TIE-02400 harjoitustyö
Kevät 2015

Dokumentin historia		
27.2.2015	antti	työohje julkaistu

1. Yleistä

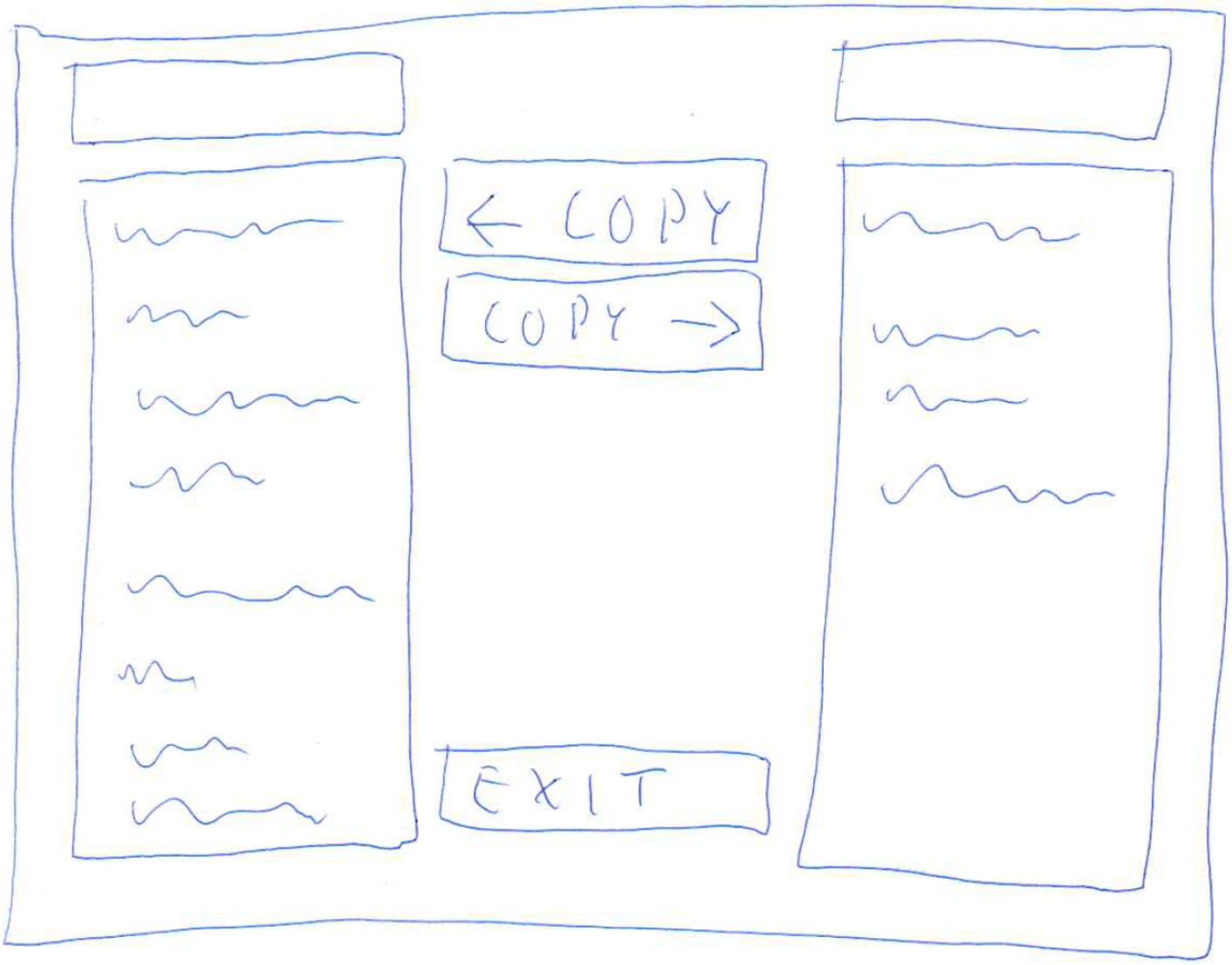
Tällä sivulla kuvataan kurssin TIE-02400 Ohjelmoinnin tekniikat kevään 2015 toteutuskerran harjoitustyö. Harjoitustyössä toteutetaan yksinkertainen etätiedostonhallintaohjelma.

