Evaluation of My Individual Project Work

Name: QI LI (QI5@KTH.SE)
University: Royal Institute of Technology KTH, Sweden
School: Information and Communication Technology
Program: TCOMK_2
Date of issue: 2017-06-09

Table of contents

Abstract

Considering the current technology trend, social system and the complexity of how people meet up during an event or activity. The project group decides to create an application that can ease the process of how people meet other participants of the event during a group activity. There are ten participants during the whole project. The group decides to separate the team into front end, back end, designer and android. Most of the contribution and cost in the project are indirect. It is complicated to trace back each contribution into individual person. The goal of this report is to use the Scrum project management system, GitHub pulse and Self Governence Developer Framework find out the contribute level of each person in the project group. The solution calculates how much time has been used in total useful lines of code and compare them with the theoretical time and the time professional used.

Keyword

Meetup, JoinUp, process, geolocation, contribution, management

Introduction

General Background

The geolocation technology is convenient for human nowadays. Everyone's smart phone has installed at least one software that can show user the path, what transportation should user take and how long time does it take to the desired destination with few click. When it comes to meet up with people other than user itself and the destination address is generated by these people. The traditional geolocation software has issues which highly depend on user know the specific address that other people generate and they all triggered by the user side. These issues can cause human mistake of the software when comes to lato nguage communication, timing and some other uncontrollable coincident.

Specific Background

The opportunity of the project has been locked down on current geolocation software did not do a good job when it comes to meet up with other people. After the team has analyzed the opportunity of the topic, the development goal has been locked down on how to ease the process of meeting up at certain geolocation point with the help of currently existing technology. During the working process in the project, the group has been separated into front end, back end, designer and Android team. Each team has their own responsibility but at same time all the work are connected together through Version Control System.

Problem

Since all the work are connected with different reason and each team member's work has been distributed all over the project, To find out the contribution level of each person directly from the code and what has each team member do will cost a lot of efforts. Since each line of code has comment and signature of each team member, the project group has considered to evaluate the contribution thought lines of the code (LOC) but the group has realized that each line of code has to absorbing function and data from other codes. The problem remains as the project group has no insight into how individual developer work is distributed across the project.

Research Method

To be able to find the insight of how individual developer work is distributed across the project, the team needs to find out what effort from each team member has been spending on during the whole project. This is also our goal of this essay. The method that we are using to reach our goal is Self-Governance developer framework (SGD) and analyzes through the whole project with the Scrum project management method. The tools we are be using are GitHub Pulse and contribution graph.

Background

In current society, the technology has become a necessary thing in human's life. What technology can do is extremely simplify the step of each life maintaining process and entertainment of humans daily routine. Base on this opportunity the project group decided to develop an application that eliminates the complexity of most common thing that people (stakeholders) do all time except work which is how to meet up someone in certain location without suffering from language barrier and the limitation of human analyzes skill.

The application must base on user's requirement and full fill our goal of the project. To be able to proceed, we constantly check what future user need and send out demo to non-developer user to collect feedback. Based on this feedback, we can highly improve our user experience and ensure the quality of the product. The development is base on Google Map Application Programming Interface. With exists Google Map user can clearly know where the user is and how to get to the location but we are base on these functions to create a unique application to reach our goal.

Team management:

The team has been separate from backend, front-end and Android. Each team uses unique technology to reach the common goal.

Front end:

The front end is using Vue.js as base framework and Element UI to improve the visual interaction with user. The front end is being written in pure JavaScript to make sure that the system is light weighted, multiplatform compatible and easy to access.

Back end and database management:

The backend is based on Sprint and database is based on Google SandGrid.

Android:

We choose to implement Java as programming language inside Android team. Since each person knows Android well and has some experience within it.

Development the Integrated development environment and Version Control System:

All the front-end code is written in JetBrain WebStorm with highly compatibility with Vue.js and Git Version Control System. In the backend, we use JetBrain IntelliJ and Android Studio for Android team. With the help of the Integrated development environment and Version Control System, the group is able to constant monitor the process of the project, handle mistake in no time and reach the continuous delivery.

Project Management Tool:

We use Scrum as project management tool and process model. It is also the breaking point for us to find out the contribution of each individual during the project work. Scrum has

provided us a really good general view of the project during each scrum meeting.It also provide us with good time estimation and risk management with it story point, breaking task and general meeting schedule system(daily scrum, sprint meeting and etc.).

Non-developer activity within Project management tool:

We have set our test case and definition of done carefully, evaluate and improve them during the whole project. We start from poor test cases to each individual has to test their own code to ensure the quality than post a review in GitHub issue and wait for other team members to review and test the code. The stories are defined as done during the scrum meeting when the code has been tested by any of the team members and the issue has been closed.

User experience Designer's design has to proceed in parallel with the project development process. Each design has its own test cases but their general test case is that the design has to pass the design meeting and more than half of the developer think the design is implementable.

The product owner is taking the same responsibility as business manager and front end mentor. All the technology question, future of the project, distribution of the team is announced by him and the scrum master is bring it up during the daily scrum or sprint planning. As soon as more than half of the team member agree on proceed the suggestion that provided by product owner, the suggestion would become an agenda that would proceed until next daily scrum or sprint planning.

Self-Governance Developer activities within Project management tool:

Each developer must make his own test cases and make sure the product has passed his own test cases before sent out a review. During the developer phase, each developer only allows to check out one task per time. If the story has been interrupted because of waiting for back-end, developer can issue a GitHub issue, set a mock inside the program and check out another task. In front end, we have strict regulation that each task is proceed in each person’s individual branch and task are merged after a pull request that has been proceeded by the mentor of each team. This highly decreases the possibility of merge conflict and duplicate story check out. If a developer has nothing to do, he would first assign as pair-programming. If he consists has nothing to do, product owner or scrum master would force assign a task to him.

The general activity for all team member:

Each team member has been asked to study Self-Governance Developer and Scrum 2 weeks before the project started. Since the project has decided to use Scrum as process tool. Each team member must participate in daily scrum that proceeds from 9.00 to 10.00 and from 13.00 to 13.15. Weekly result and future week planning are proceeding in sprint planning. The team is aiming for continuous delivery so each day after 17.00 there is a new version deployed on the website and each team member is be required to test the product and issue

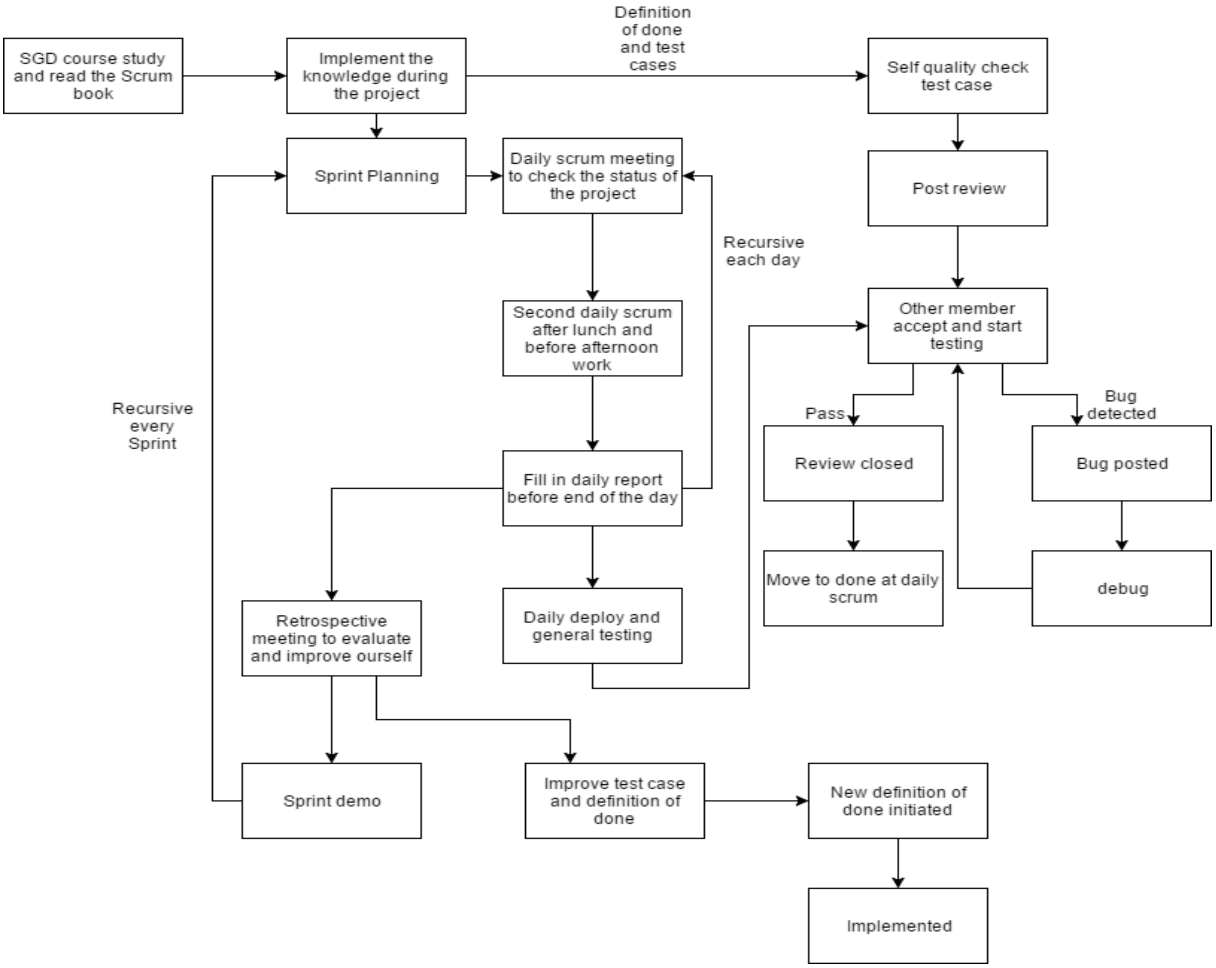
bug report on the daily scrum. After the last sprint, the project team has deployed a final version in joinup.nu and ready for production. Before we announce that this project has been successfully done, the team had one extra meeting to decide the future of the project.

Method

Before all the project group member starts with their work, each of the members had been attending Self-Governance Developer course for around two weeks. Each group member had also been required to read “Scrum and XP from the trenches” beforehand. This book would also be used as comparison data for the effort we spent on each object through the whole project. Since this was the first time for all the members to use either Self-Governance Developer or Scrum in a project, the team always had a physical book that explains Scrum and Self-Governance Developer around during each meeting or process.

