Järjestelmän mallintaminen

Järjestelmän mallinnus

- Järjestelmän mallinnus on prosessi, jossa luodaan järjestelmän abstraktit mallit. Jokainen malli antaa erilaisen näkökulman järjestelmästä.
- Mallinnus tehdään yleensä jollain graafisella notaatiolla, joka on nykypäivänä käytännössä aina Unified Modeling Language (UML)-notaatioperheen notaatio.
- Järjestelmän mallinnus auttaa analyysin tekijöitä ymmärtämään järjestelmän toiminnallisuus, ja malleja käytetään myös asiakkaiden kanssa vuoropuhelussa apuna.

Olemassa olevan järjestelmän ja tulevan järjestelmän mallit

- Olemassa olevaa järjestelmää voidaan mallintaa.
  - Mallit auttavat selventämään, mitä nykyjärjestelmä tekee.
  - Niiden avulla voidaan pohtia sen heikkouksia ja vahvuuksia.
  - Niiden avulla saadaan uudelle tulevalle järjestelmälle vaatimuksia.
- Tulevaa järjestelmää mallinnetaan vaatimusmäärittelyprosessissa.
  - Ne auttavat selittämään ehdotusta tulevan järjestelmän vaatimuksista.
  - Ohjelmiston kehittäjät käyttävät malleja dokumentoidakseen järjestelmän toteutusta varten.

- Mallipohjaisessa ohjelmistotuotannossa (model-driven engineering) rakennetaan järjestelmää toteuttamalla malleja.

Järjestelmän näkökulmat

- Ulkoinen näkökulma - mallinnetaan järjestelmän ympäristö.
- Vuorovaikutusnäkökulma - mallinnetaan järjestelmän ja sen ympäristön vuorovaikutus. Interaktiomalleilla kuvataan myös järjestelmän komponenttien keskinäinen vuorovaikutus.
- Rakenteellinen näkökulma - mallinnetaan järjestelmän organisaation tai käsiteltävän datan rakenne.
- Käyttäytymisnäkökulma - mallinnetaan, miten järjestelmä käyttäytyy.

Esimerkkejä UML kaaviotyypeistä
(Unified Modeling Language)

- Aktiviteettikaaviot (Activity diagrams): mistä tehtävistä (task) aktiviteetit koostuvat.
- Käyttötapauskaaviot (Use case diagrams): vuorovaikutus järjestelmän ja sen käyttäjien välillä.
- Sekvenssikaaviot (Sequence diagrams): vuorovaikutus käyttäjien (actor), järjestelmän ja järjestelmäkomponenttien välillä.
- Luokkakaaviot (Class diagrams), rakenteellinen kaaviotyyppi, luokkien olioiden rakenne ja assosiaatiot niiden välillä.
- Tila(kone)kaaviot (State (machine) diagrams), miten järjestelmän komponentit reagoivat tapahtumiin.

Graafisten mallien käyttö

- Keskustelu nykyisestä tai ehdotetusta järjestelmästä
  - Epätäydellinen ja epätarkka malli on mahdollinen, koska tarkoitus on vain tukea keskustelua.
- Dokumentoidaan olemassa oleva järjestelmä
  - Mallin tulisi olla paikkaansa pitävä, mutta sen ei tarvitse olla täydellinen.
- Dokumentoidaan yksityiskohtainen kuvaus tulevasta järjestelmästä
  - Mallin pitää olla sekä paikkaansa pitävä että täydellinen.

Esimerkki: Mallipohjainen suunnittelu

(Model-driven engineering)

- Mallipohjainen suunnittelu (MDE) on ohjelmistokehityksen lähestymistapa, jossa mallit eivätkä ohjelmat ovat pääasiallinen kehitysprosessin tuloste.
- Mallipohjainen arkkitehtuuri (MDA) oli MDE:n edeltäjä.
  - MDA on mallifokusoitunut lähestymistapa ohjelmiston suunnittelulle ja toteutukselle. Se käyttää UML:n malleja kuvaamaan järjestelmää.
- Ohjelmat generoidaan automaattisesti malleista.
- MDE:n kannattajat katsovat, että ne nostavat ohjelmistokehityksen abstraktiotasoa, joten kehittäjien ei tarvitse keskittyä ohjelmointikielten yksityiskohtiin tai toteutusalustan erityispiirteisiin.

MDE:n käyttökelpoisuus

- Edut
  - Sallii järjestelmän ymmärtämisen korkeatasoisena abstraktiona.
  - Koodin automaattinen generointi tekee halvemmaksi järjestelmän muokkaamisen uuteen ympäristöön.
- Haitat
  - Abstraktiomalli ei välttämättä ole sopiva toteuttamiseen.
  - Säästö koodin generoinnissa voi mennä uusien alustojen kääntäjien kehittämiseen
- Yleisesti vastaavanlaiset menettelytavat tulevat, menevät ja kehittyvät hyvin nopeasti ohjelmistotuotannossa, varsinkin eri sovellutusalueilla. Toisin sanoen, perusteiden hallinta on tärkeämpää, menetelmän voi valita sovellusalueen, tiimin ja muiden tekijöiden ehdoilla.