ŠIAULIŲ UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
INFORMATIKOS KATEDRA

Tomas Demenis
Informatikos magistratūros 2 kurso studentas

Nuotolinių studijų kurso *Programavimas grafinėje terpėje* reinžinerija

MAGISTRO DARBAS

Mokslinis vadovas
Dėst. Lina Tankelevičienė

Šiauliai, 2008 m.

Tvirtinu, jog darbe pateikta medžiaga nėra plagijuota ir paruošta naudojant literatūros sąraše pateiktus informacinius šaltinius bei savo tyrimų duomenis.

Tomas Demenis .....
(parašas)

Turinys

Įvadas

Atsirandant naujai informacijai, sparčiai keičiantis žiniomis, atradimais, naujais pasiekimais, aktuali informacija privalo būti nuolatos atnaujinama. Tai yra ypač aktualu edukacinei medžiagai. Mokymo kursų medžiaga yra fundamentali informacija, siekiant lavinti studentus. Norint užtikrinti informacijos naujumą – kursas privalo būti pertvarkytas. Informacija turi būti pateikiama aiškiai, neapkraunant studijuojančio bereikalinga papildoma informacija. Pateikiama modelį galima naudoti ir kitų nuotolinių kursų pertvarkymui.

Sistemingas darbas, vadovaujantis planu, yra svarbus kriterijus norint pasiekti tikslą. Yra galimybė išvengti nenumatytų problemų, viskas atliekama laikantis nustatytos tvarkos.

Darbo tikslas

Darbo tikslas yra sukurti ir pritaikyti reinžinerijos modelį nuotolinių studijų kursui „Programavimas grafinėje terpėje“.

Darbo uždaviniai

Tikslui pasiekti buvo iškelti tokie uždaviniai:

1. Išanalizuoti reinžinerijos sąvoką ir jos naudojimą informatikoje bei kitose srityse;
2. Išsiaiškinti NMK struktūrą ir esmines dalis;
3. Išanalizuoti NMK rengimo pagrindinius aspektus;
4. Atlikti panašaus studijų kurso, dėstomo užsienio universitetuose lyginamąją analizę;
5. Pateikti NMK teorinį reinžinerijos modelį;
6. Atlikti praktinę NSK temų reinžineriją.

Tyrimo objektas: reinžinerijos metodikos panaudojimas nuotolinių studijų srityje.

Tyrimo metodai

Darbo metu buvo naudoti įvairūs tyrimo metodai:

- Analizė: nagrinėti NSK rengimo principai, susipažinta su rengimo tvarka, peržiūrėti kiti nuotolinio mokymo kursai, išnagrinėta NMK struktūra;
- Lyginamoji analizė: atlikta panašių kursų, dėstomų užsienio universitetuose, analizė;
- Modeliavimas: sukurtas NSK reinžinerijos modelis; pagal jį modeliuota pertvarkomų temų struktūra;
- Techninės literatūros analizė: naudota rengiant 2 temų medžiagą.

Darbo mokslinė ir praktinė nauda

Darbo mokslinė nauda pasireiškia tuo, kad teorinis nuotolinių studijų kurso reinžinerijos modelis gali būti panaudojamas panašiems informatikos ar kitų mokslo sričių kursams

pertvarkyti. Darbas atliktas sistemingai: analizuota darbinė sritis, pasiūlytas apibendrintas modelis ir jis pratestuotas konkrečioje situacijoje. Taigi, pasiūlytas teorinis reinžinerijos modelis remiasi analizės rezultatais ir yra verifikuotas.

Darbo praktinė nauda – atnaujintas studijų kursas, kuris suteikia studentams galimybę lengviau ir greičiau įsisavinti studijų medžiagą. Pateikiama nauja medžiaga savo kokybe nenusileidžia užsienio universitetuose dėstomiems moduliams.

Darbo vadovas: dėst. Lina Tankelevičienė .....

(parašas)

I. Reinžinerijos samprata vadyboje ir informatikoje

Reinžinerija įvairiose srityse

Reinžinerija gali būti taikoma įvairiose srityse. Taikant reinžinerijos principus galima pasiekti veikiančios sistemos efektyvumo ir našumo padidėjimo. Dauguma, ypatingai senų, sistemų veikia neracionaliai, naudoja daug išteklių, jas apdoroti reikia daug laiko, vartotojui ar sistemos dalyviui nėra patogu naudoti jas. Norint pertvarkyti dalį ar visą sistemą iš esmės yra naudojami įvairūs reinžinerijos modeliai. Lentelėje pateikta dažniausias pasitaikantys įmonės ar organizacijos reinžinerijos apibrėžimai.

Apibrėžimas	**Paaiškinimas**
Reinžinerija – išskirtinai reiškia darbų atlikimą kitaip. Tikslas yra pajungti žmones, technologijas ir procesus į racionalesnę veiklą tam, kad geriau paremtumėme verslo strategijas, ir verslo tikslus šiandieninėje aplinkoje.	Senos verslo idėjos ir praktika šiandieninėje ekonominėje erdvėje yra nebekonkurencingos. Reikia peržiūrėti sistemą ir atlikti reikiamus pakeitimus.
Reinžinerija yra tiltas, naudojamas senai programinei įrangai migruoti į naują organizacijos aplinką.	Sparčiai vystantis IT technologijoms, reikia pertvarkyti bei adaptuoti esamą programinę įrangą ir duomenų bazes.Šiuo atveju sistemos reinžinerija užtikrina vartotojui galimybe pasiekti duomenis ir ateityje.
Reinžinerija tai esamos sistemos tyrimas ir permainų kūrimas bei palaipsnis naujovių integravimas į sistemą. (http://www.iste.uni-stuttgart.de/ps/reengineering/terminology.html)	Palaipsnis integravimas taikomas tada, kai pertvarkymo organizatoriai yra atsargūs ir baiminasi radikaliai viską keisti. Toks būdas yra vienas iš faktorių lemiančių nesėkmingą reorganizavimą.
Populiarus korporatyvinis terminas, naudojamas pergalvojimo veiksmui ir kompanijos procesų restukturizavimui įvardinti prieš pakeičiant kompiuterių sistemas yra reinžinerija. Mokoma to, jog jei neapgalvosime visko pradžioje, pakeitus sistemą problema bus tokia pati, tik sugeneruota daug greičiau. (http://www.fas.org/spp/military/docops/usaf/2020/app-v.htm)	Pertvarkymas turi būti gerai suplanuotas ir numatytos galimos grėsmės. Taikant nepatikrintus metodus ir modelius kyla tikimybė pabloginti esamą situaciją įmonėje.

Reinžinerija (reengineering) - egzistuojančios sistemos analizės ir modifikavimo procesas, atliekamas tikslu iš esmės pertvarkyti tą sistemą. Atvirkščioji inžinerija (reverse engineering) - dirbtinės egzistuojančios sistemos analizės procesas, atliekamas tikslu atkurti jos projektinę dokumentaciją. Sistemos projekto atkūrimas (design recovery) - ta atvirkščiosios inžinerijos proceso dalis, kuria, pasinaudojant žiniomis apie dalykinę sritį, išorine informacija ir dedukcija arba kitokiais samprotavimo būdais, iš tiesioginių nagrinėjamos sistemos analizės duomenų stengiamasi išgauti prasmingas aukštesniųjų lygmenų abstrakcijas, aprašančias tos sistemos sandarą ir veikimą. (http://ledas.dtiltas.lt/MIF/3_sem_egz/egz_PSI/PSI_ats_mano_norm.doc)	Kai kur yra skiriamos trys inžinerijos kryptys.

Lentelė Nr. 1 Reinžinerijos apibrėžimai

Egzistuoja ir daugiau reinžinerijos apibrėžimų, keičiantis projektui, sričiai, apimčiai ir specifikai. Tačiau bendru atvėju Reinžinerija yra esamo produkto perdirbimas panaudojant naujas idėjas, įrankius, informaciją, patirtį.

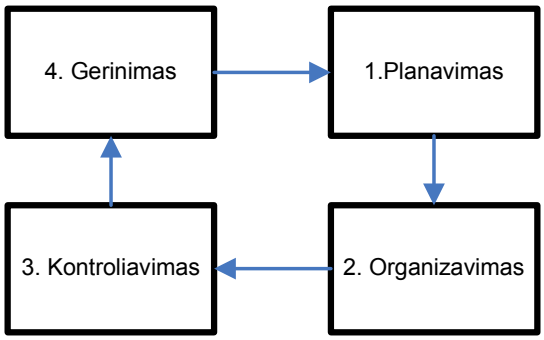
Pagrindinės nesėkmingo reorganizavimo priežastys

- bandymas tik koreguoti procesą, o ne iš esmės jį pertvarkyti;
- nesugebėjimas susikoncentruoti ties verslo procesais;
- ne visų reorganizavimo aspektų įvertinimas;
- pasitenkinimas mažais pasiekimais;
- skubotas pradėtos reorganizavimo apimties mažinimas;
- pradinis problemų reikšmingumo sumenkinimas;
- trukdymai, atsirandantys dėl įmonės nepakankamos verslo kultūros
- bandymas pasiekti, kad reorganizavimas eitų iš apačios į viršų;
- reorganizavimo lyderio nekompetentingumas;
- reorganizavimo resursų neargumentuotas taupymas;
- įmonė nelaiko reorganizavimo svarbiu prioritetu;
- reorganizavimo resursų "išbarstymas" tarp daugelio projektų;
- nesugebėjimas atskirti reorganizavimo nuo kitų verslo valdymo gerinimo programų;
- susikoncentravimas į procesų schemas, o ne į jų įdiegimą;
- bandymas įgyvendinti reorganizavimą, nieko „nenuskriaudžiant“;

- atsitraukimas, neatlaikius bendradarbių spaudimo;
- reorganizavimo darbų uždelsimas.

Įmonių vadovai nespėja efektyviai atlikti visų 4 valdymo funkcijų, todėl jiems yra reikalingi papildomi atsakingi asmenys, priešingu atveju pokyčiai įmonėse vyksta lėtai. Reorganizavimas užtikrina, kad pagrindinės valdymo funkcijos bus vykdomos efektyviai:

- planavimas – jo metu įmonės siekiai bus išreiškiami ilgesniam ar trumpesniam laikotarpiui;
- organizavimas - kiekvienam įmonės darbuotojui ir kitam verslo dalyviui aiškiai pasakoma, ką privaloma padaryti;
- kontrolė - veikla koordinuojama tokiu būdu, kad , laiku pašalinami nukrypimai nuo iš anksto numatytos veiklos programos;
- gerinimas - kontrolės metu gautų duomenų pagrindu atliekami pakeitimai procesuose.



Paveikslėlis 1: Reorganizavimo procesų ratas.

Verslo procesų reinžinerija

Verslo procesų reinžinerijos (BPR – business process reengineering) idėja 1990 metais buvo pristatyta dviejų amerikiečių, James Champy ir Michael Hammer. Tai yra apibrėžiama kaip reikšmė kurios dėka: organizacija gali pasiekti radikalių pasikeitimų daugybės įrankių pritaikymo veikloje ir technikos, kuri koncentruojasi į biznį kaip į sąryšį su pirkėjais, o ne kaip į organizacijos funkcijas. Atlikimas gali būti matuojamas išlaidomis, ciklo laiku, paslaugomis ir kokybe. Pirminis BPR tikslas yra padidinti konkurenciją operacijų tinkle supaprastinant, įvedant daugiau pasitikėjimo ir daugiau produktyvesnių procesų. Ji buvo pritaikyta darbo ir kapitalo reikalaujančioje industrijose, kaip automobilių produkcija, farmacija ir taip pat paslaugų sektoriuose (draudimas ir bankininkystė). Biznio procesų reforma pabrėžia frazę „sudaužyk Kiniją“, reiškiantį, kad firma yra pakankamai drąsi ignoruoti ar net sulaužyti procesų projektų veikimą, kuris vyko anksčiau, ir pradėti viską iš naujo.

Verslo projektų reinžinerijos idėja galima iliustruoti tikru pavyzdžiu. AT&T jėgainių sistemos Dalase, Teksase, gamina paprastai jungiamas energijos tiekimas konkurencingoje rinkoje. Originalus procesų projekto procesas kiekvienam pirkėjui, nuo pasiūlymo iki baigtos produkcijos pristatymo, buvo apibūdintas kaip 42 skirtingų susitikimų darymas. Detali analizė parodė, kad produktas nebegali būti konkurencingas dėl išlaidų ir priedo, produkto pristatymas klientui vidutiniškai užtrukdavo 53 dienas. Kompanija visiškai pataisė procesų projektavimą, sukuriant daugiafunkcines projektų komandas ir buvo galima vykdyti standartinius projektus išlaikant energijos aprūpinimą paklausų. Standartizuojant projektų komandos buvo užtikrintos dėl turimo dalių prieinamumo, nors tie patys komponentai buvo paklausūs kitiems standartiniams surinkėjams. Naujajame procesų projektų konfigūracijoje, senų veiksmų skaičius buvo išlaikytas, bet paskirstytas projektų kuopelėm, kurios yra paskirtos grupės susidarančios iš elektronikos, mechanikos, testavimo ir kokybės inžinierių, plius modelio statytojo ir projekto vadybininko, ir jos dirba uždaroje harmonijoje siekiant sumažinti našumo laiką. Kiekvienas komandos narys tapo visiškai atsakingas už savo darbą, taip atsisakant patikrinimų, peržiūrų ir formalių susitikimų. Dėka to liko tik 17 formalių žingsnių ir vienas susirinkimas. Šio BPR rezultatas- pristatymo laiko sumažinimas iki penkių dienų, arba 90% mažesnis pristatymo laikas už pirminį.

Programinės įrangos keitimas, tobulinimas, reinžinerija

Programinė įranga - tai kompiuterio vykdomų instrukcijų seka, skirta tam tikriems veiksmams atlikti. Dažniausiai tokia įranga parašoma naudojant programavimo kalbas, o vėliau kompiliuojant ar interpretuojant parašytą kodą. Programinė įranga yra kuriama konkrečiam darbui ar veiksmui atlikti. Laikui bėgant keičiasi užduotys, apimtis, iškyla kitos problemos. Programinę įrangą tenka keisti. Tai tam tikro lygio programinės įrangos reinžinerija.

Programinę įrangą neišvengiamai tenka keisti:

- Programinės įrangos naudojimo eigoje, jai iškyla nauji reikalavimai;
- Verslo aplinkos pasikeitimas;
- Tenka taisyti klaidas;
- Reikia išnaudoti pasirodžiusios naujos aparatūros privalumus;
- Gali tekti gerinti vykdymo spartą arba patikimumą .

Palaipsnis programinės įrangos atnaujinimas gali vykti keliais būdais. Šiame etape dideli sistemos architektūros pakeitimai dažniausiai nedaromi. Vykdoma mažais etapais,

nesugriaunant jau esamos visos architektūros. Atnaujinimas realizuojamas arba modifikuojant jau egzistuojančius sistemos komponentus, arba pridedant naujus. Jei norima, kad sistemos naudingumas duotoje aplinkoje nesumažėtų, ji turi būti palaikoma

Palaikymas yra neišvengiamas. Egzistuoja keli palaikymo ir keitimo būdai. Naudojant naują sistemą, gali būti pastebėtos klaidos. Tokiu atveju yra taisomos programinės įrangos klaidos. Ištaisius klaidas, sistema atitinka reikalavimus ir ją galima naudoti. Kita situacija – programinė įranga turi veikti kitoje aplinkoje (kitame kompiuteryje ar operacinėje sistemoje), nei buvo įdiegta pradžioje. Trečias būdas – sistemos funkcijų modifikavimas arba praplėtimas. Jis yra taikomas tada, kai sistema atnaujinama siekiant patenkinti naujus reikalavimus. Atnaujinimai dažniausias yra atliekami atskiriems moduliams.

Keičiant programinę įranga, kartais gali būti keičiama ir architektūra. Pvz.: daugelio liktinių sistemų architektūrą reikia keisti iš centralizuotos į kliento-serverio tipo architektūrą.

Motyvai atnaujinti:

- Aparatūrinės įrangos kainos. Serveriai yra pigesni negu didieji kompiuteriai *(mainframe);*
- Vartotojo sąsajos perspektyvos. Vartotojai nori grafinės sąsajos
- Nuotolinis priėjimas prie sistemų. Vartotojai nori prisijungti prie sistemos iš kitų, geografiškai nutolusių, kompiuterių.

Sistemos amžius

Kuo sistema senesnė, tuo sunkiau bus pakeisti jos architektūrą, nes jos struktūra bus pažeista. Gali būti nebenaudojama programavimo kalba, kuria parašytas kodas, darbiniai failai, kt. Įtakos atnaujinimo galimybėms turi sistemos struktūra. Kuo geriau sistema suskaidyta į modulius, tuo lengviau pakeisti architektūrą. Jei loginiai sluoksniai sumaišyti, perėjimas bus sunkus. Turint aiškų modulių aprašymą ir žinant kaip jie sąveikauja tarpusavyje yra žymiai lengviau atlikti pakeitimus.

Vartotojo sąsajos reorganizavimas

1. Strategija: *Realizavimas langų valdymo sistemoje*

Privalumai: Prieinamos visos sisteminės grafinės vartotojo sąsajos funkcijos. Geresnė vartotojo sąsajos veikimo sparta.

Trūkumai: Priklauso nuo platformos. Gali būti sunku pasiekti nuoseklumą.

2. Strategija: *Realizavimas tinklapio naršyklėmis*

Privalumai: Nepriklauso nuo platformos. Mažesni apmokymo kaštai, nes vartotojas susipažinęs su voratinklio galimybėmis. Lengviau pasiekti sąsajos nuoseklumą.

Trukumai: Lėčiau veikia, galimybes riboja naršyklių funkcionalumas.

Kartais neįmanoma keisti programinės įrangos architektūros, programinės dalies ar vartotojo sąsajos. Yra daug faktorių kurie lemia sistemos reinžinerijos galimybes. Todėl yra būtina prieš pradedant daryti pakeitimus atlikti nuoseklią analize ir įvertinti turimus resursus ir tikslo pasiekimo galimybes bei pasekmes.

IS planavimas, kūrimo strategija ir metodologija

Sukurti informacinę sistemą, pilnai tenkinančią vartotojo poreikius yra labai sudėtingas uždavinys. IS projektams būdinga, kad jų sukūrimui ir įdiegimui kaip taisyklė yra išleidžiama žymiai daugiau pinigų negu buvo planuota ir paprastai darbus niekada nespėjama baigti laiku. Ir netgi tada, kai sistema būna visiškai užbaigta, paaiškėja, kad ji neatlieka viso to, ko norėta, o vadybininko darbas kartais tampa dar sudėtingesnis, nei prieš sistemos įdiegimą. Tačiau IS vistiek kuriamos ir diegiamos.

IS kūrimo etapų (programavimo, testavimo, reorganizavimo) metu remiantis prieš tai paruoštomis specifikacijomis sukuriama/pakeičiama veikianti IS.

Programavimas

Jo metu specifikacijos pervedamos į programinį kodą. Remiamasi specifikacijomis kiekvienam programiniam moduliui, failams, duomenų perdavimams, ataskaitoms.

Testavimas

Etapo metu patikrinama ar sistema veikia gerai (esant žinomoms situacijoms). Apie 50% programų projektavimui skirto biudžeto gali būti skiriama testavimui. Duomenys testavimui turi būti kruopščiai paruošti, o rezultatai aiškiai matomi.

Reorganizavimas

Tai perėjimo nuo senos prie naujos IS procesas.

Šis procesas gali būti vykdomas įvairiai:

- lygiagrečiai – leidžiamos abi sistemos, kol įsitikinama, kad nauja vykdo viską gerai (tai saugiausias būdas);
- tiesioginis pakeitimas – vieną dieną pradeda veikti tik nauja sistema. Toks reorganizavimas pigesnis, tačiau rizikingas
- etapinis pakeitimas – naujoji IS įtraukiama etapais pagal atskiras IS funkcijas.

