

# CASE-eszközök

Első alkalom



# Tantárgyi ismertető

# Oktatási cél

---

- ▶ A tárgy keretében a hallgatók megismerkednek a szoftverfejlesztő programkörnyezetekkel, a CASE eszközökkel. Az eszközök bemutatásán, jellemzésén túl az eszközök általános felépítése, kiválasztásuk szempontjai kapnak fokozott hangsúlyt.
- ▶ A félévben sorra kerülő laborfoglalkozásokon a hallgatók a CASE eszközök széles skálájával és azok alkalmazási lehetőségeivel ismerkednek meg.

# Tematika

---

- ▶ A CASE eszközök jellemzése
  - ▶ Fejlődésük, a fejlődés egyes állomásai, a mai eszközök általános jellemzői, kategóriái, a kategóriák jellemzői. Általános követelmények CASE eszközökkel szemben
- ▶ A CASE eszközök felépítése
  - ▶ Az ECMA referenciamodell. A modell egyes szintjeinek jellemzői.
- ▶ A piacon fellelhető CASE eszközök
  - ▶ CASE eszközök kiválasztási szempontjai: az értékelés kritériumai, műszaki, kereskedelmi és gazdaságossági szempontok figyelembevétele. A legelterjedtebb objektumorientált szoftvefejlesztést támogató CASE eszközök.
- ▶ Alacsonyszintű CASE eszközök
  - ▶ Az implementálást és tesztelést támogató eszközök: nyomkövető, programelemző és teszteset-generáló rendszerek.



# Félévközi követelmények

---

- ▶ ***Aláírást*** az a hallgató kaphat, aki egy általa választott és az oktató által jóváhagyott, a szoftverfejlesztés folyamatát támogató szoftver jellemzéséből és alkalmazásából megfelelő minőségű előadást tartott valamelyik órán
- ▶ ***„Aláírás megtagadva, pótolható”*** bejegyzést kap az a hallgató, aki az órákon rendszeresen aktívan részt vett, de az előadást nem tartotta meg.
- ▶ ***„Aláírás megtagadva, nem pótolható”*** bejegyzést kap az a hallgató, aki az órákon nem vett részt rendszeresen és aktívan.



# Vizsga

---

- ▶ Pótlás módja:
- ▶ Az a hallgató, aki nem szerzett aláírást a félév folyamán, de a pótláshoz előírt feltételeket teljesítette, egy előre rögzített időpontban kísérletet tehet a javításra. Ekkor a félév során megismert szoftverekkel kapcsolatos kérdéseket kap, és a megszerezhető pontszám 50%-át kell elérnie a vizsgára bocsátáshoz.
  
- ▶ A vizsgajegy kialakításának módszere:
- ▶ A vizsga írásbeli jellegű. Érdemjegye a dolgozatban elért eredmény függvényében.
- ▶ Az érdemjegy: elért százalék

