Ohjelmoinnin opetus

ja

opettajien suhtautuminen opetusta kehittäviin välineisin

Sisällys

1 Johdanto

Tämä dokumentti sisältää Ohjelmoinnin perusopetuksen verkostohankkeen loppuraportista järjestelmien kuvaus- sekä tutkimus- ja tulososuudet. Varsinainen raportti kaikkine osuuksineen sekä hankkeen kuvauksineen julkaistaan seminaarin jälkeisessä lähitulevaisuudessa kaikille osallistujille muistitikuilla muun hankeeseen liittyvän materiaalin ohella.

Dokumentissa tarkastellaan ohjelmoinnin opetuksen nykytilaa ja ohjelmoinnin opettajien suhtautumista ja asenteita opetusjärjestelmiä kohtaan. Tiedot on kerätty haastattelemalla yhteensä 26 ohjelmoinnin ja algoritmien peruskurssien opettajaa suomalaisista korkeakouluista.

Mukana olevat järjestelmät ovat kaikki käytössä suomalaisissa korkeakouluissa ohjelmoinnin tai algoritmien opetuksen kursseilla. Useimmat järjestelmistä on lisäksi (ainakin osittain) kehitetty Suomessa.

Varsinainen lopullinen raportti sisältää käsillä olevan osuuden lisäksi tarkemman kuvauksen hankeesta, sen parissa toteutetuista piloteista ja katsauksen seminaareihin. Lisäksi siihen on koottu aikaisempia tutkimuksia järjestelmien käytöstä ja tehokkuudesta sekä pohdittu tarkemmin järjestelmien tehokkuuden arviointiin sopivia mittareita.

Tämän dokumentin rakenne on seuraava: luvussa 2 esitellään järjestelmät ja niiden tärkeimmät ominaisuudet lyhyesti. Tutkimus esitellään luvussa 3 ja viimeisessä luvussa käydään läpi sen tulokset.

2 Opetusjärjestelmät

Seuraavassa esitellään lyhyesti eräitä opetusjärjestelmiä. Valitut järjestelmät ovat kaikki käytössä ohjelmoinnin opetuksen kursseilla suomalaisissa korkeakouluissa. Lisäksi valtaosa järjestelmistä on kehitetty Suomessa.

Tarkasteltavat järjestelmät voidaan jakaa kolmeen kategoriaan:

Automaattinen arviointi: Automaattiset arviointijärjestelmät tarkistavat ja antavat automaattisesti palautteen (ja mahdollisesti arvosanan) opiskelijoiden palauttamista harjoitustehtävistä. Palautteenantokriteerit ovat yleensä muokattavissa sopimaan erityyppisille kursseille; lisäksi opettajat pystyvät halutessaan lisäämään henkilökohtaista palautetta automaattisesti muodostetun joukkoon. Järjestelmissä on usein lisänä erilaisia kurssinhallintaominaisuuksia, mukaan lukien ryhmien muodostaminen ja hallinta, arvosanojen siirto kirjausjärjestelmiin ja harjoitustehtävien jako- ja palautusjärjestelmä.

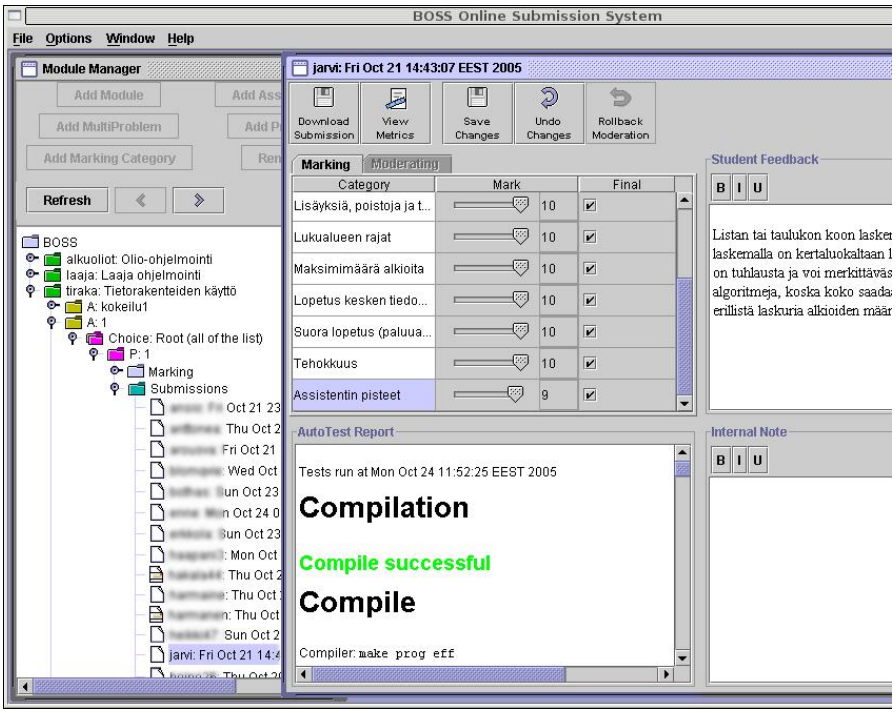
Visualisointi: Visualisointityökalut esittävät ohjelman tai algoritmin suorituksen erilaisten graafisten ja tekstimuotoisten elementtien avulla. Tarkoituksena on konkretisoida erilaisia ohjelmointiin liittyviä, vaikeasti hahmotettavia käsitteitä. Järjestelmästä riippuen opiskelijat pystyvät joko itse kirjoittamaan visualisoitavia ohjelmia tai tarkastelemaan opettajan valmistelemia visualisointeja. Yhteisenä piirteenä opiskelijat pystyvät suorittamaan visualisoinnit itsenäisesti, haluamassaan järjestyksessä ja itselleen sopivalla tahdilla.

Tehtävät: Verkossa suoritettavien ja palautettavien tehtävien avulla voidaan korvata osa perinteiseen opetukseen kuuluvista demonstraatioista tai laskuharjoituksista. Järjestelmät sisältävät joko valmiiksi määriteltyjä tai automaattisesti satunnaisin alkuarvoin generoituja harjoituksia, jotka opiskelijat ratkaisevat ja palauttavat. Järjestelmät myös yleensä antavat automaattisesti palautteen tehtävistä ja pitävät kirjaa tehdyistä tehtävistä ja saavutetuista

pisteistä; tämän mahdollistamiseksi ne sisältävät yleensä joitain kurssinhallintajärjestelmän piirteitä (tai vaihtoehtoisesti hyödyntävät ulkoista hallintajärjestelmää).

Huomattavaa on, että useat esitellyistä järjestelmistä eivät kuulu puhtaasti yhteen edellä mainituista kategorioista, vaan toteuttavat useampaan kategoriaan kuuluvia ominaisuuksia.

2.1 BOSS



Kuva 1 BOSS

BOSS on Warwickin yliopistossa kehitetty kurssien hallintatyökalu, joka mahdollistaa opiskelijoiden järjestelmän avulla palauttamien harjoitustöiden ja -tehtävien automaattisen tarkastamisen. Lisäksi järjestelmän avulla voidaan havainnoida plagiointia ja antaa palautetta palautetuista tehtävistä. Joyn ym. (2005) mukaan järjestelmän avainominaisuuksia ovat

Automaattinen arviointi: BOSS mahdollistaa automaattisen arvioinnin useilla eri tyyleillä ja strategioilla. Palautettuja ohjelmointitöitä voidaan arvioida kolmen eri mittarin avulla:

- *Oikeellisuus:* toimiiko ohjelma annettujen määrittelyiden mukaisesti?
- *Tyyli:* onko ohjelma muodostettu hyviä ohjelmointikäytäntöjä käyttäen?
- *Autenttisuus:* onko oletettavaa, että ohjelman palauttanut opiskelija on itse tuottanut ohjelman eikä plagioinut ratkaisua tai sen osia toisilta opiskelijoilta?

Palautuksen joustavuus: BOSS tukee joustavasti erityyppisten tehtävien (myös ei-ohjelmointitehtävien) palautusta. Edellisen mahdollistamiseksi tehtävien palautus ja materiaalien hallinta on erotettu itsenäiseksi kokonaisuudekseen tehtävien arvioinnista.

Pedagoginen perusta: Vaikka oppimispäämäärät ja kriteerit hyvin muodostetuille ohjelmille eroavatkin eri korkeakoulujen (ja jopa eri kurssien) välillä, voidaan tiettyjä yhtäläisyyksiä kuitenkin löytää. Tällaisiksi voidaan Joyn ym. (2005) mukaan laskea muun muassa kommenttien käyttö, ohjelmakoodin tyyli ja rakenne, dokumentaatio ja muodostetun ratkaisun algoritminen tehokkuus. BOSSin avulla voidaan määritellä kriteerit, joiden mukaan ratkaisuja arvioidaan.

Automaattinen tarkastaminen: BOSS määrittää ohjelman oikeellisuuden (eli annettujen määrittelyiden mukaisen toimivuuden) automaattisten testien avulla: se tarkastelee ohjelman toimivuutta vertaamalla ohjelman lopputulosta määriteltyyn tulostiedostoon. Koska BOSS tarjoaa suojatun ajoympäristön palautettujen ohjelmien ajoa varten, testit oikeellisuuden varmistamiseksi voidaan ajaa turvallisesti niin opettajien kuin opiskelijoidenkin järjestelmissä.

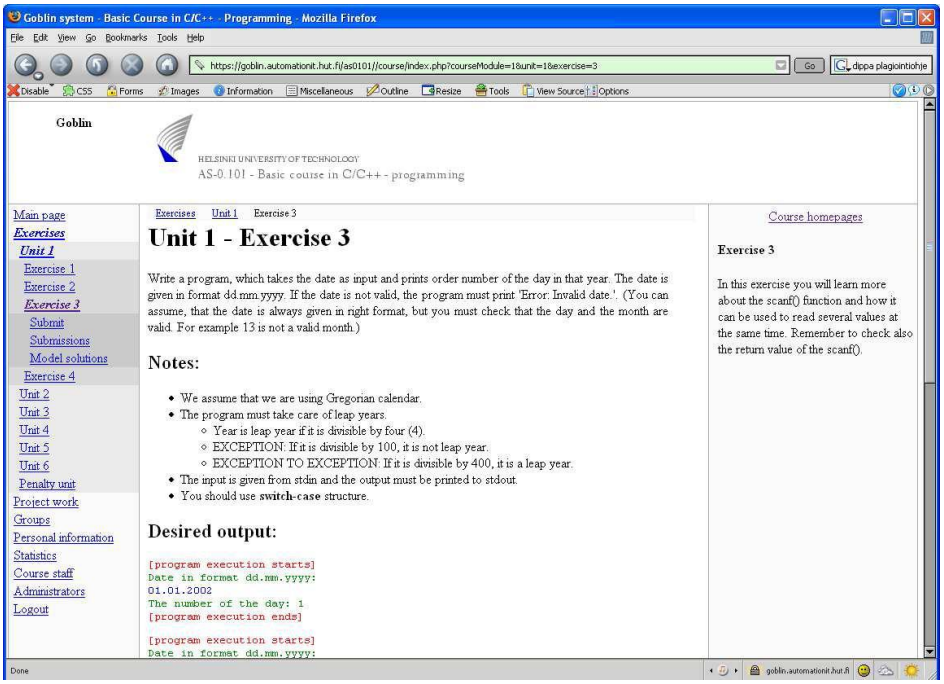
Arvioinnin asteikot: BOSS tarjoaa valmiina useita asteikkoja, joiden avulla ohjelmaa voidaan arvioida. Lisäksi uusien asteikkojen lisääminen on mutkatonta.

Palautus ja autentikointi: BOSS tarjoaa työkalut käyttäjien ja tiedostojen autentikoimiseen sekä tehtävien palautukseen.

Tuki ylläpidolle ja opettajille: BOSS mahdollistaa sen integroimisen korkeakoulun muihin järjestelmiin tarjoamalla liittymän datan tuomiseen ja viemiseen. Opettajille BOSS tarjoaa

mahdollisuuden paitsi automaattisesti merkitä palautteen harjoitustehtäviin, myös antaa halutessaan yksityiskohtaisempaa palautetta tehtävistä.

2.2 Goblin



Kuva 2 Goblin

Goblin on Teknillisessä korkeakoulussa kehitetty kurssinhallintajärjestelmä. Järjestelmä mahdollistaa muun muassa tehtävienannon, palautukset sekä ryhmien muodostuksen. Opettajat voivat järjestelmän avulla hoitaa arvostelun ja opiskelijatietojen selailun. Lisäksi järjestelmään voidaan integroida ulkopuolinen moduuli hoitamaan automaattista tarkastusta. Hiisilä (2005) esittelee järjestelmän avainominaisuuksiksi seuraavat:

Käyttäjien hallinta: Käyttäjien hallinta ja autentikoiminen sekä käyttäjäryhmien todentaminen.

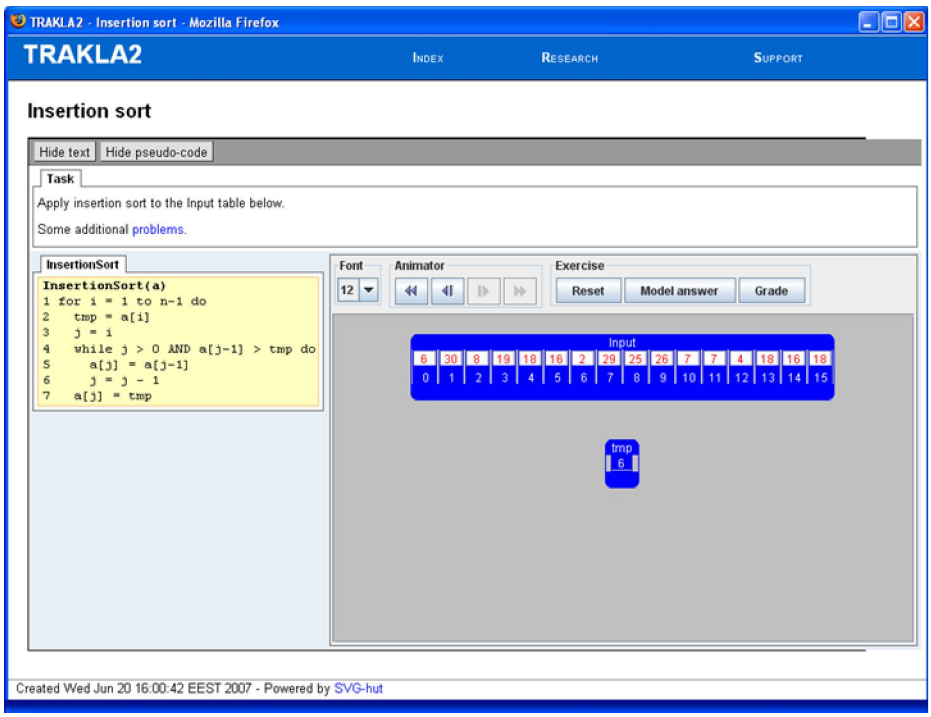
Kurssitietojen hallinta: Kurssin aikataulujen, materiaalien, tehtävien ja osallistujien hallinta.

Ryhmätöiden hallinta: Ryhmätöiden palauttamiseksi ja arvostelemiseksi järjestelmään on toteutettu ryhmien hallintaominaisuudet.

Tehtävien variointi: Harjoitustehtävien varioinnilla voidaan vähentää plagioinnin mahdollisuutta ratkaisussa. Variointiin kuuluu myös eri kieliversioiden tarjoaminen tehtävistä.

Hiisilän (2005) mukaan Teknillisessä korkeakoulussa on järjestelmän yhteydessä käytössä *EXCAPA (Experimental XML-based Program for Automatic Code Assessment)* -ohjelma, joka suorittaa ohjelmointitehtävien automaattisen tarkastamisen. EXCAPA tarkastaa tehtävän dynaamisesti, eli tarkistelee sen vastetta sille annettuun syötetiedostoon. Vertailemalla ohjelmien tulosteita voidaan päätellä toimiiko opiskelijan ohjelma määrittelyissä tarkoitetulla tavalla.

