Okruhy otázok podľa moodle

1. Kto meria úspešnosť projektu?

-metrika, softvérové meranie

2. Čo je metrika?

-je hocijaký nápad, ktorým si inšpirovaný pri pozeraní/čítaní textu (nie len textu)
-napr: riadky kódu, počet metód, komentáre, operátory a operandy, správanie objektu počas vykonávania

3. Dve triedy softvérových metrík

-produkty – programy, dokumentácia, reporty
-procesy – ktorými bol produkt vyvíjaný (životné cykly, fázy špecifikácie, dizajnu, implementácie, testovanie a údržba)

4. Čo je meranie?

-meranie je pozorovanie, ktoré je vykonávané veľmi pozorne. Používajú sa na identifikovanie nových javov
-testujú predpokladané javy
-každé meranie vyžaduje stupnicu alebo metriku
-Meranie je mapovanie z empirického sveta do formálneho, relačného sveta. V dôsledku toho je meranie číslo alebo symbol priradený objektu pomocou tohto mapovania aby určilo atribút

5. Matematická notácia pre metriky

-meranie je mapovanie z empirického sveta do formálneho, relačné sveta. V matematike je metrika funkcia - m: X-> R, kde X je ľubovoľný typ, R je skupina kladných racionálnych čísel, kde pre všetky x patriace do X platí:

- m(x) je definované
- m(x) >= 0
- m(x1 U x2) <= m(x) + m(x2)

6. Externé metriky produktu

-vychádza z klasifikácie podľa I.Sommervilla, ktorý rozdeľuje metriku na metriku produktov (merajú vlastnosti softvérových produktov) a metriku procesov (merajú vlastnosti procesov použitých na získanie týchto produktov)
-metrika produktu ďalej zahrňuje 2 kategórie
    -externá metrika produktu – vlastnosti viditeľné používateľovi
    -interná metrika produktu – vlastnosti viditeľné tímu vývojárov
-metrika nespoľahlivosti – určuje počet zostávajúcich chýb
-metrika funkcionality – posudzuje akú užitočnú funkcionalitu produkt poskytuje (AMEISE)
-metrika výkonu – posudzuje produktom využívané zdroje ako rýchlosť výpočtu, priestoru a obsadenosti

-Fan-in nám udáva počet modulov, ktoré volajú daný modul
-Fan-out nám udáva počet modulov, ktoré sú volané daným modulom
-vo všeobecnosti, moduly s veľkým fan-in sú relatívne malé a jednoduché a zvyčajne uložené na spodnej vrstve dizajnovej štruktúry. Naopak moduly ktoré sú veľké a komplexné majú zväčša malý fan-in
-z toho vyplýva, že moduly s veľkým fan-in a veľkým fan-out sú biedne navrhnuté a je potreba ich predizajnova

15. FPA - základný princíp

-FPA – function point analysis
-FP metrika bola z väčšej časti vyvinutá aby pomohla spravodlivo určovať ceny za vývoj softvéru. Počet FP je rozdielny ako úsilie potrebné na vývoj systému
-vplyv FPA na plánovanie projektu – určovanie rozsahu, posúdenie dopadu náhrady, ceny náhrady, požiadavky vyhodnotenia, prideľovanie testovacích prostriedkov, posúdenie rizík, monitorovanie progresu
-môže byť aplikovaná v ktorejkoľvek fáze vývoja, zvyčajne ale na odhad ceny
-využívaná sa preto lebo funkcionalita nemôže byť meraná

16. FPA - ako na to?

-ako použiť FPA v 5tich krokoch
-musíme vypočítať celkový počet, ktorý bude použitý na definovanie zložitosti projektu (UFP – Unadjusted funcion points)
-musíme zistiť zložitosť nastavenia hodnôt (VAF – Value adjustment factor, AFP = UFP*VAF)
-musíme zistiť LOC, vybraním programovacieho jazyka, ktorý použijeme
-posledným krokom je vybranie zložitosti softvérového projektu z COCOMO tabuľky (Constructive Cost Model)
-no a teraz by sme mali vypočítať hodnoty pre effort (úsilie) a duration (trvanie)

