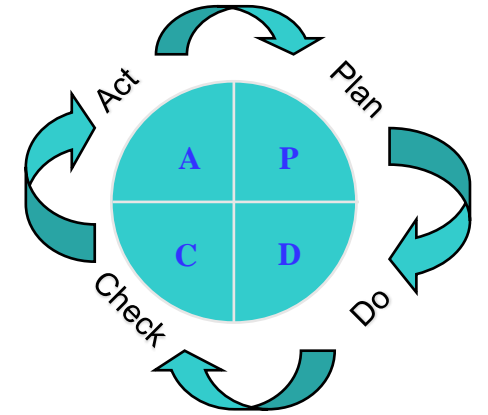


Prosessin parantaminen

Laatuliike

- Laatuliike alkoi **Japanissa 1940-luvun lopulla**
- Vuonna **1982 Deming** esitteli **Plan-Do-Check-Act (PDCA)** -mallin, jota käytetään laadun parantamisen johtamisessa (Deming 1982).
 - Sovellettu myöhemmin ohjelmistokehitykseen.
- **Laatuliike** tuli mukaan **ohjelmistokehitykseen 1980-luvun puolessa välissä**
 - SPI (Software Process Improvement) mallit ja lähestymistavat
- **Ohjelmistoprosessien arviointi- ja parantamisliike** alkoi **USA:ssa 1980-luvulla**, kun maan puolustusministeriö (Department of Defence, **DoD**) huolestui sotilasjärjestelmiensä ohjelmistongelmista.
 - Tämän jälkeen on otettu käyttöön useita erilaisia prosessin **kypsyysmalleja**, joissa on samanlaiset periaatteet



PDCA sykli

- PDCA (Plan-Do-Check-Act) periaate sisältyy useimpiin jatkuvan parantamisen lähestymistapoihin.
- Yksi esimerkki PDCA:stä on The Toyota Production System, joka on kuuluisa kaizen esimerkki

1. Plan (Suunnitellaan)

- Tavoitteiden asettaminen ja muutoksen suunnittelu

2. Do (Tehdään)

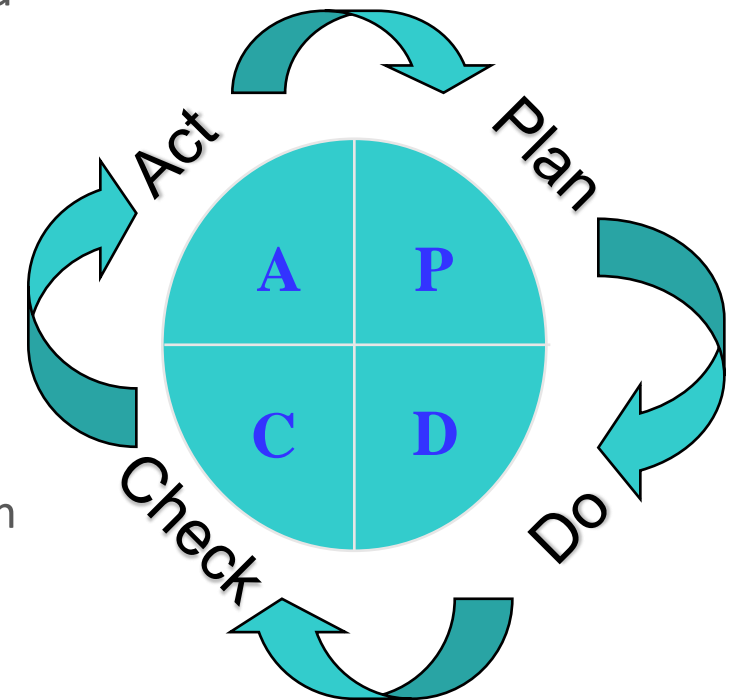
- Toteutetaan muutokset prosesseissa

3. Check (Tarkistetaan)

- Kerätään tietoa prosesseista ja tuotoksista. Sen perusteella arvioidaan tehtyjä prosessimuutoksia ja niiden vaikutuksia.

4. Act (Korjataan ja toimitaan)

- Jos tulokset olivat hyviä niin otetaan muutokset laajempaan käyttöön ja arvioidaan jatkuvasti tuloksia. Jos muutos ei ollut toimiva niin aloitetaan sykli uudestaan.



