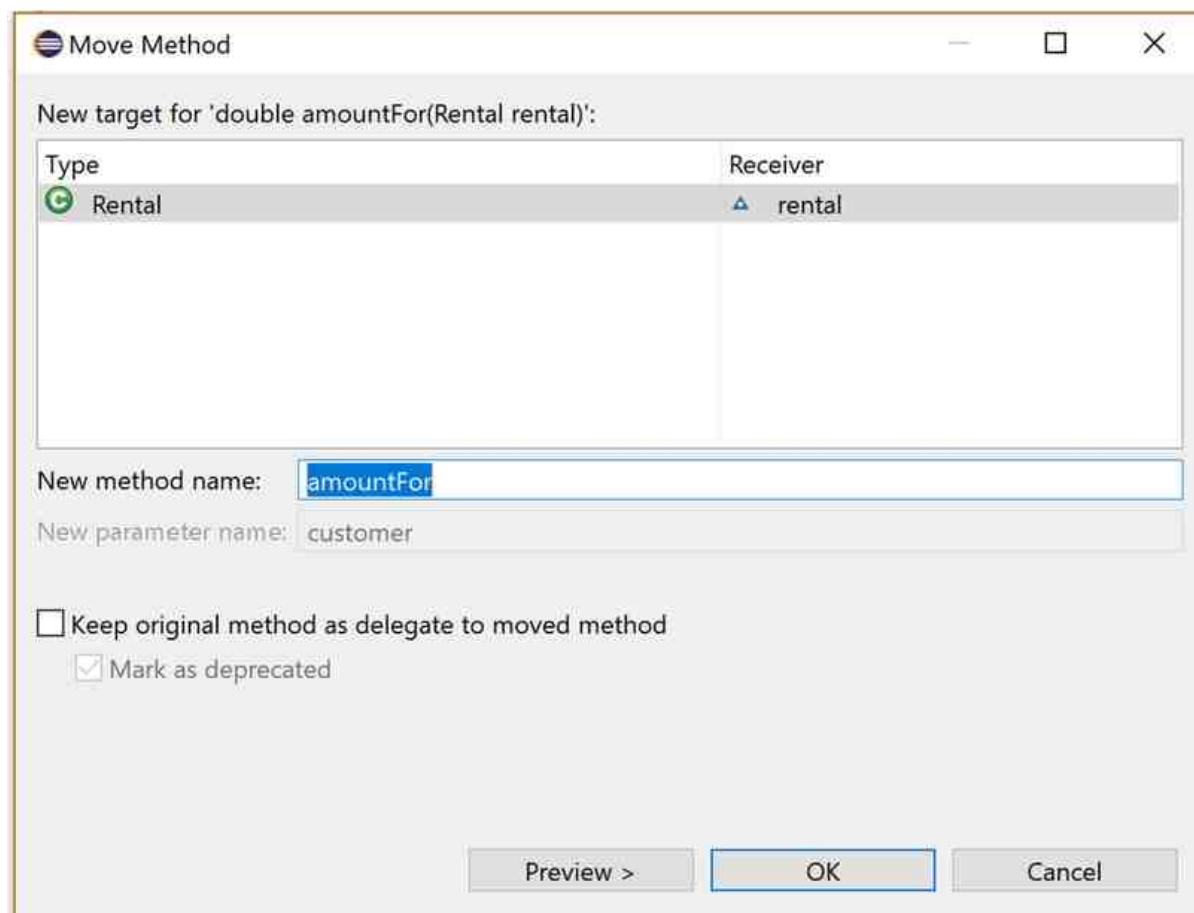
You can find each of our projects here on our team member's GitHub Account.

