Szoftver felépítmények

Zoltán László fóliáinak
felhasználásával

Szoftver felépítmények

- Szoftver felépítmény (architektúra):

  A szoftver rendszer szervezésével kapcsolatos legfontosabb döntések

  – A szoftver elemeinek és kapcsolódó felületeinek (szerkezeti) leírása

  – A viselkedést a szoftverelemek közötti együttműködés fogalmaival írjuk le

  – Szerkezeti és viselkedésbeli elemek együttesen

  – Szoftver felépítmény minták

    - használat
    - műveletkészlet
    - hatékonyság
    - újrafelhasználhatóság
    - érthetőség
    - gazdasági és technológiai megkötések és alkuk
    - esztétikai vonatkozások

Szoftver felépítmények

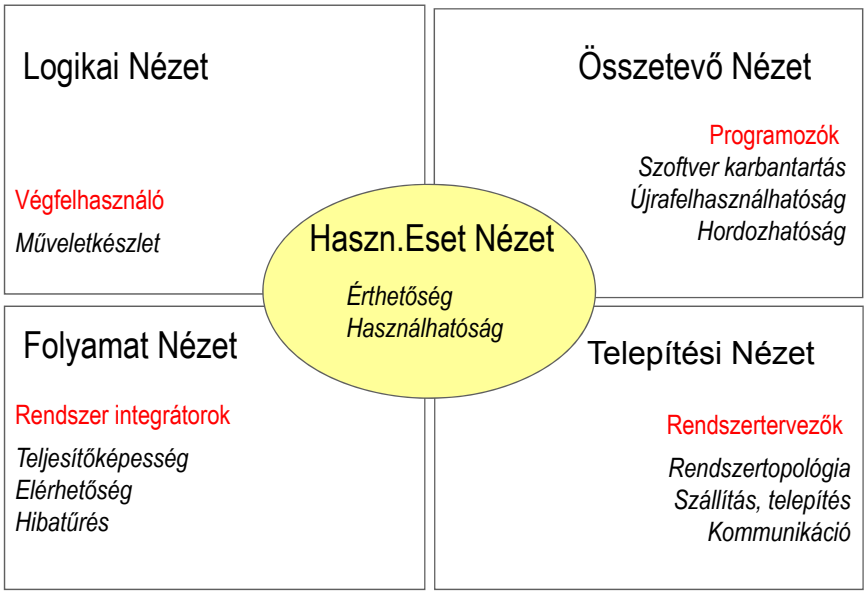
- Egy felépítmény stílusa:
  - Hasonló szerkezetű rendszerek egy családja
  - Összetevők és kapcsolóelemek halmaza
  - Az összekapcsolásra vonatkozó megszorítások halmaza
  - Szemantikus modellek, amelyek megadják hogy hogyan vezethetők le a rendszer általános tulajdonságai a részrendszerekéből
- Egy felépítmény nézete:

  a rendszer egyszerűsített leírása/elvonatkoztatás valamilyen nézőpontból, lényeges vonásokat kiemelve és lényegteleneket pedig elhagyva

Szoftver felépítmények

- Jó felépítmény jellemzői:
  - rugalmas
  - egyszerű
  - jól szétválasztott szempontrendszerek
  - felelősségek kiegyensúlyozott szétosztása
  - gazdasági és technológiai megkötések kiegyensúlyozott kezelése
- Lényeges elemek
  - A legfontosabb “üzleti” osztályok
  - A legfontosabb működési módok
  - Gépek és folyamatok
  - Rétegek és alrendszerek

Szoftverfelépítmények 4+1 nézete



Szoftver felépítmény stílusok

- Dataflow rendszerek.
  - Batch szekvenciális
  - Csövek és szűrők:
- Hívott-visszatérő rendszerek
  - Főprogram és eljárás
  - Objektum orientált rendszerek
  - Hierarchikus rétegek
- Adatközpontú rendszerek
  - Adatbázisok
  - Hipertext rendszerek
  - Blackboardok
- Virtuális gépek
  - Értelmezőprogramok
  - Szabályalapú rendszerek
- Független összetevők
  - Kommunikáló folyamatok
  - Eseményvezérelt rendszerek