Deming, W. Edwards (1986)

Prosessien parantamiseen panostaminen (% kokonaisbudjetista)

- 0 % → Vähentäminen
- 2 % → Nykyisen tilanteen ylläpito



- 4 % → Hidas parantuminen
- 10% → Nopea parantuminen

Prosessin parantaminen

- Monet ohjelmistoyritykset ovat siirtyneet ohjelmistoprosessien parantamiseen (software process improvement (**SPI**))
 - parantaakseen ohjelmistojensa **laatua**,
 - vähentääkseen **kustannuksia** tai
 - nopeuttaakseen **kehittämisprosessejaan**.
- Prosessien parantaminen tarkoittaa
 - **nykyisten prosessien ymmärtämistä** ja
 - näiden prosessien **muuttamista laadun parantamiseksi** ja **kustannusten ja kehittämiseen käytettävän ajan vähentämiseksi**.

Lähestymistavat prosessien parantamiseen

- **Prosessin kypsyiden lähestymistavassa** keskitytään prosessin parantamiseen ja projektin johtamiseen ja hyvien ohjelmistotuotannon käytäntöjen käyttöönottoon.
 - Prosessin **kypsyystaso** heijastaa sitä missä laajuudessa hyviä teknisiä tai johtamiseen liittyviä **käytäntöjä** on **otettu käyttöön** organisaation ohjelmistokehitysprosesseissa.
- **Ketterässä lähestymistavassa** keskitytään **iteratiiviseen kehitykseen** ja **yleiskustannusten vähentämiseen** ohjelmistoprosessissa.
 - Ketterien menetelmien pääominaisuudet **nopea toimitus ja toiminnallisuus** ja reagointi muuttuviin asiakasvaatimuksiin.
 - **Jatkuva** prosessien parantaminen

Prosessin parantamistoiminnot

- *Prosessin mittaus*

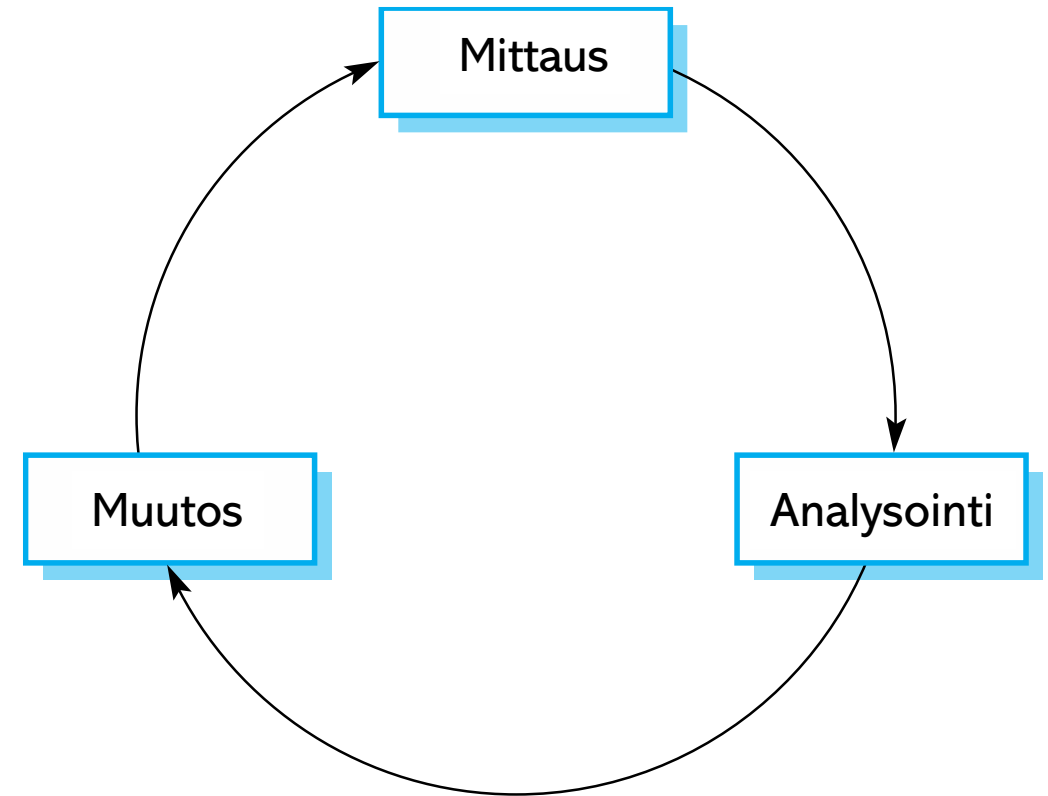
- Mitataan yhtä tai useampaa ohjelmistoproessin tai -tuotteen tekijää. Nämä mittaukset muodostavat perustason, joka auttaa näkemään, ovatko prosessiparannukset olleet tehokkaita.

