Þróunhugbúnaðar
Lokaprófs undirbúningur

Pétur

29. apríl 2018

Efnisyfirlit

1 Software Engineer

1.1 Definition

Hugbúnaðarverkfræðingur verður að hafa góð tök á forritun, slunginn í reikniritum og uppbyggingu gagna. Verkfræðingurinn þarf að hafa góð skil á nokkrum hönnunar módelum heldur en að geta forritað þau.

1.2 Important skills

- Góða samskiftar hæfileika
- Getað búið til líkön af flóknum verkefnum
- Getað skipulagt og stjórnað vinnu

2 General project work

2.1 Reasons for failure

2.1.1 Excessive schedule

Ef það er of mikið vinnuálag á verkefninu sem unnið er að þá getur verkefnið átt í hættu að vera unnið illa.

2.1.2 Changing needs

Breytingar í miðju verkefni geta haft slæm áhrif á verkefni sem unnið er að.

2.1.3 Lack of documented project plan

Mikilvægt er að hafa allt nánast ritað í stein svo vinnu menn geta verið á sömublaðsíðu.

2.2 Causes of Software Project Troubles

- Verkefni unnið í nýju software umhverfi
- Breyting á viðskiptavinum
- Tími sem fer í að læra á verkefnið
- Miskilningur, mismunandi markmið og uppgjöf á verkefni.

3 Software Process Models

3.1 Big Bang Approach

Hönnuður fær engar athugasemdir frá kúnna

3.2 The Waterfall Model

Hönnuður fær athugasemdir frá kúnna og þarf að endurtaka sum hönnunarskref aftur.

3.3 Plan-driven

Skipulags módel reynir að komast að öllu í byrjun og heldur sig við ákvarðanir.

- Reyna að skilja og plana fyrir fram stóra hluti af verkefninu og skilgreina það ítarlega í byrjun verkefnis
- Skipt er verkefni niður á hönnuði
- Föst skilar áætlun fyrir hvern þátt í verkefninu
- Lítið rými fyrir breytingar á verkefni þegar það er komið í mitt kaf
- Mikinn skilning og yfir sýn á framþróun útkomu verkefnisins
- Spíral módel ...

3.4 Agile Models

Sveigjanleg módel Skrifa og endurgera planið fyrir næstu ítranir meðan verkefninu stendur yfir.

- Byrja með hráa mynd af verkefni, fínpússa sig í gegnum margar prótótýpur og lykkjur af athugasemdum frá viðskiptavini
- Hönnuður(developer) fær að velja sér verkefni
- Skil á þáttum(component) er hægt að enduskipuleggja skilar tíma og mikilvægi hvenær sem er.
- Engar skilgreind atriði og plön
- Lítil stjórnun, yfirsín yfir hvernig verkefnið gengur og mun vera.

4 Software Engineer

4.1 Definition

Hugbúnaðarverkfræðingur verður að hafa góð tök á forritun, slunginn í reikniritum og uppbyggingu gagna. Verkfræðingurinn þarf að hafa góð skil á nokkrum hönnunar módelum heldur en að geta forritað þau.

4.2 Important skills

- Góða samskiftar hæfileikar.
- Getað búið til líkön af flóknum verkefnum
- Getað skipulagt og stjórnað vinnu.

5 Requirements Engineering

5.1 Functional requirement

Skilgreina eiginleikar sem hugbúnaðurinn verður að hafa. **Dæmi**: Ökumaður verður látin vita ef þrýstingur í dekkjum er lár.

5.2 Quality requirement

Skilgreina þægindinn á bakvið búnaðinn. **Dæmi**: Leit í leitarvélin tekur í mestalagi 1 sekúndu.

5.3 General condition

Almennt sem má búast af vörunni. **Dæmi** appið sem er verið er að þróa þarf að líta snirtilega út.

5.4 Conflicts

5.4.1 Subject conflict

Misskilið eða vitlausar upplýsingar, mismunandi túlkun staðreynda.

5.4.2 Conflict of interest

Mismunandi áhugi, mismunandi áætlanir og markmið fjárfesta.

5.4.3 Value conflict

Mismunandi skoðanir tengdir ákveðnum hlutum. Til dæmis menningarlegir mismunir.

5.4.4 Structural conflict

Ójafnvægi í valdadreifingu.

5.4.5 Imagine the university contracted you to build a new social networking site for all students. Explain two types of conflicts that you may run into during the requirements elicitation and give an example for each of them

Subject conflict ef þeir gefa mér verkefnið og ég skil ekki upplýsingarnar sem mér voru gefnar til þess að gera tengsla netsíðuna þá getur átt sér stað deila um hvernig niðurstaðan á að vera. Conflict of interest ef ég sem hönnunður er að vinna í síðunni vill leggja mikla áherslu á að útfæra einnhvað x sem ég held að væri gott fyrir vefinn. En kúninn hefur allt aðra áherslu og vill útfæra b þá getur mindast deila milli míns og kúnnans.

