Dor Carmi- 205789662

Elad Feldman- 308358947

Assignment 1:

Foundation of software engineering

Part 1-first question:

1. **The four qualities of software essence by brooks:**
    a. **Complexity**- the system's number of different elements, their interaction with each other, the complex and large structure, that can lead to unneeded side effects, un-visualized state the constitute security problems. The consequence can be that the software turns to be too hard to use, and hard to extend the program to more functions.
    b. **Conformity**- the system's complexity is arbitrary, forced by the many human institution and systems to which the interfaces must confirm. The human factor has a big part of the complexity of the design, and the complexity usually comes from the need of the software to conform to other interfaces.
    c. **Changeability**- software systems usually asked to be changed very frequently since its being tried in many cases, people find better ways to use it, the long survival of the software on changed technology and more reasons.
    d. **Invisibility-** software is invisible and un-visualizable, not possible to capture it in space, and to deal with that there are geometric abstraction tools and floor plans and diagrams that help build and architect the software's performance, so it can be evaluated by the developers and the clients and create a joined language.
2. **Prototyping:**
    a. Brooks believes that prototyping is essential at software system develop process, because clients do not necessarily know exactly what they want and cannot specify exactly what they need. The develop of rapid prototyping is a way to simulate the important interfaces, and without doing that many fundamental problems cannot be revealed and fixed on time.
    b. Berry does not agree. By his opinion, it delays the develop of the software system, it costs a lot (time and money) to build the prototype and then "throw it a way" and develop the real program from the beginning. Basing the develop of the software on the develop of the prototype leads to poor design, and the first

production is hard to modify and tend to turn bad and not work for every change.

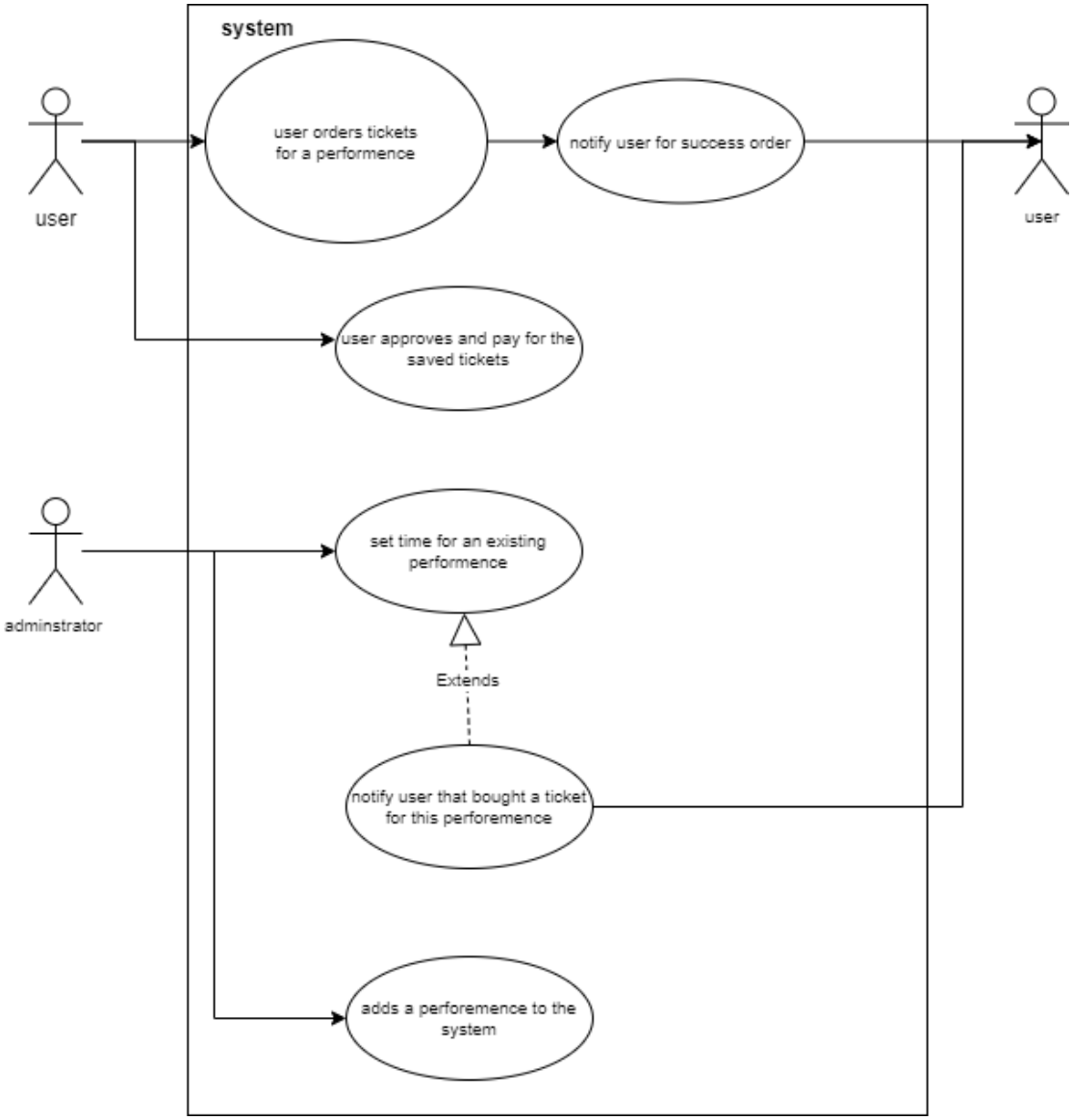
3. **BMUF vs IRUF:**
    a. **What is the difference between the two methods?** – They are different in the way they approach changes in the requirements.
    In the **BMUF** method the attempt is to clarify all the requirements at the start completely before beginning its implementation. In this method changes are not accepted easily, and after gathering all the needed requirements, any change can be rejected or denied.
    In the **IRUF** method changes are embraced. By communicating with all the stockholders using system models and use cases and are meant to be ready for incremental changes.
    b. **Which method will you choose if you are the only company in the specific market and why?** I would choose the BMUF, since many knowledge is already known by the developers, and since there is no much competition with other companies- this way it is possible to gain as much needed information about the exact requirements, and help making the develop process easier and more efficient and priciest, and for any necessary change the company might get profit.
4. **Documentation:**
    a. **Three problems:**
        i. It requires that people stop what they are doing (their needed work for the develop of the project) and concentrate on documenting.
        ii. If there is no time for documentation, the details needed to be document pile up and can get forgotten, and so the documentation might get missing and unreliable.
        iii. Because of these problems people do not trust the existing documentation, and for that it takes more time to understand the code and realize the consequences of making changes.
    b. **B-L graph:** If the documentation is **performed properly**, on time and as necessary, the **B-L upswing is both delayed and moderated**, because it helps the team working on the same project understand the decisions that took place, and it helps communicate and create a clear plan for all team working and for the decision makers.
    However, if the documentation is **performed poorly**, not consistent and not managed right, not only it doesn't improve the shape of the graph, but it make it worst**- the upswing sets in earlier and is steeper**, which means that in earlier part of the project (time), the bugs start to pile up, more and more bugs appear and the develop of the software system gets more and more complicated.
    The reason is that the documentation that was done not properly is unreliable, and so the time increases twice- both for the time that was spent on attempting to create these poor documentation, and for the

