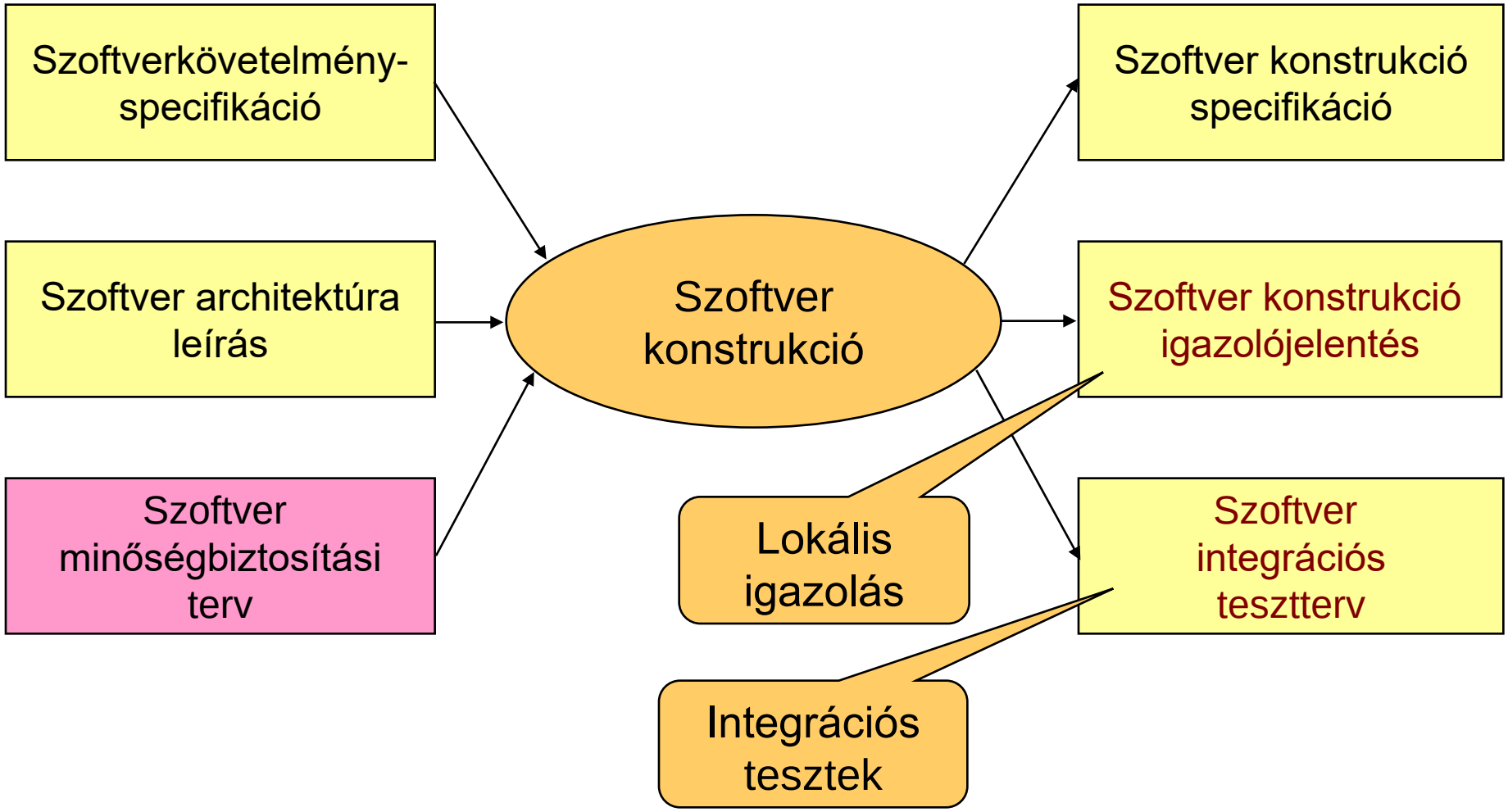
Részletes szoftver tervek ellenőrzése

Majzik István
Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék
http://www.mit.bme.hu/~majzik/