2.3 TRAKLA2



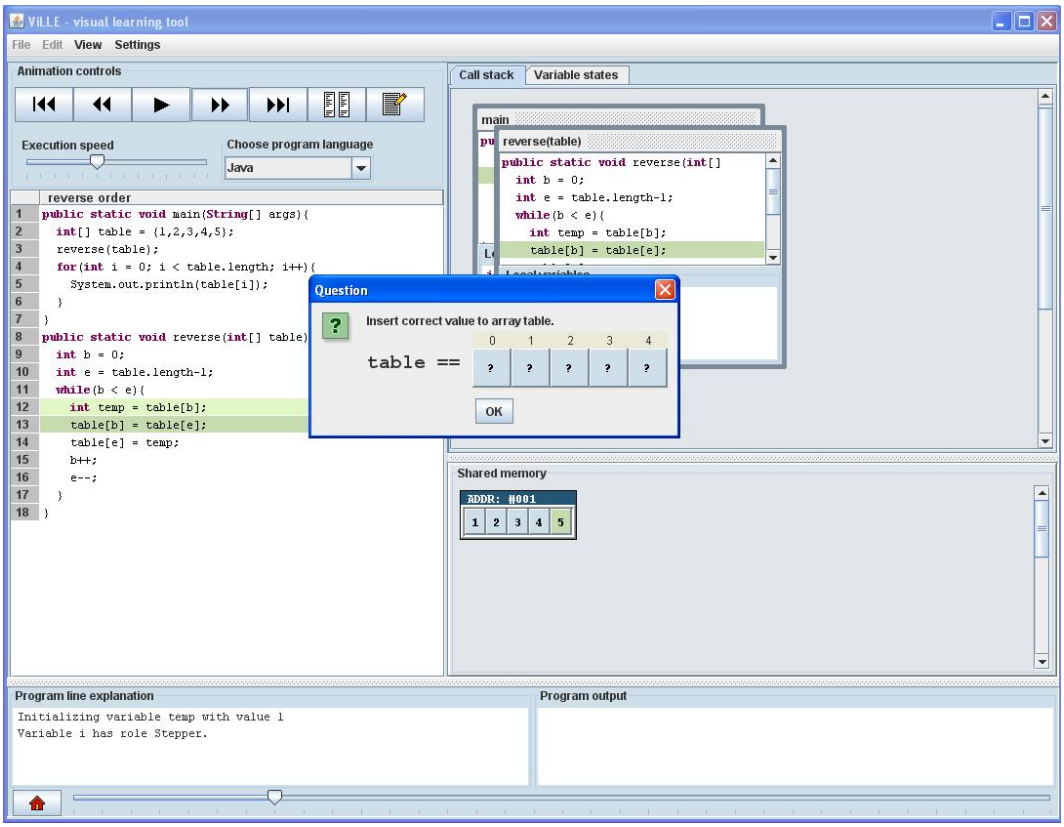
Kuva 3 TRAKLA2

TRAKLA2 on algoritmien visualisointijärjestelmä, jonka avulla käyttäjät voivat suorittaa erilaisia tietorakenteisiin ja algoritmeihin liittyviä harjoitustehtäviä ja palauttaa vastaukset niihin. Oppimisprosessissa käyttäjät jäljittelevät algoritmin toimintaa manipuloimalla

käsiteltävää tietorakennetta, esimerkiksi raahaamalla alkiot oikeille paikoilleen binääripuuhun. Tehtävät alustetaan satunnaisilla alkuarvoilla ja käyttäjät voivat halutessaan alustaa ja suorittaa tehtävän useita kertoja. Lisäksi käyttäjät voivat tarkastella tehtävän automaattista mallisuoritusta askel kerrallaan eteen- ja taaksepäin. Käyttäjät voivat järjestelmän avulla palauttaa tehtävien vastaukset; järjestelmä antaa automaattisesti välittömän palautteen ratkaisusta. Palvelin pitää kirjaa palautetuista tehtävistä ja niissä saavutetuista pisteistä. (Myller ym., 2007).

TRAKLA2 on nykyisellään käytössä tietorakenteiden ja algoritmien kursseilla ainakin Turun Yliopistossa ja Teknisessä korkeakoulussa. Molemmissa kouluissa järjestelmä on integroitu tiiviisti kurssin osaksi korvaten osittain kurssiin kuuluvat pakolliset harjoitustehtävät.

2.4 ViLLE



Kuva 4 ViLLE

ViLLE on Turun yliopistossa kehitetty visuaalinen työkalu luentokäyttöön ja itseopiskeluun. Rajalan ym. (2007) mukaan järjestelmän tavoitteena on havainnollistaa ohjelmointikielten perustoiminnallisuuksien samankaltaisuuksia, ohjelmointikielestä riippumatta. Järjestelmän avainominaisuuksia oppijan näkökulmasta ovat:

Suorituksen visualisointi: ViLLEn avulla voidaan visualisoida ohjelman suoritus askel kerrallaan. Visualisoitavan ohjelman tila esitetään eri ikkunoiden avulla: muuttujien tila, aliohjelmakutsut, ohjelmakoodi ja taulukot esitetään kukin omissa ikkunoissaan.

Kieliriippumattomuus: Järjestelmä tukee oletuksena Javan, Pythonin, C++:n, PHP:n ja JavaScriptin määrittelyjä; opettaja voi myös määritellä uusia kieliä järjestelmään. Käyttäjä voi vaihtaa suorituskieltä koska tahansa suorituksen aikana, minkä lisäksi suoritusta voidaan tarkastella kahdella eri kielellä samanaikaisesti rinnakkaisnäkymässä.

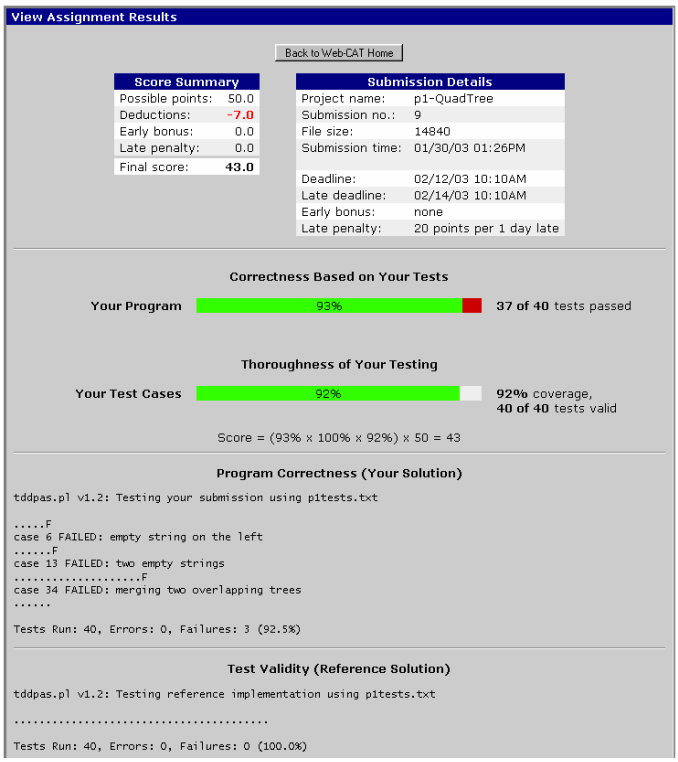
Vuorovaikutus: Käyttäjä voi ohjailla visualisoitavan ohjelman suoritusta joustavasti. Lisäksi esimerkkiohjelmien yhteyteen voidaan määritellä kysymyksiä, joihin käyttäjät vastaavat suorituksen aikana; *ViLLE tukee myös TRAKLA2-palvelimen käyttöä:*. Suoritettavan ohjelman koodia voi muokata kesken suorituksen, jolloin käyttäjä voi tarkastella muutosten vaikutusta visualisointiin.

Vastaavasti järjestelmän avainominaisuuksia opettajan näkökulmasta ovat

Muokattavuus: Esimerkkien, kysymysten ja uusien ohjelmointikielten lisääminen onnistuu sisäänrakennettujen editorien avulla.

Esimerkkien julkaisu: Opettaja voi julkaista esimerkkikokoelmansa itsenäisenä pakettina, joka voidaan ajaa verkosta tai muistitikulta. Julkaistu versio ei sisällä esimerkkien, syntaksien eikä kysymysten muokkausmahdollisuutta.

2.5 WebCat



Kuva 5 WebCat

WebCat on automaattinen arviointijärjestelmä ohjelmointitehtävien käsittelyyn. Edwardsin (2006) mukaan sitä käytetään tyypillisesti oppilaan apuna ohjelmien testauksessa ja muodostamaan konkreettista ja yksilöllistä palautetta ohjelmasta. Edwardsin mukaan WebCatin vahvuuksia ovat:

Turvallisuus: WebCat sisältää useita valinnaisia vaihtoehtoja käyttäjän ja palvelimen välisen yhteyden turvaamiseksi. Lisäksi ohjelmien virheet ja muut turvallisuusriskit tarkistetaan ennen niiden ajoa.

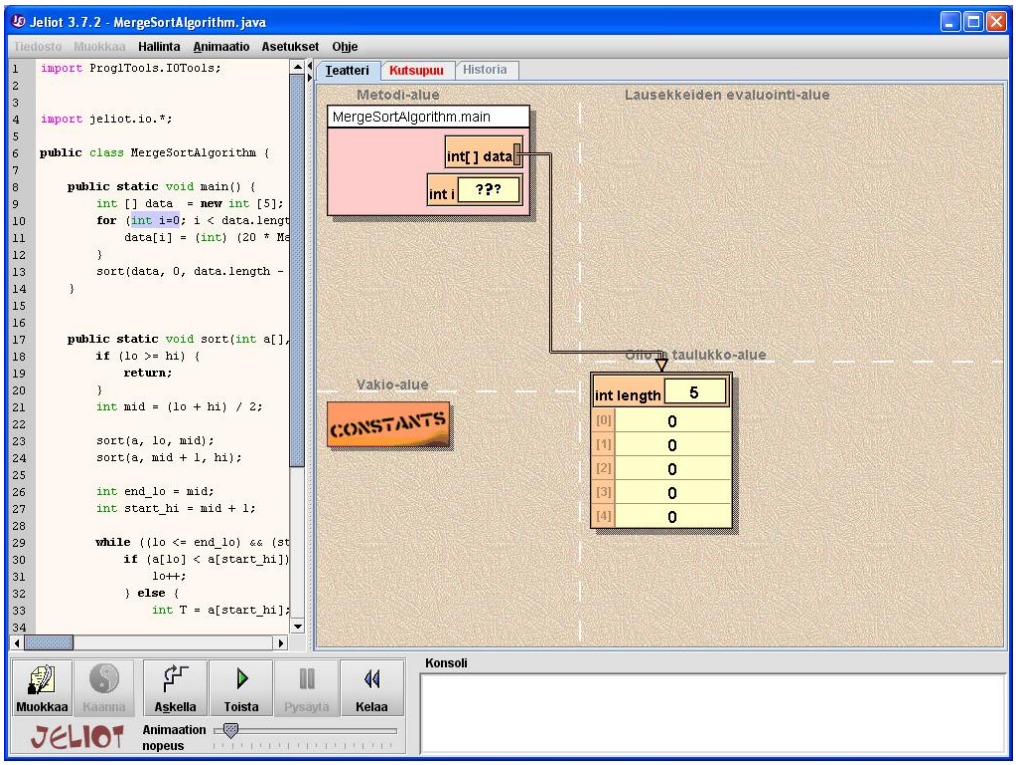
Siirrettävyys: Koska WebCat on toteutettu kokonaisuudessaan Javalla, sitä voidaan käyttää lähes kaikkien yleisimpien palvelinratkaisujen kanssa. Käyttäjät – mukaan lukien opettajat – tarvitsevat ainoastaan www-selaimen järjestelmää käyttääkseen.

Laajennettavuus: WebCat on suunniteltu toimimaan liitännäisten varassa – jopa mukana tulevat toiminnallisuudet on toteutettu kokonaan liitännäisinä. Tämän seurauksena opettaja

voi käyttää tarkastamisessa haluamiaan komponentteja ja tekniikoita. Lisäksi WebCat on edellisen johdosta kieliriippumaton.

Tuki manuaaliselle arvioinnille: Sen lisäksi, että WebCat mahdollistaa haluttujen automaattisen tarkastuksen komponenttien käytön, siinä on sisäänrakennettu tuki manuaaliselle arvioinnille: opettaja voi kirjoittaa kommenttinsa suoraan arvosteltavan ohjelman koodin sekaan. Järjestelmä lähettää lisäksi automaattisesti sähköpostin oppilaalle, kun tämän työ on arvosteltu.

2.6 Jeliot3



Kuva 6 Jeliot3

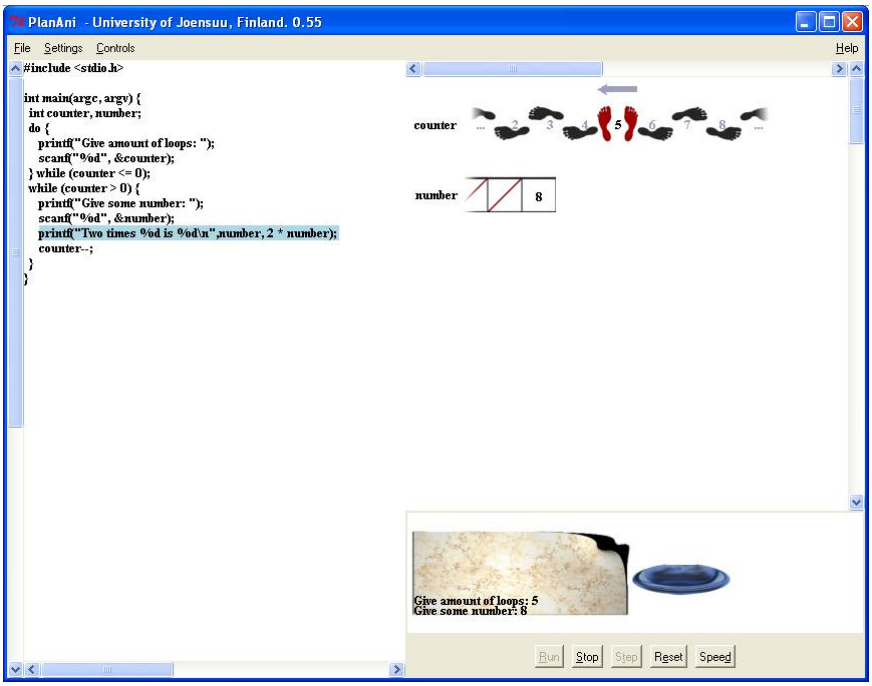
Jeliot3 on Java-kielisen ohjelman suoritusta visualisoiva järjestelmä, jonka tarkoituksena on opettaa aloittelevalle ohjelmoijalle proseduraalisen ja olio-ohjelmoinnin perusteet. Morenon ym. (2004) mukaan järjestelmän avainominaisuuksia ovat:

Visualisoinnit: Visualisoinnit on pyritty pitämään mahdollisimman johdonmukaisina; lisäksi tavoitteena on ollut niiden kokonaisvaltaisuus ja jatkuvuus.

Joustavuus: Java-tuesta on pyritty tekemään mahdollisimman laaja (toisin sanoen minkä tahansa Java-ohjelman pitäisi olla ajettavissa Jeliot3:n avulla). Lisäksi järjestelmä on helposti laajennettavissa.

Käytettävyys: Järjestelmä on pyritty pitämään mahdollisimman helppokäyttöisenä.