▶ 5	> 87%
4	75 – 87%
3	62 – 75%
2	50 – 62%
1	< 50%




# A CASE eszközökről általában

# C A S E

Computer  
Aided / Assisted  
Software / System  
Engineering





# A CASE-eszközök fogalma

---

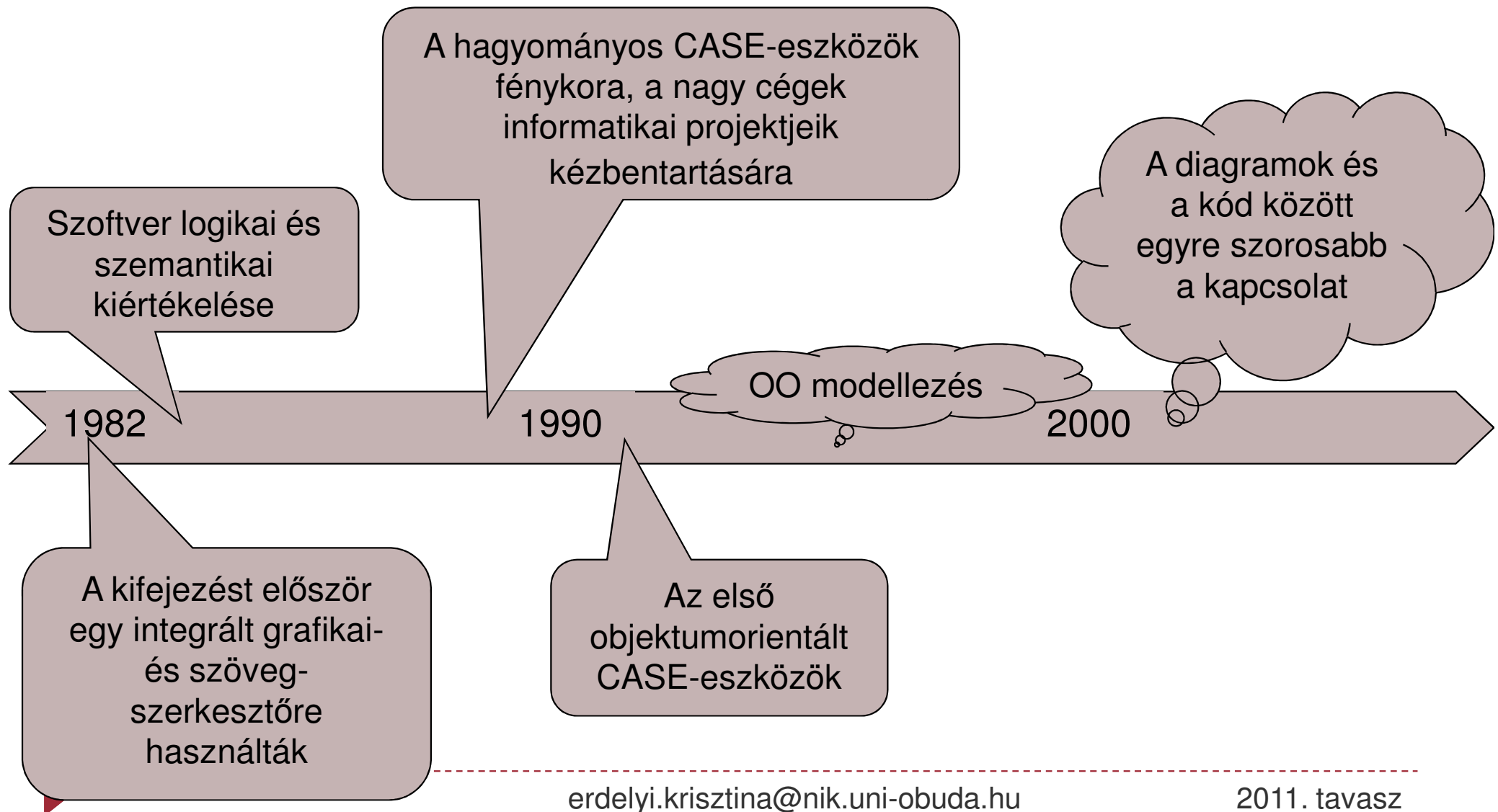
- ▶ Computer-aided Software Engineering:
  - ▶ A szoftverfejlesztést és karbantartást segítő eszközök
  - ▶ A projektmenedzsmentet és a csapattagok közti kommunikációt megvalósító rendszerek
- ▶ Fajtái:
  - ▶ Magasszintű CASE-eszközök:  
A szoftverfejlesztési életciklus analízis fázisára fókuszálnak
  - ▶ Alacsonyszintű CASE-eszközök:  
Az implementálást és tesztelést támogató eszközök



# A CASE-eszközök fejlődése

# A CASE-eszközök fejlődése

---



# 60-as és 70-es évek

---

- ▶ 60-as évek

- ▶ szövegszerkesztők specializálódása, programírás támogatása (TECO)

- ▶ 60-as évek vége, 70-es évek eleje

- ▶ grafikus szerkesztők megjelenése (HIPO diagramok, strukturált adatfolyam diagramok)

