JOENSUUN YLIOPISTO
TIETOJENKÄSITTELYTIETEEN LAITOS
Raporttisarja A

PUTTE – Puutteiden estimointijärjestelmä

Matti Niemi

Report A-2002-8

ACM D.2.
ISSN 0789-7316
ISBN 952-458-219-8

PUTTE – Puutteiden estimointijärjestelmä

Matti Niemi

Tietojenkäsittelytieteen laitos
Joensuun yliopisto
PL 111, 80101 Joensuu

Tiivistelmä

Puutteiden estimointijärjestelmä, PUTTE, on tietokantapohjainen ohjelmistoapuväline, joka on tehty mallintamaan ohjelmistotuotantoprojekteissa kerättyjen puutetietojen lukumäärän pohjalta ohjelmistotuotantoprosessissa ilmeneviä puutteita. Järjestelmän avulla voidaan tuottaa aikaisempien, samankaltaisten ohjelmistoprojektien historiatietoja käyttäen Norden/Rayleigh- ja Gamma-jakaumien mukaisesti puutteiden ilmenemismalli, jonka perusteella voidaan arvioida käynnistyvää projektia sekä dynaamisesti jo käynnissä olevaa projektia.

Avainsanat: Gamma-malli, Norden/Rayleigh-malli, puutteiden lukumäärän estimointi, ohjelmistoapuväline

1 Johdanto

Ohjelmistotuotannossa luotettavuuden mallintamista voidaan tarkastella testausvaiheen osalta tai koko ohjelmistokehityksen elinkaaren osalta. Tässä raportissa kuvattava puutteiden estimointijärjestelmä (PUTTE) mahdollistaa luotettavuuden mallintamisen vaiheistukseen perustuvan koko kehitysprosessin osalta. Koska järjestelmällä laadittava malli riippuu ajallisesti peräkkäisistä vaiheista, voidaan mallia pitää dynaamisena. PUTTE-järjestelmä mahdollistaa kuitenkin kahden tyyppisiä dynaamisia malleja. Historiatiedoista laskettava elinkaarta tukevaa mallia kutsutaan tässä raportissa pelkästään *perusmalliksi*. Käynnissä olevan projektin tiedoilla täydennettyä historiatiedoista laskettua mallia kutsutaan *dynaamiseksi malliksi*. Dynaamisuudella tarkoitetaan tässä raportissa siis mallin muuttamista projektin aikana.

Järjestelmän avulla sovitetaan Norden/Rayleigh-jakauman (Putnam, 1978) ja Gamma-jakauman (Pillai ja Nair, 1997) mukaiset käyrät pienimmän neliösumman menetelmän ja logaritmimuunnoksen avulla havaintoaineistoon sekä lasketaan mallin hyvyyttä kuvaavia tunnuslukuja näin saaduille malleille (Pillai ja Nair, 1997): ennustevinoutuma (prediction bias), ennustevaihtelu (prediction variation) ja RMSPE (Root Mean Square Prediction Error). Oletuksena siis on, että havaintoaineisto eli ohjelmistotuotantoprojekteista kerätyt puutetiedot noudattavat mainittuja jakaumia.

Luvussa 2 esitetään järjestelmän arkkitehtuuri toiminnallisuuden ja tietosisällön osalta. Luvussa 3 tarkastellaan tarkemmin järjestelmän toiminnallisuutta ja esitetään toiminnallisuuden kannalta tärkeimmät algoritmit. Luvussa 4 todennetaan järjestelmän mallinnusalgoritmien toteutuksen oikeellisuutta tunnuslukujen avulla soveltamalla laskentaa erään esimerkkitapauksen malliin ja siitä laskettuun malliin.

