Testing strategies (CH 13)

Ką apibrėžia testavimo strategija?

Kaip atliekami testai.
Testuoti visą programą ar tik jos dalį?
Ar reikia ką tik atliktus testus pakartoti kai pridedami nauji komponentai didelėje sistemoje?
Kada įtraukti klientą?

1. Kad testavimas būtų efektyvus turi būti atlikta kodo peržiūra.
2. Testavimas prasideda nuo žemesnio lygio ir eina link aukštesnio.
3. Testavimo technikos pritaikomos pagal programavimo būdą ir laiką.
4. Testavimas atliekamas programos kūrėjo ir (dideliuose projektuose) nepriklausomos testavimo grupės.
5. Testavimas ir derinimas (angl. Debugging) yra skirtingos veiklos, tačiau derinimas turi būti pritaikytas bet kurioje testavimo strategijoje

Kodėl derinimas sudėtingesnis už testavimą?

1. Simptomas ir priežastis gali būti nutolę. Simptomas gali laikinai išnykti ištaisius kitą klaidą
2. Simptomą galėjo sukelti ne klaidos, pavyzdžiui, apvalinimo netikslumai. Simptomą galėjo sukelti žmogiškoji klaida, kuri nėra lengvai atsekama. Simptomas galėjo atsirasti dėl problemų susijusiu su laiku, o ne dėl apdorojimo problemų.
3. Gali būti sudėtinga atkartoti įvesties duomenis. Simptomai gali pasikartoti nereguliariai
4. Simptomas galėjo atsirasti dėl priežasčių, kurios atsirado užduotims vykstant atskiruose procesoriuose

Kodėl kai kurie vienetų testai būna pakeičiami integracijos testais?

Integration testing is a type of **testing** to check if different pieces of the modules are working together. **Unit testing** checks a single component of an application.

Kada sistema yra baigiama testuoti?

1. Testavimas trunka iki programos atsisakymo
2. Testuojama, kol baigiasi laikas arba pinigai
3. Statistiniai metodai

Estimation (CH 23)

Kokie yra pagrindiniai projektų vertinimo būdai? Trumpai apibūdinti

LOC (angl. Lines of code)
FP (angl. Functions points)
Procesais paremtas projektų vertinimas
Vertinimas paremtas panaudos atvejais

Kaip vykdomas judraus (angl. agile) kūrimo projektų vertinimas?

1. Kiekvienas naudotojo scenarijus vertinamas atskirai.
2. Scenarijus suskaidomas į programinės įrangos inžinerijos užduočių rinkinį reikalingą, kad scenarijus būtų įgyvendintas.
3. Kiekviena užduotis vertinama atskirai.
4. Kitas būdas scenarijaus apimtį vertinti naudojant LOC, FP ar kitą apimties matavimo būdą.
5. Kiekvienos užduoties vertinimas sumuojamas, kad būtų sukurtas bendras scenarijaus vertinimas.
6. Arba scenarijaus apimties įvertis paverčiamas pastangomis (angl. effort) naudojant istorinius duomenis.

7. Visų scenarijų kurie turi būti įgyvendinti vertinimai sumuojami, kad gauti reikiamų pastangų įvertinimą

Funkcijų taškai (angl. functions points), kas tai?

Information fomain value	Optimistic	Likely	Pessimistic	Estimated	Weight	FP count
External input						
External output						
External inquiries						
Logical files						
Interface files						
Count total						

Software engineering practice (CH 5)

?

Creating an architectural design (CH 10)

Ar architektūros projektavimas atneša naudos kuriant programų sistemas? Pagrįsti atsakymą.

1. Palengvina/sukonkretina komunikaciją projekto klausimais tarp suinteresuotų šalių (stakeholders)
2. Suteikia projekto apibendrintą suvokimą
3. Dizaino sprendimai diktuoja projekto vystymasi

Kuo skiriasi architektūros stilius (style) ir architektūros šablonas (pattern). Pateikite pavyzdžių.