Tartalomjegyzék

- A részletes tervek elkészítése
  - Szoftver konstrukció
  - Modul tervezés
- Ellenőrzések
  - Verifikációs lépések
  - Formális verifikáció

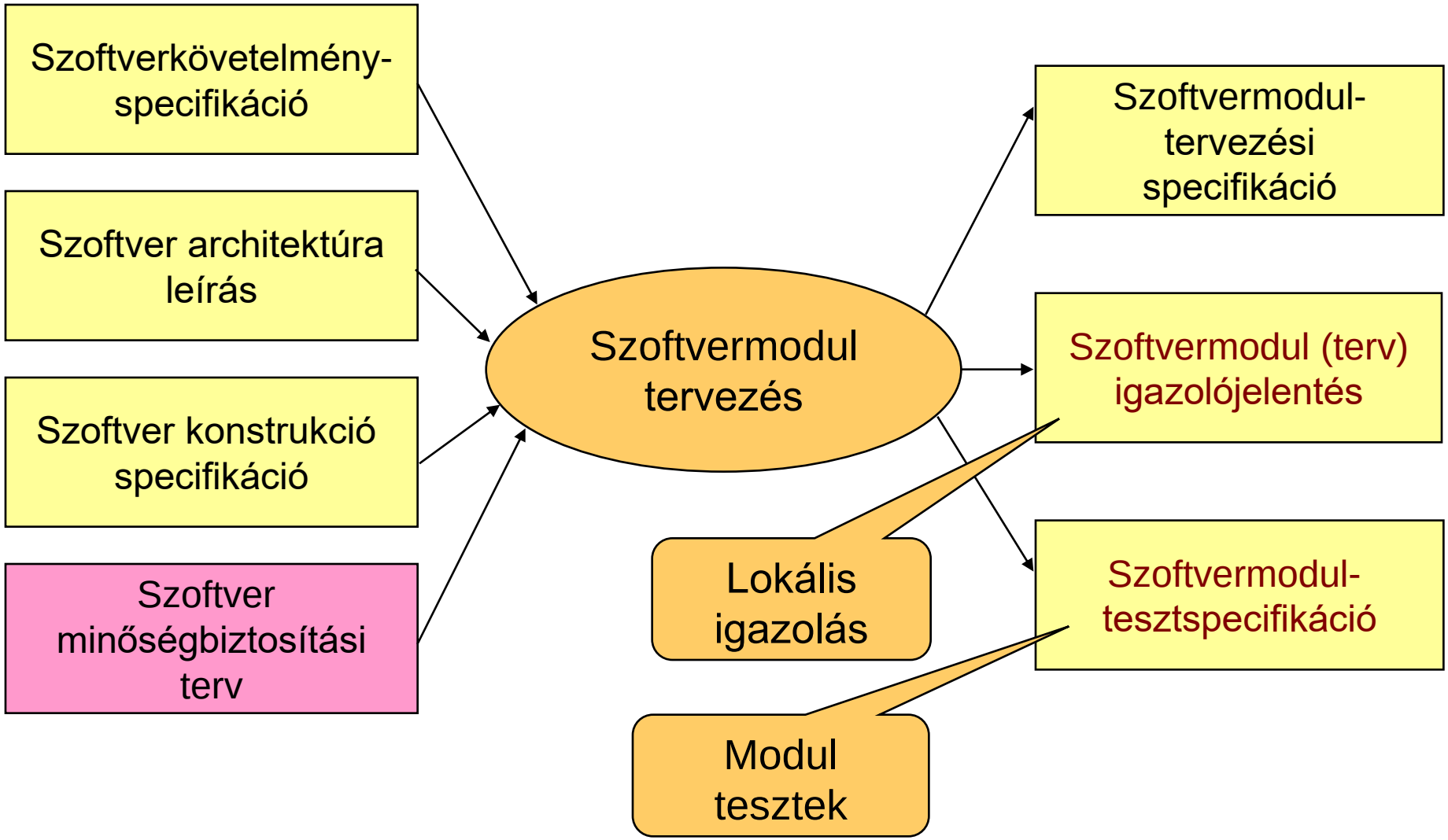
Szoftver konstrukció



Szoftver konstrukció

- Meghatározandó:
  - Rendszerszintű algoritmusok a modulok együttműködéséhez
  - Globális adatstruktúrák
- Használt leíró nyelv:
  - Információáramlás leírása (sorrendiség, időbeliség)
  - Adatstruktúrák leírása
  - Absztrakció és modularitás, verifikálhatóság
- Választható módszerek:
  - Formális, félformális, strukturált módszerek
- Speciális előírások (biztonságkritikus rendszerekben):
  - Teljesen meghatározott interfészek
  - Modulméret korlátozás, információrejtés
  - Paraméter mennyiségi korlátozás

Szoftvermodul tervezés



Szoftvermodul-tervezési specifikáció

- Modulok belső tervezése
  - Algoritmusok
  - Adatstruktúrák
- Használt leíró nyelv:
  - Implementációközeli nyelvek
    - Pl. pszeudo-kód is
  - Absztrakt(abb) nyelvek
    - Formális, félformális, strukturált metodika
    - Viselkedés leírás is hangsúlyos

Tartalomjegyzék

Verifikációs lépések

Ellenőrizendők az igazolási fázisokban:

- A részletes terv tulajdonságai és megfelelőségei
  - Teljesség, ellentmondás-mentesség, megvalósíthatóság, ellenőrizhetőség