2.7 PlanAni

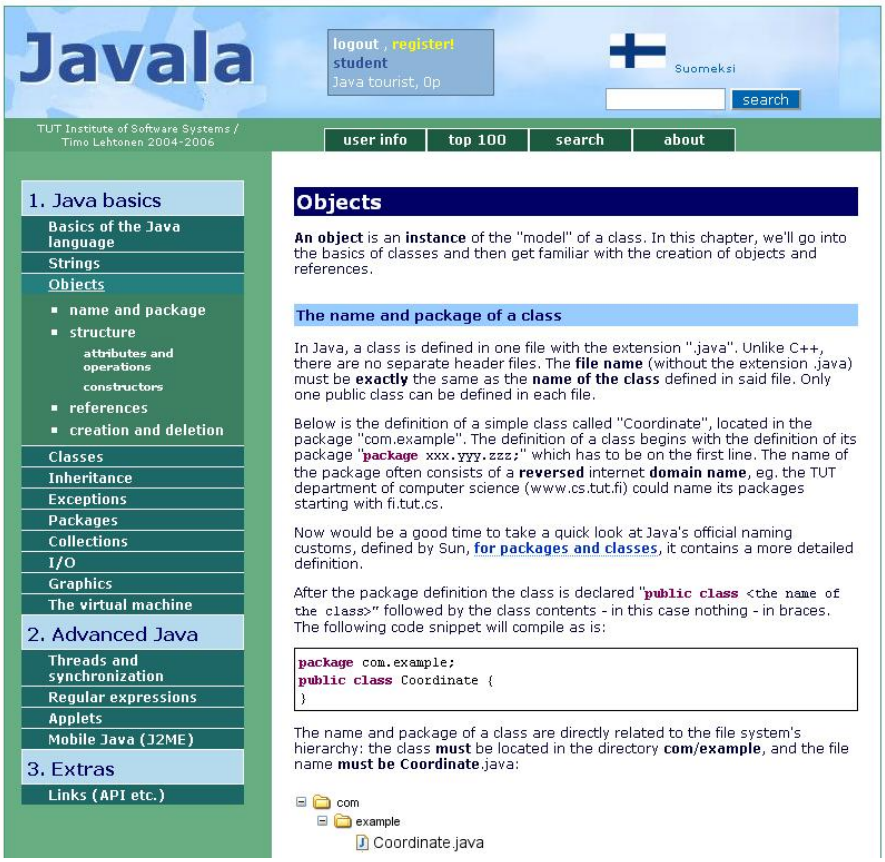


Kuva 7 PlanAni

PlanAni on järjestelmä, jonka avulla pyritään opettamaan aloitteleville ohjelmoijille muuttujien rooleja. Se visualisoi muuttujien erilaiset roolit (esimerkiksi silmukkamuuttuja on rooliltaan *askeltaja*) animaatioiden avulla. Jokaista mahdollista (opetettavaa) roolia kohden on erillinen kuva, jota animoidaan muuttujan arvon muuttuessa. Ohjelma sisältää valmiina kolme eri ohjelmointikieltä (Java, C ja Pascal) ja viisi esimerkkianimaatiota jokaisella kielellä.

Omien animaatioiden määrittely onnistuu erillisiä animaatiokomentoja lisäämällä. (Sajaniemi, 2006).

2.8 Javala



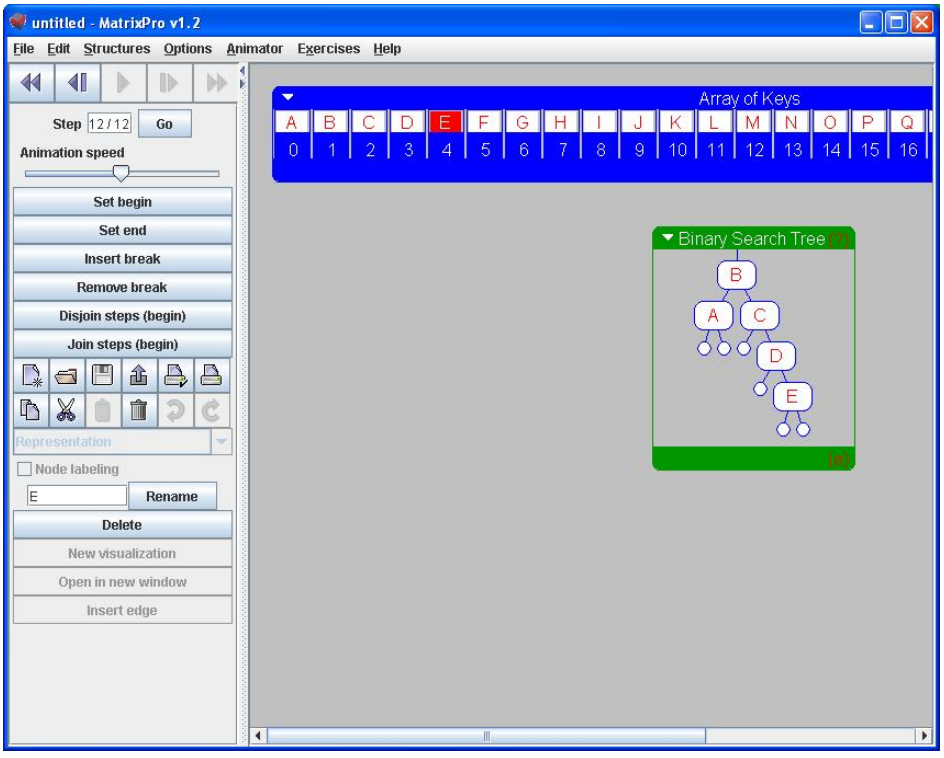
Kuva 8 Javala

Javala on avoin, verkossa toimiva opetusympäristö, jonka tarkoitus on opettaa Java-ohjelmointia aloittelijoille. Ympäristö koostuu teoriaosuuksien lisäksi harjoitustehtävistä, joissa oppilasta pyydetään toteuttamaan annettuun ohjelmarunkoon halutun lopputuloksen aikaansaava ohjelmalohko. Tehtävät on pisteytetty yhdestä kolmeen, ja Javala ylläpitää listaa parhaat pisteet saavuttaneista. Lehtosen (2005) mukaan kilpailu sijoituksista top-100-listalla motivoi oppilaita järjestelmän käyttöön: eräät opiskelijoista ovat käyttäneet järjestelmää jopa kuuden tunnin yhtämittaisten jaksojen ajan.

2.9 Scheme-robo

Scheme-robo on funktionaalisella Scheme-kielellä kirjoitettujen ohjelmointiharjoitusten arviointijärjestelmä. Kokonaisten ohjelmien sijasta se tarkistaa yksittäisiä funktioita. Saikkosen ym. (2001) mukaan järjestelmä keskittyy yksittäisiin funktioihin sen takia, että kokonaisen ohjelmakokonaisuuden oikeellisuuden tarkistaminen automaattisesti on vaikeampaa, johtuen esimerkiksi tulostuslauseiden erilaisuudesta. Oikeellisuuden lisäksi järjestelmä tarjoaa erilaisia työkaluja harjoitusten analysointiin, mukaan lukien ohjelman rakenteen ja suoritusajan analysointi sekä plagiointien selvittäminen.

2.10 MatrixPro



Kuva 9 MatrixPro

MatrixPro on järjestelmä, jonka avulla opettaja voi luoda animaatioita algoritmien suorituksesta. Karavirran ym. (2004a) mukaan järjestelmän avainominaisuuksia ovat:

Helppo käyttöönotto: Omien animaatioiden muodostaminen on vaivatonta: MatrixPro tarjoaa algoritmien automaattisen animoinnin lisäksi mahdollisuuden määritellä animaatiot itse.

Yksilöidyt animaatiot: Animaatioita pystyy muokkaamaan, sekä käyttämällä valmiita animaatiokomponentteja että säätämällä animaatioaskelia.

Animaatioiden tallennus ja lataus: Tietorakenteet ja niistä muodostetut animaatiot pystyy tallentamaan useissa eri dataformaateissa.

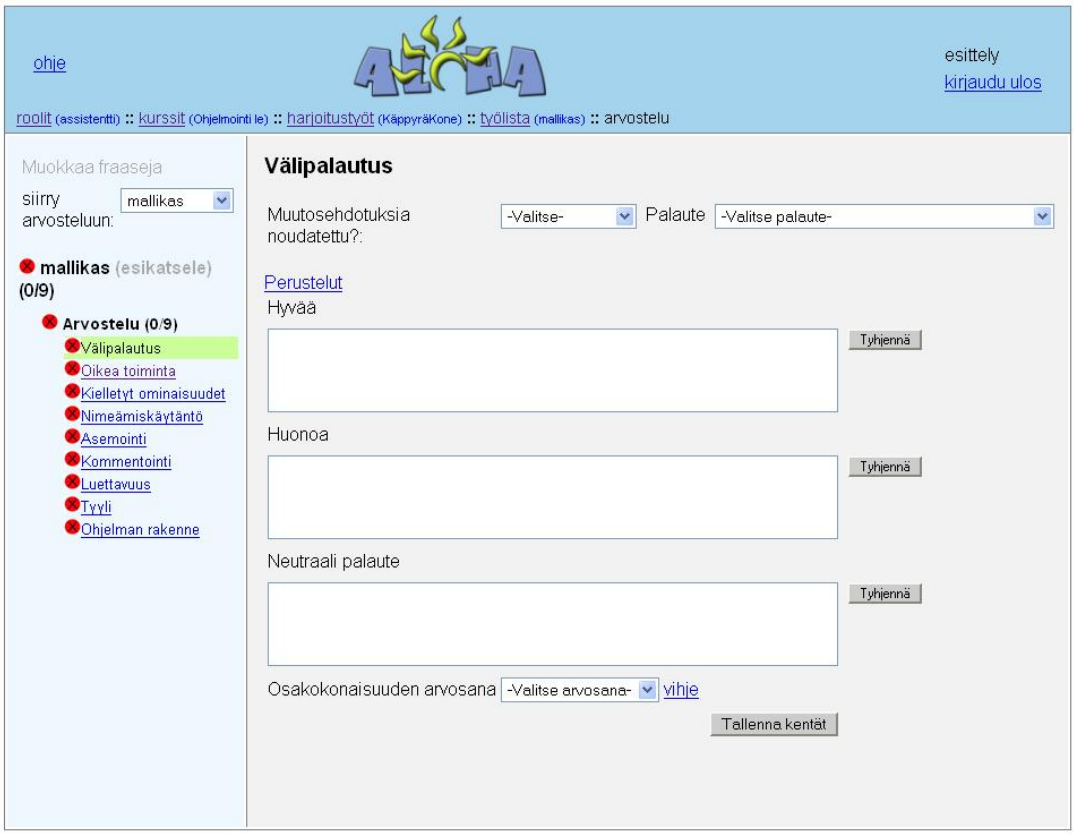
Muokattava käyttöliittymä: Käyttöliittymä on muokattavissa eri käyttäjäryhmiä varten.

Kokoelma tietorakenteita: MatrixPron mukana tulee tietorakennekirjasto, jota käyttäjä voi hyödyntää.

Harjoitukset: Käyttäjän muodostamaa simulaatiota algoritmista voidaan verrata varsinaisen algoritmin muodostamaan.

MatrixPro käyttää algoritmien animoimiseen Matrix -nimistä algoritmien visualisointijärjestelmää; myös TRAKLA2 käyttää vastaavaa järjestelmää.

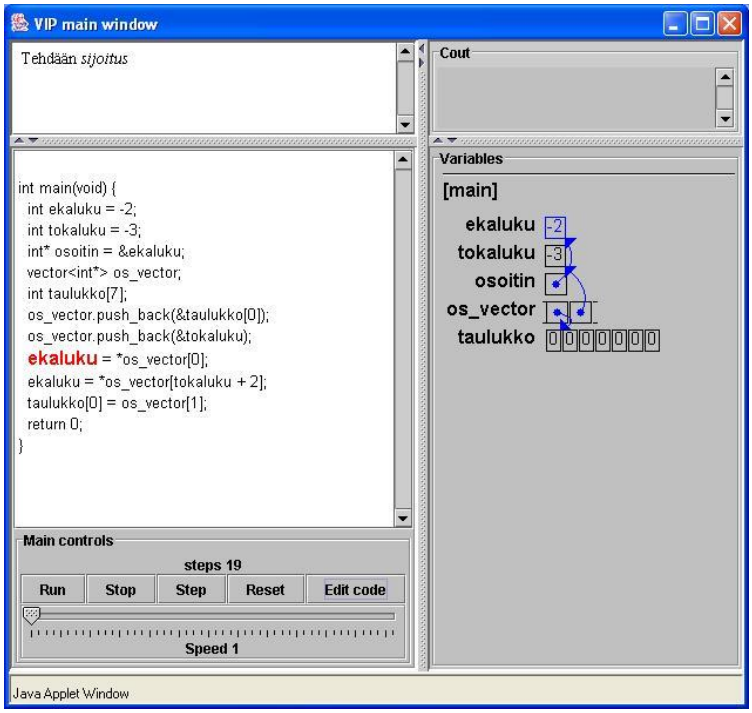
2.11 Aloha



Kuva 10 Aloha

Aloha on TTY:lla toteutettu harjoitustöiden arviointijärjestelmä. Ahoniemen & Reinikaisen (2007) mukaan sen pääasiallisena tarkoituksena on helpottaa töiden kvalitatiivista arvostelua massakursseilla. Aloha tarjoaa valmiit otsikot ja kategoriat arvostelua varten verkossa; näin arvostelu säilyy samankaltaisena tarkastavasta opettajasta ja arvostelun ajankohdasta huolimatta. Opettaja voi halutessaan muokata arvostelukategorioita sekä muuttaa niiden painotuksia. Vaikka järjestelmä on nykyisellään käytössä ainoastaan ohjelmointikurssien arvostelussa, sitä voitaisiin kirjoittajien mukaan hyödyntää muillakin kursseilla; arvostelumallin muodostaminen tosin vaatii jonkin verran ohjelmointitaitoa.

2.12 VIP



Kuva 11 VIP

VIP (Visual Interpreter) on C++ -kielisen ohjelman suoritusta visualisoiva järjestelmä. Virtasen ym. (2005) mukaan sen kehityksen päämääriä ovat olleet:

Käytön helppous: Uusien visualisointien luonnin pitää olla yksinkertaista. Lisäksi käyttöliittymän pitää olla huolella suunniteltu, jotta opiskelijat voivat käyttää järjestelmää vaivattomasti.

Kokonaisvaltaiset selitykset: Suorituksen selitysten pitää hyödyttää opiskelijoita

Muokattava koodi: Opiskelijoiden pitää pystyä muokkaamaan ohjelmakoodia kesken suorituksen.

Saavutettavuus: Järjestelmän pitää olla käytettävissä www-selaimen avulla.

Käytettäväksi ohjelmointikieleksi on valittu *C--*, joka on C++:n alijoukko. Kieli ei toteuta kaikkia C++:n ominaisuuksia, mutta on ylöspäin yhteensopiva sen kanssa.

3 Tutkimus

Ohjelmoinnin perusopetuksen verkostohankkeeseen liittyen suoritettiin tutkimus, jolla pyrittiin kartoittamaan ohjelmoinnin opetuksen tilaa suomalaisisssa korkeakouluissa. Alkuperäisenä ideana oli suorittaa tutkimus kaksiosaisesti siten, että vastaajat olisivat aluksi täyttäneet www-lomakkeen, jossa kysyttiin perustietoja kurssista, opetuskäytännöistä ja opetusjärjestelmien tuttuudesta. Näiden vastausten pohjalta oli tarkoitus syventää tietoja edelleen semistrukturoidun haastattelun avulla. Koska verkkolomakkeeseen kuitenkin tuli vastauksia verraten vähän, päädyttiin tutkimus suorittamaan pääosin kokonaan haastattelujen avulla. Haastateltavia oli yhteensä 26 (N=26). Mukana ovat lähes kaikki Suomessa ohjelmoinnin opetusta tarjoavat korkeakoulut.

3.1 Haastattelun rakenne

Haastattelussa pyrittiin selvittämään ohjelmoinnin opetukseeen liittyviä seikkoja kolmessa eri pääkategoriassa: opettaja, kurssi ja opetusvälineet. Haastattelussa käytettiin hyväksi www-lomaketta varten luotua haastattelurakennetta, mutta keskustelun annettiin toisaalta myös kulkea vapaasti aiheiden ympärillä ja sivussa. Haastattelu aloitettiin selvittämällä eräitä perustietoja haastateltavasta. Seuraavaksi käytiin läpi opetettava kurssi (mikäli haastateltavan vastuulla oli useampia kursseja, valittiin näistä ohjelmoinnin perusopetukseen läheisimmin liittyvä(t)) ja opetusmateriaalit. Samalla oli tarkoitus syventää tietoja opetuskäytännöistä ja -asenteista, sekä avoimien kysymysten että muihin aiheisiin liittyvän keskustelun kautta. Viimeisessä osiossa selvitettiin haastateltavan tietämystä ja

kokemuksia opetusjärjestelmistä ja niiden käytöstä omilla kursseilla. Haastattelulomake löytyy kokonaisuudessaan liitteestä 1.

