

# **Avointen ohjelmistojen laatu**

Päivi Pääkkö

Helsinki 18.11.2008

Seminaarityö

HELSINGIN YLIOPISTO

Tietojenkäsittelytieteen laitos

## Sisältö

<b>1</b>	<b>Johdanto.....</b>	<b>1</b>
<b>2</b>	<b>Laatu käsitteenä.....</b>	<b>1</b>
2.1	Erilaiset laatonäkökulmat.....	1
2.2	Laadun määritelmiä .....	3
2.3	Tuoteperustaiset laatumallit .....	3
<b>3</b>	<b>Suljettu vs. avoin ohjelmisto.....</b>	<b>4</b>
3.1	Kehityksen eroavaisuuksia .....	4
3.2	Eroavaisuuksia laadussa .....	6
<b>4</b>	<b>Ongelmana käyttäjien näkökulman huomioiminen .....</b>	<b>7</b>
<b>5</b>	<b>Yhteenveto .....</b>	<b>9</b>
	<b>Lähteet .....</b>	<b>10</b>

# 1 Johdanto

Sananlaskun mukaan kauneus on katsojan silmissä, vastaavasti ohjelmistoalalla ohjelmiston laatu on katsojan silmissä [FeP97, s. 337]. Tuotteen laatuun ja laadun määritelmään vaikuttaa se, kuka ohjelmistoa arvioi. Esimerkiksi käyttäjä voi kokea ohjelman laadukkaaksi, jos se on helppokäyttöinen, edullinen ostettaessa ja sen käyttöönotto on helppoa. Sama ohjelma voi esimerkiksi ohjelman ylläpitäjän kannalta tuntua vähemmän laadukkaalta, mikäli koodi on hankalasti ylläpidettävissä tai se vaatii jonkun tietynlaisen hankalan teknisen ratkaisun taakseen. Ohjelmistojen laatu on monimutkainen käsite, koska se merkitsee eri asiaa eri ihmisille ja on siten hyvin kontekstisidonnainen [KiP96]. Laatu on kuitenkin yksi tärkeimmistä ohjelmistoprojektin menestymisen mitoista [RaB05].

Tämän työn tarkoituksena on käsitellä avoimien ohjelmistojen laatua. Luvussa kaksi käsitellään erilaisia laatu näkökulmia ja –määritelmiä sekä tarkastellaan mitä nämä tarkoittavat avoimien ohjelmistojen kohdalla. Kolmannessa luvussa paneudutaan avoimen ohjelmistokehityksen erityispiirteisiin laadun näkökulmasta sekä vertaillaan avoimia ja suljettuja ohjelmistoja. Neljännessä luvussa pohditaan laatua ohjelmistojen käyttäjien näkökulmasta.

## 2 Laatu käsitteenä

Niin ohjelmistotuotannossa kuin muillakin aloilla laatu on yleensä helppo tunnistaa kun siihen törmää, mutta se on hankalasti määriteltävissä. Gillies:n mukaan laatu on usein melko läpinäkyvää, mutta sen puuttuminen on helposti havaittavissa [Gil92, s. 3]. Kiinnitämme laatuun huomiota yleensä vasta siinä vaiheessa, kun laadun puuttuminen häiritsee toimintaamme. Toisaalta muutamme herkästi käsitystämme laadusta suhteessa odotuksiimme. Jos tuote on esimerkiksi halpa, emme odota sen olevan kovin laadukas, vaan hyväksymme siinä olevat virheet. Kalliin tuotteen kohdalla samat virheet tekevät tuotteesta huonolaatuisen, koska odotimme kalliin tuotteen olevan laadukas.