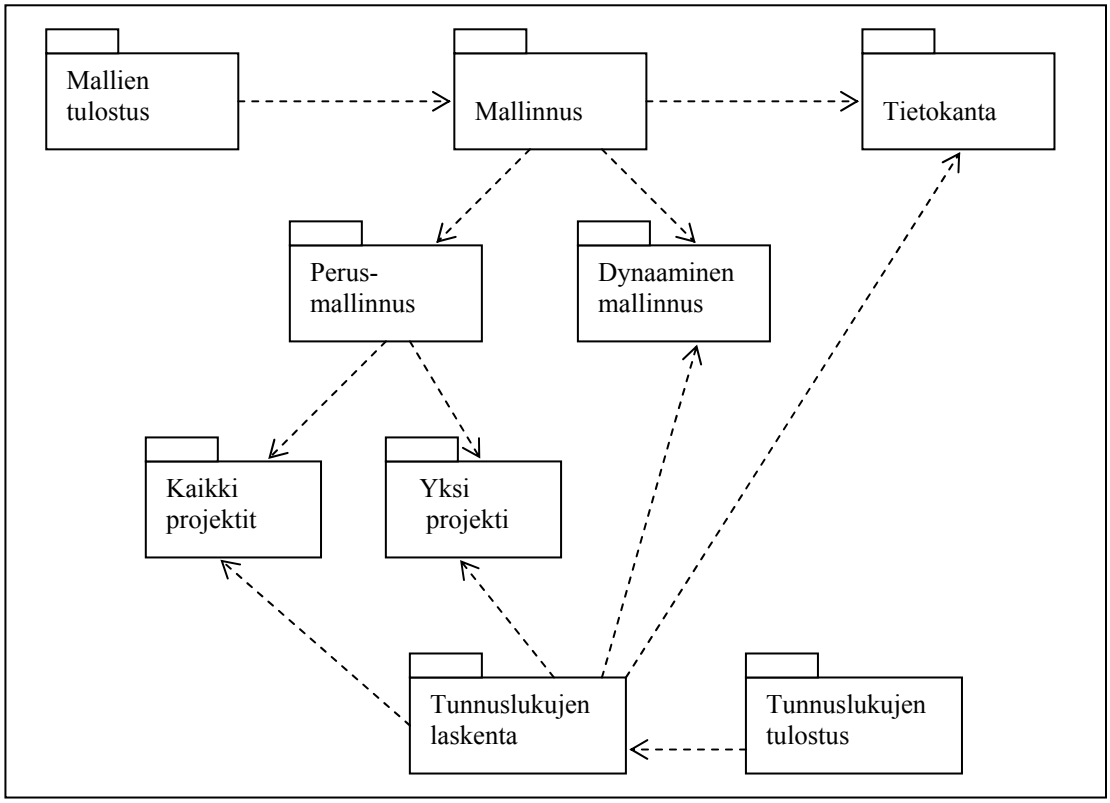
2 Järjestelmän arkkitehtuuri

2.1 Yleisrakenne

Järjestelmä jakaantuu toiminnallisesti Visual Basic –kielellä toteutettuun kahteen perusosaan: laskennallinen osa ja tulostusosa. Laskennallisessa osassa suoritetaan mallintamiseen vaadittavat laskutoimitukset sekä näin saadulle mallille suoritettavat tunnuslukujen laskutoimitukset. Tulostusosassa näytölle ohjataan laskennan tuloksena saadut mallit sekä näiden tunnusluvut. Järjestelmän yleisrakenne on esitetty kuvassa 1 UML-notaation mukaisten pakettien ja niiden välisten riippuvuuksien avulla[1].

Laskennallisessa osassa suoritettava mallinnus jakaantuu perusmallinnukseen ja dynaamiseen mallinnukseen. *Perusmallinnuksessa* oletetaan olevan *N* projektia. Järjestelmän avulla voidaan laatia malli kaikista *N* projektista ja laskea sitten tunnusluvut mallin hyvyydelle. Toinen mahdollisuus on muodostaa malli *N*-1 projektin avulla ja tutkia mallin soveltuvuutta malliin kuulumattomaan projektin osalta. *Dynaamisessa mallinnuksessa* hyödynnetään vanhaa *N*-1 projektista laskettua mallia ja käynnissä olevan projektin puutetietoja. Käynnissä olevalle projektille lasketaan uusi malli ajankohdassa *t* yhdistämällä projektissa ajankohtaan *t* mennessä kertyneet puutetiedot vanhan mallin ilmaisemiin puutetietoihin. Molemmissa mallinnustavoissa sovelletaan Norden/Rayleigh-jakaumaa ja Gamma-jakaumaa.

Tietokanta sisältää tiedot projekteista sekä tiedot lasketuista malleista. Tulosteina näytölle saadaan laskettuja malleja kuvaavat käyrät sekä tunnusluvut.

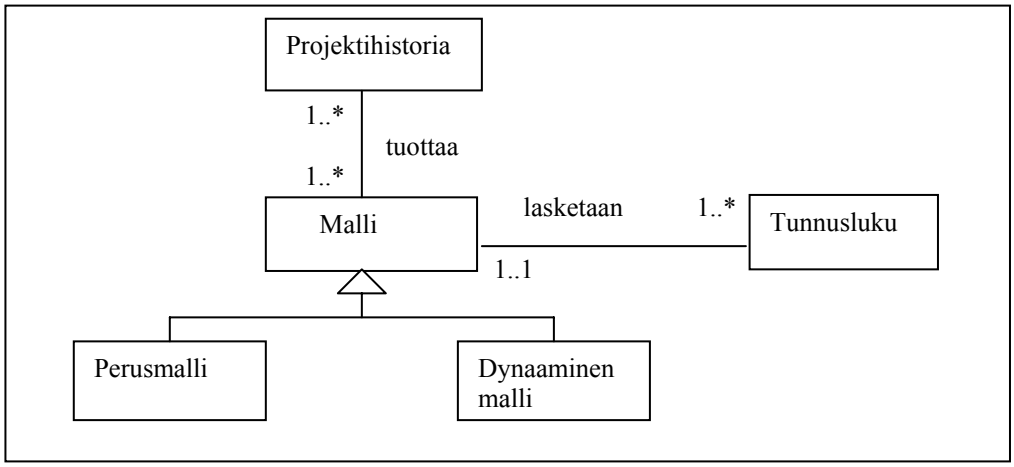


Kuva 1. Järjestelmän yleisrakenne.

