Ohjelmistotuotanto 2017 REFERAATTI
Viikko7 laskarit 8.5.2017
Satu Bennert

Ohjelmistotuotantomenetelmien kehittyminen 1950- luvulta nykypäivään

Software engineering eli ohjelmistotuotanto miellettiin alkuaikoina insinööritieteeksi, jossa kokonaisvaltaisella suunnittelulla on suurempi merkitys kuin itse toteutuksella. Siksi suunnitteluun ja sen dokumentointiin käytettiin enemmän aikaa kuin sitä annettiin itse toteutukselle ja lähellä deadlinea alkoi ylitöitä tehden integraatiohelvetti. Budjettien ja aikataulujen ylitykset ja monimutkaisuuden hallitsemattomuus johtivat ohjelmistokriisiin 1968. Aikojen saatossa on huomattu, että lopputuote on se tärkein asia, mistä asiakas on halukas maksamaan. Loistavat suunnitelmat ja dokumentaatio ovat toissijaisia. RUP-mallin tuotti UML-dokumentit, mutta silti lopputuote ratkaisee. Päädyttiin iteratiiviseen ja inkrementaaliseen ohjelmistotuotantoon. Mukana on kaikki työvaiheet: määrittely, suunnittelu, toteutus, testaus, mutta nopeutetuissa sykleissä verrattuna alkuaikojen lineaariseen vesiputousmalliin. Ohjelmia ei voi rakentaa enää vuosia, vaan kuukausia. Asiakas haluaa olla aktiivisesti mukana päättämässä koko prosessin ajan. Scrum ketteränä kehitysmenetelmänä antaa mahdollisuuden joustavaan tapaan muuttaa määrityksiä ja toiminnallisuuksia.Lyhyt (2-4 vkon) sykli varmistaa lopputuotteen kehityksen oikean suunnan tai antaa hälytyksen alati muuttuvan ympäristön ongelmista tai tarpeesta suunnan muutokseen. Ohjelmistotuotanto ei ole suoraviivaista kuin sillan rakentaminen vaan ennalta-arvaamattoman monimutkaisuuden hallintaa ja herkkää reagointia sen hetkiseen tarpeeseen.

Is design dead? Onko suunnittelu kuollut?

Sanotaan, että XP vaatii ohjelmistosuunnittelun kuolemaa. Evolutionaarinen suunnittelu tarkoittaa sitä, että järjestelmän suunnittelu kasvaa sitä mukaa kun toteutetaan. Suunnittelu on osa ohjelmointiprosessia: ohjelman kehittyessä suunnittelu muuttuu. XP:ssä onnistuu evolutionaarinen suunnittelu, jos pidetään huolta testauskäytännöistä, jatkuvasta integraatiosta ja refaktoroinnista sekä ohjelman suunnittelusta iteraatioiden välillä.

Perinteinen suunnittelu on tämän vastakohta: kaikki on suunniteltu ennakolta niin tarkasti, että toteutuksen voi ulkoistaa. Perinteisestä muistutetaan, että mitä myöhäisemmässä vaiheessa tehdään suunnittelumuutoksia, sitä kalliimmaksi se tulee.

Koodin pitäminen yksinkertaisena ja patternien käyttö on suunnittelua. Patternien käyttö on toisen mielestä yksinkertaista, toisen ei. Patternit ja niiden käyttö oikeassa tilanteessa pitää opetella. Sovella sitä ensin yksinkertaisesti ja lisää tarpeen mukaan monimutkaisuutta. Tärkeintä on olla valmis refaktoroimaan koodia uudestaan ja uudestaan - myös patternien osalta. Arkkitehtoonisiinkin muutoksiin pitää olla valmis. Jos tiedosto toimii tänään, tietokanta voi olla huomenna parempi ratkaisu.UML:n ja muiden kaavioiden käyttöä suositellaan viestinnän välineinä korkeammalla tasolla. Koodi kertoo yksityiskohdat. Keskustelu on viestinnän ykkösmuoto. Suunnittelu ei ole kuollut, vaan se on muuttanut muotoaan siten, että suunnittelu muuttuu ohjelmoinnin aikana, jolloin suunnitelmien muuttaminen ei ole niin mullistavaa tai kallista kuin perinteisessä mallissa.