time that is spent on working on the project without these documentation because they are unreliable.

5. **Object oriented paradigm:**
    - **a.** This paradigm doesn’t satisfy the silver bullet and only addresses to accidental difficulty, because it only allows to prevent design mistakes by giving a high-order expression of the design, that allows to detect and prevent accidental difficulty, but the complexity of the design stays the same and so this paradigm doesn't improve this complexity at all.
    - **b.** Berry agrees with Brooks- "there is no proof and no evidence that the software productivity has increased with the new method". He claims that the process of software engineering was not affected by the new technique- OOP, and there is still a need for something to help improving this process. Furthermore, the OOP contributes to the Information Hiding pain- difficulties in making future changes depends on having identified a right composition. The changes might cause modifying code that crosses several modules, which will cost structural change which can be very painful.

Part 2- second question:

Use cases- to explain the story



Sub-question A: acceptance tests

Action	Data	Expected Results
Administrator adds a new performance to the system.	New performance: name, description, hall name, time, location, date, and last date for purchase	**Success:** The performance added successfully to the system, so it appears in the performances page
Administrator adds a new performance to the system.	New performance: name, description, hall name, time, location, **outdated date,** and last date for purchase	**Failure:** Should not be able to enter an outdated date of performance
User orders tickets from the system.	New order: name, phone number, and seats	**Success:** The user receives a notification that the order is successfully completed
User orders tickets from the system	New order: name, phone number, and already taken seats	**Failure:** The user cannot purchase already taken seats
User confirm tickets' purchase	Open order	**Success:** Seats are marked as taken and the user adds to the list of time change observers
User confirm tickets' purchase	Order made more the one hour ago	**Failure:** Order that is outdated cannot be approved

