Unified Process

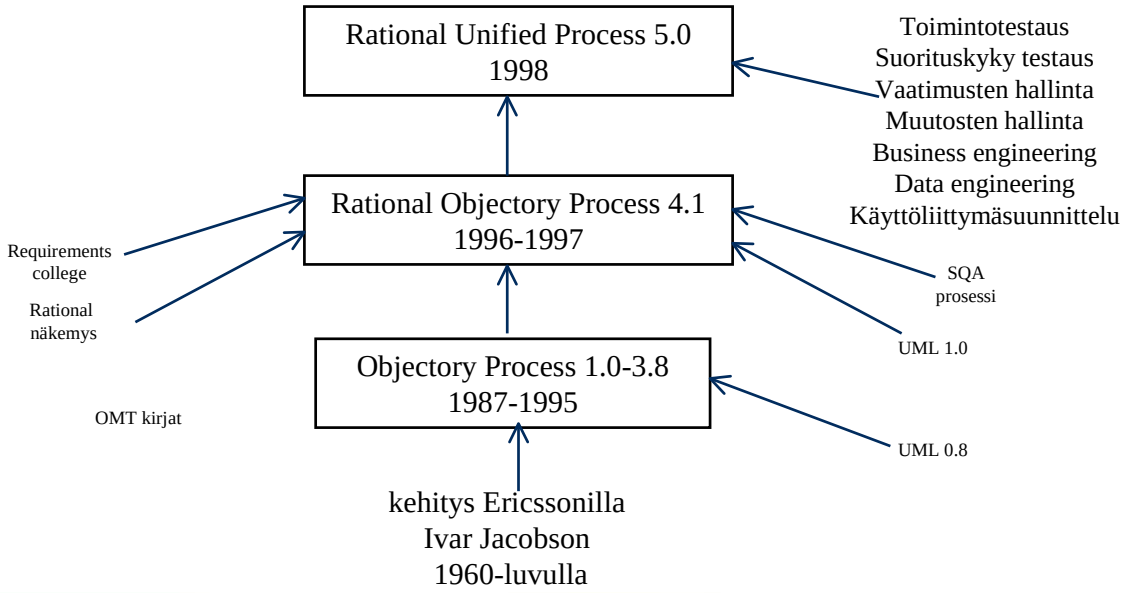
Hilkka Heikkilä

Scott Kendall(2002) The Unified Process Explained

Sisältö

- Historia
- Perusteita
- Perusperiaatteet:
  - Käyttötapausohjattu
  - Ohjelmistoarkkitehtuurikeskeinen
  - Iteratiivinen ja inkrementaalinen
- Iteraatio, inkrementti ja julkaisu
- Unified process kuvaus
- Tehtäväkokonaisuudet
- Iteraatiot ja tehtäväkokonaisuudet
- Viisi tehtäväkokonaisuutta:
  - Vaatimusten määrittely
  - Analyysi
  - Suunnittelu
  - Toteutus
  - Testaus
- Neljä vaihetta:
  - Aloitusvaihe (Inception)
  - Tarkentamisvaihe (Elaboration)
  - Konstruktiovaihe (Construction)
  - Luovutusvaihe (Transition)
- Unified process
  - onnistumiset ja haasteet
- RUP, Extreme Programming, Iconix process

Historiaa



Perusteita

- Peräkkäiset iteraatiot, joista jokainen muodostaa oman vesiputouksensa
  - Iteroiva vaihetuotteen kehittäminen
- Perusperiaatteet
  - Iteratiivinen ja inkrementaalinen
  - Käyttötapausohjattu
  - Ohjelmistoarkkitehtuurikeskeinen
- 4 päävaihetta, joiden sisällä on viisi tehtäväkokonaisuutta;
  - Tehtäväkokonaisuussykliä voidaan iteroida saman vaiheen sisällä useita kertoja
- Päävaiheet
  - Aloitusvaihe (inception)
    - Tuotekonseptin vaihtoehtojen kartoitus
  - Tarkentamisvaihe (elaboration)
    - Perusarkkitehtuurin kiinnittäminen
  - Konstruktiovaihe (construction)
    - Iteratiivinen (beta)testiversioiden toteutus, kehittäminen käyttökokemusten ja palautteen mukaan
  - Luovutusvaihe (transition)
    - Tuloksena saadaan paketoitu valmis järjestelmä
- Tehtäväkokonaisuudet
  - Vaatimusten määrittely (requirements)
  - Analyysi (analysis)
  - Suunnittelu (design)
  - Toteutus (implementation)
  - Testaus (test)

Perusperiaatteet 1/3

- **Käyttötapausohjattu**
  - Käyttötapaukset ohjaavat koko kehitystyötä; käyttötapauksien keräämisestä alkaen koodin tekemiseen ja testaukseen asti.
  - Järjestelmän käyttöön liittyvä työ- tai tehtäväkokonaisuus, joka tuottaa lisäarvoa käyttäjälle/käyttäjille, jossa toimijoina ovat käyttäjät
  - Kuvataan mitä toimintoja järjestelmältä vaaditaan. Kuvaus koostuu käyttäjistä ja käyttötapauksista käyttäjien ja järjestelmän välillä.
  - Käyttäjät eivät ole tietojärjestelmän osa. Ne voivat syöttää tietoa järjestelmään tai vastaanottaa tietoa järjestelmästä. Käyttäjät voivat olla paitsi ihmisiä, myös toisia tietojärjestelmiä, jotka kommunikoivat tämän järjestelmän kanssa ohjelmien välityksellä.
  - Hyötyjä
    - Kuvaa vaatimukset järjestelmän käyttäjän näkökulmasta
    - Käyttötapaukset kirjoitettu tavallisella kielellä – mahdollisuus ymmärtää paremmin
    - Vaatimusten jäljitettävyys paranee
    - Työ pystytään pilkkomaan palasiksi ja jakamaan osaprojekteille
  - Tavoitteita:
    - Looginen prosessi vankkaa järjestelmää kohti
    - Muuttuvien vaatimusten kanssa toimiminen
    - Taipuisa muuttamaan suunnitelmaa
    - Jatkuva integrointi
    - Aikainen ymmärrys
    - Jatkuva riskien seuranta/ fokusointi