Kaip ir testavimui, reorganizavimui sudaromas reorganizavimo planas(tvarkaraštis). Reorganizavimo metu turi būti numatytas darbuotojų apmokymas dirbti su naująja sistema.

Etapo pabaiga – dokumento, parodančio kaip sistema dirba (tiek iš vartotojo pozicijų, tiek techninių savybių) paruošimas.

Gamyba ir palaikymas

Tai etapas kuomet sistema yra pilnai instaliuojama ir baigiamas reorganizavimo etapas. Net ir šio etapo metu sistema stebima tiek vartotojo tiek techninių specialistų. Jei pastebimas modifikacijų poreikis, jie atliekami. Šie veiksmai vadinami sistemos palaikymu.

Taigi susumuojant sistemų kūrimą galime pateikti tokią lentelę:

Etapas	Paskirtis
Sistemos analizė	Apibrėžti problemas Numatyti sprendinį Nustatyti informacijos poreikius
Sistemos projektavimas	Paruošti loginio projekto specifikacijas Paruošti fizinio projekto specifikacijas Vadovauti fiziniam sistemos realizavimui
Programavimas	Projekto specifikacijų pervedimas į programinį kodą
Testavimas	Atskirų dalių testavimas Sistemos testavimas Galutinis testavimas
Reorganizavimas	Reorganizavimo planavimas Dokumentacijos paruošimas Vartotojų apmokymas
Palaikymas	Sistemos darbas Sistemos modifikavimas

Lentelė nr. 2 : IS kūrimo etapai

Atliekant sistemos reorganizavimą, visi etapai bent jau iš dalies yra paliečiami ir pereinama per juos. Iškilus pertvarkymo poreikiui, vėl yra atliekama sistemos analizė, projektuojami pakeitimai, programuojamos dalys, jos testuojamos.

II. Nuotolinio mokymo kurso rengimo ir modifikavimo kai kurie aspektai

Nuoseklumo principo realizavimas sekoje tikslai-medžiaga+veiklos-vertinimas

Trigubo nuoseklumo principas apibrėžia nuoseklumą tarp tikslų, metodų ir vertinimo (Kovertaite, 2006). Yra skiriami trys mokymosi tikslai: 1) Atlikimas: ką studentai sakys, darys ar pasieks; 2) Specialios sąlygos: kokiomis darbo sąlygomis ir kokius įrankius naudojant yra atliekamos užduotys. 3) Kriterijai: standartinis ar reikalaujamas lygis pasiekti tikslui. Turint tikslus reikia nustatyti vertinimo metodus. Vertinimo metodai turi padėti nustatyti tikslų pasiekimą. Studentai turi gebėti įsisavinti medžiagą, atlikti praktiką ir gebėti panaudoti viską gyvenime ir darbe.

Pagrindas, kuriuo paremtas nuotolinio mokymo kurso kūrimas susideda iš elementų: a) Tikslai; b) Mokymo/mokymosi metodai (apimant abu: resursus ir veiklas); c) vertinimo metodus. Šis skaidymas yra atliekamas aukštesniame abstrakčiame lygyje. Technologiniame lygyje kursas gali būti analizuojamas kaip mokymosi ir informacijos objektų rinkinys, kurie yra integruoti į pamokas, temas, sekcijas ar kitus struktūrinius vienetus. Taigi šie objektai turi būti pertvarkyti.

Nuotolinėse studijose taikomi įvairūs mokymo metodai. Todėl skirtingų sugebėjimų besimokantieji gali pasirinkti priimtiniausią mokymosi būdą.

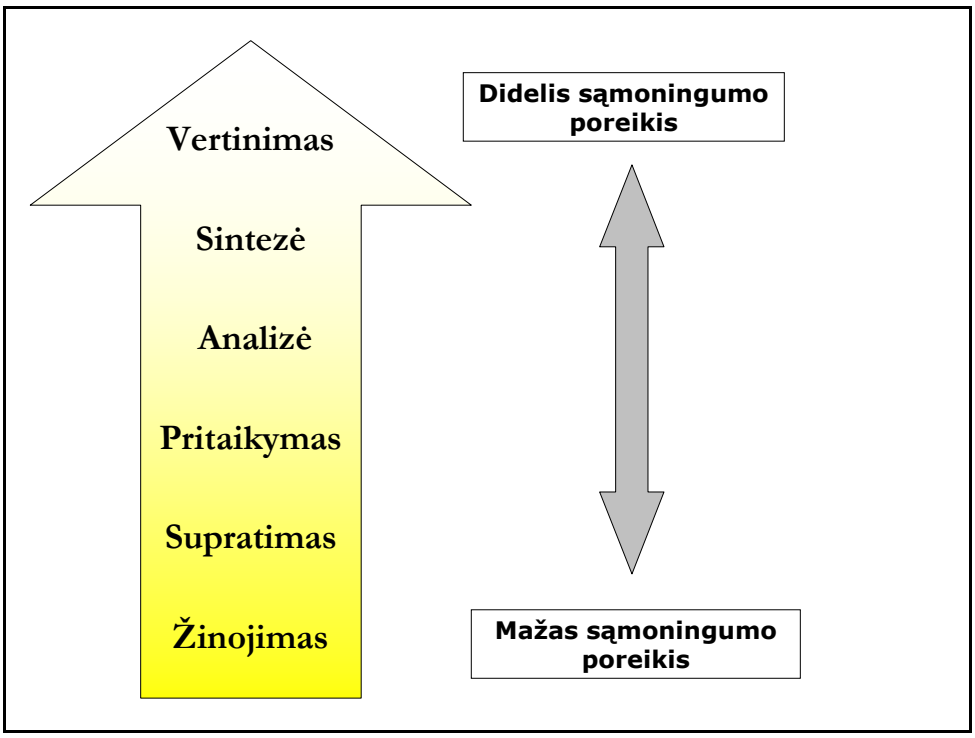
Metodas	Kas tai ir kodėl tai naudojama
Savarankiškas mokymasis	• Besimokantieji savarankiško mokymosi metodu, mokosi toje vietoje, kur jiems patogiausia: namie, darbe, bibliotekoje… Bendravimas su instruktoriumi yra asinchroninis. Kai besimokančiajam reikia konsultacijos jis gali parašyti e-paštu ar pasinaudoti kitomis komunikacijos priemonėmis. Šiuo atveju instruktorius atsakymą pateikia po tam tikro laiko, kuris dažnai yra aptariamas prieš studijas. Besimokantieji tuo pat metu gali studijuoti skirtingus dalykus priklausomai nuo jų poreikių, tikslų, mokymosi tempo, gebėjimų ir kitų parametrų. Besimokantieji gali pradėti ir baigti mokymąsi skirtingu momentu. Jie mokosi tada kai yra patogu. • Šis metodas patogus tiems besimokantiems, kurie turi pakankamai savi-disciplinos ir motyvacijos mokytis savo nusistatytu tempu.
Akis į akį	• Tokios sesijos metu bendram darbui apjungiami vienas instruktorius ir vienas besimokantysis. Kartais besimokančiųjų gali būti ir daugiau, bet tokiu atveju jų poreikiai turi sutapti. • Šis metodas naudojamas tuomet kai besimokančiajam reikia išmokti temą, kurios sėkmingam išaiškinimui neužtenka pagrindinių metodų ar tuomet kai tema turi būti

	paaiškinta kitu būdu.
Statuso įvertinimo ir bendradarbiavimo metodas	• Besimokantieji gali mokytis vieni iš kitų. Jie gali mokytis iš kitų besimokančiųjų klaidų ir gerų idėjų. Besimokantiesiems pateikus šią informaciją, tema tampa visiems geriau suprantama. Pateikiantieji informaciją įgauna gilesnį suvokimą, o kiti gauna informaciją kitokiu būdu ir kitais žodžiais, kas sąlygoja geresnį supratimą. • Šis metodas pagrinde taikomas tuomet kai besimokantiesiems reikia spontaniškai ir neformaliai pateikiamų paaiškinimų iš kieno nors kito ir kitais žodžiais. Šis metodas gali būti taikomas kaip pagrindinio metodo papildymas. Dažnai besimokantieji padalinami į nedideles grupes ir gauna specialias užduotis. Jie mokosi kaip suvokti ir pateikti įvertinimus. Jie gauna atsiliepimus apie savo darbą ir mokosi apginti savo idėjas.
Grupinis mokymasis	• Besimokantieji mokosi vienas iš kito. Jie pateikia savo informaciją, klausimus ir atsakymus tam skirtoje vietoje tinkle. Paprastai diskusijas inicijuoja ir joms vadovauja instruktorius. • Šis metodas naudojamas, kai reikia papildomos informacijos tam tikrais klausimais.
Seminarai	• Seminarai sudaro galimybę teoriją pritaikyti praktikoje. Tai yra skirtingai organizuojamos pratybos: pratimai, vaidmenų pasidalijimas… Pasinaudojant video priemonėmis besimokantieji gali įvertinti vienas kito darbą ir pasiekimus.

Lentelė nr. 3 : Mokymo metodai

Bloom taksonomija ir tikslų formulavimas informatikos srities studentams

Kompetencijos ir indikatoriai yra formuluojami vadovaujantis pasaulyje gerai žinoma ir plačiai paplitusia B.Bloom mokymosi tikslų taksonomija (Pav.2.) (Petty, 2006). Būtent todėl pateiktos kompetencijos ir jų indikatoriai yra ne vienodo lygmens. Tą turėtų suprasti dirbantys studentai ir juos konsultuojantys dėstytojai.



Pav.2. B.Bloom mokymosi tikslų taksonomija

Informatikos mokslo kompetencijos – tai kompetencijos iš daugelio mokslo ir praktikos sričių: informatikos, matematikos, fizikos ir kt. Todėl vertindami studentus, negalime ir neturime tikėtis, kad jie taps visų šių sričių ekspertais.

Kaip matome iš 5 paveikslėlio, yra skiriami 6 mokymosi rezultatų lygmenys: žinojimas, supratimas, pritaikymas, analizė, sintezė ir vertinimas. Pradedant žinojimu ir baigiant vertinimu iš bestudijuojančio yra reikalaujama vis daugiau sąmoningumo, tai yra gebėjimo gaunamas žinias pertvarkyti, sieti jas su jau turimomis ir kurti naujas žinias tam tikrose srityse.

Taigi dabar aptarsime kiekvieną iš mokymosi tikslų taksonomijos lygmenų.

- *Žinojimas* – tai gebėjimas ką nors prisiminti ir atkartoti.
- *Supratimo* lygmuo, reiškia, kad žinojimas yra suprantamas. Praktiškai tai pastebima, kai studentai gali paaiškinti tai, ką žino remdamiesi jau turimu mokėjimu ir patirtimi. Todėl supratimo, negaili būti be elementaraus žinojimo.
- Kitas aukštesnis lygmuo, *pritaikymas*, reiškia gebėjimą padaryti ar atlikti tai, kas yra žinoma, suprasta ar buvo parodyta
- *Analizės* lygmuo atspindi gebėjimus skaidyti visumą dalimis.
- *Sintezės* lygmens vertinimo užduočių atlikimas atspindi studentų gebėjimus generuoti sprendimus sprendžiant ne kasdieniškas problemas ar esant aplinkybėms, kur iš karto veikia keletas svarbių veiksnių.

Lygmuo	Vertinimo užduotys turėtų atspindėti studento gebėjimus:
Žinojimo	- atpažinti, surasti, išvardinti, įvardinti, teigti, apibrėžti, pacituoti, suskaičiuoti, papasakoti, rasti, apibūdinti.
Supratimo	- paaiškinti, perfrazuoti, susieti, suklasifikuoti, aptarti, apskaičiuoti, išplėtoti, apžvelgti.
Pritaikymo	- panaudoti, pademonstruoti, išbandyti, pakeisti, įgyvendinti, pabaigti, surasti, spręsti.
Analizės:	- suskaidyti informaciją į dalis, suprasti ryšius tarp turimos informacijos, palyginti, atskirti, sugrupuoti, padaryti išvadą, susieti, surikiuoti, išranguoti, nustatyti priežastis ir pasekmes.
Sintezės:	- suprojektuoti, sukurti, suformuluoti, integruoti, suplanuoti, pasiūlyti, pertvarkyti, organizuoti, parengti, nustatyti, pateikti naujų idėjų, iškelti hipotezę, sukritikuoti.

Lentelė Nr. 4: Vertinimo užduotys atspindinčios studentų gebėjimus įvairiuose lygmenyse

Žinojimo lygmens studijų rezultatams vertinti tinkamiausi yra testai. Jų taikymą ypač palengvina modernios informacinės technologijos. Tačiau reikia atkreipti dėmesį į tai, kad per nelyg didelis pasitikėjimas testais gali orientuoti studentus į mechaninį išmokimą nesiekiant aukštesnio lygmens mokymosi rezultatų. Gali būti, kad tarp studijuojančiųjų atsiras tokių, kurie surinkę maksimalų balų skaičių atsakinėdami į konsultanto pateiktus ar savitikros testus, per anksti save palaikys studijuojamo dalyko profesionalais. Tačiau tikroji žinojimo lygmens užduočių ir vertinimo žinojimo lygmenyje paskirtis – padėti studentams gilinti savo žinias ir eiti prie aukštesnių lygmenų mokymosi pasiekimų.

Tikslų formavimas informatikos srities studentams

Buvo atliktas tyrimas kurio metu surinkti ir klasifikuoti reikalavimai, keliami informatikos srities studentams, trejose šalyse (Amerikoje, Australijoje ir Taivane), atsižvelgiant į Bloom taksonomijos kategorijas. Labiausiai reikalinga kompetencija priklauso sintezės lygmeniui.

	Amerika (%)	Australija (%)	Taivanas (%)	Vidutinis (%)
Žinojimo	6.07	2.95	0	3.01
Supratimo	3.30	1.85	0.68	1.94
Pritaikymo	20.94	29.82	10.15	20.30
Analizės	8.81	7.37	8.12	8.10
Sintezės	40.27	41.19	53.39	45.00
Vertinimo	20.41	16.95	27.72	21.69

Lentelė Nr. 5. Reikalavimų pasiskirstymas pagal šalis

Kaip pavyzdį pateikiame kokie reikalavimai keliami Australijoje, norint pasiekti tikslus:

	Kompetencijos
Žinojimo	Duoti, gauti, sulyginti, patikrinti
Supratimo	Jausti, perteikti, sąveikauti
Pritaikymo	Diegti, valdyti, užbaigti, asistuoti, prieiti, priskirti, tinkinti, rasti, dalyvauti, įrašyti, siųsti
Analizės	Užtikrinti, identifikuoti, optimizuoti, migruoti, spręsti problemas
Sintezės	Sukurti, tobulinti, paruošti, planuoti, rašyti, integruoti, komunikuoti, surinkti, padaryti geriau
Vertinimo	Nustatyti, stebėti,naudoti, vertinti, patvirtinti, išspręsti, išrinkti, patvirtinti, palaikyti, versti, dirbti

Lentelė Nr.6. Kompetencijų pasiskirstymas pagal šalis

Computing Curricula 2001

Informatikos mokymo programoje 2001 aprašomi kursų lygiai, jų struktūros ir ryšiai. Yra skiriami trys pagrindiniai kursų lygiai. Tai pirminis arba įžanginis, skirtas pirmo – antro kurso studentams. Pagrinde yra supažįstama su medžiaga. Vidutinio lygio kursai dažniausiai skirti trečio – ketvirto kurso studentams ir kuria pagrindą tolimesnėms studijoms. Pažengusio lygio kursai yra dėstomi paskutiniaisiais metais ir yra orientuoti į konkrečius modulius, į kuriuos reikia laibiau įsigilinti. Siūloma griežtai nesilaikyti šių priklausomybių ir įdėti įdomesnę ar sunkesnę temą į mažesnio lygio kursus.

Informatikos mokymo programoje yra pateikiamas HCI modulio aprašymas.

CS250W. Human-Computer Interaction

Išsamiai aprašomos žmogaus – kompiuterio sąveikos principai ir technikos.

Būtina sąlyga: įvadas į kompiuterių mokslą.

Programa:

- Žmogaus-kompiuterio sąveikos atradimai : motyvacija, HCI kontekstai, į žmogų nukreiptas vystymas ir evoliucija, gero dizaino ir gerų dizainerių principai, inžineriniai kompromisai, supažindinimas su naudingumo testavimu.
- Į žmogų orientuotos programinės įrangos įvertinimas: tikslų įvertinimo nustatymas, įvertinimas be vartotojų, įvertinimas su vartotojais.
- Į žmogų orientuotos programinės įrangos tobulinimas: pasiekimai, charakteristika, proceso apžvalga, funkcionalumas ir naudingumas, interaktyvumo ir pristatomumo nustatymas, technikų ir įrankių prototipų kūrimas.

- Grafinės vartotojo sąsajos kūrimas: Sąveikos stilių ir technikų parinkimas; HCI ekrano dizaino aspektai; žmogaus klaidų taisymas; Kas slypi už paprasto ekrano dizaino; 3D sąveika; virtuali realybė.
- Grafinės vartotojo sąsajos programavimas: Analizavimo lygiai ir nepriklausomybė nuo dialogų; klasės; įvykių valdymas ir vartotojo elgsena; geometrijos valdymas; GUI konstruktoriai ir UI programavimo terpės; dizainas skirtas daugiaplatformėms aplikacijoms.
- HCI multimedijos sistemų aspektai: Informacijos kategorizavimas ir architektūra; Informacijos gavimas ir žmogaus elgsena; kalbos atpažinimas; informacijos panaudojimas ir mobilus naudojimasis kompiuteriu.
- HCI komunikacijos aspektai: Grupinis darbas atlikti specialias užduotis; asinchroninis grupių komunikavimas, sinchroninis grupių bendravimas; online bendruomenės; programų charakteriai ir protingi agentai

HCI modulis, dėstomas, užsienio universitetuose

Žmogaus-kompiuterio sąveika (*angl. HCI - Human-computer interaction*) yra sąveikos tarp žmogaus (vartotojo) ir mašinos (kompiuterio) studijos. Tai yra tarpdisciplinės studijos, susijusios su atskiromis sritimis, siejantis kompiuterių mokslą su daugeliu kitų studijų sritimis, jų tyrinėjimas. Sąsaja tarp vartotojo ir kompiuterio atsiranda per vartotojo sąsają (angl. user interface), į kurią įeina techninė įranga (angl. hardware) bei programinę įrangą (angl. software), kartu sudarantys tam tikrą terpę.

Universitetuose, kuriuose yra dėstomas informatikos modulis yra įtraukta ir žmogaus-kompiuterio sąveikos temą. Beabejo, pagrindiniai teiginiai bei frazės yra panašios. Tačiau skirtingose aukštosiose mokyklose modulis dėstomas skirtingai.

Užsienio universitetų puslapiuose yra viešai skelbiama modulio sudėtis, kokiu būdu studentai turės atsiskaitinėti, darbo grafikas, taip pat rekomenduojama papildoma literatūra.

Pagrindinės atsiskaitymo sudedamosios dalys yra teorinė ir praktinė. Tačiau priklausomai nuo universiteto, yra papildomai pridedami projektai, savarankiški darbai, kaupiamieji balai.

