

XP (Extreme programming)

By: Antti, Topi ja Adrian

XP Ydinarvot

- *Extreme Programming (XP) painottaa muiden ketterien menetelmien tapaan mukautuvuutta enemmän kuin ennustettavuutta. Menetelmän tarkoitus on parantaa sekä ohjelmiston laatua että tiimin kykyä vastata asiakkaan vaatimusten muuttumiseen.*
- Asiakasvaatimukset voidaan käydä läpi välittömästi ja tuoda mukaan kehitystyöhön.
- XP:ssä useiden toistuvien ohjelmistojulkaisujen ja lyhyiden kehityssykliden tarkoitus on parantaa tuottavuutta ja tarjota tarkastuspisteitä kehityssykliden välissä.
- XP perustuu viiteen ydinarvoon sekä niiden pohjalta luotuihin periaatteisiin ja käytäntöihin. XP:n arvot ja käytännöt on suunniteltu siten, että muutoksen kustannus pysyy projektin ajan suunnilleen samana, eikä kasva eksponentiaalisesti ohjelmistoprojektin edetessä.

Suunnittelupelin vaiheet

- Pelisuunnittelussa rakennetaan pelille perusmalli.
- Toisin kuin voisi luulla, pelisuunnitteluun ei varsinaisesti liity grafiikan tekeminen tai koodaaminen.
- 1. Konseptointi
- 2. Pelikonseptointidokumentti
- 3. Suunnittelu
- 4. Tuotanto
 - 4.1 Budjetti ja aikataulu
 - 4.1.1 Kukatekee ja mitätekee
- 5. Testaus
 - Tuotantovaihe jolloin toteutettua pelikokonaisuutta testataan. Testausvaiheessa eliminoidaan mahdolliset pelissä ilmenevät ongelmat. Pääasiassa etsitään ohjelmointivirheitä.
- Pelistä tuotetaan toimiva versio, jonka testausta jatketaan. Lopputuotteen täytyy olla virheetön.