The project group checks what each individual did yesterday and what would they do today during each scrum meeting in the morning from 9.00 to 10.00 and from 13.00 to 13.15. Before end of the day, each member had required finishing their daily summary report before they fill in the checkout table and leave the room. The group had been told to check result of a project in weekly basis but the team manages to do daily deployment since all the work has been pass certain quality check with certain test cases before it moves to 'done'.

Evaluation for each individual was quite important since human always learns from their own mistake. The group had to construct retrospective meeting each week before or after



Sprint demo. In the weekly retrospective meeting, each group member had been requiring to explain what was the part they are proud of and the part they feel that can be improved.

Figure 1: How we proceed our project and maintain the health of it

The test cases were also been improved during each retrospective meeting and sprint demo. We managed to improve the test case from poor quality high performance into high-quality but slightly lower performance. With the help of Self-Governance Developer framework and scrum, the team was managed to have a daily base production.

The daily report helps each team member know which part ones might distribute too much effort. This report is using the daily report as main data to reach the goal which is analyzed through following evaluation model:[1]

- Sum of effort spent on each individual activity within the whole project.
  What it means is collect what have each team member done each day and how much time have they spent on it. This makes everybody in the team evaluate themselves daily and help them find out where should they spend more time on and where they should not. It is also helping each individual in the team to know in which day who is doing what afterward.
- Distribution of the effort across activity types on an iteration basis.
  Each activity inside each sprint and each day are looping himself. It means the basic schedule of certain process manage method has been followed. In this case, the project group means the schedule and rules of scrum. Each member should distribute their effort in each activity to help the team make progress. By analyzing this part, it means the project group analyzes in more detail than the first one. The team analyzes through what each person do each day instead each week or summarize it up at the end of the day.
- Differences of effort spent on various activities in various iterations.
  In this part, each team member analyzes themselves in an even more detail level. What did each team member do in each iteration or each big effort? The activity distributes at the beginning of the project are most likely different from the part that closes to the end of the project. Since most of the develop work should be done step by step and most of the time should be moving into testing, paperwork and deliver the code.
- Comparison of distribution of effort spent on Non-developer versus Self-Governance Developer Developer activities.
  Non-developer activity can be designer, business manager, tester and etc. Non-developer act as a really important role inside the project other than Self-Governance Developer Developers. Their effort is most of the time spent on different place than developers. For example, design the User experience of the component, host the sprint planning, test individual component and etc.

[1] Evaluation of My (Our) Individual Project Work (Towards Becoming an Excellent Developer), Mira Kajko-Mattsso,Modern mjukvaruutveckling, IV1303,pp5

- Competency development.
  This part evaluate before and after the project to see how much an individual has been grown. In most of the case, the project group uses Essense kernel tool here and spider map. Inside the project group, ones can also compare himself though each retrospective meeting.
- Usefulness of the Kernel in supporting the project health and progress.

Usefulness in the ESSENCE kernel can mean differently depends on each project group. Kernel help each project group keep their project and product of the project healthy. In another way, it helped each project member monitor if they are doing the right thing and if they are building the right product. The project group is constant checking similar product that using google map Application Programming Interface and define the opportunity from it. The project group also defined the health level of each sprint in each retrospective meeting and improve it during next sprint planning. The project group had also used the quality of the product and GitHub pulse to measure this part.

Project Results

Collect method:

In each day before the project group called off the day each member has been asked to finish one daily report before this member can leave the room and sign out. The data has been collect through four weeks period long and summarize along each week. Each week start from Wednesday since that is when we have our sprint planning and start of a sprint. Sunday and Saturday are excluded from data collection except if any team member has been asked for leave during the week and need to work during the week to make up to it. The unit of the data does not end with time but relative story point. This is not decision of the team but myself. The reason the data has been collected in a relative story point is that it will be much visible and comparable afterward. Nobody will remember how much time each person has spent on individual activity at the end of the day. If we start collection specific time like hours or seconds, it will highly interrupt and distract our work. At same time some work has been done in a relatively short time, for example if design pattern went wrong and we need to initiate an emergency meeting and time is unknown. It will be better to collect the data through relative story points. The time to collect the data has been excluded from the overall data collection. Because the section 'Course 1: Time to fill "My Daily Work"' should not be considered as an overall contribution of the project. If this part is considered inside the data collection, it will create uncertainty and inaccurate to the final data and analyze. That is why in the end of the 'Course 1: Time to fill "My Daily Work"' has been stated not calculated.

Data presentation:

There are two versions of table here that present the same data. The second one is the simplified version of the first one which only shows the total amount through the week.

The data is present *in Appendix Table 1: The raw data from 4 weeks long daily report.*

In the next page, the summarized version of the raw data is presented.

Table 2: The summarized raw data of 4 weeks

Differences of effort spent on various activities in various iterations →	week 4	week 3	week 2	week 1	**End of 4th sprint**
Distribution of the effort across activity types on an iteration basis↓	**Sum of effort spent on each individual activity within the whole project**				
ACTIVITIES THAT I CONDUCT IN THE ROLE OF A STUDENT					
Course 1: Time to fill “My Daily Work"	75	90	90	not calculated	

ACTIVITIES THAT I CONDUCT IN THE ROLE OF A SKILLED GENERALIST (Non-developer)					
Managing Requirements of a Skilled Generalist Role	2645	4610	810	5167	13232
Design of a Skilled Generalist Role	3850	3914	1230	2550	11544
Testing (NON-DEVELOPER Level Testing, acceptance, system, integration) of a Skilled Generalist Role	688	274	420	210	1592
Project management of a Skilled Generalist Role	2510	1110	280	11150	15050
Total for Non-developer role	9693	8908	2740	19077	**41418**

Self Governent Developer ACTIVITIES CONDUCT IN THE ROLE OF A (Pair)/DEVELOPER					
Preliminary Activities	1024	244	282	540	2090
Planning Activities of a Developer Role	9302	15348	3394	2310	30354
Preparatory Activities of a Developer Role	8122	5875	7640	3800	25437
Coding Activities of a Developer Role	5789	8833	7310	3775	25707
Unit Testing Activities of a Developer Role	458	5274	10206	5285	21223
Evaluative Activities of a Developer Role	1521	130	93	220	1964
Debugging Activities of a Developer Role	2153	327	1329	1305	5114
Self-Assessment Activities (Document aside your self-assessment results)	2200	836	664	436	4136
Delivery of a Developer Role	90	100	44	70	304
Total for ACTIVITIES CONDUCT IN THE ROLE OF A (Pair)/DEVELOPER	30659	36867	30962	17741	**116329**
Difference between Non developer and SGD	**74911**				

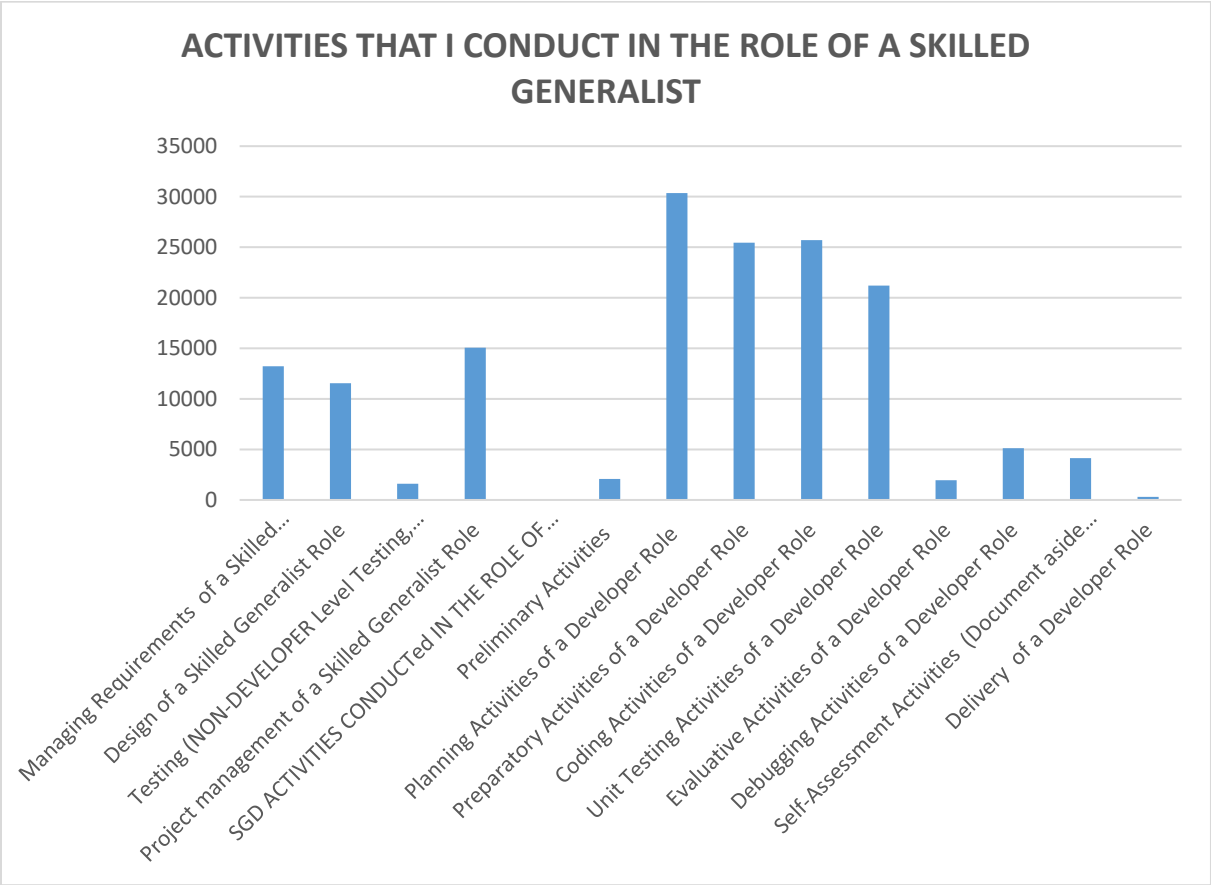
Total effort each week	40352	46875	33702	36818	157747

In the end of the 4th sprint, Managing requirement of a skilled generalist role has 13232 story points , 11544 story points for Design of a Skilled Generalist Role,1592 story points for Testing (NON-DEVELOPER Level Testing, acceptance, system, integration) of a Skilled Generalist Role and 15050 story points for Project management of a Skilled Generalist Role. These conclude all the activity as a non-developer.