Pateiksime dviejų aukštųjų mokyklų modulio aprašymus :

Universitetas	
London Metropolitian University [1]	**University of Southampton [2]**
Modulio tikslas	
Suteikti studentams galimybe kurti bei gaminti prototipines aplikacijas, kartu dalyvaujant profesionalams	Padėti suprasti studentui kaip HCI įtakoja interaktyvių sistemų kūrimą, kaip pasikeičia programinė bei aparatinė įranga.
Dėstomų temų sąrašas	

HCI principai Vartotojiškai orientuotas projektavimas Naudingumo nustatymo metodai Prototipai Vartotojo samprata Pažintiniai modeliai Reikalavimų įsisavinimas Patogumo euristika	Vartotojo psichologija Įėjimo/išėjimo įrenginiai Modeliai ir metaforos Sąsajų stiliai, grafinė sąsaja Metodologijos kūrimas Užduočių analizė Papildomi interesai
Mokymas	
Paskaitų metu studentai įsisavins teorines žinias. Sužinos pagrindus. Galės atlikti praktiką, mokydamiesi iš profesionalų. Bus įtraukti į visas sąsajos atsiradimo stadijas.	Paskaitos, savarankiški užsiėmimai, savarankiškos praktinės užduotys
Atsiskaitymas	
Trumpas testas (10 %) 3 kursiniai darbai. Kiekvienas po (30 %)	Testai klasėje (20 %) Individualios užduotys (80 %)

Lentelė Nr. 7 : Aukštųjų mokyklų modulių aprašymai

Šie universitetai neskelbia paskatų tvarkaraščio. Toliau pateiksime aukštąsias mokyklas, kurios viešai skelbia paskaitų tvarkaraštį. Galima įvertinti paskaitų tankumą, ar skubiai ar palaipsniui dėstoma medžiaga. Studentai gali rinktis konkretų modulį ar ne.

Merkurio universitetas [3]

Modulio sandara:

Paskaitos, darbas mažose grupėse, nepriklausomos, savarankiškos studijos, projektai

Mokymo sesijos

Savaitė	Paskaita	Praktinis darbas
1	Supažindinimas su moduliu	HCI realiame pasaulyje Mobilių telefonų sąsajos. VCR dizainas. Mokymasis iš blogo dizaino Pavyzdžiai http://www.baddesigns.com Kaip išvengti blogo dizaino.
2	Vartotojo atminties pažinimas	Interneto naršymas Paprastos interneto svetainės dizainas (popieriuje)
3	Sąsajos projektavimas	Įvertinimo planavimas Idėjos ir dizainas Interaktyvaus tinklalapio planavimas
4	Sąsajos projektavimas Spalvos, stilius, išsidėstymas	Finalinės diskusijos Paprastos sąsajos kūrimas DVD ir mobiliam telefonui

5	Navigacija ir svetainės struktūra	Sąsajos projektavimas Projekto atnaujinimas, panaudojant spalvas ir kitus naujus elementus
6	Į vartotoją orientuotas prieiga	Iš naujo perdaryti DVD ir mobilaus telefono sąsajas, bei sulyginti su originaliomis
7	Greitas prototipavimas	Greiti plėtojimo pratimai Išplėtoti kelis trumpus prototipus.
8	Vartotojo galimybių suvokimas	Likusio laiko įvertinimas
9	Evoliucija	Išplėtoti paprastą planą paprastai statinei internetinei svetainei.
10	Praktiškumas	Paprastų praktiškumo testų siūlymas Pratimai, praktiškumui testuoti Praktiškumo klaidų pastebėjimas
11	Atsakomosios reakcijos įvertinimas	Produktų testavimas ir įvertinimas DVD dizaino įvertinimas
12	Praktiškumo įvertinimas	Įvertinti likusį laiką
13	Vartotojų testavimas Klausimynai Testavimas pagal užduotis	Kruopštesnis vartotojo testavimas Vartotojo užduočių plėtojimas Klausimyno plėtojimas Kaip sukurti gerą apžvalgą
14	Modulio santrauka	Išmoktų dalykų įvertinimas

Lentelė Nr. 8: Modulio teminis planas

De Paulo universitetas [4]

Tikslas

Pagrinde tikslas yra išmokyti studentus pritaikyti naujas įdėjas žmogaus – kompiuterio bendravimo sistemoje. Tai galima padaryti studijuojant:

- Domėtis naujais HCI metodais, bei bandant vystyti juos;
- Išmokti kaip pritaikyti naujus HCI metodus;
- Eksperimentiškai atrasti naujus metodus;
- Diskutuoti ir pristatyti atrastus rezultatus.

Tematinis tvarkaraštis

Savaitė	**Tema**	**Projektas**
1	Kurso apžvalga.	Projekto tikslų aptarimas
2	Žmogiškasis faktorius	Galimų projektų aptarimas
3	Vartotojo poreikių analizė	Projekto pasiūlymas
4	Informacijos architektūra	Projekto pradžia

5	Sąsajos projektavimas	Projektas tęsiamas
6	Prototipo sukūrimas ir įgyvendinimas	
7	Vystymas	Neoficiali progreso ataskaita
8	Sąnaudų pateisinimas	
9	HCI praktikoje	Projektų pristatymas
10		Projektų pristatymas
11		Baigiamasis projektas

Lentelė Nr. 9: Kurso teminis planas

Apibendrinimas

Pagrindiniai dėstomo modulio principai yra panašūs. Visuose universitetuose mokoma kurti, ieškoti, pritaikyti, analizuoti ir kitais būdais apdoroti gautą informaciją. Priklausomai nuo universiteto, bei studijų programos, valandų skaičius, skiriamas HCI moduliui yra skirtingas. Kuo daugiau valandų – tuo plačiau galima analizuoti, įsigilinti. Nuo to priklauso paskaitų tematika, praktinės užduotys ir jų kiekis. Nepriklausomai nuo skiriamų valandų kiekio, pateikiama papildoma literatūra, kurią studentai gali studijuoti savarankiškai, jei jiems trūksta žinių, kurias gauna universitete.

III. Nuotolinio mokymo kurso reinžinerijos teorinis modelis

NMK struktūra

Nuotolinių studijų studentai – tai visų studijų pakopų bei formų Šiaulių universiteto studentai, kurie nuotoliniu būdu studijuoja nuosekliųjų studijų programą ar atskirą jos kursą. (ŠU studijų nuostatai)

Visų studijų formų kursai ir programos gali būti teikiami nuotoliniu būdu. Nuotolinės studijos – tai studijų būdas, grindžiamas e. mokymo(si) technologijomis, realizuojant modulius ir studijų programas virtualioje mokymo(si) aplinkoje. Savarankiškam darbui skirtą laiką studentas pats planuoja taip, kad atsiskaitytų studijų programoje nurodytu laiku. Bendravimas su dėstytoju ir atsiskaitymai vyksta pagal sudarytą tvarkaraštį.

Nuotolinės studijos vyksta pagal kursus, modulius, kurie, prieš registruojant juos studijų programų komitete, turi gauti Nuotolinių studijų centro žymą, kad yra nuotoliniai.

Nuotoliniu būdu dėstomame modulyje turi būti tarpinių užduočių ir / arba egzaminas, kuriame studento žinios būtų vertinamos tiesiogiai jam dalyvaujant.

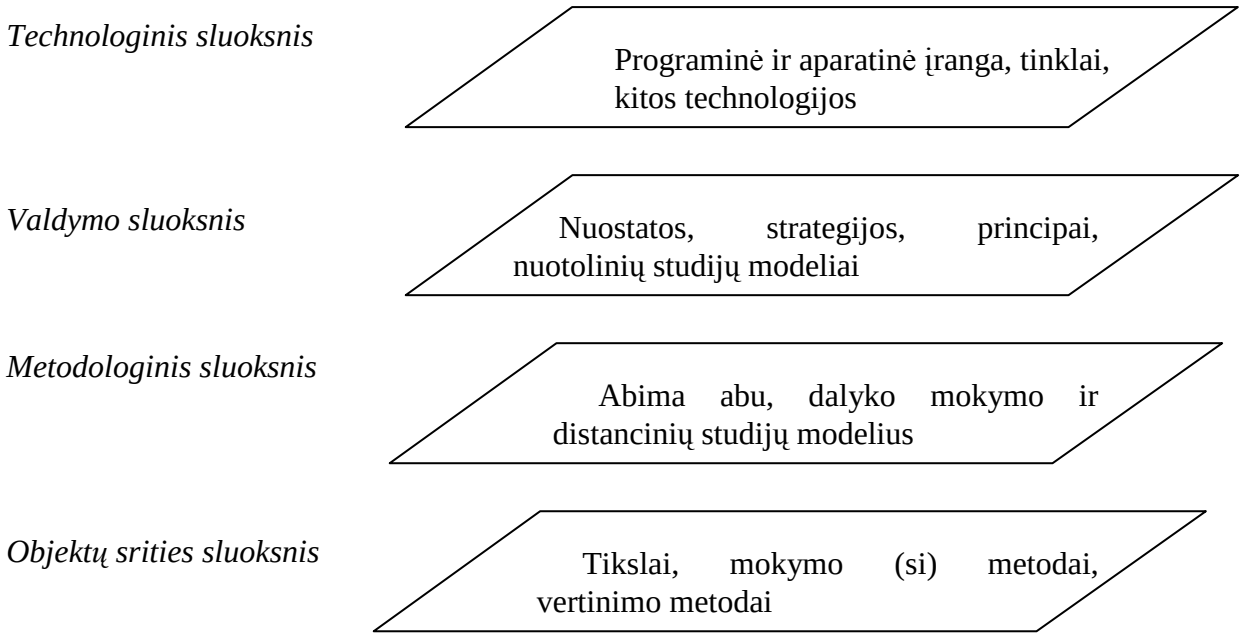
NMK reinžinerijos proceso eiga

Reinžinerijos konceptas programinės įrangos kūrimo ir valdymo srityse

Reinžinerijos koncepto įvairios interpretacijos yra naudojamos programinės įrangos kūrime ir valdymo srityse. Programinės įrangos reinžinerija yra paremta pakartotiniu sistemos diegimu norint pasiekti patogesnio valdymo [Sommerville, 2000]. Programinės įrangos reinžinerijos procesas susideda iš : a) šaltinio failo išvertimo; b) atvirkščio kūrimo; c) programos struktūros gerinimo; d) programos moduliarizacijos; e) duomenų pertvarkymo. Ne visi etapai yra reikalingi. Tai priklauso kokį lygį mes norime pertvarkyti.

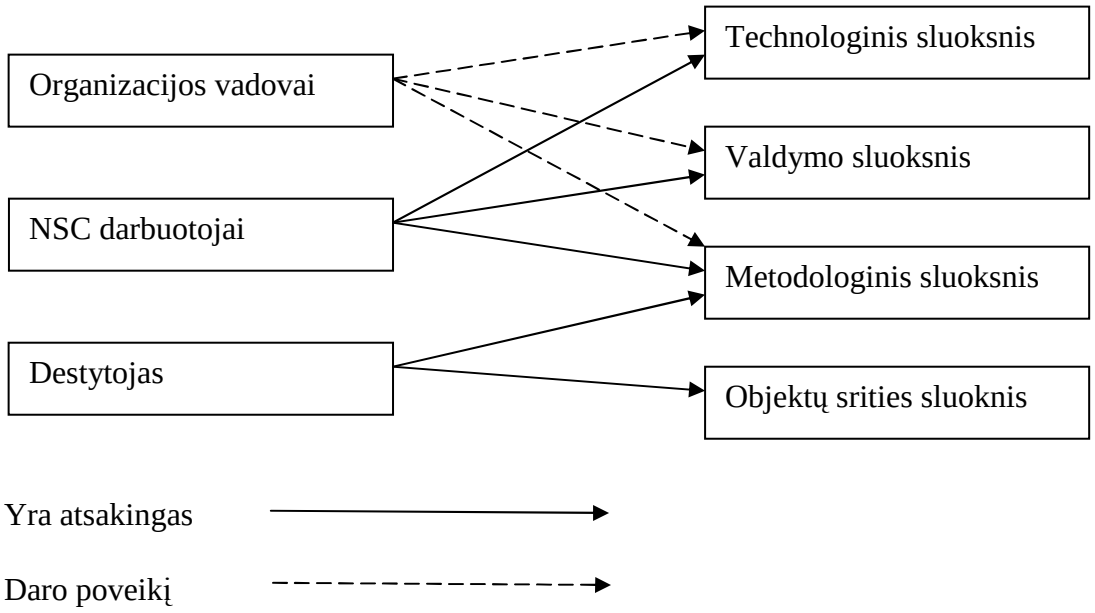
Macro ir Mikro lygiai NMK sistemos reinžinerijoje

Reinžinerija gali būti pritaikyta visai sistemai arba jos komponentams, tamtikruose abstrakcijos lygiuose. Pateikiami NMK sistemos lygiai. Tipiniai NMK sistemos lygiai iš dėstytojo akių yra parodyti paveikslėlyje 3.



Paveikslėlis 3. NMK sistemos sluoksniai iš dėstytojo perspektyvos (pagal Gruslyte, 2007])

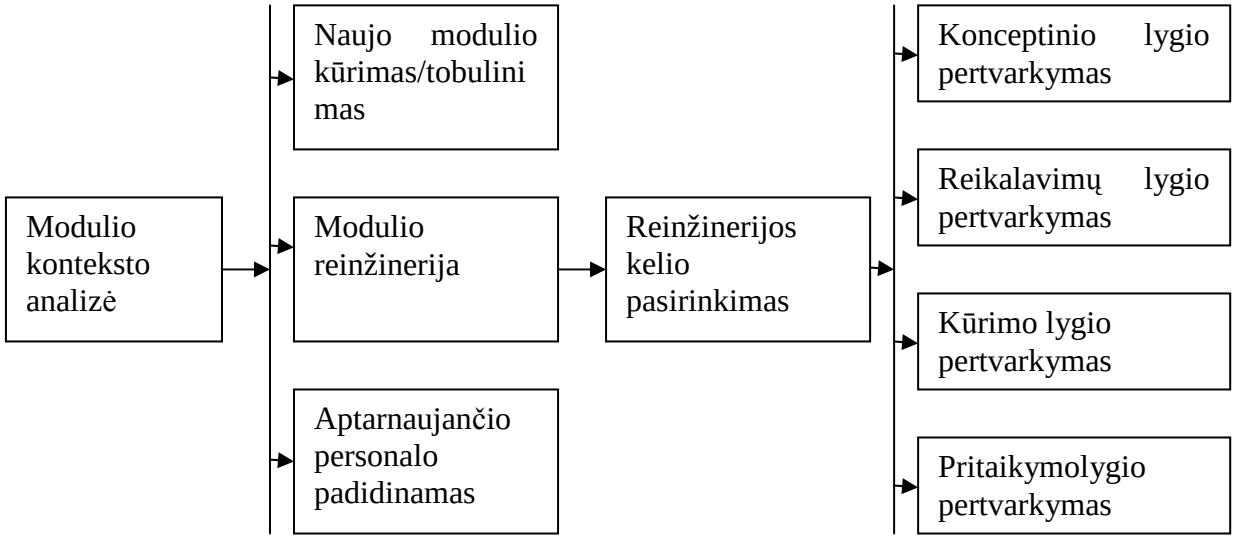
Reinžinerijos procesai šiuose sluoksniuose gali būti vykdomi kelių žmonių organizacijoje. Situacija Šiaulių Universitete (tipinė aukštoji mokykla) atvaizduota paveikslėlyje. 4



Paveikslėlis 4. Darbuotojų atsakomybės Pertvarkant NMK

Mokymo dalių reinžinerijos struktūra

Mūsų siūloma struktūra parodyta Pav.: 5. Ji paremta pagrindiniu programinės įrangos reinžinerijos modeliu, pasiūlytu [Bryne, 1992].



Paveikslėlis 5 : Mokymosi grupės reinžinerijos sistema

Pirmajame žingsnyje yra priimami sprendimai ar reikia atlikti reinžinerija. Mes turime adaptuoti nuotolini mokymosi kursą įvairiuose kontekstuose: 1) Skirtingoms studijų formoms: dieninėms, vakarinėms, neakivaizdinėms; 2) Skirtingoms studijų programoms, kur tas pats modulis turi apimti skirtingą kiekį studento darbo valandų 3) Skirtingoms studijų programoms, kur skirtingi to pačio dalyko aspektai turi būti pabrėžti. Dažniausiai yra daroma nuotolinio mokymo kurso kopija ir modifikuojamos atskiros dalys. Bet kokiu atveju tai užima daug laiko. Yra dar dvi alternatyvos nesiimant skyriaus reinžinerijos: 1) Naujo skyriaus sukūrimas. Šis kelias gali būti pasirenkamas kai naujasis mokymosi skyrius labai nutolsta nuo esamo. 2) padidinant personalo pagalbą. Nuotolinės studijos yra orientuotos studijuojantįjį ir paremtos aktyviu jo dalyvavimu. Beabejo, papildomas akademinio personalo įsikišimas į mokymo (si) procesą gali sumažinti grynos mokymosi medžiagos negatyvią įtaką. Sprendimas ar pradėti reinžinerija yra labai svarbus, nes tai įtakoja tolimesnes išlaidas. Taip pat tai smarkiai įtakoja modulio panaudojimą vėliau.

Antras svarbus žingsnis yra pasirinkti reinžinerijos lygį. Mes naudojame klasifikaciją, pasiūlytą [Bryne, 1992]. Lygiai, susiję pakeitimų tipai ir adaptacijos pateiktos lentelėje.

Reinžinerijos lygiai	**Pakeitimo tipas**	**Susyjė su:**
Koncepcinis	Galvojimo iš naujo	Pagrindas reinžinerijai. Pvz.: priimta nauja pedagoginė strategija
Reikalavimų	Reikalavimų atnaujinimas	- modulio reikalavimai, tikslai ir esamos žinios
Kūrimo	Kūrimas iš naujo	- informatyvi, navigacinė, prisistatomoji struktūra
Realizavimo	Perprogramavimas	Resursai, veikla, scenarijai

Lentelė Nr. 10 : Reinžinerijos lygių sąsajos

Aukštesnieji lygiai apima mažesnius. Pvz.: priimti nauji reikalavimai, mums reikia pertvarkyti Kūrimo ir Realizavimo skyrius.

IV. Atvejo analizė: dviejų NSK *Programavimas grafinėje terpėje* temų reinžinerija

Konteksto analizė

Nagrinėjamo modulio „Programavimas grafinėje terpėje“ tikslas yra pristatyti pagrindinius įrankius ir metodus naudojamus programų kūrimui Windows aplinkoje. Taip pat studentai įgyja žinių atlikdami praktinius darbus. Numatomi sekantys mokymo metodai: paaiškinimas, konsultavimas, diskusijos, pavyzdžių analizė, laboratoriniai darbai ir projektai.

Tema „Programų vartotojo sąsajos“

Temos tikslas: Susipažinti su žmogaus kompiuterio sąveikos teorija, apžvelgti grafinės vartotojo sąsajos vystymąsi ir dabartines pritaikymo galimybes. Tema „Programų vartotojo sąsajos“ gali būti priskiriama įvadiniams kursams (angl. introductory courses) (remiantis 3 tipais pagal CC2001).

Studentas, susipažinęs su šiuo skyriu, turi sugebėti paaiškinti gautą informaciją, aptarti atskiras jos dalis ir išplėtoti esamą informaciją. Tokiu būdu yra pasiekiamas antrasis Bloom mokymosi taksonomijos lygis – supratimas.

Tema yra modifikuota. Esama medžiaga papildoma naujais skyriais.

Skyrius	**Tematika**
Apibrėžimas	Grafinės vartotojo sąsajos apibrėžimas
Vartotojo sąsajos tipai	Įvardijami trys vartotojo sąsajos tipai
Priešistorė	Grafinės vartotojo sąsajos atsiradimo istorija
Grafinės vartotojo sąsajos „tėvas“	Žmogus, sugalvojęs ir daug prisidėjęs prie GVS atsiradimo
Taigi kas toji „Grafinė vartotojo sąsaja“	Kaip dabar suprantama sąsaja
Grafinės vartotojo sąsajos vystymosi eiga	Aptariama kaip tobulėjo ir žingsnis po žingsnio nuo komandinės eilutės iki dabartinių grafinių sąsajų
Grafinių vartotojų sąsajų pavyzdžiai	Pateikiami vizualūs pavyzdžiai
Grafinės vartotojo sąsajos tipai	Išskiriamos specifinės, naudojamos tik tam tikrose srityse sąsajos
„Žmogaus-kompiuterio sąveika“ tyrinėjimų sritis	Tiriamas HCI tinkamumas, naudingumas, intuityvumas, žmogaus ir kompiuterio komunikavimo gerinimas.
Programų vartotojo sąsajos pritaikymas neįgaliesiems	Aptariami atvejai ir galimybės kaip galima pritaikyti vartotojo sąsają neįgaliems vartotojams.
Tipinės vartotojo sąsajos projektavimo klaidos	Aptariamos pagrindinės klaidos projektuojant: informacijos kiekis, dialogai, spalvos, taisyklių nesilaikymas

Lentelė nr. 11: Temos „Programų vartotojo sąsajos“ skyrių apžvalga

Vertinimo metodai:

1. referatas -pasiekiamas 4 Bloom taksonomijos lygis. Studentas analizuoja turimą medžiagą ir žinias;

2. testai - apima 1-2 Bloom taksonomijos lygį. Studentai žino ir supranta medžiagą bei geba atsakyti i pateiktus klausimus. Testai sudaromi iš pateiktos medžiagos.