### 2.1 Erilaiset laatu näkökulmat

Laadun määritelmää voidaan lähestyä ylivertaisen, tuoteperustaisen, käyttäjäperustaisen, tuotantoperustaisen tai arvoperustaisen lähestymistavan kautta

[Gar84]. **Ylivertaisessa lähestymistavassa** laadulla tarkoitetaan synnynnäistä erinomaisuutta, jota ei voida tarkasti määritellä, mutta jonka opimme tunnistamaan kokemuksen kautta. **Tuoteperustaisessa lähestymistavassa** laadun ajatellaan olevan täsmällinen ja mitattavissa oleva muuttuja. Näkökulman mukaan eri tuotteiden laatutaso riippuu tuotteen muodostavien aineiden tai attribuuttien määrästä [Gar84]. Puhuttaessa ohjelmistojen laadusta, ajatellaan Kitchenhamin ja Pfleegerin mukaan usein tuoteperustaista näkökulmaa. Tällöin oletetaan, että tuotteen sisäisten ominaisuuksien mittaaminen ja kontrollointi parantavat tuotteen ulkoista käyttäytymistä. Kitchenhamin ja Pfleegerin mukaan ohjelmistojen laadun arvioiminen ohjelmiston sisäisillä ominaisuuksilla on houkuttelevaa, koska se tarjoaa objektiivisen ja kontekstista riippumattoman näkökulman laatuun [KiP96]. Tuoteperustainen lähestymistapa onkin avoimien ohjelmistojen kohdalla käytetyin lähestymistapa.

**Käyttäjäperustainen lähestymistapa** ottaa huomioon käyttäjien näkökulman. Se hyväksyy käyttäjien subjektiiviset ja erilaiset tarpeet. Näkökulman ongelmana on tyytyväisyyden ja laadun rinnastaminen toisiinsa [Gar84]. Nämä eivät kuitenkaan aina tarkoita samaa asiaa. Esimerkiksi asiakas voi olla tyytyväisin tietyn makuiseen jäätelöön, mutta pitää silti toisenmerkkistä ja -makuista jäätelöä laadukkaampana. Ohjelmistotuotannossa käyttäjäperustaiseen näkökulmaan liittyvät esimerkiksi käytettävyytustutkimus sekä käyttövarmuuden ja suorituskyvyn mallinnus [KiP96]. Käyttäjäperustaisen näkökulman vastakohtana on **tuotantoperustainen näkökulma**, jossa huomio keskittyy tuotteen valmistukseen. Tässä lähestymistavassa laatu määritellään vaatimuksien noudattamisena [Gar84]. Tällöin ajatellaan, että kaikki määrittelyyn tehtävät poikkeamat vähentävät tuotteen laatua. Taustalla on ajatus, jonka mukaan tuote kannattaa kustannussyistä tehdä kerralla oikein, jotta sitä ei jouduta myöhemmin korjaamaan. Näin laadun parantaminen johtaa kustannusten laskuun. Ohjelmistoalalla tuotantoperustaista näkökulmaa edustavat erilaiset ohjelmistoprosesseihin keskittyvät mallit, kuten CMM-kypsyystasomalli. Tuotantoperustaisen näkökulman ongelmana on, että se keskittyy liikaa tuotantoprosessin arviointiin ja unohtaa helposti itse tuotteen [KiP96]. Avoimien ohjelmistojen kohdalla ongelmana on myös se, että käytettävää ohjelmistoprosessia ei useinkaan ole määritelty eikä dokumentoitu [Abe07]. **Arvoperustainen lähestymistapa** vie kustannusajattelun vielä pidemmälle. Näkökulman mukaan laadukas tuote täyttää tehtävänsä sopivalla hinnalla, jolloin laatu riippuu siitä kuinka

paljon asiakkaat ovat valmiita siitä maksamaan [Gar84]. Tällöin esimerkiksi 100 euroa maksava jäätelöannos ei voisi olla kovin laadukas, koska sillä olisi hyvin vähän ostajia. Avointenohjelmistojen kohdalla kustannuslähtöiset ajattelutavat ovat kuitenkin ongelmallisia, koska käyttäjä ei maksa tuotteesta mitään tai kustannus on hyvin pieni verrattuna suljettuihin ohjelmistoihin.

