MODELLALAPÚ SZOFTVERFEJLESZTÉS

1. ELŐADÁS

BEVEZETÉS

DR MEZEI GERGELY

A MAI ELŐADÁS

I. fejezet Miért?

II. fejezet Miről?

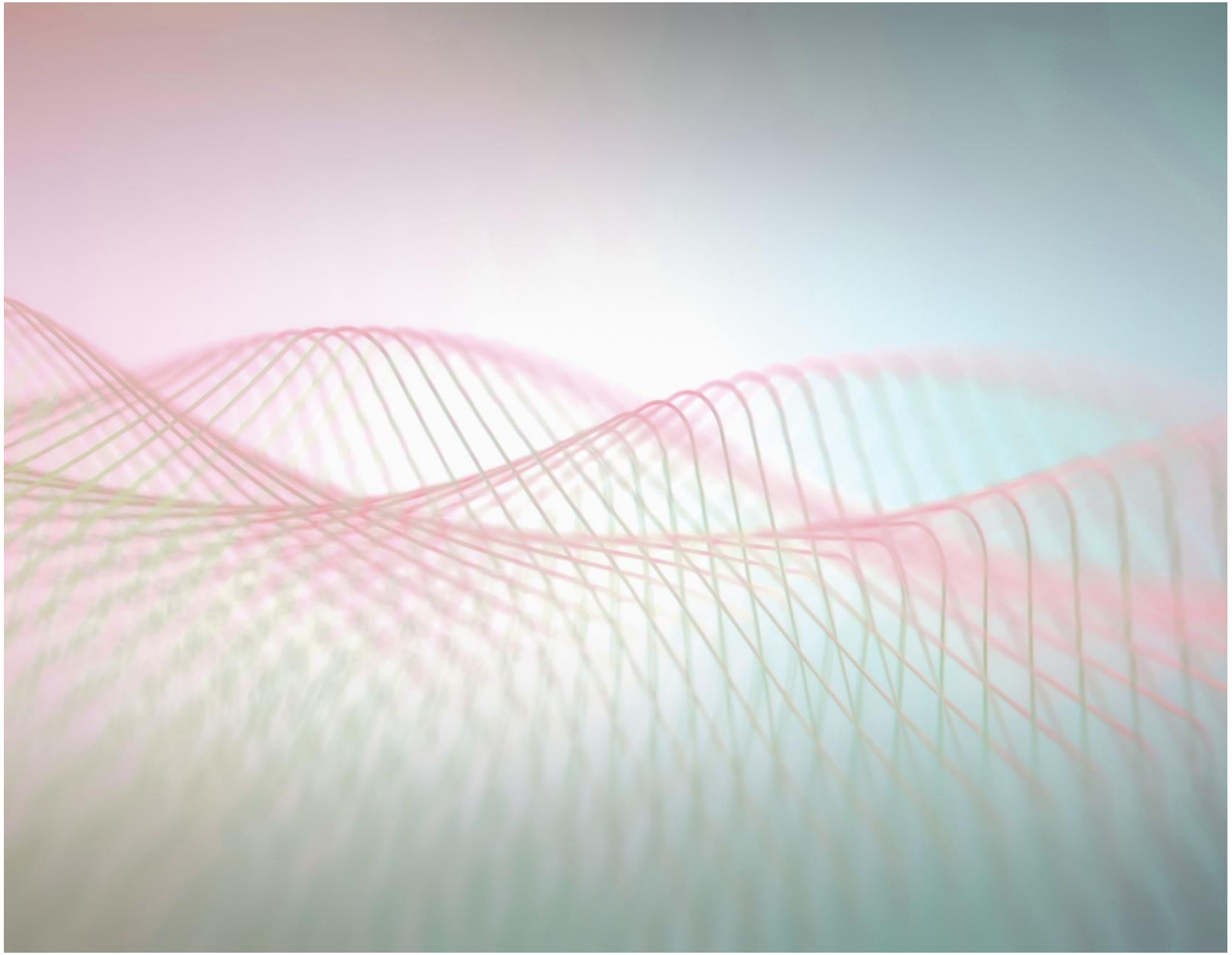
III. fejezet Hogyan?

