POLITECNICO
MILANO 1863

PROJECT for SOFTWARE ENGINEERING 2

Project Plan v1

Politecnico di Milano
A.A. 2016-2017
Prof.ssa Elisabetta Di Nitto

Students:
Diego Gaboardi
Giorgio Giardini
Riccardo Giol

INDEX

1 INTRODUCTION

1.1 PURPOSE AND SCOPE

This document is the Project Plan Document for PowerEnjoy.
It aims to analyze the complexity of the development of the whole project in term of cost and effort required to developers for a complete and consistent work. It can also be useful in order to estimate the required budget, to define the resource allocation and to establish the schedule for activities.

In the first part, in particular, we are going to estimate:

- The size of the project, in term of lines of code
- The total duration of the project, in term of months of work

This values will be used in the second part to propose a possible schedule covering all needed activities, from the requirement identification to the final implementation, giving a role to each developer.

In the last part we are going to analyze all possible risks that PowerEnjoy could face during the various phases of the project

1.2 DEFINITIONS AND ACRONYMS

1.2.1 DEFINITIONS

- COCOMO = the Constructive Cost Model or COCOCMO is a procedural software cost estimation model
- Function Points = a function point is a "unit of measurement" to express the amount of business functionality an information system provides to a user
- Gantt Chart = a Gantt chart is a graphical depiction of a project schedule

1.2.2 ACRONYMs

- FP = Function Point
- COCOMO = COnstructive COst MOde
- ILFs = Internal Logic Files
- EIFs = External Interface Files
- EIs = External Inputs
- EQs = External Inquiries
- SLOC = Source Lines Of Code

- SF = Scale Factors
- PREC = Precedentedness
- FLEX = Development Flexibility
- RESL = Risk Resolution
- TEAM = Team Cohesion
- PMAT = Process Maturity
- RELY = Required Software Reliability
- DATA = Database size
- CPLX = Product Complexity
- RUSE = Required Reusability
- DOCU = Documentation match to life-cycle needs
- TIME = Execution Time Constraint
- STOR = Main Storage constraint
- PVOL = Platform Volatility
- ACAP = Analyst Capability
- PCAP = Programmer Capability
- PCON = Personnel Continuity
- APEX = Application Experience
- PLEX = Platform Experience
- LTEX = Language and Tool Experience
- TOOL = Usage Of Software Tools
- SITE = Multisite Development
- SCED = Required Development Schedule

1.3 REFERENCE DOCUMENTS

- Assignment document: Assignment AA 2016-2017
- Previous example: PP 2015-2016.pdf
- Previous Documents:
    - IntegrationTesting_Gaboardi_Giardini_Giol_v1.pdf
    - DD_Gaboardi_Giardini_Giol_v2.pdf
    - RASD_Gaboardi_Giardini_Giol_v2.pdf
- Other documents: CII_modelman2000.0.pdf

2 SIZE, COST AND EFFORT ESTIMATION

2.1 INTRODUCTION

In this section we are going to estimate the following properties about our project: the expected size, the cost and the required effort.

So we are going to use the Function Points approach in order to obtain a good estimation of the size of our project in term of hypothetic number of lines of code.

Then for the cost and effort estimation we will use the COCOMO approach. In this way starting from the number of lines of code we will estimate the number of person hours necessary for developing the whole project.

2.2 SIZE ESTIMATION: FUNCTION POINTS

The Function Points approach allows us to obtain a good estimation of the size of our project considering the functions that are meaningful for our software. In particular we will consider the following program characteristics called function types:

- Internal Logic Files
- External Interface Files
- External Input
- External Output
- External Inquiry

In order to have a good estimation we will use this table obtained with statistical procedures from real projects.

FUNCTION TYPES	WEIGHT		
	Simple	Medium	Complex
Internal Logic Files	7	10	15
External Interface Files	5	7	10
External Inputs	3	4	6
External Outputs	4	5	7
External Inquiries	3	4	6

2.2.1 INTERNAL LOGIC FILES (ILFs)

In this section we are going to list and analyse the most important Internal Logical Files (ILFs) of our application and so the homogeneous sets of data used and managed by the application.

First of all our system has to store all the information about clients and so the following fields: name, surname, mobile phone number, email address, password, code, driving licence, credit card and his state (Dismounted, Reserving, Driving, OnBreak). All this attributes are strings (except the state) and are stored in a single table. In a similar way also the assistants are structured even if they don't have associated the driving licence, the credit card and the state.

Then another data managed by PowerEnjoy is the car which has several fields: code, model, battery, capacity which are stored as strings or integers. Then we have also the state of the car (Available, Reserved, OnCourse, OnBreak and NonAvailable) and two Boolean variable pointing out that the car is locked and in recharging state.

Then we store also the information about the town of Milan that are necessary for our application. So we consider the set of safeAreas which are identified by a code and by their coordinates. In a similar way also the parking stations are stored. However in this situation we must also save the information about rechargers which have a code and a Boolean that indicates if they are free or occupied.

Then there are the courses which are more complex than the previous one and so contains more attributes: the time of start and end, the data of start and end, the

cash amount, the position of start and end, the discounts used, the client and the car involved and the correspondent reservation. All this field are stored in a single table.

Finally there are the reservations that must contains the client, the car and the course associated, and the information about the countdown.

All these sets of data have an extremely simple structure being composed of a small number of fields. The only exceptions are courses which are more structured and therefore can be considered of medium complexity.

The following table summarizes what has been said in this paragraph.

ILF	Complexity	FPs
Clients	Simple	7
Assistants	Simple	7
Cars	Simple	7
SafeAreas	Simple	7
ParkingStations	Simple	7
Courses	Medium	10
Reservations	Simple	7
Total		52

2.2.2 EXTERNAL INTERFACE FILES (EIFs)

With External Interface Files (EIF) we mean a user identifiable group of logically related data that is used for reference purposes only. This kind of data resides entirely outside the application boundary and is maintained by another application external inputs. The external interface file is an internal logical file for another application.

In our project the only external data sources are the following:

- The mapping service which manages all the information about the positions in term of latitude and longitude;
- The payment system which handles the payments that occur in the services offered by the application.

In both the situation the amount of data that is used is extremely small and so their complexity is considered simple.

EIF	Complexity	FPs
Map position	Simple	5
Payments	Simple	5
Total		10

2.2.3 EXTERNAL INPUTS (EIs)

In this section we are going to consider all the typologies of interactions with external users of our application. In fact with External Input we mean data that are coming external to the application and may be used to maintain one or more internal logical files. In order to be clear the inputs will be divided according to the types of user.

Clients:

- Login/Logout: the client must insert his user name and the password in the proper form.
- Registration: the client must insert all required data which are immediately verified.
- Modify profile: the client after the login must insert the new credentials.
- Reserve a car: the client must insert his current position or a specific one and then has to select the car that he wants to pick up. He can also enable the saving option in order to find safe areas near the destination inserted.
- End course/Leave car in break: the client must answer if he wants to end is ride or leave the car in break.

Assistants:

- Login/Logout: it is the same situation as in the client
- Change car state: the assistant must insert the new state of the car. If the modified state in NonAvailable he must add the type of damage.