Maurice's [2]GitHub

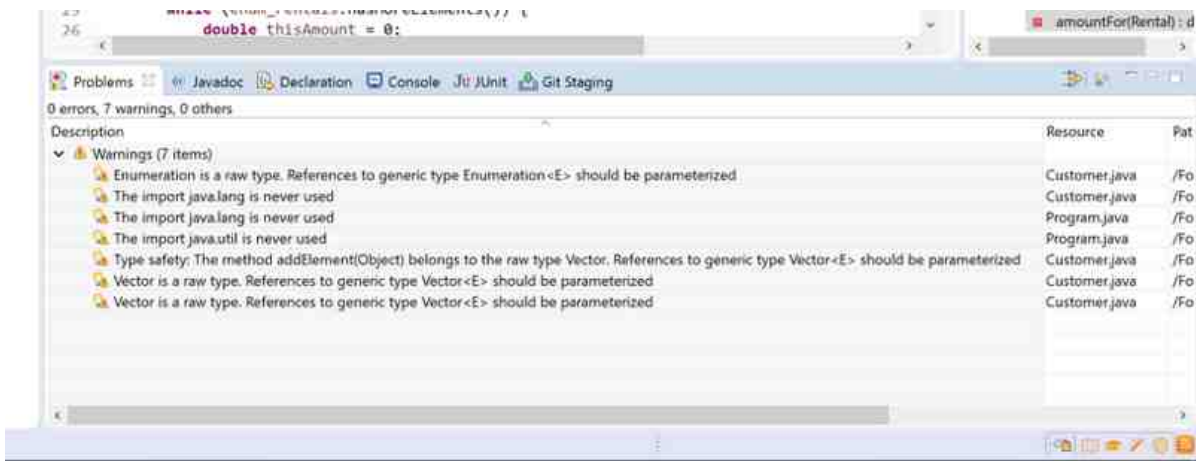
Mirko's [3]GitHub

Alex' [4]GitHub

In the screenshot below you can see, how Eclipse is helping us to refactor, first you can automatically move methods.



Furthermore Eclipse shows warnings like unused imports etc. while you code, so you might inspect these warnings early on.



Best Regards,

Team Neco

1. https://www.csie.ntu.edu.tw/~r95004/Refactoring_improving_the_design_of_existing_code.pdf
2. <https://github.com/momo5502/refactoring>
3. <https://github.com/LokiTheMango/Fowler-Refactor>
4. <https://github.com/ShiroOrihara/FowlerRefactoring>

dennyfl (2018-05-23 08:15:10)

Hey Team Neco, great work this week. I quickly checked your repositories and it looks like you understood the basic principle of refactoring. Please fix the spelling on "exercice " to "exercise" and the blog entry will be even better :) best regards Team DFFC

necoproject (2018-05-23 08:19:14)

Hi Team DFFC, thanks for your good feedback, we changed it. Best Regards Team Neco

thomaspoetzsch (2018-05-23 08:21:57)

Hey there, We really liked your blog article, and your repos on GitHub seem fine. It would be interesting to see how your projects perform on Codacy. Best regards, Team PPR

necoproject (2018-05-23 08:29:05)

Thank you for the response, we added codacy to one of the projects. Greetings, NeCo

W6 - Design Pattern (2018-05-29 07:27)

Hi everyone,

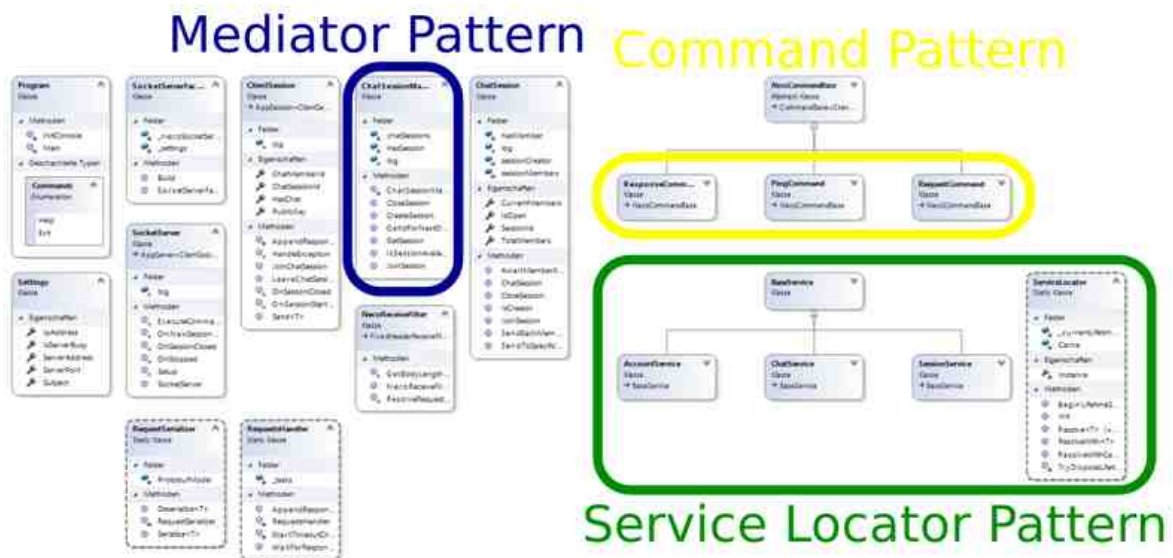
this week we diged further into refactoring and implemented some design patterns.