As a self governance developer, Preliminary Activities has 2090 story points, 30354 story points for Planning Activities of a Developer Role, 25437 story points for Preparatory Activities of a Developer Role, 25707 story points for Coding Activities of a Developer Role,21223 story points for Unit Testing Activities of a Developer Role, 1964 story point for Evaluative Activities of a Developer Role, 5114 story points for Debugging Activities of a Developer Role, 4136 story points for Self-Assessment Activities and 304 story points for Delivery of a Developer Role.

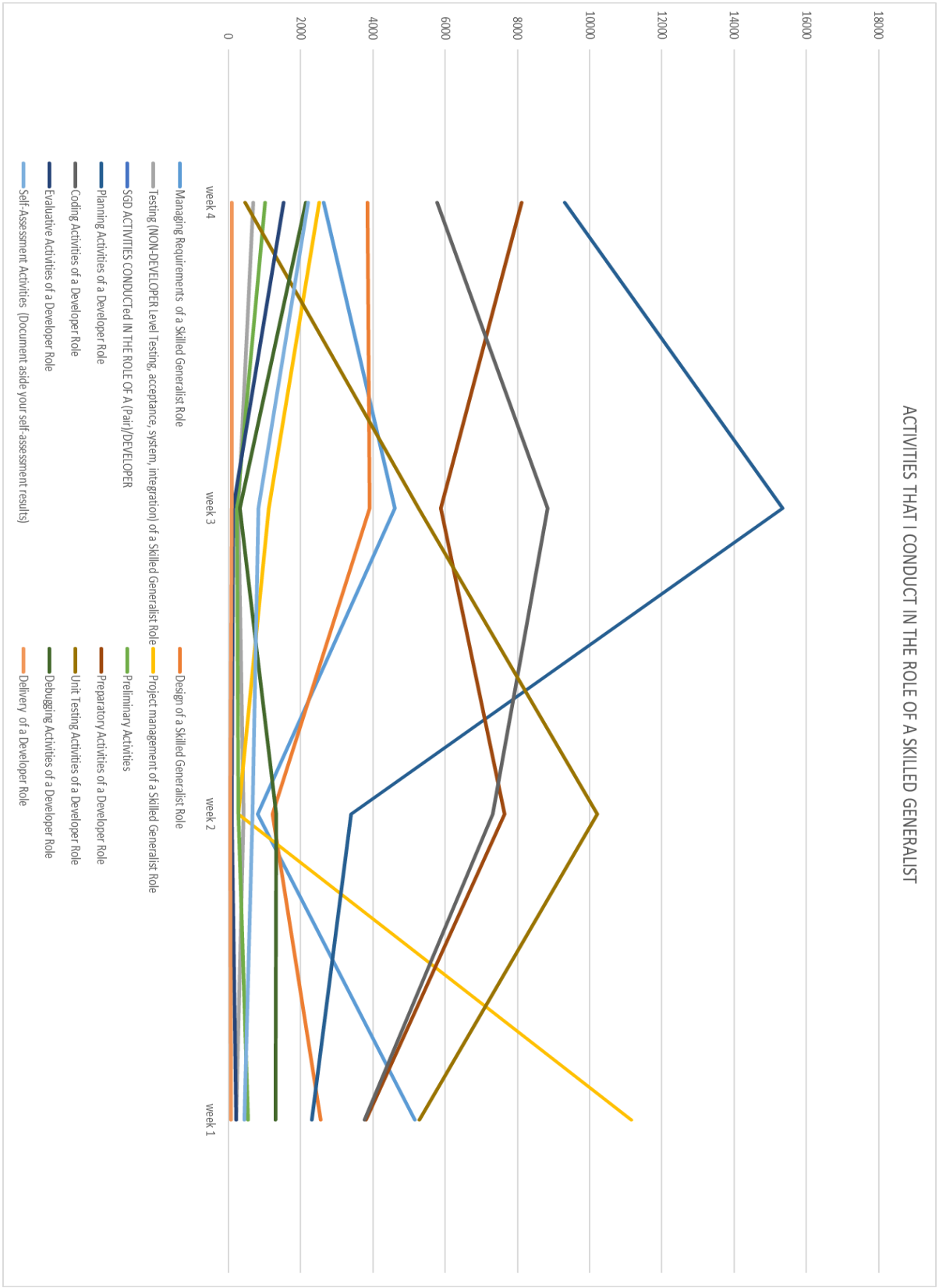
In total, ones have spent around 41418 story points for non-developer activity and for Self-Governance developer activity ones has spent 74911 story point after the 4th sprint. The difference between them is 74911 story points which means one has spent more time as Self-governance developer activity than non-developer activity.

Graph 1: Effort spent during the whole project



Data Analyze

Graph 2: The effort difference during each sprint



The daily report helps each team member knows in which part ones might engage too many time and in another part, ones might spend more time on it. Corresponding the report each team member can improve his daily work schedule. From sum of the effort in each week than comparing what the owner of the daily report into the theoretical time that book said should be used, ones can clearly see which part have been spent more effort on and which are not. Bring up the daily report into the retrospective meeting is a great idea to improve the time planning in next sprint.

Use other tools to combine with daily report for further analyze
- Sum of effort spent on each individual activity within the whole project.
The project group can also estimate the contribution of each individual from this report but it is the only limited on estimation. For accurate calculation about how much effort did someone spent on this project, the project group also need to count the lines of code each individual write and story point one burns.

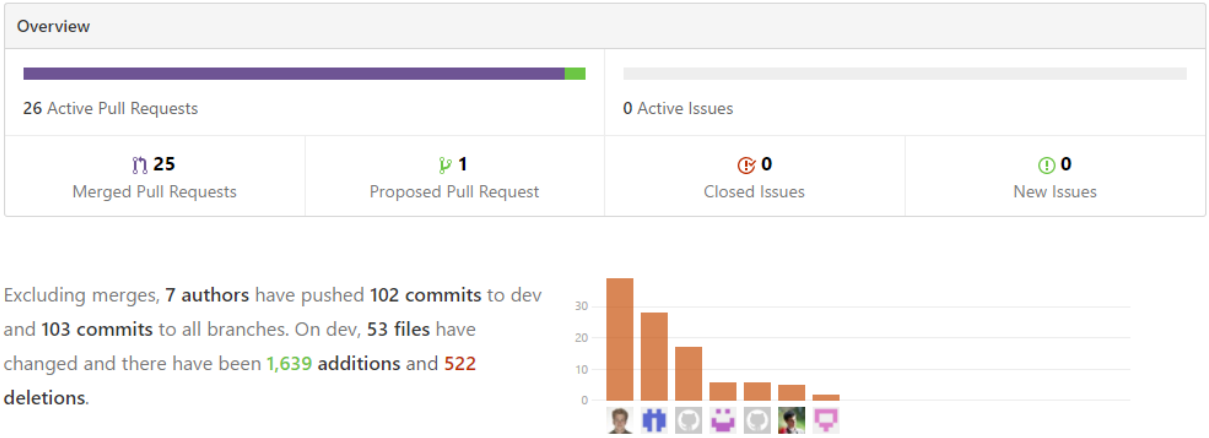


Figure 2: GitHub pulse data at the end of the last sprint

GitHub pulse is a really good tool when it comes to counting the contribution and useful lines of code. In here, ones can see the overall contribution of each team member and how many of them actually take effect on the project. Ones also can see how many issues is happening to the project and how many have the group already been fixed. The project group can also determine the actual contribution according to lines of code though GitHub contribution graph which stated below. The team start has dramatically performance boost (code testing, bug fix and new feature developer work) since 4th of may. This is because of the group introduced the new definition of done and test cases. A lot of bugs has been fixed before the next feature has been done.

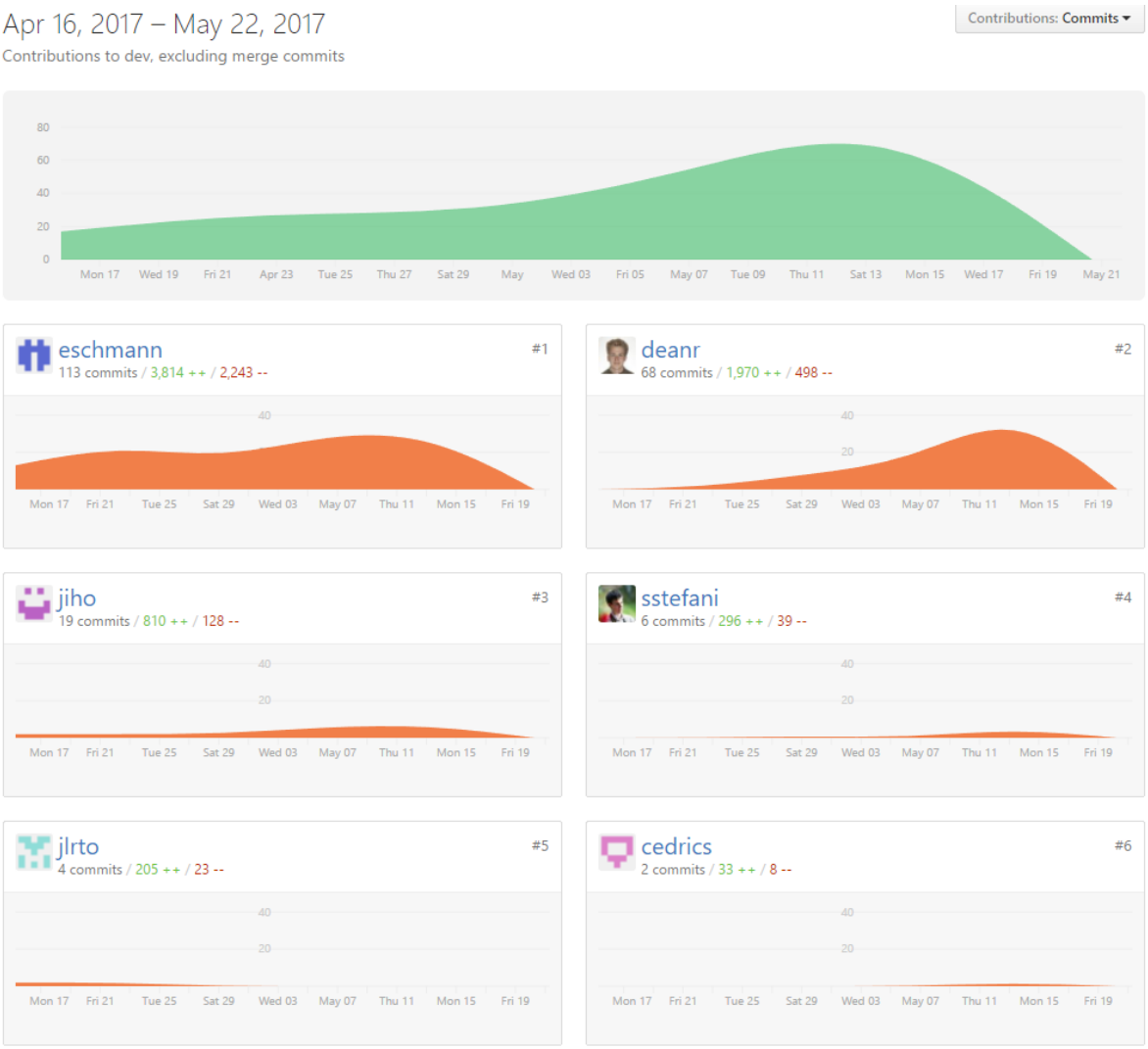


Figure 3: GitHub graph as contribution graph

The reason of difference in effort spent between Non-developer and SGD

- Distribution of the effort across activity types on an iteration basis

Since each sprint has been separate into sprint planning, daily scrum, develop work and testing, the daily report has also certain part that connects to each of them. Depends on the date and plan of the day, some of the parts might be zero. For example, if the team decide to have a sprint planning on Thursday which most of the time it takes whole day, all the coding and developer role is zero and same as all the testing roles. Since one person can only do one role at a time. This makes the project group or each team member easily to see what was happening that day and what was he doing from the daily report.