Toteutettava ohjelma

Tarkoituksena on toteuttaa pieni etätiedostonhallintaohjelma, jolla voi tarkastella muiden tietokoneiden tiedostojärjestelmiä ja kopioida niiden sisältöä järjestelmältä toiselle. Ohjelmaa käynnistettäessä muodostetaan yhteys halutulle palvelimelle. Tämän jälkeen päästään ohjelman pääikkunaan, jonka kaksi näkymää näyttävät paikallisen koneen ja palvelimen tiedostojärjestelmät. Kummastakin näkyy kerralla yhden hakemiston sisältö, ja hakemistohierarkiassa voi liikkua ylös- ja alaspäin tarpeen mukaan. Erikoistapauksena yhteys etäjärjestelmään voidaan jättää muodostamatta, ja esittää paikallinen tiedostojärjestelmä kummassakin näkymässä.

Haluttujen hakemistojen ollessa auki näkymissä voidaan kummasta tahansa valita *hakemistoalkioita* (directory entry) ja kopioida ne toiseen. Käsiteltävistä alkioista tutuimpia ovat tavalliset tiedostot ja hakemistot, joita kutsuttakoon *perusalkioiksi*. Muita *erikoisalkioita* ovat esimerkiksi muihin alkioihin osoittavat symboliset linkit.

Ohjelmaan on jo toteutettu valmiiksi etäyhteyden muodostaminen ja etäalkioiden käsittely. Itse toteutettaviksi jäävät muut osat kuten käyttöliittymä, paikallisten alkioiden käsittely ja varsinaiset operaatiot. Alla on hahmotelma siitä, miltä valmiin ohjelman käyttöliittymä voisi näyttää.



Käyttötapaus

Seuraava toimintojen sarja havainnollistaa ohjelman käyttöä ja toimintaa. Kyseessä on esimerkki, eikä valmiin toteutuksen tarvitse seurata sitä orjallisesti.

- Käyttäjä käynnistää ohjelman ja saa eteensä valmiiksi toteutetun kirjautumisdialogin.
- Käyttäjä syöttää dialogiin haluamansa etäpalvelimen tiedot ja käskee muodostaa yhteyden. Uusi dialogi kysyy häneltä salasanaa ja hän syöttää sen. Valmis toteutus muodostaa etäyhteyden ja luovuttaa sitten kontrollin ja rakentamansa yhteysolion itse toteutettavalle ohjelman osalle.
- Itse toteutettava ohjelman pääikkuna aukeaa näkyviin. Ikkunassa on kaksi näkymää, joista toinen näyttää paikallisen hakemiston sisällön ja toinen etäpalvelimella olevan hakemiston sisällön. Oma toteutus pääsee käsiksi etäpalvelimen hakemistoalkioihin yhteysoliolta saatavien etäalkio-olioiden kautta. Lisäksi ikkunassa on joukko painikkeita ja/tai valikoita eri toiminnoille.
- Käyttäjä liikkuu molemmissa hakemistonäkymissä haluamiinsa hakemistoihin.
- Käyttäjä valitsee etäpalvelimen hakemistosta tiedoston ja käynnistää sen kopioinnin paikallisen koneen hakemistoon. Käyttöliittymästä näkee, että kopiointi on meneillään, kunnes tiedosto on kokonaan kopioitu.
- Käyttäjä palaa paikallisessa hakemistohierarkiassa vähän matkaa ylöspäin.
- Käyttäjä valitsee useita tiedostoja kerralla ja käynnistää niiden kopioinnin etäpalvelimelle. Kesken kopioinnin yhteysolion kautta saatu etäalkio-olio tuottaa poikkeuksen nimikonfliktista. Tämän seurauksena oma toteutus kertoo käyttäjälle, mitä tapahtui ja että kopiointi on keskeytetty. Loppuja tiedostoja ei kopioida, mutta jo kopioidut tiedostot jätetään paikoilleen.
- Käyttäjä katsoo etähakemistonäkymästä, mitkä kopioitavista tiedostoista jäivät puuttumaan. Hän valitsee ne paikallisesta näkymästä ja kopioi ne.
- Käyttäjä sulkee ohjelman.