Tema yra patalpinta http://alfa.distance.su.lt

Kursas: E. mokymosi projektavimas

Paveikslėlis 6: tema „Programų sistemų projektavimo“ Moodle mokymo(si) sistemoje

Tema „Qt biblioteka“

Temos tikslas yra supažindinti su Qt bibliotekos atsiradimu, vystymusi, jos atskiromis dalimis. Pateikiami praktiniai pavyzdžiai. Tema yra sukurta visiškai nauja,

Tema „Qt biblioteka“ kaip ir prieš tai nagrinėta tema, gali būti priskiriama įvadiniams kursams (angl. introductory courses) (remiantis 3 tipais pagal CC2001).

Tema yra suskaidyta į dvi dalis. Pirma dalis – teorija. Antra dalis – praktika. Teorinė dalis gali būti priskiriama antrajam Bloom mokymosi taksonomijos lygiui (supratimas)

Studentas, susipažinęs su Qt bibliotekos praktine dalimi, turi sugebėti panaudoti gautą informaciją, išbandyti pateiktus pavyzdžius, pakeisti programinį kodą, siekiant pasiekti norimą rezultatą. Tokiu būdu yra pasiekiamas trečiasis Bloom mokymosi taksonomijos lygis – pritaikymas.

Skyrius	**Tematika**
Qt istorija	Apžvelgiama Qt istorija
Qt yra daugiaplatformė	Apžvelgiamas Qt daugiaplatformiškumo galimybės
Qt dizaineris	Qt dalis – dizaineris, jo galimybės ir nauda
Qt Kalbininkas	Qt dalis – Kalbininkas, jo galimybės ir nauda
QT Asistentas	Qt dalis – Asistentas, jo galimybės ir nauda
Qtopia	Platformos, veikiančios Qt pagrindu telefonuose ir PDA apžvalga ir ypatybės
Reikalavimai	Qt veikimo užtikrinimui keliami reikalavimai mašinoms.
Praktinė dalis	
Instaliavimas Windows aplinkoje	Aprašomas ir iliustruojamas Qt diegimas Windows aplinkoje
Qt diegimas įvairiose aplinkose	Qt diegimas Windows, Mac OS ir X11 aplinkose
C++ ir Qt sąveika	Aprašomi dviejų programų sąveikos būdai
3 programų pavyzdžiai	Pateikiami ir paaiškinami trys pavyzdžiai, kuriuos studijuojantysis gali ir turi išbandyti savarankiškai.
Objektų stiliai	Pateikiami įvairių objektų vizualiniai stiliai, priklausant nuo aplinkos.

Lentelė nr. 12 : Temos „Qt biblioteka“ skyrių apžvalga

Vertinimo metodai:

1. referatas -pasiekiamas 4 Bloom taksonomijos lygis. Studentas analizuoja turimą medžiagą ir žinias;

2. praktinės užduotys - apima 2-3 Bloom taksonomijos lygį. Studentai supranta ir gali pritaikyti praktiškai turimas žinias.

Praktinės užduoties pvz.: modifikuoti pateiktą programą taip kad pakistu objektų, esančių aplikacijoje spalva/kiekis/veikimo būdas.

Tema yra patalpinta http://alfa.distance.su.lt

Kursas: E. mokymosi projektavimas

Paveikslėlis 7: tema „Qt Bibliotekos“ Moodle mokymo(si) sistemoje

Išvados

Reinžinerijos sąvokos sampratos analizė parodė, kad priklausomai nuo srities ir laikotarpio, reinžinerija yra apibrėžiama kitokia terminologija. Tačiau pagrindinė mintis išlieka tokia pati – pertvarkyti, atlikti darbus kitaip, kurti permainas, integruoti naujoves.

Užsienio universitetų analogiškų kursų analizė parodė, kad visi universitetai veikia skirtingai, tačiau turi bendrų temų savo kursuose. Taip pat pateikiami papildomi literatūros šaltiniai, norintiems įsigilinti. Kiekvienas universitetas remiasi tarptautine informatikos mokymo programa (CC 2001) tik iš dalies. Pateikta programa yra bendra, rekomendacinio pobūdžio. Kiekviena mokymo įstaiga atlieka savo interpretacijas ir laisvai modifikuoja modulį bei pritaiko jį savo universitetuose.

Sukurtas teorinis NSK reinžinerijos modelis yra tinkamas nuotolinių kursų, nebūtinai informatikos studijų krypties, reinžinerijai. Jis yra universalus, ir gali būti pritaikomas plačiai. Yra galimybė pasirinkti nuo kurio lygio pradėti pertvarkymą: koncepcinio, reikalavimų, dizaino, diegimo lygio. Laikantis nuoseklumo: atlikti analizę ir parengiamuosius darbus, nuspręsti kurio lygio reinžinerija bus atliekama - nesunku įgyvendinti norimus pakeitimus.

Atlikus praktinę temų reinžineriją buvo sėkmingai atnaujinta mokymo medžiaga bei sukurti ir paaiškinti pavyzdžiai, kurie padeda lengviau įsisavinti medžiagą.

Nuotolinių studijų kurso *Programavimas grafinėje terpėje* reinžinerija

Anotacija (santrauka)

Šiame magistro darbe mes analizuojame nuotolinių studijų kurso bendrai ir mokymo medžiagos atskiromis kurso dalimis pertvarkymą. Reinžinerijos koncepcinis modelis yra įvairiai interpretuojamas ir yra taikomas programinės įrangos rengimo, ar vadybos moksluose. Tai yra daroma, kad sistemas būtų galima geriau panaudoti. Jos analizuojamos ir pertvarkomos tolimesniam naudojimui. Mes sujungiame minėtų sričių reižinerijos koncepto reikšmes ir naudojame tai kaip metodinį pagrindą nuotolinių studijų srityje.

Mes analizuojame nuotolinių studijų kurso struktūrą. Tada pristatome kursus paruoštus ir naudojamus užsienio universitetuose, susijusius su ‘Programavimu grafinėje terpėje‘, trigubo pastovumo principu, Bloom taksonomija ir jos pritaikymu kompiuterijos mokslų studijavimui, informatikos mokymo programa 2001 (angl. computing curriculla 2001). Toliau, mes siūlome konceptualią nuotolinių studijų pertvarkos struktūrą iš dėstytojo perspektyvų ir pristatome atvejų analizę, kurioje pora temų yra pertvarkytos, atsižvelgiant į trigubo pastovumo principą ir reikalavimus kompiuterių mokslų studentams.

Reengineering the Distance Study Course ‚Programming in GUI‘

Summary

In this master thesis we analyse a problem of reengineering of a distance study system, in general, and the learning material of a separate course, in particular. Reengineering concept with its different interpretations is used in software engineering and management sciences. It deals with making systems better maintainable, examination and reconstitution of the system for further reimplementation. We combine the meaning of reengineering concept in both mentioned areas and employ it as methodological background in distance study area.

We analyse the structure of the distance study course. Then we introduce the courses, prepared and delivered in foreign universities and related to ‘Programming in GUI’ course, triple consistency principle, Bloom taxonomy and its applicability to computer science studies, Computing Curricula 2001. Further, we propose a conceptual distance study course reengineering framework from the lecturer’s perspective and present a case study, in which two topics were reengineered, considering triple consistency principle and requirements for computer science students.

Literatūra:

1. Žilvinas Ledas (ledas.dtiltas.lt/MIF/3_sem_egz/egz_PSI/PSI_ats_mano_norm.doc)
2. Alan S. Michaels (www.e-competitors.com/Glossary/Terms_R.htm)
3. Michael R. Olsem (www.stsc.hill.af.mil/resources/tech_docs/RENGVOL1.DOC Psl.9)
4. University of Stuttgart, Germany (www.iste.uni-stuttgart.de/ps/reengineering/terminology.html)
5. Federation of American Scientists (www.fas.org/spp/military/docops/usaf/2020/app-v.htm)
6. http://www.milijardierius.lt/28ProcesuReorganizavimas/28ProcesuReorganizavimas.htm
7. Ian Sommerville (http://kopustas.elen.ktu.lt/~rsei/SE4/20.PI.keitimas.ppt)
8. VDU informatikos katedra (www3.vdu.lt/Inf-sist/siu-metu/pask2.doc)
9. http://www.londonmet.ac.uk/module-catalogue/2/bs/bs2051.cfm
10. http://www.ecs.soton.ac.uk/admissions/ug/syllabus.php?unit=COMP2006.
11. http://mercury.tvu.ac.uk/hci/teaching.html
12. http://facweb.cs.depaul.edu/cmiller/capstone/syllabus.html
13. http://www.ku.lt/centrai/krc/tutor/#Bendriej1
14. http://www.vgtu.lt/studijos/?id=3.701
15. http://nsc.vtu.lt/main.php?id=12
16. ŠU studijų nuostatos (www.su.lt/filemanager/download/1994/studiju_nuostatai0411.pdf)
17. Petty G. Šiuolaikinis mokymas. Vilnius: Tyto alba, 2006.
18. http://en.wikipedia.org/wiki/Computer_accessibility
19. http://infas.lzua.lt/algikurl/Idalis.htm#_Toc40881778
20. http://www.mif.vu.lt/~moroz/priemone/hci.pdf
21. http://lt.wikipedia.org/wiki/Interfeisas

Priedų sąrašas

1. Programų vartotojo sąsajos – studijų medžiaga.
2. Qt bibliotekos apžvalga - studijų medžiaga.
3. Qt praktinė užduotis – studijų medžiaga
4. Tezės

Priedas Nr. 1. Programų vartotojo sąsajos – studijų medžiaga.

Įžanga

Šiandien beveik kiekvienas išsivysčiusiame pasaulyje su asmeniniais kompiuteriais bendrauja vienokia ar kitokia forma. Mes juos naudojame darbe, namuose, mokykloje. Kompiuteri mes naudojame pramogaudami, dirbdami, ieškodami informacijos, kaip įrankį mūsų žinių ir išsilavinimo praplėtimui. Mums jau tapo įprasta, kad kai atsisėdame prie kompiuterio, jis su mumis bendrauja grafinės vartotojo sąsajos pagalba. Mes tikimės bendrauti su juo visų pirma naudodami pelytę, kuria vykdome programas, spustelėdami piktogramas, taip pat mes galime atlikti įvairius veiksmus grafinėje aplinkoje. Tačiau taip parasta buvo nevisada.

Apibrėžimas

Grafinė vartotojo sąsaja (angl. GUI),- tai kompiuterio sąsaja, kuri naudoja grafines piktogramas (ikonas) su papildomais aprašais. Kompiuterio vartotojas naudodamas įvesties įrenginį, tokį kaip pelę, manipuliuoja šiomis piktogramomis bei langais. Tai visiškai priešingas kompiuterio valdymas, nei naudojant komandinės eilutės vartotojo sąsają, kurioje reikia rašyti tekstines komandas, norint įvykdyti kompiuterio programą.

Taip pat vartojo sąsaja galima apibrėžti kaip reikšmių kuriomis vartotojai bendrauja su skirtingais įrenginiais ir kompiuterinėmis programomis visuma.

Grafinė vartotojo sąsaja turi 2 pagrindines funkcijas:

Įvedimas (angl. input), galimybė vartotojams kontroliuoti sistemą.

Išvedimas (angl. output), galimybė sistemai informuoti vartotojus, dar vadinamas atgaliniu ryšiu (angliškai – feedback)

Sąsaja (angl. interface) - tai tam tikras komunikacijos kanalas, per kurį užtikrinamas bendravimas tarp kompiuterio ir žmogaus.

Šis kanalas turi dvejopas priemones:

Fizinės priemonės (klaviatūra, kompiuterio ekranas, pelė);

Vizualinės išraiškos priemonės (įvairiausios informacijos pateikimo metaforos, tekstai, iliustracijos ir pan.).

Vartotojo sąsaja yra sudaryta iš tokių elementų:

Koncepcinis vartotojo modelis (angl. conceptual model) - kaip vartotojas supranta savo užduotį ir ko jis siekia. Čia yra užkoduotas žmogaus individualus kompiuterinės sistemos vertinimas - kaip pasinaudodamas šia sistema žmogus galės pasiekti savus tikslus. Šis koncepcinis modelis dar vadinamas žmogaus elgesio schema.

Informacijos pristatymas (arba ženklų kalba), kurį žmogus mato kompiuterio ekrane (angl. presentation language) - kaip žmogus bendrauja su informacija;

Žmogaus veiksmų kalba (angl. action language) - kokius veiksmus, norėdamas išspręsti užduotį, atlieka žmogus. Informacijos valdymo metodus programos kūrėjas jau yra numatęs iš anksto - tai yra interaktyvumą skatinančios priemonės. Ši kalba - tai bus žodžiai ar komandos, kurias vartotojas įves iš kompiuterio klaviatūros, rinksis iš meniu, ar pasirinks kokius nors objektus su pele.

Įdiegimo modelis (angl. implementation model). Šia dalimi domisi kompiuterių mokslo specialistai.

Vartotojo sąsajos tipai

Komandų eilutės : naudojama vis rečiau, pvz., operacinėje sistemoje MS-DOS komandų eilutėje pasirodęs kvietimas A:> arba C:\> “prašo” ar “kviečia” vartotoją įvesti komandą.

meniu tipo : naudojama daugumoje šiuolaikinių taikomųjų programų. Ji siūlo vartotojui panaudoti reikalingą komandą iš pateikto sąrašo (meniu) (paspaudus kai kuriuos klavišus arba pele)

grafinė sąsaja: būdinga vadinamųjų “langų” sistemų savybė. Šios sistemos vaizduoja programas, dokumentus ir kitus objektus paveikslėliais (piktogramomis) kompiuterio ekrane atidarytuose languose. Grafinėse sąsajose naudojamos tradicinės priemonės ir metaforos, pvz., darbo stalas ir įvairūs įrankiai.

Priešistorė

Kaip kiekviena besivystanti technologija, taip ir kompiuteriu raidos istorijoje galime įžvelgti, kad kai kurios GVS (grafinė vartotojo sąsaja) arba GUI (angl. graphical user interfase) vystymo idėjos buvo sumąstytos gerokai anksčiau nei atsirado galimybė sukurti kompiuterį galintį realizuoti šitą idėją. Vannevar Bush‘as buvo vienas pirmųjų žmonių išreiškęs tokias mintis. Ankstyvaisiais 1930-taisiais jis pirmasis aprašė įrenginį, kurį jis vadino „Memex“. V. Bush‘as įsivaizdavo jį atrodantį kaip stalas su dviem jautriais paspaudimui grafiniais vaizduokliais, klaviatūra ir skaitliu prijungtu prie jo. Tai leis vartotojui kreiptis į visą žmogaus pažinimą, naudojant jungtis labai panašiai veikiančias kaip hipernuorodos. Šiuo aspektu toks skaitmeniniu pagrindu veikiantis kompiuteris dar nebuvo išrastas, sukurtas, todėl nebuvo galimybės išbandyti šį įrenginį realiai veikiantį, todėl V. Bush‘o idėjos nebuvo plačiai skelbiamos ir diskutuojamos tuo metu.

Kaip bebūtų, pradedant kažkur apie 1937-uosius atsirado keleto žmonių grupelės visame pasaulyje panorusių kurti skaitmeninius kompiuterius. Antrojo Pasaulinio karo metu tokio tipo projektui buvo skirtas labai didelis finansavimas, nes išaugo labai didelis programuojamų skaičiavimo mašinų poreikis. Tokio tipo mašinų reikėjo siekiant atlikti rezultatyvius artilerinius skaičiavimus, ar net priešo slaptaraščiams iššifruoti. Kadangi trūko kvalifikuotų specialistų galinčiu programuoti tokius įrenginius, tai buvo manoma, kad greitesnis sistemos perpratimas bus pasitelkiant grafinę aplinką.

Grafinės vartotojo sąsajos „tėvas“

Douglas Englebart (žr. 1 pav.) baigė elektroninės mechanikos studijas. 1948 metais įsidarbino NACA Institute. Kaip šaltiniai teigia, vieną dieną važiuodamas į darbą jis visiškai netikėtai suvokė, kad jo tikrasis pašaukimas nėra darbas su mažais projektais, kurie gelbsti keliems žmonėms, o darbas kuris padėtų visai žmonijai. Jis prisiminė V. Bush‘o pamastymus ir pradėjo ieškoti būdų kaip juos būtų galima realizuoti, kad tai padėtų žmonėms siekiantiems įgyti didesnį intelektinį išsilavinimą.

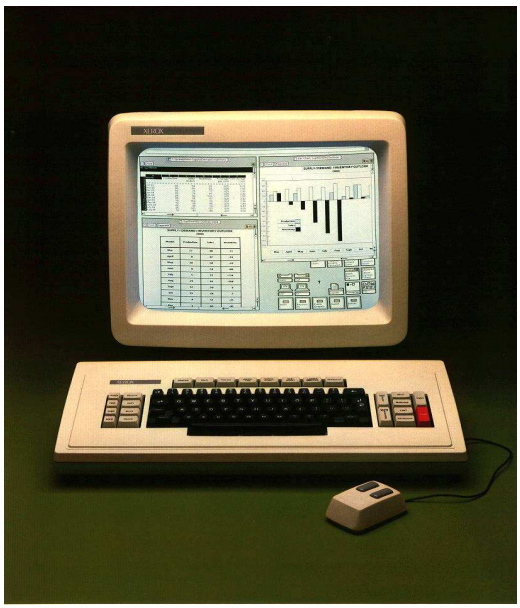
1 pav. Douglas Engelbart 1968m.

Karo metu D. Engelbart‘as dirbo radaro operatoriumi, taigi jis galėjo perprasti vaizduojamosios sistemos sukūrimą pasitelkiant katodo spinduliuojančius vamzdžius, kurių pagalba galima informaciją atvaizduoti kaip grafinį vaizdą. Taigi prasidėjo potencialių finansuotojų paieška kurie galėtų remti jo laukines idėjas, kurias įrodyti (realizuoti) buvo ilga ir sunki užduotis. 1955 metais D. Engelbart‘as įgijo mokslų daktaro laipsnį. Tais pačiais metais pradėjo dirbti Stenfordo Tyrimų Institute. Dirbdamas šiame institute jis patentavo begalę kompiuterinių komponentų. Maždaug apie 1959 metais jis pelnė pakankamą pripažinimą už atliktus mokslinius tyrimus. Tai buvo lemtingas faktorius, kurio dėka jo projektą pradėjo finansuoti Jungtinių valstijų Oro pajėgos. 1962 metais D. Engelbart‘as viešai paskelbė savo originalios idėjos užuomazgas straipsnyje: „Augmenting Human Intellect“. Pagrindinė straipsnio perteikta mintis buvo tai, kad nereikia siekti kompiuteriais atkartoti žmogaus intelektą, o kompiuterį naudoti kaip pagalbinį įrankį praplečiantį jį. Vienas iš hipotetinių pavyzdžių buvo: architektūrinio pastatų projektavimas naudojant kažką panašaus į dabar labai gerai žinoma automatinio projektavimo sistemą CAD. Tai buvo milžiniškas šuolis, įvykęs 1962 metais. Šia idėja galėjo realizuoti tik vieninteliai kompiuteriai, kurie tuo metu egzistavo kaip gigantiškos tarnybinės stotys, su kuriomis vartotojas galėjo bendrauti tik taip vadinamu "batch processing" būdu. Vartotojas įkraudavo programą pasinaudodamas galybę niukso kortų. Deja, dėl ribotų atminties resursų programa galėdavo veikti tik labai ribotą laiką, ko pasėkoje norint ką nors apdoroti prireikdavo keliolikos valandų ar netgi dienų. D. Engelbart‘as ir jo augantis personalas dirbo keletą metų plėtodami idėjas ir technologijas, kurios galiausiai baigėsi 1968 metais viešoje demonstracijoje prieš tūkstančius profesionalių kompiuterininkų.