We've chosen a [1]service locator pattern to be used on our server. This pattern is typical for c #.

Therefore we rebuild our `InfrastructureInitializer` to a `SocketServerFactory`.

As we already used some patterns since the beginning of development, we decided not to compromise the code integrity completely. Therefore not really much changed in our code.

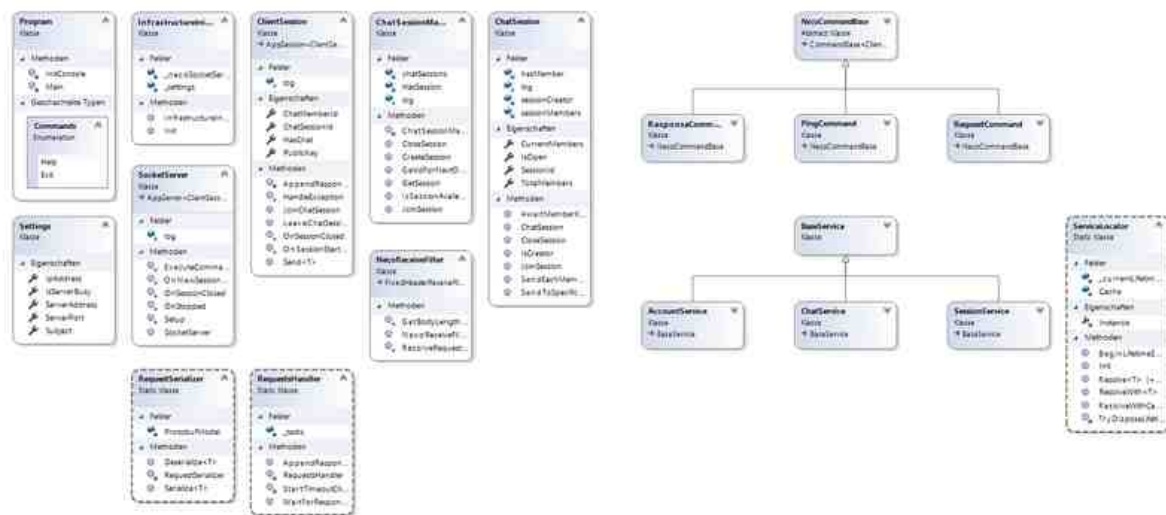
[2]



The picture above unfolds our pattern used.

The pictures below show our class diagramm before we implemented the pattern

[3]



Kind regards,

Team Neco

1. https://en.wikipedia.org/wiki/Service_locator_pattern
2. https://raw.githubusercontent.com/Haus4/NeCo/develop/docs/img/server_patterns.png
3. https://github.com/Haus4/NeCo/raw/develop/docs/img/before_server_cd.PNG

possiblynotpotatoguy (2018-05-29 08:04:20)

Hey Team Neco, your pattern looks good. It is great that you used patterns from the beginning, even if you didn't realize using them. We did the same. You could improve your blog post, if you could link to a bigger version of your picture. Best Regards Team react

necoproject (2018-05-29 08:14:11)

Thank you for your comment, links to bigger pictures have been added. Best regards, NeCo

dennyfl (2018-05-29 08:30:18)