3.1.1 Tiedot opettajasta

Opettajasta itsestään selvitettiin perustietojen lisäksi tietoja ajankäytöstä kurssin eri osa-alueisiin sekä tutkimukseen ja opetukseen liittyvän yhteistyön (muiden kuin oman oppilaitoksen henkilöiden kanssa) laajuutta ja laatua. Perustietojen osalta selvitettiin ainakin seuraavat tiedot:

- *Henkilötiedot:* nimi, oppilaitos, työvuosien määrä opetustehtävissä ja virkanimike.
- *Ajankäyttö:* työtuntien määrä vuodessa, vastuun jakautuminen opetus- ja tutkimustehtävien (ja muiden tehtävien) välillä sekä ajankäyttö kurssin eri osa-alueisiin, mukaan lukien valmistelu, luennointi ja tarkastus.
- *Yhteistyö:* yhteistyön määrä ja laatu sekä yhteistyötahot eri korkeakouluissa.

3.1.2 Tiedot kurssista

Kurssista kysyttiin tietoja viidellä eri osa-alueelle:

- *Perustiedot:* kurssin nimi, opetuksessa käytettävä ohjelmointikieli, kurssin laajuus ja tyyppi (perus-, aine- tai syventävät opinnot). Lisäksi selvitettiin osanottajamäärä (tai ainakin suuruusluokka, mikäli tietoja ei ollut helposti saatavilla), läpäisyprosentti ja keskiarvosana.
- *Vastuualueet:* kurssin eri osa-alueiden vastuuhenkilöt, tarkoituksena lähinnä selvittää kuinka paljon kurssista vastaava henkilö (eli käytännössä haastateltu) käyttää apuvoimia kurssin eri osa-alueilla.
- *Opetus:* kurssilla käytössä olevat menetelmät ja niiden kokonaismäärä. Lisäksi selvitettiin miten harjoitustyöt ja muut vastaavat menetelmät

käytännössä toimivat, koska eroavaisuuksia eri korkeakoulujen välillä löytyy melkoisesti.

- *Opetusmateriaalit:* Mitä opetusmateriaaleja kurssilla on käytössä ja mistä ne ovat peräisin.

- *Painotus opetuksessa:* mikä on opetuksen painopiste, ts. mitä asioita voidaan pitää kurssin tärkeimpinä oppimispäämäärinä.

Mikäli haastateltavana oli useampia henkilöitä samasta oppilaitoksesta, ja molempien opetettava kurssi oli sama, pyrittiin osuudessa selvittämään erityisesti kurssin käytäntöjen välisiä eroja.

3.1.3 Opetusjärjestelmät

Viimeisessä osuudessa pyrittiin selvittämään tietoisuutta opetusjärjestelmistä ja asenteita niitä kohtaan. Lisäksi selvitettiin muita käytössä olevia opetusjärjestelmiä ja niiden käytön laajuutta sekä toiveita uusien järjestelmien suhteen. Haastattelu jakautui neljään osa-alueeseen:

- *Tietoisuus järjestelmistä:* haastateltaville esitettiin yhdentoista verkostohankkeeseen liittyvän järjestelmän nimet ja pyydettiin kertomaan, kuinka tuttuja järjestelmät olivat.

- *Väittämät järjestelmien soveltuvuudesta ja toiminnallisuudesta:* järjestelmistä yleisesti esitettiin väittämiä niiden soveltuvuuteen ja toiminnallisuuteen liittyen. Haastateltavat arvioivat väittämiä asteikolla yhdestä viiteen (1 – täysin eri mieltä, 5 – täysin samaa mieltä).

- *Käyttöönotto:* osiossa selvitettiin haastateltavan aikomuksia ottaa jokin mainituista tai jokin vastaava järjestelmä käyttöön opetuksessaan. Lisäksi pyrittiin selvittämään millaisten syiden takia järjestelmät jätetään ottamatta käyttöön, tapauksesta riippuen joko väittämien tai avoimien vastausten avulla.

- *Toiveita järjestelmille käyttöönottoon liittyen:* viimeisessä osiossa kysyttiin, oliko haastateltava joskus kaivannut jotain muunlaista välinettä jonkin opetuksensa osa-alueen apuvälineeksi. Lisäksi selvitettiin kuinka paljon aikaa haastateltava olisi valmis käyttämään järjestelmien käyttöönottoon ja opetteluun. Lopuksi haastateltavalle tarjottiin mahdollisuus esittää muita haastattelun aikana mieleentulleita mielipiteitä niin opetuksesta kuin järjestelmistäkin.

3.2 Haastattelut käytännössä

Haastatteluissa noudatettiin samaa kysymyspohjaa kuin mitä verkossa olevassa lomakkeessakin oli. Keskustelun annettiin kuitenkin rönsyillä vapaasti, haastattelijan ohjatessa tarvittaessa suuntaa siten, että kaikkiin lomakkeessa oleviin kysymyksiin saatiin vastaukset. Kaikki haastattelut nauhoitettiin, minkä lisäksi haastattelija teki muistiinpanoja koko haastattelun ajan. Nauhoituksia käytettiin purkaessa lähinnä selventämään muistiinpanoja tarpeellisissa kohdin.

4 Tulokset

Tässä luvussa esitellään haastattelujen ja verkkokyselyn tulokset jaettuna kolmeen kategoriaan: ohjelmoinnin opettajat, ohjelmoinnin ja algoritmien peruskurssit ja opetusjärjestelmät. Viimeisessä osiossa esitellään lisäksi haastateltavien vapaita kommentteja ohjelmoinnin opetukseen ja opetusjärjestelmiin liittyen.

4.1 Ohjelmoinnin opettajat

Haastatteluun (tai kyselyyn) osallistui yhteensä 26 (N=26) ohjelmoinnin tai algoritmien peruskurssien opettajaa lähes kaikista ohjelmoinnin opetusta antavista korkeakouluista Suomessa (ks. Taulukko 1). Yleisin virkanimike oli *lehtori* (11 kpl) ja seuraavaksi yleisimmät *tutkija* (4 kpl) ja *assistentti* (3 kpl). Haastateltujen opettajien virkanimikkeet on esitetty taulukossa 2.

Taulukko 1 Haastateltavien oppilaitokset

Helsinki	1
Joensuu	3
Jyväskylä	2
Kuopio	1
Lappeenranta	1
Oulu (tekninen)	2
Tekninen korkeakoulu	6
Oulu	2
Tampere (tekninen yo)	2
Tampere	1
Turku	2
Åbo Akademi	3

Taulukko 2 Haastateltujen opettajien virkanimikkeet

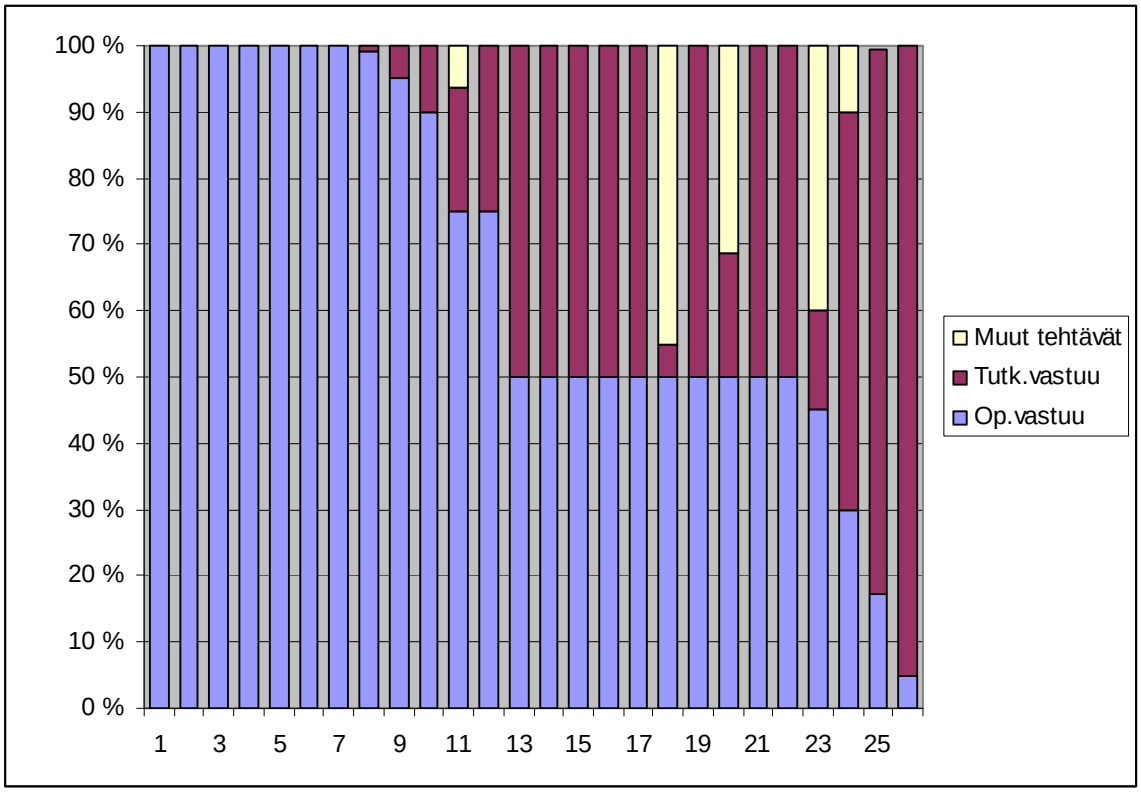
Assistentti:	3 kpl
Lehtori:	11 kpl
Opettava tutkija:	1 kpl
Professori:	2 kpl
Tuntiopettaja:	2 kpl
Tutkija:	4 kpl
Tutkimusapulainen:	1 kpl
Yliassistentti:	2 kpl

Haastatelluilla oli työvuosia opetustehtävissä 3-32. Opettajien jakautuminen opetuskokemuksen mukaan on esitetty taulukossa 3.

Taulukko 3 Haastateltujen opettajien työkokemus opetustehtävistä

0-10 vuotta	11 kpl
11-20 vuotta	8 kpl
21-30 vuotta	5 kpl
yli 30 vuotta.	2 kpl

Kahdeksan haastatelluista ilmoitti, ettei heidän toimenkuvaansa kuulu opetusvastuun lisäksi muita tehtäviä. Lopuilla oli vaihteleva määrä tutkimus- tai muuta vastuuta (sisältäen esimerkiksi opintoneuvonnan). Vastuualueet on esitetty kuvaajassa 1.



Kuvaaja 1 Opettajien vastuualueet

4.1.1 Opettajien ajankäyttö

Haastateltuja pyydettiin arvioimaan ajankäyttöään kurssiin (ks. luku 5.2) liittyen neljässä eri kategoriassa: kurssin ja materiaalien valmistelu, luennointi, tehtävien tarkastus ja tenttien tarkastus. Ajankäyttö kurssin suunnitteluun vaihteli 20:sta tunnista 250:n tuntiin. Tehtävien ja tenttien tarkastukseen käytettävä aika luonnollisesti vaihtelee (myös) sen mukaan, miten paljon osanottajia kurssilla on. Yhteensä kurssia kohti käytetty aika vaihteli 60 ja 400 tunnin välillä. Määrään pitää edelleen lisätä muihin kurssiin liittyvien tehtävien (sähköposteihin vastaaminen, vastaanotot, tehtävien hienosäätö jne.) vaatima aika, joka opettajien mukaan vaihteli yhdestä viikkotunnista 20:en. Opettajien ajankäytön keskiarvot on esitetty taulukossa 4.

Taulukko 4 Opettajien ajankäyttö tunteina kurssia kohti keskimäärin

Suunnittelu ja materiaalien valmistelu:	124 h
Luennointi ja muu lähiopetus:	28 h
Tehtävien tarkastus:	25 h
Tenttien tarkastus:	29 h
Kurssin ajankäyttö yhteensä:	206 h

Useat haastatelluista kertoivat kuitenkin ajankäyttönsä vaihtelevan huomattavasti: kurssin suunnitteluun ensimmäisellä kerralla käytettiin huomattavasti enemmän aikaa kuin mitä seuraavina vuosina – useimmat kertoivat lähinnä päivittävänsä harjoitus- ja demonstraatiotehtävät. Toisaalta opetuskielen vaihdon (esimerkiksi Javasta Pythoniin) tai yhdistymisen toiseen korkeakouluun arveltiin nostavan ajankäyttötarvetta huomattavasti. Ajankäyttö yhtä opintopistettä kohti vaihteli 13 ja 80 tunnin välillä keskiarvon ollessa n. 31 tuntia.

4.1.2 Opetukseen ja tutkimukseen liiittyvä yhteistyö

Haastatteltavia pyydettiin määritteleмään yhteistyönsä määrää ja laatua, mukaanlukien kaikki opetukseen tai tutkimukseen (erikseen) liittyvä kommunikaatio *muiden kun oman oppilaitoksen opettajien kanssa*. Tutkimuksen yhteydessä (niillä opettajista, jotka tutkimusta yleensä tekivät) yhteistyötä esiintyi laajemminkin, esimerkiksi yhteisten artikkelien tai tutkimuskohteiden kautta. Opetukseen liittyvä yhteistyö sen sijaan oli vähäisempää. Yhteistyömuodoiksi opetuksen yhteydessä mainittiin:

- verkostohankkeen tai kehitysverkoston seminaareihin osallistuminen,
- keskustelu ja tietojen vaihto muiden opettajien kanssa,
- muissa korkeakouluissa opettaminen,
- kurssien yhteinen suunnittelu AMK:n kanssa,
- kokemuksien vaihtaminen ohjelmointikielen vaihtoon liittyen,

- oman opetusmateriaalin antaminen vapaaseen käyttöön (tai suoraan toisen korkeakoulun käyttöön),
- yhteistyö opetusjärjestelmien kehittämisessä ja/tai ylläpidossa,
- opetusjärjestelmiin liittyvien tehtävien kehittäminen muille korkeakouluille sekä
- muiden alan ihmisten tunteminen.

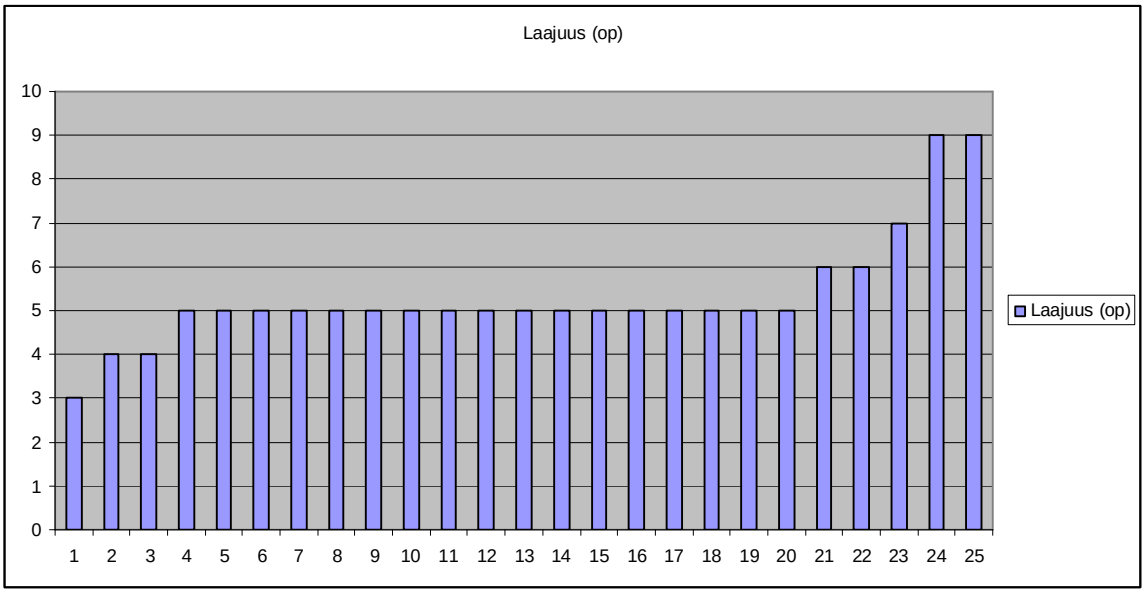
Syyksi yhteistyön vähyyteen mainittiin muun muassa (opetuskäytössä) harvinaisemman ohjelmointikielen käyttö opetuksessa ja se, että opettajan pitää olla itse vastuussa kurssistaan.