Dataflow rendszerek: adatáramláson alapuló rendszerek, a leírás alapja az adatok mozgása és
átalakulása
o Csövek és szűrők: minden komponensnek van ki- és bemenete, a szűrök módosítják
is az átmenő adatokat, a csövek csak kapcsolatot biztosítanak. A szűrők a
komponensek, a csövek pedig a kapcsolóelemek.
o Batch szekvenciális rendszer: csövekből és szűrőkből áll ez is, de a szűrök minden
bemenetüket egyetlen entitásként kezelik

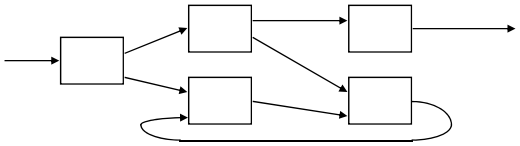
Hívás-és-visszatérés rendszerek:
o Főprogram és eljárások: adott egy főprogram, ami különböző eljárásokat hív meg, és
a visszaadott adatokkal dolgozik tovább.
o OO rendszerek: minden adat egy objektumba van zárva. A komponensek az objektumok, a kapcsolóelemek az eljárások. Minden objektum maga felelős az adataiért és műveleteiért. Adatrejtés is lehetséges. Hátrány, hogy kommunikációhoz
ismerni kell az objektum azonosítóját.
o Hierarchikus rétegek: egy rendszer több hierarchikus szintre (rétegre) van osztva,
minden szint az alatta lévő réteget szolgálja ki. Egy réteg egy elvonatkoztatási vagy
fizikai réteget jelent. Egy szintre a hasonló elvonatkoztatási szinten lévő, de egymástól független elemek kerülnek. Ezzel a problémák jól feloszthatóak, az egyes
részek újrafelhasználhatók. Hátrány a hatékonyság (egy szint mindig csak a felette
lévőt hívhatja). Ilyen a két- és három rétegű szoftver felépítmény.

Független komponensek:
o Eseményvezérelt rendszerek: nem közvetlen eljáráshívás van, az egyes komponensek
eseményeket váltanak ki, más komponensek pedig figyelnek bizonyos típusú
eseményeket. Egy esemény egy bizonyos típusú eljárás meghívását váltja ki.
 Virtuális gépek:
o Értelmezőprogramok: egy virtuális gépet megvalósító szoftver szimulátor. Általában
négy része van: az értelmezett program, a program állapota, az értelmező állapota és
az értelmező maga. Egy programozás nyelv felfogható, mint egy „virtuális nyelv gép”.

Adatközpontú rendszerek: két jellemző részük van: egy központi adat struktúra, ami az
adatok aktuális állapotát reprezentálja, és független komponensek gyűjteménye, ami az
adattárolón dolgozik.
o Adatbázisok: ha a bemenő adat tranzakció választja ki a futtatandó folyamatot, akkor
hagyományos adatbázis használható
o Blackboard rendszerek: ha a bemenő adatoknál az adattároló állapota határozza meg
a futtatandó folyamatot, akkor blackboard a rendszer. Specializált alrendszerekből áll, amik egy közös adattárral (ez a blackboard) dolgoznak. Az egyes alrendszerek eredményeinek összefésülése adja a teljes rendszer eredményét.

Csövek és szűrők

– Egy program működésének felbontása rész-elemi lépésekre
– Ezek egymással adatkapcsolatban vannak, egyesek eredményei mások adatai
– Egyes részlépések adatai a teljes program adatai, más részlépések eredményei a teljes program eredményei