Taigi kas toji „Grafinę vartotojo sąsaja

DOS, UNIX ir kitos komandinės eilutės operacinių sistemų ilgą laiką buvo kritikuojamos už sudėtingą operacinės sistemos vartotojo sąsają. Ši sąsaja vystėsi nuo šešiasdešimtųjų pabaigos – septyniasdešimtųjų pradžios kas karta priversdama tobulėti ir pačius kompiuterius. Naudojant terminalines sistemas dalijamų resursų mini kompiuteriuose, UNIX plėtojo programinę įranga ribotų resursų sistemose orientuotose į vartotojo sąsają. Ši sąsaja buvo revoliucija tuo metu dėl jos paprastumo ir pajėgumo.

Toliau sekė tyrimai Xerox PARC sistemoje, kurioje viskas vyko visiškai kitaip. Šioje sistemoje grafinėmis vartotojo sąsajomis kurios pakeitė neoninių lempų displėjus ir komandinę eilutę su dideliais grafiniais vaizduokliais, piktogramomis, multi langų ir įvesties įrenginiais (pelė, planšetai). Tyrimai parodė, kad žmonės žymiai greičiau gali išmokti naudotis taikomosiomis programomis, kai jos turi grafinę sąsaja nei, kad programomis valdomomis komandinėmis eilutėmis. Be to, grafinėje sąsajoje gerokai lengviau įsiminti charakteringas programos veikimo vietas ir dauguma funkcijų.



GUI aplinkoje įvesties nustatymai yra suprojektuoti daugeliui kompiuterio programų kaip piktogramų, ikonų masyvai , kurie yra grafiniai simboliai su vykdymų nuostatomis. Kitaip tariant jie reprezentuoja vykdomą programą. Tokiu būdu vartotojui maksimaliai supaprastinamas programos vykdymas, kuomet užtenka tik įvesties įrenginiu (pvz. pele, ar panašios paskirties kitas įrenginys) suaktyvinti piktogramą. GUI atsiradimo pradžioje atsirado tokiu vartotojų, kurie teigė, kad vykdyti komandinę eilutę yra žymiai greičiau, nei grafinėje vartotojo sąsajoje. Jie vartotojus vadindavo WIMPs‘ais - Window, Icons, Mice and Pointer users (Langų, ikonų, pelių ir žymeklių vartotojais.)

Grafinės vartotojo sąsajos vystymosi eiga

Vartotojo sąsajos istoriją galima suskistyti į tris etapus:

“Kepalo” vartotojo sąsaja (anliškai – Batch interface), 1945 – 1968 m., tai nesąveikaujanti vartotojo sąsaja, kur vartotojas iš anksto tiksliai nurodo visas vykdymo instrukcijas yra išvedimas atsakymas ir viskas. Kompiuteris nereikalauja jokių papildomų informacijos (instrukcijų) įvedimų.

Komandinės eilutės vartotojo sąsaja (angliškai – Command-line user interface), 1969-1983 m., tai vartotojo sąsaja, kur vartotojas numato įvedimą renkant komandines eilutes su kompiuterio klaviatūra, o sistema numato išvedimą, spausdinant tekstą kompiuterio vaizduoklyje.

Grafininė vartotojo sąsaja (angliškai Graphical user interfaces arba tiesiog GUI), 1984 iki dabar, tai vartotojo sąsaja kuri priima įvedima iš įvairių įrenginių, tokių kaip klaviatūra, pelė ir sugeneruoja aiškų grafinį išvedimą į kompiuterio vaizduoklį.

Grafininės vartotojo sąsajos smulkesnė raida:

1973 Xerox PARC Alto

1980 Three Rivers Computer Corporation Perq

1981 Xerox PARC Star

1983 Apple Lisa (strongly influenced by Xerox PARC systems); Visi Corp Visi On

1984 MIT Project X begins (contributors include Sun, IBM, HP, and DEC) Apple Macintosh; Digital Research GEM (known for its use on Atari ST computers)

1985 Amiga Workbench 1.0 ; Windows 1.0 ; Geos (for Commodore 64 and Apple II)

1986 First commercial release of X for UNIX

1987 Color Macintosh; Windows 2.03 ; Acorn Arthur

1988 NeXT; Apple GS/OS; IBM OS/2 Version 1.10 Presentation Manager (Microsoft wrote much of this)

1990 Windows 3.0 (First successful Microsoft GUI);Amiga Workbench 2; PC-GEOS

1992 Windows 3.1/3.11; IBM OS/2 Version 2.0 with Workplace Shell; Amiga Workbench 3

1993 Windows for Workgroups 3.11, NT 3.1/3.5

1994 XFree86, an Open Source implementation of X11R6, is released. (Linux gets a GUI) QNX Photon microGUI

1995 Windows 95 / Microsoft Bob; BeOS

1996 Windows NT 4.0; OS/2 Warp 4 with Workplace Shell

1997 Apple Mac OS 8

1998 Windows 98

1999 RISC OS 4

2000 Windows 2000

2001 Mac OS X released; Windows XP

Grafinių vartotojų sąsajų pavyzdžiai

Su grafine vartotojo sąsaja yra susidūrę dauguma vartotojų, nesvarbu kokią operacinę sistema Mac ar Windows naudojate, jų programos yra kilusios Xerox PARC laboratorijoje vėlyvais 1970-aisiais. Pirmoji tai naudojo Apple operacinė sistema pirmajame Macintosh kompiuteryje. Vėliau Microsoft kompanija panaudojo daug Apple idėjų savo pirmojoje Windows operacinės sistemos versijoje įdiegtoje į IBM tipo kompiuterius. Geriausi pavyzdžiai sistemų kurios naudoja grafinę vartotojo sąsają yra Mac OS, Microsoft Windows, Nextstep ir X Window sistema. Vėliau jos buvo išplėstos su pagalbiniais įrankiais tokiais kaip Motif (CDE), Qt (KDE) ir GTK+ (GNOME).

Pavyzdys kaip atrodo Windows XP grafinė vartotojo sąsaja.

Pavyzdys kaip atrodo Apple Mac OS X grafinė vartotjo sąsaja.

Pavyzdys kaip atrodo viena iš X Window sistemos KDE grafinė vartotojo sąsaja, naudojama Unix tipo sistemose.

Grafinės vartotojo sąsajos tipai

Grafinės vartotojo sąsajos (GUI) kurios nėra PARC (angl. Palo Alto Research Center) vartotojo sąsajos (PUI) ypač naudojamos kompiuteriniuose žaidimuose, o progresyvios grafinės vartotojo sąsajos yra kuriamos virtualiai realybei ir kol kas jos vis dar kūrimo stadijoje. Daug mokslinių grupių Šiaurės Amerikoje ir Europoje dirba prie kintamos vartotojo sąsajos (angliškai – Zooming User Interface - ZUI), kuris yra logiškas grafinės vartotojo sąsajos progresas, naudojantis 3D judėjimą su 2D arba 2,5D vektoriniais objektais.

Kai kurios grafinės vartotojo sąsajos yra sukurtos pagal negailestingus tam tikrų grupių reikalavimus (angliškai – vertical markets), taip pat žinomas kaip „specifines varotojo sąsajas“. Vienas tokios grafinės sąsajos atstovas yra dabar žinoma „touchscrean“ programinė įranga plačiai naudojama restoranuose. Naujausi telefonai ir žaidimų sistemos taip pat naudoja šią specifinę grafinę vartotojo sąsają.

„Žmogaus-kompiuterio sąveika“ tyrinėjimų sritis

Žmogaus-kompiuterio sąveika (angl. HCI - Human-computer interaction) yra sąveikos tarp žmogaus (vartotojų) ir kompiuterio studijos. Tai yra tarpdisciplinės studijos, susijusios su atskiromis sritimis, siejantis kompiuterių mokslą su daugeliu kitų studijų sritimis, jų tyrynėjimas. Sąsaja tarp vartotojo ir kompiuterio atsiranda per vartotojo sąsają (user interface), į kurią įeina techninė įranga (hardware) bei programinę įrangą (software), kartu sudarantys tam tikrą terpę.

HCI psichologija – viena iš HCI tyrinėjimo sričių, tiria pažinimo proceso teorijos taikymą ir naudotojo elgsenos empirinės analizės problematiką. Tiria tokius aspektus kaip suvokimas, dėmesio sutelkimas, dėmesio atkreipimas, motyvacija, atmintis, tikslingumas, sprendimų priėmimas.

Tinkamumas (angl. usability) prieš nenaudingumą (angl. usefulness) – HCI dizaino metodologija siekia sukurti patogią vartotojo sąsają, t.y. turi būti lengvai ir efektyviai valdoma. Tačiau dažniausiai pats pagrindinis reikalavimas yra vartotojo sąsajos naudingumas, kad atliktų reikiamus uždavinius, tačiau ne vartotojo sąsajos patogumas.

Intuityvumas (angl. intuitive) ir paprastumas (angl. natural). Prekiautojai džniausiai spekuliuoja programinės įrangos produktais, sakydami, jog jie "intuityvūs" bei "paprasti", t. y. lengva su jais dirbti, dažniausiai dėl to, jog turi skurdžią grafinę vartotojo sąsają. Daugelis HCI tyrimų laiko tokius tvirtinimus nepagrįstais (skurdus GUI (grafinė vartotojo sąsaja) dizainas gali būti netinkamas naudojimui).

Žmogaus-kompiuterio sąveikos traktavimas: informatikai akcentuoja programų sistemų sąsajų projektavimą, psichologams HCI – tai žmogaus poreikių tyrimo sritis, ergonomikos specialistams yra svarbūs asmens gebėjimai ir ribojimai.

Pagrindinis HCI tyrinėjimų srities tikslas – patobulinti sąveiką tarp vartotojų ir kompiuterių. Sukurta sąsają turi palengvinti bendravimą tarp kompiuterio ir vartotojo, technikos panaudojimas turi būti paprastas, patogus ir suprantamas paprastam vartotojui.

Programų vartotojo sąsajos pritaikymas neįgaliesiems

Naudodamiesi pritaikymo neįgaliesiems vedliu (angl. Accessibility Wizard), pasirinkčių piktograma (angl. Accessibility Options) ir kitomis valdymo skydo funkcijomis, galime keisti sistemos Windows išvaizdą ir elgseną. Galima keisti klaviatūros, ekrano, garso bei pelės funkcijas.

Galima sukurti sparčiuosius klavišus. Spartieji klavišai ypač naudingi asmenims, kuriems sunku naudotis pele kaip standartiniu naršymo įrenginiu.

Pvz.:

ALT+rodyklė į kairę pereiti atgal, prie ankstesnės Žinyno temos.

ALT+rodyklė dešinėn pereiti pirmyn, prie kitos Žinyno temos

CTRL+P spausdinti dabartinę Žinyno temą.

Kompiuterio pritaikymas vartotojams, turintiems regos negalia

Sukurti aplinka akliems arba silpnariagiams yra sunki užduotis. Norint tai įgyvendinti, reikia kad kompiuteris galėtų „skaityti“tekstą balsu. Beabejo vartotojas privalo turėti galimybe balsu įvesti tekstą. Tam reikalinga balso atpažinimo sistema. Taip pat egzistuoja periferiniai įrenginiai, tokie kaip Brailio spausdintuvai galintys atspausdinti tekstą Brailio raštu.

Apie 8% žmonių, daugiausiai vyrų, neskiria spalvų. Sistemoje spalva turėtų būti ne pagrindinis ar vienintelis būdas išskirti informaciją.

Kompiuterio pritaikymas vartotojams, turintiems fizinę negalia

Kai kurie žmonės negali naudotis įvesties įrenginiais (pale, klaviatūra). Norint pritaikyti aplinką tokiems vartotojams reikia pritaikyti ivedimo įrenginius, sukurti papildomą programinę įrangą, kuri padėtu vartotojui bendrauti su kompiuteriu.

Tipinės vartotojo sąsajos projektavimo klaidos

Vartotojo sąsajos projektavimo klaidos pasitaiko gana dažnai. Kartais programuotojas kurdamas programą nepajunta ar nepajaučia, kad jo programinėje sąsajoje spalvos nedera kartu arba yra informacijos perteklius/trūkūmas. Programuojant vartotojo sąsaja reikia mepamiršti keleto taisyklių: vartotojas neturi žinyno apie programą, o jei ir turi, tai jo neskaitys; faktiškai, vartotojas nieko negali skaityti, o jeigu gali, to nepadarys.

Tai nėra greižtos taisyklės, tačiau projektuojant patartina vadovautis jomis, jeigu siekiamas rezultatas yra kuo didesnis programos patugumas ir palankumas vartotojui.

Vartotojai, kurie dirba kompiuteriu nedaug, yra pakankamai baikštūs įvairiems pranešimams (lentelėms). Pranešimai, turi būti aiškūs, teisingai suformuluoti, neapkrauti pertekline informacija. Pvz.: uždarant programos langą nebutina rašyti daug žodžių, nes vartotojas, pamatęs ilgesnį pranešimą, gali išsigasti lyg padaręs kažką negero ir nepagalvojęs paspausti „ne“ mygtuką.

Informacijos perteklius pacioje programoje taip pat yra neigiama savybė. Vartotojai yra orientuoti į konkrečią informaciją. Todėl papildomi dalykai turi būti pasiekiami tik to reikalaujant (mygtukai, praplėtimai ir pan.)

Komponentų išdėstymas formoje turi būti taip pat patogus. Mygtukai ir laukai, skirti informacijos įvedimui turi būti patogiai išdėstyti.

Priedas Nr.2 Qt bibliotekos apžvalga - studijų medžiaga.

Qt istorija

Pirma kartą Qt įrankių rinkinys tapo prieinamas plačiajai visuomenei 1995 metų gegužę. Jo kūrėjai yra Haavard Nord ir Eirik Chambe-Eng. Haavardas ir Eirikas susitiko Norvegijos technologiniame universitete Trondheim mieste.

1990 vasarą, Haavardas ir Eirikas dirbo kartu vystydami C++ duomenų bazės programą, kuri turėjo apdoroti ultragarsinius vaizdus. Sistema turėjo veikti Unix, Mac ir Windows sistemose.

Haavardas iškėlė mintį, jog reikalinga objektiškai orientuota atvaizdavimo sistema. Diskusijų metu jie priėmė sprendimą, kad užsiims objektiškai orientuotos daugiaplatformės grafinės vartotojo sąsajos įrankių kūrimu.

1991 metais Haavardas pradėjo rašyti klases, kurios tapo Qt pagrindu. Jam talkino Eirikas dizaino klausimais. Sekančiais metais Eirikui kilo idėja apie signalus ir talpyklas (angl. signals and slots). Tai paprasta, bet galinga GUI programavimo paradigma. Iki 1993 abu jaunuoliai sukūrė pirmąjį Qt grafinį branduolį ir galėjo kurti savo valdiklius.

1994 metais du jauni programuotojai atėjo į rinką neturėdami pinigų, klientų ir baigtų produktų. Klasės prefiksui buvo pasirinkta raidė „Q“. Ši raidė atrodė gražiai. Raidė „t“ simbolizuoja įrankių rinkinį (angl. toolkit). Kompanija įsikūrė 1994 kovo 4.

1996 metais išleistos patobulintos Qt 1.0 ir 1.1 versijos. Tais pačiais metais pasirodė ir KDE projektas.

1997 metais išleistos Qt 1.2 ir Qt 1.3 Qt tapo standartiniu įrankiu C++ grafinės vartotojo sąsajos programavimui Linux terpėje.

Jau 1999 birželį pasirodė Qt 2.0. Ši versija gerokai skyrėsi nuo savo pirmtakių. Buvo smarkiai pakeista architektūra, produktas tapo stipresnis ir labiau išdirbtas. Taip pat papildyta 14 naujų klasių ir realizuotas Unikodo palaikymas. Atsirado atviro kodo ir viešoji licencijos.

2000 išleista nauja Qt, kuri galėjo būti X11 pakaitalu. Tais pačiais metais kompanija išleido pirmąją "Qtopija" versiją. Qt gavo apdovanojimus už geriausiai įtvirtintą Linux sprendimą 2001 ir 2002 metais.

2001 pasaulį išvydo Qt 3.0. Dabar jau Qt veikė Windows, Unix, Linux ir Mac OS sistemose. Qt 3.0 turėjo 42 naujas klases, o kodas viršijo 500 000 eilučių.

Mažiau nei per dešimtmetį Qt iš paslaptingo produkto, suprantamo tik grupelei žmonių tapo sistema, turinti dešimtis tūkstančių atviro kodo vystytojų visame pasaulyje.

Qt yra daugiaplatformė

Qt API ir įrankiai yra suderinami visose palaikomose platformose. Tai leidžia programų kūrėjams išmokti vieną API ir kurti programas, kurių veikimas visiškai nepriklausytu nuo naudojamos platformos.

Qt veikia šiose platformose :

Qt/Windows (Microsoft Windows XP, 2000, NT 4, Me/98)
Qt/Mac (Mac OS X)
Qt/X11 (Linux, Solaris, HP-UX, IRIX, AIX, kiti Unix variantai)
Qtopia Core - Qt prievadas integruotas Linux sistemoje.

Qt apima skirtingas platformas – specialias Unix API, Windows ir MAC taip pat APIs naudojamas failų tvarkymui, tinkliniam darbui, procesų valdymui, duomenų bazių prieigai ir kitoms operacijos atlikti.

Qt programos veikia visose pagrindinėse platformose sukompiliavus iš vieno šaltininio kodo (angl. source code).

Qt lengva naudoti

Qt programuotojai turi išmokti tik vieną API tam kad galėtų rašyti programas, kurios veikia praktiškai visur.

Programuotojui nebereikia jaudintis dėl skirtingų sąsajų kūrimo skirtingoms platformoms ir versijoms. Galima skirti daugiau laiko ir dėmesio programos kokybei.

Įveikti kūrimo sunkumus

Qt Dizaineris: galingas grafinės vartotojo sąsajos kūrimo įrankis. Qt kalbininkas: įrankis paspartinantis programinės įrangos pritaikymą įvairioms kalboms. Qt asistentas: dokumentacijos ruošimo įrankis. Ir Qt auganti įvairiapusė ekosistema, praktiškos programos (angl. utility) ir mokymasis, užtikrina, visokeriopą pagalbą.

Qt – daugiaplatformis C++ vystymas

Qt yra aukštos kokybės, daugiaplatformių, greitų programų vystymo standartas. Tai apima C++ klasės biblioteka ir įrankiai naudojami kurti daugiaplatformes ir internacionalines programas.

Qt klasių bibliotekos tikslas

Qt klasių bibliotekos tikslas yra pateikti beveik pilną rinkinį daugiaplatformių aplikacijų infrastruktūrinių klasių visų tipų C++ programoms. Qt siūlo laiko patikrintą pagrindinių aplikacijų ir objektų modelį, kuris apima galimybes valdyti įvykius, gijas (angl. threat) ir procesus.

Qt klasės bibliotekos

Qt klasių biblioteka yra sparčiai auganti biblioteka iš daugiau nei 400 C++ klasių, kurios apima visą infrastruktūrą, reikalingą baigtinėms programoms kurti ir vystyti. Elegantiška Qt API turi brandžius objektinius modelius, didelę klasių kolekciją, pritaikyta GUI programavimui, duomenų bazių kūrimui, tinkliniam darbui, XML, OpenGL integracijai ir kitiems veiksmams.