[1] UML (Unified Modeling Language) on Boochin et al. (1999) kehittämä mallinnuskieli.

2.2 Tietokannan rakenne

Kuvassa 2 on kuvattu järjestelmän sisältämien tietojen looginen rakenne UML-mallinnuskielen mukaisena luokkakaaviona. Kustakin projektista tallennetaan puutteiden lukumäärä kussakin projektin vaiheessa. Näistä projektitiedoista tuotetaan malleja, joille kullekin lasketaan tunnuslukuja. Tuotettu malli voi kuvan 2 mukaisesti olla joko perusmalli tai dynaaminen malli kohdassa 2.1 selitetyllä tavalla. Tunnuslukuja ei tallenneta fyysisesti tietokantaan.



Kuva 2. Järjestelmien sisältämien tietojen looginen rakenne.

Fyysisenä tietokantana käytetään MS Access -tietokantaa Projektit.mdb, joka sisältää tietokantataulut Projektihistoria ja LaskentaArvot, jotka on esitetty kuvissa 3 ja 4 SQL-kielen mukaisilla luontilauseilla.

Kuvan 3 Projektihistoria-tauluun tallennetaan manuaalisesti tai järjestelmän ulkopuolisella järjestelmällä tiedot toteutuneista projektien puutetiedoista. Kukin projekti yksilöidään yksikäsitteisen projektinumeron avulla ja kustakin projektista tallennetaan yhdelle riville puutetiedot *T*:ssä ajankohdassa. Kukin ajankohta voidaan tulkita tietyn vaiheen päättymiseksi. Ajankohdat ovat siis ajallisesti nousevassa järjestyksessä.

```
CREATE TABLE Projektihistoria(
            projektinumero NUM PRIMARY KEY,
            ajankohta_1  FLOAT NOT NULL DEFAULT 0,
            ajankohta_2  FLOAT NOT NULL DEFAULT 0,
            …
            ajankohta_T  FLOAT NOT NULL DEFAULT 0
            )
```

Kuva 3. Projektihistoria -taulu.

Kuvan 4 LaskentaArvot-tauluun järjestelmä tallentaa puutetiedoista laskettujen mallinnustiedot, joita käytetään tunnuslukujen laskemisessa. Yhdelle riville lasketaan tietyssä ajan-

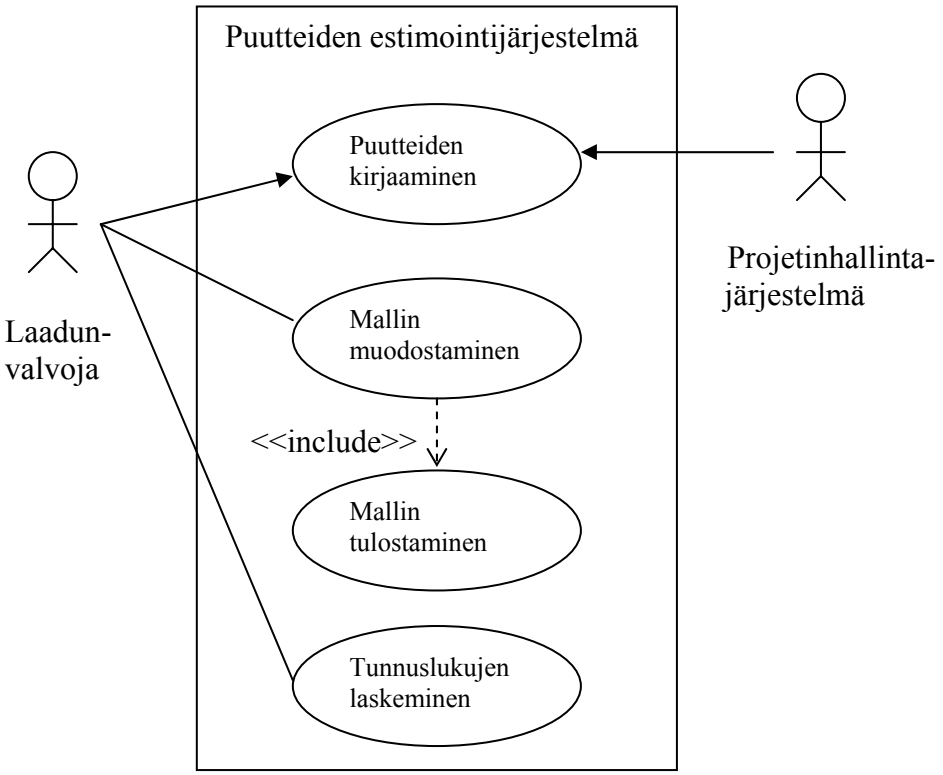
kohdassa *t* tarvittavat tiedot: perusmallille laskettu arvo, dynaamiselle mallille laskettu arvo, perusmallin ja dynaamisen mallin arvojen erotus, puutteiden lukumäärän ja perusmallin välinen erotus sekä puutteiden lukumäärä ja dynaamisen mallin välinen erotus.

```
CREATE TABLE LaskentaArvot(
        ajankohta NUM  PRIMARY KEY,
        puutteidenLkm FLOAT,
        perusmalli FLOAT,
        dynaaminenMalli FLOAT,
        perusmalliMiinusDynaaminenMalli FLOAT,
        puutteidenLkmMiinusPerusmalli FLOAT,
        puutteidenLkmMiinusDynaaminenMalli FLOAT
        )
```