4.2 Ohjelmoinnin ja algoritmien peruskurssit

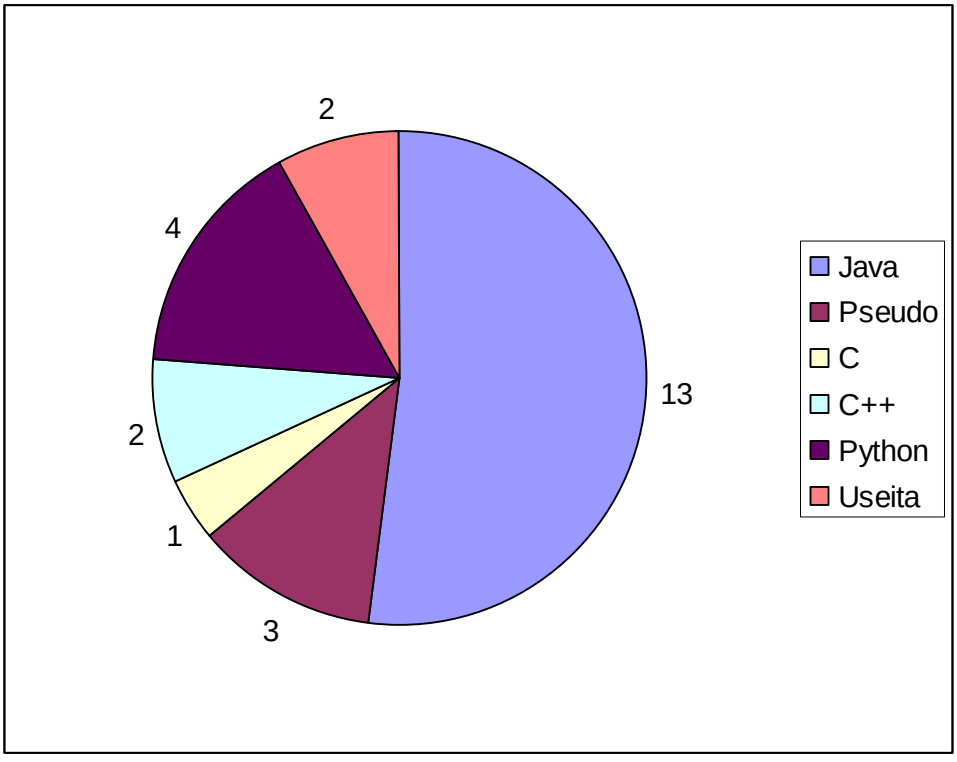
Haastateltuja pyydettiin määrittämään eräiden kurssin perustietojen lisäksi vastuun jakautuminen kurssin eri osa-alueilla sekä opetuksen painopiste kurssilla. Haastateltujen vastuualueena olevat kurssit olivat muutamaa poikkeusta lukuunottamatta ohjelmoinnin ja algoritmien peruskursseja. Mikäli haastatellun vastuualueseen kuului useampia kursseja, valittiin tarkemman tarkastelun alle juuri peruskurssi. Mikäli useampi opettaja opetti samassa oppilaitoksessa saman nimisen kurssin eri versiota, tarkasteltiin kursseja erillisinä kursseina (johtuen pitkälti siitä, että samoista määrittelyistä huolimatta kurssit saattoivat – esimerkiksi historiallisista syistä – olla käytännössä hyvinkin erilaisia).

4.2.1 Kurssien perustiedot

Kursseista 19 oli ohjelmoinnin peruskursseja loppujen käsitellessä tietorakenteiden ja algoritmien perusteita. Kurssien laajuus (kuvaaja 2) vaihteli kolmesta yhdeksään opintopisteeseen. Pääasiallisena opetuskielenä (kuvaaja 3) käytettiin Javaa (13 kurssia, tosin ainakin kahdella näistä on jo päätetty vaihtaa kieli Pythoniin ensi syksynä), Pythonia (4 kpl), pseudokieltä (4 kpl), C:tä (2 kpl) ja C++:aa (1 kpl). Kursseista viisi kuului aineopintoihin loppujen ollessa perusopintoja. Kaikki kursseista olivat pakollisia pääaineenaan tietojenkäsittelyä tai tietotekniikkaa (tai vastaavaa linjaa) opiskeleville.



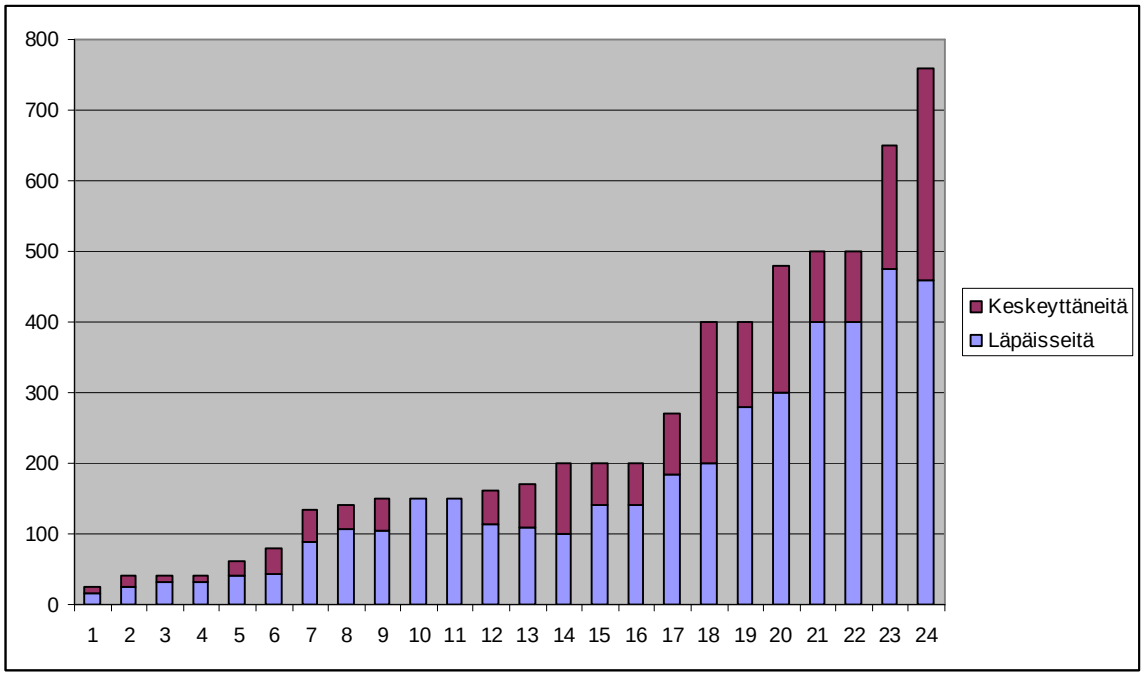
Kuvaaja 2 Kurssien laajuus opintopisteinä



Kuvaaja 3 Kursseilla käytetty ohjelmointikieli

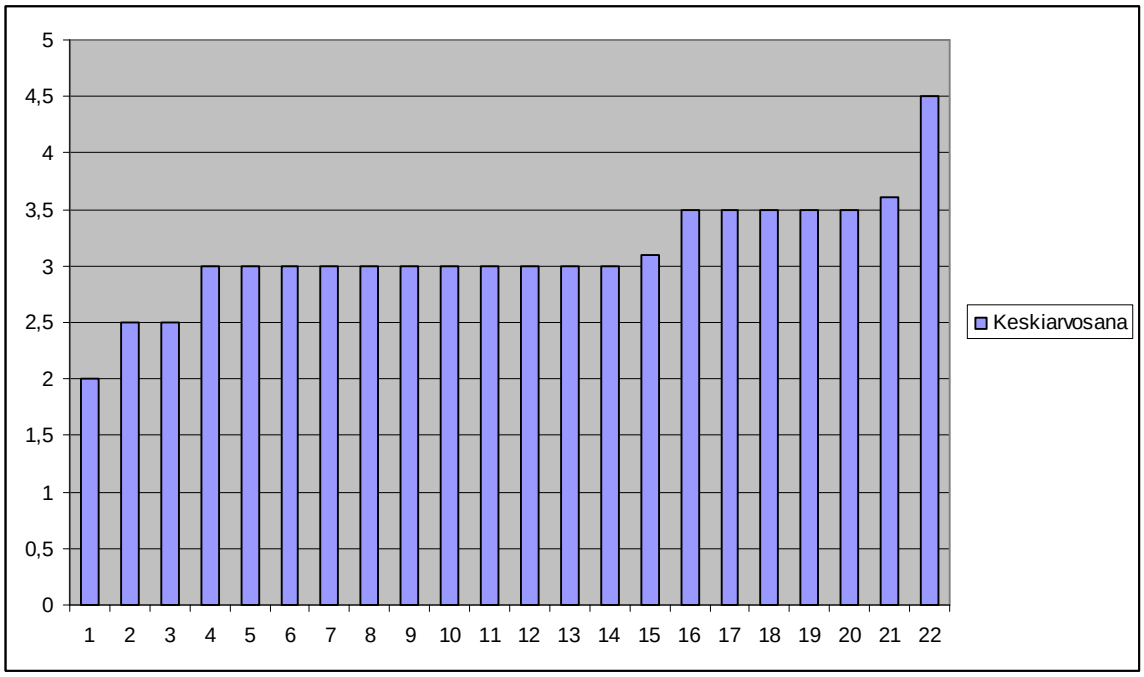
Kurssien osanottajamäärä vaihteli 25:sta aina 760:n asti. Läpäisyprosentti (tarkoittaen niitä kurssille ilmoittautuneita opiskelijoita, jotka saivat sen suoritettua ennen seuraavan syksyn

kurssin alkua) vaihteli muutamaa ääripäätä lukuunottamatta melko tasaisesti 60 % ja 80 % välillä. Parhaimmillaan tosin läpäisyprosentti hipoi kahta sataa, koska eräässä korkeakoulussa saatiin kurssimuutoksen myötä osa vanhoista opiskelijoista läpäisemään uusittu kurssi. Osanottajamäärät ja läpäisy- ja keskeytysprosentit on koottu kuvaajaan 4.



Kuvaaja 4 Kurssien osanottajamäärät ja läpäisseiden osuus (ka: 246 osall., läpäisy% 70,4)

Kurssien keskiarvosana (mikäli otetaan ainoastaan kurssin läpäisseet huomioon) vaihtelee 2:n ja 4,5:n välillä. Suurin osa keskiarvosanoista sijoittuu kuitenkin kolmosen ympärille. Opettajien mukaan poikkeuksellisen hyvää keskiarvoa selittää yleensä se, että kurssin joka tapauksessa läpäisevät ainoastaan ne opiskelijat, jotka oikeasti osaavat jotakin. Vastaavasti poikkeuksellisen alhaista keskiarvoa opettajat selittivät muuttuneen sisäänottokäytännön aiheuttamalla opiskelija-aineksen keskimääräisellä huonontumisella. Kurssien keskiarvosanat (sikäli kun ne ovat tiedossa) on koottu kuvaajaan 6.



Kuvaaja 5 Kurssien keskiarvosanat

Kurssit koostuivat melko samankaltaisista aineksista, joskin määrät vaihtelivat jonkin verran. Kaikkiin kursseihin kuului luentoja (määrän vaihdellessa 18 ja 50 tunnin välillä) ja yksi tai kaksi tenttiä. Lisäksi kursseihin kuului vaihteleva määrä demonstraatiotehtäviä (tarkoittaen laskuharjoitustyyppisiä pienryhmissä läpikäytäviä harjoitustehtäviä) ja harjoitustehtäviä (tarkoittaen tässä yhteydessä laajempaa, erikseen palautettavaa ohjelmointitehtävää). Kurssien opetusmenetelmät on koottu taulukkoon 5.

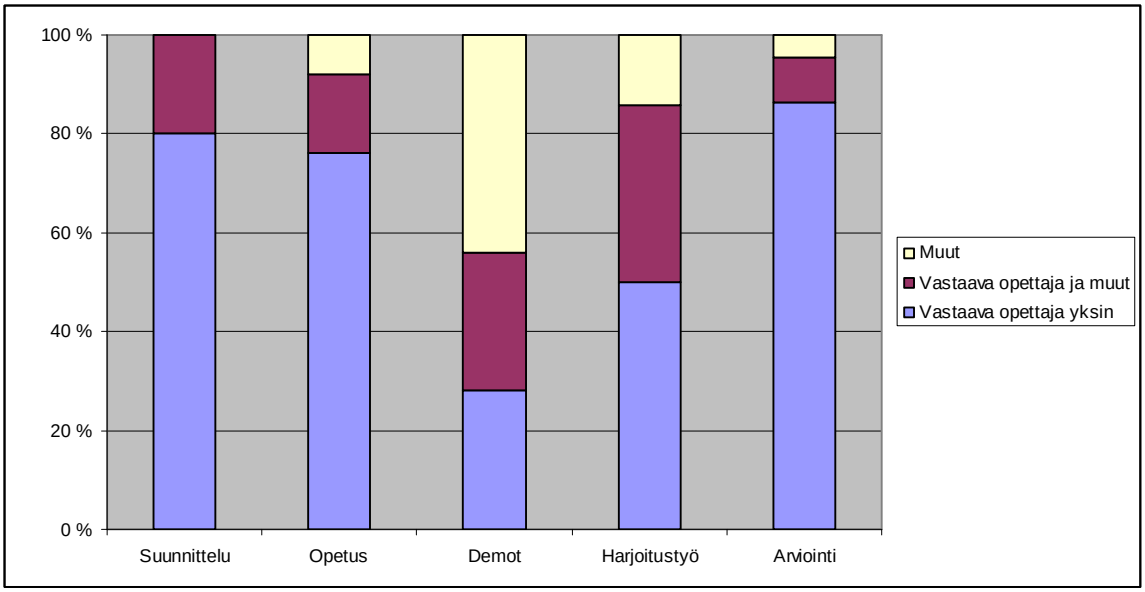
Taulukko 5 Kursseilla käytettävät opetusmenetelmät

	Laajuus op.	Luentoja (h)	Demoja (h)	Harjoitustöitä (kpl)	Tenttejä (kpl)
	3	28	18	-	1
	4	30	70	-	1
	4	36	36	4	1
	5	18	18	-	1
	5	20	-	3	1
	5	20	-	3	1
	5	22	-	1	1
	5	28	10	-	1
	5	28	28	1	2
	5	28	8	-	1
	5	28	12	-	1
	5	30	12	-	1
	5	32	-	-	1
	5	36	16	-	1
	5	36	36	4	1
	5	36	20	-	1
	5	40	12	-	1
	5	40	40	-	1
	6	18	-	1	-
	6	50	20	1	1
	7	24	20	1	2
	9	30	26	2	1
	9	36	36	1	2
Keskiarvo	*5,35*	*30,17*	*24,33*	*2*	*1,14*

4.2.2 Vastuu kurssien eri osa-alueilla

Haastateltuja pyydettiin määrittämään vastuuhenkilöt kurssin eri osa-alueilla, mukaanlukien suunnittelu, opetus ja arvostelu. Tulosten mukaan kurssien suunnittelu on yleensä yhden henkilön (yleensä haastateltava) vastuulla, joskin muutamat haastatelluista kertoivat, että eräitä pääpiirteitä suunnittellaan yhdessä ohjelmoinnin muiden opettajien kanssa. Harjoitustöissä (sisältäen sekä demonstraatiotehtävät että laajemmat harjoitukset) vastuu jakautui tasaisemmin haastateltujen ja kurssiassistenttien kesken. Kurssien arvostelu

näyttää olevan yksinomaan vastuuopettajien harteilla, joskin osalla kursseista arvosana määräytyi ainakin osittain harjoitustöistä saatujen tulosten mukaan. Kurssien eri osa-alueiden vastuun jakautuminen on esitetty kuvaajassa 6.