- ▶ 70-es évek

- ▶ az adatfolyam alapú tervezés és a strukturált analízis előretörése megkívánta az adatszótárak alkalmazását, ami lehetővé tette automatizmusok, szabályok beépítését és ezáltal a dokumentumok ellenőrzését, illetve bizonyos dokumentumok automatikus generálását

# 80-as évek

---

## ▶ 80-as évek eleje

- ▶ Computer Aided Documentation
- ▶ Computer Aided Diagramming
- ▶ Analysis and Design Tools

## ▶ 80-as évek közepe

- ▶ Automatic Design Analysis and Checking
- ▶ Automated System Information Repository

## ▶ 80-as évek vége


- ▶ Automatic Code generation from Design Specification
- ▶ Linking Design automation

# 90-es évektől

---

- ▶ 90-es évek eleje
  - ▶ Intelligent Methodology Driver
  - ▶ Habitable User Interface Reusability as a Development Methodology
- ▶ És napjaink





# Funkcionális nézőpont szerinti osztályozás

# Funkcionális nézőpont szerinti osztályozás

---

- ▶ A CASE-eszközöket sajátos funkcióik alapján osztályozzuk
- ▶ Ez alapján a következő eszköztípusokat határoztam meg:
  - ❑ Tervezői eszközök (pl. becselő eszközök)
  - ❑ Szerkesztő eszközök
  - ❑ Változtatáskezelő eszközök
  - ❑ Konfigurációkezelő eszközök
  - ❑ Prototípuskészítő eszközök (pl. felhasználói interfész generátorok)
  - ❑ Módszertámogató eszközök
  - ❑ Nyelvi feldolgozó eszközök
  - ❑ Programelemző eszközök
  - ❑ Tesztelő eszközök
  - ❑ Nyomkövető eszközök
  - ❑ Dokumentációs eszközök
  - ❑ Újratervezési eszközök
  - ❑ Kommunikációs eszközök
  - ❑ Projektkezelő eszközök





# Business System Planning Tools

---

- ▶ Lehetővé teszi informatikai szempontból a vizsgált szervezet modellezését. (Információs egységek: kibocsátó, nyelő; információ típusok: jelentések, igények, listák; információ áramlási utak)
- ▶ pl: Interactive Engineering Workbench (knowledgeware)

# Projectmanagement Tools

---

## ▶ Project Planning Tools

- ▶ segítséget nyújt a project méretének (ráfordítás, költség) felmérésében
- ▶ a project jellemzőinek felmérése (komplexitás, tapasztalat, stb.)
- ▶ lehetőséget ad a paraméterek változtatásával „játékra”, hangolásra.
- ▶ lehetővé teszi (támogatja) a project „schedulinget” task-ok definiálása, pathnetwork megadása CPM támogatásával)
- ▶ pl: DEC Plan

## ▶ Requirement Tracing Tools

- ▶ célja: az igény specifikáció elemzése, formalizálása, konformitásának biztosítása
- ▶ a specifikáció szövegét elemzi és hasonlítja össze az adatbázisban eltárolt objektumokkal
- ▶ pl: Teamwork / RQT (Cadre Technologies)

## ▶ Metrics and Management Tools

- ▶ támogatja a project menedzselését, vezénylését (sokszor szakértői rendszerrel is) és project haladásának mérését
- ▶ pl: checkpoint (Software Productivity Research)



# Support Tools

---

## ▶ Documentation Tools

- ▶ A cégek a ráfordítás 20-30%-át a dokumentációk elkészítésére áldozzák.
- ▶ Ezért általában az analízis-tervező tool-ok adatkapcsolatban vannak egy DTP alapú dokumentáció készítő tool-lal.
- ▶ pl.: PageMaker (Aldus)

## ▶ System Software Tools

- ▶ az operációs rendszer nyújtotta szolgáltatások összessége
- ▶ pl: UNIX, VMS

## ▶ Quality Assurance Tools

- ▶ Az elkészítendő software minőségére ügyelő, biztosító tool.
- ▶ Mindig tartalmaz minőség ellenőrző részt is. Pl. forráskódot összehasonlít előre megadott nyelvi szabállyal.
- ▶ pl: Auditor (Software Corporation)