5.5 Technical detail in user story

5.5.1 User story recipe

{ Role/Perspective, Goal/Function, Rational/Benefit(optional)}

- Role/Perspective: Sem notandi
- Goal/Function: Ég vill getað leitað af hótelum með fillterum
- Rational/Benefit(optional): , svo að ég get fundið hótelið sem ég vill bóka

5.5.2 How much technical detail should be in a user story

Í notendar sögu þá ætti ekkert af tæknilegum atriðum, lausnum að koma framm. Ef tæknilegu atriðin koma fram þá þarf kúnninn að geta skilið þau líka. Kúninn og ég við þurfum að geta skilið notendasögurnar.

5.5.3 Example: good user story

Bókun flugs: Notandi verður að geta bókað flug á ákveðnum tíma og dagsetningu.

5.5.4 Example: bad user story

Nota jflex fyrir UI: Notendar viðmótið mun nota jflex til þess að sýna gögninn á flottan máta.

6 Effort Estimation

Leið til þess að ákvaða hversu mikinn tíma þarf til þess að setja í hver verk.

6.1 Planing poker

Planing poker er þannig að hver meðlimur hóps fær 13 spil sem hafa mismunandi daga fjölda til þess að geta metið hvað hvert verkefni taki langan tíma. Einn hringur er þannig að verkefni er lagt á borð svo velja allir eitt spil í leind og sína svo allir spilin sín og ræða niðursöðurnar sínar. Í sameiningu þá er tekið ákvörðun útfrá röksæmdarfærslu annara, hverssu mikinn tíma hópurinn heldur í raun og veru hverssu mikinn tíma fer í verkið.

6.1.1 Based on differing skills and experience, individual team members may estimate different efforts for any requirement. Discuss whether planning poker eliminates this problem

Ef það kemur upp að hópur af fólki sem er að spila planing poker og er með breytilegt getustig. Þá er hættan að fólk mun kjósa mjög breytilega. Það er samt gott því þá opnar það fyrir umræðuna afhverju eru svörin svona breytileg. Það getur reynst að fólki finnist verkið svo flókið fyrir sig að það kýs rosalega háa tölu. Sama fyrir fólkið sem finnst þetta rosalega létt sem mun kjósa þá lágu töluna. Með þessa niðurstöður þá getur fólkið sem hefur gert þennan verkþátt áður og veit betur en fólkið sem kaus mjög hátt fyrir verkþáttin. Sagt þeim ástæðuna afhverju þau kusu svona lágt og hvernig verkefnið verður leist. Þar með sannfært fólkið með minni kunnátuna að breyta um skoðun og færast nær sinni skoðun. Þetta getur virkað einnig í hina áttina. Fólkið sem heldur að þetta er auðvellt en í raun og veru þá sá fólkið sem kaus háatölu vandarmál sem væri erfitt að leisa. Það segir fólkinu sem kaus lágt frá þessu vandarmáli og fær það til þess að kjósa nær sér í tölu. Planing poker mun leisa þetta vandarmál að vissuleiti því planing poker er góð leið til þess að vekja upp umræður fólks á verkefninu sem mun vera unnið að.

6.1.2 Assume your team came up with an effort spread of 8, 13, 40 and 40 for a particular user story. Interpret and deal with the result

Gefum okkur að það var kosið fyrir notenda sögu x. þar sem það var kosið 8, 13, 40 og 40 þá er líklega misræmi í getustigi. Þeir sem kusu minnst þá 8 þarf að útskíra fyrir hinum afhverju hann haldi að útfæra þessa notendasögu muni vera svona létt. 13 mun vera skilningsríkar á því afhverju 8 kaus 8. En svo er það að 8 og 13 þurfa líkleg ræða mikið við 40 og 40 afhverju þarna er svona mikill munur. Það er hugsanlegt að það er mikill munur á getustigi, að báðar 40 vita ekki neitt hvernig á að útfæra notendarsöguna eða að þær sjá stæri vanda sem 8 og 13 sjá ekki. Svo getur þetta virkað á hinn bógin líka að af því að 8 og 13 kunna að gera verkefnið og báðar 40 sjá fram á það að þurfa að læra mikið til þess að geta unnið verkið. Það getur líka reynst að 8, 13 sjá strax lausnina en báðar 40 sjá hana ekki en þegar 8 og 13 hjálpa til við að útskíra lausnina þá sjá 40 og þá munu þeir vera á sömublaðsíðu.