Sub-question B: hidden assumptions and acceptance tests for them

User story	Point of view	Hidden assumption	How it was revealed	High level acceptance tests		
				Action	Data	Result
Adding performance	Customer (mifa'al ha'pais)	Two different performances should not be accruing at the same place in the same date	This hidden assumption was revealed by thinking of the customer viewpoint, for the customer not to "screw up" and set by mistake two performances at the same place	Add two performances at the same date and place	2 new performances: name, description, hall name, same time, same location, same date, and last date for purchase	Failure: cannot add two performances at the same time and/or at the same place
Adding performance	Customer (mifa'al ha'pais)	When filling in the performance data one cannot set up a "last date to purchase" before today and not after the performance's date	This hidden assumption was revealed by thinking of the customer viewpoint, for the customer not to "screw up" and set by mistake last date to purchase before today or after the performance	Add a performance with a "last date to purchase" before today	New performance: name of performance, description, hall name, time, location, date, and outdated last date for purchase.	Failure: cannot set up a performance with last date to purchase before today
Adding performance	Customer (mifa'al ha'pais)	A performance can be added without a specific time and can be changed later	By discussing the way performances are schedule right now with the customer	The administrator changes an existing performance exact hour	Existing performance: name, description, hall name, time, location, date and outdated last date for purchase and new exact time	Success: the performance time is updated
Order tickets to a performance	User	A user can cancel the pre-paid order	By asking the user what can irritate you	The user cancels an order	Existed open order and the order's number	Success: the order is successfully cancelled, and the seats are free to order again
Order tickets to a performance	User	When the order is finished successfully the user will get an order number	By asking the user how you would want the system to respond?	The user submits new order	New order: name, phone number, and seats	Success: the user received an order number

Sub-question C: data table for the tests

User story: adding performances									
Test	**Characteristic**	**Name of performance**	**Description**	**Hall name**	**date**	**Location**	**Last date to purchase**	**Hour**	**Note**
Administrator adds a new performance to the system	good	“you”	Drama	“ha’tarbut"	23/12/2020	Haifa	4/12/2020	none	Normal Scenario
Administrator adds a new performance to the system	sad	“you”	Drama	“ha’tarbut”	1/1/19	Haifa	3/12/2020	none	Bad time: time that passed, wrong input
Administrator adds a new performance to the system	bad	"you"	Very long input…	"A2%^"	1/1/1400	London	1/1/1400	none	Bad input- invalid, too long, does not make sense
Two different performances should not be accruing at the same place in the same date	good	- “Lion king” - “Toy story”	- kids - kids	-“Habima" -“Habima”	1/1/2020	- Afula - Afula	20/12/2020	- None - None	Normal scenario
Two different performances should not be accruing at the same place in the same date	bad	- “Lion king” - “Toy story”	- kids - kids	-“Habima” -"AbiMa"	1/1/2000	- Afula - Afula	20/12/2020	- None - None	Same place, not the same input- should not work- should fail due to place unavailable

User story: Ordering tickets for a performance									
Test	**Characteristics**	**Confirmed**	**Order number**	**Name**	**Phone number**	**Seats**	**Time issued**	**Expected result**	**Note**
User orders tickets from the system	Happy	No	Xx-55-32	Ido shwarz	050-1112223	23J, 24J, 25J	15/07/2020 16:53	Success: order opens, user gets the order's number	Normal scenario
User orders tickets from the system	Sad	No	Xx-55-32	Ido sh2arz	050-11122	23J, 24J, 250K	15/07/2020 16:53	Fail: Suggests the user to correct the specific data	User enters data incorrectly
User orders tickets from the system	Bad	No	-0---	Very long name....	999-999-999-999	100000J, 240000J, 250000000J	15:07/2020 16:53	Fail: either the data entered is extreme or there is a failure by the server or the system	The user is notified that there is a problem, and the order is cancelled
User confirm tickets purchase	Happy	Yes	Xx-55-32	Ido shwarz	050-1112223	23J, 24J, 25J	15/07/2020 17:25	Success: the order is confirmed	Normal scenario
User confirm tickets purchase	Sad	No	Xx-55-3	----	-----	-----	15/07/2020 17:25	Fail: cannot confirm the tickets	The order number is incorrect, the system will

									suggest closest order numbers
User confirm tickets purchase	Bad	No	Xx-55-32	----	----	----	15/07/2020 20:00	Fail: cannot confirm the tickets	either the user confirmed too late or the system is not working, and the confirmation cannot happen
A user can cancel the pre-paid order	Happy	No	Yy-32-48	Maya Cohen	054-9638525	66A, 67A	20/9/2019 09:30	Success: the order is cancelled	Normal scenario- open ordered that is not confirmed
A user can cancel the pre-paid order	Sad	Yes	ZZ-59-63	---	---	---	---	Fail: cannot find order	Incorrect order number, or an order that is already confirmed and payed
A user can cancel the pre-paid order	Bad	No	ZZ-59-63	Asaf Levi	058-9876543	1A,1B	01/01/2020	Fail: cannot delete order	Either the performance is in the past, or there is a problem with the db. and the order cannot be cancelled

Part 3- third question:

Sub questions A and B- implemented in the java code.

Sub questions C-

To modify the tests so that they will communicate with the real software:

- In the proxy class chose to set real to a new instance of the implementation of the service class that has the functionality and connects to the real software implementation
- The tests stay the same and use the same functionality from the interface so nothing changes, and they can stay the same- no need to re-compile them.