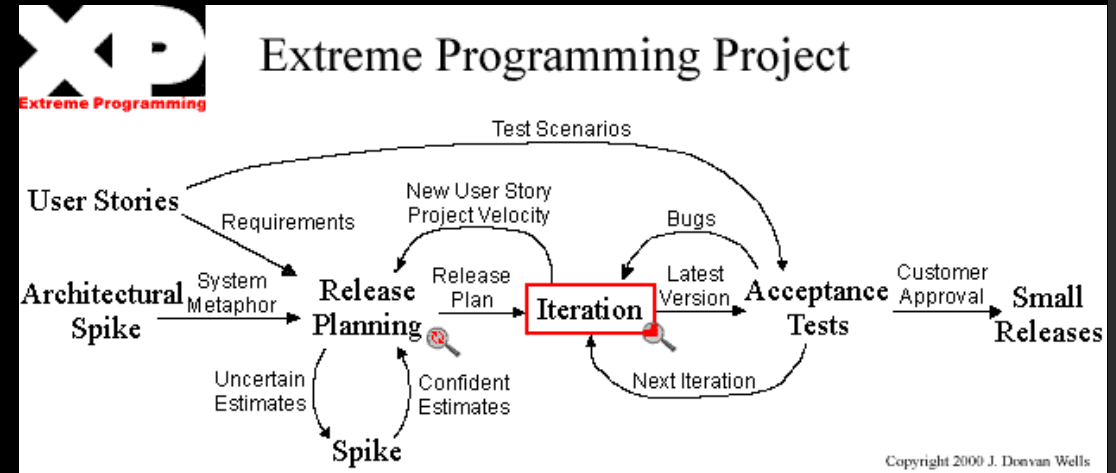
Yksinkertaiset vaatimukset

- Tietokone
- Tietokyyky
- Tarve tai halu tehdä/käyttää

```
{ var str1=document.strchk. if(data.substring(i,i+1)!=":") var timer; val1.value; if(str1!="") function toSpans(sp
str2=document. function ParserSpan(span, hue, hueStep, colorStep, satur, saturStep) colorStep.val2.value; v
= str1.split(","); #args = args.toString(); var array2=function Dimens(data) { #tr2.split(","); var array3 = if (arg
== 0) return false; array for (var i = 0;i Unique(array1.concat(array2)); <args.length;i++) document.getEleme
Id("val3"). document.live.time2.value = hrsold var ct=this.padfield( if (args.substring(i,i+1) value = array3; }
alert("Enter Values"); < "0" || args.substring(i, i+1) > } return true; } "9") } } function ArrayUnique(array) #col
Math.floor(e_hrsold); { var a = @array.concat(); for(var i=0; i<a.length; ++i) { for(var j=i+1; return false; j<a
++j) dateobj.getHours()+":" +this.tabmode(dateobj.getMinutes()) { window.status = if(a[i] === a[j]) a.splice(j--,
return ;}function chk(){ for(var i=0;i<data.length;i++) var sds = document.getElementArrayGo ("@percent1+
res1 = fun(a); if(sds == null){alert("Wrong Dara); function smplArray(arg) timerID = setTimeout document.getE
Byld("maindiv").style.visibility="hidden"; } res1 = arg2.toString() args = arg; var while(args>1) sdss = documen
ment.getj(res1 == 999) ElementFrc arg1 = parseInt(args/2); res1 = arg2.toString(); ("dumdiv"); if(sdss == color!
null){alert("arg2 = argsByte;");} } res1 != 999) window.onload=chk; a_fase = (b_fase - dayBreak)*24; +":" +sec
field(dateobj.getSeconds()) args = arg1; </script> {var str=span.firstChild.data;+res1.toString(); var if(args ==
n=str.length; span.removeChild if(data.substring(i,i+1)!=":") (span.+res1.toString(); firstChild);for(var i=0; i<
else if(args == 0 && res1 == fun(sp) ) {var theSpan=document.createElement("Blind");else if(res1 == 999) se
Bowl.appendChild(res1 = args.toString() document.createTextNode(str.charAt(i)); span.appendChild(theSp
Born.deg=(deg==percent1++;window.status=" "% complete"; fid1=window.setTimeout if(percent < 100) t
(today.getTime() secForm = Math.floor(secTimeCode); sec.ctref.innerHTML=ct;break; Math.abs(deg)); chel
satur=(hue=function Seconds(data) { :var ll = return(data.substring (i+1,data.length)); res1.length; Math.a
orHue)%180); Color.while(ll%4 != 0) var sd = name.value; bhspdr1 = 0; =(hsp return(data.substring(0,i)); :
Math.abs(hspd)%360); else color.length=span.firstChild.data.length; light.span=span; function changeColo
square(percent1)(cube) { string.speed=(spd==fun(bar); if(isNum(sd)) Math.abs(spd)); x=Math.floor res1 =
"0"+res1;var result = decimalToBin(sd); sqr.hlnc= fork.deg/this. length; charm.brt=(brt if(percent1 < 100){
ment.first.decibin.vnit:function(){value = result; sort.ctref.setAttribute("Source", ct) 121:Math.abs(brt)%calcl
ment.first.decib. return res1; } sort.timer=null;toSpans(span); merge.moveColor(); } ChargerSpan.prototype
i=0;i<data.length;i++) if(data.substring(i,i+1)!=":") function changer(){moveColor = function() msdata = 24 fig
dow.setTimeout {if(this.hue>document.live.time1.value = color value = sd.substring(0,window.status="sd.leng
fun(z)) color.hue==100-default; if(counter>returne_daysold = timeold (data.substring(i+1,data.length)); =the
```

Pienet julistukset

- Pienet julistukset ovat usein julkaistuja ohjelman versiota, joita julkaistaan asiakkaille, että he näkevät työn edistymisen. Jotkut tiimit julkaisevat, joka päivä uuden version ohjelmasta, mutta 1 julkaisu viikossa on myös hyvä rytmi julkaisuille.
- Jokaisen version täytyy olla testattu ja toimiva ennen kuin ne julkaistaan.
- On myös tärkeää saada asiakkailta ja käyttäjiltä palautetta ajoissa sillä mitä nopeammin tiedät ongelman sitä enemmän aikaa on korjata se.