A MAI ELŐADÁS

I. fejezet Miért?

II. fejezet Miről?

III. fejezet Hogyan?



MIÉRT MODELLEZÉS?

SZOFTVERFEJLESZTÉS – HOGYAN KEZDŐDÖTT?

- Tom Kilburn – Manchester Small-Scale Experimental Machine – 1948
  - Első programkód
  - Program = lyukkártya
  - 52 perc alatt: 2^18 = 262 144
- Fortran – 1957
  - Első magas szintű programozási nyelv
  - Fordítóprogram
- C nyelv – 1972
- Objektumorientált nyelvek: Simula (1967), Smalltalk (1970)

SZOFTVERFEJLESZTÉS – HOGYAN KEZDŐDÖTT?

- C++ - 1984
- Java – 1995
  - Java Virtual Machine – hordozhatóbb kód
- C# - 2000
  - Intermediate Language – nyelvek közti átjárhatóság
- Python, Swift, Go, Kotlin, …

HOVA TART MA A SZOFTVERFEJLESZTÉS?

- Mik az elvárások?
  - Növekvő alkalmazás méret
  - Csökkenő idő
  - Kevesebb hiba
  - Magasabb minőség
- Megoldás?

MERRE HALADUNK?

- Ne írd meg azt…
- … amit mások már megírtak!
  - Mindenre van (fél)kész megoldás (library, komponens)
  - Telepíts és konfigurálj kódolás helyett!
- … amit egy olcsóbb munkaerő is megírhat!
  - Code monkey-k, vagy méginkább: automatizálás és kódgenerálás
  - Miért nem írja meg a megrendelő?

A MEGOLDÁS: CHATGPT?

- Mottó: “ Természetes nyelven megfogalmazott specifikációból a mesterséges intelligencia működőképes alkalmazást gyárt ”
- Nehézségek
  - A specifikáció
  - A működőképes alkalmazás
  - Biztonsági garanciák, minőségbiztosítás

LOW CODE – NO CODE

- Mottó: "Termékfejlesztés programozók nélkül"
  - Grafikus felület, "összekattintgatós" alkalmazáslogika
  - Gyors fejlesztés
  - Limitált felhasználási terület
  - Üzleti fogalmak, ellenőrzések, folyamatok

LOW CODE – NO CODE

Low code

- Alkalmazásfejlesztés minimális kódolással
- Gyors betanulás és fejlesztés
- Általában grafikus szerkesztő
- Részben limitált kifejezőerő
- Fejlesztőknek és üzletembereknek

No code

- Alkalmazásfejlesztés kódolás nélkül
- Szinte nulla betanulás, azonnali fejlesztés
- Általában grafikus szerkesztő
- Limitált kifejezőerő
- Üzletembereknek

LOW CODE – NO CODE

- Erős trend
  - Microsoft PowerApps
  - Google App Sheet
  - Amazon Honeycode & Amplify Studio
  - Apple SwiftUI
  - 2024.-re a fejlesztések 65%-a ezen az alapon fut majd
- Mi a titok?

LOW CODE – NO CODE

- A siker titka
  - Beszéljünk a probléma nyelvén!
  - Koncentráljuk a tényleges feladatra!
  - Hagyjuk el a repetitív részeket!
  - Legyen tömör, átlátható!
  - Ne kelljen tudni hozzá programozni, csak ha muszáj!