All these external inputs are extremely easy because they involve few inputs.
The first exceptions is client registration in which the user has to insert a lot of data that has to be verified and so it can be considered of medium complexity.
Then the reservation of a car is complex because there are interactions with several other entities such as car, course, client and safe area.

In the following table we show the function points associated with the External Inputs.

EI	Complexity	FPs
Client login/logout	Simple	2x3
Client registration	Average	4
Modify profile	Simple	3
Reserve a car	Complex	6
End course / leave the car	Simple	3
Assistant login/logout	Simple	2x3
Change car state	Simple	3
Total		31

2.2.4 EXTERNAL INQUIRIES (EQs)

In this section we are going to analyse the External Inquiries that occur in our application. They are elementary processes that send data or control information outside the application boundary in order to present data to a user through the retrieval of data or control information from an ILF or EIF.

In PowerEnjoy there are the following External Inquiries:

- A client can retrieve the actual reservation and the historical one.
- A client can retrieve the actual course and the historical one
- A client can retrieve all available cars near a position inserted or calculated by the GPS.
- A client can retrieve the sets of safe areas near a position inserted (this process occurs when he activates the saving option).
- An assistant can retrieve the list of cars and the information about their state.

All these operations can be carried out easily through queries that extract data from a singular table and so can be considered of simple complexity.

EQ	Complexity	FPs
Retrieve reservation	Simple	3
Retrieve course	Simple	3
Retrieve car available near position	Simple	3
Retrieve safe areas near position	Simple	3
Retrieve cars and their state (by assistants)	Simple	3
Total		15

2.2.5 EXTERNAL OUTPUTS (EOs)

In this section we are going to examine External Output and so the elementary processes that send data or control information outside the application boundary. The primary purpose of external outputs is to present information to a user through processing logic other than, or in addition to the retrieval of data or control information.

The situation in which our system has to communicate to the user outside the context of an inquiry are the following:

- Notify the client that the registration has been completed successfully.
- Notify the client that the reservation has been accepted.
- Notify the client that the payment has been completed successfully.
- Notify the client that the car has been locked properly.
- Notify the client that the ReservationCountdown or the CourtesyCountdown expired
- Notify the assistant that the car state has been changed correctly.

All these operations are extremely easy because they involve only few data in the interactions with the external users and for this reason they can be considered of simple complexity.

In the following table the External Output are listed with the Function Points associated.

EO	Complexity	FPs
Registration completed notification	Simple	4
Reservation accepted notification	Simple	4
Payment completed notification	Simple	4
Car locked notification	Simple	4
Countdown expired notification	Simple	4
Car state changed notification	Simple	4
Total		24

2.2.6 OVERALL ESTIMATION

Finally in this last section we summarize the results obtained in the previous paragraphs. In particular the following table shows the function points associated with all function types

Function type	FPs
Internal Logic Files	52
External Logic Files	10
External Inputs	31
External Inquiries	15
External Outputs	24
Total	132

So in conclusion we have 132 Function Points that according to the following table can be converted in lines of code.

- High: SLOC = 132 * 67 = 8844
- Median: SLOC = 132 * 49 = 6468
- Average: SLOC = 132 * 46 = 6072
- Low: SLOC = 132 * 15 = 1980

LANGUAGE	SLOC/FP			
	Average	Median	Low	High
JEE2	46	49	15	67

2.3 COST AND EFFORT ESTIMATION: COCOMO II

In this section we're going to estimate cost and effort needed to develop our project using COCOMO II.

2.3.1 SCALE DRIVERS

This is an analysis of our scale drivers:

- Precedentedness: this value reflects the experience of our developers in developing project similar to this product. The selected value is LOW, form the moment that we develop a large scale project for the first time.
- Development Flexibility: this value reflects how much our project in flexible in term of obligation to respect requirement and external constraint. The selected value is LOW, from the moment that we are forced to respect closely the requirement, even if we are quite free in the choice of technologies.
- Architecture/Risk Resolution: this value reflects the level of consciousness about what are the risks, the budget and the schedule for our project. The selected value is HIGH, from the moment that we performed an extensive analysis of risk and a schedule estimation.
- Team Cohesion: this value reflects how much project members know each other and are able to work in a team. The selected value for our team VERY HIGH.
- Process Maturity: this value reflects the maturity of our software organization. The selected value is LEVEL 4, from the moment that we didn’t have particular problems in developing our project, although this is our first work of this type.

Below is reported the official COCOMO II table that assign a value for each scale factor

Scale Factors	Very Low	Low	Nominal	High	Very High	Extra High
PREC	thoroughly unprecedented	largely unprecedented	somewhat unprecedented	generally familiar	largely familiar	thoroughly familiar
SF_j	6.20	4.96	3.72	2.48	1.24	0.00
FLEX	rigorous	occasional relaxation	some relaxation	general conformity	some conformity	general goals
SF_j	5.07	4.05	3.04	2.03	1.01	0.00
RESL	little (20%)	some (40%)	often (60%)	generally (75%)	mostly (90%)	full (100%)
SF_j	7.07	5.65	4.24	2.83	1.41	0.00
TEAM	very difficult interactions	some difficult interactions	basically cooperative interactions	largely cooperative	highly cooperative	seamless interactions
SF_j	5.48	4.38	3.29	2.19	1.10	0.00
PMAT	Level 1 Lower	Level 1 Upper	Level 2	Level 3	Level 4	Level 5
SF_j	7.80	6.24	4.68	3.12	1.56	0.00

This is the result of our evaluation:

Scale Driver	Factor	Value
Precedentedness (PREC)	LOW	4.96
Development flexibility (FLEX)	LOW	4.05
Risk resolution (RESL)	HIGH	2.83
Team cohesion (TEAM)	VERY HIGH	1.10
Process maturity (PMAT)	LEVEL 4	1.56
Total		14.5

2.3.2 COST DRIVERS

Required Software Reliability (RELY)

The system is a novelty and a great innovation for the city. This type of service aims to expand and reach most of the population so, in order to spread smoothly and avoid important financial losses, it must be enough reliable. The selected value is HIGH.

RELY Descriptors:	slight inconven-ience	low, easily recoverable losses	moderate, easily recoverable losses	high financial loss	risk to human life	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	0.82	0.92	1.00	1.10	1.26	n/a

Data Base Size (DATA)

This measure tries to capture the effective size of our database. An estimation is to reach about 1 GB of database, distributed in an interval of SLOC which goes from about 2000 to about 9000. According to COCOMO II scale the selected value is NOMINAL.

DATA* Descriptors		Testing DB bytes/Pgm SLOC < 10	10 ≤ D/P < 100	100 ≤ D/P < 1000	D/P ≥ 1000	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	0.90	1.00	1.14	1.28	n/a

Product Complexity (CPLX)

According to the COCOMO II rating levels this value is set to HIGH.

Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	0.73	0.87	1.00	1.17	1.34	1.74

Developed for Reusability (RUSE)

Requirements do not impose external reusability for our software. Reusability is limited across project itself, so this value is set to NOMINAL.

RUSE Descriptors:		none	across project	across program	across product line	across multiple product lines
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	0.95	1.00	1.07	1.15	1.24

Documentation Match to Life-Cycle Needs (DOCU)

In our project documentation correctly reflects all needs in term of product life-cycle, so this value is set to NOMINAL.