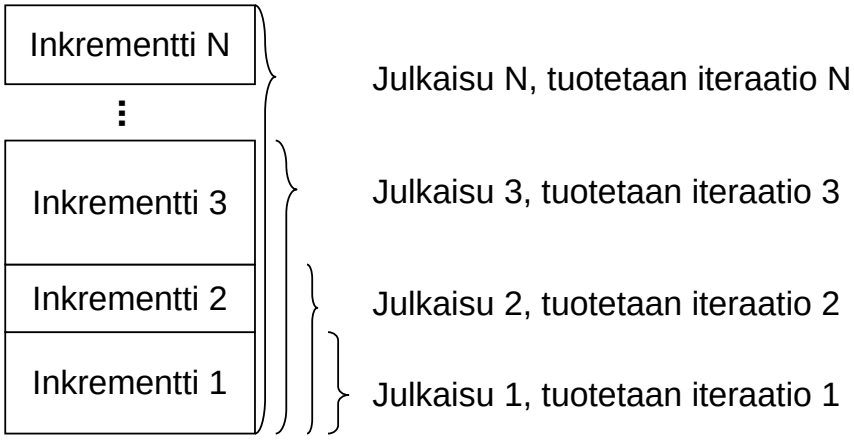
Perusperiaatteet 2/3

- **Ohjelmistoarkkitehtuurikeskeinen**
  - Arkkitehtuuri on välttämätön perusta, jolle järjestelmä rakennetaan
  - Käyttötapaukset kertovat tehtävän, arkkitehtuuri muodon – näitä kehitetään rinnakkain
  - Tavoite on rakentaa arkkitehtuuri joka mahdollistaa vaatimusten realisoinnin nyt ja tulevaisuudessa
  - Arkkitehtuuri on yleinen näkemys, jonka jokaisen tiimiläisen tulee jakaa, jotta tuotteesta tulisi vankka, taipuisa, laajennettava ja kannattava
  - Arkkitehtuuri määritellään aikaisessa vaiheessa, mutta se tarkentuu ja laajentuu projektin aikana
  - Tehdään erilaisia näkymiä järjestelmästä, että saadaan parempi kuva rakenteesta
  - Miksi ohjelmistoarkkitehtuuri on tärkeää UP:ssä
    - Auttaa ymmärtämään kokonaiskuva
    - Pystytään jakamaan ohjelmistokehityksen resurssit
    - Mahdollistaa uudelleenkäytön
    - Kehittää ohjelmistojärjestelmää jatkuvasti
    - Ohjaa käyttötapauksia

Perusperiaatteet 3/3

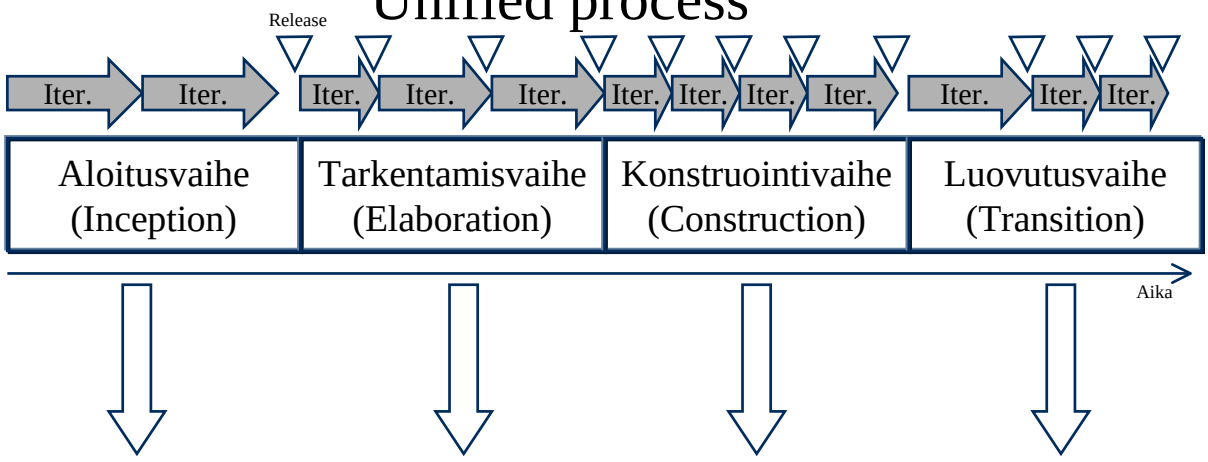
- **Iteraatio (Iteration)**
  - Miniprojekti (tietyt toiminnot), joka toteutetaan iteraatiosuunnitelman mukaan ja tuloksena on julkaisu (release) (ulkoinen tai sisäinen)
  - Iteraation vaiheet ovat analyysi, suunnittelu, toteutus ja testaus eli lähes koko elinkaari perinteisessä ohjelmistokehityksessä.
  - Iteraatioita voi olla monta tuotekehityksen elinkaaren aikana.
  - Täydentyvää kehittämistyötä (ratkaisua syventävää)
  - Ratkaisua koskevaa tietoa lisäävä ja syventävä työstäminen
  - Paranevan tietämyksen ja stabiloituvien piirteiden mukaan ottaminen järjestelmään
  - Palaute, ”concurrent engineering”
- **Inkrementti**
  - Pieni, suoritettava/toimiva osa järjestelmästä, joka voidaan testata oikeassa laitteistossa. Laajentaa tuotteen tähän astista toiminnallisuutta.
  - Toiminnallisuus
  - Lisäävää kehittämistyötä (ratkaisua laajentava)
  - Ominaisuuksia tuotteeseen lisäävä työstäminen
  - Vaiheittain kehittäminen
  - Ydinjärjestelmä+ inkrementit

Iteraatio, inkrementti ja julkaisu



- Julkaisu “N” koostuu koko toiminnallisuudesta, joka on kehitetty vastaavassa iteraatiossa “N” ja kaikissa edellisissä iteraatioissa
  - eli julkaisu 3 = inkrementti 1 + inkrementti 2 + inkrementti 3
- Iteraatio “N” tuottaa vastaavan inkrementin “N” ja ylläpitää edellisten inkrementtien toiminnallisuudet ("1", … , "N-1"). Eli tuottaa julkaisun “N”
- Iteraatio voi muuttaa (mahdollisesti myös poistaa) edellisten inkrementtien toiminnallisuuksia.

Unified process



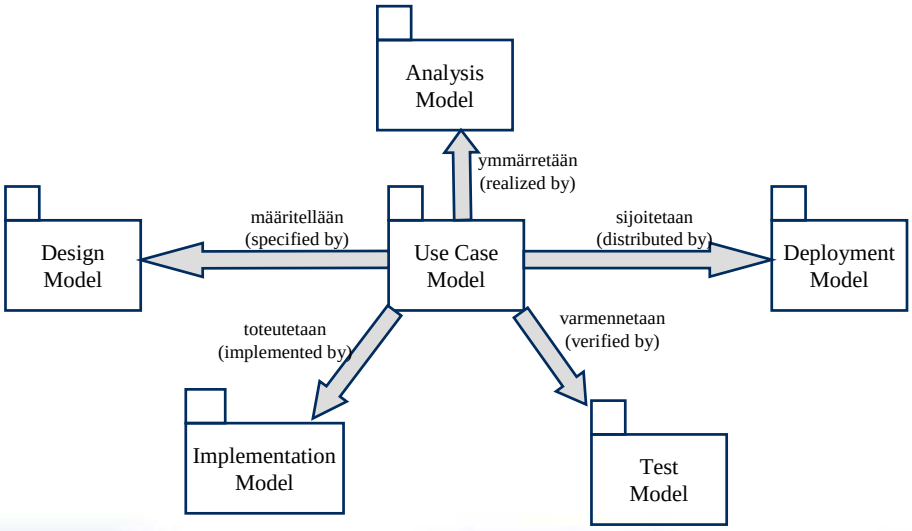
- Tuotteen ominaisuudet
- Alustavat mallit
- Alustava tuotearkkitehtuuri
- Tarvittaessa prototyyppi
- Riskit
- Alustava projektisuunnitelma
- Onnistumiskriteerit