17. COCOMO 81

-COCOMO – Constructive Cost Model, algoritmický softvér pre odhad nákladov modelu. Využíva základné regresné zložene s parametrami, ktoré sú odvodené od historických dát projektu ako aj súčasných a budúcich charakteristík projektu
-existujú 2 verzie: COCOMO 81 (waterfall - vodopád), COCOMO II (moderné SW procesy)
-*COCOMO 81*
-skladá sa z hierarchie troch podrobných a presných foriem, Organic, Semidetached, Embedded
-prvá úroveň Basic je dobrá na rýchly a hrubý odhad nákladov na softvér, presnosť je však obmedzená pre nedostatok faktorov
-druhá úroveň Intermediate – už berie do úvahy tieto faktory
-tretia úroveň Detailed – dodatočné vyúčtovanie pre vplyv jednotlivých fáz projektu

18. COCOMO II

-mnoho spoločností si ho osvojilo, je však len časťou toho čo je požadované pre doručenie sofistikovaného riešenia pre zúčastnené strany

-treba vypĺňať medzery, ktoré scrum ignoruje. Zas iné metódy sú rozličné čo môže byť neskôr mätúce pre obe strany

27. RAD

-dáva menší dôraz na plánovacie úlohy a väčší na vývoj

28. DSDM

-je projektový framework, hlavne použíteľný ako softvérová vývojová metóda

29. XP, párové programovanie

-XP (extreme – extrémne programovanie) – je agilná metodológia softvérového vývoju, ktorá predpisuje určité činnosti všetkým zúčastnením vývojového procesu. Činnosti sú ale vedené do extrému, vďaka tomu by sa malo toto programovanie lepšie prispôsobiť sa zmenám požiadavkám zákazníka a dodať softvér vo vyššej kvalite

30. TDD

-Test Driven Development je založený na malých stále sa opakujúcich krokoch vedúcich k lepšej efektivite celého systému. Jednoducho povedané najprv sa definuje funkcionalita, potom sa napíšu testy, ktoré túto funkcionalitu overujú a až tak príde na rad napísanie kódu a úprava toho kódu

31. FDD

-Feature Driven Development, základným modelom v tejto metodológii je doménový model na vysokej úrovni abstrakcie. Popisuje celý systém a slúži tak k minimalizácii problémov a spolupráci jednotlivých častí vytvorené rôznymi programátormi
-dá sa preložiť ako vývoj riadený užitočnými vlastnosťami.. nah

32. Prototypovanie

-tak dačo vykecať už z PRPS, nejaké tools, nejaké postupy, cez doménovú analýzu, konceptuálny model, scenáre, persony, protyp..

33. DAD

-Disciplined Agile Delivery
-poskytuje viac súdržný prístup k dosiahnutie doručenia agilného riešenia
-umožňuje zjednodušiť procesné rozhodovanie pri vývoji a správe softvéru; stavia na mnoho postupoch agilného softvérového vývoja viď Scrum, agilné modelovanie..
-rozvíja scrum pomocou agilného modelovania, XP,UP,Kanban..
-bol navrhnutý v IBM ale voľne dostupný a nevyžaduje žiadne IBM tools; je riadený cieľmi, škálovateľný a zohľadňuje podnik
-má životný cyklus založený na porovnávaní rizík a prínosov

-tri aspekty – review (posudok), inspection (inšpekcia), correction (oprava)
-inšpekcia – kontrola procesu, či je všetko vykonané tak ako by malo byť, zahŕňa posudok finálnych a pracovných verzií dokumentov
-posudok – kontroluje výstupy z procesov, z finálnych dokumentov
-oprava – zmena v procese alebo v dokumente

41. Dokumentácia požiadaviek

-príučky pomocou špeciálnej šamblóny
-požiadavky špecifikácie: CRC karty, formálne metódy, voľný text, tabuľky, UML diagramy, User stories (kto a čo v systéme má robiť)
-sú menej dôležité pri agilnom vývoji

42. Vývojový tím

-Tím – skupina ľudí, spolupráca, organizovanie tímu, motivácia
-Zákazník – podľa jeho chuti
-Soft skills – komunikácia, schopnosť pracovať skupine, osobnostné skills ako organizátor, líder, detektív, konzultant..