1. Architektūros požymis skirtas apibūdinti sistemos kategorijai, apimančiai
2. An Architectural Pattern is a way to implement an Architectural Style; Arch. šablonas siūlo architektūrinio sprendimo pagrindą panašaus konteksto ar žanro užduočių aibei
3. An architectural Style is a specialization of element and relation types, together with a set of constraints on how they can be used. On the other hand, an architectural Pattern expresses a fundamental structural organization schema for software systems. It provides a set of predefined subsystems, specifies their responsibilities, and includes rules and guidelines for organizing the relationships between them.

Styles

- Client-server (Centralizuotų duomenų)
- Call-return

- Layered
- Peer-to-peer
- MVC

Pattern

Architectural patterns for software define a specific approach for handling some characteristic of the system. For example “Kitchen”.

- Teisių valdymo (access control) (Windows vs Unix)
- Lygiagretumo (concurrency) (Round robin vs Priority)
- Paskirstymo (distribution) (Centralized, Decentralized)
- Patvarumo (persistence) (ORM,DAO)

Kam (kurioms suinteresuotoms šalims) architektūros aprašas gali būti skirtas?
Programuotojams
Architektams
Klientams
Tech. Įrangos tiekėjams

Kokie yra pagrindiniai architektūros konteksto komponentai?

Vyresniosios sistemos (superordinate)
Pavaldžios sistemos (subordinate)
Aktoriai (actors) vartotojai
Vienalygės sistemos (peers)

Risk management (CH 25)

Kuo skiriasi rizikos valdymo reaktyvi ir proaktyvi strategijos?

•Reaktyvi strategija:

Projektas stebimas ir jei atsiranda problemos, komanda bando jas spręsti spontaniškai –įsijungia gaisro gesinimo režimas. Jei nepavyksta problemos išspręsti greitai ar tinkamai –iškyla didelis pavojus visam projektui.

•Proaktyvi strategija:

Prasideda prieš darbų vykdymo pradžią. Identifikuojama potenciali rizika, įvertinama jos tikimybė ir poveikis, įvardintos rizikos surūšiuojamos, sukuriamas rizikos valdymo planas.

Kokie yra rizikos analizės ir valdymo žingsniai?
1.Rizikos identifikavimas
2.Rizikos įvertinimas
•Kiekvienos rizikos analizė(įvertinama rizikos tikimybė ir poveikis, t.y. žala, kurią ji gali padaryti)
•Rizikos rūšiavimas(suskirstoma pagal tikimybę ir poveikį)
3.Sudaromas planas rizikoms su didele tikimybe ir poveikiu valdyti
4.Vykdomas rizikos mažinimas, stebėsena ir valdymas

Kas yra rizikos įvertinimas? Kaip sudaroma rizikos lentelė?
RE – P x C
RE – risk exposure
P – rizikos tikimybė
C – projekto išlaidos rizikos įvykio atveju
Lentelė:

Risk	Category	Probability	Impact	RMMM

Kas yra rizikos mažinimas, stebėsena ir valdymas (angl. RMMM)? Kokie dokumentai gaunami taikant šią strategiją? Mitigation,Monitoring,Management
Mitigation: Mažinimo strategijos parengimas
Monitoring: Kaip keičiasi projekto metu
Management: Konkretūs veiksmai, jeigu rizika įvyktų

Gaunami RIS (Risk Information Sheet)
Atskirai dokumentuojama kiekviena rizika
Rizikos apibrėžimas:
Kontekstas (kada ir kaip rizika gali įvykti)
Mitigation.monitoring
Management
Dabartinė stadija (kas daroma dėl šios rizikos)

Testing tactics (CH 14)
Ką vadiname baltos dėžės testavimu?
Testuojamos vidinės konstrukcijos, tokios kaip ciklai ar loginės struktūros
Ką vadiname juodos dėžės testavimu?
Testuojami funkciniai reikalavimai.
Ką parodo nepriklausomų kelių skaičius?
Testų skaičių, reikalingų padengti programą
Kas yra ciklomatinis sudėtingumas?
Programos atvejų kelių grafo nepriklausomų kelių skaičius
Regionų skaičius
Predikatinių viršūnių skaičius + 1
Briaunos – Viršūnės + 2

Process models (CH 3)

Kas yra proceso modelis?