- Täydennetyt mallit
- Toteutettu toimiva perusarkkitehtuuri
- Arkkitehtuurikuvaus
- Riskit
- Seuraavan vaiheen projektisuunnitelma
- Onnistumiskriteerit
- Alustava käyttöohje

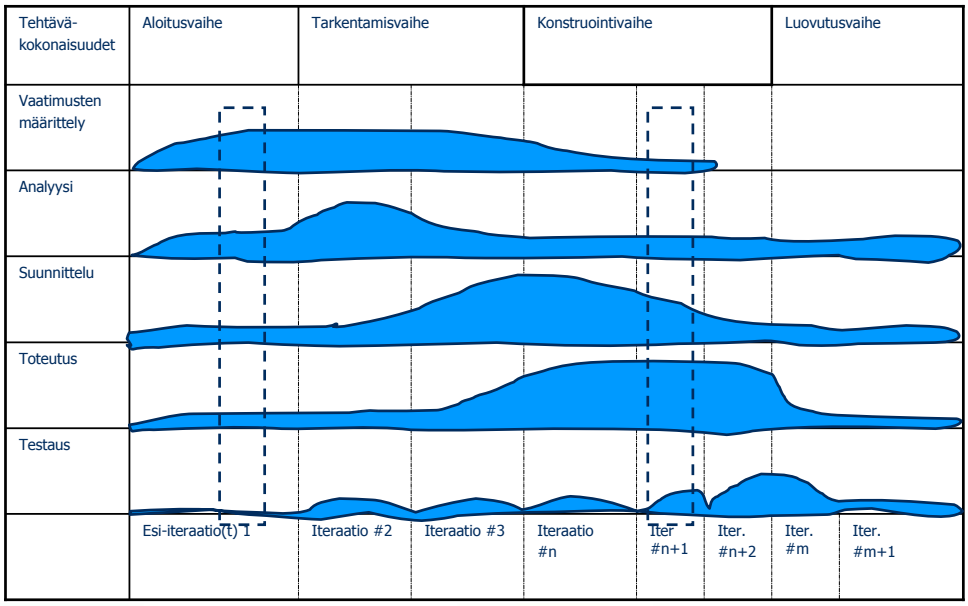
- Lähes täydelliset mallit
- Beta - versio
- Arkkitehtuurikuvaus
- Seuraavan vaiheen projektisuunnitelma
- Onnistumiskriteerit
- Käyttöohje

- Installointivalmis ohjelmisto
- Dokumentit
- Täydelliset mallit
- Arkkitehtuurikuvaus
- Käsikirjat
- www-palvelut yms.

Tehtäväkokonaisuudet



Iteraatiot ja tehtäväkokonaisuudet



Tehtäväkokonaisuudet

Vaatimusten määrittely (requirements)
Analyysi (analysis)
Suunnittelu (design)
Toteutus (implementation)
Testaus (test)

Vaatimusten määrittely 1/2

- Päämäärä
  - Saavuttaa yhteisymmärrys järjestelmän rakenteesta
  - Listata vaatimusehdokkaat
  - Tunnistaa ja neuvotella toiminnalliset vaatimukset
  - Määritellä ei-toiminnalliset vaatimukset
- Työn tekijät
  - Järjestelmäsuunnittelija
  - Käyttötapausmäärittelijä
  - Käyttöliittymä suunnitelija
  - Arkkitehti
- Aktiviteetit
  - Rakenna aluemalli
  - Rakenna bisnesmalli
  - Etsi toimijat ja käyttötapaukset
  - Tee käyttöliittymän prototyyppi
  - Priorisoi käyttötapaukset
  - Kirjaa tarkemmin käyttötapaukset
  - Rakenna käyttötapausmalli

Vaatimusten määrittely 2/2

- Tuotokset (artifacts)
  - Aluemalli (Domain model)
    - Tärkeimmät todellisuuden asiat ja käsitteet, joita uusi järjestelmä käsittelee
    - Asiat ja käsitteet esitetään luokkina (class) ja niiden välisinä suhteina
  - Bisnesmalli (Business model)
    - Bisnes käyttötapausmalli ja bisnes objektimalli
    - Bisnes käyttötapausmalli koostuu bisnesprosesseista ja bisnes toimijoista (actors), joita ovat asiakkaat ja partnerit
    - Bisnes objektimalli selittää kuinka bisnes käyttötapausmallit ymmärretään työn tekijöiden kannalta käyttäen bisnesolioita (esim. tilit, lomakkeet)
  - Toimija (actor)
    - Ihminen, toinen järjestelmä, tietokanta jne.
  - Käyttötapaus (use case)
    - Sarja toimenpiteitä, joita toimija tekee yhdessä järjestelmän kanssa saavuttaakseen tietyn päämäärän
    - MITÄ järjestelmän pitää tehdä
    - Normaali sarja toimenpiteitä + poikkeuksellinen sarja toimenpiteitä
  - Käyttöliittymä prototyyppi (user interface)
  - Käyttötapausmalli
    - Koottu käyttötapauksista, joista jokainen sisältää toimijoita ja käyttötapauksia
  - Arkkitehtuurikuvaus
    - Näkymä käyttötapausmallista
  - Lisävaatimukset
    - Ei-toiminnalliset vaatimukset: suorituskyky, turvallisuus, varmuuskopiointi…
  - Sanasto (glossary)
    - Samanlainen käsitys käytetyistä termeistä

Analyysi 1/2

- Päämäärä
  - Todellinen ymmärrys asiakasvaatimuksista
  - Tuottaa analyysimalli
  - Tuotekehityksen paino alkaa siirtymään pois asiakkaista kohti kehittäjien tarpeita, jotta järjestelmä rakennettaisiin oikein
- Työn tekijät
  - Arkkitehti
  - Käyttötapaus insinööri
  - Komponentti insinööri
- Aktiviteetit
  - Tee arkkitehtuuri analyysi
  - Analysoi käyttötapaus
  - Analysoi luokka
  - Analysoi paketti

Analyysi 2/2

- Tuotokset
  - Analyysiluokka
    - Sisältää ominaisuuksia, mutta ei toimintoja
    - Rajapinta (boundary class), entiteetti (entity class) tai ohjaus (control class)
    - Rajapinnan kanssa työn tekijä on vuorovaikutuksessa
    - Entiteetti sisältää pitkäikäistä tietoa
    - Ohjaus ilmentää sovelluslogiikkaa
  - Käyttötapaus realisointianalyysi
    - Kuinka työn tekijät ja järjestelmä toimivat käyttötapauksessa analyysiluokkien avulla kerrottuna
    - Tehdään kestävyysanalyysi (robustness analysis), jossa analysoidaan käyttötapaus lause lauseelta ja määritellään objektit ja niiden väliset suhteet
  - Analyysipaketti
    - UML paketti, joka sisältää analyysi luokat ja käyttötapauksen realisoinnin
  - Analyysimalli
    - Koostuu analyysipaketeista
  - Arkkitehtuurikuvaus
    - Analyysimallin näkökulmasta