Result from daily report

From daily report, ones can generally see that the project group has spent so much time on sprint planning and daily scrum. Especially sprint planning, project group have used a whole day to it each week. Sometimes even more. This is the fact that the group has involved too much technical discussion in each sprint planning. During the retrospective meeting, each group member mentioned this issue and suggest for improvement. Nobody wants the next

sprint to suffer since everybody work against the same goal. This situation has been improved through time which can be seen from the graph.

Put project length into consideration

- Differences of effort spent on various activities in various iterations

By the time that the project goes the time distribution of the project starts to differ. This happened because the longer the time goes the less code that each person need to write. It means that more and more people are starting finish with all they can do and have nothing to do. The scrum master and product owner would distribute these people into testing and debug section. These team members start fixing the issues on GitHub. There also another part of team members that start doing documentation work and website update.

After the second sprint, the team realizes the longer times goes on the less developer work remain. At same time, the documentation and debug work start increasing. This due to loads of bug that team created before when the group has bad definition of done and test cases. All these bugs start stacking together and create huge issues. From the graph ones can see at week two there is huge amount of testing going on and it starts decreasing though the time. That is the time the team starts to implement the new test cases and definition of done. This can be seen from the overall developer role performance from the graph. The developer role effort has been incredibly decreased since last two sprints and the reason is that the group added more and more detailed test case and definition of done. This is a huge trade off but the team does manage to create the continuous delivery. Each line of code is quality ensured.

The team decides to let designer start doing some coding during the last one and a half sprint because all the design has been done and the dedicated designer needs to do something else to kill the time. As this essay mentioned above, our design work happened parallel with the developing work but coding still takes longer time than designing. When the times goes, designer has nothing to design. This also due to most of the developer work role has been replaced by tester and stuff that can be used within elementUI. Since our product owner also takes the role as mentor for the front-end team, the effort that product owner spent is as same as all other developers.

Competency development and method

The project team does not have a certain way to measure the competency but during each retrospective meeting, each member has time to evaluate the skill within their role in that week. Before the project group starts with the projects, the product owner has done a basic knowledge check for each skill level are all the team members in. This makes the team realize that most of the team member who takes developer role in front end was under the assists level because Javascript is not inside ICT School syllabus. In this case for front end team started to evaluate the skill level from the second sprint after everybody got their hands-on JavaScript and can play around it a bit.

Table 8. The Generic Competency Levels

Level 1 Assists	Level 2 Applies	Level 3 Masters	Level 4 Adapts	Level 5 Innovates
❑ Has a basic understanding of the concepts ❑ Is able to act in a professional manner ❑ Is able to correctly respond to basic questions within his/her domain ❑ Is able to perform most basic functions within the domain ❑ Is able to follow instructions and complete basic tasks ❑ Is able to perform tasks under supervision	❑ Is able to collaborate within the team ❑ Is able to satisfy routine demands and simple work requirements ❑ Can handle simple challenges with confidence ❑ Is able to perform tasks under minimal supervision ❑ Can handle simple work requirements but needs guidance in handling any complications or difficulties ❑ Is able to reason about the context and draw sensible conclusions	❑ Is able to satisfy most demands and work requirements ❑ Is able to speak the domain language with ease and accuracy ❑ Is able to communicate and explain his/her work ❑ Is able to give and receive constructive feedback ❑ Knows the limits of his/her capability and when to call on more expert advice. ❑ Works at a professional level with little or no guidance.	❑ Is able to satisfy complex demands and work requirements ❑ Is able to communicate with others working outside the domain ❑ Can direct and help others working within the domain ❑ Is able to adapt his/her way of working to work well with others, both inside and outside their domain	❑ Has many years of experience and is currently up to date in what is happening within the domain ❑ Is recognized as an expert by peers ❑ Supports others in working on a complex professional level ❑ Knows when to innovate or do something different and when to follow normal procedure ❑ Develops innovative and effective solutions to the current challenges within the domain

Figure 4: The Generic Competency Levels from ESSENCE Kernel – Quick Reference Guide[2]

Improvement though each sprint and method

- Usefulness of the Kernel in supporting the project health and progress

During each retrospective meeting, the scrum master and team members realize the improvement of their skill base on the ESSENCE The Generic Competency table. After project has been done each member's competency level is somehow between applies and masters. This causes the overall performance and the innovation of the project increased dramatically. Overall the time has been spent less and less on developer role but more on planning. This is because of the story point has started burning so fast and the group needs to put new stories on the scrum board. To be able to do it the team need a new sprint planning meeting.

The ESSENCE kernel has been helping the project team to maintain the health and progress through the entire project. In some cases, ESSENCE is started helping the team before the project start.

[2] ESSENCE Kernel – Quick Reference Guide, Version 0.3, SEMAT Inc, pp 22

Figure 5. Opportunity: Overview and alpha state cards[3]

With the help from ESSENCE Opportunity alpha state cards, the team manages to identify the problem and defined the solution of the problem then propose a valuable idea for the project. These standards have been stick with the team through the entire project. The team constant identifies the opportunity and value of the project then make decision from it. In this case, the team can define the meaning of usefulness as maintain the value of the project and make sure the project went as planned from product point of view and team point of view. By means of product, the team has constantly refactored the code and make sure the architecture of the code stay the same in no matter what reason. By means of team, the team has a healthy working environment and constantly collaboration should happen between each team member.

[3] ESSENCE Kernel – Quick Reference Guide, Version 0.3, SEMAT Inc, pp 9

To be able to obtain the contribution level of the project team member and have an insight of how each individual developer work is distributed across the project. The project group has use external tool within Version Control System and combine it with existence daily report. This helps us to have an insight of Lines of Codes(LOC). Though the entire project, the project group has actively use retrospective meeting to improve the efficiency of the project and correct the mistake that has been done in the last sprint. From the result of each retrospective meeting, the team has also adjusted the non-developer role and Self-Governance Developer role to make sure the effort is balance distributed and maintain the stability of the production. Because of team actively use the ESSENCE kernel, the quality and value of the project have been insured and the project is not outdated when the team release the final product.

Conclusion and Evaluation

To be able to solve the problem that the project team has no insight into how each individual developer work is distributed across the project the project group needs to find out what the effort is being spent on, evaluate the distribution of the developer activities across the whole project. The team needs to find the theoretical time or the time that professionals are using. Compare them and come to a percentage estimation. The team can count the lines of code combined with the time that has been spent on these codes. To be able to extract these data ones can determine from the daily report, GitHub pulse and GitHub graph. Look at how much hours have ones spent on his Self-Governance Developer developer role and find the time on Github pulse and graph on the data that person looking at. To combine them and determine the performance of that team member though that specific timing, the team has to combine daily report with the pulse on GitHub during the effort analyze process. This is because of what can be happened is ones spend 6 hours on 2 lines of efficiency code.

In order to evaluate if the project group is constantly doing the right thing with right time used, the project group has used the book "Scrum and XP from the trenches" as comparison data.By the data that been stated in the book "Scrum and XP from the trenches" each sprint planning should be around 1 hours and same as the daily scrum. In this case, the team has excessed the theoretical time a lot since the team has spent 8 hours per sprint planning. Otherwise, the team should perform quite well during this 4 sprint. The team has a 1-hour daily scrum plus 15 minutes after lunch, the burndown chart has under the average line almost every day and takes 1 to 2 hours per sprint for retrospective meeting which all of these are as same as the book stated. [4]

[4] Scrum and XP from the trenches,2nd Edition- Director's Cut,InfoQ,Henrik Knibetg

Bibliography

[1]M. Kajko-Mattsso, "Evaluation of My (Our) Individual Project Work (Towards Becoming an Excellent Developer)", Modern mjukvaruutveckling, IV1303, pp5, 2017.

[2,3]I. Spence, C. Peraire, M. A.V Nelson, B. Palank, M. M KajkoMattson, W. Menezes, B. Myburgh, A. Bayda and P. E. McMahon, "ESSENCE Kernel – Quick Reference Guide", no. 03, pp. 9,22, 2017.

[4]H. Kniberg, M. Cohn and J. Sutherland, Scrum and XP from the Trenches, 1st ed. C4Media, 2015.