## ▶ Database and SCM Tools

- ▶ SCM - Software Configuration Management
- ▶ az adatbázisok az egyik alapja a CASE eszközöknek, ill. a fejlesztés mechanizmusának
- ▶ széles tartományban mozognak az egyszerűtől a relációson át az objektum orientált adatbázisig
- ▶ a konfiguráció menedzsmenthez előnyösebb az OODMS alkalmazása, mert jobban tükrözi a valóságot
- ▶ feladata: a konfiguráció (HW, SW) összeállítása, a változások ügykezelése (jelentések kiadása mindenkivel, akit érint)
- ▶ pl: CCC (Softool Corp)



# Analysis and Design Tools

---

## ▶ SA/SD Tools

- ▶ A tool-ok jó része a strukturált analízis és tervezés metodikáján alapul.
- ▶ Ez egy igen elterjedt modellezési technika, ami a követelmény specifikációtól kiindulva a szerkezeti tervezésig jut el.
- ▶ speciális jelölésrendszert alkalmaz, analízis és tervezési heurisztikát, valamint egy transzformációt (mapping) az analízistől a tervezésbe.
- ▶ pl: Teamwork (Cadre)



## PRO/SIM Tools (Prototyping and Simulation)

- ▶ Lehetőséget biztosít a még nem létező rendszer bemutatására, biztosítva ezzel a megrendelőnek a készülő rendszerről alkotott kép tökéletesítését. Jobb specifikáció, pontosabb munka → kisebb költség
- ▶ A legtöbb rendszer lehetőséget ad a készülő termék funkcionális és viselkedési modelljének elkészítésére (viselkedés pl.: feldolgozási sebesség, adat output sebesség)
- ▶ lényegében a modellezés eszköze egy formális specifikáció, amiből az újabb rendszerek már (pl: Ada) kódot tudnak generálni.
- ▶ pl.: ADAS (Cadre Technologies)

## ▶ Interface Design and Development Tools

- ▶ A generált kódok 50-80%-a valamilyen kapcsolatban van a User Interface kezelésével.
- ▶ User Interface Tool Kits
- ▶ User Interface Management System
- ▶ User Interface Development System
- ▶ pl: WINDOWS (Microsoft), Motif (Open Software Foundation)

## ▶ Analysis and Design Engine

- ▶ Az analysis and design tool-ok új generációját hívják analysis and design engines-nek.
- ▶ Ezek szabály alapú architektúrával rendelkeznek, mely lehetővé teszi, hogy a tool tetszőleges analízis és tervező módszerre átállítható legyen.
- ▶ Minden ilyen tool tudja az SA/SD-t de támogatják a JSD, DSSD, DAST, HOOD-ot is
- ▶ pl: DEC Design



# Programming tools

---

## ▶ Conventional Coding Tools

- ▶ editor, compiler, debugger
- ▶ hosszú ideig ezek voltak az egyetlen Software tool-ok. Ezért sokáig azt hitték, hogy az SE probléma lényegében kódolási probléma.
- ▶ Ma már a megelőző fázisok tool-jait követő (eredményeit felhasználó) eszközzé váltak

## ▶ Fourth-Generation Coding Tools

- ▶ lehetővé teszi a rendszer megfogalmazását magasabb absztrakciós szinten
- ▶ nem csak a kódgenerálást biztosítja, hanem a specifikáció korrektségének ellenőrzését is támogatja, segítve ezzel a felhasználónak jobban megfelelő program készítését

## ▶ Object-Oriented Programming Tools

- ▶ az OOP megjelenésével és elterjedésével tömegével jelentek meg a piacon a tool-ok is.
- ▶ általában harmadik generációs felhasználói interfésszel rendelkeznek (windows, egér)
- ▶ pl: Eiffel (Interactive Software Engineering), Smalltalk



