Object Modelling Technique (OMT) Áttekintés

Ficsor Lajos
Miskolci Egyetem
Általános Informatikai Tanszék

Általános INFORMATIKAI Tanszék

Az OMT eredete

Rumbaugh et al.: Object-Oriented Modelling and Design, 1991

Előnyei:

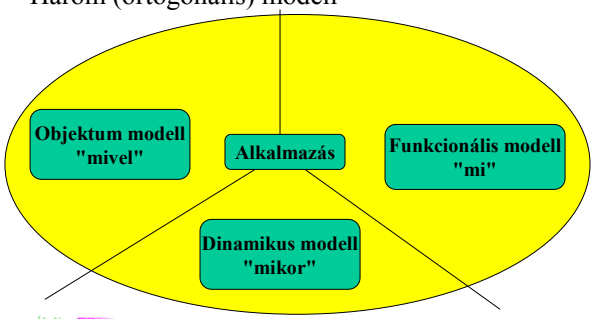
- világos fogalmak
- jó átmenet a strukturált módszerekből
- konkrét és gyakorlatias tanácsok a fejlesztéshez
- jól követhető lépéssorozat

Hátrányai:

- *túl* egyszerű
- követelmény-analízishez csak a nyelvtani elemzés (Abbot módszere)

Az OMT szemlélete

Három (ortogonális) modell



Az objektum modell

"MIVEL?"

A rendszer általános struktúrája

Funkcionális dekompozíció helyett ***strukturális***

Jelölésrendszer:

- osztály diagramm
- komponens diagram

Eredetileg saját jelölésrendszer, a továbbiakban ahol lehet, **UML**

A dinamikus modell

"MIKOR?"

A rendszer építőelemeinek **viselkedése** (időbeli változása)

Minden objektumra:

- hogyan változtatja állapotát
- hogyan hat a környezetére

Eszközök:

- állapot diagram (UML)
- sorrend diagram (UML)
- eseményfolyam-diagram (saját)

A funkcionális modell

"MI?"

A rendszeren belüli számítások, feldolgozások.

Nem tartalmaz időt.

Megadhatjuk:

- az objektum modell ***műveleteit***
- a dinamikus modell ***akcióit***
- az objektum modell ***korlátozó feltételeit***

Eszköze:

- DFD (eredeti)
- aktivitás és együttműködési diagramm (UML)

A fejlesztés fázisai 1.

1. Analízis

- a rendszer *lényeges* elemeinek leírása
- a feladat szöveges leírásának elemzésével

2. Rendszertervezés (system design)

- alrendszerekre bontás
- a megvalósítás stratégiai döntései
  - erőforrások elosztása
  - optimalizálandó tulajdonságok
  - alrendszerek közötti kommunikáció
  - végrehajtás/vezérlés módja

A fejlesztés fázisai 2.

3. Objektum tervezés (object design)

- a három modell összhangba hozása
- adatszerkezetek és algoritmusok

Mindhárom fázisban a három modellt használjuk, de különböző absztrakciós szinteken

4. Implementáció

- a modell lefordítása egy programozási nyelvre

Analízis (elemzési) fázis

- A feladat szöveges megfogalmazásából indul ki
- Az alkalmazás szakterületének fogalmaival dolgozik
- Kapcsolat a felhasználóval
- Bonyolult rendszernél több iteráció
- Elkészített modellek:
  - objektum modell
  - dinamikus modell
  - funkcionális modell

Analízis fázis / Objektum modell 1.

- **Osztályok azonosítása** (a szövegben általában *főnevek* utalnak lehetséges osztályokra)
- **Megfelelő osztályok kiválasztása.** Törlendők:
  - redundáns osztályok
  - felesleges osztályok
  - pontatlan osztályok
  - attribútumok
  - műveletek
  - szerepkörök
  - implementációs elemek
- **Osztályok leírása – repository!**

Analízis fázis / Objektum modell 2.