- *Prosessianalyysi*

- Nykyistä prosessia arvioidaan ja tunnistetaan prosessin heikkoudet ja vahvuudet. Voidaan kehittää prosesseja kuvaavia prosessimalleja.

- *Prosessin muutos*

- Ehdotetaan prosessimuutoksia joidenkin havaittujen prosessin heikkouksien korjaamiseksi. Nämä otetaan käyttöön ja sykli jatkuu jotta voidaan kerätä tietoja muutosten tehokkuudesta.



Prosessin mittaus

- Aina kun mahdollista, tulisi kerätä **määrällistä** tietoa prosessista
 - Tämä on kuitenkin erittäin vaikeaa, jos organisaatiolla **ei ole** selkeästi **määriteltyjä** standardeja **prosesseille**, koska silloin ei tiedetä mitä mitata.
 - **Prosessi** täytyy ehkä ensin **määritellä** ennen kuin mittaukset ovat mahdollisia.
- **Prosessin mittauksia** tulisi käyttää **prosessin parannusten arviointiin**
 - **Organisaation tavoitteiden** tulisi toimia **parantamisen ohjauksena**.

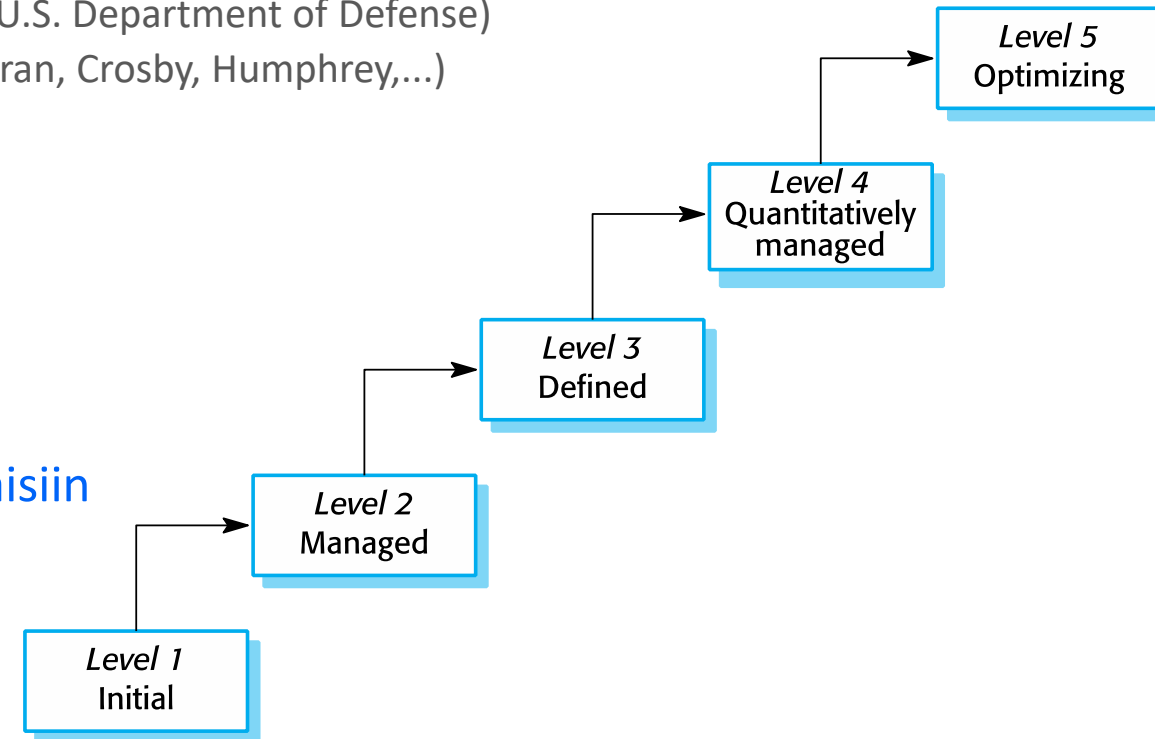
Prosessin metriikat - esimerkkejä

- **Aika** prosessitoimintojen valmiiksi saamiseen
 - Esim. kalenteriaika tai panostus tietyn toiminnon tai prosessin valmiiksi saamiseen.
- **Resurssit**, joita prosessia tai toimintoa varten tarvitaan
 - Esim. kokonaispanostus henkilöpäivinä.
- Tiettyjen tapahtumien **esiintymien määrä**
 - Esim. löydettyjen virheiden määrä.

SEI's CMM

- Kypsyystasomalli (Capability Maturity Model CMM)
 - SEI (Software Engineering Institute, Carnegie Mellon University)
 - Arviointimalli joka kehitettiin alun perin **toimittajien** kyvykkyyksien arviointiin (1987)
 - Kehittämisen rahoitti Yhdysvaltain Puolustusministeriö **DoD** (U.S. Department of Defense)
 - Perustuu pitkäaikaisiin laatuajattelun perinteisiin (Deming, Juran, Crosby, Humphrey,...)
 - Hyvin laajalti käytetty **sisäisten prosessien parantamisessa**
- Korvattu **CMMI:llä** (Capability Maturity Model Integration)
- **CMM/CMMI on suosituin** arviointimenetelmä
 - de Facto käytetyin standardi erityisesti Yhdysvalloissa
 - Erittäin suosittu monikansallisissa suurissa yrityksissä
- On toiminut kehittämisen inspiraationa **useisiin johdannaisiin**

CMM ei ole sama kuin CMMI !!!