Oppimistavoitteet

Työssä on tarkoitus harjoitella seuraavia asioita:

- Olemassa olevan ohjelmakoodin toimintaan tutustuminen ja sen käyttö tarjottujen rajapintojen avulla (sopimussuunnittelun hyödyntäminen)
- Toiminnallisuuden lisääminen sovellukseen siihen määriteltyjen rajapintojen mukaan (sopimussuunnittelun noudattaminen)
- Graafisen käyttöliittymän toteuttaminen osaksi ohjelmaa
- Poikkeusten käsittely ja virhetilanteiden hallinta
- Oman koodin testaus yksikkötasolla

2. Vaatimukset

Harjoitustyöhön on toteutettava vähintään tietyt perusominaisuudet, jotta sen suoritus voidaan hyväksyä. Lisäksi tarjolla on joukko vapaaehtoisia lisäominaisuuksia, joita toteuttamalla voi korottaa työn arvosanaa.

Pakolliset ominaisuudet

Harjoitustyössä on toteutettava ainakin alla olevat ominaisuudet. Niiden mukaisella minimitoteutuksella työstä voi ansaita **korkeintaan arvosanan 4** (huom, oli aiemmin 3, mutta työmäärää helpotettu). Korkeampi arvosana edellyttää myös joidenkin lisäosien toteutusta.

Ohjelman tarvitsee osata käsitellä ainoastaan tiedostojärjestelmän perusalkioita, eli tiedostoja ja hakemistoja. Samoin ohjelman on tarpeen toimia vain Linux-järjestelmien kanssa (eli POSIX-tiedostojärjestelmästä poikkeavaan käyttäytymiseen ei tarvitse varautua).

- Ohjelmaan on toteutettu pääkäyttöliittymä, jossa on näkymät hakemistoihin kahdessa tiedostojärjestelmässä.
- Hakemistoalkioita voi selata hakemistonäkymissä ja hakemistojen välillä liikkua.
- Tiedostoja voi kopioida hakemistonäkymien välillä. Niitä voi valita kopioitavaksi kerralla yhden tai useamman. Hakemistoja ei tarvitse voida kopioida. Kopioinnin aikana on oltava nähtävissä, että operaatio on meneillään, mutta käyttöliittymän ei tarvitse reagoida syötteisiin.
- Erikoisalkioiden läsnäolo tarkasteltavissa hakemistoissa ei saa sekoittaa ohjelmaa, mutta niitä ei tarvitse käsitellä mitenkään. Ne voidaan ohittaa operaatioissa eikä niitä tarvitse näyttää hakemistonäkymissä.
- Seuraavista virhetilanteista toivutaan hallitusti: etäyhteyden katkeaminen, luku- tai kirjoitusoikeuksien puuttuminen, nimikonfliktit (esim. kopioidaan tiedostoa hakemistoon, joka jo sisältää samannimisen tiedoston). Virhetilanteessa meneillään oleva operaatio voidaan peruuttaa tai keskeyttää, mutta ohjelman on päädyttävä järkevään tilaan ja käyttäjälle annettava ilmoitus virheestä.
- Luokalle `Copyer` on kirjoitettu yksikkötestit sen pakollisille ominaisuuksille. `Copyer`iin liittyvien lisäominaisuuksien testausta ei vaadita. Testauksen helpottamiseksi käytettävissä on valmiiksi toteutetut luokat virtuaalisille hakemistoalkioille, joiden avulla testejä voi suorittaa käsittelemättä oikeaa tiedostojärjestelmää.

Lisäominaisuudet

Alla olevia lisäominaisuuksia ei tarvitse toteuttaa, mutta hyvästä lisäosan toteutuksesta voi ansaita korotuksen arvosanaan. Korotus on 0,5 arvosanaa kustakin lisäosasta. Korotuksen saaminen edellyttää, että lisäosa on **dokumentoitu** palautuksessa. Töiden tarkastajat eivät etsi lisäosia palautetusta koodista.

Useita lisäominaisuuksia toteutettaessa niiden on pääsääntöisesti toimittava myös yhdessä. Esimerkiksi jos toteutetaan tuki alkioiden tietojen tarkastelulle ja symbolisten linkkien käsittelylle, on myös symbolisten linkkien tietoja voitava tarkastella.