- **Asszociációk azonosítása.** Igék vagy igei kifejezések utalhatnak rá. Pl.:
  - fizikai elhelyezkedés (része, alkotja ...)
  - tárgyas igékkel kapcsolatos cselekvések (vezeti, leveszi ...)
  - kommunikáció (üzeni, átadja ...)
  - birtoklás (van neki, hozzátartozik ...)
  - előírt feltételeknek való megfelelés (tag, alkalmazásban áll ...)

Analízis fázis / Objektum modell 3.

- **Megfelelő asszociációk kiválasztása.** Törlendők:
  - törölt osztályok közötti asszociációk
  - lényegtelen asszociációk
  - implementációs asszociációk
  - akciók
  - származtatott asszociációk
- **Ternáris (illetve többszörös) asszociációk átalakítása**

Analízis fázis / Objektum modell 4.

- **Asszociációk szemantikájának ellenőrzése**
  - megfelelő elnevezés
  - szerepkör nevek megadása
    - mindig szükséges ha egy osztály önmagával áll asszociációban
    - elhagyható, ha a kapcsolat nevéből egyértelműen következik
  - számosság meghatározása
  - minősített asszociációk kiválasztása
  - hiányzó asszociációk feltárása

Analízis fázis / Objektum modell 5.

- **Attributumok azonosítása** (melléknevek, birtokos szerkezetek)
- **Megfelelő attributumok kiválasztása**
  - származtatott attributum elhagyása vagy megjelölése
  - ha egy adat önálló létezéssel rendelkezik, az objektum
  - az objektum azonosító implementációs elem
  - a minősítő, ha az értéke azonosítás jellegű, elhagyandó, illetve asszociációhoz rendelhető
  - név jellegű attribútum legyen minősítő

Analízis fázis / Objektum modell 6.

  - az attribútum osztályhoz vagy asszociációhoz kapcsolódik?
  - implementációs részleteket jelentő attribútum (belső érték) elhagyandó
  - egy, a többi attribútumhoz lazán kapcsolódó attribútum: az osztályt több osztályra célszerű bontani
- **Általánosítás** (pl. jelzős kapcsolat azonos tárgyakhoz) - a több osztályban előforduló attribútumok kiemelése, származtatási hierarchia kialakítása

Analízis fázis / Objektum modell 7.

- **Elérési utak tesztelése**
  - a működéshez szükséges asszociációk rendelkezésre állnak-e
  - többszörös multiplicitás esetén a kiválasztáshoz van-e minősítő vagy név
- **Modulok meghatározása**

Az egymáshoz szorosan kapcsolódó elemek csoportosítása. (komponens diagram)

- **Iterációs finomítás, ha a rendszer összetettsége indokolja**

Analízis fázis / Dinamikus modell 1.

A rendszer (és objektumai) időbeli viselkedése

- **Forgatókönyvek készítése:** a működést kísérő események
  - felhasználó - rendszer közötti párbeszéd
  - belső objektumok közötti kommunikáció
  - tipikus, speciális, különleges esetek
- **Felhasználói felület** (vázlatosan)
  - vezérlés
  - információcsere

Analízis fázis / Dinamikus modell 2.

- **szekvencia diagram rajzolása**
  - események és objektumok azonosítása a forgatókönyvekben
  - első lépésben autonóm, konkurrens objektumokat tételezünk fel
  - az események közötti ok-okozati összefüggések feltárása
    - objektumok alá-fölérendeltsége vezérlési szempontból
    - objektumok aktív időszakainak meghatározása (téglalapok!)

Analízis fázis / Dinamikus modell 3.

- **állapotdiagram rajzolása** objektumonként
  - minden szekvencia diagram a végrehajtás egy adott sorrendjét adja meg
  - szükség szerint strukturáljuk a diagrammokat
- **eseményfolyam diagram készítése**
  - Osztályok nyilakkal (események) összekötve.
  - Egy osztály valamennyi objektuma által küldött és fogadott eseményeket az adott osztályhoz rendeljük
  - Az osztályok interface-ét pontosítja
  - Az események sorrendje most lényegtelen

Analízis fázis / Funkcionális modell 1.