Suunnittelu 1/2

- Päämäärä
  - Sisältää päätökset objektien jakautumisesta, samanaikaisuudesta, tietokannoista, käyttöliittymä, tapahtumista jne.
  - Tuottaa suunnittelumalli ja sijoittelumalli
- Työn tekijät
  - Arkkitehti
  - Käyttötapausinsinööri
  - Komponentti-insinööri
- Aktiviteetit
  - Tee arkkitehtuurisuunnittelu
  - Suunnittele käyttötapaus
  - Suunnittele luokka
  - Suunnittele alijärjestelmä

Suunnittelu 2/2

- Tuotokset
  - Suunnitteluluokka
    - Sisältää enemmän määreitä ja toimintoja
    - Käytetään termejä sillä ohjelmointikielellä, jolla toteutus tehdään
  - Käyttötapaus toteutussuunnittelu
    - Kuvaa kuinka aktorit ja järjestelmä suorittaa annetun käyttötapauksen
  - Rajapinta (interface)
    - Kokoelma toimintoja jotka edustavat palveluita, joita luokka, alijärjestelmä tai komponentti tarjoaa
  - Suunnittelualijärjestelmä
    - UML: suunnitteluluokat, luokat, alijärjestelmän rajapinnat ja käyttötapaus toteutussuunnittelu
    - Erikoistapaus palvelun alijärjestelmä
  - Suunnittelumalli
  - Arkkitehtuurikuvaus
    - Suunnittelumallin näkökulmasta
    - Sijoittelumallin näkökulmasta
  - Sijoittelumalli
    - Määrittelee fyysisen järjestelmän organisaation, tietokoneen solmujen (node) avulla

Toteutus 1/2

- Päämäärä
  - Tehdä toimiva versio järjestelmästä, joka voidaan toimittaa beta-asiakkaille arvioitavaksi
- Työn tekijät
  - Arkkitehti
  - Komponentti-insinööri
  - Järjestelmäintegroija
- Aktiviteetit
  - Suorita arkkitehtuurin toteutus
  - Toteuta luokka
  - Toteuta alijärjestelmä
  - Suorita yksikkötesti (unit test)
  - Integroi järjestelmä

Toteutus 2/2

- Tuotokset
  - Komponentti
    - Fyysinen ja vaihdettavissa oleva järjestelmän osa, jolla on tietyt rajapinnat
  - Rajapinta
  - Alijärjestelmän toteutus
  - Toteutusmalli
  - Arkkitehtuurikuvaus
    - Toteutusmallin näkökulmasta
  - Integrointisuunnitelma

Testaus 1/2

- Päämäärä
  - Varmistua, että järjestelmä on laadukas
  - Varmistua, että järjestelmä toteuttaa asetetut vaatimukset
- Työn tekijät
  - Testausinsinööri
  - Komponentti-insinööri
  - Integrointitestaaja
  - Järjestelmätestaaja
- Aktiviteetit
  - Suunnittele testaus (yhdelle iteraatiolle, koko järjestelmälle)
  - Toteuta testaus
  - Suorita integrointitestaus
  - Suorita järjestelmätestaus
  - Analysoi testaus

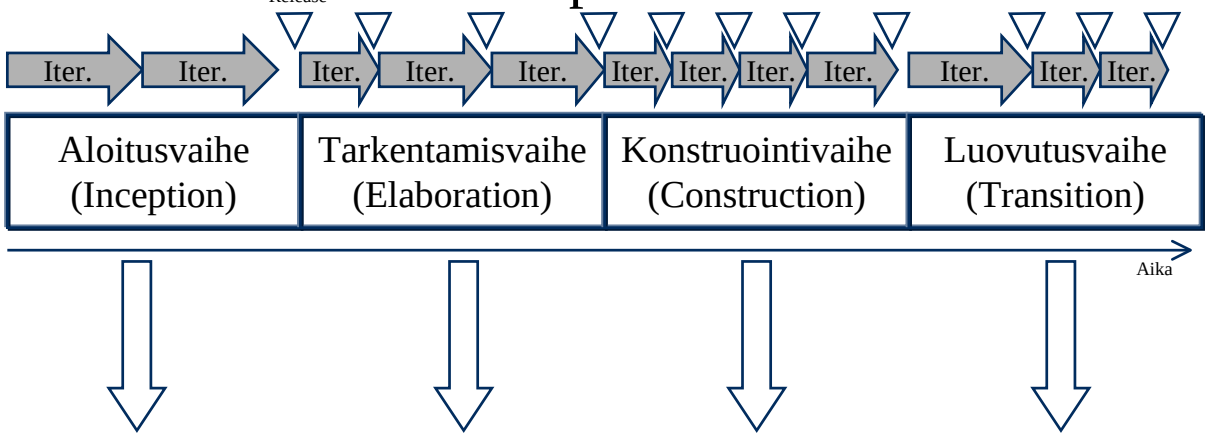
Testaus 2/2

- Tuotokset
  - Testitapaus
    - Syöttötiedot, tulokset, edellytykset joita testissä tarvitaan
    - Black-box: testitapaukset johdetaan suoraan käyttötapauksista
    - White-box: testitapaukset johdetaan käyttötapauksien toteutussuunnittelusta
    - Integrointitestaus ja järjestelmätestaus
  - Testitoteutus
  - Testauskomponentti
    - Testitapausten automatisointi
  - Testausmalli
  - Testisuunnitelma
  - Virhe (defect)
  - Testitulosten analysointi - testiraportti

Työn tekijät ja tehtäväkokonaisuudet

Työn tekijät	Vaatimusten määrittely	Analyysi	Suunnittelu	Toteutus	Testaus
Järjestelmä-suunnittelija	Rakenna aluemalli Rakenna bisnesmalli Etsi toimijat ja käyttötapaukset Rakenna käyttötapausmalli				
Käyttötapaus-määrittelijä	Tarkenna käyttötapaukset				
Käyttöliittymä-suunnittelija	Tee käyttöliittymän prototyyppi				
Arkkitehti	Priorisoi käyttötapaukset Päivitä arkkitehtuurikuvaus	Tee arkkitehtuuri-analyysi	Tee arkkitehtuuri-suunnittelu	Suorita arkkitehtuuri-toteutus	
Käyttötapaus-insinööri		Analysoi käyttötapaus	Suunnittele käyttötapaus		
Komponentti-insinööri		Analysoi luokka Analysoi paketti	Suunnittele luokka ja paketti	Toteuta luokka Suorita yksikkötesti Toteuta alijärjestelmä	Suunnittele testaus Analysoi onnistuminen
Järjestelmä-integroija				Integroi järjestelmä	
Testaus-insinööri					Toteuta testaus
Integrointi-testaaja					Toteuta integrointi-testaus
Systeemi-testaaja					Toteuta systeemi-testaus

Unified process



- Tuotteen ominaisuudet
- Alustavat mallit
- Alustava tuotearkkitehtuuri
- Tarvittaessa prototyyppi
- Riskit
- Alustava projektisuunnitelma
- Onnistumiskriteerit

- Täydennetyt mallit
- Toteutettu toimiva perusarkkitehtuuri
- Arkkitehtuurikuvaus
- Riskit
- Seuraavan vaiheen projektisuunnitelma
- Onnistumiskriteerit
- Alustava käyttöohje

- Lähes täydelliset mallit
- Beta - versio
- Arkkitehtuurikuvaus
- Seuraavan vaiheen projektisuunnitelma
- Onnistumiskriteerit
- Käyttöohje

- Installointivalmis ohjelmisto
- Dokumentit
- Täydelliset mallit
- Arkkitehtuurikuvaus
- Käsikirjat
- www-palvelut yms.