Hey guys, i like your picture, as it clearly shows where the patterns are used in your project. The problem is that people might not know what your pattern does, so maybe explain them briefly, or give links to some additional information? best regards DFFC

necoproject (2018-05-29 08:40:19)

Hi DENNYFL, we added a link to additional information, so you are able to look it up if you want to. Thanks for your reply Kind regards, Team NeCo

1.3 June

W7 - Metrics (2018-06-06 07:40)

Hi,

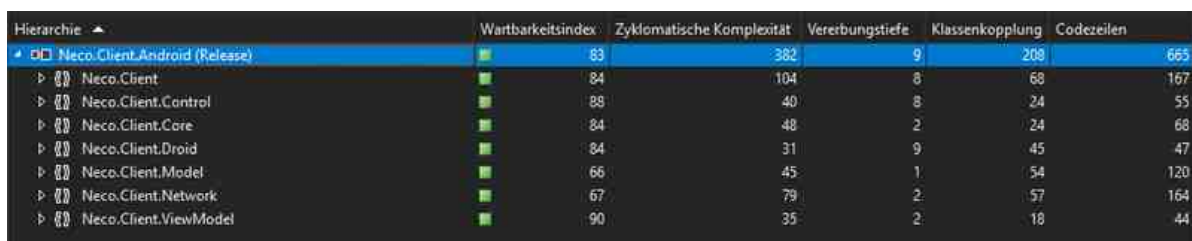
this week we looked closer at metrics. We used the metrics tool inside our IDE Visual Studio 17 and therefore got some detailed insights into our code.

Getting it to work for Xamarin took a bit of time and effort, due to the different architectures.

It calculates several metrics:

- [1]Cyclomatic complexity
- [2]Depth of inheritance
- [3]Class coupling
- [4]Maintainability index
- Lines of code

[5]



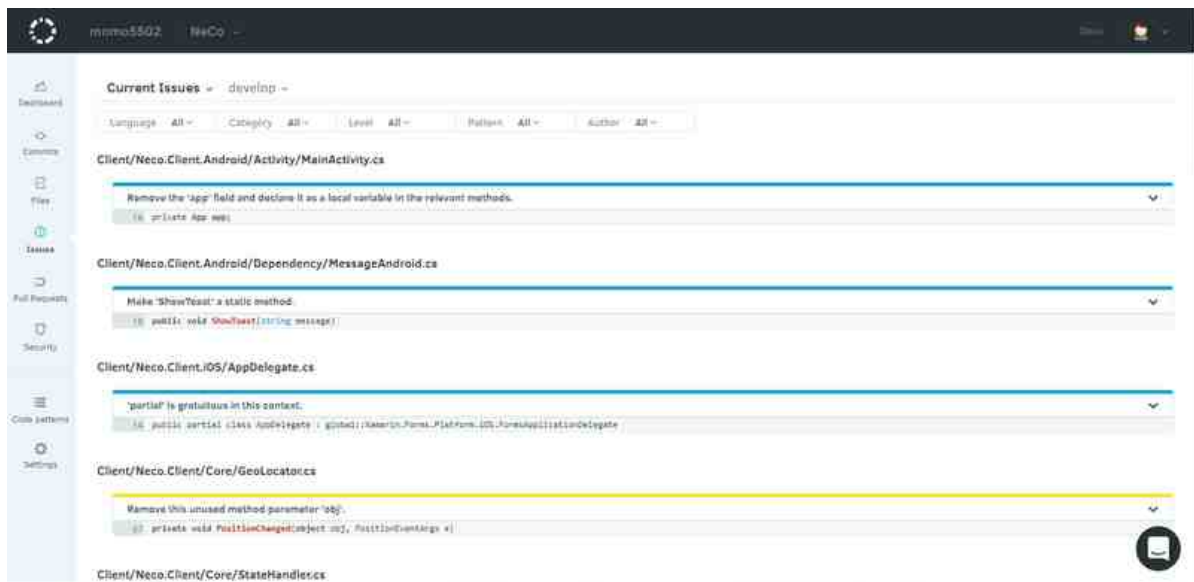
Hierarchie	Wartbarkeitsindex	Zyklomatische Komplexität	Vererbungstiefe	Klassenkopplung	Codezeilen
Neco.Client.Android (Release)	83	382	9	208	665
Neco.Client	84	104	8	68	167
Neco.Client.Control	88	40	8	24	55
Neco.Client.Core	84	48	2	24	68
Neco.Client.Droid	84	31	9	45	47
Neco.Client.Model	66	45	1	54	120
Neco.Client.Network	67	79	2	57	164
Neco.Client.ViewModel	90	35	2	18	44