DOCU Descriptors:	Many life-cycle needs uncovered	Some life-cycle needs uncovered.	Right-sized to life-cycle needs	Excessive for life-cycle needs	Very excessive for life-cycle needs	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	0.81	0.91	1.00	1.11	1.23	n/a

Execution Time Constraint (TIME)

The percentage of available execution time used by our application is estimated to be about 70%. So this value is set to HIGH.

TIME Descriptors:			≤ 50% use of available execution time	70% use of available execution time	85% use of available execution time	95% use of available execution time
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	n/a	1.00	1.11	1.29	1.63

Main Storage Constraint (STOR)

The amount of storage usage with respect to the availability of the hardware is expected to be less than 50%. So this value is set to NOMINAL.

STOR Descriptors:			≤ 50% use of available storage	70% use of available storage	85% use of available storage	95% use of available storage
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	n/a	1.00	1.05	1.17	1.46

Platform Volatility (PVOL)

Our platform is fundamentally the mobile operating system. We estimate our client application to need a new release about once every 6 months. For this reason, this value is set to NOMINAL.

PVOL Descriptors:		Major change every 12 mo.; Minor change every 1 mo.	Major: 6 mo.; Minor: 2 wk.	Major: 2 mo.;Minor: 1 wk.	Major: 2 wk.;Minor: 2 days	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	0.87	1.00	1.15	1.30	n/a

Analyst Capability (ACAP)

Our analyst team have a good skill in analysis of requirements and design of the software. They have faced the problem in a complete and accurate way, analysing all the problems of the real world. For this reason, this value is set to HIGH.

ACAP Descriptors:	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.42	1.19	1.00	0.85	0.71	n/a

Programmer Capability (PCAP)

From the moment that the project has not yet been implemented this parameter is only an estimation. We think to have pretty good programming abilities, so this parameter is set to HIGH.

PCAP Descriptors	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.34	1.15	1.00	0.88	0.76	n/a

Personnel Continuity (PCON)

Our personnel are expected to be the same during the whole duration of the project. For this reason, this value is set to VERY HIGH.

PCON Descriptors:	48% / year	24% / year	12% / year	6% / year	3% / year	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.29	1.12	1.00	0.90	0.81	

Applications Experience (APEX)

Our project team don’t have a great experience in the development of application of this type, so this value is set to LOW.

APEX Descriptors:	≤ 2 months	6 months	1 year	3 years	6 years	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.22	1.10	1.00	0.88	0.81	n/a

Platform Experience (PLEX)

This is our first project developed using Java EE platform. Our previous project didn't include the usage of databases and networking. For this reason, this value is set to LOW.

PLEX Descriptors:	≤ 2 months	6 months	1 year	3 years	6 year	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.19	1.09	1.00	0.91	0.85	n/a

Language and Tool Experience (LTEX)

We have a good experience with Java programming language, but this is our first project developed using Java EE platform and tools of this kind. This value is set to NOMINAL.

LTEX Descriptors:	≤ 2 months	6 months	1 year	3 years	6 year	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.20	1.09	1.00	0.91	0.84	

Use of Software Tools (TOOL)

We use for the development of our application moderately integrated life-cycle management tool, so the selected value is HIGH.

TOOL Descriptors	edit, code, debug	simple, frontend, backend CASE, little integration	basic life-cycle tools, moderately integrated	strong, mature life-cycle tools, moderately integrated	strong, mature, proactive life-cycle tools, well integrated with processes, methods, reuse	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.17	1.09	1.00	0.90	0.78	n/a

Multisite Development (SITE)

Even if members of our team live in the same city for the whole duration of the development of this project, we largely use internet services for communication and organization of the work (whatsapp, social network). For this reason, this value is set to HIGH.

SITE: Collocation Descriptors:	Inter-national	Multi-city and Multi-company	Multi-city or Multi-company	Same city or metro. area	Same building or complex	Fully collocated
SITE: Communications Descriptors:	Some phone, mail	Individual phone, FAX	Narrow band email	Wideband electronic communication.	Wideband elect. comm., occasional video conf.	Interactive multimedia
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.22	1.09	1.00	0.93	0.86	0.80

Required Development Schedule (SCED)

In developing our project, we respect a nominal schedule for a project requiring an amount of effort of this kind, without particular stretch-out or acceleration. For this reason, the selected value is NOMINAL.

SCED Descriptors	75% of nominal	85% of nominal	100% of nominal	130% of nominal	160% of nominal	
Rating Level	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multiplier	1.43	1.14	1.00	1.00	1.00	n/a

This is the result of our evaluation

Cost Driver	Factor	Value
Required Software Reliability (RELY)	HIGH	1.10
Database size (DATA)	NOMINAL	1.00
Product complexity (CPLX)	HIGH	1.17
Required Reusability (RUSE)	NOMINAL	1.00
Documentation match to life-cycle needs (DOCU)	NOMINAL	1.00
Execution Time Constraint (TIME)	HIGH	1.11
Main storage constraint (STOR)	NOMINAL	1.00
Platform volatility (PVOL)	NOMINAL	1.00
Analyst capability (ACAP)	HIGH	0.85
Programmer capability (PCAP)	HIGH	0.88

Personnel continuity (PCON)	LOW	0.81
Application Experience (APEX)	LOW	1.10
Platform Experience (PLEX)	LOW	1.09
Language and Tool Experience (LTEX)	NOMINAL	1.00
Usage of Software Tools (TOOL)	HIGH	0.90
Multisite development (SITE)	HIGH	0.93
Required development schedule (SCED)	NOMINAL	1.00
Total		0.8686

2.3.3 EFFORT EQUATION

This is the final equation that estimate the effort in Person-Month (PM):

Effort = A * $\prod_{i=1}^{n} EM_i$ * $KSLOC^E$

with:

A = 2.94 PM/KSLOC (Cocomo II constant)

$\prod_{i=1}^{n} EM_i$ = 0.72555 (product of cost drivers)

KSLOC = from 1980 to 8844 (derived from size estimation)

E = B + 0.01 * $\sum SFi$

with:

B = 0.91 (Cocomo II constant)

$\sum SFi = 14.5$ (sum of scale driver)

Lower bound (SLOC Low)

Effort = 2.94 * 0.8686 * $1980^{0.91+0.01*14.5}$ = 7676 ≈ 8PM

Upper bound (SLOC High)

Effort = 2.94 * 0.8686 * $8844^{0.91+0.01*14.5}$ = 37229 ≈ 37PM

2.3.4 SCHEDULE ESTIMATION

This is the equation that estimate the final schedule:

Duration = C * $\text{Effort}^{D+0.2*(E-B)}$

with:

C = 3.67

D = 0.28

E and B defined in the previous paragraph

Lower bound (SLOC Low)

Duration = 3.67 * $7.676^{0.28+0.2*(1.055-0.91)}$ = 6.9 month

Upper bound (SLOC High)