- Hakemistojen rekursiivinen käsittely. Hakemistoja sisältöineen voidaan kopioida ja käsitellä muilla tuetuilla operaatioilla. (1)
- Leikepöytä. Tuettujen alkioiden kopiointi-, leikkaus- ja liittämisoperaatiot, joiden avulla tiedostoja voi siirtää yhden järjestelmänäkymän sisällä kierrättämättä niitä toisen näkymän kautta.
- Tuettujen alkioiden poisto- ja uudelleennimeämisoperaatiot. Hakemistoja tarvitsee voida poistaa vain, jos niiden rekursiivista käsittelyä tuetaan. Uudelleennimeäminen on toteutettava järkevästi niin, ettei alkiota kopioida.
- Tuettujen alkioiden tietojen (esim. koko, päiväykset, oikeudet) näyttäminen joko hakemistonäkymissä tai erillisessä dialogissa. Hakemistojen koko sisältöineen tarvitsee näyttää vain, jos niiden rekursiivista käsittelyä tuetaan.
- Tuettujen alkioiden oikeuksien hallinta. Alkioille voidaan asettaa luku-, kirjoitus- ja suoritusoikeudet käyttäjälle, ryhmälle ja muille.
- Symbolisten linkkien ja niitä sisältävien hakemistorakenteiden käsittely tuetuilla operaatioilla helpoissa

tapauksissa. Hakemistonäkymissä on voitava kulkea linkkien osoittamien hakemistojen sisään. Operaatioissa linkkejä voidaan käsitellä joko itsenäisinä alkioina tai ikään kuin ne olisivat linkittämiään alkioita. Tämän lisäosan toteutuksen tarvitsee osata käsitellä ainoastaan symbolisia linkkejä, jotka osoittavat kokonaan erillisiin hakemistorakenteisiin (eli muodostavat puumaisen rakenteen). Hankalampia linkkien muodostamia silmukoita tai verkkoja ei tarvitse osata käsitellä, eikä ohjelman tarvitse toimia järkevästi tilanteissa, joissa niitä tulee vastaan. (1)
- Tuettuja alkioita voi kopioida raahaamalla niitä hiirellä hakemistonäkymästä toiseen.
- Järkevä(hkö)n edistymispalkin näyttäminen tuetuille operaatioille, joko päänäkymässä tai erillisessä dialogissa. Tämä edellyttää operaation koon arviointia jo sen suorituksen alkuvaiheessa. Ajankäytön arvioita ei tarvitse esittää. (2)
- Tuettujen alkioiden kopioinnin (ja siirtämisen, jos tuettu) toteuttaminen asynkronisesti siten, että järjestelmä ei pysähdy odottamaan operaation tulosta vaan muita operaatioita voi suorittaa samanaikaisesti. (2)
- Näkymät, jotka esittävät tiedostojärjestelmien hakemistohierarkiat puurakenteina ja joista voidaan vaihtaa auki olevia hakemistoja. Voidaan toteuttaa erillisinä näkyminä hakemistonäkymien rinnalle tai vaihtoehtoisena moodina samoihin näkymiin.
- Yhteyden muodostaminen useampaan kuin yhteen etäjärjestelmään kerralla. Käyttöliittymään voi esimerkiksi toteuttaa kummankin hakemistonäkymän yläpuolelle alasvetovalikon, josta voi vaihtaa kulloinkin näytettävää järjestelmää.
- Virhetilanteiden monipuolisempi hallinta. Esimerkiksi etäyhteyden katketessa se yritetään muodostaa uudelleen ja nimikonfliktissa tarjotaan dialogissa vaihtoehdot ylikirjoittaa vanha tiedosto, nimetä kopioitava tiedosto uudelleen tai peruuttaa operaatio.
- Ohjelman pakollisille osille on muodostettu kattavat rajapintamäärittelyt (parametrit ja paluuarvot, esi- ja jälkiehdot, poikkeukset). Lisäominaisuuksiin liittyville rajapinnoille määrittelyjä ei ole välttämätöntä tehdä.
- Ohjelmaa testataan kattavammin kuin on vaadittu. Testit on dokumentoitu hyvin.
- Oma lisäominaisuus. Ohjelmaan on toteutettu jokin itse keksitty tämän listan ulkopuolinen lisäominaisuus. Omat lisäominaisuudet on hyväksytettävä etukäteen kurssihenkilökunnalla, jotta niistä voisi saada pisteitä.