Those are the values before our code refactoring. Surprisingly, after having done the refactoring task, the metric values didn't change notably. The overall cyclomatic complexity even increased. This shows that refactoring the code by extracting methods, simplifying expressions or performing small optimizations doesn't change the algorithm used, but only restructures it and makes it more readable to the user most of the times. Code metrics instead seem to analyze the algorithm itself neglecting (partly) how well structured the implementation is.

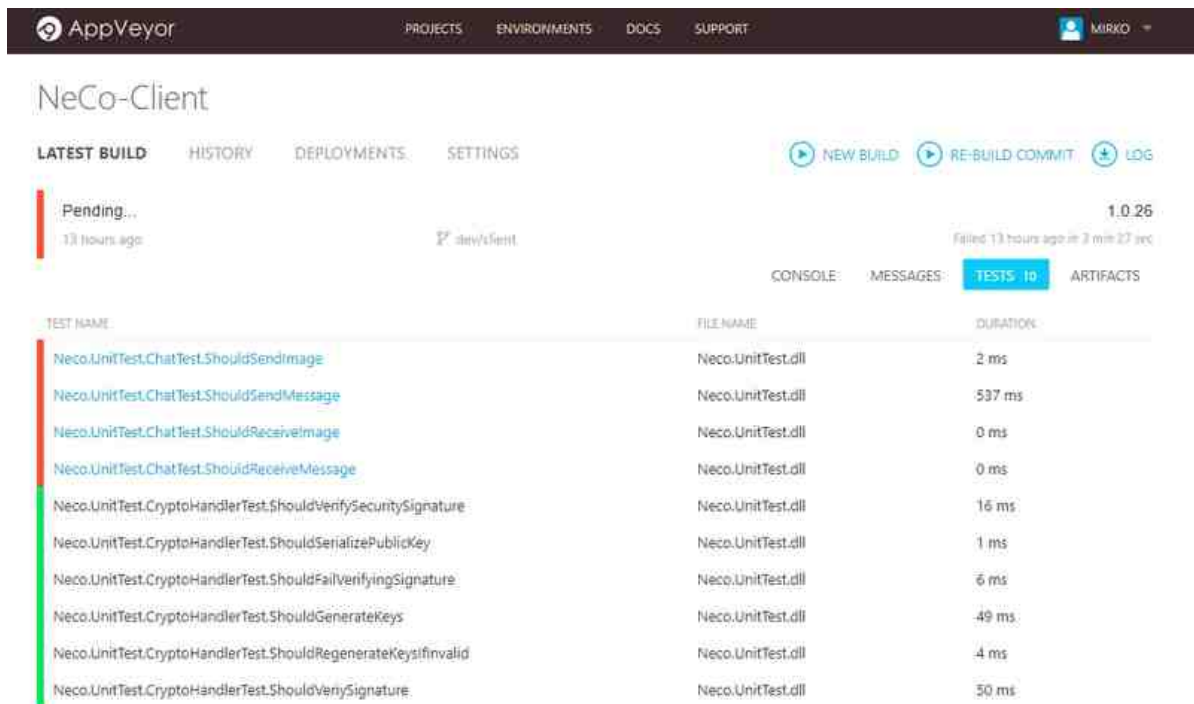
This actually turns the code metrics into a useful tool to get a feeling of how good the logic is, but doesn't necessarily help to refactor small parts of the code.