Päävaiheet

Aloitusvaihe (inception)
Tarkentamisvaihe (elaboration)
Konstruktiovaihe (construction)
Luovutusvaihe (transition)

Aloitusvaihe

- Päämääränä määritellä tietyn järjestelmän bisnestapaus
- Ensimmäisen version perusteella sitoudutaan kehitysprojektiin, myöhemmillä perustellaan projektin jatkamista
- Määritellään mikä on järjestelmän sisältö ja miten se toimii ympäristönsä kanssa
- Tutki korkean tason toiminnalliset ja ei-toiminnalliset vaatimukset
- Kokoa arkkitehtuuriehdotus
- Tunnista kriittisimmät riskit, joita projekti voi kohdata ja määrittele miten projekti tiimi niiden kanssa toimii
- Määrittele muutamia arvoja, esimerkiksi tuottavuus (ROI), jotka perustelevat projektia taloudelliselta kannalta
- Ei ole tarkoitus olla kovin aikaa vievää ja kallista tutkimusta
- Ajatus on saada järjestelmästä tarpeeksi tietoa, jotta tiedetään kannattaako sitä ruveta kehittämään
- Tarkenna bisnestapausta ja projektisuunnitelmaan aloitusvaiheen tulosten perusteella
- Isoin ja tärkein tehtäväkokonaisuus tässä vaiheessa on Vaatimusten määrittely
- Korkeintaan kaksi iteraatiota tässä vaiheessa

Aloitusvaihe

- Projektipäällikön tehtävät ennen muita tehtäväkokonaisuuksia
  - Suunnittele aloitusvaihe tarkemmin ja muut vaiheet abstraktimmin -> projektisuunnitelman ensimmäinen versio (budjetti, aikataulu, resurssit, iteraatiot)
  - Määrittele ja sovi yhdessä asiakkaiden kanssa projektin visio tarkemmin
  - Määrittele arviointikriteerit tälle vaiheelle seuraavilta alueilta
    - Kattavuus, vaikutusala
    - Ylemmän tason vaatimukset
    - Arkkitehtuuriehdotus
    - Kriittiset riskit
    - Bisnestapaus
- Vaatimusten määrittely
  - Rakenna aluemalli
    - Tarkoituksena päästä yhteisymmärrykseen järjestelmän koostumuksesta, ”mitä menee sisään ja mitä tulee ulos”
  - Rakenna bisnesmalli
    - Helpottaa tunnistamaan päätoimijat ja käyttötapaukset
    - Saada katsaus korkeantason bisnesprosesseihin
  - Etsi toimijat ja käyttötapaukset
  - Priorisoi käyttötapaukset
    - Kohdistuvat kriittisiin toiminnallisuuksiin
    - Suurimmat riskit
  - Tarkenna käyttötapauksia
    - Huomio lähinnä miten toimii tässä vaiheessa
- Analyysi
  - Tee arkkitehtuurianalyysi
  - Analysoi käyttötapaukset
    - Vain ne käyttötapaukset, joita tarvitaan arkkitehtuuriehdotuksen määrittelyyn
    - Esim. kestävyysanalyysi (robustness analysis)
- Suunnittelu
  - Tee arkkitehtuurisuunnittelu

Aloitusvaihe

- Tuotokset
  - Projektisuunnittelu
    - Ominaisuuslista
    - Ensimmäinen versio projektisuunnitelmasta
  - Vaatimusten määrittely
    - Riskilista
    - Aluemalli
    - Bisnesmalli
    - Käyttötapausmalli
    - Arkkitehtuuriehdotuksen kuvaus
    - Käyttötapauksien priorisoitu lista
  - Analyysi
    - Analyysimalli
  - Suunnittelu
    - Sijoittelumalli
    - Suunnittelumalli
- Iteraatioiden lopussa ja koko aloitusvaiheen lopussa mietittävää
  - Onko kaikille asiakkaille ja osakkaille selvää millainen järjestelmä tullaan rakentamaan?
  - Ymmärtääkö jokainen ja on samaa mieltä korkeamman tason vaatimuksista?
  - Onko arkkitehtuuriehdotus tarpeeksi hyvä?
  - Onko kriittiset riskit huomioitu ja suunnitelmat niiden varalta tehty?
  - Onko taloudellista jatkaa projektia?
- Projektipäällikön tehtävät ennen seuraavaa tarkentamisvaihetta
  - Visio tulisi laajentaa bisnestapaukseksi
    - Perustelee projektin jatkumisen tarkentamisvaiheeseen
    - Taloudelliset arviot ovat aika karkeita, lyhytjänteisiä ja ei kovinkaan tarkkoja yksityiskohdiltaan tässä vaiheessa

Tarkennusvaihe

- Päämääränä löytää suurin osa toiminnallisista vaatimuksista ja tuottaa vakaa arkkitehtuuripohja
- Määritä kaikille toiminnallisiin vaatimuksiin liittyvät käyttötapaukset
- Luo perusarkkitehtuuri
- Siirrä huomio kriittisistä riskeistä merkittäviin riskeihin (esim. vaikuttavat aikataulun lipsumiseen ja budjetin ylittämiseen)
- Määrittele järjestelmän laatutaso ei-toiminnallisten vaatimusten suhteen (esim. luotettavuus ja vasteaika)
- Tarkenna bisnestapausta ja projektisuunnitelmaan tarkennusvaiheen tulosten perusteella (mm. aikataulu ja kustannukset)
- Painopiste määrittyy sen mukaan missä on merkittävimmät riskit
  - Jos tekniset riskit ovat suuret -> arkkitehtuuri tärkeää
  - Jos vaatimukset ovat epäselviä -> vaatimuksien määrittely ja käyttötapausten määrittely tärkeää
- Suurin osa analyysi tehtäväkokonaisuudesta tapahtuu tässä vaiheessa
- Suunnitteluvaihe on analyysivaiheen ohella toinen tärkeä tehtäväkokonaisuus tarkennusvaiheessa