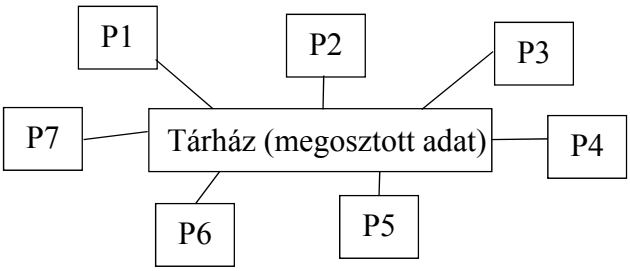


Csövek és szűrők

- Szűrő: adatok átalakítását végző szoftverösszetevő
- Cső: az adatok közlekedését lehetővé tevő kapcsolóelemek
- Megszorítások: az adatok természetére és értékére vonatkozóan
- Előnyök:
  – Könnyű újrafelhasználhatóság
  – Könnyű karbantarthatóság és átépíthetőség
  – Párhuzamos végrehajtásra alkalmas
- Hátrányok
  – „batch szekvenciális gondolkodásmód”
  – Az elemi lépések egyensúlyvesztése
  – Interaktív alkalmazásokra nem használhatók

Blackboard rendszerek

- Specializált alrendszerek, amelyek eredményeinek összefésülése adja a teljes rendszer eredményét
- Az alrendszerek egymástól függetlenül dolgoznak, és közös adatterületen kommunikálnak

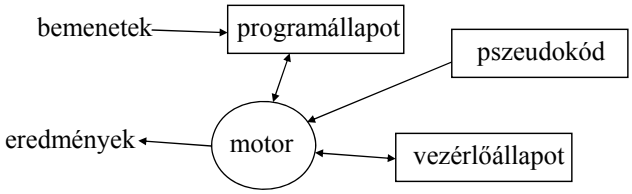


Blackboard rendszerek

- Alrendszerek
  - különválasztott, független műveletek.
  - Minden adatcsere a blackboardon keresztül történik
  - Központi adattárolás
- Előnyök
  - A vezérlés, feldolgozás és adat könnyű szétválaszthatósága
  - Tűri a zajos adatot és a bizonytalan eredményképzést.
- Hátrányok
  - Nehezen tesztelhető.
  - Alacsony hatékonyság
  - Magas fejlesztési költség

Értelmezőprogramok

- Az értelmezőprogram egy virtuális gépet megvalósító szoftver szimulátor
- Részei:
  - Értelmezőmotor
  - Tárterület az interpretált kód számára
  - A motor belső vezérlőállapotát ábrázoló tár
  - Az interpretált gép állapotát ábrázoló tár



Objektumorientálás

- Adatleképezés és műveletei egy objektumba összefoglalva
- Előnyök:
  - Egy objektum felelős a saját adataiért és műveleteiért
  - Az adatok ábrázolása rejtett a külső objektumok előtt
- Hátrányok:
  - Objektumokkal való kommunikációhoz egy azonosító ismeretére van szükség

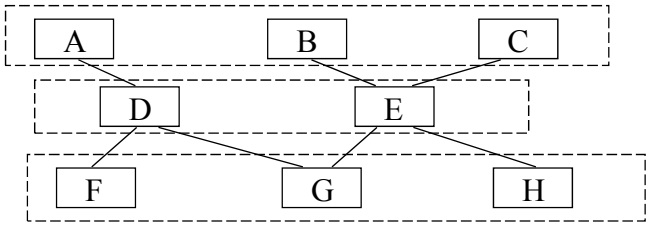
Eseményvezérelt, implicit hívásos

Implicit hívás:
- az eljárások közvetlen meghívása helyett az alrendszerek eseményeket küldenek szét
- Más alrendszerek az eseményhez kezelőeljárásokat rendelhetnek.
- Események fellépésekor a rendszer meghívja az eseménykezelő eljárásokat

- Alrendszerek
  - objektumfelület interface + események
- Hátrányok
  - A vezérlést hangsúlyozzuk a számítás helyett
  - Adatcsere igény
  - Helyességi ellenőrzés/bizonyítás problémás