43. Zákazník a jeho roly
44. Manažment financií

-platy, čas zákazníka, náklad na SW/HW, poplatky

45. Manažment hardvéru a iných hmotných zdrojov

-administrácia, vývoj, skladovanie (databáza napr), servis a údržba

46. Manažment návrhu a implementácie

-správa vývoja
-systémový návrh, návrh modulov, kódenie, integrácia
-kvalita, dokumentácia a vývoj, zdroje: ľudia, softvér, hardvér, peniaze

47. Zabezpečenie kvality vo fázach návrhu a implementácie

-verification, overovanie – robíme to dobre?
-inspection (inšpekcia) – kontrolovanie procesov
-review(posudok) – kontrolovanie výstupov procesov
-simulation(simulácia) - testovanie programátorom, simulovanie operácii v inom prostredí
-correction(opravy) – zmeny v procese alebo dokumente
-documentation(dokumentácia) – vstupy pre testovanie

48. Dokumentácia návrhu a implementácie

-príručky podľa špecifických šablón
-vizuálne a textové dokumentácie: CRC karty, formálne metódy, tabuľky, uml diagramy, zdrojové kódy

-tak je tam taký graf v prednáškach ale kto vie na čo sa snaží poukázať, asi že požiadavky by sa mali potom otestovať abo nevieeem uš

56. Zabezpečenie kvality vo fáze testovania

-testovanie modulov – kvalita modulov a kódu
-integračné testovanie – kvalita rozhraní a ich implementácia
-testovanie systému – kvalita adaptácie na prostredie, kvalita konfigurácie
-akceptačné testovanie – kvalita systému, obchodná hodnota, odhad a validácia

57. Dokumentácia k testovaniu

-Reporty využívajúce špecifické šablóny
-vizuálna a textová dokumentácia – vložíme sufix alebo prefix TEST
-skúšobná hodnotiaca správa
-dôležité v agilnom vývoji

58. Retest

-testovanie pri ktorom beží prípad testu, ktorý zlyhal keď bol spustený aby overil úspešnosť nápravy

59. Regresný test

-testovanie skôr testovaných programov, aby zabezpečil, že vady neboli zavedené v nezmenených oblastiach, ako výsledok iných zmien. Vykonáva sa keď je zmenený softvér alebo prostredie

60. Retrospektíva

-štruktúrovaný spôsob ako zachytiť získané poznatky a vytvoriť špecifický plán v krokoch pre zlepšenie ďalších projektov alebo fáz v projekte
-retrospektívne stretnutie: stretnutie na konci projektu, počas ktorého členovia tímu vyhodnocujú projekt, lekcie, ktoré sa naučili a môžu byť použité v ďalšom projekte

61. Hodnotenie tímu a jednotlivcov

-osobný a profesijný rast
-Informovanosť tímu (povedomie tímu)
-teamwork – tímová spolupráca

62. Ako funguje CMMI?

-CMMI – Capability Maturity Model Integration – je model základných procesov v organizácií spolu s odporúčaním ako ich efektívne implementovať. Zahŕňa v sebe tri oddelené modely:
    -CMMI for development (CMMI-DEV) – vývoj produktov a služieb
    -CMMI for Services (CMMI-SVC) – služobné ustanovenia, manažment
    -CMMI for Acquisition(CMMI-ACQ) – prínos pre produkty a služby
-CMMI neponúka organizáciam certifikáty ale môžu byť organizácie ocenené

Čo je retestovanie(retesting)?

-spúšťanie testovacích prípadov, ktoré v minulosti zlyhali ale následkom zmien v kóde by už mali byť úspešné

-konfirmačné testovanie

Ktorá postupnosť činností vystihuje vodopádový model vývoja programu?

-špecifikácia, návrh, implementácia, testovanie

Na čo slúžia akceptačné testy?

-Overenie kvality systému ako celku, validácia a určenie jeho úžitkovej hodnoty

Ako definujeme chuť/túžbu zákazníka?

-FP(m+1) ~ FP(m)*1,01

Charakterizujte testovanie softvéru

-testovanie je spúšťanie programu za účelom nájdenia chýb

Čo je skratka COCOMO

-Constructive Cost Model

Aký bude odhad počtu vývojárov na projekte o veľkosti 1500 funkčných bodov na základe experimentálne overených pravidiel?

-10

Aké sú výhody zákazníka?

-nepoberá od nás výplatu

-je jediný, kto vie odhaliť určité chyby

Čo je cieľom statickej analýzy kódu?

-odhaliť inštrukcie, ktoré nikdy nebudú vykonané

-nájsť prázdne cykly

-nájsť redundatné deklarácie

Čo je validácia?

-proces na určenie toho, či vytvárame ten správny produkt

Aký bude odhad počtu ľudí zodpovedných za údržbu na projekte o veľkosti 1500 funkčných bodov na základe experimentálne overených pravidiel?

-3

Čo je korekcia?

-oprava chýb vo výsledkoch procesov

Čo je regresné testovanie?

-Testuje sa vplyv zmeneného kódu na predtým otestovaný kód (ktorý bol zväčša v poriadku