Qt klasės bibliotekos turinys iš pirmo žvilgsnio

Qt klasės bibliotekų tikslas yra tiekti beveik pabaigtas daugiaplatformių programų klases visų tipų C++ taikomoms programoms.

Qt siūlo tvirtą, laiko patikrintą pagrindinį programos ir objekto modelį su įvykių pasikartojimo apdorojimo ir valdymo galimybėmis. Qt taip pat palaiko signalų ir talpyklų mechanizmą, kuris leidžia laisvą komponentų jungimą, nepriklausomai nuo failų tipų. Qt 4 versijoje, bibliotekos buvo išskaidytos į daug funkcionalių komponentų, suteikiančių galimybę paprasčiau skirstyti ir testuoti.

Pagrindinė Qt savybė yra platus GUI kūrimo ir valdymo įrankių rinkinys. Tai leidžia kurti savo valdymo sistemą.

Qt yra daug daugiau nei paprasta GUI įrankių rinkinys, nes galima atlikti begale papildomų veiksmų tokių kaip: failų tvarkymas, darbas tinkle, procesų valdymas, duomenų bazių prieiga, įvykių pasikartojimo valdymas, XML, OpenGL integracija ir kt. Šios funkcijos veikia bet kurioje platformoje, kuri palaiko Qt.

Galimybės

Qt tikslas yra padaryti programuotojus produktyvesniais. To siekiama įvairiais būdais: atminties valdymu, išdėstymu, pritaikymu įvairioms kalboms, pasitelkiant signalų ir talpyklų mechanizmą.

Lengvesnis atminties valdymas

Qt supaprastina programavimą ir testavimą valdant sudėtingas atminties vietas. Programuotojas gali susikoncentruoti tik į aukščiausio lygio objektus. Žemesnio lygio objektai yra automatiškai ištrinami, kai ištrinamas juos naudojantis aukščiausio lygio objektą.

Atminties valdymo poreikis nėra visiškai pašalintas. Tam pasiekti yra mažinamas kalbų kolekcijos ir C++ resursų naudojimas.

Qt dizaineris

Qt dizaineris yra galinga programa skirta kurti GUI ir formas. Tai leidžia greitai sukurti galingą vartotojo sąsają pritaikant ją konkrečiai kalbai ir platformai. Vienas arba integruotas Qt Dizaineris turi galingas priemones tokias kaip peržiūrėjimo režimas, automatinis objektų išdėstymas, palaikymą konkretiems objektams, papildomas funkcijas ir daug daugiau.

Galinga išdėstymo sistemos integracija

Qt Dizaineris turi galingą automatinę dydžių bei šriftų keitimo sistemą. Tai leidžia kūrėjams kurti dialogus kurie yra patrauklūs, atrodo profesionaliai ir nepriklauso nuo monitoriaus, šrifto dydžio, kalbos, stilių ir platformos, kurią pasirinks galutinis vartotojas.

Naujasis Qt 4 Dizaineris leidžia įterpti papildomus valdymo mygtukus į jau veikiančią sistemą. Tai dar labiau praplečia ir supaprastina programos naudojimą ir dialogų kūrimą.

Specialaus valdymo integracija

Qt yra lankstus ir plečiamas. Tai suteikia galimybe naudoti specialius valdymo mygtukus ir komandas. Valdymo komandos gali būti kuriamos nuo pradžių, arba panaudojant standartines iš Qt bibliotekos.

Qt Kalbininkas

Qt turi įrangius padedančius greičiau ir paprasčiau atlikti pritaikymo įvairioms kalboms procesus. Naudojant Qt kalbininką kūrėjų komandos gali išversti tekstą padidinant tikslumą ir pagreitinant lokalizacijos procesą.

Qt Kalbininkas

Qt kalbininkas turi sąsają, kuri sutrumpina ir padeda susisteminti grafinės vartotojo sąsajos vertimo procesą. Qt kalbininkas surenka visą vartotojo sąsajos tekstą, kuris bus rodomas galutiniam vartotojui ir pateikia jį paprastame lange. Kai vienas vartotojo sąsajos tekstas yra išverčiamas, programa pereina prie kito tok kol visi išverčiami. Tai pagreitina procesą ir suteikia daugiau tikslumo vertimams.

Platus unikodo palaikymas

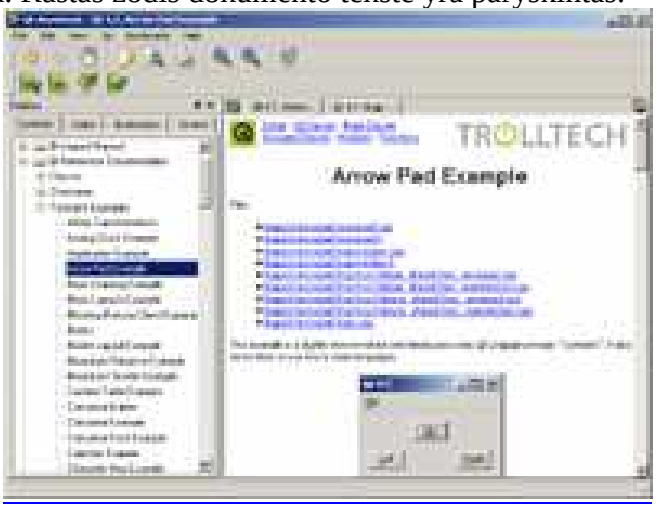
Qt unikodo palaikymas suteikia ramybę programų kūrėjams. Jie gali būti užtikrinti, jog programa veiks ir vartotojo, naudojančio ne vakarietiško (angl. Non-Western) stiliaus rašymo sistemą. Tai plati sistema, prižiūrinti dauguma pasaulio abėcėlių (įskaitant Arabų, Kinų, hebrajų, japonų ir korėjiečių) be žmogaus įsikišimo.

QT Asistentas

Pilnai pritaikomas konkrečioms programoms pagalbos failas ir dokumentacijos naršyklė. Naudojant Qt Asistentą programų kūrėjai ženkliai sumažina dokumentacijos rengimo laiko sąnaudas.

Qt Asistentas veikia panašiai kaip interneto naršyklė, pateikdamas dokumentaciją kaip puslapių su nuorodomis visumą. Galima puslapius sužymėti, taip pat yra standartiniai mygtukai pereiti vienu puslapiu pirmyn, atgal ir pakartotinai peržiūrėti jau aplankytus puslapius. Puslapiams atvaizduoti naudojama HTML. Informacijos rengėjai gali naudoti savo įrankius dokumentacijai parengti ir talpinti į Qt Asistentą.

Qt Asistentas taip pat turi paieškos funkciją. Galima ieškoti pagal raktinį žodį (funkcijų ir klasių) arba pagal kelis žodžius (dokumentacijoje). Dokumentai, rasti paieškos metu yra išdėstomi svarbumo tvarka. Rastas žodis dokumento tekste yra paryškintas.



Paskirstyta pagalbos sistema

Qt vartotojams nebereikia rūpintis savos pagalbos sistemos kūrimu. Qt Asistentas leidžia naudotis duomenų bazėse sukaupta medžiaga apie Qt biblioteka sukurtas programas. Programų kūrėjai gali kurti savo dokumentacijas, kuriomis gali dalintis su kitais.

Qtopia

Qtopia – tai delninio kompiuterio/telefono sistema veikianti Qt pagrindu.

Qtopia yra pirmoji taikomųjų programų aplinka, kuri perkelia personalinio kompiuterio naudojimosi galimybes į mobiliuosius aparatus. Qtopia jau yra taikoma naujame “Sharp” firmos Zaurus SL-5500 angl. PDA. Qtopia naudinga gamintojams, prekiautojams, informacinių technologijų korporacijų vadybininkams ir patiems vartotojams. Produktas leidžia gamintojams pagaminti galingus, intuityvius ir efektyvius ištekliais asmeninius skaitmeninius padėjėjus (angl. Personal Digital Assistant, PDA), internetinius reikmenis ar bet koki kitą mobilų aparatą, kuris veikia naudojant LINUX aplinką.

Į Qtopia komplektą įeina ir platus programų skirtų našumo didinimui, informacijos tvarkymui, interneto naršymui, pramogoms ir įvairių operacinių sistemų darbalaukių tvarkymui. Tai apima adresų knygelę, kalendorių, darbotvarkės grafiką, teksto redagavimo programas, MPEG ir MP3 grotuvus, tarptautinis miestų laikrodis, žaidimus

Pilną Trolltech produktų liniją sudaro Qtopia, Qt/Desktop, Qt/Embedded, Qt Designer; spartus aplikacijų plėtojimo įrankis ir Qt Linguist – internacionalizavimo įrankis.

Qtopia yra tiesiogiai pasiekiama ir nemokama visiems atviro šaltinio vartotojams.

Qt/Embedded

Qt/Embedded įdiegiamas ir veikia naudodamas labai mažą dalį atminties bet kokiame aparate, kuriame įdiegti Linux, nenaudojama net X11.

Qt/Embedded naudoja tą patį API, kaip ir plačiai naudojamos Qt/Windows ir Qt/X11 versijos.

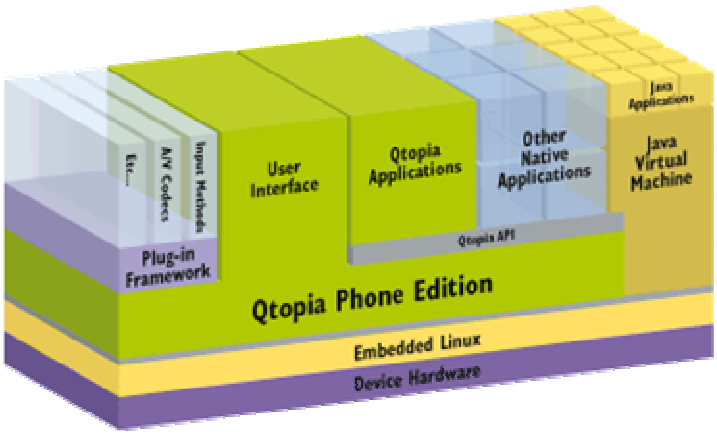
Mažiau yra daugiau

Qt/Embedded yra modifikuojamas ir transformuojamas. Jūs galite išsirinkti jums reikiamus Qt bruožus ir tiesiog išmesti tuos, kurie jums nereikalingi. Kadangi Qt/Embedded veikimas nėra pagrįstas X11, tai jo atminties reikalavimai yra žymiai mažesni. Renkantis reikiamus aparato bruožus, atminties reikalavimai gali kisti nuo 800KB iki 3MB. Be to, aplikacijos įrašytos su Qt užima daug mažiau vietos, nei naudojantis kitais įrankiais. Qt gali būti naudojamas pradedant mažiausiais aparatais ir baigiant aukšto technologinio lygio darbinėmis stotimis.

Qtopia Telefono versija

Qtopia, pritaikyta telefonams yra plati programinė platforma ir Linux pagrindu veikiančių telefonų vartotojo sąsaja. Tai praplėsta iš anksto integruotomis programomis, leidžiančiomis dizaineriams ir programuotojams sukurti puikias ypatybes turinčius telefonus.

Naujovės ir diferencijavimas



Qtopia įveda platformą, naudojamą pramonės lyderių tokių kaip Google, Adobe ir Skype į Linux mobilią erdvę. Ši C++ klasės programų sistema yra optimizuota mobiliems įrenginiams su ribota atmintimi. Qtopia draugiškos vartotojo sąsajos įrankis ir nauja programų sąsaja suteikia galimybę greitai ir efektyviai vystyti sistemą.

Qtopia telefono versijos ypatybės

Adaptuojama vartotojo sąsaja

Qtopia adaptuojama sąsaja palaiko daug ekrano dydžių iki 24 bitų taške spalvomis (16 milijonų spalvų) ir langų persišviečiamumą. Taip pat galima naudoti vektorinę grafiką, animuotas ikonas, atlikti keletą operacijų vienu metu (angl. multitasking). Yra galimybė naudoti dvigubą ekraną atlenkiamiems įrenginiams.

Išplėstas internacionalizavimas

Užkariauti globalią rinką yra lengviau naudojant Qtopia telefono versiją, kuri palaiko išplėstą unikodą ir tekstą, rašomą iš dešinės į kairę. Ekrano dydis, įskaitant valdiklio dydį ir vietą yra automatiškai pataisomas išvertus tekstą ir atvaizduojant iš dešinės į kairę rašomam tekstui.

Pridedama sistema

Qtopia platformos galimybės gali būti išplėstos naudojant pridedamą sistemą. Sistema siūlo gerai veikiančių kūrimo aplinką naujoms galimybėms: įvedimo metodai, šriftai, modemo tvarkyklės, media kodai ir kitos naudingos galimybės.

Dinaminiai įgalintojai

Qtopia platformos įgalintojai palaiko kitas telefonų versijos galimybes, įtraukdamos SQL duomenų bazę paprastesniam duomenų valdymui ir pridedamą sistemą lengvesniam naujų galimybių tokiu kaip įvedimo metodai ir šriftai valdymui.

Jungiamumas

Qtopia telefono versija palaiko plačią bevielio prisijungimo būdų sritį įskaitant AT-komandinio modemo integraciją, 802.11 su dinaminiu atradimu, Bluetooth OPP ir IrDA.

PDA versija

versija 2.2

Pilnas valdymas

Gamintojai gali keisti ir pritaikyti Qtopia sau. Nustatymai, pritaikymas ir suasmeninimas gali būti atliekami ir galutinio vartotojo iniciatyva.

Lankstumas

Qtopia suteikia kūrėjams inovacijų laisvę, kartu pateikdami pilną programos kodą, dokumentaciją ir minimalius techninės įrangos poreikius. Populiari kūrimo aplinka skatina kūrybiškumą ir suteikia galimybę praplėsti sistemą papildomomis programomis.

Qtopia PDA versijos ypatybės

Vartotojo sąsaja

Qtopia konfigūruojama sąsaja yra sukurta skirtingiems ekranų dydžiams, įskaitant 240x320 ir 480x640 taškų. Ir gali būti keičiami tarp kraštovaizdžio (angl. landscape) ir portretinio (angl. portrait) vaizdo padėties. Galima naudoti iki 24 bitų ekraną, animuoti grafinę vartotojo sąsają. Qtopia palaiko langų persišvietimą.

Internacionalizavimas

Qtopia PDA versija naudoja Unikodą ir turi supaprastintą kinų, anglų, amerikiečių, japonų, vokiečių, prancūzų, Portugalų ir italų kalbas. Yra galimybė laisvei persijungti iš vienos kalbos į kitą. Išsidėstymo variklis automatiškai pataiso mygtukų ir užrašų dydžius išverstam tekstui.

Sinkronizacijos sistema

Vartotojai gali sinkronizuoti savo kontaktus, kalendorinius įvykius, multimedijinius failus ir dokumentus naudodami Qtopia darbastalį arba Trolltech PIM rinkinį ar Outlook Express.

Reikalavimai

Qtopia telefono versija

Kūrimo aplinka:	Linux branduolys 2.4 ar naujesnis GCC versija 3.3.5 ar aukštesnė
Techninė platforma:	Visi procesoriai, kuriuos palaiko Linux su C++ kompiliatorium ir kadrų buferio tvarkykle. Palaikoma ARM®, x86®, MIPS®, PowerPC®

Qtopia PDA versija

Kūrimo aplinka:	Linux branduolys 2.4 ar naujesnis GCC versija 2.95 ir aukštesnė
Pėdsakas:	Minimaliai 8MB Flash ROM, 16MB RAM (įskaitant Linux) Standartiškai 16MB Flash ROM, 32MB RAM (įskaitant Linux)
Hardware platform:	Visi procesoriai, kuriuos palaiko Linux su C++ kompiliatorium ir kadrų buferio tvarkykle. Palaikoma ARM®, x86®, MIPS®, PowerPC®

Priedas Nr.3 Qt praktinė užduotis – studijų medžiaga

Praktinė užduotis

1. Parašykite programą, kuri parodytu langą su užrašu ir mygtuku. Mygtukas turi būti veikiantis. (Pavyzdys programa „Hello Qt“).

2. Išsiaiškinkite kaip veikia ir parašykite programą, kuri susietu du objektus : QSpinBox ir QSlider. (Pavyzdys Amžiaus programa).

Nekomercinės versijos diegimo instrukcija

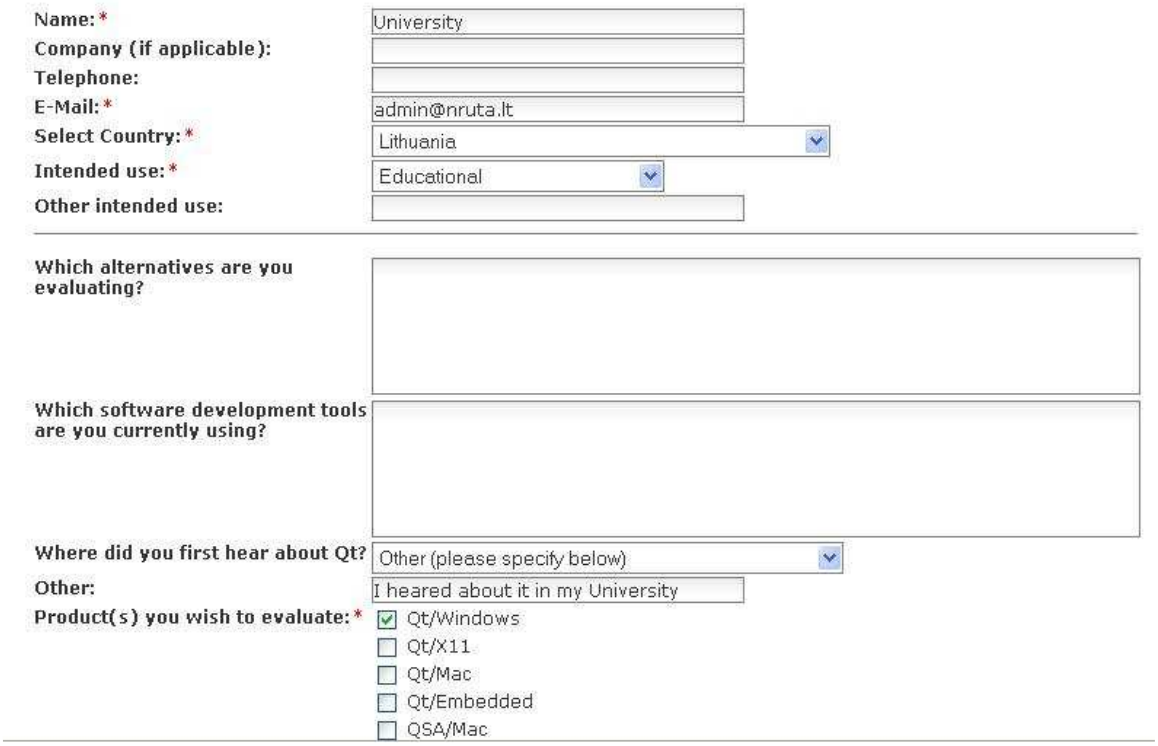
1. Registracija internete.

Norint persisiųsti nekomercinę versiją reikia užsiregistruoti sistemoje.

Registruokite savo vardą, elektroninio pašto adresą ir mes jums atsiūsime licensijos numerį 30 dienų bandomajam laikotarpiui.

Įsitikinkite jog įvedėte savo elektroninio pašto adresą teisingai.

Būtini laukeliai pažymėti žvaigždute *



Užpildžius registracijos formą gauname laišką su nurodymais.

2. Instaliavimas Windows aplinkoje

Gautame laiške aprašomos programos diegimo instrukcijos.

Persisiuntus paketą, aktyvuojame jį dvigubu pelės klavišo paspaudimu ir sekant instrukcijas instaliuojame.