Tarkennusvaihe

- Projektipäällikön tehtävät ennen muita tehtäväkokonaisuuksia
  - Suunnittele tarkennusvaihe tarkemmin ja muut vaiheet abstraktimmin -> projektisuunnitelma (budjetti, aikataulu, resurssit, iteraatiot)
  - Visio tulisi laajentaa bisnestapaukseksi
    - Perustelee projektin jatkumisen tarkentamisvaiheeseen
    - Taloudelliset arviot ovat aika karkeita, lyhytjänteisiä ja ei kovinkaan tarkkoja yksityiskohdiltaan tässä vaiheessa
  - Määrittele arviointikriteerit tälle vaiheelle seuraavilta alueilta
    - Vaatimukset
    - Perusarkkitehtuuri
    - Merkittävimmät riskit
    - Bisnestapaus
- Vaatimusten määrittely
  - Rakenna aluemalli
    - Tarkoituksena saada valmiiksi tarkennusvaiheen aikana järjestelmän koostumus, ”mitä menee sisään ja mitä tulee ulos”
  - Rakenna bisnesmalli
    - Tarkoituksena saada valmiiksi tarkennusvaiheen aikana
  - Etsi toimijat ja käyttötapaukset
  - Käyttöliittymän prototyyppi
  - Priorisoi käyttötapaukset
    - Mitä missäkin iteraatiossa tehdään
  - Tarkenna käyttötapauksia
    - Normaali ja vaihtoehtoiset (vikatapaukset ja vähemmän tyypilliset) reitit
  - Rakenna käyttötapausmalli
- Analyysi
  - Tee arkkitehtuurianalyysi
  - Analysoi käyttötapaukset
    - Kaikki iteraation käyttötapaukset
    - Esim. kestävyysanalyysi (robustness analysis)
  - Analysoi luokka
  - Analysoi paketit
- Suunnittelu
  - Tee arkkitehtuurisuunnittelu
  - Suunnittele käyttötapaus
  - Suunnittele luokka
  - Suunnittele alijärjestelmä
- Toteutus
  - Tee arkkitehtuuritoteutus
    - Mitkä komponentit tarvitaan iteraatiossa tehtäviin alijärjestelmiin
    - Komponenttien ja solmujen yhdistäminen
  - Toteuta luokka
  - Tee yksikkötesti (unit test)
    - Sekä black-box että white-box
  - Toteuta alijärjestelmä
  - Integroi järjestelmä
- Testaus
  - Suunnittele testaus
  - Toteuta testaus
  - Suorita integrointitestaus
  - Suorita järjestelmätestaus
  - Arvioi testitulokset

Tarkennusvaihe

- Tuotokset
  - Jokaisella iteraatiokerralla
    - Suoritettava perusarkkitehtuuri
  - Projektisuunnittelu
    - Projektisuunnitelma
    - Riskilista
  - Vaatimusten määrittely
    - Aluemalli
    - Bisnesmalli
    - Käyttötapausmalli
    - Arkkitehtuurikuvaus
  - Analyysi
    - Analyysimalli
    - Arkkitehtuurikuvaus
  - Suunnittelu
    - Käyttöönottomalli
    - Suunnittelumalli
  - Toteutus
    - Toteutusmalli
  - Testaus
    - Testimalli
- Iteraatioiden lopussa
  - Pitäisi keskittyä arkkitehtuuriin
  - Jokaisen arkkitehtuurikuvaukseen kuuluvan mallin pitäisi kehittyä jokaisen iteraation aikana
- Tarkennusvaiheen arviointi kokonaisuutena
  - Ovatko kaikki asiakkaat ja osakkaat samaa mieltä projektin suunnasta?
  - Ymmärtääkö jokainen ja on samaa mieltä tarkennetuista vaatimuksista?
  - Onko perusarkkitehtuuri vakaa, toteutettavissa ja kehittymiskelpoinen kun vaatimukset kehittyvät ja ominaisuuksia lisätään?
  - Onko merkittävät riskit huomioitu ja suunnitelmat niiden varalta tehty?
  - Onko taloudellista jatkaa projektia?
- Projektipäällikön tehtävät ennen seuraavaa tarkentamisvaihetta
  - Todellisen bisnestapauksen määrittely: aikataulut, resurssit, kustannukset pitäisi olla hyvin tiedossa

Konstruktiovaihe

- Päämääränä on tuottaa toimiva versio järjestelmästä beta-asiakkaille
- Viimeistele käyttötapausmalli (kaikki toiminnalliset vaatimukset on sovittu ja sovitettu järjestelmässä)
- Viimeistele kaikki muutkin mallit (analyysi, suunnittelu, käyttöönotto, toteutus ja testaus)
- Päivitä arkkitehtuurikuvaus vastaamaan tuotettavaa järjestelmää
- Seuraa kriittisiä (vaikuttaa järjestelmän elinkelpoisuuteen) ja merkittäviä (vaikuttaa budjettiin, aikatauluun tai kumpaankin) riskejä
- Suurin osa työstä konstruktiovaiheessa menee toteutus tehtäväkokonaisuuteen
- Suurin osa integrointi – ja järjestelmätestauksesta tehdään konstruktiovaiheessa

Konstruktiovaihe

- Projektipäällikön tehtävät ennen muita tehtäväkokonaisuuksia
  - Suunnittele konstruktiovaihe tarkemmin ja luovutusvaihe abstraktimmin -> projektisuunnitelma (budjetti, aikataulu, resurssit, iteraatiot)
  - Määrittele arviointikriteerit tälle vaiheelle seuraavilta alueilta
    - Onko järjestelmä tarpeeksi kypsä ja vakaa julkaistavaksi betaversiona?
    - Osaavatko betakäyttäjät käyttää järjestelmää eli onko tarpeeksi dokumentaatiota käyttäjien tueksi?
- Vaatimusten määrittely
  - Etsi toimijat ja käyttötapaukset
  - Käyttöliittymän prototyyppi
  - Priorisoi käyttötapaukset
    - Mitä missäkin iteraatiossa tehdään
  - Tarkenna käyttötapauksia
    - Normaali ja vaihtoehtoiset (vikatapaukset ja vähemmän tyypilliset) reitit
  - Rakenna käyttötapausmalli
- Analyysi
  - Tee arkkitehtuurianalyysi
  - Analysoi käyttötapaukset
    - Kaikki iteraation käyttötapaukset
    - Esim. kestävyysanalyysi (robustness analysis)
  - Analysoi luokka
  - Analysoi paketit
- Suunnittelu
  - Tee arkkitehtuurisuunnittelu
  - Suunnittele käyttötapaus
  - Suunnittele luokka
  - Suunnittele alijärjestelmä
- Toteutus
  - Toteuta luokka
  - Tee yksikkötesti (unit test)
    - Sekä black-box että white-box
  - Toteuta alijärjestelmä
  - Integroi järjestelmä
- Testaus
  - Suunnittele testaus
  - Toteuta testaus
  - Suorita integrointitestaus
  - Suorita järjestelmätestaus
  - Arvioi testitulokset

Konstruktiovaihe

- Tuotokset
  - Suoritettava järjestelmä
  - Projektisuunnittelu
    - Projektisuunnitelma
  - Vaatimusten määrittely
    - Käyttötapausmalli
    - Arkkitehtuurikuvaus
  - Analyysi
    - Analyysimalli
  - Suunnittelu
    - Käyttöönottomalli
    - Suunnittelumalli
  - Toteutus
    - toteutusmalli
  - Testaus
    - testimalli
- Iteraatioiden lopussa
  - Pitäisi keskittyä koko järjestelmään kokonaisuutena
  - Jokaisessa järjestelmän koonnissa tulisi järjestelmän parantua
- Tarkennusvaiheen arviointi kokonaisuutena
  - Onko järjestelmä valmis julkaistavaksi beta-asiakkaille?
- Projektipäällikön tehtävät ennen seuraavaa luovutusvaihetta
  - Todellisen bisnestapauksen määrittely: aikataulut, resurssit, kustannukset pitäisi tarkentaa viimeisten tietoen mukaisiksi