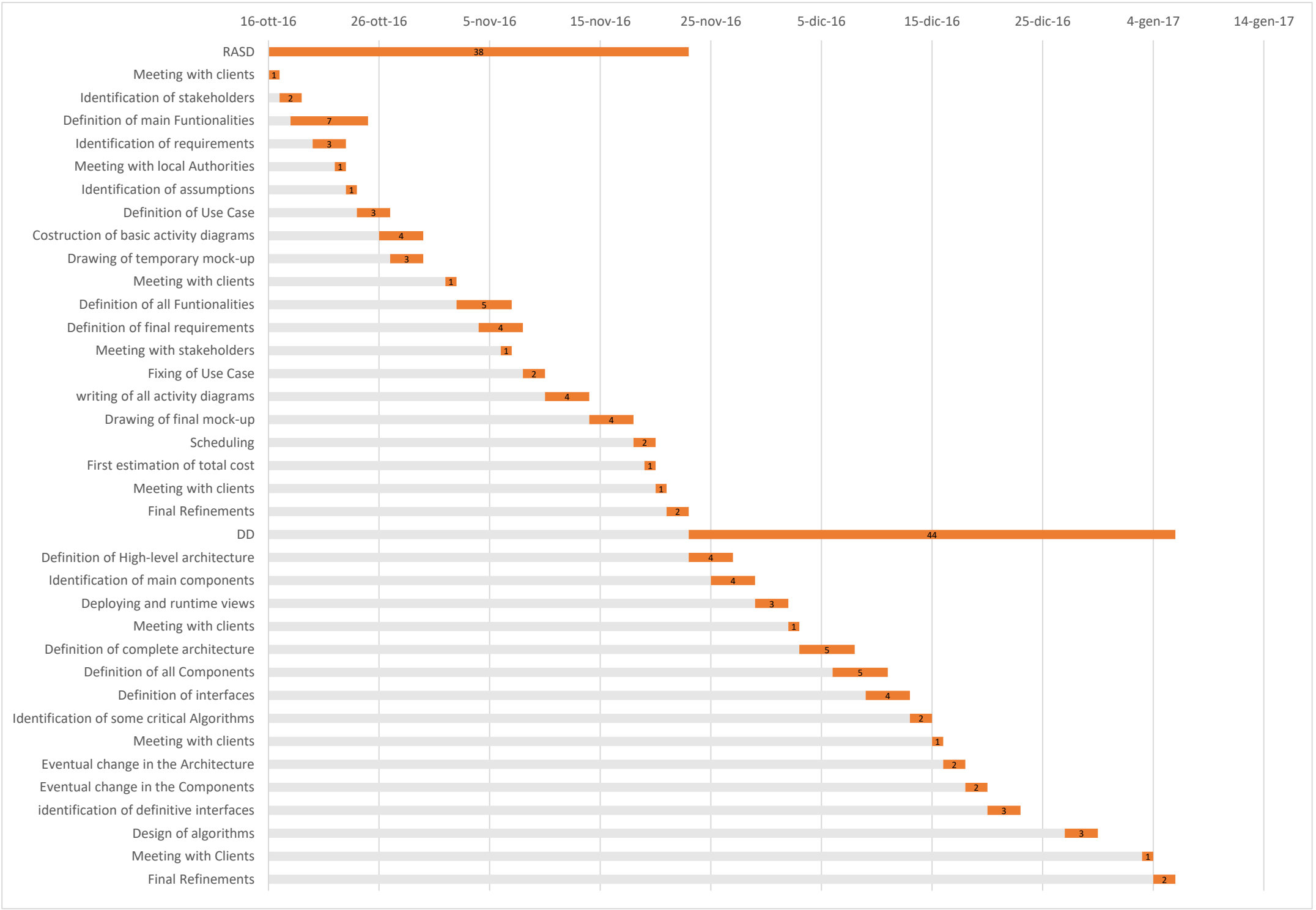
Duration = 3.67 * $37.229^{0.28+0.2*(1.055-0.91)}$ = 11.2 month

3 SCHEDULE

In this section we represent the schedule of our software in a time table. Every phase is represented with its amount of hours spent to be completed starting from the documentation of the software until its deployment.

The first part concerning the Requirements Analysis and Specifications Document and the Design Document is treated in specific since they are documents already finished. However, the diagrams are treated as the project was developed in a real contest, so there are some tasks not affronted by us but that should be completed if the project was real (like the meetings).
The development part is a statistic prevision of the amount of hours that will be spent in each phase. It respects the person months calculated in the previous paragraph with COCOMO. In the planning we have kept the Upper Bound to be sure to not exceed too much from the schedule. During the development of the software the time table can change due for some unexpected events occurred in this phase.

Below there are two Gantt diagrams about the scheduling of our project. The first is about the documentation. The second is about the future development of our software. In the graph the sections are highlighted with all upper case label. The numbers written in the bars are the day spent for the activity.


16-ott-16
26-ott-16
5-nov-16
15-nov-16
25-nov-16
5-dic-16
15-dic-16
25-dic-16
4-gen-17
14-gen-17
RASD
Meeting with clients
Identification of stakeholders
Definition of main Funtionalities
Identification of requirements
Meeting with local Authorities
Identification of assumptions
Definition of Use Case
Costruction of basic activity diagrams
Drawing of temporary mock-up
Meeting with clients
Definition of all Funtionalities
Definition of final requirements
Meeting with stakeholders
Fixing of Use Case
writing of all activity diagrams
Drawing of final mock-up
Scheduling
First estimation of total cost
Meeting with clients
Final Refinements
DD
Definition of High-level architecture
Identification of main components
Deploying and runtime views
Meeting with clients
Definition of complete architecture
Definition of all Components
Definition of interfaces
Identification of some critical Algorithms
Meeting with clients
Eventual change in the Architecture
Eventual change in the Components
identification of definitive interfaces
Design of algorithms
Meeting with Clients
Final Refinements
38
1
2
7
3
1
1
3
4
3
1
5
4
1
2
4
4
2
1
1
2
44
4
4
3
1
5
5
4
2
1
2
2
3
3
1
2