Proceso aprašas apimantis veiksmus, užduotis, kuriuos reikia atlikti prog. įr. Kūrime.

Kokius žinote klasikinius proceso modelius? Kokie yra kiekvienam iš jų būdingi bruožai?

Krioklio

Linijinis judėjimas nuo vienos veiklos prie kitos. Reikalavimas apibrėžti ir stabilūs

Inkrementinis

Programa pateikiama dalimis. (Gali būti lygiagrečiai)

Pirma dalis atitinka pagrindinius reikalavimus, bet stokoja papildomo funkcionalumo.

Reikalauja mažiau žmonių.

Greičiau galima pateikti riboto funkcionalumo programą.

Iteratyvus

Pateikiamas programos prototipas

Kiekvienos iteracijos metu programa yra pertrarkoma ar tobulinama

Unifikuotas procesas

Iteratyvus ir inkrementinis

Visos fazės gali persidengti ir vykti vienu metu (skirtinguose inkrementuose)

Kuo inkrementinis proceso modelis skiriasi nuo iteratyvaus?

Žr viršui

Kaip suprantate iteratyvų inkrementinį proceso modelį?

Turi ir inkrementinio ir iteratyvaus modelių savybių. T.y. programos kūrias vykdomas dalimis, Kiekviena fazė vykdomas iteratyviai, tačiau po jos turi būti naujas *inkrementas (release)*, kas yra prieš tai buvusios programos patobulinimas.

Performing user interface design (CH 12)

Kokius sąsajų tipus galima atskirti?

Tarp programos komponentų

Tarp komponentų ir kitų objektų už programos ribų

Tarp vartotojo ir programos

Kokias "Auksines taisykles" vartotojo sąsajoje išskyrė autorius Theo Mandel?

Suteikti naudotojui kontrolę

Sumažinti naudotojo “cognitive load”

Interfeisas turi būti pastovus

Kokie gali būti analizės ir projektavimo modeliai?

Vartotojo modelis – apie vartotojus, kokie jie yra , jų patirtis su sistema (pradedantysis, vidutinis, veteranas)

Dizaino modelis – apie visą sistemą

Sistemos suvokimo modelis – kaip naudotojas naudojasi sistema

Įgyvendinimo modelis- kaip sistema bus sukurta

Koks yra vartotojo sąsajos kūrimo procesas?

Spiralės būdas

Apima 4 veiklas:

Vartotojo, užduočių ir aplinkos analizė

Vartotojo sąsaja

Vartotojo sąsajos įgyvendinimas

Vartotojo sąsajos tikrinimas

Quality management (CH 26)

Kas yra kokybė ir kaip užtikrinti, jog galutinis produktas būtų kokybiškas?

Aiškiai apibrėžti funkciniai ir eksploataciniai reikalavimai, aiškiai dokumentuoti plėtros standartai ir netiesioginės charakteristikos, kurių tikimasi iš visos profesionaliai sukurtos programinės įrangos. Programinei įrangai iškelti reikalavimai yra pagrindas, kuriuo remiantis vertinama kokybė. Reikalavimų neatitikimas yra kokybės trūkumas.

Kokybės-naudos analizės

Lyginamoji analizė

Skirtumas tarp defekto ir klaidos, bei paaiškinti kas įtakoja kokybės užtikrinimo išlaidų augimą?

Defektas – praleista klaida išleistame produkte. Išlaidos auga priklausomai kokiame etape buvo aptikta klaida/defektas

Kaip vyksta formalios (officialios) techninės peržiūros? Paaiškinkite procesą.

3-5 žmonės

Ne ilgiau kaip dvi valados žmogui

posėdis trunka ne ilgiau kaip 2 valandas

Prežiūrų raportavimas bei įrašų išsaugojimas.

Vienas iš peržiūroje dalyvaujančių rūpinasi įrašų išsaugojimu:

Kas buvo peržiūrėta

Kas vykdė peržiūrą?

Kas buvo aptikta/išvados.

Programinės įrangos kokybės užtikrinimas, kodėl jis reikalingas?

Requirement

Design

Code

Quality control effectiveness