Kuva 4. LaskentaArvot -taulu.

3 Toiminnallisuus

Puutteiden estimointijärjestelmän perustoiminnot on esitetty kuvassa 5 UML-mallinnuskielen mukaisena käyttötapauskaaviona. *Puutteiden kirjaamisen* vaatimaa toiminnallisuutta ei ole ohjelmoitu nykyiseen järjestelmäversioon, vaan se pitää suorittaa manuaalisesti suoraan tietokantaan. Periaatteessa voisi olla mahdollista saada puutetiedot tietokantaan myös ulkopuolisen järjestelmän kautta. Tällainen järjestelmä voisi olla esimerkiksi projektien suunnitteluun ja valvontaan käytettävä projektinhallintajärjestelmä.

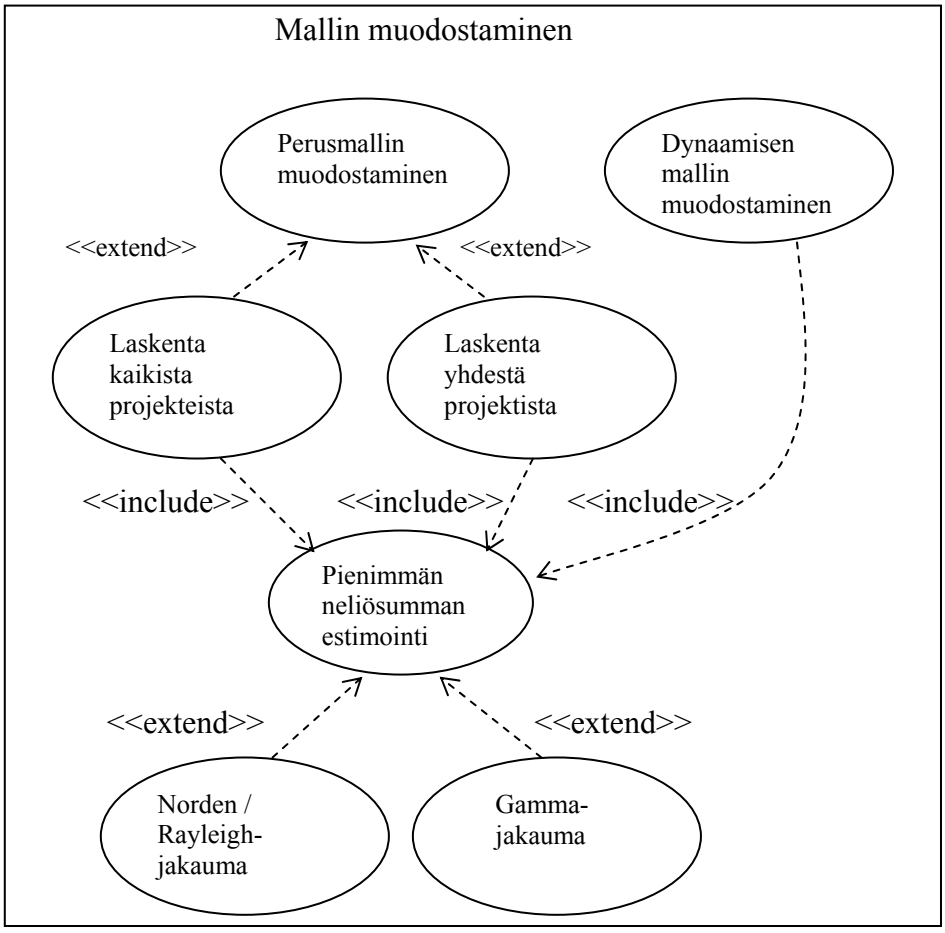


Kuva 5. Puutteiden estimointijärjestelmän käyttötapauskaavio.

Varsinaisia järjestelmän perustoimintoja ovat siis Mallin muodostaminen, johon voidaan katsoa sisältyväksi Mallin tulostamisen, ja Tunnuslukujen laskeminen. Näitä perustoimintoja vastaavat käyttötapaukset on tarkennettu kuvissa 6 – 8.