  - Megfelelőségek: Előző fázisokban rögzített követelmények alapján
- Teszt tervek teljessége

Verifikációs módszerek:

- Statikus elemzés
  - Ellenőrzőlisták, hibabecslés
  - Vezérlési folyam elemzése (pl. strukturáltság)
  - Adatáramlás elemzése (pl. hozzáférési sorrend, inicializálás, felhasználás)
  - Határérték elemzés (pl. viselkedés határértékek esetén)
  - Nemkívánatos információáramlás („alattomos komponensek”)
  - Szimbolikus végrehajtás
- Dinamikus elemzés
  - Prototípus készítés és animáció
  - Szimuláció
- Formális verifikáció

Formális verifikáció

- Matematikai eszközök használata
  - Diszkrét matematika, matematikai logika
  - Formális nyelv: Formális szintaxis és szemantika
    - Tulajdonság leírás (követelmények: tulajdonság specifikáció)
    - Viselkedés leírás (terv, implementáció)
  - Matematikai algoritmus: analízis (verifikáció), szintézis
- Algoritmusok a verifikációhoz
  - „Önmagában való” vizsgálat (pl. ellentmondás-mentesség)
  - „Változások” vizsgálata (pl. tervezői döntések)
  - Tulajdonság leírás és viselkedés leírás összevetése
- Kritikus pont: A valós probléma formalizálása
  - Nem automatizálható
  - Egyszerűsítés, absztrakció gyakran szükséges (ezt validálni kell)

Formális szintaxis (áttekintés)

- Matematikai definíciók:

$KS = (S, R, L)$ és AP, ahol

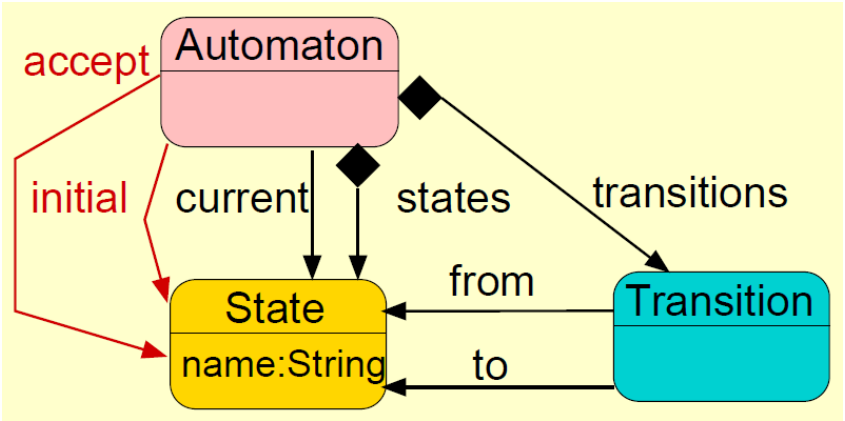
$AP = \{P, Q, R, \ldots\}$

$S = \{s_1, s_2, s_3, \ldots s_n\}$

$R \subseteq S \times S$

$L: S \rightarrow 2^{AP}$

- BNF: HML ::= true | false | p∧q | p∨q | [a]p | <a>p
- Metamodell:



- Absztrakt szintaxis (nyelvtani szabályok) és konkrét szintaxis (megjelenítés)

Formális szemantika (áttekintés)

- Műveleti (operációs) szemantika: „Programozóknak”
  - Megadja, mi történik a számítások során
  - Egyszerűbb formalizmusra épít: pl. állapotok, akciók
- Axiomatikus szemantika: „Helyességbizonyításhoz”
  - Állítás nyelv + axiómakészlet + következtetési szabályok
  - Elő- és utófeltételek adják meg a jelentést
  - Pl. automatikus tételbizonyító rendszerekhez
- Denotációs szemantika: „Fordítóprogramokhoz”
  - Szintaxis által meghatározott „jel −> jelölt dolog” leképzés
  - A „jelölt dolog” tipikusan mint matematikai domén adott
    - Számítási szekvencia, vezérlési gráf, állapothalmaz, …
      és ezeken definiált műveletek (összefűzés, unió, …)
  - A modellek vizsgálata a mögöttes matematikai objektum vizsgálatára vezethető le
  - Komponálható szemantika: |P op Q| = |op| (|P|,|Q|)

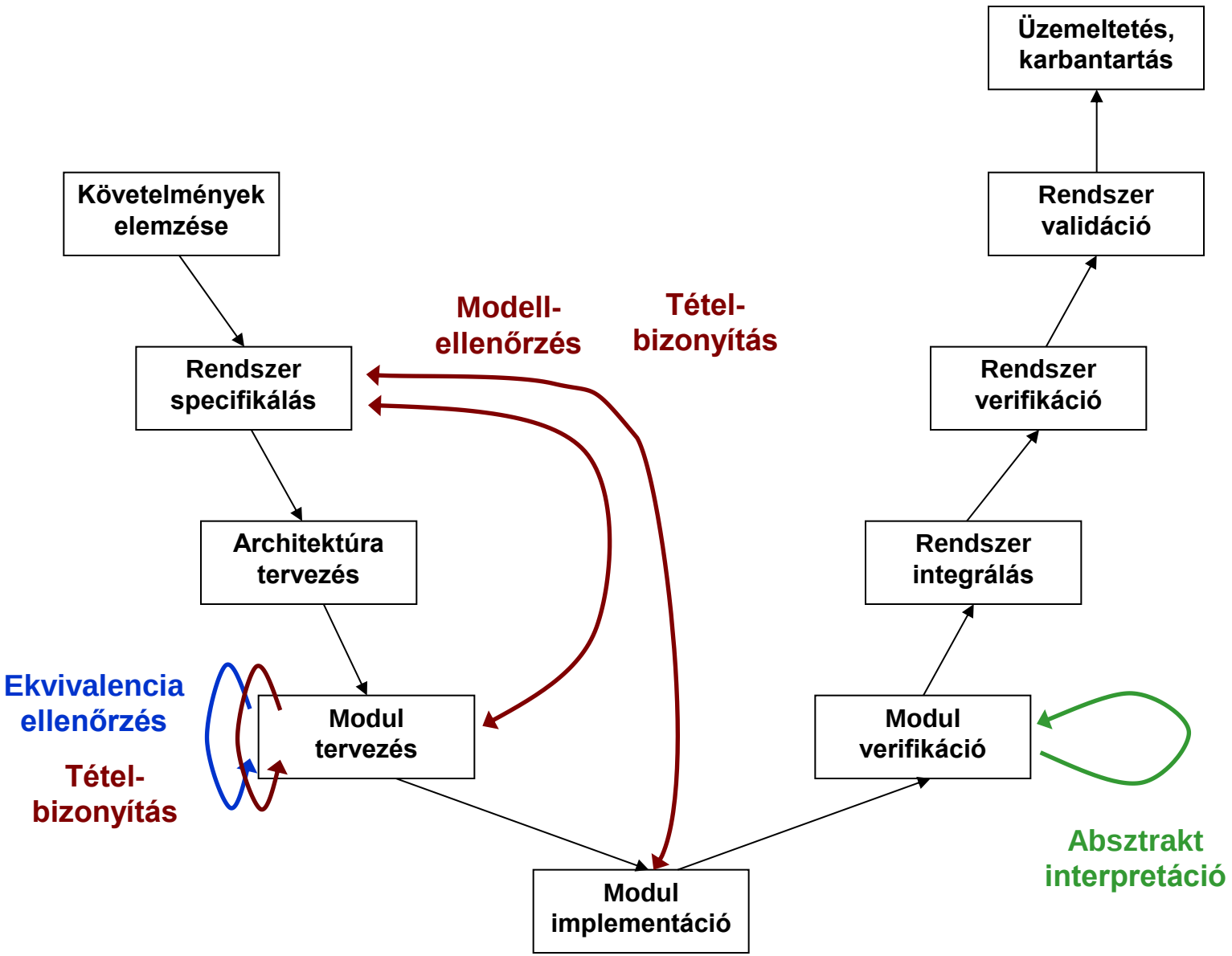
A legelterjedtebb formális verifikációs technikák