A rendszer értékeinek számítási módja. DFD rajzolása

- **Be- és kimeneti értékek azonosítása**
  - rendszeren belüli vagy külső események paraméterei
- **Adatfolyam-diagram megrajzolása**
  - a bemeneti értékekből hogyan származtathatók a kimeneti értékek
  - több szint!

Analízis fázis / Funkcionális modell 2.

- **A transzformációk specifikálása**
  - bármilyen (egyértelműséget biztosító) módszer használható
  - implementációs elemeket nem tartalmazhat!
- **Objektumok közötti korlátozások** meghatározása
  - nem ki-bemenet jellegű funkcionális függőségek
  - a lehetséges értékek halmazának szűkítése
- **Optimalizálandó értékek meghatározása**
  - a rendszerre előírt nem funkcionális követelmények teljesítéséhez szükséges elemekre

Analízis fázis / Műveletek azonosítása 1.

Meghatározzuk az egyes osztályokhoz telepítendő műveleteket. Ezek lehetnek:

- **események** a dinamikus modellből:
  - a generált esemény lehet a fogadó objektum művelete
  - kivételként kezelt esemény - a generáló objektum dolgozza fel
- állapotdiagram **tevékenység**ei és **akció**i
  - az állapotdiagram egy objektumhoz tartozik, így ezen műveletek helye adott

Analízis fázis / Műveletek azonosítása 2.

- az adatfolyam diagram **processz**ei - itt a legnehezebb a művelet osztályhoz telepítése
  - tartozhatnak egy osztályhoz, de lehetnek több objektumon végrehajtandó műveletek is
- **szabvány műveletek**: az objektumhoz természeténél fogva hozzátartozó műveletek
  - az objektum modellből származtathatók
  - attribútumok és kapcsolatok elérése
  - akkor is definiáljuk, ha a jelenlegi modell nem használja fel

Rendszertervezés 1.

A megvalósítás magas szintű döntései

- **Alrendszerekre bontás -** az alapvető architektúra. Szokásos megoldások:
  - réteges (zárt és nyílt architektúra)
    - a legfelső réteg maga a szakterületi modell (az analízis fázis eredménye)
    - szinte mindig van **megjelenítési** és **adattárolási** réteg
  - partíciók - egymáshoz lazán kapcsolódó részek
  - osztott (distributed) architektúra - az egyes részek különböző csomópontokon

Rendszertervezés 2.

- **A rendszer topológiájának meghatározása**
  - információáramlás az alrendszerek között
- **Lényegi konkurrencia meghatározása**
  - amely nem egyszerűsíthető szekvenciális végrehajtássá
- **Alrendszerek processzorokhoz és taszkokhoz rendelése**
  - erőforrás igények
  - kapcsolódás eszköze

Rendszertervezés 3.

- **Adattárolás eszközének kiválasztása**
  - kell-e adatbáziskezelő, ha igen, milyen
- **Globális erőforrások elosztásának szabályozása**
  - memória, processzor, hálózat stb.
- **Vezérlés elvének kiválasztása**
  - eljárásorientált (függvényhívások sorozata)
  - eseményorientált (pl.grafikus felhasználói felületek vagy egyéb külső vezérlés)
  - valódi párhuzamos

Rendszertervezés 4.

- **Rendszer határfeltételeinek meghatározása.** Tevékenységek:
  - a rendszer telepítésekor (első indulás előtt)
  - a rendszer üzemszerű indulásakor (inicializálás)
  - a rendszer üzemszerű leállásakor (terminálás)
  - hibás befejeződéskor
- **Felhasználói felület / külső interface tervezése**

Objektumtervezés - feladata 1.

Az osztályok implementációtól független valamennyi részletének tisztázása.

Az implementációs eszköz ismerete szükséges.

Fontosabb lépései:

- A három modell egyesítése
- Algoritmus tervezés
- Asszociációk tervezése
- Láthatóság biztosítása
- Nem OO környezethez való illesztés

Objektumtervezés - feladata 2.

