Formális módszerek

GM_IN003_1

Bevezetés

Formális módszerek

- Formális módszer ≠ formalizált módszer(tan)
- Formális eljárások alkalmazása a fejlesztésben
  - nincs olyan formális eljárás, ami egy komplex rendszer minden aspektusát leírná
  - struktúrális és *viselkedési* leírások
  - tervezés, terv verifikálás

Formális módszerek (folyt.)

- Formális eljárások
  - közös koncepciók
    - rögzített szintaxis, szemantika
    - matematikai eszközök
    - eltérő jelölés
  - absztrakt matematikai modellek a rendszer állapotáról
  - egyre részletesebb modellek
    - -> implementáció (absztrakció csökken)
    - a specifikáció finomítása

Bevezetés

Formális módszerek (folyt.)

- Szigorú módszerek a rendszer tervezésben és fejlesztésben
- Szimbólikus (matematikai) logikai konstrukciók => formálisak (reprezentációk, eszközök)
- Rendszerrel szembeni bizalom erősítése
  - implementáció
  - követelmények
- Formális modellek létrehozása, formális követelményeknek megfelelőség mechanikus bizonyítása (formális végrehajtás)

Formális módszerek felhasználása

- Más elemzési és tervezési módszerek kiegészítésére
- Bugok felderítése (kódban, specifikációban)
- Fejlesztési idő csökkentése (tesztelés)
- Bizonyos rendszer tulajdonságok biztosítása
- Automatizálás lehetővé tétele
- Fejlesztés minőségbiztosítása

Alkalmazási területek

- real time adatbázisok
- hardver tervezés
- biztonság kritikus alkalmazások (orvosi, nukleáris, hadi technikai, közlekedési)

Motiváció

- A technológiai rendszerek kis hibái katasztrófákhoz vezethetnek
- A mindenütt jelenlévő szoftverek hibái egyre több területen okoznak meghibásodásokat
- A szoftver minőség egyre jelentősebb gazdasági és jogi probléma

Minőségbiztosítási kérdések

- Minőség megközelítések
  - folyamat (gyártás) minőség -> termék minőség
  - szolgáltatás minőség
- Szoftver fejlesztés specialitása
  - nem egy helyes konstrukciót kell megfelelő minőségben reprodukálni => hangsúly a konstrukción

A megbízhatóság mérnöki megközelítése

- Precizitás (mérések, számítások, becslések)
- Redundancia (“túl méretezés/tervezés”)
- Robusztus tervezés (kis hibák ne legyenek katasztrófálisak)
- Alrendszerek szeparálása
- Alkalmas tervezési minták, “best practice” követése

Szoftver-rendszerek problémái

- A szoftverek nem folytonos függvényként működnek
- A HW redundancia nem segít a szoftverhibákon
- Az alrendszerek nem szeparálhatók tisztán
- A költség hatékonyság fontosabb a megbízhatóságnál
- A megbízható szoftverek tervezése nem kiforrott

Szoftverek korrektségének biztosítása

- Általános stratégia: tesztelés
- Más megközelítések: szoftver folyamat minőségbiztosítás, módszertanok
- Tesztelés
  - szoftver hibák ellen: specifikált működés ellenőrzése
  - külső hibák tűrésére: hiba injektálás, hibaterjedés ellenőrzése

Bevezetés

A tesztelés korlátai

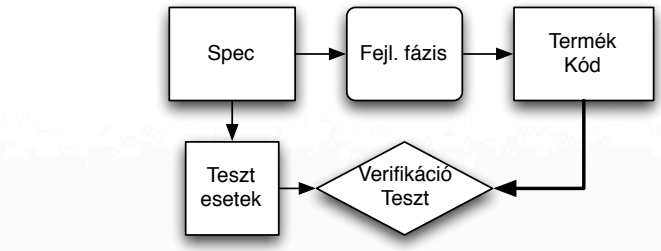
- A tesztelés célja hibák felfedése (javításra) és nem a hibátlanság igazolása
  - kimerítő tesztelés lehetetlen
- A tesztesetek reprezentativitása központi kérdés
  - váratlan, ritka esetek?
- A tesztelés költséges
  - tesztesetek tervezése, végrehajtása

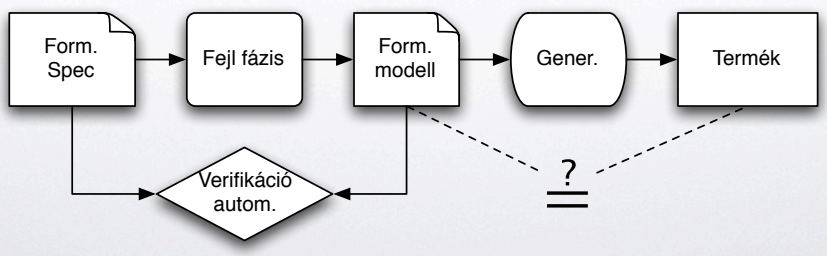
Formális módszerek és tesztelés

- Tesztelési kihívások
  - Tesztesetek előállítása
    - random (hibafelfedés?, olcsó)
    - intelligens (kézi, munkaigényes)
    - *automatikus* (formalizált spec. szükséges)
    - lefedési kérdések
  - Végrehajtás (megfigyelés)
- Formális módszerek (model alapú) alk. teszteset generálásra