Appendix

Table 1: The raw data from 4 weeks long daily report

Activities	**Week 4**										
ACTIVITIES THAT I CONDUCT IN THE ROLE OF A STUDENT	Mon	People	Tue	People	Wed	People	Wed	People	Fri	People	**Total**
Course 1: Time to fill “My Daily Work"	10	1	15	1	15	1	15	1	15	1	**75**
ACTIVITIES THAT I CONDUCT IN THE ROLE OF A SKILLED GENERALIST											
Managing Requirements of a Skilled Generalist Role											2645
REQ 1: Identify requirements	10	5	50	4	15	5	150	5	20	5	**269**
REQ 2: Analyze requirements	10	5	50	4	15	5	150	5	20	5	**264**
REQ 3: Change requirements	10	5	50	4	15	5	150	5	20	5	**264**
REQ 3.1: Identify impact of change (time it takes to identify changes everywhere in the system)	10	5	50	4	15	5	150	5	20	5	**264**
REQ 3.2: Make change(s) (time it takes to make changes everywhere in the system)	10	5	50	4	15	5	150	5	20	5	**264**
REQ 4: Plan requirements	10	5	50	4	15	5	150	5	20	5	**264**
REQ 4.1: Estimate effort	10	5	50	4	15	5	150	5	20	5	**264**
REQ 4.2: Prioritize requirements	10	5	50	4	15	5	150	5	20	5	**264**
REQ 4.3: Other planning activities related to requirements (specify)	10	5	50	4	15	5	150	5	20	5	**264**
REQ 5: Other, specify	10	5	50	4	15	5	150	5	20	5	**264**
Design of a Skilled Generalist Role											
DES 1: Design system or system component (high-level design)'	0	10	100	10	340	10	150	10	10	10	**650**
REQ 2: Analyze requirements	0	10	100	10	340	10	150	10	10	10	**640**
DES 3: Change design	0	10	100	10	340	10	150	10	10	10	**640**
DES 3.1: Identify impact of change (time it takes to identify changes everywhere in the system)	0	10	100	10	340	10	150	10	10	10	**640**
DES 3.2: Make change(s) (time it takes to make changes everywhere in the system)	0	10	100	10	340	10	150	10	10	10	**640**
DES 4: Other, specify	0	10	100	10	340	10	150	10	10	10	**640**

Testing (NON-DEVELOPER Level Testing, acceptance, system, integration) of a Skilled Generalist Role											
TEST 1: Define tests	0	10	50	3	10	10	20	10	0	10	**123**
TEST 2: Analyze tests	0	10	50	3	10	10	20	10	0	10	**113**
TEST 3: Change tests	0	10	50	3	10	10	20	10	0	10	**113**
TEST 3.1: Identify impact of change (time it takes to identify changes everywhere in the system)	0	10	50	3	10	10	20	10	0	10	**113**
TEST 3.2: Make change(s) to tests (time it takes to make changes everywhere in the system)	0	10	50	3	10	10	20	10	0	10	**113**
TEST 4: Other, specify	0	10	50	3	10	10	20	10	0	10	**113**
Project management of a Skilled Generalist Role											
PROJ-MAN 1: Plan project/part of project (iteration)	0	10	0	10	60	10	150	10	0	10	**260**
PROJ-MAN 2: Analyze project/ project plan /part of project (iteration)	0	10	0	10	60	10	150	10	0	10	**250**
PROJ-MAN 3: Change project plan /part of project plan (iteration)	0	10	0	10	60	10	150	10	0	10	**250**
PROJ-MAN 4: Evaluate your project work	0	10	0	10	60	10	150	10	0	10	**250**
PROJ-MAN 5: Manage risks	0	10	0	10	60	10	150	10	0	10	**250**
PROJ-MAN 5.1: Identify risks	0	10	0	10	60	10	150	10	0	10	**250**
PROJ-MAN 5.2: Analyze risks	0	10	0	10	60	10	150	10	0	10	**250**
PROJ-MAN 5.3: Manage risks	0	10	0	10	60	10	150	10	0	10	**250**
PROJ-MAN 6: Customer-related activities, specify	0	10	0	10	60	10	150	10	0	10	**250**
ROJ-MAN 7: Other, specify	0	10	0	10	60	10	150	10	0	10	**250**
Self Governent Developer ACTIVITIES CONDUCTed IN THE ROLE OF A (Pair)/DEVELOPER											
Preliminary Activities											
PR- 1: Review and agree on the overall or part of the project plan.	0	10	15	4	60	10	60	10	0	10	**179**
PR-2: Revise and ensure that the technology to be used is tested and understood.	0	10	15	4	60	10	60	10	0	10	**169**
PR-3: Revise and understand any appropriate internal (organizational) and external standard(s).	0	10	15	4	60	10	60	10	0	10	**169**
PR-4: Learn/relearn the organizational implementation and unit (developer) testing way of working.	0	10	15	4	60	10	60	10	0	10	**169**

PR-5: Review and revise your personal implementation and unit (developer) testing way of working.	0	10	15	4	60	10	60	10	0	10	**169**
PR-6: Other, specify	0	10	15	4	60	10	60	10	0	10	**169**
Planning Activities of a Developer Role											
PL-1: Review the requirement(s) for the unit(s) to be developed.	0	10	300	3	0	10	300	10	30	10	**673**
PL-2: Prepare (make) and/or review the design specification(s) for the unit(s) to be developed.	0	10	300	3	0	10	300	10	30	10	**663**
PL-3: Resolve unclear questions and uncertainties.	0	10	300	3	0	10	300	10	30	10	**663**
PL-4: Determine and document your implementation and unit (developer) testing goals.	0	10	300	3	10	10	300	10	30	10	**673**
PL-5: Determine your implementation and unit (developer) testing strategy.	0	10	300	3	0	10	300	10	30	10	**663**
PL-6: Determine appropriate implementation and testing practices.	0	10	300	3	0	10	300	10	30	10	**663**
PL-7: Identify standards to be used for meeting your goals.	0	10	300	3	0	10	300	10	30	10	**663**
PL-8: Set your own personal deadlines to be met during your implementation and unit (developer) testing work.	0	10	300	3	0	10	300	10	30	10	**663**
PL-9: Estimate effort and resources required for carrying out your work.	0	10	300	3	0	10	300	10	30	10	**663**
PL-10: Schedule your work.	0	10	300	3	0	10	300	10	30	10	**663**
PL-11: Review your implementation and unit (developer) testing plan to ensure that it is realistic and achievable.	0	10	300	3	0	10	300	10	30	10	**663**
PL-12: Identify risks related to your plan.	0	10	300	3	0	10	300	10	30	10	**663**
PL-13: Plan for managing any identified risks.	0	10	300	3	0	10	300	10	30	10	**663**
PL-14: Other, specify	0	10	300	3	0	10	300	10	30	10	**663**
Preparatory Activities of a Developer Role											
P-1: Prepare(make) and/or review your low-level design(s) of the code to be written or changed.	150	2	360	3	340	5	100	5	50	2	**1017**

P-2: Prepare (make) an impact analysis of your low-level design(s).	150	2	360	3	340	5	100	5	50	2	**1015**
P-3: Determine the types of unit (developer) test cases and their order.	150	2	360	3	340	5	100	5	50	2	**1015**
P-4: Create and/or revise your unit (developer) test case base.	150	2	360	3	340	5	100	5	50	2	**1015**
P-5: Revise the existing unit (developer) regression test base, if relevant.	150	2	360	3	340	5	100	5	50	2	**1015**
P-6: Create or modify stubs and drivers, if required.	150	2	360	3	340	5	100	5	50	2	**1015**
P-7: Prepare your unit (developer) testing environment and check whether it is appropriate for you work.	150	2	360	3	340	5	100	5	50	2	**1015**
P-8: Other, specify	150	2	360	3	340	5	100	5	50	2	**1015**
Coding Activities of a Developer Role											
C-1: Write/rewrite your code.	200	4	150	3	340	5	150	5	300	4	**1161**
C-2: Compile/ recompile your code as required.	200	4	150	3	340	5	150	5	300	4	**1157**
C-3: Make notes on your compilation errors, if necessary.	200	4	150	3	340	5	150	5	300	4	**1157**
C-4: Make notes on your defects	200	4	150	3	340	5	150	5	300	4	**1157**
C-5: Other, specify	200	4	150	3	340	5	150	5	300	4	**1157**
Unit Testing Activities of a Developer Role											
T-1: Check whether the unit (developer) test case base meets the given requirements and design.	0	10	30	4	0	10	0	10	0	10	**74**
T-2: Check whether the unit (developer) regression test base meets the given requirements and design.	0	10	30	4	0	10	0	10	0	10	**64**
T-3: Remedy requirements problems in your unit (developer) regression and/or test cases base, if any.	0	10	30	4	0	10	0	10	0	10	**64**
T-4: Perform dynamic testing by executing code.	0	10	30	4	0	10	0	10	0	10	**64**
T-5: Perform static (human) testing by reviewing your code.	0	10	30	4	0	10	0	10	0	10	**64**
T-6: Record/write down test results.	0	10	30	4	0	10	0	10	0	10	**64**
T-7: Other, specify	0	10	30	4	0	10	0	10	0	10	**64**

Evaluative Activities of a Developer Role											
E-1: Analyze your unit (developer) testing results.	50	3	30	3	300	10	0	10	100	3	**509**
E-2: Depending on the unit (developer) testing results, determine your next step(s).	50	3	30	3	300	10	0	10	100	3	**506**
E-3: Other, specify	50	3	30	3	300	10	0	10	100	3	**506**
Debugging Activities of a Developer Role											
D-1: Identify the source of (an) error(s).	100	2	150	5	150	5	150	5	150	2	**719**
D-2: Determine solution(s) for eliminating the sources of error(s).	100	2	150	5	150	5	150	5	150	2	**717**
D-3: Other, specify	100	2	150	5	150	5	150	5	150	2	**717**
Self-Assessment Activities (Document aside your self-assessment results)											
A-1: Assess your own development work.	80	4	150	5	150	5	0	5	150	4	**553**
A-2: Identify causes of your mistakes.	80	4	150	5	150	5	0	5	150	4	**549**
A-3: Identify improvement areas in your own way of working.	80	4	150	5	150	5	0	5	150	4	**549**
A-4: Other, specify	80	4	150	5	150	5	0	5	150	4	**549**
Delivery of a Developer Role											
S-2: Deliver your code.	0	10	0	10	0	10	0	10	0	10	**50**
S-3: Other, specify	0	10	0	10	0	10	0	10	0	10	**40**
	* I took a half day off because i have a doctor appointment										
Activities	**Week 3**										
ACTIVITIES THAT I CONDUCT IN THE ROLE OF A STUDENT	Mon	People	Tue	People	Wed	People	Wed	People	Fri	People	**Total**
Course 1: Time to fill “My Daily Work"	10	10	10	10	10	10	10	10	10	10	**90**
ACTIVITIES THAT I CONDUCT IN THE ROLE OF A SKILLED GENERALIST											
Managing Requirements of a Skilled Generalist Role											4610
REQ 1: Identify requirements	0	10	10	10	200	10	200	10	10	10	**470**
REQ 2: Analyze requirements	0	10	10	10	200	10	200	10	10	10	**460**
REQ 3: Change requirements	0	10	10	10	200	10	200	10	10	10	**460**
REQ 3.1: Identify impact of change (time it takes to identify changes everywhere in the system)	0	10	10	10	200	10	200	10	10	10	**460**