- Ütemezési szerkezet kialakítása
- Optimalizálás
- Deklarációs sorrend meghatározása
  - implementációs és deklarációs függőségek minimalizálása
- Modultervezés
  - a modulokon belüli osztályok logikailag összefüggjenek
  - modulok közötti kommunikáció minimális legyen

Objektumtervezés - modellek egyesítése 1.

- **Az objektum modell elemzése**
  - közvetlenül formalizálható információk
- **A funkcionális modell beépítése** - általában ez a legnagyobb probléma
  - adatjellegű elemek - leképezés objektumokra vagy azok attribútumaira. Új osztályok jelentkezhetnek!
  - folyamatok leképezése
    - folyamat és metódus absztrakciós szintje eltérő lehet
    - egy funkció melyik osztály metódusára képezhető le?

Objektumtervezés - modellek egyesítése 2.

- tárolók és külső entitások leképezése
  - objektumok!
  - a hozzá tartozó folyamatokat az objektumokhoz kell kapcsolni

- **A dinamikus modell elemzése**
  - üzenetváltások leképezése metódushívásokra
  - visszatérési érték általában objektum
- **Osztályok egyedi vizsgálata**
  - osztály önmagában konzistens megvalósítása. Pl.
    - konstruktor - destruktor feladata, kezelő metódusok stb.

Objektumtervezés - algoritmus tervezés

- üzenet algoritmusok
- implementációs adatstruktúrák
- újabb (technikai) osztályok megjelenése lehetséges
- Szempontok
  - áttekinthetőség és módosíthatóság - objektumon belüli dekompozíció
  - komplexitás
  - osztálykönyvtárak használata

Objektumtervezés - Aszociációk tervezése

egyirányú asszociációk

- 1-1 és 1-n : tartalmazás vagy beágyazott mutatók
- n-m: asszociációs objektum - normalizálás!

kétirányú asszociációk: call-back technika

- a tartalmazó (vagy hívó) objektum átadja paraméterként a saját címét vagy referenciáját a tartalmazottnak (vagy hívottnak)

Objektumtervezés - láthatóság biztosítása

forgatókönyvek és kommunikációs diagrammok alapján

- globális objektum - mérlegelés után!
- argumentum - feltételekhez kötött!
- tartalmazás - gyakori
- elérési függvények
- közvetítő objektum

Objektumtervezés - illesztés

Nem OO környezethez való illesztés

- "Csomagoló" osztályok
- Kész osztálykönyvtárak (pl. MFC)
- Adatbáziskezelő interface-ek

Objektumtervezés - ütemezés 1.

Ütemezés implementációjának módja függ

- a használt fejlesztői környezet lehetőségeitől
- a futtatási környezet tulajdonságaitól

A modell aktív és passzív objektumainak száma

- nincs aktív - "alkalmazás" objektum, ami indít
- egy aktív: az indít
- több aktív: az ütemezést meg kell oldani

Objektumtervezés - ütemezés 2.

Ütemezési megoldások:

- belső, nem preemptív - belső ütemező
- belső, preemptív belső ütemező objektum
  - folytathatóság
  - kritikus szakaszok - kizárás
- időosztásos operációs rendszerrel
  - minden aktív objektum külön taszkban
  - inter-processz kommunikáció
  - szinkronizálás, kölcsönös kizárás

Objektumtervezés - ütemezés 3.

- többprocesszoros rendszer használata
  - objektumok és processzoros összerendelése
  - kommunikáció az interprocessz kommunikációhoz hasonló
- elosztott rendszer használata
  - objektumok és csomópontok összerendelése
  - hálózati kommunikáció!
  - osztott objektum modell szabványok használata

Objektumtervezés - optimalizálás

- **Optimalizálás tervezése**
  - redundáns elemek tárolása az elérés gyorsítására
  - végrehajtás sorrendjének megváltoztatása
  - ideiglenes tárolás (pl. memóriában)
  - hatékonyabb algoritmusok
  - **veszély:** a jól megtervezett struktúra felborítása