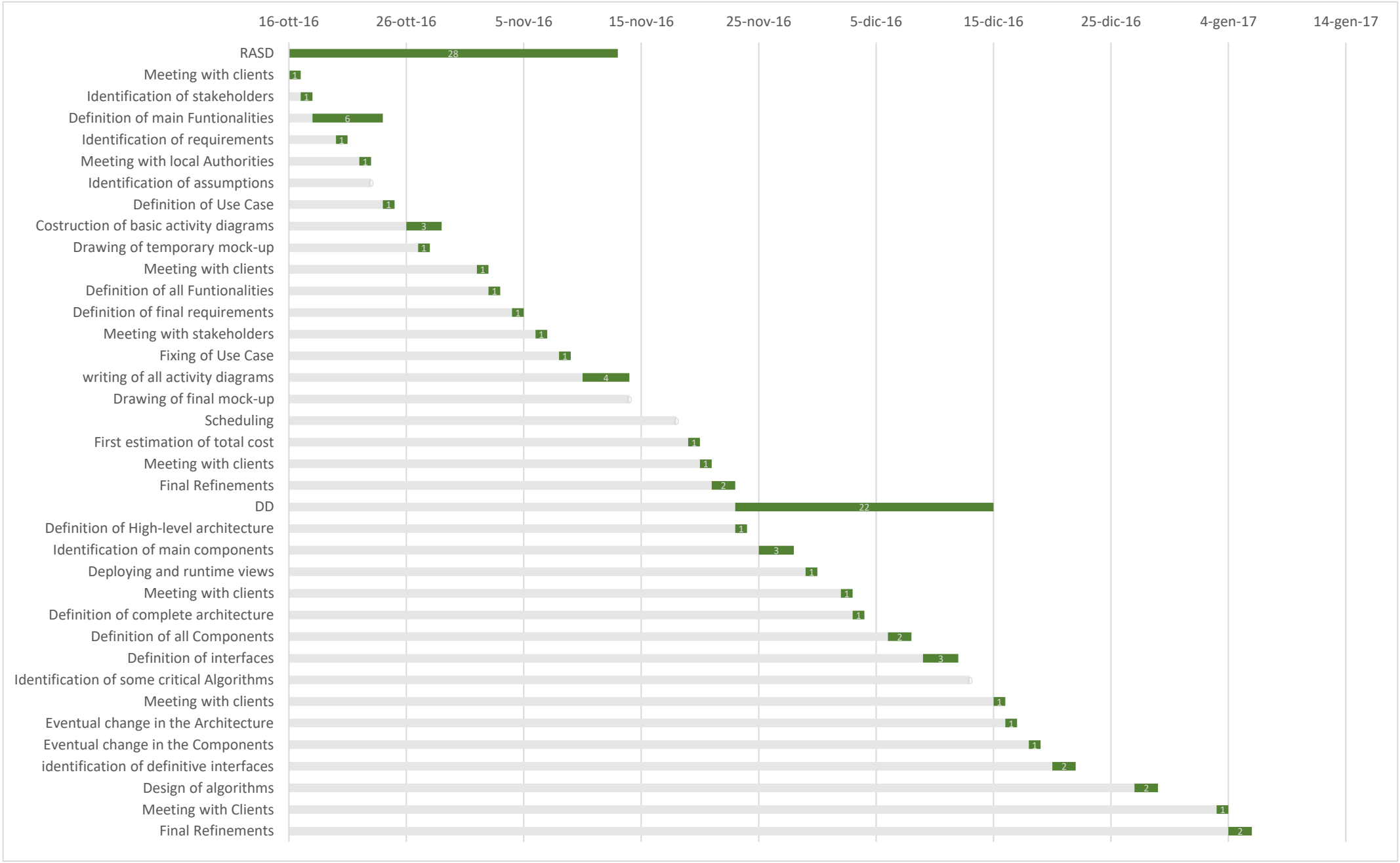

9-gen-17
28-feb-17
19-apr-17
8-giu-17
28-lug-17
16-set-17
5-nov-17
25-dic-17
13-feb-18
4-apr-18
DEVELOPMENT
325
Subcomponent development
100
External component Analysis
40
Code inspection
100
Unit testing
110
Subcomponent integration
20
Component development
60
Component Integration
38
Integration testing
50
System Testing
30
Apps Development
40
Apps integration
30
Apps testing
40
Internal Demostration
5
Fixes
15
Second Internal Demostration
5
Demostration to Clients
2
Demostration to Stakeholders
2
Fixes
5
Demostration to Clients
2
Demostration to Stakeholders
2
Final Fixes
4
Documents revision
3
User Manual
20
DEPLOYMENT
30
System Deployment
30
Teaching meeting to amministrators
3
Teaching meeting with Assistants
3
Preparations
10
START-UP
21
System first start
21
Real-Life Simulation
20
Main updates and checks
21

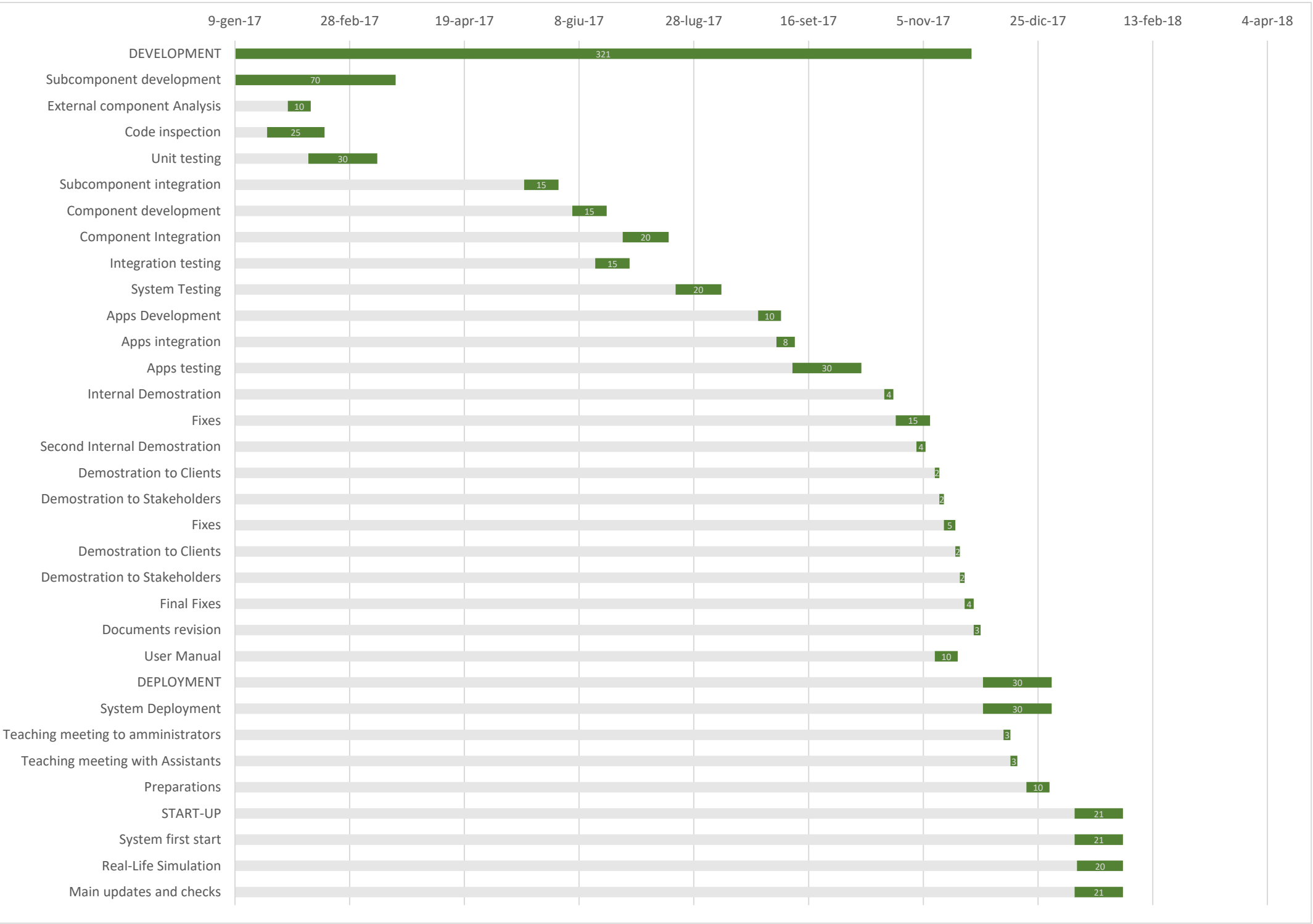
4 RESOURCE ALLOCATION

In this section we are represented the different amounts of work for each team member. Many tasks were faced before all together and then finalized by one member. The amount of days for each team member are relative both for the team work and for individual work. The days of work are represented at the beginning of the period concerning the task, but they can be distributed in all the period. Some more specific and realistic diagrams can be written during the progress in the development and deployment processes.