## 2.2 Laadun määritelmiä

Pressmanin mukaan ohjelmisto käsittää tietokoneohjelman, tietorakenteet ja näihin liittyvän dokumentaation, joiden tarkoituksena on palvella jotakin johdonmukaista toimintatapaa tai vaadittua hallintaa [Pre00, s. 242]. Ohjelmistojen laadulle on hankala antaa vastaavaa yksiselitteistä ja kattavaa määritelmää [Jon08, s. 454-455], sillä laatu on abstrakti käsite. Wongin mukaan määritelmiä tuntuu olevan yhtä monta kuin on aiheesta kirjoittavia ihmisiä. Laatu täytyy kuitenkin määritellä, jotta organisaatio voi tietää koska se on saavuttanut tietyn laatutason [Won06, s. 56]. Esimerkiksi Juranin mukaan laatu koostuu tuotteen ominaisuuksista, jotka vastaavat asiakkaan tarpeita ja tarjoavat siten tyytyväisyyttä tuotteeseen, sekä puutteiden puuttumisesta [Jur88, s. 2.2]. Myös IEEE:n laatumääritelmä on yleisesti käytössä. Siinä ohjelmistojen laadulla tarkoitetaan sitä astetta, jolla järjestelmä, komponentti tai prosessi täyttää määritellyt vaatimukset sekä asiakkaan ja käyttäjän tarpeet tai odotukset [Rad90, s. 60]. Määritelmä noudattaa verifiointin ja validoinnin periaatteita, jossa tuotetta verrataan sekä määriteltyyn dokumentaatioon että asiakkaan todellisiin tarpeisiin. Modernein laadun määritelmä sen sijaan löytyy Pressmanilta. Hänen määritelmänsä mukaan laadulla tarkoitetaan eksplisiittisesti todettujen toiminnallisuus- ja suorituskykyvaatimusten noudattamista, eksplisiittisesti dokumentoitua kehitysstandardia ja implisiittisiä ominaisuuksia, joita odotetaan kaikilta ammattimaisesti tuotetuilta ohjelmistoilta [Pre00, s. 198]. Määritelmässä huomioidaan kaikki kolme ohjelmistotuotannossa käytettävää laatu näkökulmaa: tuoteperustaisen näkökulman vaatimusten noudattaminen, tuotantoperustaisen näkökulman standardoitu kehitys sekä käyttäjän näkökulman implisiittiset odotukset.

## 2.3 Tuoteperustaiset laatumallit

Ohjelmistotuotteen laatua mittaavissa laatumalleissa laatu koostuu useista erilaisista laatuattribuuteista, kuten esimerkiksi käytettävyydestä, ylläpidettävyydestä ja

luotettavuudesta. Laatuattribuutit jaetaan yleensä sisäisiin ja ulkoisiin laatuattributteihin. Fentonin ja Pfleegerin mukaan sisäiset laatuattribuutit kuvaavat ohjelmiston sisäisiä ominaisuuksia, jotka voidaan mitata tarkastelemalla ainoastaan itse ohjelmistotuotetta. Ulkoisten laatuattribuuttien kohdalla tarkastellaan puolestaan ohjelmiston käyttäytymistä suhteessa sen ympäristöön [FeP97, s. 74]. Ohjelmiston laatu määritellään vertaamalla ohjelmistoa haluttujen laatuattribuuttien yhdistelmään, jolloin laatu tarkoittaa sitä, kuinka suuren määrän ohjelmisto näistä halutuista laatuattribuuteista täyttää [Sch04].

McCall ja Boehm kehittivät 1970-luvun loppupuolella ensimmäiset hierarkkiset mallit ohjelmistojen laadun kuvaamiseksi. Malleissa erilaiset laatutekijät jaetaan pienempiin osiin, kunnes lopulta päädytään mitattavissa oleviin yksinkertaisiin mittoihin. Mallit määrittivät ohjelmiston laadun korkealla abstraktiotasolla ja ne toimivat pohjana monille muille laatumalleille, kuten esimerkiksi ISO/IEC 9126 standardissa määritellylle ISO-laatumallille. Kitchenhamin ja Pfleegerin mukaan ei ole olemassa mitään tiettyä yleismaailmallista laatumallia tai laadun määritelmää mikä sopii kaikkialle, koska laadun käsite on monimutkainen [KiP96]. Tämän vuoksi useat organisaatiot ovat päätyneet määrittelemään itse oman laatumallinsa tai yhdistelemään tunnettuja laatumalleja toisiinsa päätyen niin sanottuun tee-se-itse-malliin. Fentonin ja Pfleegerin mukaan tee-se-itse-mallissa laatu määritellään jokaisen ohjelmistotuotteen kohdalla yhteistyössä asiakkaan kanssa [FeP97, s. 340-341].