A SZOFTVERFEJLESZTÉS – MA

- Igény: gyorsan, jót és sokat
- Megoldás:
  - Absztrakciós szintet kell növelni
    - *Assembly → C → C++ → Java/C# → …*
  - Konfiguráció programozás helyett
    - *Mindenre van félkész megoldás*
  - Generálni, amit csak lehet
    - *C++ template, generált constructor + destructor, property*

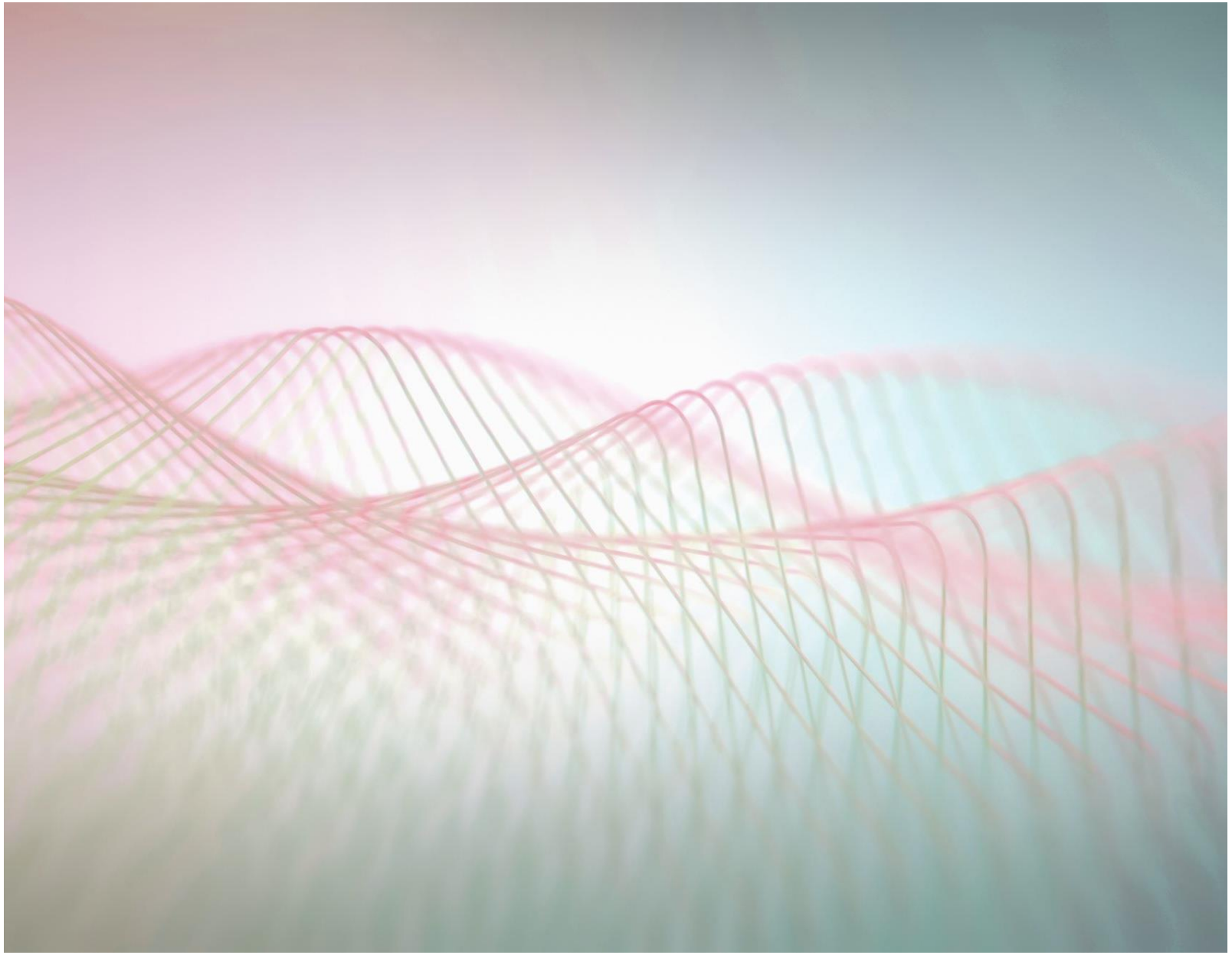
…. pontosan ezt adja a **Modellezés**

A MAI ELŐADÁS

I. fejezet Miért?