Järjestelmän käyttäjä, joka kuvassa 5 on nimetty roolilla Laadunvalvoja, voi valita valikosta ylimmällä tasolla joko toiminnon Mallin muodostaminen tai Tunnuslukujen laskeminen. Seuraavalla tasolla kuvan 6 mukaan järjestelmän käyttäjä voi valita joko Perusmallin muodostamisen tai Dynaamisen mallin muodostamisen.

Perusmallin muodostaminen voidaan tehdä kaikista *N* projektista tai vain yhdestä projektista. Jos se tehdään kaikista projekteista, järjestelmä lukee tietokannan Projektit.mdb taulusta Projektihistoria kaikkien vanhojen projektien puutemäärät ja muodostaa puutetietojen perusteella *pienimmän neliösumman menetelmällä* (Spiegel, 1961) perusmallin, jonka tallentaa tauluun LaskentaArvot. Vastaava laskenta voidaan tehdä myös yhdestä projektista, jonka järjestelmän käyttäjä valitsee valintalistasta. Kummassakin tapauksessa käyttäjä voi vielä valita mallinnustavan: Norden/Rayleigh- tai Gamma-jakauma.



Kuva 6. Käyttötapauksen ”Mallin muodostaminen” tarkennus.

Dynaamisen mallin muodostamiseksi järjestelmä muodostaa ensin Projektihistoria-taulun *N*-1 projektin tietojen perusteella perusmallin. Yksi Projektihistoria-taulun projekteista on ”käynnissä oleva” projekti, joka jätetään perusmallista pois käyttäjän valinnan perusteella. Käyttäjävalinnalla ilmaistaan myös ajankohta *t*, johon käynnissä oleva projekti on edennyt.

Järjestelmä yhdistää perusmallin ja käynnissä olevan projektin puutetiedot ja laskee niiden perusteella uuden mallin käynnissä olevalle projektille. Uusi malli on ajankohdasta riippuvainen ja tästä syystä dynaaminen projektin edetessä. Dynaaminen malli voidaan tuottaa sekä Norden/Rayleigh- että Gamma-jakauman mukaisesti pienimmän neliösumman menetelmällä. Järjestelmän toteuttama pienimmän neliösumman menetelmä on algoritmin 1 mukainen kaavan (1) Norden/Rayleigh-jakaumalle. Algoritmi 2 esittää vastaavan laskennan kaavan (2) Gamma-jakaumalle.

$$p(t) = 2Kate^{-at^2} \tag{1}$$

$$f(t \mid \alpha, \beta) = 4K / t_d^3 t^2 e^{-2/t_d} \tag{2}$$

Algoritmi 1. Norden/Rayleigh-mallin laskenta pienimmän neliösumman menetelmällä.

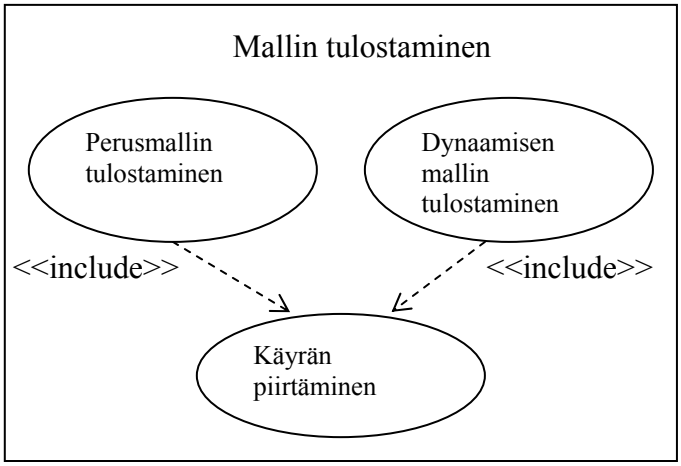
```
// Mallin muodostaminen kaikista projekteista.
// Pisteparien (t,v) lukumäärä m, missä t on ajanjakso ja v on
// puutteiden lukumäärä.
// Ajanjaksojen lukumäärä T.
Muodosta tietuejoukko P taulun Projektihistoria puutetiedoista;
m := 0; sx := 0; sy := 0;
for t := 1 to T do
    Siirry tietuejoukon P alkuun projektiin p;
    while tietuejoukossa P on alkioita do
        m := m+1;
        v := puutteiden lukumäärä projektissa p ajanhetkellä t;
        y := ln(v/t);
        x := t·t;
        sy := sy+y;
        sxx := sxx+x·x;
        sx := sx+x;
        sxy := sxy+x·y;
    end while
end for
// Lasketaan kaavan (1) kertoimet td, K ja a.
q := (sy·sxx-sx·sxy)/(m·sxx-sx·sx);
r := (m·sxy-sx·sy)/(m·sxx-sx·sx);
td := Sqr(-1/(2·r));
K := Exp(q+ln(td·td));
a := 1/(2·td·td);
```