Modell / technika	**Viselkedés leírás (alapszintű)**	**Tulajdonság leírás (alapszintű)**
Modellellenőrzés	Kripke struktúra, Kripke tranzíciós rendszer	Temporális logika, elsőrendű logika
Ekvivalencia ellenőrzés	LTS (Labeled Transition System), automata	LTS, automata (mint referencia viselkedés)
Tételbizonyítás	Dedukciós rendszer	Bizonyítandó tétel (elsőrendű logika)
Statikus analízis (absztrakt interpretáció)	Kripke struktúra (programból absztrakcióval)	Elsőrendű logika, assertion

A technikák előnyei és korlátai

- Modellellenőrzés, ekvivalencia ellenőrzés
  - ☺ Teljesen automatikus, explicit kimerítő (teljes) vizsgálat
  - ☺ Ellenpélda generálás (debuggoláshoz)
  - ☹ Állapottér robbanás (részben kezelhető)
- Tételbizonyítás
  - ☺ Skálázható nagy rendszerekre (pl. indukció)
  - ☺ Nagy kifejezőerő
  - ☹ Interaktív (segítséget igényel, pl. ciklus invariánsok megtalálása, bizonyítási stratégia megadása)
  - ☹ Nincs ellenpélda
- Statikus analízis (absztrakt interpretáció)
  - ☺ Állapottér csökkentés
  - ☹ Absztrakció nehezen automatizálható

Verifikációs technikák szerepe



Tervek és modellek szerepe a formális verifikációban

- Viselkedés leírás (modell)
  - Alapszintű:
    - KS, LTS, KTS, automaták, Büchi automaták
  - Magasabb szintű:
    - Vezérlés orientált: Hierarchikus automata, Petri-háló
    - Adatfeldolgozás orientált: Adatfolyam háló
    - Kommunikáció orientált: Processz algebrák
  - Mérnöki modellek:
    - UML diagramok formális szemantikával
- Tulajdonság leírás (követelmény)
  - Alapszintű:
    - Elsőrendű logika, temporális logika, referencia automata
  - Magasabb szintű:
    - Scenario: MSC, LSC, PSC, …

Összefoglalás

- A részletes tervek elkészítése
  - Szoftver konstrukció
  - Modul tervezés
- Ellenőrzések
  - Verifikációs lépések
  - Formális verifikáció