REQ 3.2: Make change(s) (time it takes to make changes everywhere in the system)	0	10	10	10	200	10	200	10	10	10	**460**
REQ 4: Plan requirements	0	10	10	10	200	10	200	10	10	10	**460**
REQ 4.1: Estimate effort	0	10	10	10	200	10	200	10	10	10	**460**
REQ 4.2: Prioritize requirements	0	10	10	10	200	10	200	10	10	10	**460**
REQ 4.3: Other planning activities related to requirements (specify)	0	10	10	10	200	10	200	10	10	10	**460**
REQ 5: Other, specify	0	10	10	10	200	10	200	10	10	10	**460**
Design of a Skilled Generalist Role											
DES 1: Design system or system component (high-level design)'	340	2	120	2	30	4	30	4	120	2	**654**
REQ 2: Analyze requirements	340	2	120	2	30	4	30	4	120	2	**652**
DES 3: Change design	340	2	120	2	30	4	30	4	120	2	**652**
DES 3.1: Identify impact of change (time it takes to identify changes everywhere in the system)	340	2	120	2	30	4	30	4	120	2	**652**
DES 3.2: Make change(s) (time it takes to make changes everywhere in the system)	340	2	120	2	30	4	30	4	120	2	**652**
DES 4: Other, specify	340	2	120	2	30	4	30	4	120	2	**652**
Testing (NON-DEVELOPER Level Testing, acceptance, system, integration) of a Skilled Generalist Role											
TEST 1: Define tests	0	10	0	10	2	10	2	10	0	10	**54**
TEST 2: Analyze tests	0	10	0	10	2	10	2	10	0	10	**44**
TEST 3: Change tests	0	10	0	10	2	10	2	10	0	10	**44**
TEST 3.1: Identify impact of change (time it takes to identify changes everywhere in the system)	0	10	0	10	2	10	2	10	0	10	**44**
TEST 3.2: Make change(s) to tests (time it takes to make changes everywhere in the system)	0	10	0	10	2	10	2	10	0	10	**44**
TEST 4: Other, specify	0	10	0	10	2	10	2	10	0	10	**44**
Project management of a Skilled Generalist Role											
PROJ-MAN 1: Plan project/part of project (iteration)	0	10	20	10	15	10	15	10	20	10	**120**
PROJ-MAN 2: Analyze project/ project plan /part of project (iteration)	0	10	20	10	15	10	15	10	20	10	**110**
PROJ-MAN 3: Change project plan /part of project plan (iteration)	0	10	20	10	15	10	15	10	20	10	**110**

PROJ-MAN 4: Evaluate your project work	0	10	20	10	15	10	15	10	20	10	**110**
PROJ-MAN 5: Manage risks	0	10	20	10	15	10	15	10	20	10	**110**
PROJ-MAN 5.1: Identify risks	0	10	20	10	15	10	15	10	20	10	**110**
PROJ-MAN 5.2: Analyze risks	0	10	20	10	15	10	15	10	20	10	**110**
PROJ-MAN 5.3: Manage risks	0	10	20	10	15	10	15	10	20	10	**110**
PROJ-MAN 6: Customer-related activities, specify	0	10	20	10	15	10	15	10	20	10	**110**
ROJ-MAN 7: Other, specify	0	10	20	10	15	10	15	10	20	10	**110**
Self Govenent Developer ACTIVITIES CONDUCTed IN THE ROLE OF A (Pair)/DEVELOPER											
Preliminary Activities											
PR- 1: Review and agree on the overall or part of the project plan.	3	10	3	10	2	3	2	3	3	10	**49**
PR-2: Revise and ensure that the technology to be used is tested and understood.	3	10	3	10	2	3	2	3	3	10	**39**
PR-3: Revise and understand any appropriate internal (organizational) and external standard(s).	3	10	3	10	2	3	2	3	3	10	**39**
PR-4: Learn/relearn the organizational implementation and unit (developer) testing way of working.	3	10	3	10	2	3	2	3	3	10	**39**
PR-5: Review and revise your personal implementation and unit (developer) testing way of working.	3	10	3	10	2	3	2	3	3	10	**39**
PR-6: Other, specify	3	10	3	10	2	3	2	3	3	10	**39**
Planning Activities of a Developer Role											
PL-1: Review the requirement(s) for the unit(s) to be developed.	340	4	340	4	30	4	30	4	340	4	**1100**
PL-2: Prepare (make) and/or review the design specification(s) for the unit(s) to be developed.	340	4	340	4	30	4	30	4	340	4	**1096**
PL-3: Resolve unclear questions and uncertainties.	340	4	340	4	30	4	30	4	340	4	**1096**
PL-4: Determine and document your implementation and unit (developer) testing goals.	340	4	340	4	30	4	30	4	340	4	**1096**
PL-5: Determine your implementation and unit (developer) testing strategy.	340	4	340	4	30	4	30	4	340	4	**1096**
PL-6: Determine appropriate implementation and testing practices.	340	4	340	4	30	4	30	4	340	4	**1096**

PL-7: Identify standards to be used for meeting your goals.	340	4	340	4	30	4	30	4	340	4	**1096**
PL-8: Set your own personal deadlines to be met during your implementation and unit (developer) testing work.	340	4	340	4	30	4	30	4	340	4	**1096**
PL-9: Estimate effort and resources required for carrying out your work.	340	4	340	4	30	4	30	4	340	4	**1096**
PL-10: Schedule your work.	340	4	340	4	30	4	30	4	340	4	**1096**
PL-11: Review your implementation and unit (developer) testing plan to ensure that it is realistic and achievable.	340	4	340	4	30	4	30	4	340	4	**1096**
PL-12: Identify risks related to your plan.	340	4	340	4	30	4	30	4	340	4	**1096**
PL-13: Plan for managing any identified risks.	340	4	340	4	30	4	30	4	340	4	**1096**
PL-14: Other, specify	340	4	340	4	30	4	30	4	340	4	**1096**
Preparatory Activities of a Developer Role											
P-1: Prepare(make) and/or review your low-level design(s) of the code to be written or changed.	120	3	100	3	200	4	200	4	100	3	**737**
P-2: Prepare (make) an impact analysis of your low-level design(s).	120	3	100	3	200	4	200	4	100	3	**734**
P-3: Determine the types of unit (developer) test cases and their order.	120	3	100	3	200	4	200	4	100	3	**734**
P-4: Create and/or revise your unit (developer) test case base.	120	3	100	3	200	4	200	4	100	3	**734**
P-5: Revise the existing unit (developer) regression test base, if relevant.	120	3	100	3	200	4	200	4	100	3	**734**
P-6: Create or modify stubs and drivers, if required.	120	3	100	3	200	4	200	4	100	3	**734**
P-7: Prepare your unit (developer) testing environment and check whether it is appropriate for you work.	120	3	100	3	200	4	200	4	100	3	**734**
P-8: Other, specify	120	3	100	3	200	4	200	4	100	3	**734**
Coding Activities of a Developer Role											
C-1: Write/rewrite your code.	340	3	340	3	360	10	360	10	340	3	**1769**
C-2: Compile/ recompile your code as required.	340	3	340	3	360	10	360	10	340	3	**1766**

C-3: Make notes on your compilation errors, if necessary.	340	3	340	3	360	10	360	10	340	3	**1766**
C-4: Make notes on your defects	340	3	340	3	360	10	360	10	340	3	**1766**
C-5: Other, specify	340	3	340	3	360	10	360	10	340	3	**1766**
Unit Testing Activities of a Developer Role											
T-1: Check whether the unit (developer) test case base meets the given requirements and design.	0	10	0	10	360	6	360	6	0	10	**762**
T-2: Check whether the unit (developer) regression test base meets the given requirements and design.	0	10	0	10	360	6	360	6	0	10	**752**
T-3: Remedy requirements problems in your unit (developer) regression and/or test cases base, if any.	0	10	0	10	360	6	360	6	0	10	**752**
T-4: Perform dynamic testing by executing code.	0	10	0	10	360	6	360	6	0	10	**752**
T-5: Perform static (human) testing by reviewing your code.	0	10	0	10	360	6	360	6	0	10	**752**
T-6: Record/write down test results.	0	10	0	10	360	6	360	6	0	10	**752**
T-7: Other, specify	0	10	0	10	360	6	360	6	0	10	**752**
Evaluative Activities of a Developer Role											
E-1: Analyze your unit (developer) testing results.	0	10	0	10	0	10	0	10	0	10	**50**
E-2: Depending on the unit (developer) testing results, determine your next step(s).	0	10	0	10	0	10	0	10	0	10	**40**
E-3: Other, specify	0	10	0	10	0	10	0	10	0	10	**40**
Debugging Activities of a Developer Role											
D-1: Identify the source of (an) error(s).	20	3	20	3	15	6	15	6	20	3	**111**
D-2: Determine solution(s) for eliminating the sources of error(s).	20	3	20	3	15	6	15	6	20	3	**108**
D-3: Other, specify	20	3	20	3	15	6	15	6	20	3	**108**
Self-Assessment Activities (Document aside your self-assessment results)											
A-1: Assess your own development work.	37	6	37	6	37	6	37	6	37	6	**209**
A-2: Identify causes of your mistakes.	37	6	37	6	37	6	37	6	37	6	**209**
A-3: Identify improvement areas in your own way of working.	37	6	37	6	37	6	37	6	37	6	**209**

A-4: Other, specify	37	6	37	6	37	6	37	6	37	6	**209**
Delivery of a Developer Role											
S-2: Deliver your code.	10	10	0	10	0	10	0	10	0	10	**50**
S-3: Other, specify	10	10	0	10	0	10	0	10	0	10	**50**
Activities	**Week 2**										
ACTIVITIES THAT I CONDUCT IN THE ROLE OF A STUDENT	Mon	People	Tue	People	Wed	People	Wed	People	Fri	People	**Total**
Course 1: Time to fill “My Daily Work"	10	10	10	10	10	10	10	10	10	10	**90**
ACTIVITIES THAT I CONDUCT IN THE ROLE OF A SKILLED GENERALIST											
Managing Requirements of a Skilled Generalist Role											810
REQ 1: Identify requirements	vacation	vacation	16	10	0	10	0	10	35	10	**81**
REQ 2: Analyze requirements	vacation	vacation	16	10	0	10	0	10	35	10	**81**
REQ 3: Change requirements	vacation	vacation	16	10	0	10	0	10	35	10	**81**
REQ 3.1: Identify impact of change (time it takes to identify changes everywhere in the system)	vacation	vacation	16	10	0	10	0	10	35	10	**81**
REQ 3.2: Make change(s) (time it takes to make changes everywhere in the system)	vacation	vacation	16	10	0	10	0	10	35	10	**81**
REQ 4: Plan requirements	vacation	vacation	16	10	0	10	0	10	35	10	**81**
REQ 4.1: Estimate effort	vacation	vacation	16	10	0	10	0	10	35	10	**81**
REQ 4.2: Prioritize requirements	vacation	vacation	16	10	0	10	0	10	35	10	**81**
REQ 4.3: Other planning activities related to requirements (specify)	vacation	vacation	16	10	0	10	0	10	35	10	**81**
REQ 5: Other, specify	vacation	vacation	16	10	0	10	0	10	35	10	**81**
	vacation	vacation									
Design of a Skilled Generalist Role	vacation	vacation									1230
DES 1: Design system or system component (high-level design)'	vacation	vacation	56	4	50	5	50	5	60	4	**230**
REQ 2: Analyze requirements	vacation	vacation	56	4	50	5	50	5	30	4	**200**
DES 3: Change design	vacation	vacation	56	4	50	5	50	5	30	4	**200**
DES 3.1: Identify impact of change (time it takes to identify changes everywhere in the system)	vacation	vacation	56	4	50	5	50	5	30	4	**200**