II. fejezet Miről?

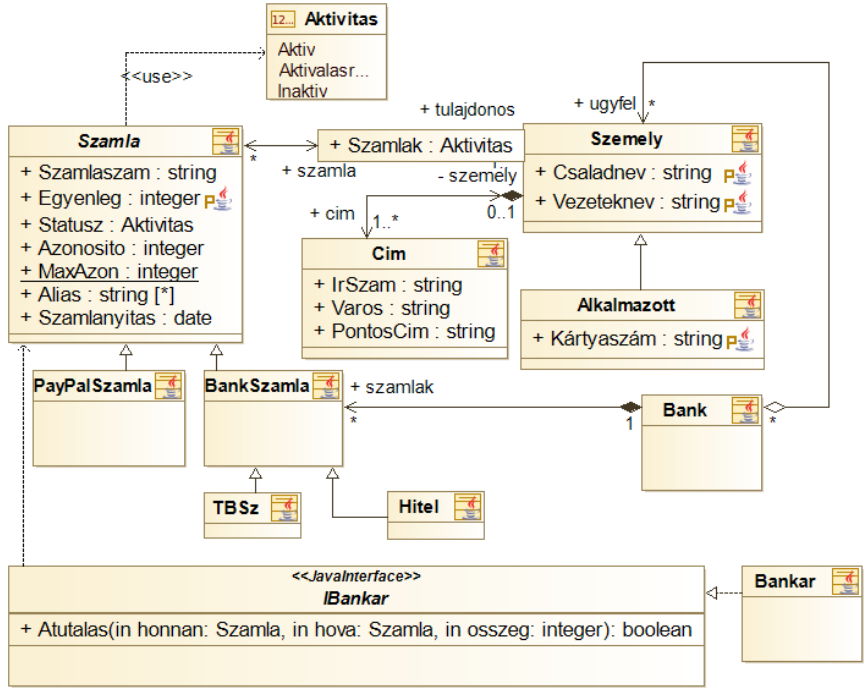
III. fejezet Hogyan?



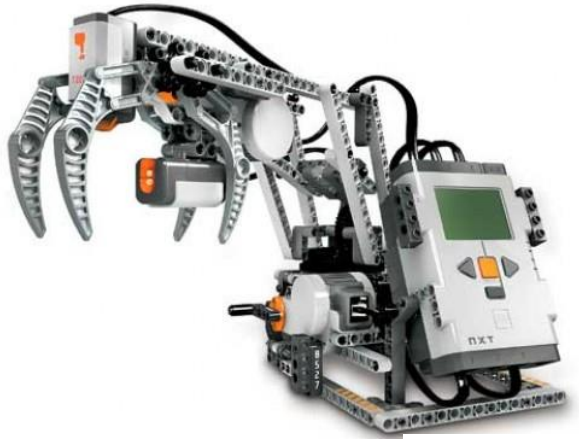
UNIVERZÁLIS, VAGY EGYEDI?

UML

- Modellezés – ahogy már ismeritek: UML
  - Szoftvermérnökök közös modellező nyelve
  - Magas absztrakciós szint
  - Szabványos jelölésrendszer
  - Gazdag eszköztámogatás
  - Nehézkes használni: limitált testreszabhatóság
  - Részleges kódgenerálás
  - Hol van itt a low code – no code?