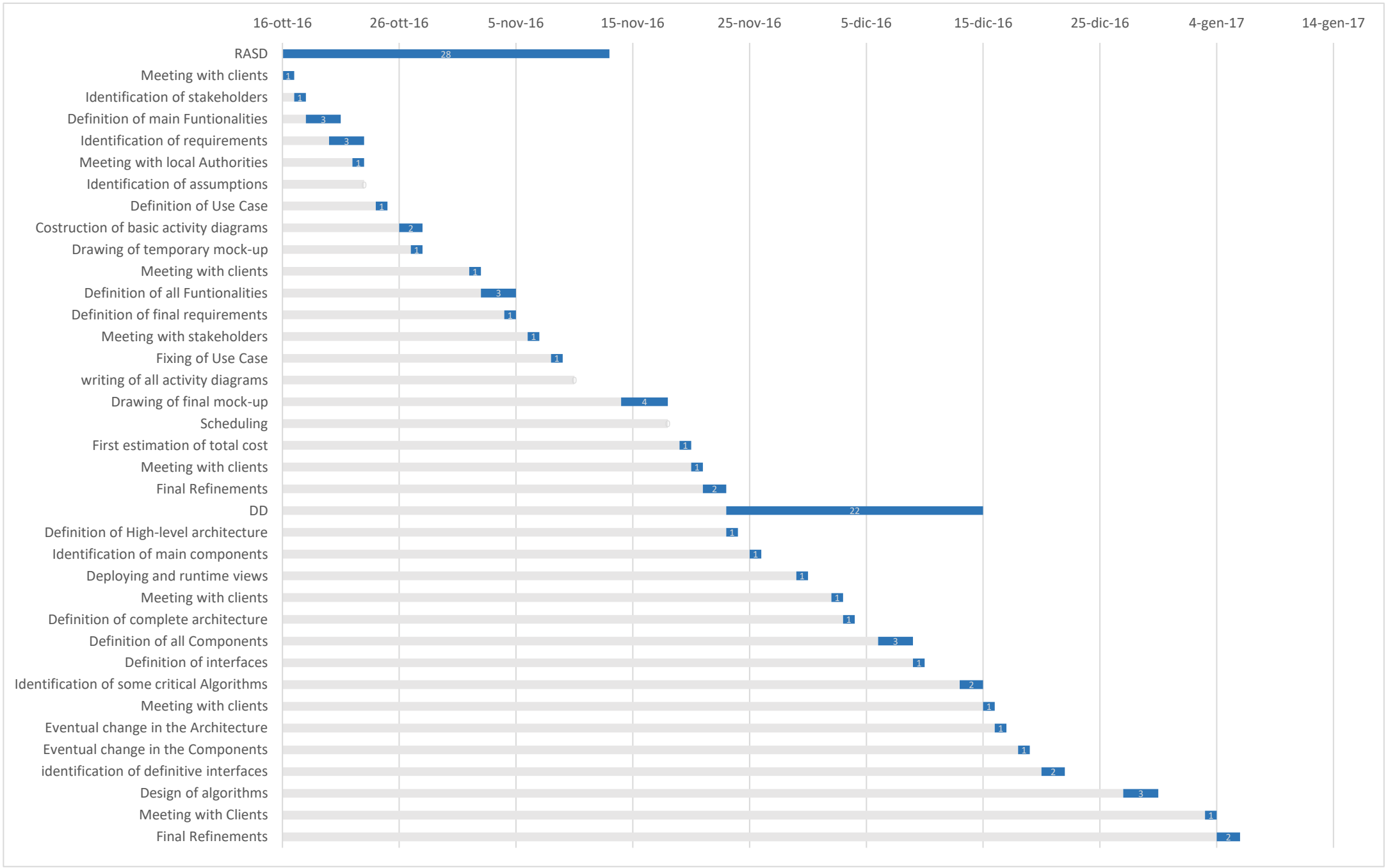
Even here the diagrams are split in two, to be more readable. The division of the first part reflect quite closely the true division of the work. For the second part the division are scheduled according to the different arguments treated by the different team members.