DES 3.2: Make change(s) (time it takes to make changes everywhere in the system)	vacation	vacation	56	4	50	5	50	5	30	4	**200**
DES 4: Other, specify	vacation	vacation	56	4	50	5	50	5	30	4	**200**
	vacation	vacation									
Testing (NON-DEVELOPER Level Testing, acceptance, system, integration) of a Skilled Generalist Role	vacation	vacation									**420**
TEST 1: Define tests	vacation	vacation	15	10	0	5	0	5	35	10	**70**
TEST 2: Analyze tests	vacation	vacation	15	10	0	5	0	5	35	10	**70**
TEST 3: Change tests	vacation	vacation	15	10	0	5	0	5	35	10	**70**
TEST 3.1: Identify impact of change (time it takes to identify changes everywhere in the system)	vacation	vacation	15	10	0	5	0	5	35	10	**70**
TEST 3.2: Make change(s) to tests (time it takes to make changes everywhere in the system)	vacation	vacation	15	10	0	5	0	5	35	10	**70**
TEST 4: Other, specify	vacation	vacation	15	10	0	5	0	5	35	10	**70**
	vacation	vacation									
Project management of a Skilled Generalist Role	vacation	vacation									**280**
PROJ-MAN 1: Plan project/part of project (iteration)	vacation	vacation	3	10	0	5	0	5	5	10	**28**
PROJ-MAN 2: Analyze project/ project plan /part of project (iteration)	vacation	vacation	3	10	0	5	0	5	5	10	**28**
PROJ-MAN 3: Change project plan /part of project plan (iteration)	vacation	vacation	3	10	0	5	0	5	5	10	**28**
PROJ-MAN 4: Evaluate your project work	vacation	vacation	3	10	0	5	0	5	5	10	**28**
PROJ-MAN 5: Manage risks	vacation	vacation	3	10	0	5	0	5	5	10	**28**
PROJ-MAN 5.1: Identify risks	vacation	vacation	3	10	0	5	0	5	5	10	**28**
PROJ-MAN 5.2: Analyze risks	vacation	vacation	3	10	0	5	0	5	5	10	**28**
PROJ-MAN 5.3: Manage risks	vacation	vacation	3	10	0	5	0	5	5	10	**28**
PROJ-MAN 6: Customer-related activities, specify	vacation	vacation	3	10	0	5	0	5	5	10	**28**
ROJ-MAN 7: Other, specify	vacation	vacation	3	10	0	5	0	5	5	10	**28**
	vacation	vacation									
Self Governent Developer ACTIVITIES CONDUCTed IN	vacation	vacation									

THE ROLE OF A (Pair)/DEVELOPER											
Preliminary Activities	vacation	vacation									282
PR- 1: Review and agree on the overall or part of the project plan.	vacation	vacation	2	3	10	10	10	10	2	3	**47**
PR-2: Revise and ensure that the technology to be used is tested and understood.	vacation	vacation	2	3	10	10	10	10	2	3	**47**
PR-3: Revise and understand any appropriate internal (organizational) and external standard(s).	vacation	vacation	2	3	10	10	10	10	2	3	**47**
PR-4: Learn/relearn the organizational implementation and unit (developer) testing way of working.	vacation	vacation	2	3	10	10	10	10	2	3	**47**
PR-5: Review and revise your personal implementation and unit (developer) testing way of working.	vacation	vacation	2	3	10	10	10	10	2	3	**47**
PR-6: Other, specify	vacation	vacation	2	3	10	10	10	10	2	3	**47**
	vacation	vacation									
Planning Activities of a Developer Role	vacation	vacation									3394
PL-1: Review the requirement(s) for the unit(s) to be developed.	vacation	vacation	65	6	50	6	50	6	60	6	**243**
PL-2: Prepare (make) and/or review the design specification(s) for the unit(s) to be developed.	vacation	vacation	65	6	50	6	50	6	59	6	**242**
PL-3: Resolve unclear questions and uncertainties.	vacation	vacation	65	6	50	6	50	6	59	6	**242**
PL-4: Determine and document your implementation and unit (developer) testing goals.	vacation	vacation	65	6	50	6	50	6	59	6	**242**
PL-5: Determine your implementation and unit (developer) testing strategy.	vacation	vacation	65	6	50	6	50	6	59	6	**242**
PL-6: Determine appropriate implementation and testing practices.	vacation	vacation	65	6	50	6	50	6	59	6	**242**
PL-7: Identify standards to be used for meeting your goals.	vacation	vacation	65	6	50	6	50	6	59	6	**242**
PL-8: Set your own personal deadlines to be met during your implementation and unit (developer) testing work.	vacation	vacation	65	6	50	6	50	6	59	6	**242**

PL-9: Estimate effort and resources required for carrying out your work.	vacation	vacation	65	6	50	6	50	6	59	6	**242**
PL-10: Schedule your work.	vacation	vacation	65	6	50	6	50	6	60	6	**243**
PL-11: Review your implementation and unit (developer) testing plan to ensure that it is realistic and achievable.	vacation	vacation	65	6	50	6	50	6	60	6	**243**
PL-12: Identify risks related to your plan.	vacation	vacation	65	6	50	6	50	6	60	6	**243**
PL-13: Plan for managing any identified risks.	vacation	vacation	65	6	50	6	50	6	60	6	**243**
PL-14: Other, specify	vacation	vacation	65	6	50	6	50	6	60	6	**243**
	vacation	vacation									
Preparatory Activities of a Developer Role	vacation	vacation									7640
P-1: Prepare(make) and/or review your low-level design(s) of the code to be written or changed.	vacation	vacation	220	3	240	6	240	6	240	3	**955**
P-2: Prepare (make) an impact analysis of your low-level design(s).	vacation	vacation	220	3	240	6	240	6	240	3	**955**
P-3: Determine the types of unit (developer) test cases and their order.	vacation	vacation	220	3	240	6	240	6	240	3	**955**
P-4: Create and/or revise your unit (developer) test case base.	vacation	vacation	220	3	240	6	240	6	240	3	**955**
P-5: Revise the existing unit (developer) regression test base, if relevant.	vacation	vacation	220	3	240	6	240	6	240	3	**955**
P-6: Create or modify stubs and drivers, if required.	vacation	vacation	220	3	240	6	240	6	240	3	**955**
P-7: Prepare your unit (developer) testing environment and check whether it is appropriate for you work.	vacation	vacation	220	3	240	6	240	6	240	3	**955**
P-8: Other, specify	vacation	vacation	220	3	240	6	240	6	240	3	**955**
	vacation	vacation									
Coding Activities of a Developer Role	vacation	vacation									7310
C-1: Write/rewrite your code.	vacation	vacation	360	10	360	6	360	6	360	10	**1462**
C-2: Compile/ recompile your code as required.	vacation	vacation	360	10	360	6	360	6	360	10	**1462**
C-3: Make notes on your compilation errors, if necessary.	vacation	vacation	360	10	360	6	360	6	360	10	**1462**
C-4: Make notes on your defects	vacation	vacation	360	10	360	6	360	6	360	10	**1462**
C-5: Other, specify	vacation	vacation	360	10	360	6	360	6	360	10	**1462**

	vacation	vacation									
Unit Testing Activities of a Developer Role	vacation	vacation									10206
T-1: Check whether the unit (developer) test case base meets the given requirements and design.	vacation	vacation	360	6	360	6	360	6	360	6	**1458**
T-2: Check whether the unit (developer) regression test base meets the given requirements and design.	vacation	vacation	360	6	360	6	360	6	360	6	**1458**
T-3: Remedy requirements problems in your unit (developer) regression and/or test cases base, if any.	vacation	vacation	360	6	360	6	360	6	360	6	**1458**
T-4: Perform dynamic testing by executing code.	vacation	vacation	360	6	360	6	360	6	360	6	**1458**
T-5: Perform static (human) testing by reviewing your code.	vacation	vacation	360	6	360	6	360	6	360	6	**1458**
T-6: Record/write down test results.	vacation	vacation	360	6	360	6	360	6	360	6	**1458**
T-7: Other, specify	vacation	vacation	360	6	360	6	360	6	360	6	**1458**
	vacation	vacation									
Evaluative Activities of a Developer Role	vacation	vacation									93
E-1: Analyze your unit (developer) testing results.	vacation	vacation	0	10	0	10	0	10	1	10	**31**
E-2: Depending on the unit (developer) testing results, determine your next step(s).	vacation	vacation	0	10	0	10	0	10	1	10	**31**
E-3: Other, specify	vacation	vacation	0	10	0	10	0	10	1	10	**31**
	vacation	vacation									
Debugging Activities of a Developer Role	vacation	vacation									1329
D-1: Identify the source of (an) error(s).	vacation	vacation	10	6	200	6	200	6	15	6	**443**
D-2: Determine solution(s) for eliminating the sources of error(s).	vacation	vacation	10	6	200	6	200	6	15	6	**443**
D-3: Other, specify	vacation	vacation	10	6	200	6	200	6	15	6	**443**
	vacation	vacation									
Self-Assessment Activities (Document aside your self-assessment results)	vacation	vacation									664
A-1: Assess your own development work.	vacation	vacation	37	6	37	6	37	6	37	6	**166**
A-2: Identify causes of your mistakes.	vacation	vacation	37	6	37	6	37	6	37	6	**166**
A-3: Identify improvement areas in your own way of working.	vacation	vacation	37	6	37	6	37	6	37	6	**166**