### 3 Suljettu vs. avoin ohjelmisto

Suljettujen ja avointen ohjelmistojen kehitys eroaa monilta osin toisistaan, siten myös niiden laadussa voi olla eroja. Seuraavassa luvussa vertaillaan avoimia ja suljettuja ohjelmistoja erityisesti laadun näkökulmasta katsottuna.

#### 3.1 Kehityksen eroavaisuuksia

Avoimien ohjelmistojen kehittäminen poikkeaa huomattavasti suljettujen ohjelmistojen kehityksestä, mikä vaikuttaa myös laadunhallintaan. Kehitettäessä avointa ohjelmistoa esimerkiksi laadunvarmistus- ja testausmenetelmät sekä riskienarviointiprosessi ovat usein epämuodollisia (taulukko 3.1). Avoimissa projekteissa projektin jäsenet työskentelevät vapaaehtoisesti ja monesti ilman

rahallista korvausta. Zhao et al. tekemän tutkimuksen mukaan lähes 60 prosenttia avoimista projekteista lähdetään kehittämään kehittäjän omien tarpeiden vuoksi ja 77 prosenttia kehittäjistä kehittää ohjelmistoa vapaa-ajallaan [ZhE03]. Lisäksi projekteissa käytetään yleensä projektin suunnitteluun ja aikataulutukseen vain vähän aikaa ja projekteissa aloitetaan usein suoraan koodaamaan ilman muodollista ja vahvistettavaa suunnitteluvaihetta [Abe07].

<b>Suljettu ohjelmisto</b>	<b>Avoin ohjelmisto</b>
Hyvin määritelty kehitysmenetelmä	Kehitysmenetelmää ei yleensä ole määritelty tai dokumentoitu
Laaja projektin dokumentaatio	Vähän projektin dokumentaatiota
Muodollinen ja jäsentynyt testaus- ja laadunvarmistusmenetelmä	Ei-jäsentynyt ja epämuodollinen testaus- ja laadunvarmistusmenetelmä
Analyytikot määrittelevät vaatimukset	Ohjelmoijat määrittelevät vaatimukset
Muodollinen riskienarviointiprosessi – valvottu ja johdettu koko projektin ajan	Ei muodollista riskienarviointiprosessia
Mitattavissa olevia tavoitteita käytetään koko projektin ajan	Vain muutamia mitattavissa olevia tavoitteita
Virheiden löytämiseen käytetään mustalaatikkotestausta niin varhaisessa vaiheessa kuin vain on mahdollista	Virheiden löytämiseen käytetään mustalaatikkotestausta vain prosessin loppuvaiheessa
Laatuun liittyvää empiiristä todistusaineistoa käytetään jatkuvasti päätöksenteon apuna	Laatuun liittyvää empiiristä todistusaineistoa ei kerätä
Ryhmän jäsenet on määrätty töihin	Ryhmän jäsenet ovat valinneet työnteon
Muodollinen suunnitteluvaihe suoritetaan ja vahvistetaan ennen ohjelmoinnin aloittamista	Projekteissa aloitetaan yleensä suoraan ohjelmoimaan
Projektin suunnitteluun ja aikataulutukseen käytetään paljon vaivaa	Projektin suunnitteluun ja aikataulutukseen käytetään vähän vaivaa

Taulukko 3.1: Laadunhallinta avoimen- ja suljetunlähdekoodin ohjelmistokehityksessä [Abe07].

## 3.2 Eroavaisuuksia laadussa

Voisi kuvitella, että avointen ohjelmistojen laatu olisi selkeästi huonompaa, kun tuotantoprosessissa on näin merkittäviä eroja. Näin ei kuitenkaan välttämättä ole. Vaikka avoimien ohjelmistojen kehitys poikkeaa merkittävästi suljettujen ohjelmistojen kehityksestä, ovat avoimet ohjelmistot tästä huolimatta laadukkaita. Niiden täytyy siis saavuttaa laatu muilla keinoin kuin perinteisissä suljetuissa ohjelmistotuotantoprojekteissa. Stamelos et al. tekemässä tutkimuksessa käytiin läpi yli sata avointa ohjelmistoa. Tutkimuksessa huomattiin, että koodin rakenteellinen laatu oli korkeampaa kuin odotettiin ja verrattavissa suljettuihin ohjelmistoihin [SAO02].