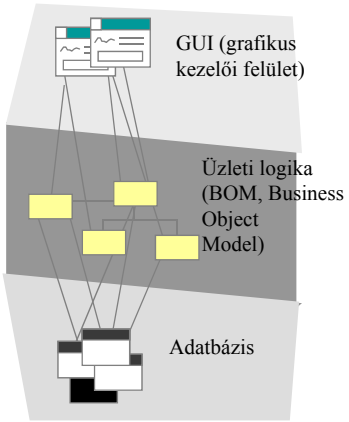
Réteges szoftverfelépítmény

- Rétegek:
  - Elvonatkoztatási szinteket és fizikai rétegeket (számítógépeket+szoftvereket) jelentenek. A hasonló elvonatkoztatási szinten levő műveletek/szolgáltatások azonos környezetbe kerülnek.
  - Vízszintes és függőleges tagoltság



- Rétegek
  - Szoftverösszetevők
    - Minden réteg egymástól független szoftverösszetevőkből áll.
  - Megszorítás
    - A ***j*** réteg szolgáltatásai csak a ***j+1***. rétegből használhatók
  - Előnyei
    - Absztrakciós szinteket támogató tervezési módszerek használhatók.
    - Az egyes összetevők újrafelhasználása könnyű.
    - Könnyű a rendszerek karbantartása és továbbfejlesztése.
    - Skálázhatóság
  - Hátrányai
    - A megoldás nem minden rendszerre keresztülvihető
    - Hatékonyság: nem könnyű a magas és alacsonyszintű összetevők megkülönböztetése
    - Hatékonyság: Egy magasszintű rétegből az alacsonyszintű réteg nem szólítható meg közvetlenül

- Ügyfél-kiszolgáló logikai rétegek

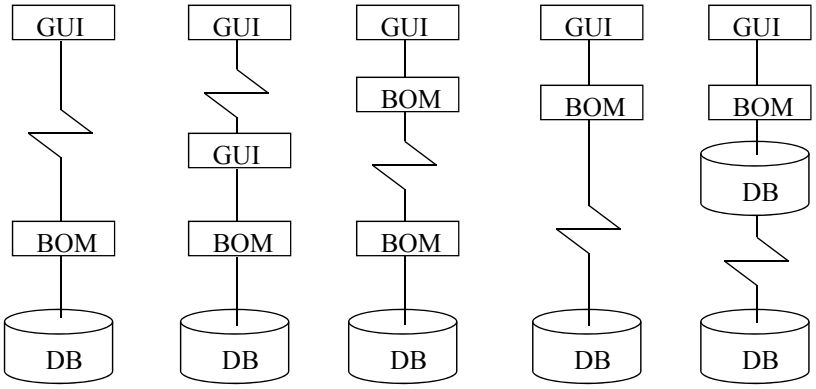


Front-end, előtér

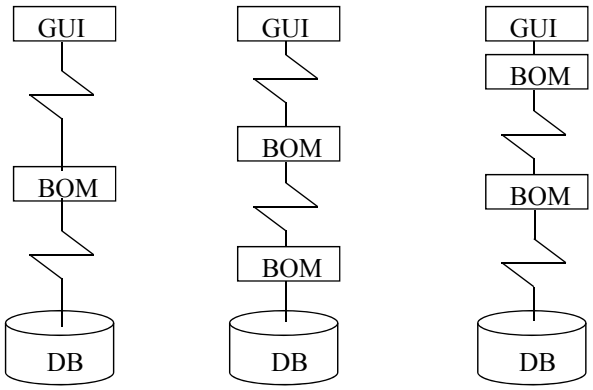
Középréteg

Back-end, háttér

- Kétrétegű ügyfél-kiszolgáló változatok
  1. Üzleti logika és adatbázis ugyanazon a gépen
  2. Központi kezelői felület és távoli terminálok
  3. Üzleti logika részben az ügyfélnél
  4. Üzleti logika teljes egészében az ügyfélnél
  5. …ugyanez helyi adattárolással



- Háromrétegű ügyfél-kiszolgáló változatok
  1. Klasszikus háromrétegű szerkezet
  2. Üzleti logika részben a háttérben
  3. Üzleti logika részben az ügyfélnél



Vékonyabb ügyfél, vastagabb kiszolgáló