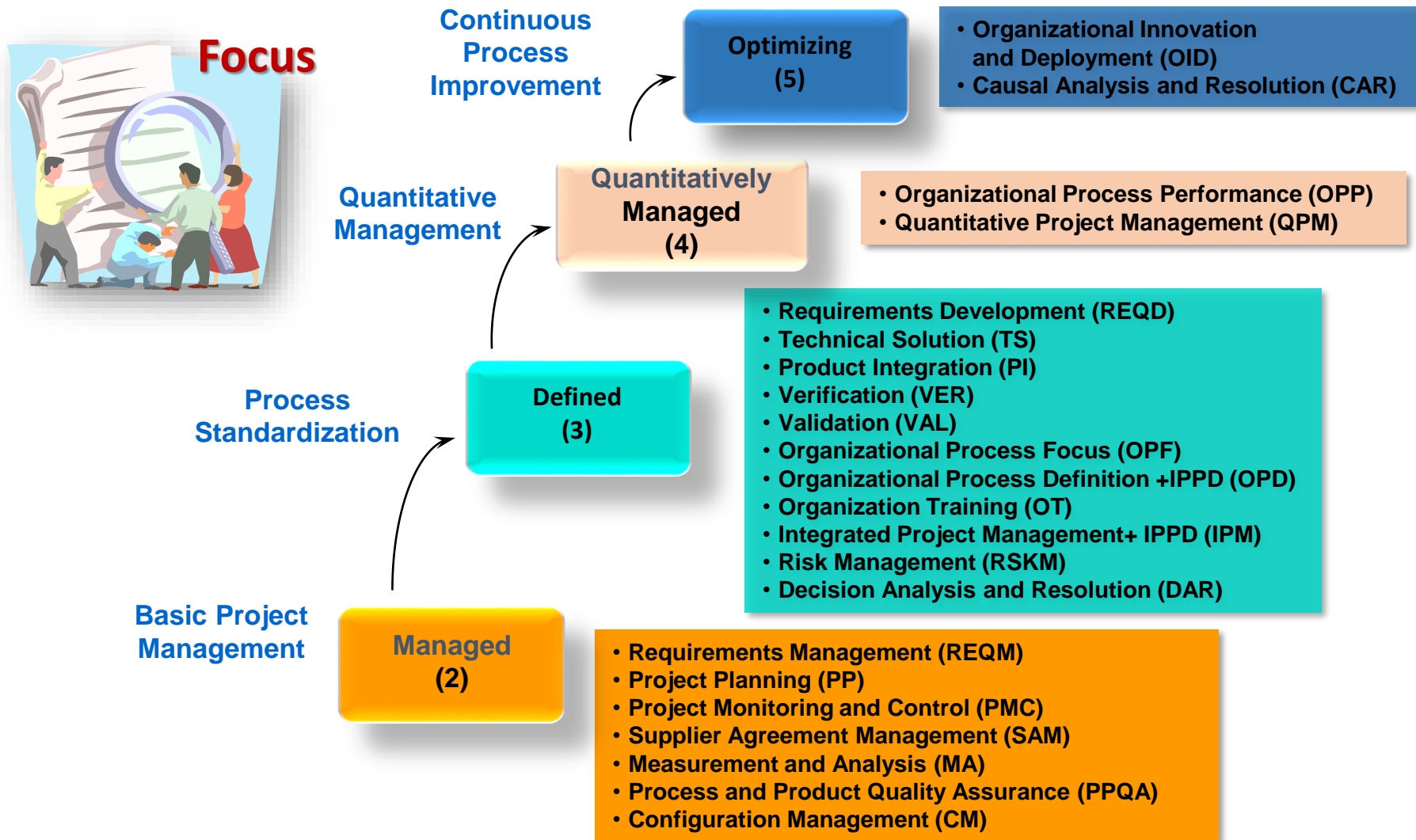


Capability Maturity Model Integration

CMMI kypsyystasot








CMMI tasot ja prosessialueet



Avainkohdat 1/3

- 🔑 Ohjelmistoprosessit ovat ohjelmistojärjestelmän tuottamiseen liittyviä **toimintoja**.
- 🔑 Ohjelmistoprosessimallit ovat näiden prosessien abstrakteja kuvauksia.
- 🔑 Yleiset prosessimallit kuvaavat ohjelmistoprosessien rakennetta.
 - 🔑 Esimerkkejä näistä yleistä malleista ovat ‘**vesiputous**’ malli, **inkrementaalinen** kehittäminen, ja **uudelleenkäyttöön** painottunut kehittäminen.

Avainkohdat 2/3

-  **Vaatimustenhallinta**prosesissa määritellään ohjelmiston vaatimukset (spesifikaatio)
-  **Suunnittelun ja toteutuksen** prosessit liittyvät vaatimusmäärittelyn muuttamiseen toimivaksi ja suoritettavaksi ohjelmistojärjestelmäksi. 
-  **Ohjelmiston validointi** on prosessi, jossa tarkistetaan että järjestelmä on määritelmänsä mukainen ja että se vastaa järjestelmän käyttäjien todellisia tarpeita.
-  **Ohjelmisto kehittyy** kun olemassa olevaa järjestelmää muutetaan vastaamaan uusia vaatimuksia. Ohjelmiston on kehityttävä pysyäkseen hyödyllisenä.

Avainkohdat 3/3

- 🔑 Muutoksissa pärjäämiseksi prosessien tulisi sisältää toimintoja kuten prototypointi ja vaiheittainen toimittaminen.
- 🔑 Prosessit voidaan rakentaa iteratiivista kehittämistä ja toimittamista varten, jotta muutokset voitaisiin tehdä häiritsemättä koko järjestelmää.
- 🔑 Tärkeimmät lähestymistavat prosessien parantamiseen ovat ketterät lähestymistavat, joiden tarkoituksena on vähentää prosessin kokonaiskustannuksia ja kypsyyteen perustuvat lähestymistavat, joissa pyritään prosessien parempaan hallintaan ja ohjelmistotuotannon hyvien käytäntöjen käyttöön.
- 🔑 CMMI prosessin kypsyysmallissa tunnistetaan kypsyystasot jotka vastaavat hyvien ohjelmistotuotannon käytäntöjen käyttöä.