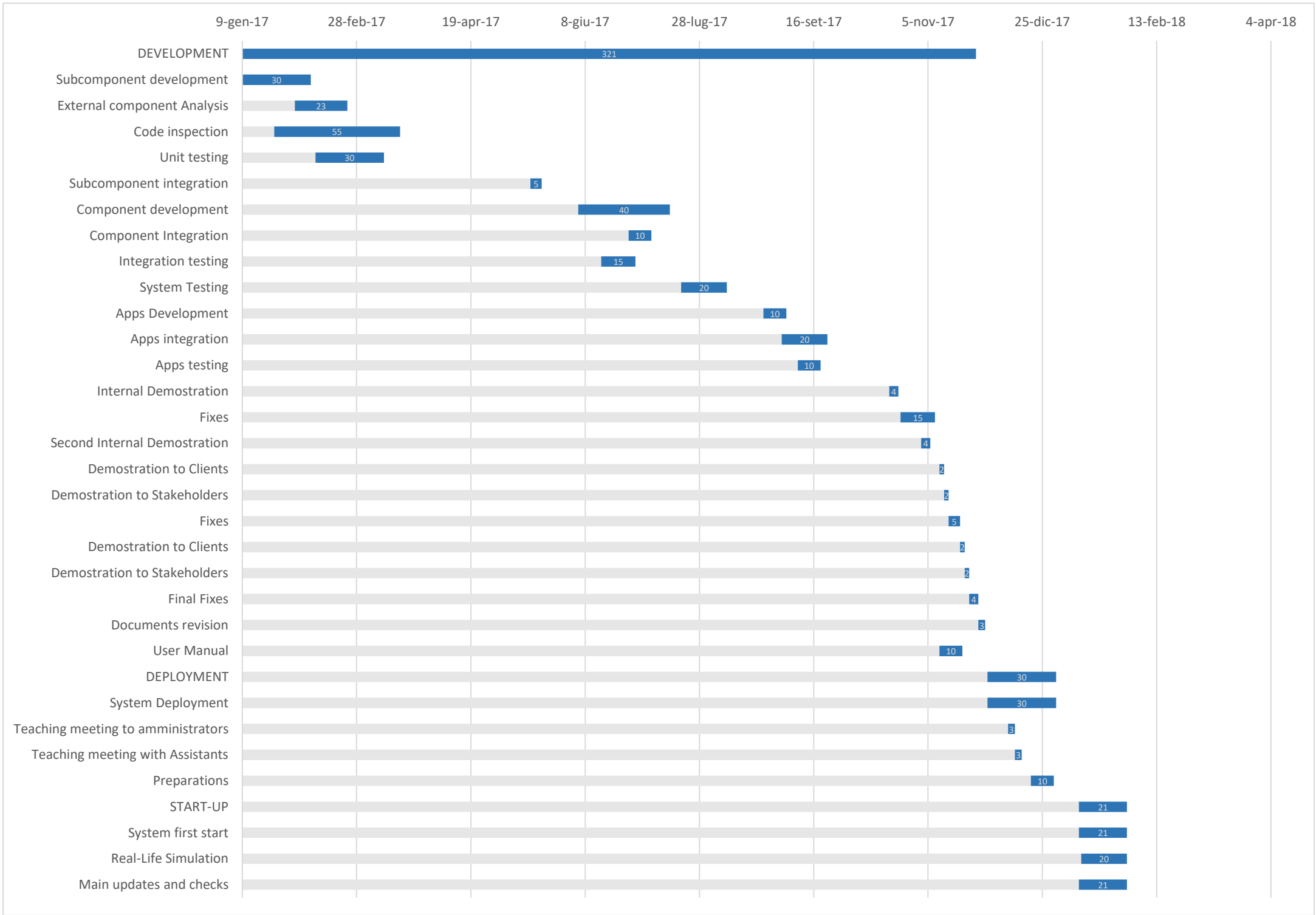
Diego Gaboardi



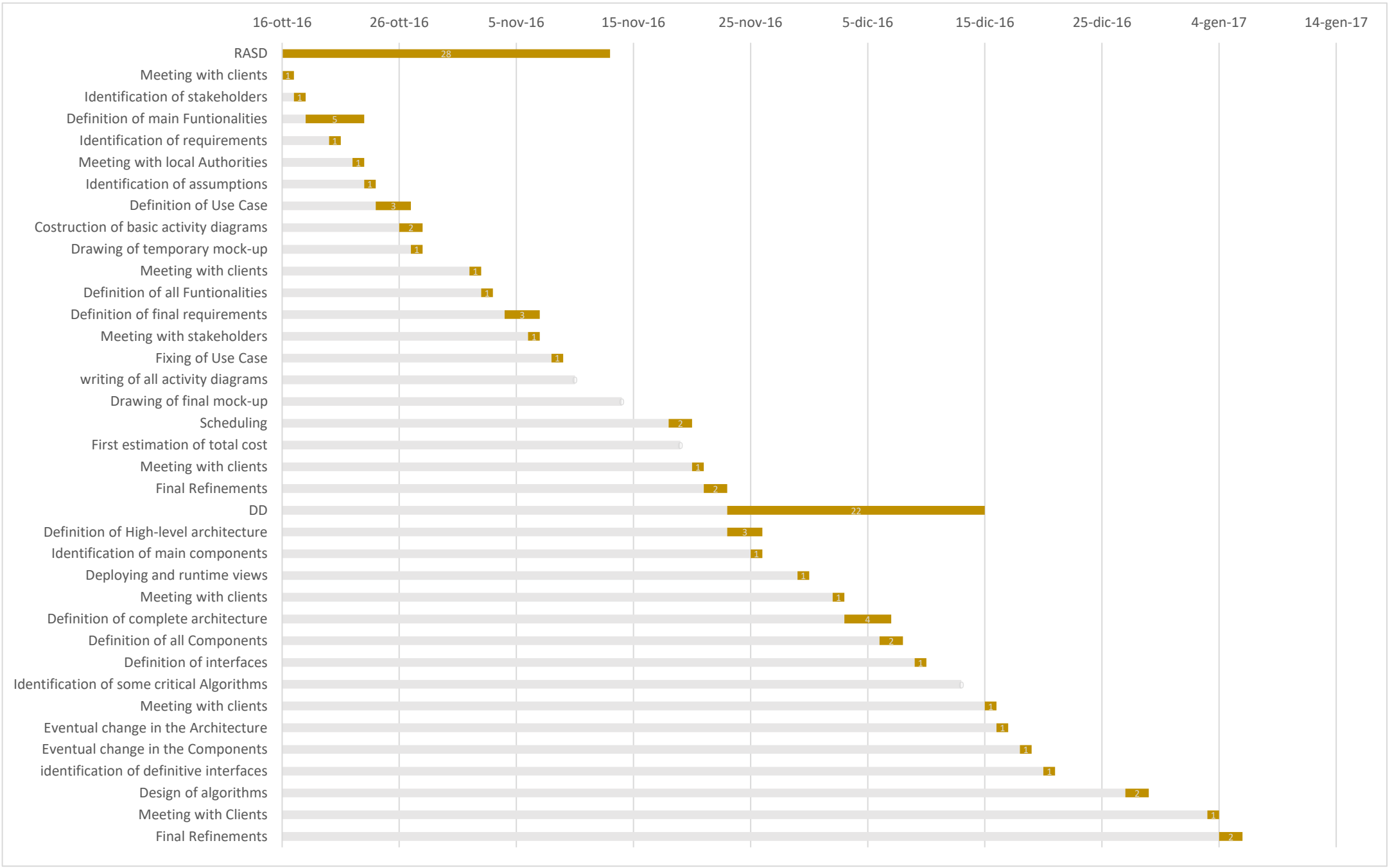

9-gen-17
28-feb-17
19-apr-17
8-giu-17
28-lug-17
16-set-17
5-nov-17
25-dic-17
13-feb-18
4-apr-18
DEVELOPMENT
321
Subcomponent development
70
External component Analysis
10
Code inspection
25
Unit testing
30
Subcomponent integration
15
Component development
15
Component Integration
20
Integration testing
15
System Testing
20
Apps Development
10
Apps integration
8
Apps testing
30
Internal Demostration
4
Fixes
15
Second Internal Demostration
4
Demostration to Clients
2
Demostration to Stakeholders
2
Fixes
5
Demostration to Clients
2
Demostration to Stakeholders
2
Final Fixes
4
Documents revision
3
User Manual
10
DEPLOYMENT
30
System Deployment
30
Teaching meeting to amministrators
3
Teaching meeting with Assistants
3
Preparations
10
START-UP
21
System first start
21
Real-Life Simulation
20
Main updates and checks
21