Bevezetés

Formális módszerek és tesztelés





Specifikálás - követelmények

- Elvárt jellemzők
  - biztonsági jellemzők (pl. kölcsönös kizárás)
  - élőségi jellemzők (pl. éhezés elkerülése)
  - konkurencia, elosztottsági jellemzők (pl. holtpont mentesség)
  - nem funkcionális követelmények (pl. futási idő, memória használat, használhatóság)

Specifikálás (folyt.)

- A teljes viselkedési specifikálás igényelné:
  - a kód teljesíti a funkcionalitására vonatkozó előírásokat
  - adatkonzisztencia és invariancia előírásokat
  - modularitás, egységbezárási kérdések leírását
  - fejlesztési fázisok termékei közötti ekvivalencia és finomítási viszonyok leírását

Formális módszerek alkalmassága

- A formális bizonyítás helyettesíthet sok (akár végtelen) tesztesetet
- A formális módszerek javítják a specifikálás minőségét
- A formális módszerek garantálhatnak bizonyos rendszer-jellemzőket

FM alkalmassága (folyt.)

- Nem alkalmasak a teljes rendszer korrektségének bizonyítására
- Nem helyettesítik teljes egészében a tesztelést
  - nem natív kód szintjén működnek
  - sok nem formalizálható jellemző van
- Nem helyettesítik a megfelelő tervezési eljárásokat

A formális modellezés nehézségei

- A valóság leképezése
  - formális követelmény specifikációra
    - túl egyszerűsítés
    - nem teljesség
  - formális működési/végrehajtási modellre
    - modellezés (leképezés) pontossága

Bevezetés

Specifikáció formalizálása

- A formális specifikáció jól formáltsága és konzisztenciája géppel ellenőrizhető
- A formálisan definiált specifikáció nem teljessége észlelhető
- Az implementáció verifikálásának sikertelenségéből vissza lehet következtetni a specifikáció hibáira

Rendszerek leírása

- Leírási szintek
  - Absztrakt szint
    - Véges állapottér
    - Automatikus bizonyítás elvben lehetséges
    - Az egyszerűsítés elkerülhetetlen
  - Konkrét szint
    - Nem véges és komplex adatszerkezetek
    - Valódi programozási nyelvi modellek
    - Automatikus bizonyítások általában lehetetlenek

Rendszerek leírása (folyt.)

- Leírások kifejezősége
  - Egyszerű leírások
    - egyszerű, általános jellemzők
    - véges sok megkülönböztetett eset
    - alacsony precizitás
    - automatikus bizonyítások lehetősége
  - Komplex leírások
    - teljes viselkedési leírás
    - végtelen domének feletti kvantifikálás
    - magas precizitás

Automatikus tételbizonyítás

- Automatikus bizonyítás (batch mode)
  - a bizonyításközben nem kell interakció
  - eszközök paramétereket be kell hangolni
  - formális specifikálás kézzel
- Félautomatikus bizonyítás (interaktív)
  - bizonyítás közben kellhet interakció
  - ismerni kell az eszköz belső működését
  - a bizonyítást *ellenőrzi* az eszköz

Modell ellenőrzés

- Annak az automatikus ellenőrzése, hogy egy modell rendelkezik-e bizonyos jellemzőkkel (pl. holtpont mentesség)
- Ipari alkalmazás
  - HW verifikálás
  - SW verifikálás
    - vezérlő rendszerek, protokollok
    - absztrakciók ellenőrzése

Szemantikai kérdések

- Miről (és miért) szól a formális modell?
  - operációs szemantika (állapotok, akciók)
  - axiómatikus szemantika (helyesség bizonyítás)
  - jelölési szemantika (metamodellek, kódgenerálás)

Kihívások

- A valódi szoftver rendszerek hatalmasak és komplexek
  - az elosztottsággal komplexitási robbanás jár
- Eldönthetetlenség
- Mérnöki kérdések
  - erőforrások (emberi, gép, idő)
  - kielégítő megoldások (elfogadható megbízhatóság/kockázat)

Kihívások (folyt.)

- Milyen szintű absztrakció kell?
  - fizikai, áramköri, hálózati szint
  - bináris, bájtkód, forráskód
  - architektúra, algoritmus, folyamat, modell
- Mik a lényeges jellemzők?
  - típus biztonság, részleges/teljes korrektség, holtpont, éhezés, élőség, reaktivitás

Bevezetés

Kihívások (folyt.)