Paulson et al. tutkivat puolestaan viittä avoimiin ohjelmistoihin liittyvää yleistä käsitystä, joita ovat: avoin ohjelmistokehitys mahdollistaa nopeamman järjestelmän kasvun, avoimet projektit edistävät enemmän luovuutta, avoimet projektit menestyvät yksinkertaisuutensa vuoksi, avoimissa ohjelmistoissa on yleisesti ottaen vähemmän virheitä, koska virheet löydetään ja korjataan nopeammin, sekä avoimet ohjelmistot ovat modulaarisempia [PSE04]. He vertailivat tutkimuksessaan kolmea suljettua ja kolmea avointa ohjelmistoa ja löysivät tukea ainoastaan kahdelle näistä käsityksistä. Ensinnäkin Paulson et al. tekemän tutkimuksen mukaan avoimet projektit edistävät enemmän luovuutta, sillä avoimiin ohjelmistoihin lisättiin enemmän uutta toiminnallisuutta [PSE04]. Toiminnallisuuden lisäämistä tukee nopea julkaisusykli, joka pitää Aberdourin mukaan koodintarkastajat ja -kehittäjät kiinnostuneina ja motivoituneina [Abe07]. Zhao et al. tekemän tutkimuksen mukaan jopa 43 prosentissa tutkituista ohjelmistoista julkaistiin uusi versio joka kuukausi [ZhE03].

Toisekseen Paulson et al. tekemän tutkimuksen mukaan avoimissa ohjelmistoissa on vähemmän virheitä, sillä niitä muokataan ahkerammin [PSE04]. Korkea modulaarisuus ja suuri joukko virheiden etsijöitä ja korjaajia johtavat pieneen virhetiheyteen [Abe07]. Erityisesti virheiden pienempi määrä viittaa avoimien ohjelmistojen korkeampaan laatuun, sillä virheettömyyttä pidetään yleisesti yhtenä merkinä ohjelmiston korkeasta laadusta. Zhao et al. tekemän tutkimuksen mukaan noin 75 prosentissa tutkituista projekteista käytettiin konfiguraationhallinta työkaluja

ja yli 61 prosentissa käytettiin virheiden jäljitys työkaluja. Testaamiseen käytettiin myös melko paljon aikaa, keskimäärin 21 prosenttia [ZhE03]. Näiden avulla projektit pystyvät hallitsemaan ohjelmiston virheitä ja siinä tapahtuvia muutoksia, mikä näkyy ohjelmiston laadussa. Mielenkiintoista Zhao et al. tekemässä tutkimuksessa oli myös se, että 61 prosentilla tutkimukseen osallistuneista kehittäjistä oli kokemusta ohjelmistojen kehityksestä yli viiden vuoden ajan [ZhE03]. Heitä voidaan pitää alan ammattilaisina, joten voitaisiin olettaa, että avoimissa ohjelmistoissa on vähän aloittelijoiden tekemiä virheitä.

Aberdourin mukaan korkealaatuinen avoin ohjelmisto on riippuvainen suuresta ja pysyvästä yhteisöstä, mikä mahdollistaa koodin nopean kehityksen, tehokkaan virheenpoiston ja uusien ominaisuuksien lisäämisen. Pysyvän yhteisön syntymisen mahdollistavat koodin modulaarisuus, hyvä dokumentaatio, tutoriaalit, kehitystyökalut sekä palkitseva ja tunnustusta antava kulttuuri [Abe07]. Saavuttaakseen pysyvän kehityksen ja korkealaatuisen ohjelmiston täytyy sekä järjestelmän että yhteisön kehittyä Aberdourin mukaan samanaikaisesti.