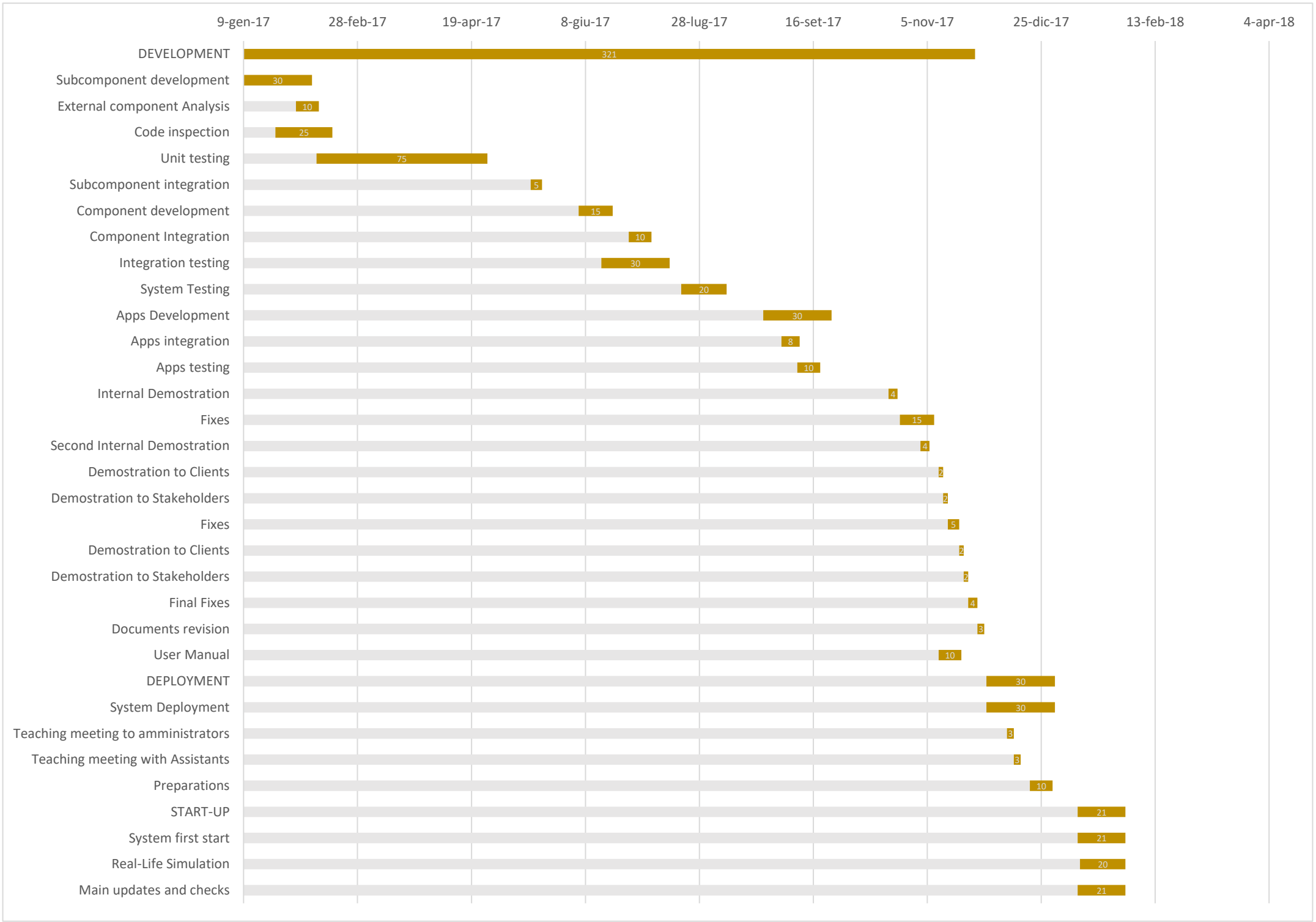
Giorgio Giardini




9-gen-17
28-feb-17
19-apr-17
8-giu-17
28-lug-17
16-set-17
5-nov-17
25-dic-17
13-feb-18
4-apr-18
DEVELOPMENT
321
Subcomponent development
30
External component Analysis
23
Code inspection
55
Unit testing
30
Subcomponent integration
5
Component development
40
Component Integration
10
Integration testing
15
System Testing
20
Apps Development
10
Apps integration
20
Apps testing
10
Internal Demostration
4
Fixes
15
Second Internal Demostration
4
Demostration to Clients
2
Demostration to Stakeholders
2
Fixes
5
Demostration to Clients
2
Demostration to Stakeholders
2
Final Fixes
4
Documents revision
3
User Manual
10
DEPLOYMENT
30
System Deployment
30
Teaching meeting to amministrators
3
Teaching meeting with Assistants
3
Preparations
10
START-UP
21
System first start
21
Real-Life Simulation
20
Main updates and checks
21

Riccardo Giol




9-gen-17
28-feb-17
19-apr-17
8-giu-17
28-lug-17
16-set-17
5-nov-17
25-dic-17
13-feb-18
4-apr-18
DEVELOPMENT
321
Subcomponent development
30
External component Analysis
10
Code inspection
25
Unit testing
75
Subcomponent integration
5
Component development
15
Component Integration
10
Integration testing
30
System Testing
20
Apps Development
30
Apps integration
8
Apps testing
10
Internal Demostration
4
Fixes
15
Second Internal Demostration
4
Demostration to Clients
2
Demostration to Stakeholders
2
Fixes
5
Demostration to Clients
2
Demostration to Stakeholders
2
Final Fixes
4
Documents revision
3
User Manual
10
DEPLOYMENT
30
System Deployment
30
Teaching meeting to amministrators
3
Teaching meeting with Assistants
3
Preparations
10
START-UP
21
System first start
21
Real-Life Simulation
20
Main updates and checks
21

5 RISK MANAGEMENT

In this section we evaluate and analyse some risks that can occur for our software. The risks can be technical, relative to the development of the software, relative to quality or timeliness or relative to business aspects, like the market.

All the doubts relative to the administration of certain areas and management of some legal situations have to be discussed during the documentation with the authorities. The economic management, government grants and how these can change in the future have to be discussed. The economic warranty has to be discussed too. Some not calculated risk of this kind can brings important losses at the company.

Other legal risks can be linked with some changing in the legislation. The condition for a valid driver can change with another kind of driver licence or some other necessary data. This brings to an extension of the client data and so an upgrade of the system. The valid electric plugs can change and so they have to be replaced as the power point of the cars. The cars can turn to be no more accordant to the laws, and so they may be substituted.

An economic risk can be the increasing of the cost of the electricity, and it can weight on the recharging of the cars. Other new taxes and increases of some goods can bring at a loss of util. So we always try to keep a good margin in the income.

There are risks linked to some accidental events. All the accidents are managed by the warranty. If some client has a problem with a car, he can always call a green number and some assistants will help him. The assistants can manage many aspects linked to course, client, car and reservation and so solve any problems. If the problem is linked to some charging station the assistant can change the states of the station or temporary remove it, or the plug, from the system.

It is even to evaluate the ambient in which the system will be inserted. In this ambient we have to monitor the concurrency systems, in order to provide a service that can be competitive in the market. We have to evaluate and exploit well the territory in which our system work, allocating in strategic and well diffused position the charging stations. This can be decisive for providing a well service to the client, and so to bring them to use it. The system has to be competitive especially in its debut. So for the first period some bonuses can be available for the new subscribers.

Some risks that can occur during the development and documentation of the software can be linked to the absence of an important figure in some specific sector. It is avoided by having always some personnel member that can hold the charge of the absent member. The problem of the absence of some important figure can occur even during the utilization of the system, and it is avoided in the same way. For how concern the publication times, all the deadlines have an error margin.

Problems can derivate from a changing in the interfaces with the external components. Any changing has to be presented at the company in advance, so the differences with the previous components can be managed. A no-managed changing can bring some dysfunctionalities in some services. Problems can occur even with the vendor of the tablet for the car and with the provider of its OS for the adjustment to do for well integrate it in the car system.

The loss of personal data and in particular of the source data can be disastrous. So all the data have to be replicate in different databases in different locations.

6 HOURS OF WORK

The writing of this document took overall about 13 hours of head-work.

In particular we used 3 ours in order to divide the work equally and to discuss about the most critical issues. Then we spent 8 hours individually in order to complete our tasks.

At the end we reread the complete document in the last 2 hours.

7 USED TOOLS

The tools we used to create this document are:

- Microsoft Word 2016: to write and assemble the document
- Dropbox e GitHub: to share work
- Microsoft Excel 2016: to create the Gantt Charts