Instaliacinė programa paprašys rakto:

Licensee name: University
Qt License Key: BGKX-RM5-Q4M-2CX-LYFX-VTM87-B905

SVARBU: raktą reikia įvesti būtent taip, nes kitu atveju instaliuoti nebus galima.

Instaliavimus programą verta peržiūrėti pavyzdžius. Taip pat patartina perskaityti dokumentaciją.

Išinstaliavimas

Norint išinstaliuoti programą tereikia pasirinkti valdymo skyde (angl. Control Panel) meniu punktą „pridėti/pašalinti programas“.

Persisiuntimas

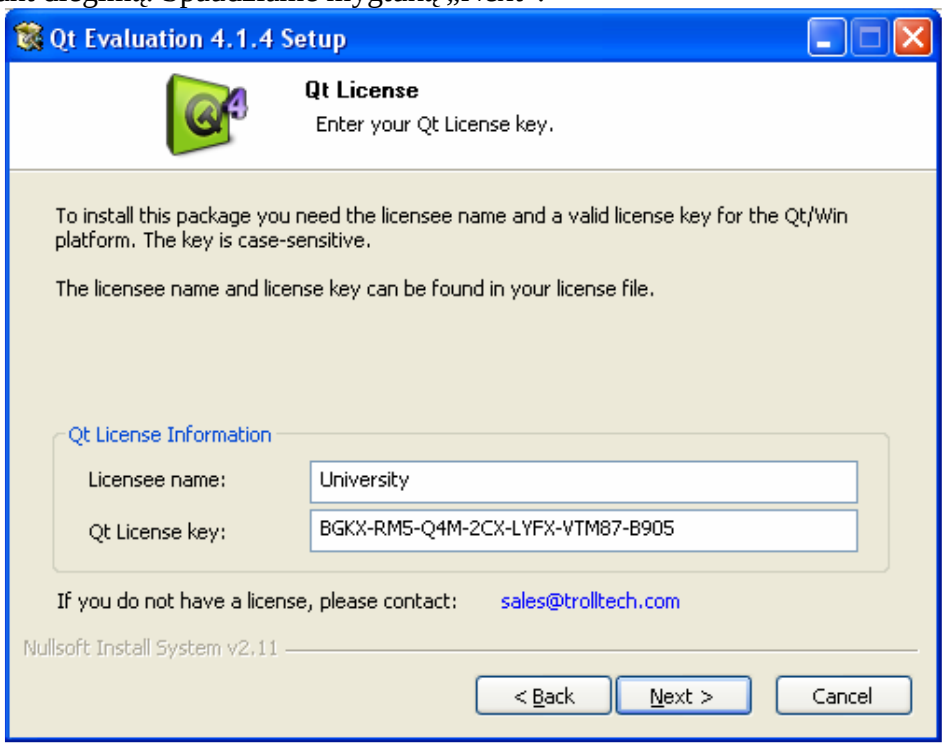
Nekomercinė versija gali būti persisiųsta iš:
Dėl Microsoft Visual Studio 2005
http://www.trolltech.com/developer/download/qt-win-eval-4.1.4-vs2005.exe

Dėl Microsoft Visual Studio .NET 2003
http://www.trolltech.com/developer/download/qt-win-eval-4.1.4-vs2003.exe

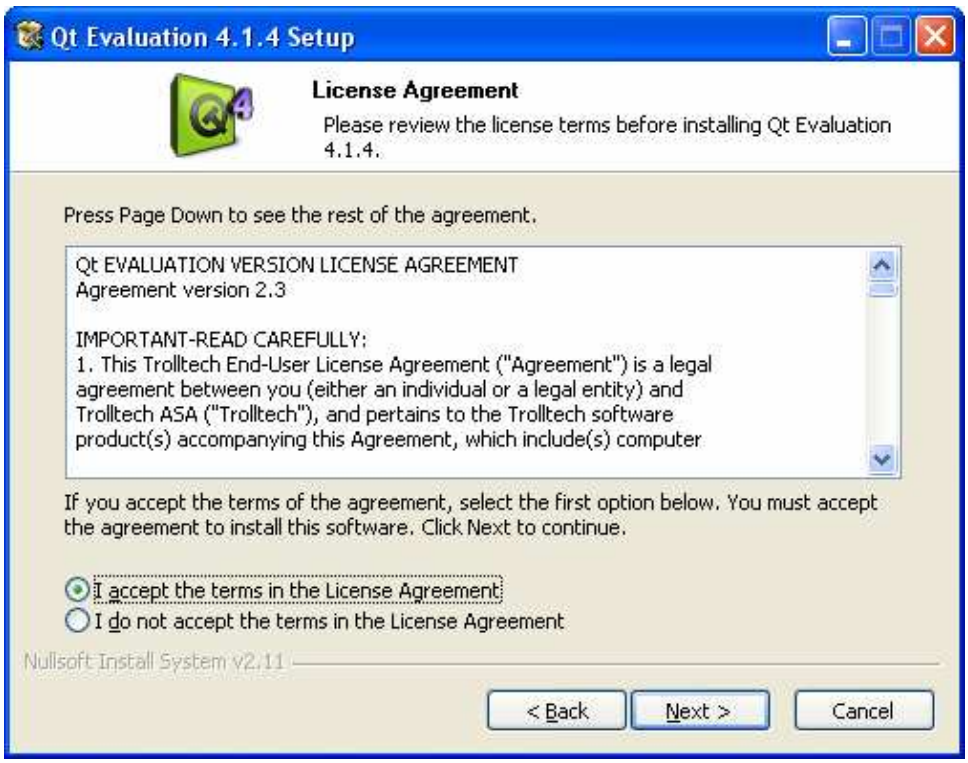
Dėl Microsoft Visual C++ 6
http://www.trolltech.com/developer/download/qt-win-eval-4.1.4-vc60.exe

3. Qt diegimas Windows aplinkoje

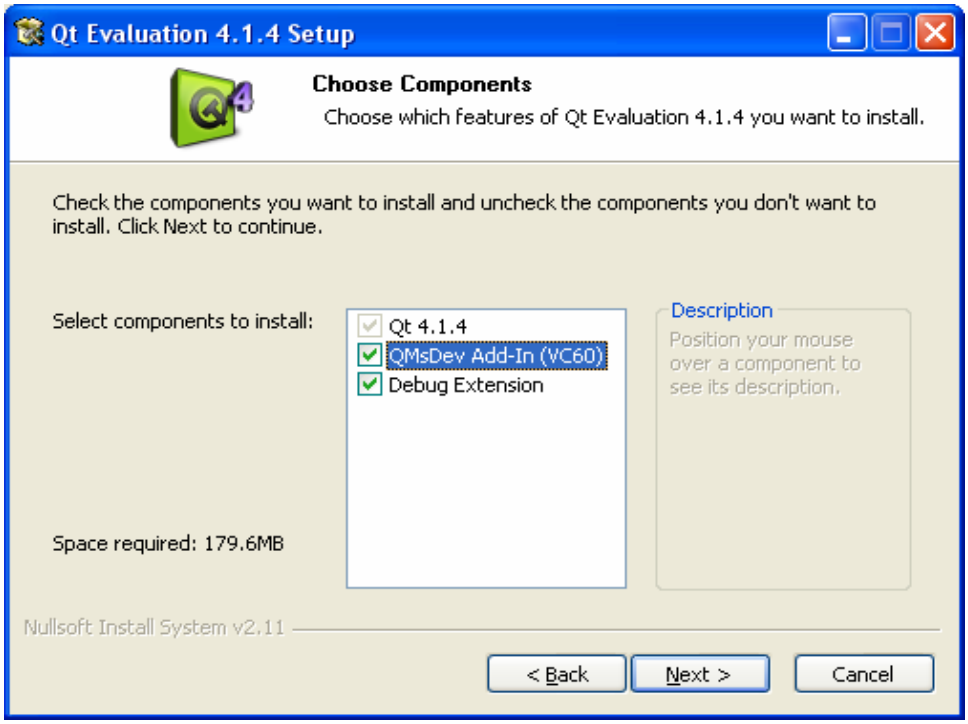
Aktyvavus programą, diegimo vedlys duoda trumpus nurodymus ką reikia padaryti prieš pradedant diegimą. Spaudžiame mygtuką „Next“.



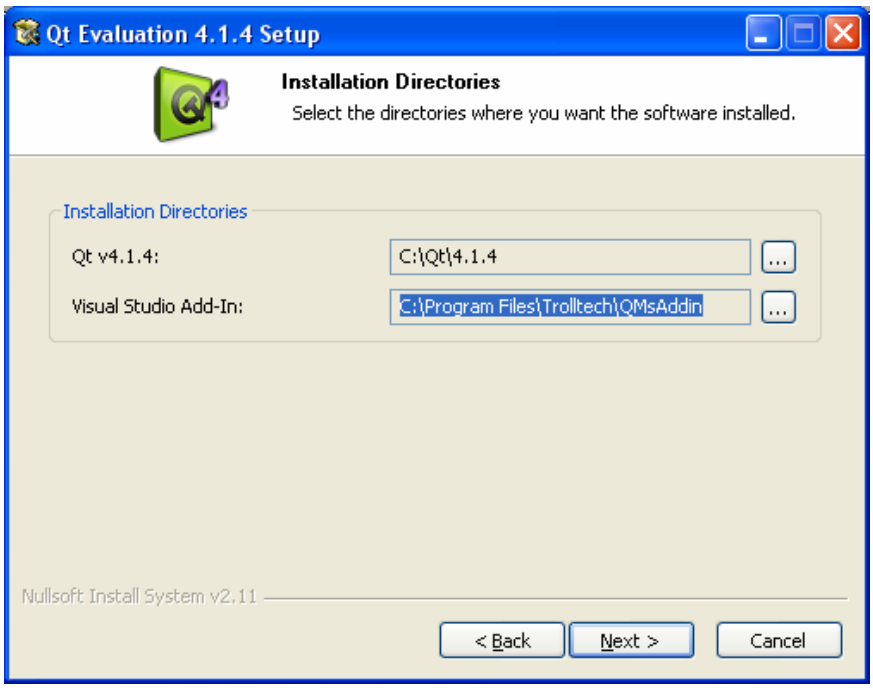
Antrajame lange įrašome vardą bei serijinį numerį, kurį mums atsiuntė elektroniniu paštu. Spaudžiame mygtuką „Next“.



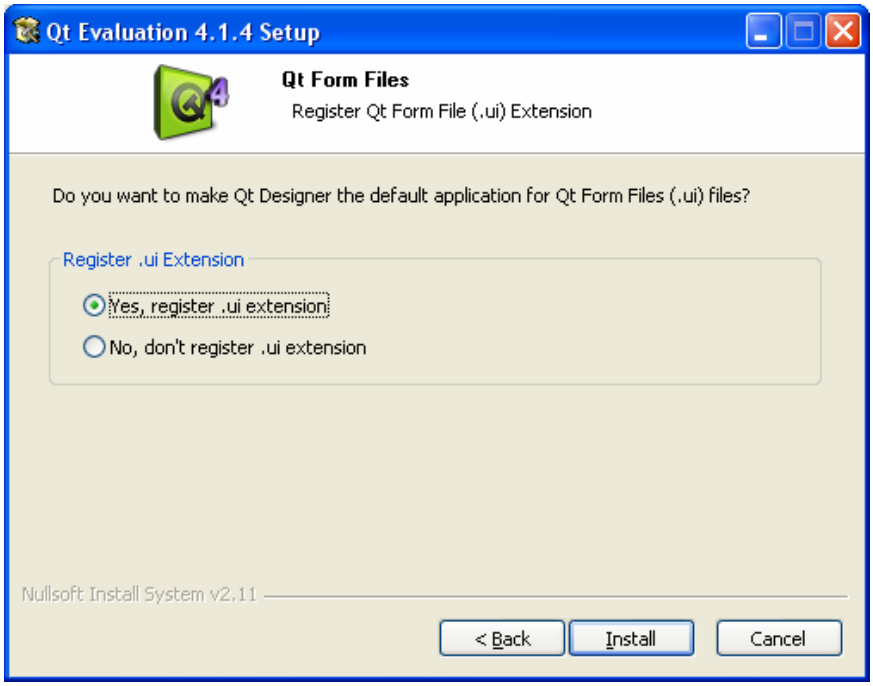
Norint tęsti diegimą, būtina sutikti su sutarties sąlygomis. Pažymime “I accept the terms in the License Agreement”. Spaudžiame mygtuką „Next“.



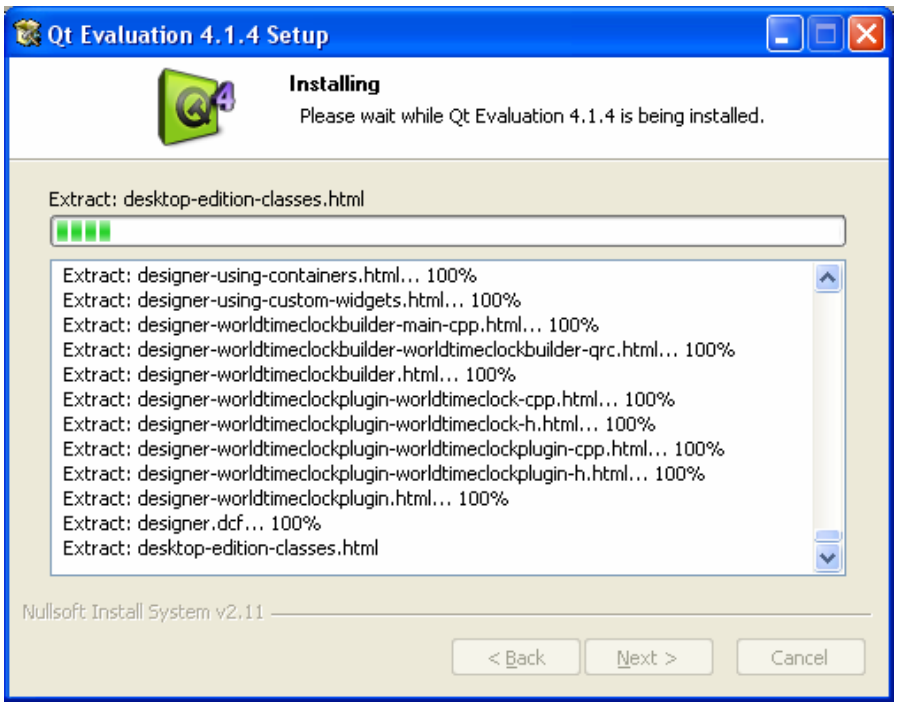
Pasirenkame kurias programos dalis norime diegti. Spaudžiame mygtuką „Next“.



Išsirenkame programos direktoriją. Pagal nutylėjimą ji bus diegiama į “C:\Qt\4.1.4”(skaičiai gali skirtis jei bus diegiama kita versija). Visual Studio priedai diegiami į “C:\Program Files\Trolltech\ QMsAddin”. Jei direktorijos netenkina, mes galime jas pakeisti paspaudus  . Spaudžiame mygtuką „Next“.



Pasirenkame jog norime registruoti Qt kaip pagrindinę programą failams su *.ui plėtiniu atidaryti. Spaudžiame mygtuką „Next“.



Vyksta programos diegimas. Reikia truputi palaukti.

Pasibaigus programos diegimui galima pasirinkti ar norime peržiūrėti dokumentaciją ir pavyzdžius. Jei taip – reikia pažymėti punktą “Show Documentation”. Spaudžiame mygtuką “Finish”.

4. Qt diegimas Mac OS

Mac OS X diegimas vyksta iš terminalo. Jeigu sistemoje n4ra instaliuoto C++ kompiliatoriaus, tai reikia padaryti prie6 diegiant Qt. Papra62iausias b8das tai padaryti yra instaliuoti GCC iš Apple Developer įrankių CD.

1. Išarchyvuojame failą iš CD:
2. `cd /Developer`
3. `tar zxf /Volumes/Qt\ 3\ Programming/mac/qt-mac-free-3.2.1.tar.gz`

Archyvas išskleidžiamas į `/Developer/qt-mac-free-3.2.1`.

4. Sukuriame nuorodą į direktoriją `/Developer/qt`:

```
5. ln -sf qt-mac-free-3.2.1 qt
```

6. Pasirenkame Qt aplinką.

Kintamieji nustatomi skirtingai priklausomai nuo naudojamos aplinkos. Jei jūsų vardas “kelly”, jūs galite sužinoti kokią aplinką naudojat, pasitelkiant “finger” komandą:

```
finger kelly
```

Jei jūsų aplinka yra `bash`, `ksh`, `zsh`, or `sh`, pridėkite papildomas eilutes prie `.profile` failo namų direktorijoje:

```
QTDIR=/Developer/qt
PATH=$QTDIR/bin:$PATH
MANPATH=$QTDIR/doc/man:$MANPATH
DYLD_LIBRARY_PATH=$QTDIR/lib:$DYLD_LIBRARY_PATH
export QTDIR PATH MANPATH DYLD_LIBRARY_PATH
```

Jei jūsų aplinka yra `csh` or `tcsh`, pridėkite sekančias eilutes prie `.login` failo:

```
setenv QTDIR /Developer/qt
setenv PATH $QTDIR/bin:$PATH
setenv MANPATH $QTDIR/doc/man:$MANPATH
setenv DYLD_LIBRARY_PATH $QTDIR/lib:$DYLD_LIBRARY_PATH
```

Jeigu jums išmetė klaidą "undefined variable", reikia paskutines dvi eilutes pakeisti tokiomis:

```
setenv MANPATH $QTDIR/doc/man
setenv DYLD_LIBRARY_PATH $QTDIR/lib
```

Kai viską atlikote teisingai, nustatymus reikia suaktyvinti. Paprasčiausias kelias tai padaryti yra uždaryti terminalo langą ir atidaryti naują.

7. Paleiskite konfigūravimo įrankį naujame terminale pasirinkę jums patinkančiomis nuostatomis Qt bibliotekos ir įrankių kūrimui:

```
8. cd $QTDIR
9. ./configure
```

Galima paleisti `./configure -help` ir pamatyti sąrašą konfigūracijos ypatybių.

Pavyzdžiui galima naudoti `–thread` įpatybę norint sukurtį gyjinę (angl. threaded) bibliotekos versiją

10. Tipas `make`.

11. Įgalinti programos paleidimą iš paieškos laukelio:

Jei naudosime `-static` įpatybę, mūsų programos turės savyje Qt biblioteką. Programas bus galima paleisti iš paieškos laukelio automatiškai. Priešingu atveju programa turės naudotis bibliotekomis, esančiomis jūsų sistemoje. Tai galima įgyvendinti sukuriant dvi simbolines nuorodas:

```
ln -sf $QTDIR/lib/libqt.3.dylib /usr/lib
ln -sf $QTDIR/lib/libqui.1.dylib /usr/lib
```

Jeigu kuriate daug gijų turinčią programą, pakeiskite `libqt.3.dylib` į `libqt-mt.3.dylib` pirmoje `ln` komandoje.

Kuriant šias nuorodas, gali reikėti administratoriaus teisių. Tokiu atveju turime įvykdyti komandas :

```
sudo ln -sf $QTDIR/lib/libqt.3.dylib /usr/lib
sudo ln -sf $QTDIR/lib/libqui.1.dylib /usr/lib
```

Jeigu neturite administratoriaus teisių, arba norite diegti Qt vietiniam vartotojui, galima naudoti nuorodas:

```
ln -sf $QTDIR/lib/libqt.3.dylib $HOME/lib
ln -sf $QTDIR/lib/libqui.1.dylib $HOME/lib
```

kai pir minėjome ankščiau, jei kuriate daug gijų turinčią programą, būtina pakeisti `libqt.3.dylib` į `libqt-mt.3.dylib`.

Jeigu norite keisti Qt diegimo procesą arba įvyko klaida, peržiūrėkite INSTALL failą, kuris yra $QTDIR.