Algoritmi 2. Gamma-mallin laskenta pienimmän neliösumman menetelmällä.

```
// Mallin muodostaminen kaikista projekteista.
// Pisteparien (t,v) lukumäärä m, missä t on ajanjakso ja v on
// puutteiden lukumäärä.
// Ajanjaksojen lukumäärä T.
Muodosta tietuejoukko P taulun Projektihistoria puutetiedoista;
m := 0; sx := 0; sy := 0;
for t := 1 to T do
    Siirry tietuejoukon P alkuun projektiin p;
    while tietuejoukossa P on alkioita do
        m := m+1;
        v := puutteiden lukumäärä projektissa p ajanhetkellä t;
        y := ln(v/(t·t));
        x := t;
        sy := sy+y;
        sxx := sxx+x·x;
```

```
        sx := sx+x;
        sxy := sxy+x·y;
    end while
end for
// Lasketaan kaavan (2) kertoimet t_d ja K.
q := (sy·sxx-sx·sxy)/(m·sxx-sx·sx);
r := (m·sxy-sx·sy)/(m·sxx-sx·sx);
td := -2/r;
K := td·td·td/4·Exp(q);
```

Mallin tulostaminen (kuva 7) perusmallin sekä dynaamisen mallin osalta tarkoittaa käyrän piirtämistä, joka molemmissa malleissa tapahtuu samalla tavalla. *Perusmallin* tapauksessa tulostetaan yksi käyrä. *Dynaamisessa mallissa* tulostetaan kaksi käyrää, joista toinen kuvaa vanhoista projekteista tuotettua mallia ja toinen käynnissä olevalle projektille laskettua mallia. Kuvissa 9 ja 10 on esitetty esimerkit dynaamisesta mallista. Algoritmi 3 ilmentää käyrän piirtämisen perusajatuksen.



Kuva 7. Käyttötapauksen ”Mallin tulostaminen” tarkennus.

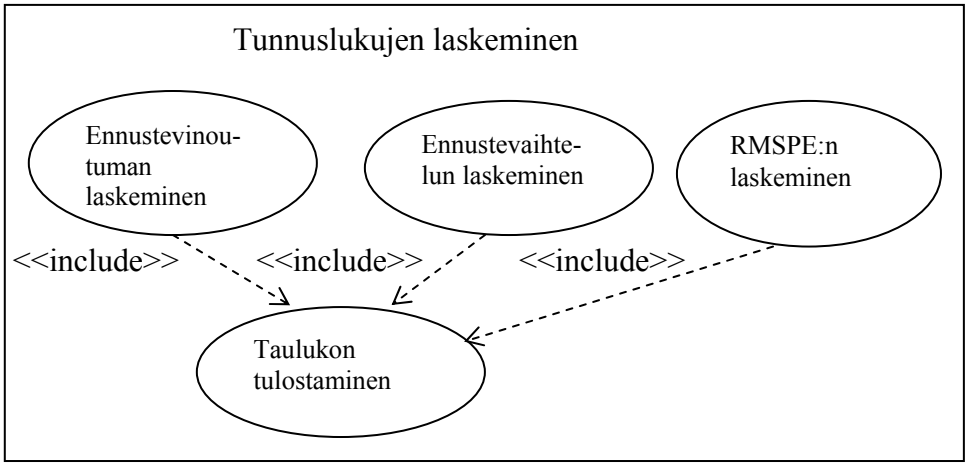
Algoritmi 3. Käyrän piirtäminen.

```
// Ajanjaksojen t lukumäärä T.
// Ajanjakson t puutteiden lukumäärä p.
// Visual Basicin piirtofunktio Line.
// Vasen marginaali Lmarg ja alamarginaali Dmarg.
// Lomakkeen horisontaali koko ScaleHeight.
// Koordinaaston kahden x-pisteen piirtoetäysyys toisistaan Xstep.
// Koordinaaston kahden y-pisteen piirtoetäysyys toisistaan Ystep.
// Puutteiden lukumäärän p vanha arvo Apu.
// Koordinaatiston skaalauskerroin Sdiv.
// Visual Basicin pyöristysfunktio Round.
Piirrä koordinaatisto;
for t := 1 to T do
    Laske mallille kaavan (1) tai (2) mukaisesti puutteiden
    lukumäärä p ajanhetkellä t;
    Skaalaa arvo p koordinaatiston mittakaavaan;
    if t = 1 then
        Line(Lmarg,(ScaleHeight-Dmarg))-((Lmarg+t·Xstep),
        (ScaleHeight-Dmarg-Round(p)·Ystep/Sdiv));
    else
```

```
        Line(Lmarg·(t-1)·Xstep),(ScaleHeight-Dmarg-Round(Apu·
        Ystep/Sdiv)))-((Lmarg+t·Xstep),ScaleHeight-Round(p·
        YStep/Sdiv)));
    end if
end for
```

Tunnuslukujen laskeminen tarkoittaa tunnuslukujen arvojen laskemista ja tulostamista näytölle taulukkomuotoon ennustevinoutuman, ennustevaihtelun ja RMSPE:n osalta. Laskenta tapahtuu algoritmin 4 mukaisesti. Dynaamisen mallin osalta järjestelmä laskee myös ennusteen poikkeaman PED perusmallista kaavalla (3).