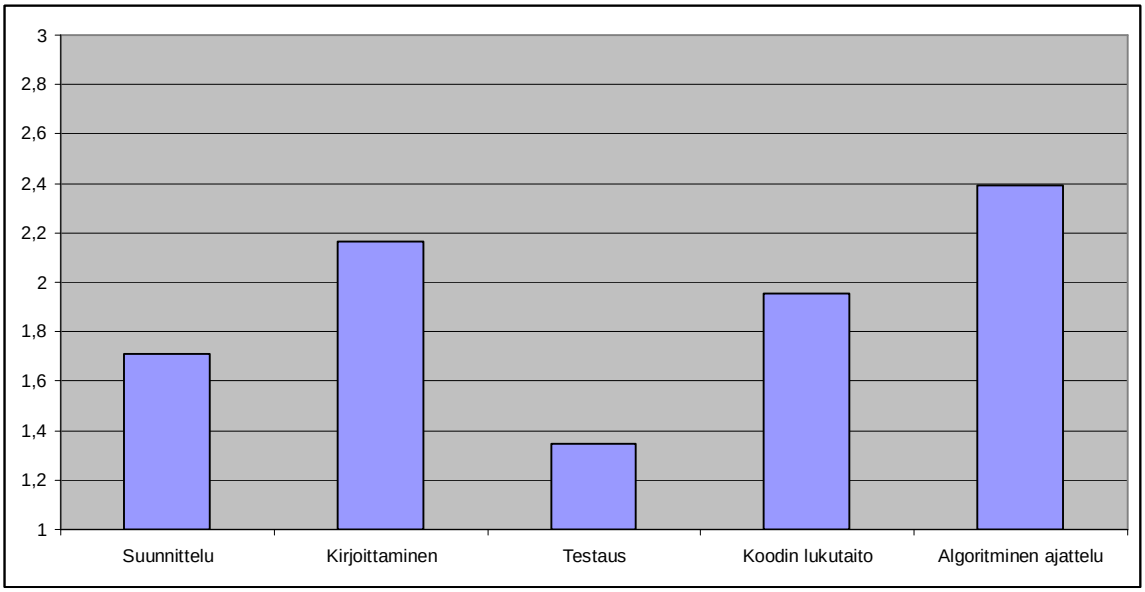


Kuvaaja 6 Vastuu kurssien eri osa-alueista

Kursseilla käytetään melko samankaltaisia opetusmateriaaleja. Valtaosa (86% opettajista) käyttää kurssillaan itse valmistamaansa luentomonistetta (tai kokoelmaa luentokalvoista). Huomionarvoisesti useimmat mainitsivat itse muokanneensa monisteen haluamakseen, vaikka olisivatkin perineet vanhan version kurssin aikaisemmalta vastuuopettajalta. Useimmat suosittelivat lisäksi yhtä tai useampaa kirjaa opiskelijoille lisämateriaaliksi, pakollisena varsinaista kirjaa ei kuitenkaan ollut käytössä yhdelläkään kurssilla. Yleisimpiä kirjasuosituksia olivat Walter Savitchin *Absolute Java* ja Arto Wiklan *Ohjelmoinnin perusteet Java-kielellä*. Muu kursseilla käytettävä materiaali koostuu ohjelmointiesimerkeistä, mallivastauksista ja muusta verkkomateriaalista; lähes kaikki haastatellut ilmoittivat tuottaneensa myös nämä materiaalit täysin itsenäisesti.

4.2.3 Opetuksen painopiste

Haastateltavia pyydettiin arvioimaan opetuksensa painopistettä sen mukaan, kuinka tärkeänä oppimispäämääränä he pitivät viittä ohjelmoinnin osa-aluetta (ohjelmien suunnittelu, ohjelmien kirjoittaminen, ohjelmien testaus, ohjelmakoodin lukutaito ja algoritminen ajattelu) omalla kurssillaan. Vastaukset arvioitiin kolmeportaisella asteikolla: vähemmän tärkeä (1), tärkeä (2) ja erittäin tärkeä (3). Kaikkien vastausten keskiarvot on esitetty kuvaajassa 7.



Kuvaaja 7 Opetuksen painopiste kaikilla kursseilla

Haastatellut mainitsivat lisäksi erikseen kurssiensa tärkeimmäksi oppimispäämääksi muun muassa seuraavat seikat:

- ”Tärkeintä on oppia ymmärtämään mitä ohjelmointi on”,
- ”Tärkeintä oppia abstrahointi ja kapselointi sekä muut olio-ohjelmoinnin erot algoritmisiin ohjelmointikieliin” ja
- ”Tärkeintä oppia algoritminen ajattelu, muut osa-alueet sisältyvät siihen implisiittisesti”.

Ohjelmien testausta painottivat lähinnä ne opettajat, jotka käyttivät harjoitustehtävien palautukseen sellaista järjestelmää (esim. WebCat), joka edellyttää testien ajamista

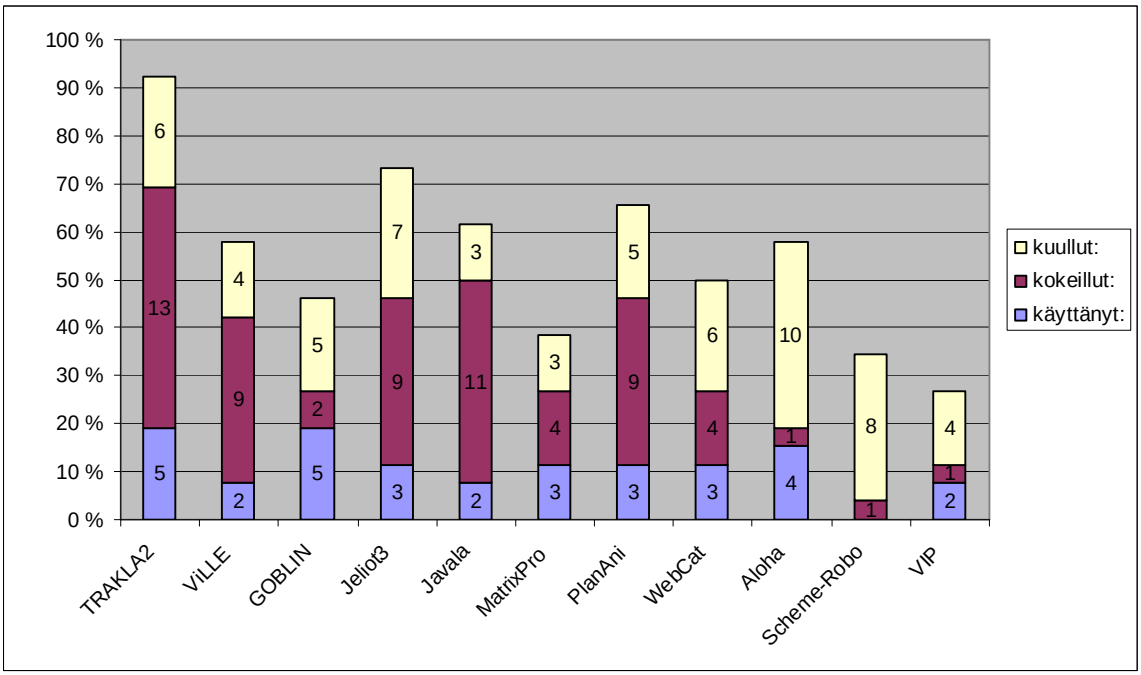
automaattisen tarkastuksen yhteydessä. Useimpien muiden haastateltujen mielestä testaaminen oli tärkeää, mutta sen erilliseen läpikäyntiin ei yleensä ollut aikaa kurssilla.

4.3 Opetusjärjestelmät

Haastattelussa kerättiin tietoja opetusjärjestelmistä neljässä eri pääkategoriassa: kuinka tuttuja järjestelmät ovat, millaisia asenteita haastateltavilla on yleisesti järjestelmiä kohtaan, millaisia ajatuksia haastateltavilla on järjestelmien käyttöönottoon liittyen ja millaisia muita järjestelmiä he ovat joskus kaivanneet opetuksensa avuksi. Tutkimukseen mukaan otettujen yhdentoista järjestelmän lisäksi kerättiin tietoa muista käytössä olevista vastaavista järjestelmistä ja niiden käytöstä sekä alkuperästä.

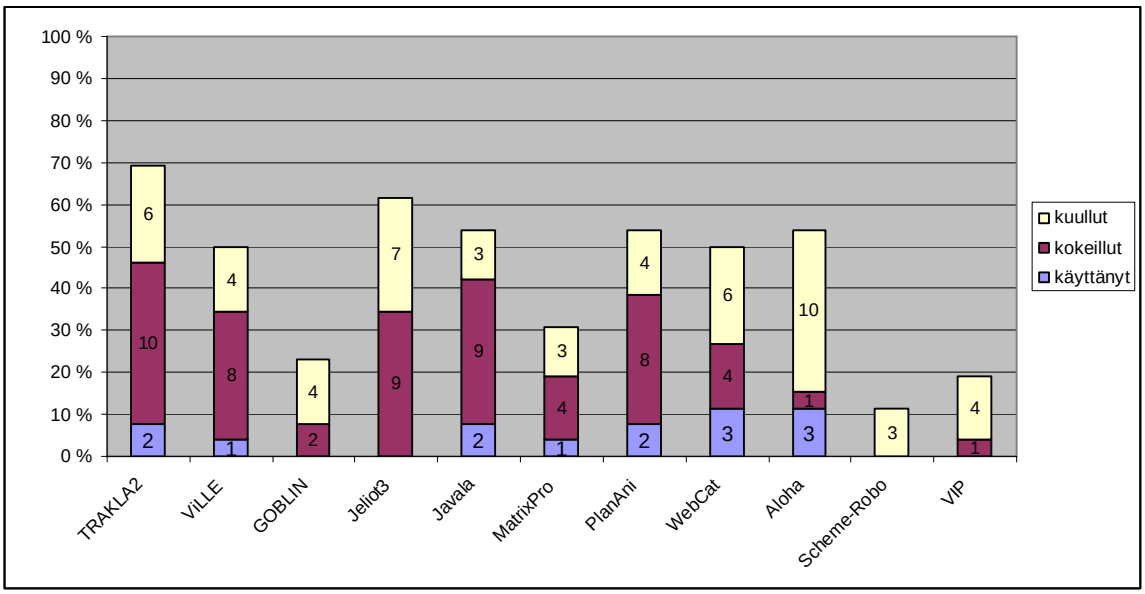
4.3.1 Tietoisuus järjestelmistä ja asenteet niitä kohtaan

Tietoisuutta järjestelmistä kerättiin luettelemalla yhdentoista tutkimuksessa mukana olleen järjestelmän nimet, ja pyydettiin haastateltavia kertomaan ovatko he kuulleet järjestelmästä, kokeilleet sitä itse vai jopa käyttäneet sitä opetuksessaan. Vastaukset on koottu kuvaajaan 8.



Kuvaaja 8 Haastateltavien tietoisuus järjestelmistä

Koska on oletettavaa, että järjestelmiä käytetään eniten (ja ne ovat muutenkin kaikkein tutuimpia) niiden ”kotona”, eli siinä korkeakoulussa, jossa ne on kehitetty, laskettiin tietoisuudet uudestaan siten, että jokaisen järjestelmän tuloksista poistettiin kaikki ne vastaajat, jotka työskentelevät siinä korkeakoulussa, jossa järjestelmä on kehitetty. Tulokset on esitetty kuvaajassa 9.

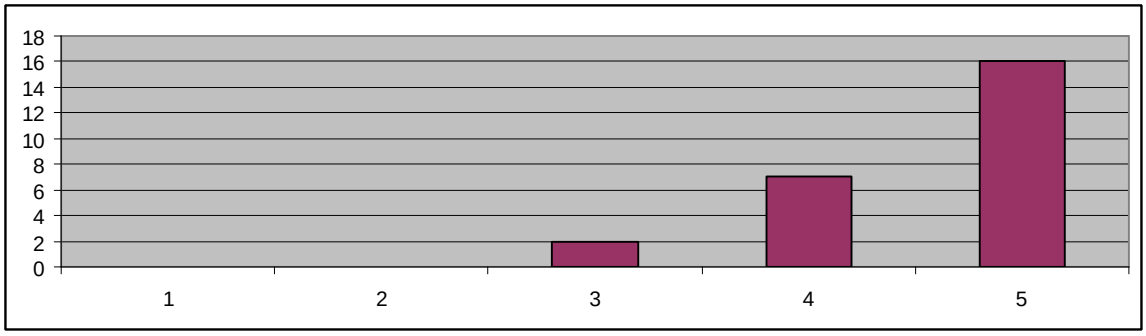


Kuvaaja 9 Tietoisuus järjestelmistä kun "kotietu" on poistettu

4.3.2 Asenteet järjestelmiä kohtaan

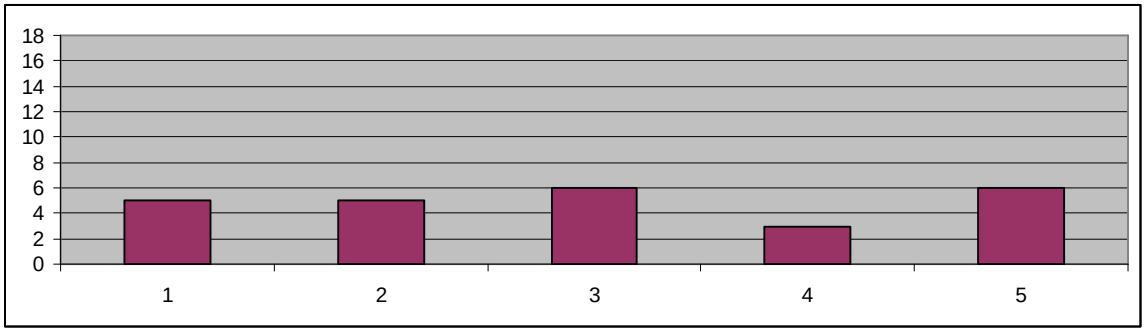
Haastateltavien asenteita järjestelmiä kohtaan selvitettiin yhdentoista järjestelmiä koskevan väittämän avulla. Väitteet liittyivät järjestelmien yleiseen soveltuvuuteen ja toiminnallisuuteen, ja haastateltavat vastasivat niihin asteikoilla yhdestä viiteen (1 – täysin eri mieltä, 3 – eos., 5 – täysin samaa mieltä).

Väittämä 1: *Järjestelmät soveltuvat korkeakoulutason opetukseen (keskiarvo 4,56)*



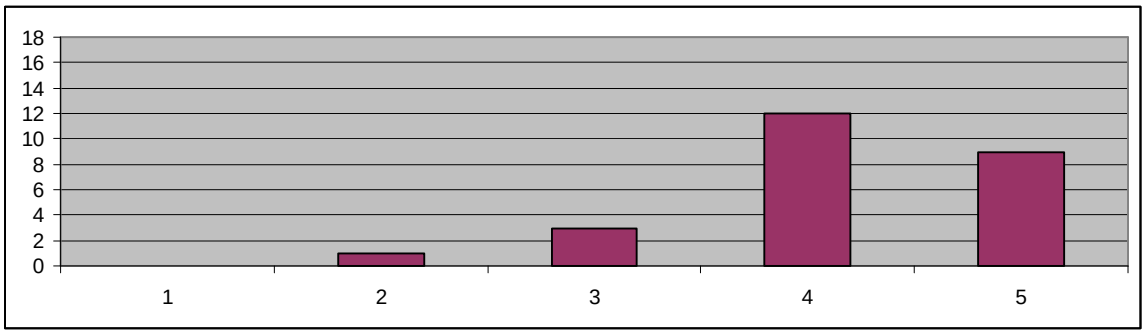
Vastaajat olivat jokseenkin yksimielisiä siitä, että järjestelmät soveltuvat korkeakoulutason opetukseen. Eräs haastateltavista kommentoi, että ”nollataso osaamisessa on sama koulutuksen asteesta riippumatta – järjestelmät siis soveltuvat minkä tahansa asteen koulutukseen”.

Väittämä 2: *Järjestelmien käyttö lisää opettajan työmäärää (keskiarvo 3,00)*



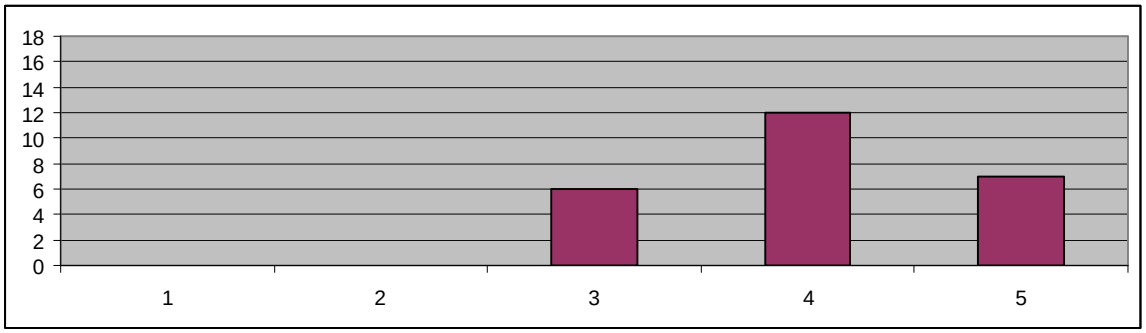
Vastaajien mukaan järjestelmät sekä lisäävät että vähentävät työmäärää, mutta ennen kaikkea ne muuttavat työn luonnetta, ja auttavat suuntaamaan työn fokusta kohti vähemmän triviaaleja tehtäviä.