Additionally, we integrated codacy to our project which shows us additional issues and points of optimization:

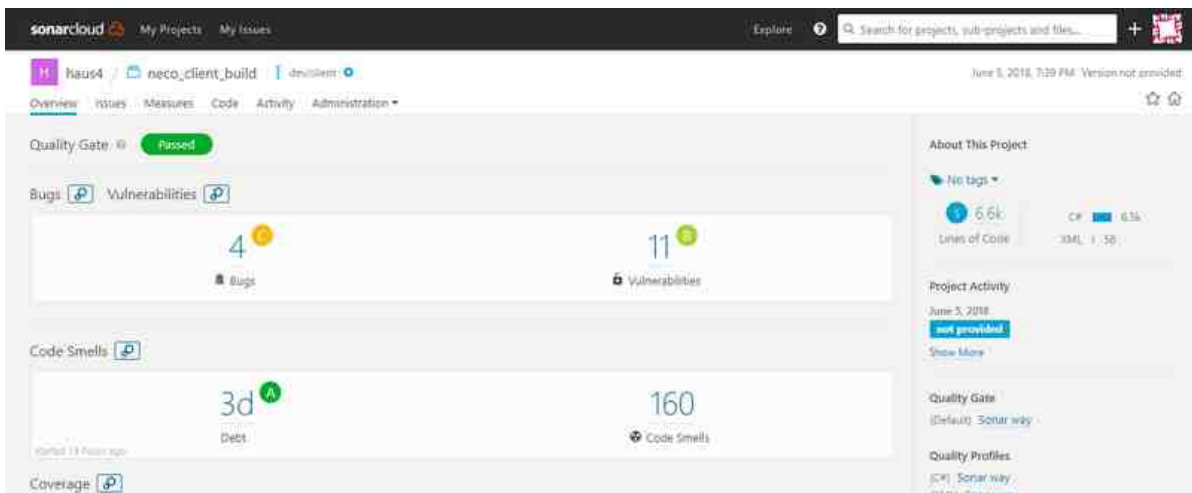
[6]



Also here is a picture of our build-server running our tests:



Finally we also added sonarqube to our project to track bugs/codesmells and coverage:



Kind regards,

Team Neco

1. <https://blogs.msdn.microsoft.com/zainnab/2011/05/17/code-metrics-cyclomatic-complexity/>
2. <https://blogs.msdn.microsoft.com/zainnab/2011/05/19/code-metrics-depth-of-inheritance-dit/>
3. <https://blogs.msdn.microsoft.com/zainnab/2011/05/25/code-metrics-class-coupling/>
4. <https://blogs.msdn.microsoft.com/zainnab/2011/05/26/code-metrics-maintainability-index/>
5. <https://necoproject.files.wordpress.com/2018/06/metric.png>
6. <https://necoproject.files.wordpress.com/2018/06/codacy.png>

dennyfl (2018-06-06 08:31:54)

Hey Team Neco, looks good, i like how you used a lot of different tools, so people can compare the results. You should definitely add some code example that your metrics tool told you to fix. And also add a screenshot after the fix, so we see the difference. And depth of inheritance and cyclomatic complexity are in the same line, i think u wanted to separate them :)

best regards DFFC

necoproject (2018-06-06 08:40:43)

Hi DennyFL, thanks for your feedback. We needed to work long on the implementation of code metrics, therefore we're going to add the code examples soon, as our code improves over the future. You will see the difference very soon. And we separated the two things. kind regards, Team NeCo

superwomantinf16b4 (2018-06-06 08:44:25)

Hello Team Neco, Your blog entry looks good. But it would be nice to see an example where your metrics tool asks you to change something and you have decided to ignore the advice. Of course, you should also give reasons for this decision. Best regards, Team SuperWoman

necoproject (2018-06-06 09:06:45)

Hi team SuperWoman, first thanks for your feedback. Our IDE Visual Studio actually doesn't show us these type of advices, but we got codacy, which does this. Best regards, Team NeCo

W8 - Installation Test (2018-06-12 12:16)

This week we published our app on the playstore. Therefore we needed a google developer account which we fortunately had. If you want to install it on your own Android device you can follow the link below.