6.2 Anchoring effect

Ef þú ert með tölu í hausnum á þér þá mun hún hafa áhrif á matið þitt. Jafnvel ef þú ert sérfræðingur á sviðinu. Dæmi: Hvað varð Ghandi 144 ára gamall. Þá er manneskjan svö föst á tölunni og mun giska í kringum hana.

7 Project Planning

7.1 Calculating new velocity

Byrjunar hraði er stundum metinn þannig að valið er 0.7 sem er dæmigerður verktími fyrir hugbúnaðargerð. Einnig er hægt að velja hraða úr fyrri verkefnum. Nýr vinnu hraði er alltaf endur reiknaður eftir hvern áfanga til þess að fá raunhæfa sýn yfir vinnu getu hópsins. Dæmi um reikning á vinnu hraða:

- Summa saman persónu daga úr verkefninu sem klárað var
- Deila því með persónu dögunum
- Hef 20 vinnu daga, 3 vinnendur, kláruð verk 38 persónu dagar

- $\frac{KlarudVerkIPersonuDogum}{vinnendur*vinnudaga}$
- Hraðinn er: $\frac{38}{3*20} = 0.6$

7.2 Predicting how much you can get done next time

Með nýja hraðanum er hægt að meta hverssu mikið það er hægt að áorka í næsta þætti. Dæmi:

- Næsti þáttur er 1 mánuður 20 dagar
- Þrír sem ætla að vinna að verkinu
- þannig tími sem er metinn í að hverssu mikið verður klárað þann mánuð er
- Þannig að í næsta þætti þá mun aðeins ná að klára svon marga vinnu daga $60 * 0.6 = 36$

7.3 Assume your 4-person team completed tasks comprising 20 person-days in a 2-week iteration. Calculate the velocity you should assume when planning the next iteration

Tvær vikur þá hef ég tíu vinnudaga sem unnið var. þannig að þá er hraðinn $\frac{20}{4*10} = 0.5$

7.4 Imagine the a client expects more functionality in a release then you will be able to complete until the deadline, based on your effort estimates. Suggest a strategy you could use in this situation

Með höndla nýja verkefnið eins og önnur, skoða verkefnið og fá á hreinnt hvað gera skal. Brjóta verkefnið upp ef þess þörf er á. Meta og forgangsraða bæta svo við burn down chart. Það er hægt að semja við kúnna að það þurfi að færa verkefni yfir á næsta þátt eða sleppa hluta verkefnis til þess að koma því nýja inn.

8 Object-Oriented Analysis and Design

8.1 Associations

Tenging opin örvar oddur [class|car] -wheels -> [class|wheel] Sem sýnir að bíll hefur tengingu við dekk, bíll þarf dekk til þess að keyra.

8.2 Aggregation

[class|StoreInventory]<>—*-[class|tire] Klasarnir eru ekki háðir hvort öðrum en StoreInventory geymir dekk.

8.3 Composite

[class|wheel]-1—<*>[class|tire] Fyllt ör merkir að hlutirnir lifa aðeins ef þeir eru saman ef einn eyðilegst þá skemmast þeir báðir.

8.4 UML University library diagram

...

8.5 Generalization

Er samband milli yfirklasa og undirklasa í uml er það tóm ör –|>. Fer þannig from að það er safnað saman klösum sem hafa svipaða eiginleika eins og sport bíll og fjölskildu bíll. Þannig er hægt að generalize þá þannig að þeir munu vera undir bíl klasa.

8.6 Specialization

Merkir að búa til nýjan klasa sem erfir frá öðrumklasa, sérhæfing.

8.7 Generalization and specialization seem to be contracting the same concept(inheritance). Explain why both terms are accurate nevertheless

Þetta virkar svipað þegar maður er að teikna uml og hefur marga flokka þá er hægt að flokka þá í undir og yfir flokka. Svo þegar kemur að setja þá í java þá er hætt að sérhæfa klasa og láta undirklasa erfa frá yfirklasa til þess að undir klasinn getur notað eiginleika yfir klasans.

8.8 Abstract classes

Leið til þess að setja in föll sem verða síðar meira ekki útfærð seinna meir. Klasi verður að vera absstrackt ef að minstakost ein aðferð er abstrackt. Má vera abstract þó allar aðferðir hafa verið gerðar.

8.9 Interface

- Interface er túlkað þannig að það hefur mengi aðgerða sem klasi eða þáttur(hlutur) þar sem aðrir klasar hafa aðgang að aðgerðonum.
- Svipað og abstract klasi sem inniheldur bara abstract methods. En í interface þá verður hlut klasi b að útfæra aðferðir í interface a.