- Tudásgát
  - erősen absztrakt
  - munka ráfordítás igényes
  - néhány területen nincsenek eszközök
    - diszkrét~folytonos idő
    - idővariáns nem lineáris rendszerek

Occam borotvája

- Heurisztika - a legegyszerűbb még megfelelő modellt kell választani
- Absztrakciók
  - irreleváns részletek mellőzése -> fontos jellemzők maradnak
  - további redukció a komplexitás csökkentésére
- Elfogadott, bevált modellek alkalmazása

Szoftver rendszerek modellezése

- Különböző rendszer nézetek
  - funkcionális/relációs/szekvenciális nézetek
    - a rendszer mint input/output reláció, függvény
    - absztrakt adattípusok a rajtuk értelmezett műveletekkel (szekvenciákkal)
  - struktúrális nézetek
    - komponensek és kapcsolataik
    - OO modellezés

Szoftver rendszerek modellezése

- Viselkedési modellek
  - reaktivitás
  - konkurencia/elosztottság
  - interakció ismeretlen környezettel
- Modellezési szempontok, kritériumok
  - fontos jellemzők
  - funkcionális korrektség, terminálás, holtpontok, éhezés

Bevezetés

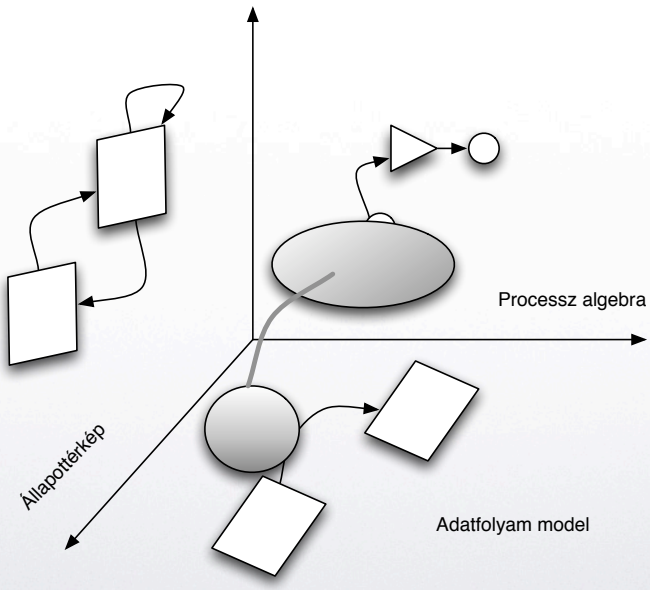
Szoftver rendszerek modellezése

- Konkurens/Interaktív rendszerek
  - nem determinisztikusan előforduló események
  - állapot átmenet rendszerek
    - számítás = állapotok szekvenciája átmenetek mentén
    - állapot = rendszer (globális) állapot
    - átmenetek = állapotok változása
  - végrehajtási modellek
    - lineáris (számítási utak)
    - elágazó (számítási fák)

Konkurrens reaktív rendszerek

- Reaktív rendszer: interakció a környezettel, nem termináló rendszerek (kommunikációs protokollok, HW áramkörök)
- Konkurrens rendszerek: egyidőben működő komponensek
  - aszinkron (interleaved) rendszerek (egyszerre csak egy komponens lép)
  - szinkron rendszerek (minden komponens egyidőben lép)

Több szempontú leírás



Formális módszer használata

- Fogalmi tér kialakítása
- Probléma formalizálás ezen keretek között
- Formális modell elemzése
- Elemzési eredmények használata

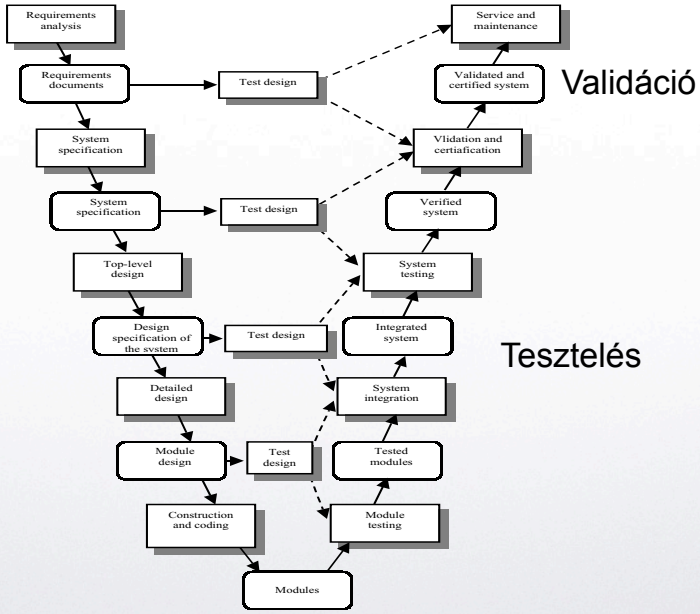
Bevezetés

Biztonság kritikus szoftverek

- Szabványokban fejlesztési előírások
  - követelmény specifikációra
  - tervezésre és fejlesztés
  - formális eljárások használatának követelménye

Verifikáció és validáció

Statikus ellenőrzés
Modell végrehajtás
Formális verifikáció



Validáció

Tesztelés

Bevezetés

Formális módszerek a gyakorlatban