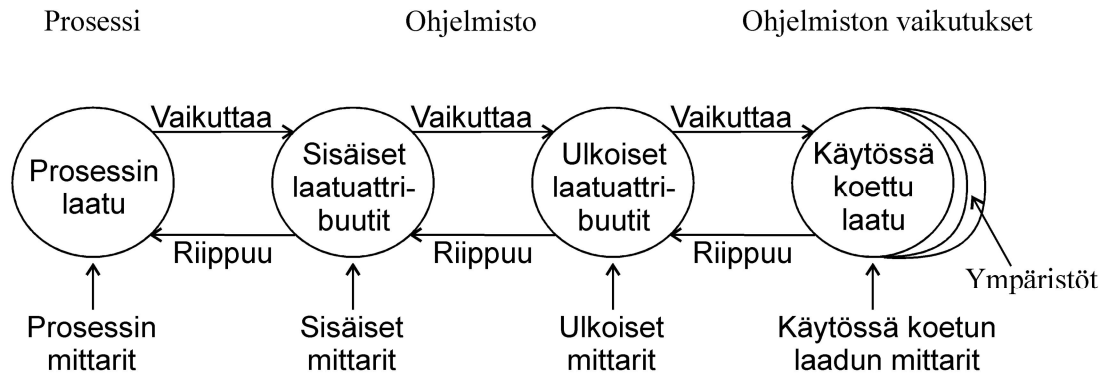
## 4 Ongelmana käyttäjien näkökulman huomioiminen

Perinteisesti käyttäjät eivät ole osanneet vaatia ohjelmistoilta korkeaa laatua. On totuttu liiaksi siihen, että ohjelmistot välillä kaatuilevat tai eivät muuten täytä vaatimuksia. Suljettujen ohjelmistojen toimittajat eivät usein kiinnitä suurta huomiota ohjelmistojen laatuun, koska he saavat yleensä rahansa toimittamansa ohjelmiston huonosta laadusta huolimatta [Off02]. Usein toimittajille jopa maksetaan myöhemmin lisää, jotta he korjaisivat aikaisemmat virheensä. Tämä ei kuitenkaan päde avoimien ohjelmistojen kohdalla, joiden kehittäjät eivät yleensä saa työstään rahaa.

Ohjelmiston koko elinkaaren kannalta ajateltuna sen tärkein sidosryhmä ovat ohjelmiston käyttäjät, koska ohjelmiston tulee täyttää jokin käyttäjien tarve. Ilman käyttäjiä koko ohjelmisto on täysin turha. Ohjelmistotuotannossa tuotantoorganisaatio katsoo ohjelmiston laatua tyypillisesti tuote- tai tuotantoperustaisesta näkökulmasta, kun taas asiakkaat ja markkinointiosasto katsovat laatua käyttäjäperustaisesta näkökulmasta [KiP96]. Koska ohjelmisto tarvitsee käyttäjiä, tulisi laatua mielestäni tarkastella käyttäjäperustaisesta näkökulmasta perinteisen

tuotanto- tai tuoteperustaisen näkökulman sijaan, sillä nämä näkökulmat eivät huomioi ohjelmiston kaikkein tärkeimmän sidosryhmän eli käyttäjien näkökulmaa tarpeeksi hyvin. Käyttäjien näkökulman huomioiminen voi kuitenkin olla hankala, koska käyttäjien näkökulma ohjelmistoon on usein subjektiivinen. Avoimissa ohjelmistoissa käyttäjän näkökulman huomioiminen voi olla erityisen hankalaa koska ohjelmoijat määrittelevät ohjelmiston vaatimukset. Toisaalta taas avoimessa ohjelmistokehityksessä käyttäjät voivat osallistua projektiin aktiivisesti ehdottamalla uusia ominaisuuksia, raportoimalla virheistä sekä avustaa kehittäjiä pyydettäessä [ZhE03].