On the playstore it is published to everyone and got already downloads. We also tested the successful installation with some of our friends and family.

For a more detailed insight we propose you to take a short survey, which we made for us to know, how everything works and if there are any problems.

You can find the survey [1]here, thanks for participating.

The results of the test can be inspected [2]here.

[3]



1. <https://de.surveymonkey.com/r/J5BPWFC>

2. <https://de.surveymonkey.com/results/SM-928N6PMCL/>

3. <https://play.google.com/store/apps/details?id=de.dhbw.neco.client&pcampaignid=MKT-Other-global-all-co-prtnr-py-PartBadge-Mar2515-1>

possiblynotpotatoguy (2018-06-14 10:26:31)

Hey team neco, the installation of your app was successful. We could connect to other users and chat. The app works. Great job. Best Regards Team react

W9 - Final Exam (2018-06-20 08:00)

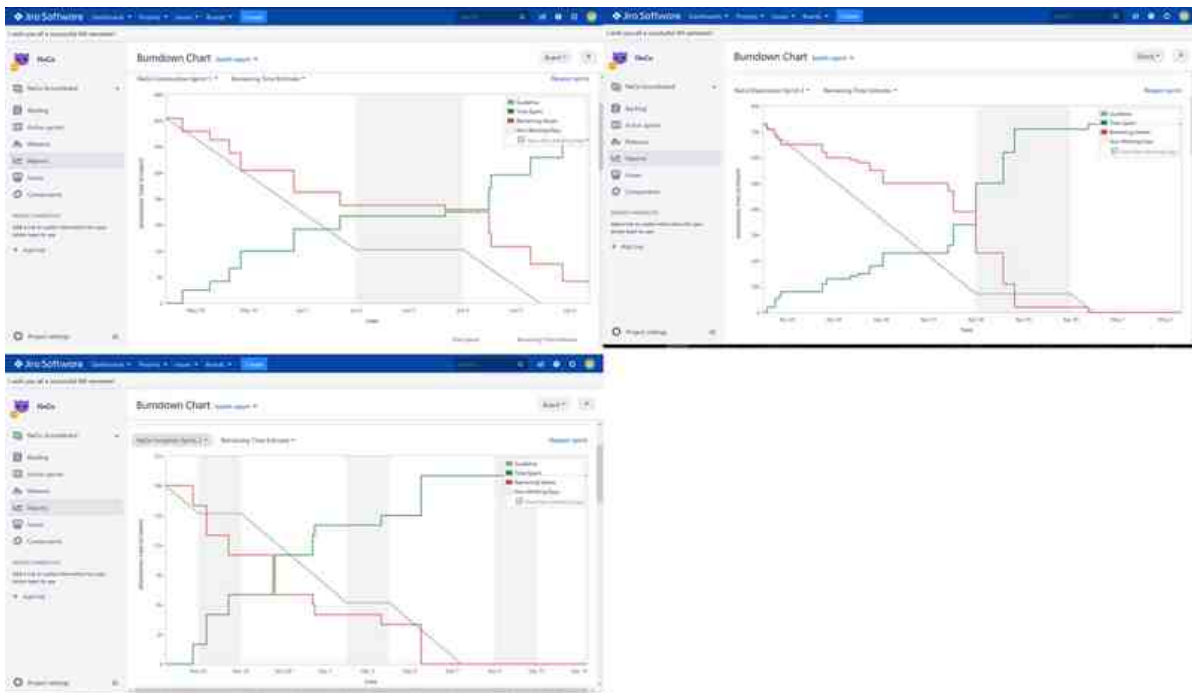
This week we have our final exam, therefore this is going to be our last post on this blog for this second semester.

So finally we can take a short break and resume what we have done.

You can visit our whole documentation [1]here.

Furthermore we want to show three clean sprints, which we made this semester, using Jira.

[2]



Also you can find a video of using our app [3]here.

Kind regards,

Team NeCo

1. <https://github.com/Haus4/NeCo/tree/master/docs>
2. https://raw.githubusercontent.com/Haus4/NeCo/develop/docs/img/cleanSprint_all.png
3. <https://sendvid.com/jtl15615y>