(1) Nämä lisäosat yhdessä (hakemistojen rekursiivinen käsittely ja symboliset linkit) voidaan lukea myös Tietorakenteiden ja algoritmien kurssin harjoitustyöksi, jos ne täyttävät tietyt lisäkriteerit. Katso yksityiskohdat täältä.

(2) Nämä lisäosat yhdessä (asynkroninen kopiointi ja edistymispalkki) voidaan lukea myös Rinnakkaisuuden kurssin harjoitustyöksi, jos ne täyttävät tietyt lisäkriteerit. Katso yksityiskohdat täältä.

3. Toteutus

Työympäristö haetaan kurssisvn:stä. Valmiit toteutuksen osat tulevat sinne saataville.

Ympäristö

Työtä varten toteutettu valmis ohjelmarunko on saatavilla kurssisvn:stä ryhmän omasta tietovarastosta. Heti rungon hakemisen jälkeen kannattaa myös asettaa seuraava svn-ignore, joka pitää huolen siitä, että QtCreatorin käyttäjäkohtaiset asetukset eivät vahingossa talletu versiohallintaan:

```
cd {projektin trunk-hakemisto}
svn propset svn:ignore '*.pro.user' -R .
```

Näiden toimien tuloksena pitäisi alla oleva hakemistorakenne.

```
ryhmän trunk
│
├── SSH-Client.pro
│
├── julkinen
│   └── harjoitustyo
│       └── Staff
│           ├── ConnectionInterface.h
│           ├── CopyerInterface.h
│           ├── DirectoryEntryInterface.h
│           ├── DirectoryInterface.h
│           ├── FileInterface.h
│           ├── Initialization.h
│           ├── Staff.pro
│           ├── ...
├── Student
│   ├── Copyer.cpp
```

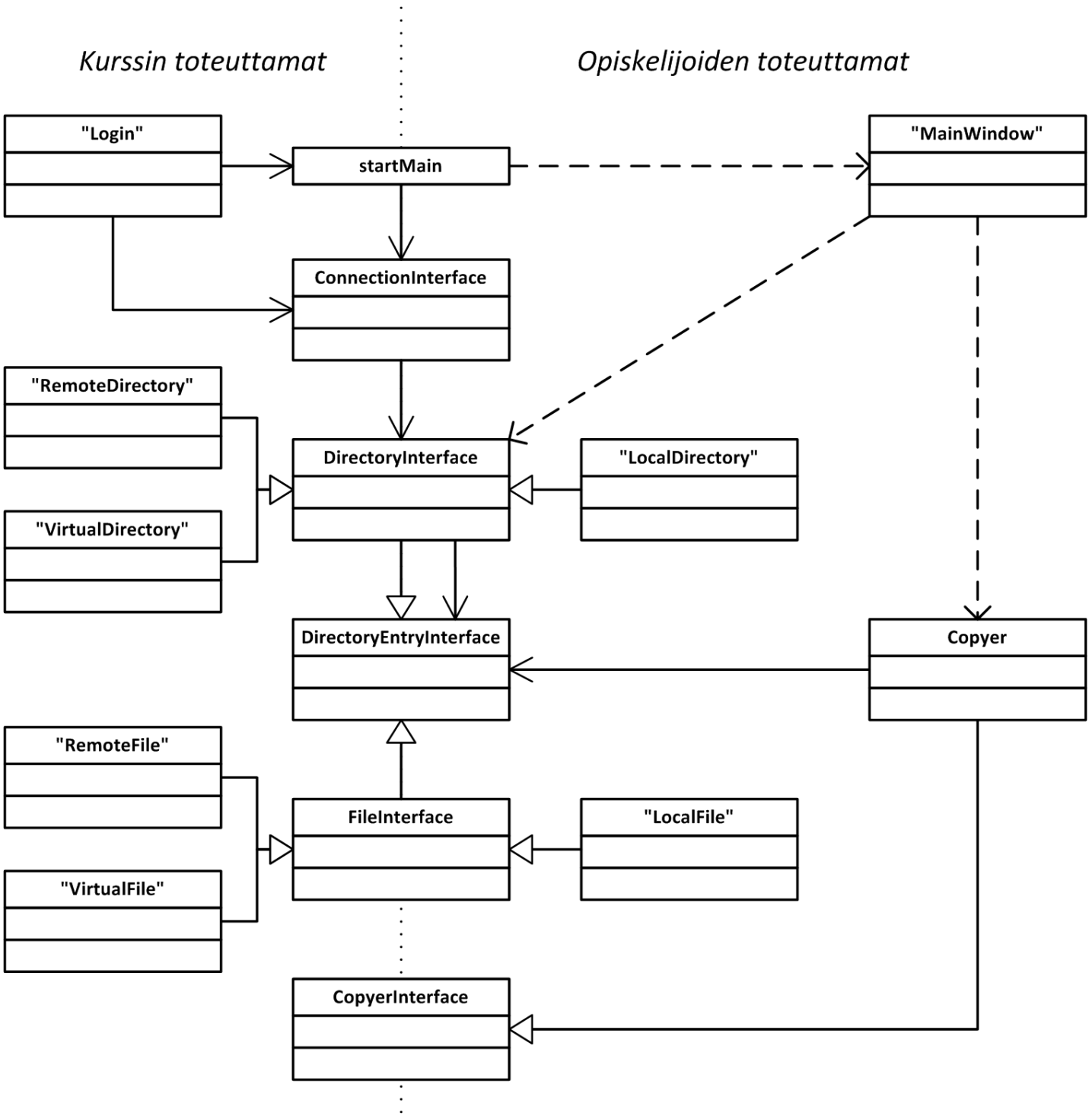
```
│   ├── Copyer.h
│   ├── Initialization.cpp
│   └── Student.pro
└── Test
    ├── CopyerTest.cpp
    └── Test.pro
```