# Integration and Testing Tools

---

## ▶ Static Analysis Tools

- ▶ kód alapú teszt tool
- ▶ specializált teszt nyelv
- ▶ követelmény specifikáció alapjai
- ▶ pl: BAT / ACT (McCabe)

## ▶ Dynamic Analysis Tools

- ▶ intrusive tool: (a SW-be beillesztett utasítások segítik a dinamikus tesztelést)
- ▶ nonintrusive tool: (HW, párhuzamos processzor segíti a tesztelést)
- ▶ általában időket, időviszonyokat ad meg
- ▶ pl.: Software Analysis Workstation (Cadre Technologies)

## ▶ Test Management Tools

- ▶ vezérli és koordinálja a tesztelés lépéseit, fájlban tárolva rögzítjük a teszteseteket és adatokat
- ▶ Általában a felhasználó által konfigurálható
- ▶ pl.: DTM (DEC)



# Prototyping Tools

---

- ▶ drawing tools
- ▶ screen painter tools
- ▶ data design tools



# Maintenance Tools

---

## ▶ Reverse Engineering Tools

- ▶ statikus tool-ok: forrásprogramot olvasva kirajzolják grafikusan a program szerkezetét, vezérlési szerkezetét, adatfolyamát
- ▶ dependency analysis tools: felrajzolja a függőséget a különböző modulok között
- ▶ Dynamic Reverse Engineering Tools: futás közben elemzi a program viselkedését (monitorozik)
- ▶ pl: PathMap (Cadre Technologies)

## ▶ Re-engineering Tools

- ▶ code restructuring tools: kód átalakítását, átstrukturálást támogatja
- ▶ adat reengineering tools: interaktív módon ad lehetőséget az adatszerkezet változtatására és a változtatás hatásának elemzésére





# CASE eszközök csoportosítása II.

# Case eszközök csoportosítása II.

---

- ▶ **Módszer-orientált CASE eszközök**
  - ▶ ebbe a csoportba tartozik a CASE eszközök többsége
  - ▶ az analízis, és a tervezés fázisát támogatják, gyakran tartalmazznak kódgenerátort is
  - ▶ szöveges és grafikus felülettel rendelkeznek
  - ▶ A módszerek széles skáláját támogatják az SA/SD -től az UML-ig
- ▶ **Programozási nyelv-orientált CASE eszközök**
  - ▶ általában egy nyelvre épülnek
  - ▶ az egyes eszközök szoros integrációja jellemzi
  - ▶ pl.: ADA környezet, Smalltalk környezet

# Case eszközök csoportosítása II.

---

- ▶ Adminisztratív CASE eszközök
  - ▶ a szoftverkészítési folyamat egyes részeit támogatja
  - ▶ pl.: konfiguráció menedzsment
- ▶ Szerszámkészletek (Tool-ok)
  - ▶ nem a hagyományos értelemben vett CASE eszközök, inkább különböző funkciók, alacsony szintű szolgáltatások összevonása egységes környezetben
  - ▶ pl.: UNIX Programming Work Bench
- ▶ Struktúra-orientált CASE eszközök
  - ▶ az eszközök a dokumentumok alapvető struktúráját használják (szöveghez kötött és/vagy szöveg-érzékeny szintaktika)
  - ▶ szorosan együttműködő szerkesztést, analízist és transzformációt és végrehajtást végző eszközök
  - ▶ pl.: IPSEN, Mentor

# Case-eszközök csoportosítása II.

---

- ▶ Meta-CASE eszköz
  - ▶ CASE eszközök CASE eszközök készítésére
  - ▶ IPSEN, Mentor

# Általános követelmények CASE eszközökkel szemben

# A cél: Minőségi szoftver létrehozása

---

- ▶ A szoftvertechnológia célja minőségi szoftvertermék létrehozása
- ▶ Ehhez sokféle eszköz és módszertan áll rendelkezésre



# Kérdések a szoftverfejlesztés közben

---

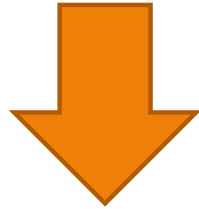
- ▶ Hogyan kezdjük neki az elemzésnek?
- ▶ Hogyan tudjuk kezelni az elemzők csoportjait?
- ▶ Mikor kell abbahagynunk az elemzést és rátérni a tervezésre?
- ▶ Honnan tudjuk, hogy a terv helyes?
- ▶ Mikor kezdjük kódolni?