8.10 Two key differences between abstract classes and interfaces in java

abstract klasi getur verið með óskilgreindar og skilgreindar aðferðir en ekki interface. Í abstract klasa er hægt að veita hvaða leifi sem er á aðferðir, attribute en í interface þá er alltaf default public sett á föll. Abstract klasi getur erft frá abstract klasa og ekki útfært aðferðir. Interface er vanarlega túlkað sem tjáning á getu en abstract klasi tjáning af tegund.

8.11 Explain the difference between aggregation and composition of classes in an object-oriented model

Aggregation Klasi er með geymslu í sér[fylki,hlaða,...] og geymir tilvik af öðrum klasa í sér til dæmis: [class|hilla]<>—-[class|bækur]. En composition væri þá að tveir klasar lifa ekki án hvor annars eins og samsenting klasans dekk og bíls sem myndar bíl. Ef það væri bara bíll þá myndi ekki geta myndast tilvik á þeim klasa.

9 Object-Oriented Programming

9.1 Attributes

Ástand ílát með breytilegum upplýsingum, ekki sínilegt fyrir utan hluts. Þetta er breyta.

9.2 Behavior

Aðgerðir sem hluturinn getur framkvæmt.

9.3 Identity

Tilvik af hlutinum til dæmis: hlutur í alvöru heiminum, minnishólf í tölvu.

9.4 Static in java

Static er bundið við klasan og er hægt að nálgast án þess að hafa tilvik af honum. En meðlima breytur eru bundnar við tilvik af klasa.

9.5 Explain why any static methods of java class can access only the static attributes of the class

Af því að static attributes er bundinn við klasan en ekki að það sé gert tilvik af klasanum til þess að geta notað fallið úr nýja tilvikinu.

9.6 Class variable

Static breyta

9.7 Instance variable

Tilviksbreyta

9.8 Instance vs Class

Það þarf tilvik af klasa til þess að hafa aðgengi af breytum og föllum en í static breytum þá þarf það ekki.

9.9 Explain how singleton pattern ensures that only one instance of a class can exists in the system

Með því að gera smiðin í klasanum private og static fall sem skilar static final hlut af klasanum.

10 Test money class implement function isequal

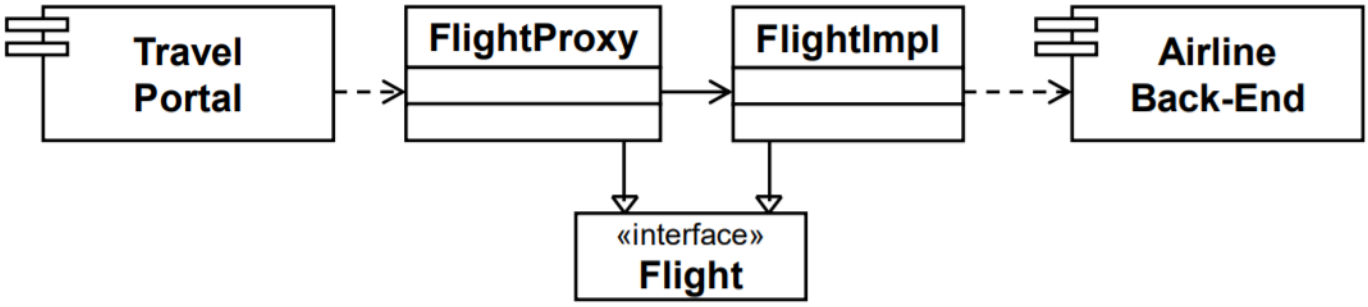
10.1 a) an equal amount in the same currency and compare it to stored item in Money

```
import static org.junit.jupiter.api.Assertions.*;

import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

class isEqual {

        @BeforeEach
        void setUp() throws Exception {
        }

        @AfterEach
        void tearDown() throws Exception {
        }

        @Test
        void test() {
                Money money = new Money(100,"$");
                boolean output = money.isEqual(money);
                assertEquals(true,output);} }
```

11 Design Patterns

12 Explain the purpose of Proxy pattern, and give an example of a scenario (outside the travel domain) where its use would be beneficial

Leifa viðmótinu vinna með proxy hluti sem líkjast raunverulegum gagnarhlutum, en innihaldið mun þá innhalda dýru gögni ef þess þörf er á. Hér er mynd sem sýnir hvernig þetta lítur út í raunveruleikanum:



13 Polymorphism

Aðferðir sem eru yfirskrifaðar þannig að sama aðferðin er hægt að nota fyrir marga hluti.