Väittämä 3: *Järjestelmät tukevat oppimisprosessia (keskiarvo 4,16)*



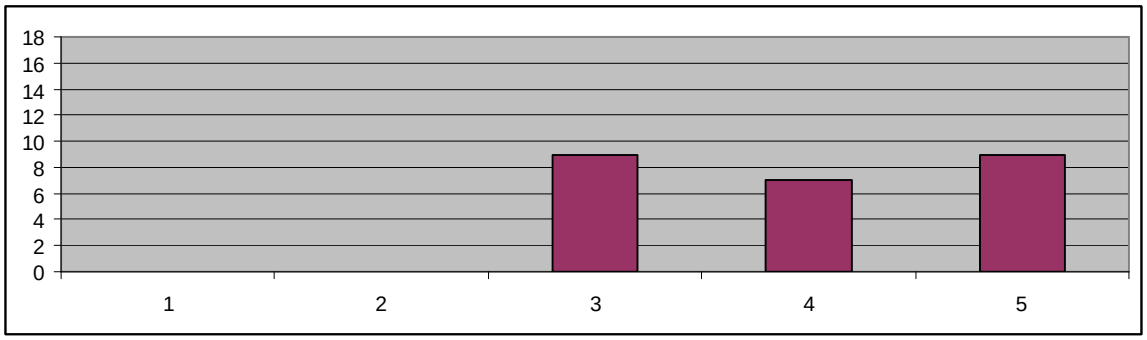
Vastaajien mukaan oikeanlaisten järjestelmien käyttö voidaan nähdä oppimisprosessia tukevana.

Väittämä 4: *Järjestelmät parantavat oppimistuloksia (keskiarvo 4,04)*



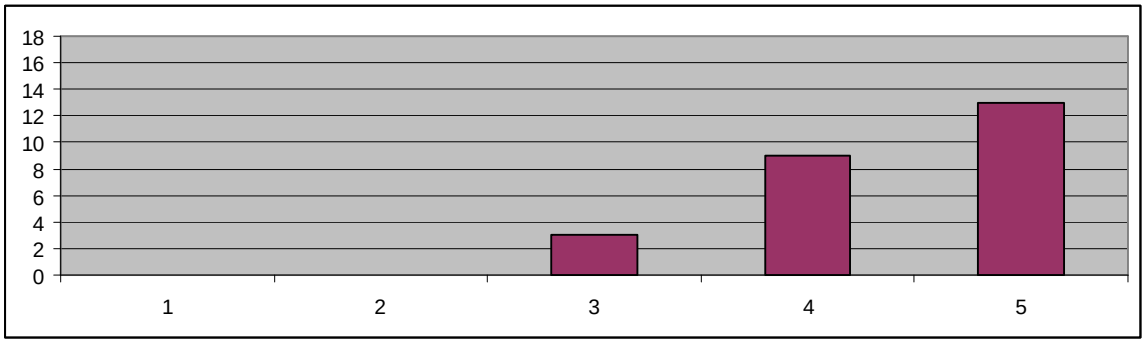
Vastaajilla oli vaihtelevasti kokemuksia järjestelmistä, mutta niitä käyttäneistä opettajista valtaosa oli sitä mieltä, että ne parantavat oppimistuloksia. Esimerkiksi eräs vastaajista kertoi, että ”Jeliot3:n käyttö on parantanut tuloksia huomattavasti, muista järjestelmistä ei ole kokemuksia”.

Väittämä 5: *Järjestelmät lisäävät opiskelumotivaatiota (keskiarvo 4,00)*



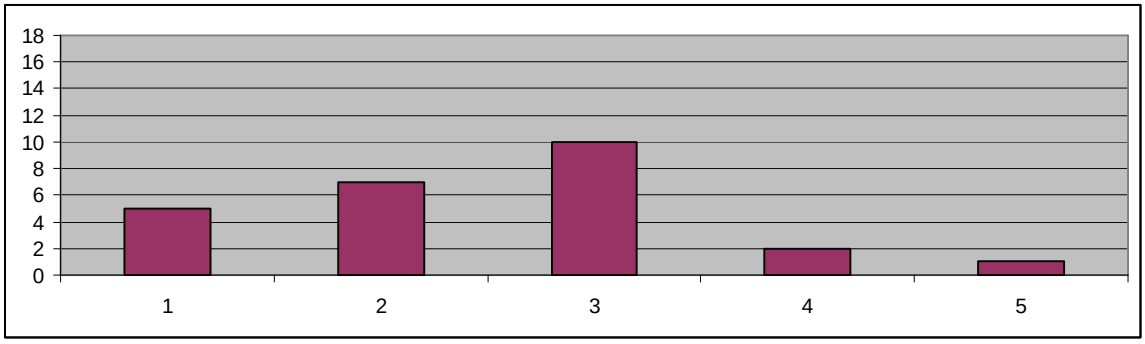
Vastaajat olivat yleisesti sitä mieltä, että järjestelmät parantavat opiskelumotivaatiota, tosin ei välttämättä kaikkien opiskelijoiden kohdalla; eräs haastatelluista kertoi uskovansa, että ”vaikutus opiskelumotivaatioon riippuu opiskelijan tasosta: huonosti menestyvät opiskelijat todennäköisesti pitävät järjestelmän käytöstä vähemmän kuin paremmin menestyvät”.

Väittämä 6: *Järjestelmät monipuolistavat kurssia (keskiarvo 4,4)*



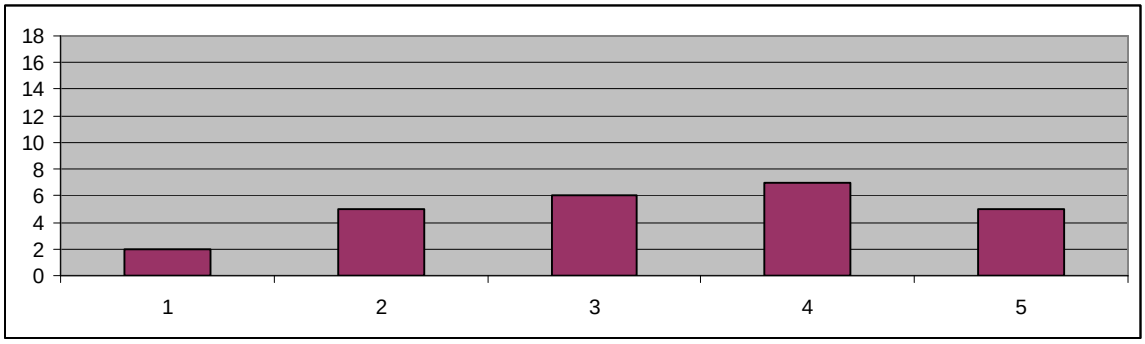
Vastaajien mukaan järjestelmät monipuolistavat kurssia erityisesti siksi, että ne ”mahdollistavat asiat, jotka eivät muuten aika- ja resurssipulan vuoksi olisi mahdollisia”.

Väittämä 7: *Järjestelmät parantavat opettajan ja oppilaan välistä vuorovaikutusta (keskiarvo: 2,48)*



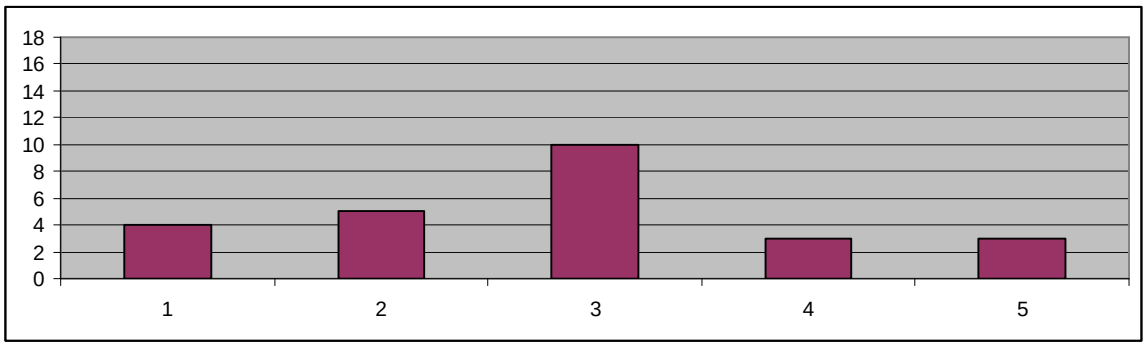
Oletettavaa on, että kaikki järjestelmistä eivät edes voi parantaa vuorovaikutusta. Mahdollisuuden tähän tarjoavat sellaiset palautusjärjestelmät, joiden avulla opiskelija ja opettaja voivat kommunikoida; eräs vastaajista kuitenkin kommentoi tätä kertomalla ettei ”sähköpostin tai vastaavan viestinnän käyttö ei sinänsä vielä edes ole vuorovaikutusta”.

Väittämä 8: *Järjestelmät ovat riittävän helppokäyttöisiä (keskiarvo 3,32)*



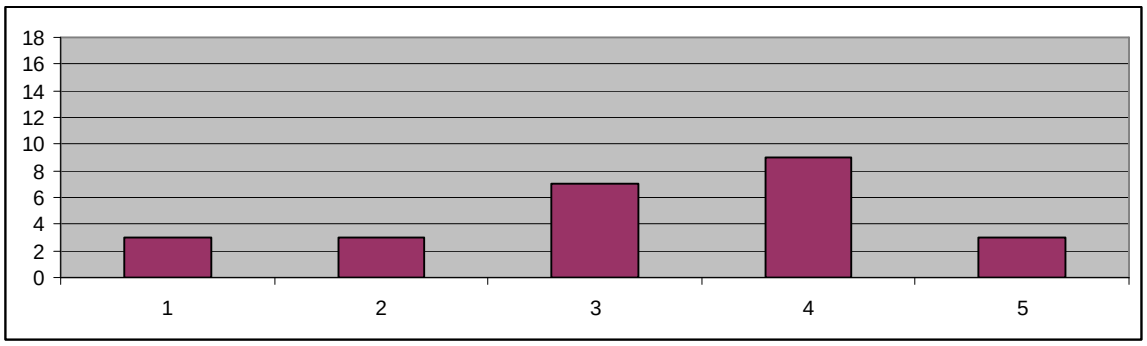
Pääosa vastaajista oli sitä mieltä, että järjestelmien helppokäyttöisyys vaihtelee suuresti. Mielipiteet siitä, minkätyyppiset järjestelmät ovat riittävät helppokäyttöisiä vaihtelivat kuitenkin: eräs haastateltu kertoi, että hänen kokemuksensa mukaan ”Javala on helppokäyttöinen, PlanAni taas ei”. Toisen mukaan ”Goblin on *opiskelijoille* riittävän helppokäyttöinen, kun taas ViLLE ei ole.” Opettajan näkökulmasta WebCatia pidettiin kuitenkin helpompana kuin Goblinia. Lisäksi useampi vastaaja piti kurssien hallintajärjestelmien käyttöä yleisesti liian vaikeana.

Väittämä 9: *Järjestelmien käyttöönotto on riittävän helppoa (keskiarvo 2,84)*



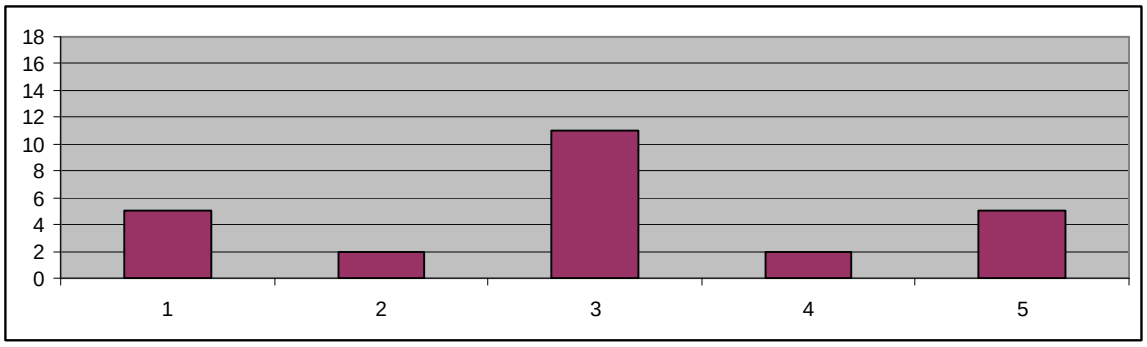
Mielipiteet käyttöönotosta vaihtelivat paljolti sen mukaan, mikä opettajan rooli oli käyttöönotossa ollut, ja millaisia järjestelmiä oli otettu käyttöön. Suhteellisen yleinen mielipide kuitenkin oli, että automaattisen tarkastuksen järjestelmissä käyttöönoton vaiva on suhteellisen suuri, kun taas visualisointijärjestelmien käyttöönotto on melko vaivatonta.

Väittämä 10: *Järjestelmien ominaisuudet ovat riittävät (keskiarvo 3,24)*



Yleinen mielipide oli, että järjestelmien ominaisuudet ovat pääasiassa riittävät, mutta lähes jokaisesta puuttuu jotain omassa käytössä tarpeellista. Lisäksi eräs vastaajista mainitsi, että ”omaa opetusfilosofiaa on joskus vaikea toteuttaa järjestelmien avulla, vaikka ominaisuudet muuten sinänsä olisivatkin riittävät”. Kyse voi siis ominaisuuksien puutteen lisäksi olla vääränlaisista tai väärällä tavalla toteutetuista ominaisuuksista.

Väittämä 11: *Järjestelmien ohjeet ovat riittävät (keskiarvo 3,0)*



Yleisin mielipide ohjeisiin liittyen oli, ettei niitä ole tarvittu. Osa vastaajista oli kuitenkin ehdottoman tyytyväinen ohjeiden määrään ja osa ehdottomasti sitä mieltä, että ohjeita pitäisi olla enemmän, ja niiden laadun pitäisi olla parempi.

4.3.3 Järjestelmien käyttö ja käyttöönotto

Tutkimuksessa mukana olleiden järjestelmien (ks. kuvaaja 8) lisäksi haastatellut ilmoittivat käyttävänsä erinäisiä muita (usein omaa tuotantoa olevia) järjestelmiä. Erityisesti omatekoiset harjoitustehtävien palautusjärjestelmät näyttävät olevan yleisiä: 5 vastaajaa ilmoitti, että heillä on käytössä omassa korkeakoulussa varta vasten tehtävien palautusta varten kehitetty järjestelmä. Osa vastaajista käytti lisäksi tehtävien palautukseen muita tutkimukseen kuulumattomia järjestelmiä, kuten BOSSia tai Moodlea. Tehtävien palautus- ja arviointijärjestelmiä käytettiinkin ylivoimaisesti eniten: kymmenen 26:sta vastaajasta ilmoitti käyttävänsä jonkinlaista apuvälinettä palautukseen ja arviointiin. Visualisointijärjestelmien luentokäyttö sen sijaan oli harvinaista: MatrixPro, Jeliot3, ViLLE ja VIP olivat kukin ainoastaan yhden haastatellun käytössä luentojen yhteydessä; itsenäisen opiskelun apuvälineenä niitä kuitenkin käytettiin useammin. Luentokäytössä suurin osa vastaajista kertoi, että perinteinen ”tussilla kalvoille”-visualisointi täyttää kaikki heidän tarpeensa, eikä muutokselle sinänsä ole tarvetta.

Mukana olleista järjestelmistä haastateltavat kertoivat ottavansa eri tarkoituksiin käyttöön (joko menossaolevalla tai tulevalla kurssillaan) seuraavat:

GOBLIN (3 haastateltavaa): Käyttöön helpottamaan tehtävien palautusta ja automaattista arviointia.