Käyttäjän näkökulman huomioimiseksi Bevan esittelee 1990-luvun loppupuolella käytössä koetun laadun [BeM94, Bev95a, Bev95b, Bev99]. Artikkeleissa hän lähtee liikkeelle käytettävyydestä, jonka hän katsoo olevan käytössä koettua laatua tietyssä ympäristössä [BeM94]. Siten käytössä koettua laatua voidaan käyttää käytettävyyden mittaamiseen [Bev95a], jolloin käytettävyyden tavoitteena on käytössä koetun laadun saavuttaminen [Bev95b]. Myöhemmissä artikkeleissa Bevan päätyy siihen, että käytössä koettu laatu on laajempi käsite kuin käytettävyys, koska siihen voivat vaikuttaa kaikki laatuominaisuudet [Bev99]. Esimerkiksi käyttäjien käytössä kokema laatu on seurausta tuotteen toiminnallisuudesta, käyttövarmuudesta, käytettävyydestä ja suorituskyvystä, kun taas ohjelmiston ylläpitäjän käytössä kokema laatu riippuu ylläpidettävyydestä [Bev99]. Bevanin esittelemä käytössä koettu laatu lisätään myöhemmin ISO/IEC 9126 standardissa määriteltyyn ISO-laatumalliin. Mallissa kuvataan kaksiosainen ohjelmistojen laadun malli (kuva 4.1). Ensimmäisessä osassa käsitellään itse ohjelmistotuotteen laatua ja toisessa osassa ohjelmiston käytössä koettavaa laatua [ISO01]. ISO-laatumallin mukaan ohjelmistoprosessin laatu vaikuttaa ohjelmiston sisäisiin laatuattributteihin. Nämä puolestaan vaikuttavat ohjelmiston ulkoisiin laatuattributteihin, jotka vaikuttavat käytössä koettuun laatuun. Toisin sanoen käytössä koettu laatu on käyttäjän kokema sisäisten ja ulkoisten laatuattribuuttien yhdistetty vaikutus. Siten käytössä koettu laatu pitäisi olla tavoite, joka voidaan saavuttaa laadukkaalla tuotteella [CoO06].



Kuva 4.1: Laatu ohjelmiston elinkaarella [ISO01].

Ohjelmistoprosessin laatu heijastaa tuotantoperustaista laatonäkökulmaa ja ohjelmistotuotteen laatu heijastaa tuotepерustaista näkökulmaa. Käytössä koettu laatu puolestaan kuvaa käyttäjän näkökulmaa ohjelmistojen laatuun ja se voidaan määrittellä tuotteen kykyä täyttää käyttäjän määrittelemät tehokkuuteen, tuottavuuteen, käyttöturvallisuuteen ja tyytyväisyyteen liittyvät tavoitteet tietyssä toimintaympäristössä [ISO01].

Côté et al. vertailivat vastikään eri laatumalleja keskenään etsiessään laajasti hyväksyttyä ja käyttökelpoista ohjelmistojen laadun mallia [CSG07]. He vertailivat keskenään McCall:n, Boehm:n, Dromey:n ja ISO/IEC 9126 standardin laatumalleja muun muassa sen mukaan tukivatko ne kaikkia Garvinin määrittelemiä eri laatonäkökulmia. Côté et al. tulivat siihen tulokseen, että ISO/IEC 9126 standardin määrittelemä ISO-laatumalli on näistä malleista paras, koska se oli ainut laatumalli, joka täytti heidän mallille asettamansa kriteerit [CSG07].

## 5 Yhteenveto

Laatu on monimutkainen käsite ja sen määritelmä riippuu siitä, mistä näkökulmasta laatua katsotaan. Avoimien ohjelmistojen kohdalla laatua katsotaan perinteisesti tuotepерustaisesta näkökulmasta, jolloin laatua voidaan mitata mittaamalla tuotteen ominaisuuksia. Sekä avoimista että suljetuista ohjelmistoista löytyy laadukkaita ja vähemmän laadukkaita ohjelmistoja ja onkin mahdoton sanoa onko avoimet ohjelmistot laadukkaampia kuin suljetut ohjelmistot. Vaikka ohjelmistoja on hankala asettaa paremmuusjärjestykseen laadun kuitenkin yleensä tunnistaa kun siihen törmää.