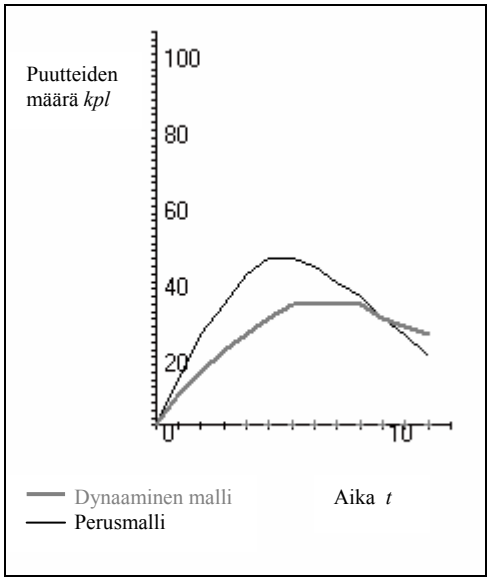
$$PED = \frac{Dyna_{RMSPE} - Perus_{RMSPE}}{Perus_{RMSPE}} * 100 \qquad (3)$$



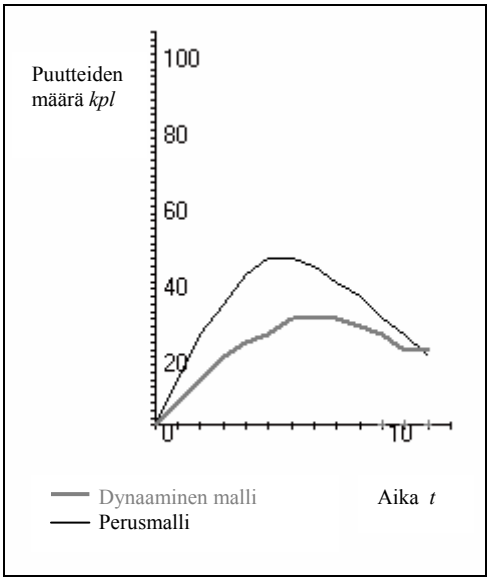
Kuva 8. Käyttötapauksen ”Tunnuslukujen laskeminen” tarkennus.

Algoritmi 4. Tunnuslukujen laskeminen.

```
// Ajanjaksojen t lukumäärä T.
// Tietuejoukko P tunnuslukujen tarvitsemista erotuksista.
// Ennustevinoutuma Bias.
// Ennustevaihtelu Variation.
// Ennustevinoutuman neliön keskiarvon neliöjuuri RMSPE.
Muodosta T alkiota sisältävä tietuejoukko P taulun LaskentaArvot
sarakkeesta perusmalliMiinusDynaaminenMalli,
puutteidenLkmMiinusPerusmalli tai puutteidenLkmMiinusDynaaminenMalli;
Bias := 0;
for t := 1 to T do
        Bias := Bias+P[t];
end for
Bias := Bias/T;
Variation := 0;
for t := 1 to T do
        Variation := Variation+(P[t]-Bias)·(P[t]-Bias);
end for
Variation := Sqrt(Variation/(T-1));
RMSPE := Sqrt(Bias·Bias+Variation·Variation);
```



Kuva 9. Dynaaminen malli ajankohdassa t3.



Kuva 10. Dynaaminen malli ajankohdassa t9.

4 Testaus

Järjestelmän oikeellisuus mallinnuksen osalta on testattu aineistosta lasketun mallin ja mallista lasketun mallin avulla. Testauksen olettamuksena on, että mallin ja siitä lasketun mallin arvojen ennustevinoutuma on nolla, koska näiden mallien arvojen välinen erotus on nolla. Toisin sanoen aineistosta lasketun mallin antaminen lähtöaineistona PUTTE-järjestelmälle tuottaa mallin, joka on sama kuin lähtöaineistona käytetty malli.

Taulukossa 1 on esitetty erään projektin puutteiden lukumäärät ajankohdissa *t* sekä näiden puutetietojen perusteella lasketun Norden/Rayleigh-käyrän mukaisen mallin arvot vastaavissa pisteissä eri tarkkuuksilla.

Taulukko 1: Ajankohdissa *t* projektista laskettu malli eri tarkkuuksilla.