Jeliot3 (3 haastateltavaa): Käyttöön itsenäisen opiskelun avuksi ja luennoille; eräs haastateltavista kertoi, että on käyttänyt järjestelmää aikaisemminkin, mutta jättänyt sen sitten (ei-määritellyistä syistä) pois käytöstä.

ViLLE (5 haastateltavaa): Käyttöön sekä itsenäisen opiskelun avuksi että luennoille. Eräs haastateltava mainitsi, että ViLLE ”kuulostaa hyvältä alkuhieromiseen”. Muita syitä käyttöönotolle olivat mm. se, että ViLLE ”helpottaa siirtymistä kielestä toiseen” ja ”selkeyttää funktion ja rekursion käsitettä”. Toisaalta eräs haastateltavista kertoi, että oli kyllä koekäyttänyt ViLLEä, mutta ei halunnut ottaa sitä käyttöön.

TRAKLA2 (2 haastateltavaa): Käyttöön erityisesti itsenäisen opiskelun avuksi. Syiksi mainittiin erityisesti se, että järjestelmän avulla voidaan ”korvata osa demonstraatiotehtävistä helpommin [automaattisesti] tarkastettavilla”. Eräs vastaajista kertoi lisäksi harkitsevansa TRAKLA2:n käyttöönottoa, muttei ollut vielä varma päätöksestään.

PlanAni (2 haastateltavaa): Käyttöön erityisesti luennoilla; lisäksi toinen haastateltavista aikoi käyttää järjestelmää myös itsenäisen opiskelun apuna.

Lisäksi useat haastatelluista kertoivat harkitsevansa jonkinlaista palautusjärjestelmää. Haastatteluun kuulumattomista järjestelmistä käyttöön oltiin ottamassa muunmuassa visualisointijärjestelmät Animal ja BlueJ.

4.3.4 Miksi järjestelmät jätetään ottamatta käyttöön?

Haastattelussa pyrittiin lisäksi määrittelemään syitä, joiden takia he ovat jättäneet järjestelmän tai järjestelmiä ottamatta käyttöön. Seuraavassa on esitetty haastattelusta

poimittuja syitä järjestelmien käyttämättä jättämiseen; kunkin väittämän perässä on lisäksi esitetty, liittyykö se tarkastusjärjestelmiin, visualisointijärjestelmiin, vai sekä-että.

Kurssin käytäntöihin liittyvät syyt:

- Kurssin opetuskäytäntöihin liittyviä syitä olivat esimerkiksi se, että kurssi oli haastateltavan mielestä ”liian kevyt” järjestelmiä varten: siihen ei nykymuodossaan sisältynyt ollenkaan demonstraatioita, eikä niihin ollut tarvetta tulevaisuudessakaan. *(visualisointi)*

- Tulevat käytännön muutokset kurssissa, esimerkiksi toiseen korkeakouluun yhdistymisen takia, vaikuttivat siihen, että haastateltujen mielestä nyt ei ”ollut aika tehdä muutoksia kurssin sisältöihin tai käytäntöihin”. *(järjestelmät yleisesti)*

- Järjestelmät on haastatellun mielestä suunnattu lähinnä aloittelijoille, jatkokursseilla niiden hyöty on vähäisempi. *(visualisointi)*

Järjestelmien sisällölliset puutteet ja epäkohdat:

- Erään haastatellun mielestä järjestelmät ovat ”liian tarkkoja syntaktisista seikoista (esim. puolipisteen käyttö), eivätkä ne näin ollen kykene erottamaan todellista ymmärrystä muotoseikoista”. *(järjestelmät yleisesti)*

- ”Automaattinen palaute on aina huonompaa kuin ihmisen antama” ja tähän liittyen ”ohjelmointityyliä on mahdotonta arvostella automaattisesti”. Lisäksi palautetta pidettiin vaikeaselkoisempana. *(tarkastusjärjestelmät)*

- Usean tietorakennekursseja opettavan haastateltavana mielestä suurin osa järjestelmistä (TRAKLA2:a lukuunottamatta) on suunnattu ohjelmointikurssien tarpeisiin, eikä niillä ole tietorakenteiden ja algoritmien kursseilla vastaavaa käyttöä. *(järjestelmät yleisesti)*

- Järjestelmät ovat lisäksi jotain muuta opetuskieltä käyttävien mielestä liian fokusoituneita Javaan. *(järjestelmät yleisesti)*

- Järjestelmistä ei löydy algoritmista arviointia (esimerkiksi suoritusajan arviointi). *(tarkastusjärjestelmät)*

Järjestelmien käytön vaikeuteen liittyvät syyt:

- Erään haastateltavan mukaan ”järjestelmät todennäköisesti entisestään monimutkaistavat [muutenkin vaikeita] asioita helpottamisen sijasta” (*visualisointi)*

- Toinen haastateltava kertoi, että ”suurin ongelma on se, että järjestelmien käyttö on liian vaikeaa: suurimmasta osasta en saa itsekään tolkkua” *(järjestelmät yleisesti)*

- Lisäksi vaikka ”järjestelmät ovat opettajalle kyllin helppokäyttöisiä”, eivät opiskelijat saa niistä haastateltavan mukaan mitään tolkkua. *(järjestelmät yleisesti)*

Opiskelijoihin liittyvät syyt:

- Erään haastateltavan mukaan ”opiskelija-aines ei nykyisen valintajärjestelyn takia omaa riittävästi matemaattisia taitoja edes nykyisestä kurssimateriaalista selviämiseen.” *(visualisointi)* Lisäksi ”opiskelija-aines on liian kirjavaa yhden tietyn järjestelmän käyttöönottamiseksi.” *(järjestelmät yleisesti)*

- Toinen haastateltava kertoi uskovansa, että ”liiallinen luottamus testausautomaattiin johtaa siihen, etteivät opiskelijat opettele itse muodostamaan testiaineistoa.” *(tarkastusjärjestelmät)*

- Eräs haastateltavista mainitsi, että ”kysymykset aliarvioivat oppijaa; lisäksi merkintätavat eivät ole yleisten käytäntöjen mukaisia”. *(visualisointijärjestelmät)*

Järjestelmien käyttämiin menetelmiin liittyvät syyt:

- Erään haastateltavan mukaan ”ihimillisen palautteen korvaaminen on järjestelmästä riippumatta todella hankalaa”. *(järjestelmät yleisesti)*
- Toinen haastateltava piti menetelmiä vääränlaisina: ”järjestelmät eivät visualisoi ohjelmia sellaisilla tavoilla, joita itse pitäisin havainnollisimpina”. *(visualisointijärjestelmät)*

Muita syitä:

- Erään haastatellun mukaan se, että järjestelmä ei ole omaa tuotantoa, on riittävä syy jättää se ottamatta käyttöön. *(järjestelmät yleisesti)*

Toisaalta esimerkiksi sen, ettei oppimistuloksista ole todisteita, ei katsottu olevan yleisesti kovin merkittävä syy jättää järjestelmä ottamatta käyttöön. Erään haastatellun sanoin ”oma intuitio on järjestelmää valittaessa kuitenkin tärkein syy”.

4.3.5 Millaisia järjestelmiä opettajat kaipaavat?

Haastattelussa pyrittiin lisäksi selvittämään, ovatko opettajat jollain opetuksen osa-alueella jossain vaiheessa kaivanneet käyttöönsä järjestelmää helpottamaan tai automatisoimaan jotain toimintaa, mutta eivät ole tällaista järjestelmää kuitenkaan löytäneet. Toiveet on jaettu seuraavassa kategorioihin.

Luentokäyttö:

- Järjestelmä vuokaavioiden animoimiseksi luennoilla.
- Visualisointijärjestelmä aliohjelmien ja niiden parametrien sekä taulukoiden visualisoimiseen.

- Piirtoalusta, jolla voisi kätevästi kirjoittaa PowerPoint-esitysten päälle.
- Järjestelmä, joka toisi interaktiivisuutta luentoihin; esimerkiksi mobiiliäänestysjärjestelmä, jonka avulla voisi helposti selvittää, kuinka suuri osa opiskelijoista oikeasti ymmärsi esitetyn asian.

Harjoitustehtävien palautus ja tarkastus:

- Oikeasti toimiva automaattisen palautuksen järjestelmä.
- Järjestelmä, joka tarkastaisi automaattisesti palautetusta ohjelmasta sen, onko se ”järkevästi” tehty.
- Automaattisen tarkastuksen järjestelmä, jossa testaus olisi nykyjärjestelmiä paremmin toimiva.
- Järjestelmä, joka saisi automaattisesti palautetuista tehtävistä vilppiyritykset kiinni.
- Järjestelmä Python-kielisten ohjelmien palautukseen, automaattiseen tarkastukseen ja testaukseen.

Kurssien ja tehtävien hallinta:

- Järjestelmä resurssien jakamiseen opiskelijoiden kesken (esimerkiksi harjoitustyöaiheet, assistenttien tapaamisajat jne.)
- Järjestelmä opiskelijoiden organisoimiseksi (pitäen sisällään esimerkiksi opiskelijan kanssa tehdyt erityissopimukset); luonnollinen toteutus käytännössä olisi esimerkiksi liitännäinen sähköpostiohjemaan.
- Järjestelmä helpottamaan ilmoittautumisten hallintaa.

Tentit:

- Konetentit.
- Järjestelmä tenttien tarkastuksen helpottamiseen ja jopa automatisoimiseen.

Muita järjestelmiä ja keinoja:

- Järjestelmä, joka helpottaisi siirtymistä Javasta Scalaan.
- Valmis kurssimateriaali kokonaisuudessaan.
- Järjestelmä lopputöiden ohjaamisen apuvälineeksi.
- Opiskelijoille nykyistä helpompi tapa asentaa käännösympäristö.

4.4 Kommentteja opetuksesta ja järjestelmistä

Haastattelun lopuksi haastateltaville annettiin mahdollisuus esittää anonyymisti kommentteja opetukseen ja järjestelmiin liittyen. Seuraavaan on koottu näitä kommentteja, samoin kuin sellaisia haastattelussa esille nousseita asioita, jotka eivät suoranaisesti liity aikaisemmin esitettyihin kysymyksiin.

4.4.1 Ajatuksia opetuksesta ja oppimisesta

- ”Ajattelun pitäisi olla kaikkein tärkein asia – opetusjärjestelmien vaarana on, että ne voivat mekanisoida toimintaa, jollon ajattelu jää toissijaiseksi”.
- ”Oppimisen pitäisi aina olla päällimmäinen ajatus; nykyisellään tuntuu, että järjestelmät ovat enemmän itseisarvoja kuin välineitä oppimiseen”
- ”Perusasiat (valinta- ja toistorakenne yms.) ovat tärkeimmät asiat. Opetuksen pitäisi aina lähteä niistä liikkeelle”
- ”Olen tyytyväinen siihen, että algoritmien opetuksessa on nykyisellään siirrytty teoriasta kohti käytännön ohjelmointitoteutusta.”
- ”Opetuksessa pitäisi enemmän hyödyntää konkreetisia välineitä ja asioita. Tällaisten keinojen ja ideoiden vaihtoa varten pitäisi olla olemassa jokin foorumi”

- ”Algoritmeja ei pidä opettaa liian formaalisti”

- ”Ensimmäisen ohjelmointikurssin pitää olla kokonaisuudessaan laitoksella itse tuotettu, jotta opiskelijat saadaan alusta alkaen sitoutettua laitoksen käytäntöihin.”

- ”Tärkeintä on liittää ohjelmoinnin opetus muuhun opetukseen.”

- ”Ohjelmointia oppii vain ohjelmoimalla: jos koko kurssilla tehdään vain 1-2 lyhyttä harjoitustyötä, ei ohjelmointirutiinia edes voi syntyä.”

4.4.2 Ajatuksia opiskelijoista

- ”Opiskelijoiden algebran taidot ovat nykyään suorastaan ala-arvoiset.”

- ”Opiskelijoiden motivaatiovaikeudet ohjelmoinnin opetuksessa ovat todellinen ongelma, koska ilman ohjelmointitaitoa ei mikään onnistu työelämään siirryttäessä.”

- ”Opiskelijat yrittävät aina huijata järjestelmää, mikäli tämä on vain jotenkin mahdollista.”

- ”Opiskeljat ovat yksinkertaisia: yhdellä järjestelmällä pitää hoitaa vain yksi asia.”

4.4.3 Ajatuksia järjestelmistä

- ”Järjestelmien pitäisi olla helppokäyttöisempiä, samoin dokumentoinnin pitäisi olla nykyistä kattavampi.”

- ”Yleissuhtautumiseni järjestelmiin on joka tapauksessa positiivinen, edellyttäen, että ne ovat riittävän helppokäyttöisiä.”

- ”Järjestelmän ensimmäinen vaatimus on, että sen tulee olla riittävän helppokäyttöinen ja motivoiva opettajan näkökulmasta.”

- ”Järjestelmien käyttöönoton suurin este on yleensä se, etteivät ne sovi sisällöllisesti omalle kurssilleni.”

- ”Järjestelmät ovat varmasti päteviä, mutta ne sopinevat parhaite toisenlaisille kursseille.”

- ”Järjestelmät julkaistaan yleensä keskeneräisinä. Lisätyö erityisesti ulkoasun parissa olisi tärkeää ennen jakelua.”

- ”Järjestelmien suurin hyöty on se, että niiden avulla saadaan huijattua opiskelijat käyttämään enemmän aikaa aiheen parissa.”

5 Lähdeluettelo

Ahoniemi, T. & Reinikainen, T. 2007. ALOHA - a grading tool for semi-automatic assessment of mass programming courses. Teoksessa Proceedings of the 6th Baltic Sea Conference on Computing Education Research, s. 139-140.

Edwards, S. 2006. What is WebCat – Web-CAT Wiki. <http://web-cat.cs.vt.edu/WCWiki/WhatIsWebCat> 26.6.2008.

Hiisilä, A. Kurssinhallintajärjestelmä ohjelmoinnin perusopetuksen avuksi. 2005. Diplomityö, Teknillinen korkeakoulu, Suomi.

Joy, M., Griffiths, N. & Boyatt, R. 2005. The BOSS Online Submission and Assessment System. ACM Journal on Educational Resources in Computing, 5(3).

Karavirta, V., Korhonen, A., Malmi, L. & Stalnacke, K. 2004. MatrixPro - A Tool for Demonstrating Data Structures and Algorithms Ex Tempore. Teoksessa Proceedings of the IEEE International Conference on Advanced Learning Technologies (ICALT'04), s. 892-893

Lehtonen, T. 2005. Javala – Addictive E-Learning of the Java Programming Language. Teoksessa Proceedings of Kolin Kolistelut / Koli Calling – Fifth Annual Baltic Conference on Computer Science Education. Joensuu, Finland, 41–48.

Moreno, A., Myller, N., Sutinen, E. & Ben-Ari, M. 2004. Visualizing Programs with Jeliot 3. Teoksessa Proceedings of the Working Conference on Advanced Visual Interfaces (AVI 2004), Gallipoli (Lecce), Italy. ACM Press, New York, 373-380.

Myller, N., Laakso, M.-J. & Korhonen, A. 2007. Analyzing engagement taxonomy in collaborative algorithm visualization. Teoksessa Proceedings of the 12th annual SIGCSE conference on Innovation and technology in computer science education. ACM, s. 251–255.

Rajala, T., Laakso, M.-J., Kaila, E. & Salakoski, T. 2007. ViLLE – A language independent program visualization tool. Teoksessa Proceedings of the Seventh Baltic Sea Conference on Computing Education Research (Koli Calling 2007), Koli National Park, Finland. Conferences in Research and Practice in Information Technology, 88, Australian Computer Society. Raymond Lister & Simon (toim).

Saikkonen, R., Malmi, L. & Korhonen, A. 2001. Fully automatic assessment of programming exercises. Teoksessa Proceedings of the 6th annual conference on Innovation and technology in computer science education, Canterbury, U-K, s. 133-136

Sajaniemi, J. 2006. Teaching Roles of Variables.
<http://www.cs.joensuu.fi/~saja/var_roles/teaching.html> 26.6.2008.

Virtanen, A., Lahtinen, E. & Järvinen, H.-M. 2005. VIP, a visual interpreter for learning introductory programming with C++. Proceedings of The Fifth Koli Calling Conference on Computer Science Education, s. 125-130.