- `SSH-Client.pro` on projektitiedosto, josta koko projektin saa avattua QtCreatoriin.
- `julkinen`-hakemiston alla ovat valmiit toteutuksen osat. Tämän hakemiston sisältöä ei saa muokata.
- `Student`-hakemiston alle sijoitetaan kaikki omat toteutuksen osat. Joillekin näistä annetaan pieni toteutusrunko, jotta projekti menee heti kääntäjästä läpi.
- `Test`-hakemiston alle kirjoitetaan `Copyer`-luokalle vaaditut yksikkötestit.

Luokkakaavio

Alla olevassa kuvassa on esitetty ohjelman oleellisimmat luokat. Lainausmerkeissä olevien luokkien oikeilla nimillä ei ole työn kannalta merkitystä.

HUOM! Opiskelijoiden toteuttamien luokkien osalta kaavio on *täysin* kuvitteellinen, luokkia voi olla enemmän (ja todennäköisesti onkin), ja niiden väliset suhteet voivat olla aivan erilaiset.



Valmiina annetut luokat ja rajapinnat

Ohjelmaan on toteutettu valmiiksi mm. alla olevat osat.

- Valmis toteutus huolehtii etäyhteyden muodostamisesta ja pyytää sitten itse toteutettavaa koodia rakentamaan ohjelman päänäkymän kutsumalla `Initialization`-tiedostoissa määriteltyä `startMain()`-funktiota.
- `startMain()`-funktiolle välitetään parametrina lista `ConnectionInterface`n toteuttavia olioita, joista jokainen edustaa yhtä luotua etäyhteyttä. Näiden kautta pääsee käsiksi etäyhteyden päässä oleviin hakemistoihin ja tiedostoihin.
- `DirectoryEntryInterface`-, `FileInterface`- ja `DirectoryInterface`-rajapintaluokat määrittelevät rajapinnat hakemistoalkioita (tiedostoja ja hakemistoja) kuvaaville olioille. Ohjelmaan voidaan toteuttaa tuki myös muille erikoisalkioille (esim. symboliset linkit), joten periytymishierarkian ei voi olettaa olevan täydellinen.
- Ylläoleville rajapinnoille on tehty valmiina toteutukset etäyhteyden läpi käsiteltäville alkioille. Samoin on tehty testauksessa hyödylliset virtuaaliset alkiototeutukset, jotka eivät kytkeydy mihinkään tiedostojärjestelmään.
- `CopyerInterface` määrittelee rajapinnan kopiointioperaatiot suorittavalle luokalle. Rajapintaluokkaa ei

käytetä ohjelman sisältä, mutta se kiinnittää kopioijan rajapinnan testausta varten.