A-4: Other, specify	vacation	vacation	37	6	37	6	37	6	37	6	**166**
	vacation	vacation									
Delivery of a Developer Role	vacation	vacation									44
S-2: Deliver your code.	vacation	vacation	0	10	0	6	0	6	0	10	**22**
S-3: Other, specify	vacation	vacation	0	10	0	6	0	6	0	10	**22**
Activities	**Week 1**										
ACTIVITIES THAT I CONDUCT IN THE ROLE OF A STUDENT	Mon	People	Tue	People	Wed	People	Wed	People	Fri	People	**Total**
Course 1: Time to fill “My Daily Work"											**0**
ACTIVITIES THAT I CONDUCT IN THE ROLE OF A SKILLED GENERALIST											
Managing Requirements of a Skilled Generalist Role											
REQ 1: Identify requirements	15	10	120	10	120	10	120	10	0	10	**415**
REQ 2: Analyze requirements	15	10	120	10	120	10	120	10	0	10	**415**
REQ 3: Change requirements	15	10	110	10	110	10	110	10	0	10	**385**
REQ 3.1: Identify impact of change (time it takes to identify changes everywhere in the system)	15	10	110	10	110	10	110	10	0	10	**385**
REQ 3.2: Make change(s) (time it takes to make changes everywhere in the system)	15	10	110	10	110	10	110	10	0	10	**385**
REQ 4: Plan requirements	15	10	340	10	340	10	340	10	0	10	**1075**
REQ 4.1: Estimate effort	15	10	90	10	90	10	90	10	0	10	**325**
REQ 4.2: Prioritize requirements	15	10	90	10	90	10	90	10	0	10	**325**
REQ 4.3: Other planning activities related to requirements (specify)	15	10	340	10	340	10	340	10	0	10	**1075**
REQ 5: Other, specify	15	10	109	10	109	10	109	10	0	10	**382**
Design of a Skilled Generalist Role											
DES 1: Design system or system component (high-level design)'	340	5	0	10	0	10	0	10	50	5	**425**
REQ 2: Analyze requirements	340	5	0	10	0	10	0	10	50	5	**425**
DES 3: Change design	340	5	0	10	0	10	0	10	50	5	**425**
DES 3.1: Identify impact of change (time it takes to identify changes everywhere in the system)	340	5	0	10	0	10	0	10	50	5	**425**

DES 3.2: Make change(s) (time it takes to make changes everywhere in the system)	340	5	0	10	0	10	0	10	50	5	**425**
DES 4: Other, specify	340	5	0	10	0	10	0	10	50	5	**425**
Testing (NON-DEVELOPER Level Testing, acceptance, system, integration) of a Skilled Generalist Role											
TEST 1: Define tests	0	5	0	10	0	10	0	10	0	5	**35**
TEST 2: Analyze tests	0	5	0	10	0	10	0	10	0	5	**35**
TEST 3: Change tests	0	5	0	10	0	10	0	10	0	5	**35**
TEST 3.1: Identify impact of change (time it takes to identify changes everywhere in the system)	0	5	0	10	0	10	0	10	0	5	**35**
TEST 3.2: Make change(s) to tests (time it takes to make changes everywhere in the system)	0	5	0	10	0	10	0	10	0	5	**35**
TEST 4: Other, specify	0	5	0	10	0	10	0	10	0	5	**35**
				10		10		10			
Project management of a Skilled Generalist Role				10		10		10			
PROJ-MAN 1: Plan project/part of project (iteration)	0	5	360	10	360	10	360	10	0	5	**1115**
PROJ-MAN 2: Analyze project/ project plan /part of project (iteration)	0	5	360	10	360	10	360	10	0	5	**1115**
PROJ-MAN 3: Change project plan /part of project plan (iteration)	0	5	360	10	360	10	360	10	0	5	**1115**
PROJ-MAN 4: Evaluate your project work	0	5	360	10	360	10	360	10	0	5	**1115**
PROJ-MAN 5: Manage risks	0	5	360	10	360	10	360	10	0	5	**1115**
PROJ-MAN 5.1: Identify risks	0	5	360	10	360	10	360	10	0	5	**1115**
PROJ-MAN 5.2: Analyze risks	0	5	360	10	360	10	360	10	0	5	**1115**
PROJ-MAN 5.3: Manage risks	0	5	360	10	360	10	360	10	0	5	**1115**
PROJ-MAN 6: Customer-related activities, specify	0	5	360	10	360	10	360	10	0	5	**1115**
ROJ-MAN 7: Other, specify	0	5	360	10	360	10	360	10	0	5	**1115**
Self Governent Developer ACTIVITIES CONDUCTed IN THE ROLE OF A (Pair)/DEVELOPER											
Preliminary Activities											
PR- 1: Review and agree on the overall or part of the project plan.	10	10	10	10	10	10	10	10	10	10	**90**
PR-2: Revise and ensure that the technology to be	10	10	10	10	10	10	10	10	10	10	**90**

used is tested and understood.											
PR-3: Revise and understand any appropriate internal (organizational) and external standard(s).	10	10	10	10	10	10	10	10	10	10	**90**
PR-4: Learn/relearn the organizational implementation and unit (developer) testing way of working.	10	10	10	10	10	10	10	10	10	10	**90**
PR-5: Review and revise your personal implementation and unit (developer) testing way of working.	10	10	10	10	10	10	10	10	10	10	**90**
PR-6: Other, specify	10	10	10	10	10	10	10	10	10	10	**90**
Planning Activities of a Developer Role											
PL-1: Review the requirement(s) for the unit(s) to be developed.	50	5	10	10	10	10	10	10	50	5	**165**
PL-2: Prepare (make) and/or review the design specification(s) for the unit(s) to be developed.	50	5	10	10	10	10	10	10	50	5	**165**
PL-3: Resolve unclear questions and uncertainties.	50	5	10	10	10	10	10	10	50	5	**165**
PL-4: Determine and document your implementation and unit (developer) testing goals.	50	5	10	10	10	10	10	10	50	5	**165**
PL-5: Determine your implementation and unit (developer) testing strategy.	50	5	10	10	10	10	10	10	50	5	**165**
PL-6: Determine appropriate implementation and testing practices.	50	5	10	10	10	10	10	10	50	5	**165**
PL-7: Identify standards to be used for meeting your goals.	50	5	10	10	10	10	10	10	50	5	**165**
PL-8: Set your own personal deadlines to be met during your implementation and unit (developer) testing work.	50	5	10	10	10	10	10	10	50	5	**165**
PL-9: Estimate effort and resources required for carrying out your work.	50	5	10	10	10	10	10	10	50	5	**165**
PL-10: Schedule your work.	50	5	10	10	10	10	10	10	50	5	**165**
PL-11: Review your implementation and unit (developer) testing plan to ensure that it is realistic and achievable.	50	5	10	10	10	10	10	10	50	5	**165**
PL-12: Identify risks related to your plan.	50	5	10	10	10	10	10	10	50	5	**165**
PL-13: Plan for managing any identified risks.	50	5	10	10	10	10	10	10	50	5	**165**

PL-14: Other, specify	50	5	10	10	10	10	10	10	50	5	**165**
Preparatory Activities of a Developer Role											
P-1: Prepare(make) and/or review your low-level design(s) of the code to be written or changed.	220	5	0	10	0	10	0	10	240	5	**495**
P-2: Prepare (make) an impact analysis of your low-level design(s).	240	5	0	10	0	10	0	10	240	5	**515**
P-3: Determine the types of unit (developer) test cases and their order.	240	5	0	10	0	10	0	10	240	5	**515**
P-4: Create and/or revise your unit (developer) test case base.	240	5	0	10	0	10	0	10	240	5	**515**
P-5: Revise the existing unit (developer) regression test base, if relevant.	220	5	0	10	0	10	0	10	240	5	**495**
P-6: Create or modify stubs and drivers, if required.	220	5	0	10	0	10	0	10	240	5	**495**
P-7: Prepare your unit (developer) testing environment and check whether it is appropriate for you work.	220	5	0	10	0	10	0	10	240	5	**495**
P-8: Other, specify	0	5	0	10	0	10	0	10	240	5	**275**
Coding Activities of a Developer Role											
C-1: Write/rewrite your code.	360	5	0	10	0	10	0	10	360	5	**755**
C-2: Compile/ recompile your code as required.	360	5	0	10	0	10	0	10	360	5	**755**
C-3: Make notes on your compilation errors, if necessary.	360	5	0	10	0	10	0	10	360	5	**755**
C-4: Make notes on your defects	360	5	0	10	0	10	0	10	360	5	**755**
C-5: Other, specify	360	5	0	10	0	10	0	10	360	5	**755**
Unit Testing Activities of a Developer Role											
T-1: Check whether the unit (developer) test case base meets the given requirements and design.	360	5	0	10	0	10	0	10	360	5	**755**
T-2: Check whether the unit (developer) regression test base meets the given requirements and design.	360	5	0	10	0	10	0	10	360	5	**755**
T-3: Remedy requirements problems in your unit (developer) regression and/or test cases base, if any.	360	5	0	10	0	10	0	10	360	5	**755**

T-4: Perform dynamic testing by executing code.	360	5	0	10	0	10	0	10	360	5	**755**
T-5: Perform static (human) testing by reviewing your code.	360	5	0	10	0	10	0	10	360	5	**755**
T-6: Record/write down test results.	360	5	0	10	0	10	0	10	360	5	**755**
T-7: Other, specify	360	5	0	10	0	10	0	10	360	5	**755**
Evaluative Activities of a Developer Role											
E-1: Analyze your unit (developer) testing results.	50	10	0	10	0	10	0	10	0	10	**90**
E-2: Depending on the unit (developer) testing results, determine your next step(s).	0	10	0	10	0	10	0	10	0	10	**40**
E-3: Other, specify	50	10	0	10	0	10	0	10	0	10	**90**
Debugging Activities of a Developer Role											
D-1: Identify the source of (an) error(s).	200	5	0	10	0	10	0	10	200	5	**435**
D-2: Determine solution(s) for eliminating the sources of error(s).	200	5	0	10	0	10	0	10	200	5	**435**
D-3: Other, specify	200	5	0	10	0	10	0	10	200	5	**435**
Self-Assessment Activities (Document aside your self-assessment results)											
A-1: Assess your own development work.	37	5	0	10	0	10	0	10	37	5	**109**
A-2: Identify causes of your mistakes.	37	5	0	10	0	10	0	10	37	5	**109**
A-3: Identify improvement areas in your own way of working.	37	5	0	10	0	10	0	10	37	5	**109**
A-4: Other, specify	37	5	0	10	0	10	0	10	37	5	**109**
Delivery of a Developer Role											
S-2: Deliver your code.	0	5	0	10	0	10	0	10	0	5	**35**
S-3: Other, specify	0	5	0	10	0	10	0	10	0	5	**35**