5. Qt diegimas X11

Norint diegti Qt X11 platformoje, jūs turite būti prisijungęs kaip root vartotojas, nebent yra nustatytos atskiros privilegijos pasirinktiems katalogams.

1. Pakeičiame direktoriją į kurią norime diegti Qt. pavyzdžiui:

```
2. cd /usr/local
```

3. Išskleidžiame archyvo failus:

```
4. cp /cdrom/x11/qt-x11-free3.2.1.tar.gz.
5. gunzip qt-x11-free-3.2.1.tar.gz
6. tar xf qt-x11-free-3.2.1.tar
```

Sukuriama direktorija `qt-x11-free-3.2.1`, jeigu jūsų CD-ROM įrenginys yra sumontuotas kaip `/cdrom`.

7. Įdiegiame keletą aplinkų.

Kintamieji nustatomi skirtingai priklausomai nuo naudojamos aplinkos. Jei jūsų vardas “gregory”, jūs galite sužinoti kokią aplinką naudojat, pasitelkiant “finger” komandą:

```
finger gregory
```

Jei aplinka yra `bash, ksh, zsh`, or `sh`, pridėkite eilutes `.profile` faile namų direktorijoje:

```
QTDIR=/usr/local/qt-x11-free-3.2.1
PATH=$QTDIR/bin:$PATH
MANPATH=$QTDIR/doc/man:$MANPATH
LD_LIBRARY_PATH=$QTDIR/lib:$LD_LIBRARY_PATH
export QTDIR PATH MANPATH LD_LIBRARY_PATH
```

Jei jūsų aplinka yra `csh` arba `tcsh`, pridėkite papildomas eilutes `.login` faile:

```
setenv QTDIR /usr/local/qt-x11-free-3.2.1
setenv PATH $QTDIR/bin:$PATH
setenv MANPATH $QTDIR/doc/man:$MANPATH
setenv LD_LIBRARY_PATH $QTDIR/lib:$LD_LIBRARY_PATH
```

Jei išmeta klaidą "undefined variable", Pakeiskite paskutines dvi eilutes į :

```
setenv MANPATH $QTDIR/doc/man
setenv LD_LIBRARY_PATH $QTDIR/lib
```

Priklausomai nuo aplinkos naudojimo, jeigu jūs diegsite Qt programą AIX aplinkoje, reikai pakeisti visus `LD_LIBRARY_PATH` į `LIBPATH`. Jeigu diegiame HP-UX aplinkoje, pakeiskime `LD_LIBRARY_PATH` į `SHLIB_PATH`.

Atlikus pakeitimus jums reikia prisijungti iš naujo arba atnaujinti `.profile` arba `.login` failus. Priešingu atveju negalite tęsti.

Paleidžiame konfigūravimo įrankį su jūsų pasirinktomis ypatybėmis. Sukuriame Qt biblioteką su įrankiais:

```
8. cd $QTDIR
9. ./configure
```

Galima paleisti `./configure -help` norint gauti konfigūracijos ypatybių sąrašą.

10. Tipas `make`.

Jeigu norite keisti Qt diegimo procesą arba įvyko klaida, peržiūrėkite INSTALL failą, kuris yra $QTDIR.

6. C++ ir Qt sąveika

Šiame skyriuje parodyta kaip sujungti paprasta C++ su Qt tam kad sukurti kelias grafinės vartotojo sąsajos (angl. GUI) programas.

“Hello Qt”

Paprasta Qt programa:

```
1   #include <QApplication>                    //Pridedamas Qapplication klasės apibrėžimas
2  #include <QPushButton>                  //Pridedamas QPushButton klasės apibrėžimas
3
4  int main(int argc, char *argv[])                          //Programos pradžia
5   {
6    QApplication app(argc, argv);                          //Sukuriamas app objektas naudojant Qapplication
7
8      QPushButton hello("Hello world!");          //Sukuriamas mygtukas
9      hello.resize(100, 30);                     //Nustatomas mygtuko dydis
10
11      hello.show();                              //Mygtukas parodomas
12      return app.exec();                         //Gražinamas valdymas
13   }
```

Iš pradžių išsiaiškinsime kiekvieną eilutę, tada pamatysime kaip kompiliuoti ir paleisti programą.

Eilutės 1 ir 2 apibrėžia `QApplication` ir `Qlabel` klases.

6 eilutė sukuria `QApplication` objektą programos resursams valdyti. `QApplication` konstruktorius reikalauja `argc` ir `argv` nes Qt palaiko kelis komandinės eilutės argumentus.

8 eilutė sukuria mygtuką,ant kurio bus parašyti įžymūs žodžiai „Hello World“.

9 eilutė nustato mygtuko dydį.

11 eilutė padaro mygtuką matomą. Objektai visada sukuriami paslėpti, tam kad prieš parodant galėtumėme pakeisti mums reikiamas ypatybes.

12 eilutė perduoda programos valdymą Qt. Šiame taške programa pereina į laukimo režimą ir laukia vartotojo veiksmų tokių kaip pelės mygtuko paspaudimo.

Vartotojo veiksmai sukuria įvykius, taip pat vadinamus pranešimais. Programa į tai reaguoja atlikdama konkrečią funkciją.

“Hello” Windows XP aplinkoje

Laikas testuoti programą. Visų pirma reikės instaliuoti Qt 4.1.4. Taip pat reikės “Hello” programos kodo. Failas pavadintas `hello.cpp`. Galima išsaugoti kodą patiems, arba nusikopijuoti iš pavyzdžio direktorijos `\examples\chap01\hello\hello.cpp`.

Iš komandinės eilutės pakeiskite direktoriją į `hello`, ir parašykite

```
qmake –project
```

sukuriame nepriklausomos platformos failą (`hello.pro`), tada rašome

```
qmake hello.pro
```

sukuriame projekto failą, kuris yra pritaikytas konkrečiai platformai. Paleiskite `make` tam kad sukurtumete programą ir paleiskite ją įrašydami `hello` Windows aplinkoje, `./hello` Unix, ir `open hello.app` Mac OS X sistemose. Jei naudojate Microsoft Visual C++, jums reikės rašyti `nmake` vietoje `make`. Galite sukurti Visual Studio projekto failą iš `hello.pro` parašydami

`qmake -tp vc hello.pro` ir apdoroti bei kompiliuoti programą pasinaudojant Visual Studio.

Quit programa


```
#include <QApplication>              //Pridedamas Qapplication klasės apibrėžimas
#include <QFont>                                 //Pridedamas QFont klasės
apibrėžimas
#include <QPushButton>                                      //Pridedamas QPushButton
klasės apibrėžimas

    int main(int argc, char *argv[])        //Programos pradžia
    {
        QApplication app(argc, argv);    //Sukuriamas app objektas naudojant
Qapplication

        QPushButton quit("Quit");           //Sukuriamas mygtukas "Quit" su "Quit"
užrašu
        quit.resize(75, 30);                            //Nustatomas mygtumo dydis
        quit.setFont(QFont("Times", 18, QFont::Bold));
                                                  //Nustatomas šriftas, jo dydis
        QObject::connect(&quit, SIGNAL(clicked()),
                                 &app, SLOT(quit()));
                               //sujungiami signalas "clicked" su pasekme "Quit"
        quit.show();                      //mygtukas parodomas
        return app.exec();                        //gražinamas programos valdymas
    }
```

Qt objektai išskiria signalus tam kad nustatytu jog įvyko vartotojo veiksmas arba būsenos pasikeitimas. Pavyzdžiui, `QPushButton` išskiria `clicked()` signalą kai vartotojas paspaudžia mygtuką. Signalas gali būti sujungtas su funkcija, kuri automatiškai aktyvuojama aptikus signalą. Mūsų pavyzdyje mes sujungiame mygtumo `clicked()` signalą su `QApplication` objekto `quit()` funkcija. `SIGNAL()` ir `SLOT()` makro komandos yra sintaksės dalis.

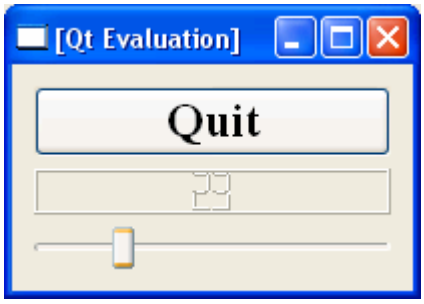
Dabar padarysime programą. Jūs sukūrėte direktoriją, pavadinimu `quit` kurioje yra `quit.cpp`. Įvykdykite `qmake` komandą `quit` direktorijoje, tam kad sugeneruoti projekto failą. Po to įvykdykite dar kartą makefile sugeneravimui:

```
qmake -project
qmake quit.pro
```

Dabar padarykite programą ir aktyvuokime ją. Jei paspausite Quit, programa baigs darbą.

Kitas pavyzdys parodo kaip naudoti signalus ir funkcijas dviejų objektų sinkronizavimui. Programa keičiant slankiklio padėtį keičia jo skaitinę išraišką.

Blokų programa



Programa susideda iš trijų objektų: `QLCDNumber`, `QSlider`, ir `QPushButton`. `QVBox` yra programos pagrindinis objektas. `QLCDNumber`, `QSlider`, ir `QPushButton` aktyvuojami `QVBox viduje`. Jie yra `QVBox vaikai`.

```
 1   #include <QApplication>
 2   #include <QFont>
 3   #include <QLCDNumber>
 4   #include <QPushButton>     //Pridedami klasių apibrėžimai
 5   #include <QSlider>
 6   #include <QVBoxLayout>
 7   #include <QWidget>

 8    class MyWidget : public QWidget           //Sukuriami objektai:
 9    {
 10   public:
 11       MyWidget(QWidget *parent = 0);
 12   };
 13   MyWidget::MyWidget(QWidget *parent)
 14       : QWidget(parent)
 15   {
 16       QPushButton *quit = new QPushButton(tr("Quit"));
 17                                                 //Sukuriamas mygtukas
 18       quit->setFont(QFont("Times", 18, QFont::Bold));
 19
  20     QLCDNumber *lcd = new QLCDNumber(2);
  21                          //Sukuriamas LCD ekranas skaičiams parodyti
  22      lcd->setSegmentStyle(QLCDNumber::Filled);
  23      QSlider *slider = new QSlider(Qt::Horizontal);
  24                                                  //Sukuriamas slankiklis
  25      slider->setRange(0, 99);
  26      slider->setValue(0);
  27
  28      connect(quit, SIGNAL(clicked()), qApp, SLOT(quit()));
  29                                      //nurodomos vartotojo veiksmų pasekės.
  30      connect(slider, SIGNAL(valueChanged(int)),
  31                                      //Paspaudus Quit programa baigs darbą
  32      lcd, SLOT(display(int)));
  33                           //Pakeitus slankiklio poziciją, keisis skaičiai
  34      QVBoxLayout *layout = new QVBoxLayout;
  35     //naudojant QVBoxLayout nustatome, jog visi objektai būtų lango
viduje
  36      layout->addWidget(quit);
  37             // ir keičiant lango dydį, objektų dydis keičiasi kartu
  38      layout->addWidget(lcd);
  39      layout->addWidget(slider);
  40      setLayout(layout);
  41   }
  42   int main(int argc, char *argv[])
  43                    // Pagrindinės programos pradžia
  44   {
  45       QApplication app(argc, argv);
```

```
46      MyWidget widget;
47      widget.show();                          // Objektas parodomas
48      return app.exec();                      // Gražinamas valdymas
49  }
```

Nuo 8 iki 12 eilutės vyksta `pagrindinio objekto` generavimas. Jei kompiliuojant šioje vietoje kyla klaida, reiškia kad jūs naudojate senesnę nei 3.2 versiją.

Nuo 13 iki 26 sukuriami `QSlider, QLCDNumber ir QPushButton objektai.` Mes nenustatėme nei vieno objekto pozicijos ar dydžio, tačiau objektai gražiai telpa pagrindinio lango viduje. Taip yra dėl to, kad `QVBoxLayout` automatiškai priskiria reikšmingą poziciją ir parenka dydi priklausomai nuo situacijos. Qt turi daug klasių panašių į `QVBoxLayout` išlaisvinančių mus nuo sunkaus darbo nustatant objektų pozicijas ir dydžius.

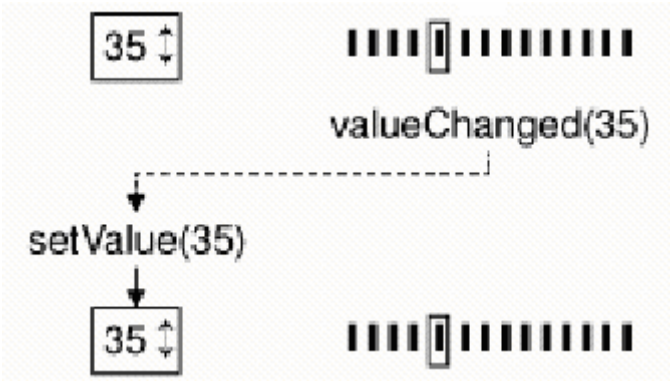
28 ir 29 Stebi mygtuko „Quit“ paspaudimą. Jei tai įvyksta organizuojamas programos baigtis.

Nuo 30 iki 32 eilutės nustato LCD laukelio reikšmę pagal skaitiklį. `connect()` kvietimas užtikrina, jog abu objektai yra sinchronizuoti ir rodo tą pačią reikšmę. Kai keičiasi slankiklio reikšmė, yra išskiriamas `valueChanged(int)` signalas ir aktyvuojama `display(int)` funkcija.

Nuo 33 iki 41 eilutės nustato jog visi objektai yra lango viduje. Ir nurodo jog keičiant lango dydį, keičiasi pačių objektų esančių viduje dydis.

Nuo 42 iki 49 pagrindinės programos tekstas.

Pakeitus vieną reikšmę, pasikeičia kita



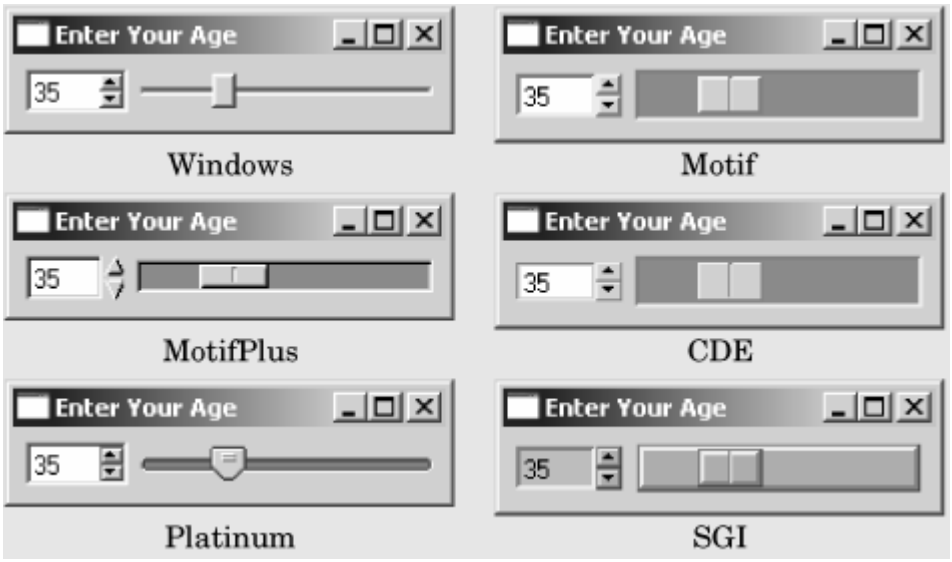
Nurodymų dokumentacijos naudojimas

Qt nurodymų dokumentacija yra išskirtinis įrankis bet kuriam Qt vystytojui, nes jis apima visas Qt klases ir funkcijas. Qt 3.2 turi virš 400 viešų klasių ir daugiau kaip 6000 funkcijų.)

Objektų stiliai

Nuotraukas, kurias matėme ankščiau buvo Windows XP stiliaus, tačiau Qt veikia ir kitose platformose.

Stiliai, kurie galimi visur



Qt programos vartotojai gali nepaisyti tradicinių stilių naudodami `-style` komandinės eilutės ypatybę. Pavyzdžiui paleisti amžiaus programą Platinum stiliumi Unix OS paprasčiausiai rašome

```
./age -style=Platinum
```

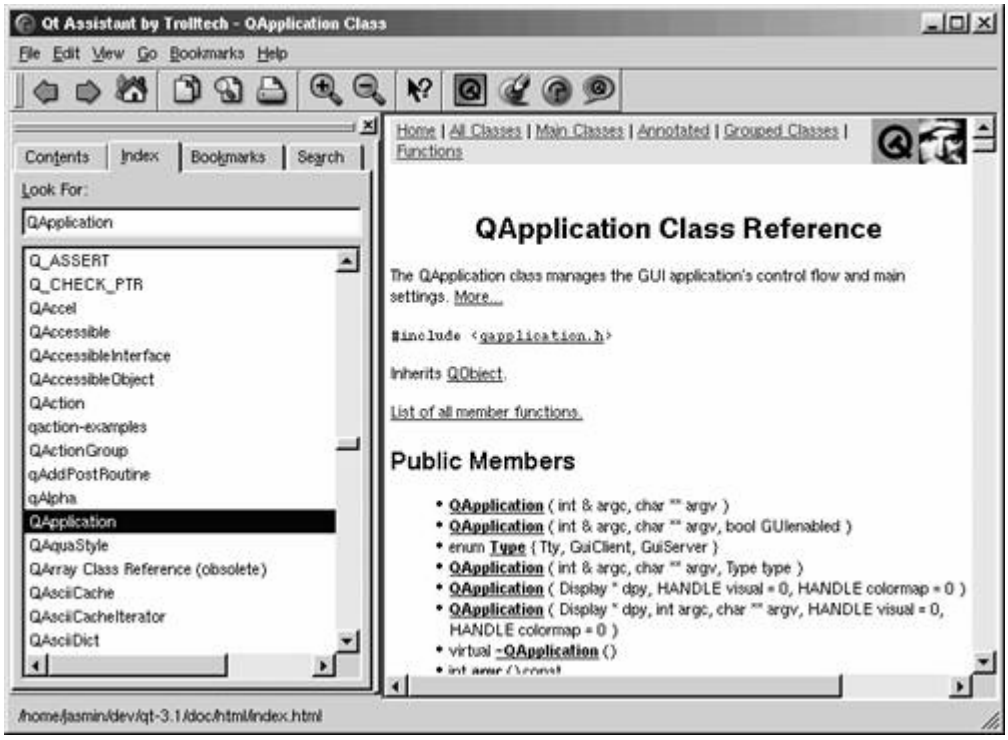
komandinėje eilutėje.

Konkrečių platformų stiliai

Ne taip kaip kiti stiliai, Windows XP ir Mac yra galimi tik gimtosiose platformose.

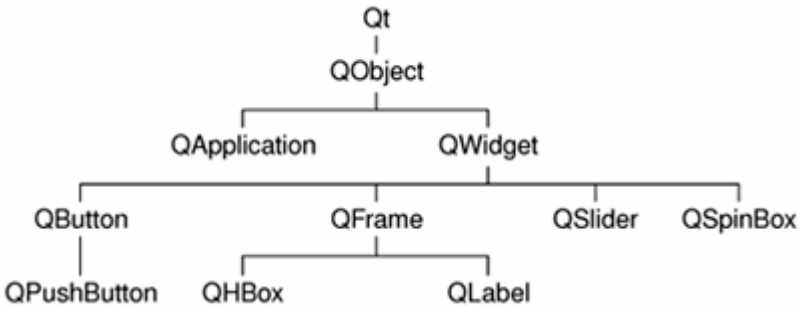
Dokumentacija pasiekiama HTML formatu Qt's `doc\html` direktorijoje ir gali būti peržiūrėta naudojant bet kurią naršyklę. Taip pat galime naudoti Qt asistentą.

Qt dokumentacija Qt Asistente



.

Qt klasių hierarchijos medis



Dokumentacija yra pasiekiama internete adresu: http://doc.trolltech.com/.