# Mai igények a szoftverekkel szemben

---

- ▶ A vásárlók magasabb színvonalat követelnek, mint pár évvel ezelőtt, tisztában vannak a lehetőségekkel
- ▶ Sokrétegű, sok szolgáltatást tartalmazó alkalmazások
- ▶ Webalkalmazások



- ▶ A szoftverek bonyolultsága nő, így a fejlesztés is egyre nehezebb



# A CASE eszközök fajtái

---

- ▶ Tervezői eszközök
- ▶ Szerkesztő eszközök
- ▶ Változtatáskezelő eszközök
- ▶ Konfigurációkezelő eszközök
- ▶ Prototípuskészítő eszközök
- ▶ Módszertámogató eszközök
- ▶ Nyelvi feldolgozó eszközök
- ▶ Programelemző eszközök
- ▶ Tesztelő eszközök
- ▶ Nyomkövető eszközök
- ▶ Dokumentációs eszközök
- ▶ Újratervezési eszközök
- ▶ Kommunikációs eszközök
- ▶ Projektkezelő eszközök



Manapság már a szoftverfolyamat minden részéhez létezik megfelelő eszköz

De  
mennyibe  
kerül ez  
nekünk?



# Pro és kontra

---

## A CASE-eszközök előnyei

- ▶ Segít a diagramok és jelölések egységesítésében
- ▶ Segíti a kommunikációt a fejlesztőcsapat tagjai között
- ▶ Automatikus ellenőrzéseket tesz lehetővé
- ▶ Időt és energiát takaríthatunk meg
- ▶ Elősegíti a modellek és komponensek újrahasznosítását

## A CASE-eszközök hátrányai

- ▶ A dokumentáció rugalmasságát korlátozza
- ▶ Az eszköz kapacitása korlátozó hatású lehet
- ▶ A szintaktikai helyesség és a teljesség még nem jelenti a követelmények teljesülését
- ▶ Az eszköznek költségei vannak



# A CASE-eszközök költségei

---

- ▶ Betanítás költségek
- ▶ Kipróbálás költségei
- ▶ Frissítés költségei
- ▶ A karbantartás és a folyamatos használat költségei
- ▶ Időveszteségek az eszköz használata miatt
  - ▶ Várakozás az eszközre
  - ▶ Túlzott precizitás, a diagram csinosítása
- ▶ Más eszközre való átálláskor felmerülő költségek
- ▶ A nem megfelelő integráció más eszközökkel csökkenti a termelékenységét



# Túl sok eszköz - Megoldások

---

- ▶ Integrált fejlesztői eszköz
  - ▶ Általában egy fejlesztői környezetbe épülve
  - ▶ Pl. Visual Studio Team System
- ▶ Csak azt használjuk, amire tényleg szükségünk van
  - ▶ Néha hatékonyabb a papír és a ceruza
  - ▶ Pl. agilis szoftvertervezési módszertanok vélik ezt



# Témakörök

---

- ▶ Projektmenedzsment támogatása
- ▶ Analízis, tervezés támogatása – modellező eszközök
  - ▶ ArgoUML
- ▶ Felhasználói felület tervezők
- ▶ Kódgenerálás
- ▶ Optimalizálás, teljesítménymérés
  - ▶ vTune
- ▶ Tesztelést segítő eszközök
  - ▶ Nessus, Openload, Watin, Nunit, Silk Test, Selenium ...
- ▶ Integrált fejlesztői környezetek eddig nem tanult lehetőségei
  - ▶ Visual Studio, NetBeans
  - ▶ Tesztelés, projektmunka, verziókezelés
- ▶ Dokumentáció készítése
- ▶ Szoftverkövetés