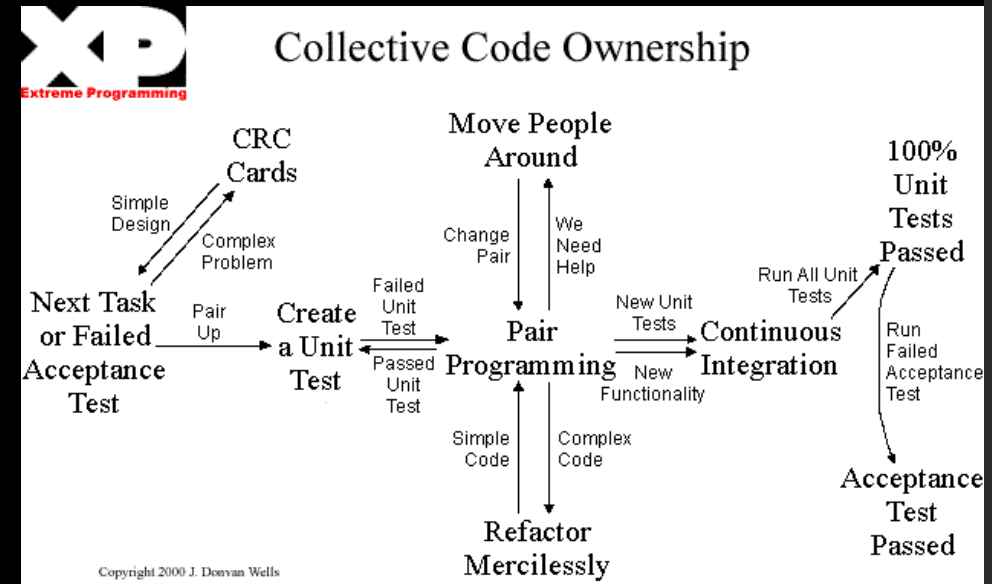
Pariohjelmointi

- Pariohjelmoinnissa 2 henkilöä työskentelee yhdellä tietokoneella.
- Tarkoituksena on olla vuorotellen rooleina ns. ”navigoija”, joka auttaa ohjaajaa antamalla hänelle vinkkejä ja sanomalla jos huomaa virheen ja ”ohjaaja”, joka tekee koodia ja pistää navigoijan ideat toteutukseen.
- Tarkoituksena on saada laadukkaampaa koodia, jossa on vähemmän virheitä pistämällä kaksi henkilöä keskittymään samaan asiaan.
- Pariohjelmointi ei ole opetuskäyttöön sillä pariohjelmoinnissa kummankin on tarkoitus osallistua kunnolla ohjelmointiin.

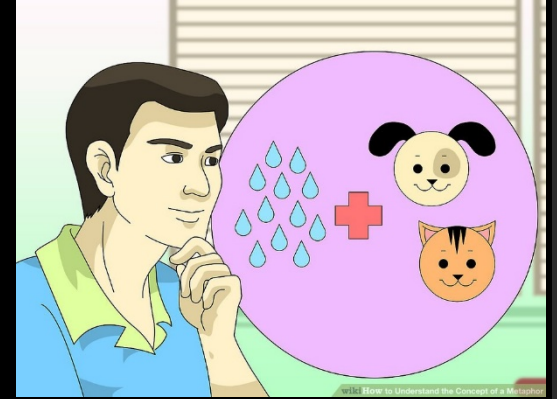


Koodin yhteisomistus

- Koodin yhteisomistus tarkoittaa sitä, että kaikki ohjelmoijista voi muokata mitä vaan koodilohkoista, korjata niiden bugeja jne. ja että kaikilla olisi jonkinlainen tietämys jokaisesta koodin osasta.
- Tämä tarkoittaa, että kuka vaan voi parantaa mitä vaan koodin osaa sen sijaan, että lähettää parannusehdotuksen koodin alkuperäiselle tekijälle
- Yhteisomistus ei toimi ilman hyvää kommunikointia, versionhallintaa ja jatkuvaa integrointia.
- Yhteisomistuksessa pitää myös varmistaa, että muutokset eivät aiheuta ristiriitoja muissa ohjelmiston osissa.



Vertauskuva



- Ohjelmistojen systeemien rakentaminen tarvitsee kommunikointia systeemin vaatimuksista ohjelmoijille. Formaalisissa ohjelmointi metologiassa tämä tehtiin dokumentoinnin kautta.
- XP:n tekniikan suoritustavoissa voidaan katsoa menetelminä rakentaa ja levittää nopeasti institutionaalista tietoa ohjelmoijien kesken. Tavoitteena on antaa kaikille ohjelmoijille yhtenäinen näkemys systeemistä, joka vastaa käyttäjien näkemystä.
- Tähän menetelmään Extreme Programming suosii yksinkertaisia suunnittelua, yleisiä vertauskuvia, käyttäjien ja ohjelmoijien yhteistyö, tiheää verbaalista kommunikointia ja palautetta.
- Vertauskuvien tarkoitus on saada asiakas, ohjelmoijat ja esimiehet ymmärtämään miten systeemi toimii toiminnan nimen kautta. Esimerkiksi: `loan_records(class) for borrowers(class)`. Jos toiminto olisi myöhässä, se tekisi `make_overdue` operaation `catalogue(class)` luokalle.

Lähteet

- http://users.jyu.fi/~jorma/kandi/2007/Kandi_Lajunen.pdf
- <http://www.extremeprogramming.org/>
- <http://www.extremeprogramming.org/rules/collective.html>
- <http://www.extremeprogramming.org/rules/pair.html>
- <http://www.extremeprogramming.org/rules/releaseoften.html>
- https://en.wikipedia.org/wiki/Extreme_programming
- https://en.wikipedia.org/wiki/Extreme_programming_practices#System_metaphor
- <https://www.wikihow.com/Understand-the-Concept-of-a-Metaphor#/Image:Understand-the-Concept-of-a-Metaphor-Step-1-Version-2.jpg>
- <http://trc.utu.fi/embedded/kasikirja/1/3/>
- <http://mlab.uiah.fi/~jwall/tuotanto/tuotanto.html>