Luovutusvaihe

- Päämääränä on tuottaa toimiva versio järjestelmästä kaikille asiakkaille
- Valmistaudu järjestelmän käyttöönottoon
- Valmista dokumentointi
- Käyttöönotto
- Ohjelmiston virittäminen asiakkaiden mukaan
- Kerää tietoja vioista ja korjaa niitä
- Tehtäväkokonaisuudet eivät tässä vaiheessa näy Unified process – kuvauksessa
- Järjestelmän pitäisi olla toimivassa kunnossa tässä vaiheessa
- Yleensä vain yksi iteraatio

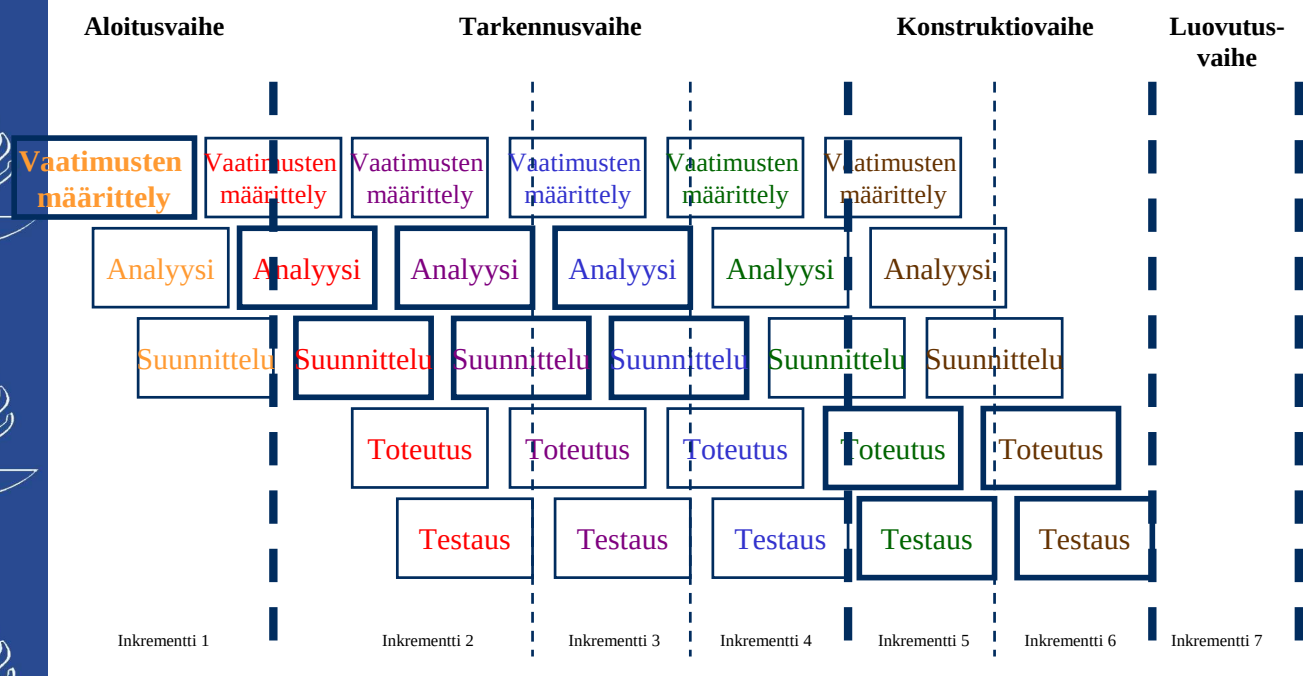
Luovutusvaihe

- Projektipäällikön tehtävät ennen muita tehtäväkokonaisuuksia
  - Suunnittele konstruktiovaihe tarkemmin -> projektisuunnitelma (budjetti, aikataulu, resurssit, iteraatiot)
  - Määrittele arviointikriteerit tälle vaiheelle seuraavilta alueilta
    - Onko käyttäjän hyväksymistestit läpäisty?
    - Onko dokumentointi oikeata ja hyödyllistä?
    - Onko käyttäjät yleisesti ottaen tyytyväisiä tuotteeseen?
- Muut tehtävät
  - Julkaise Betajulkaisu
    - Tarvittava dokumentaatio
    - Asiakastuki
  - Asenna Betajulkaisu
    - Miten vikojen löytymisestä raportoidaan?
  - Reagoi testituloksiin
    - Muutokset järjestelmässä muistettava päivittää malleihinkin
  - Viritä tuote erilaisiin käyttäjäympäristöihin
    - Kansainvälisyys (kieli, valuutta, lait ja säännöt, politiikka)
    - Platform ja infrastruktuuri
    - Tiedon siirtäminen ja muuntaminen olemassaolevista systeemeistä
  - Viimeistele tuotokset
    - Käyttäjä ja operaattori käsikirjat
    - On-line apu
    - Koulutusmateriaali
    - Mallit (analyysi, suunnittelu, käyttötapaus, sijoittelu, toteutus ja testaus)
    - Arkkitehtuurikuvaus

Luovutusvaihe

- Tuotokset
  - Suoritettava ohjelmisto
  - Projektisuunnitelma
  - Asennusohjelmisto
  - Lakisääteinen dokumentaatio
  - Käyttäjä dokumentaatio (ulkoinen)
  - Opetusmateriaali (ulkoinen)
  - Eri mallit (sisäinen)
  - Arkkitehtuurikuvaus (sisäinen)
- Luovutusvaiheen arviointi kokonaisuutena
  - Toimiiko järjestelmä tyypillisimmissä käyttäjäympäristöissä?
  - Onko virittäminen ei-tyypillisiin ympäristöihin selkeä tehdä?
- Projektipäällikön tehtävät lopussa
  - Verrata budjetoitua bisnestapausta toteutuneisiin lukuihin asiakkaiden ja omistajien kanssa
    - Tallettaa vertailu tulevien projektien käyttöön
    - Analysoida miten projekti saavutti tavoitteet käyttäjätyytyväisyys ja kannattavuus mielessä
  - Analysoi projektin kulku
    - Kaikki mitattavat suureet projektista
    - Myös abstraktit asiat (esim. tiedon kulku)
    - Painopistealueita
      - Mahdollinen projektin tuotosten uudelleenkäyttö
      - Koulutus ja mentorointi
      - Prosessi itsessään

Esimerkkiprosessi



Unified Process

Onnistumiset

- Riskien pienentäminen
- Tarkemmat työmäärä- ja aikatauluarviot
- Ongelmat huomataan aikaisemmin
- Vaatimusten muuttaminen
- Projektitiimin yhtenäisyys
- Palautetta eri vaiheiden, ja siten eri tekijöiden, välillä
- Projekti voidaan aloittaa pienemmillä resursseilla
- Mahdollista tuottaa prototyyppi aikaisessa vaiheessa
- Parempi yhteistyö asiakkaan kanssa
- Virheistä oppiminen
- Joustavuus
- Jatkuva integrointi
- Kehittyminen on näkyvää