Itse ohjelmaan on toteutettava ainakin seuraavat osat:

- `startMain`-funktio, jota kutsutaan, kun etäyhteys on muodostettu. Siellä pitäisi saattaa ohjelma valmiiksi käyttöä varten ja muun muassa luoda ohjelman päänämymä.
- Päänäkymä toteutetaan omassa luokassaan. Sieltä on jotenkin päästävä käsiksi paikallisiin hakemistoalkioihin ja kopiointioperaatioihin, mutta kytköksen ei tarvitse olla luokkakaavion tapaan suora. Pääikkunan pohjan voi luoda tekemällä `Student`-aliprojektille uuden tiedoston tyyppiä `Qt`->`Qt Designer Form Class` ja valitsemalla templateksi `Main Window`.
- Perusalkioiden rajapinnat on toteutettava paikallisesti käsiteltäville alkioille.
- `Copyer`, joka suorittaa alkioden kopiointioperaatiot. Tälle luokalle on myös kirjoitettava yksikkötestit (aliprojektin `Test` luokkaan `CopyerTest`), minkä vuoksi sen rajapinta on kiinnitetty etukäteen periyttämällä se `CopyerInterface`-luokasta.

Luonnollisesti toteutukseen voi olla syytä sisällyttää muitakin luokkia.

Annettujen luokkien ja rajapintojen dokumentaatio

Valmiina annettujen luokkien ja rajapintojen Doxygen-dokumentaatio löytyy kurssin kotisivuilta.

Testaus

Harjoitustyön hyväksymiseksi vaaditaan, että `Copyer`-luokalle on kirjoitettu kattavat yksikkötestit. Laaditut testit palautetaan arvosteltavaksi varsinaisen ohjelmakoodin mukana. Luonnollisesti ohjelman muutkin osat ja ohjelma kokonaisuutena on syytä testata niiden toimivuuden varmistamiseksi, mutta niiden testausta ei erikseen arvostella. Vaadittua laajempi testaus on kuitenkin mahdollista lukea lisäominaisuudeksi ja siten korottaa arvosanaa.

Palautettavia testejä on tarkoitus voida ajaa myös muiden tekemiä toteutuksia vastaan. Testit olisi siis mahdollisuuksien mukaan hyvä suunnitella hyväksymään mikä tahansa `CopyerInterface`-rajapinnan määrittelyn mukaan oikeellinen toiminnallisuus, ei vain oman toteutuksen mukaista toiminnallisuutta. Kaikkea mahdollista ei kuitenkaan tarvitse yrittää ottaa huomioon, eikä luokan toimivuutta ole tarpeen testata muiden kuin pakollisten ominaisuuksien osalta. Esimerkiksi testattaessa hakemiston antamista addSource-metodille voitaisiin lopputuloksena hyväksyä sekä poikkeus että onnistunut suoritus, mutta onnistuneenkaan suorituksen jälkeen ei ole tarpeen kokeilla kopioinnin käynnistämistä.

`Copyer`in testausta ei ole tarkoitus tehdä oikeilla paikallisilla tai etähakemistoalkioilla. Niiden sijaan käytetään kurssin puolesta tarjottuja virtuaalisia hakemistoalkioita. Tällä tavoin testeissä ei tarvitse yrittää etsiä tai luoda sopivia oikeita tiedostoja, eivätkä ne pääse sekoittamaan oikeaa tiedostojärjestelmää.

Vaaditut `Copyer`-luokan yksikkötestit kirjoitetaan osaksi työympäristöstä löytyvää `Test`-projektia luokkaan `CopyerTest`. Mahdollisia muita automoituja testejä varten on luotava omat aliprojektinsa. Näihin on tuotava mukaan tarvittavat tiedostot muista aliprojekteista; katso mallia `Test.pro`-tiedostosta.

Vinkkejä

- Qt:n käyttöliittymän listat (esim. `QListWidget`) tukevat suoraan useiden alkioiden valitsemista, kunhan valintamoodin asettaa sopivaksi.
- Tiedostojen käsittelyä toteutettaessa kannattaa katsoa, mitä palveluita Qt:n omat tiedostorajapinnat tarjoavat, ennen kuin lähtee tekemään asioita suoraan POSIX-rajapintojen kautta. QDir-oliota käytettäessä on syytä huomata, että muutokset tiedostojärjestelmässä päivittyvät näkyviin vasta kun sille ajaa `refresh`-kutsun.