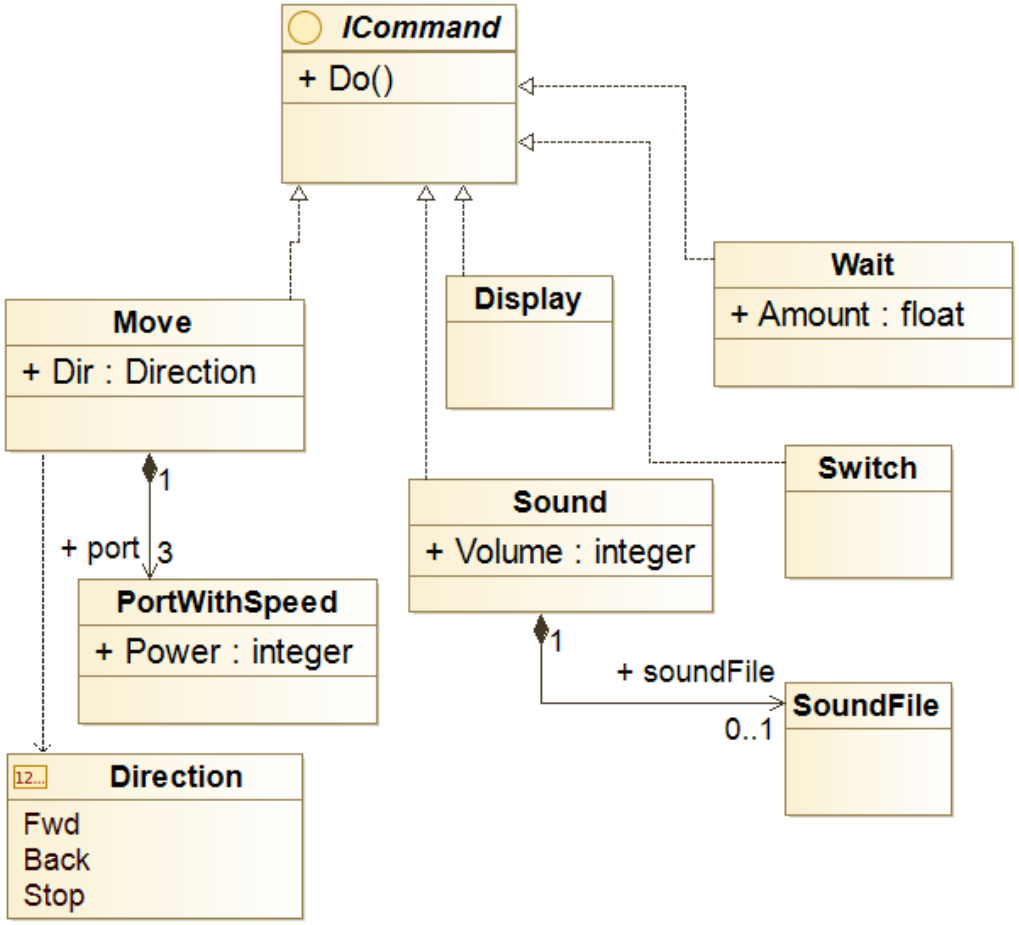
EGY PÉLDA: LEGO MINDSTORMS



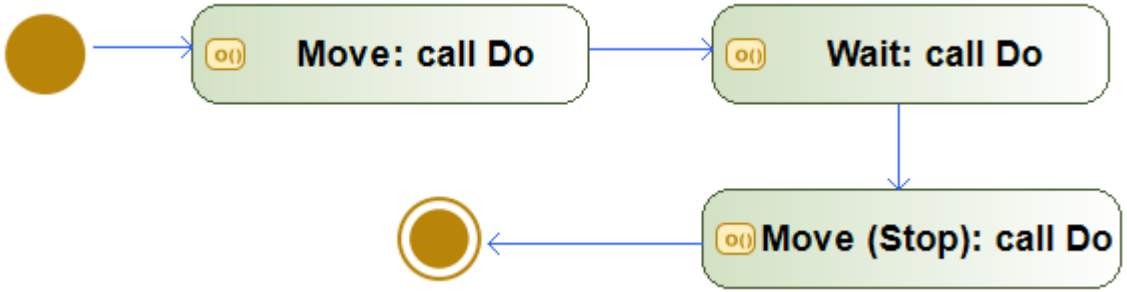
MINDSTORMS – FIZIKAI FELÉPÍTÉS

1. Szenzorok: fény, hang, érintés, ultrahang
2. 3 precíz szervomotor
3. Lego építőelemek a robotok építéséhez

MINDSTORMS – UML



- Menjen előre 1mp-ig, majd álljon meg!