Haasteita

- Kommunikointi
- Lisääntynyt projektin hallinta
- Kokonaisuuden hallinta ja samalla aikaa inkrementtaalisten osasten hallinta
- Epäselvät vaatimukset
- Julkaisujen lisääntyminen
  - Ylläpito ja vikojen korjaus jokaiselle julkaisulle
  - Integroinnin lisääntyminen
- Dokumentoinnin lisääntyminen
- Työtaakka on lisääntynyt, työ on uuvuttavaa
  - 2 inkrementtiä menossa samaan aikaa – ei hengähdystaukoa
  - Aikataulutavoitteet lyhyillä väleillä
- Uudet haasteet testaukselle
- Työt voidaan helposti siirtää

RUP, XP ja ICONIX

The Rational Unified Process

- Hyvin yksityiskohtainen eritystapaus Unified prosessista, joista voi valita itselleen sopivimmat
- Rationalin myymä tuote, joka sisältää
  - Laajat HTML sivut – elektronisesti jaossa, modulaarinen
  - Työkalumentoreita, jotka antavat tukea Rationalin työkalusetin käyttäjille
  - Templaatit suurimmalle osalle prosessin tuotoksista
  - Erilaisia oppaita
- RUP sisältää 9 työvaihetta
  - Projektin hallinta
  - Bisnes mallintaminen
  - Vaatimukset
  - Analyysi ja suunnittelu
  - Toteutus
  - Testaus
  - Kokoonpanon ja Muutosten hallinta
  - Ympäristö
  - Käyttöönotto
- Viisi tuotossettiä
  - Hallintosetti (projektin suunnittelu ja toteutus)
  - Vaatimussetti
  - Suunnittelusetti
  - Toteutussetti
  - Käyttöönottosetti
- Toimijoita noin kaksi kertaa enemmän kuin UP, mm.
  - Työkaluvastaava
  - Prosessi insinööri
  - Tietokanta suunnittelija
  - Tekninen kirjoittaja

http://www-3.ibm.com/software/awdtools/rup/

The Rational Unified Process

RUP 8 pääperiaatetta

- Hyökkää suurimpien riskien kimppuun aikaisin ja jatkuvasti –tai ne hyökkäävät sinun kimppuun
- Varmistu, että tuotat lisäarvoa asiakkaillesi
- Keskity ajettavaan ohjelmistoon
- Ota muutokset huomioon aikaisessa vaiheessa
- Sovi perusarkkitehtuuri aikaisessa vaiheessa
- Rakenna järjestelmäsi komponenteista
- Työskennelkää yhtenä tiiminä
- Tee laadusta elämäntapa

Extreme Programming (XP)

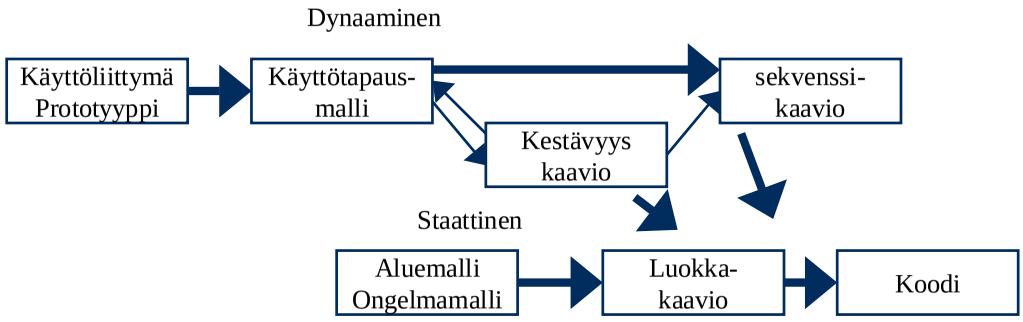
- XP = kevyt metodologia pienille tai keskisuurille tiimeille muuttuvien vaatimusten kanssa
- Neljä arvoa
  - Kommunikointi
  - Yksinkertaisuus
  - Palaute
  - Rohkeus
- Perusperiaatteet
  - Nopea palaute (henkilökohtainen, testaus)
  - Oleta yksinkertaiset ratkaisut
  - Inkrementaalinen muutos
  - Hyväksyä muutos
  - Laatutyö
- Toissijaisia periaatteita
  - Pienet alkuperäiset sijoitukset
  - Konkreettiset kokeet
  - Ota vastuuta
- Kehitystavat
  - Koodaus
    - Refaktorointi
    - Pariohjelmointi
    - Jatkuva integrointi
  - Testaus
    - Yksikkötestit kirjoitetaan ennen kuin koodataan
    - Kaikki yksikkötestit läpäisty ennen järjestelmään liittämistä
    - Toimintotestaus tärkeää, osan testeistä kirjoittaa asiakkaat
  - Kuunteleminen
    - Keskusteluja tämän hetken ongelmista ja vaatimuksista (asiakas)
  - Suunnittelu
    - Yksinkertainen
    - Kerran ja vain kerran
- Suunnittelukilpailu (Planning Game)
  - Tutkiminen
  - Sitoutuminen
  - Ohjaaminen

XP ja RUP

- Yhteiset asiat
  - Kommunikointi
  - Jatkuva integrointi
  - Yksinkertaisuus
- Onko XP räätälöity versio RUP?
- Tärkeimmät erot
  - RUP ylhäältä alas näkökulma järjestelmään ja XP alhaalta ylös
  - XP ei usko dokumentointiin tai muodolliseen mallintamiseen
  - XP sopii pienille ja keskisuurille tiimeille (10-12)

ICONIX process

- RUP ja XP väliltä
  - Kehittäjä Dough Rosenberg
  - Käyttötapausohjattu, mutta ei niin paljon muuta sälää kuin RUP:ssa
  - Melko pieni ja tiivis, mutta sisällyttää analyysin ja suunnittelun
  - Käytetään UML:ää
  - Vaatimusten jäljitettävyys tärkeää



http://www.iconixsw.com/

"imagine"

imagine there's no requirements. It's easy if you try
just a bunch of coders, reachin for the sky
imagine all the people, coding for today

imagine there's no schedules. It isn't hard to do
no silly project deadlines, no one supervising you
imagine all the people, coding hand in hand

you may say I'm an extremer but I'm not the only one
i hope someday you'll join us and make coding lots more fun.

imagine oral documentation. I wonder if you can
no need for UML diagrams. Just words passed, man to man imagine just refactoring, playing in the sand

you may say I'm an extremer, but I'm not the only one
i hope someday you'll join us and make coding lots more fun.

"uml won't write my code"

(sing to the tune of "can't buy me love")

won't write my code
uml won't write my code
won't write my code
no, no, no, no
say you want me to use UML
i say it's a waste of time
i'd rather go by the code smell
Cause the code is the design
i don't care for UML diagrams
Cause UML won't write my code
won't write my code
UML won't write my code
won't write my code
no, no, no, no
you say you want me to document
well i'll tell you hell no
its a waste of time to document
i'm not afraid to tell you so
why the hell would i want to document
when i could be writing code
just writing code
cause its really all i know
just writing code
go, go, go, go

Lyrics by Doug and Rob Rosenberg, ICONIX
http://www.iconixsw.com/songsoftheextremos.html