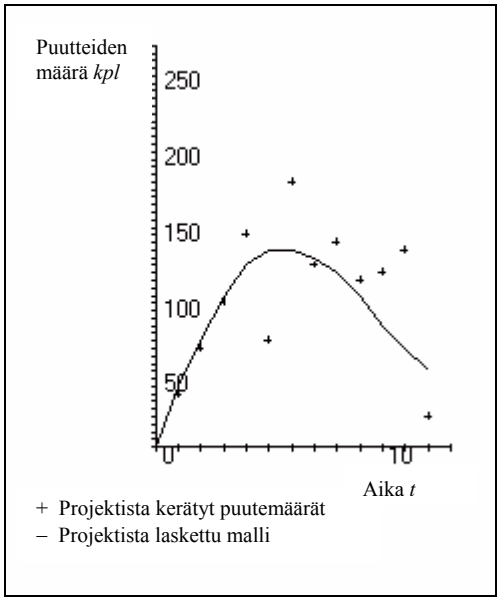
		Mallin arvot eri tarkkuuksilla			
Aika *t*	Puutteiden lkm	0 desimaalia	2 desimaalia	4 desimaalia	15 desimaalia
1	37	38	37,55	37,5543	37,554348993463200
2	66	72	71,73	71,7284	71,728424418542100
3	95	100	99,64	99,6440	99,643960262569500
4	139	119	119,32	119,3234	119,323372028117000
5	70	130	129,91	129,9088	129,908839764665000
6	174	132	131,67	131,6711	131,671109506788000
7	119	126	125,83	125,8274	125,827393853717000
8	136	114	114,23	114,2279	114,227964312650000
9	109	99	98,99	98,9914	98,991435041497700
10	113	82	82,17	82,1666	82,166567160689300
11	128	65	65,48	65,4780	65,477973861555600
12	21	50	50,18	50,1832	50,183225166750700

Taulukossa 2 on laskettu ennustevinoutuma, ennustevaihtelu ja RMSPE mallista lasketulle mallille, kun lähtötietoina käytetään taulukon 1 mallin arvoja eri tarkkuuksilla. Taulukosta 2 nähdään, että ennustevinoutuma lähenee nollaa, kun lähtötietojen tarkkuutta kasvatetaan nollasta desimaalista 15 desimaaliin. Tästä voidaan päätellä, että järjestelmä suorittaa Norden/Rayleigh-mallin laskennan lineaarisine logaritmimuunnoksineen oikein. Vastaavalla periaatteella on testattu Gamma-mallin laskenta ja saatu samansuuntaiset tulokset.

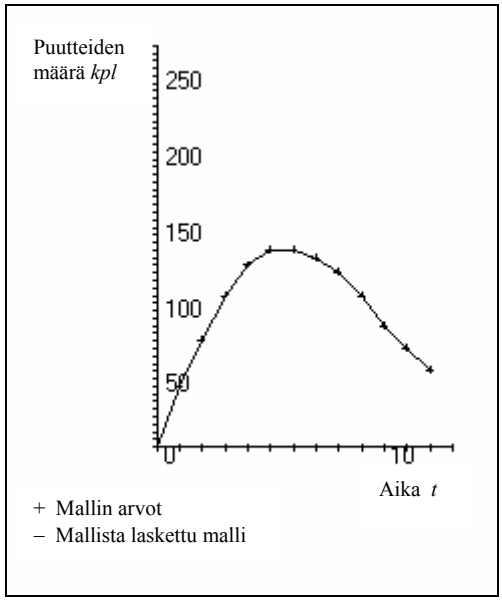
Taulukko 2: Tunnuslukujen laskenta eri syöttötarkkuuksilla.

Syöttötarkkuus	Ennustevinoutuma	Ennustevaihtelu	RMSPE
Kokonaisluku	-0,0542769821117428	0,262483480586933	0,268036505663294
2 desimaalia	0,000717946721688752	0,00283764492092528	0,00292705930798074
4 desimaalia	0,0000067094871785874	0,0000376302620563392	0,0000382237339964113
15 desimaalia	6,21724893790088E-14	2,06671798478482E-13	2,15820876473786E-13

Käyttämällä syötteenä taulukon 1 puutearvoja saadaan kuvan 11 mukainen malli. Kuvassa 12 on esitetty vastaava mallista laskettu malli käyttämällä lähtötietoina taulukossa 1 esitettyjä mallin arvoja 15 desimaalin tarkkuudella. Kuvasta 12 voidaan havaita, että muodostunut malli kulkee lähtötietoina annettujen mallin pisteiden kautta, kuten taulukon 2 perusteella pitääkin olla. Tämä osoittaa, että myöskin piirtoalgoritmi on ohjelmoitu oikein.



Kuva 11: Malli



Kuva 12: Mallista laskettu malli.

Viitteet

Booch G., Rumbaugh J., Jacobson I.: *The Unified Modeling Language User Guide*, Addison-Wesley, Reading, Massachusetts, USA, 1999.

Pillai K., Nair S.V.S. 1997. A Model for Software Development Effort and Cost Estimation, *IEEE Transactions on Software Engineering*, 23(8), 485-497.

Putnam L.H. 1978. A General Empirical Solution to the Macro Software Sizing and Estimation Problem*, IEEE Transactions on Software Engineering*, 4, July, 345-361.

Spiegel, M.R. 1961. *Theory and Problems of Statistics*, McGraw-Hill, New York.