## Lähteet

- Abe07 Aberdour M., Achieving Quality in Open Source Software. *IEEE Software* vol. 24, nro. 1, (tammikuu/helmikuu 2007), sivut 58-64.
- Bev95a Bevan N., Measuring usability as quality of use. *Software Quality Journal*, vol. 4, nro. 2, (kesäkuu 1995), sivut 115-130.
- Bev95b Bevan N., Usability is Quality of Use. *Proceedings of the 6th International Conference on Human Computer Interaction*, Yokohama, heinäkuu 1995.
- Bev99 Bevan N., Quality in use: Meeting user needs for quality. *The Journal of Systems and Software*, vol. 49, nro. 1, (joulukuu 1999), sivut 89-96.
- BeM94 Bevan N. ja Macleod M., Usability measurement in context. *Behaviour & Information technology*, vol. 13, nro. 1-2, (1994), sivut 132-145.
- CSG07 Côté M-A., Suryn W. ja Georgiadou E., In search for a widely applicable and accepted software quality model for software quality engineering. *Software Quality Journal*, vol. 15, nro. 4, (joulukuu 2007), sivut 401-416.
- CoO06 Covella G. ja Olsina L., Assessing Quality in Use in a Consistent Way. *Proceedings of the 6th international conference on Web engineering (ICWE '06)*, Kalifornia, heinäkuu 2006, sivut 1-8.
- FeP97 Fenton N. E. ja Pfleeger S. L., *Software Metrics, A Rigorous & Practical Approach*. PWS Publishing Company, Boston, 1997.
- Gar84 Garvin D., What Does "Product Quality" Really Mean?. *Sloan Management Review*, vol. 26, nro. 1, (syksy 1984), sivut 25-43.
- Gil92 Gillies A. C., *Software Quality, Theory and Management*. Chapman & Hall, Lontoo, 1992.
- ISO01 ISO/IEC 9126-1, Software engineering - Product quality - Part 1: Quality model. Geneve, Sveitsi, 2001, sivut 1-25.
- Jon08 Jones C., *Applied Software Measurement, Global Analysis of Productivity and Quality, Third Edition*. Mc Graw Hill, Yhdysvallat, 2008.
- Jur88 Juran J. M., *Juran's Quality Control Handbook, Fourth Edition*. Mc Graw-Hill, Yhdysvallat, 1988.
- KiP96 Kitchenham B. ja Pfleeger S. L., Software Quality: The Elusive Target. *IEEE Software*, vol. 13, nro. 1, (tammikuu 1996), sivut 12-21.

- Off02 Offutt J., Quality Attributes of Web Software Applications. *IEEE Software*, vol. 19, nro. 2, (maaliskuu 2002) , sivut 25-32.
- PSE04 Paulson J.W., Succi G., Eberlein A., An Empirical Study of Open-Source and Closed-Source Software Products. *IEEE Transactions on Software Engineering*, vol. 30, nro. 4, (huhtikuu 2004), sivut 246-256.
- Pre00 Pressman R. S., *Software Engineering, A Practitioner's Approach, European Adaptation, Fifth Edition*. McGraw-Hill Publishing Company, Lontoo, 2000.
- Rad90 Radatz J. et al., IEEE Standard Glossary of Software Engineering Terminology. Standards Coordinating Committee of the Computer Society of the IEEE, Yhdysvallat, 1990.
- RaB05 Raja U. ja Barry E., Investigating Quality in Large-Scale Open Source Software. *ACM SIGSOFT Software Engineering Notes*, vol. 30, nro. 4, (heinäkuu 2005), sivut 1-4.
- Sch04 Schneidewind N. et al., IEEE Standar for a Software Quality Metrics Methodology. Software Engineering Standards Committee of the IEEE Computer Society, Yhdysvallat, 2004.
- SAO02 Stamelos I., Angelis L., Oikonomou A. ja Bleris G. L., Code quality analysis in open source software development. *Information Systems Journal*, vol. 12, nro. 1, (tammikuu 2002), sivut 43-60.
- Won06 Wong B., Different Views of Software Quality. Teoksessa *Measuring Information Systems Delivery Quality*, Duggan E. W. ja Reichgelt H., Idea Group Publishing, 2006, sivut 55-89.
- ZhE03 Zhao L. ja Elbaum S., Quality assurance under the open source development model. *The Journal of Systems and Software*, vol. 66, nro. 1, (huhtikuu 2003), sivut 65-75.