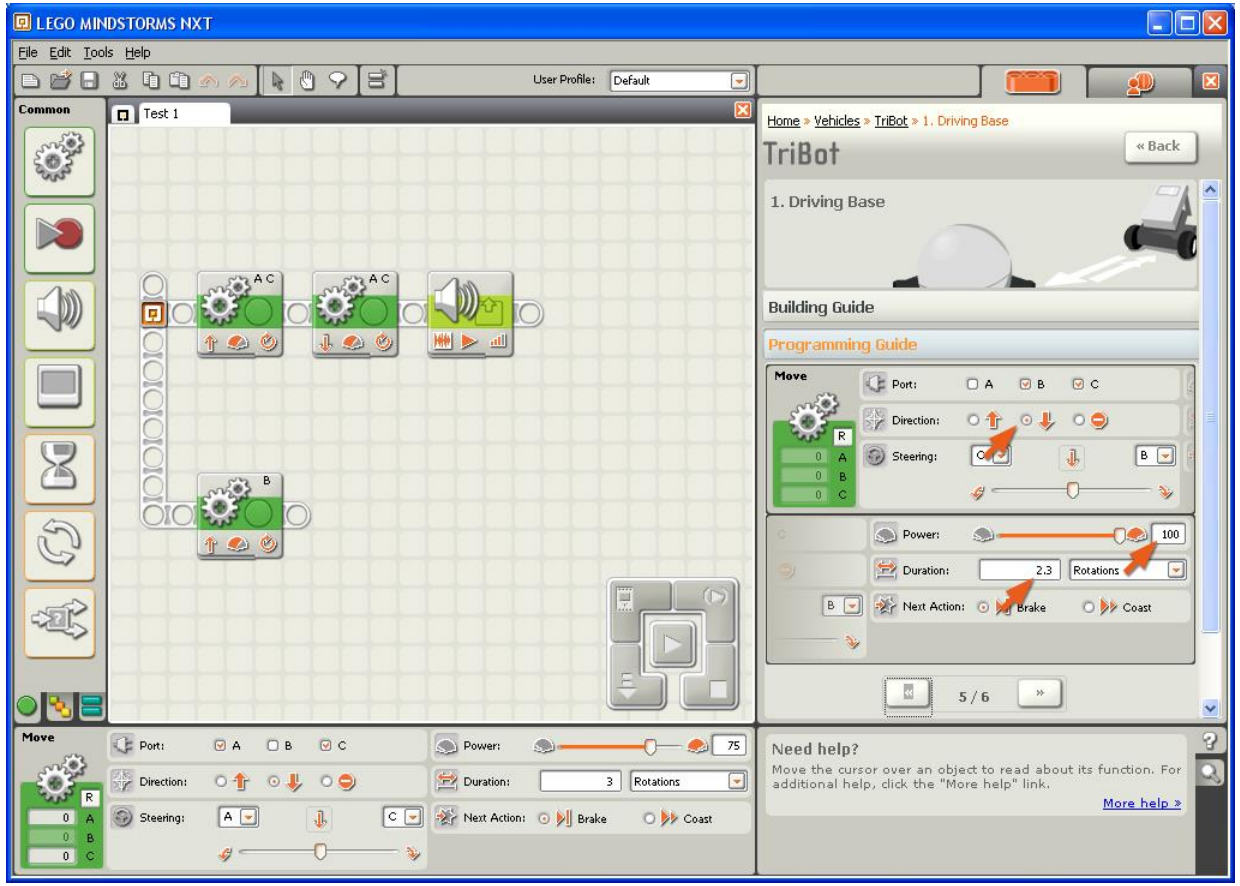


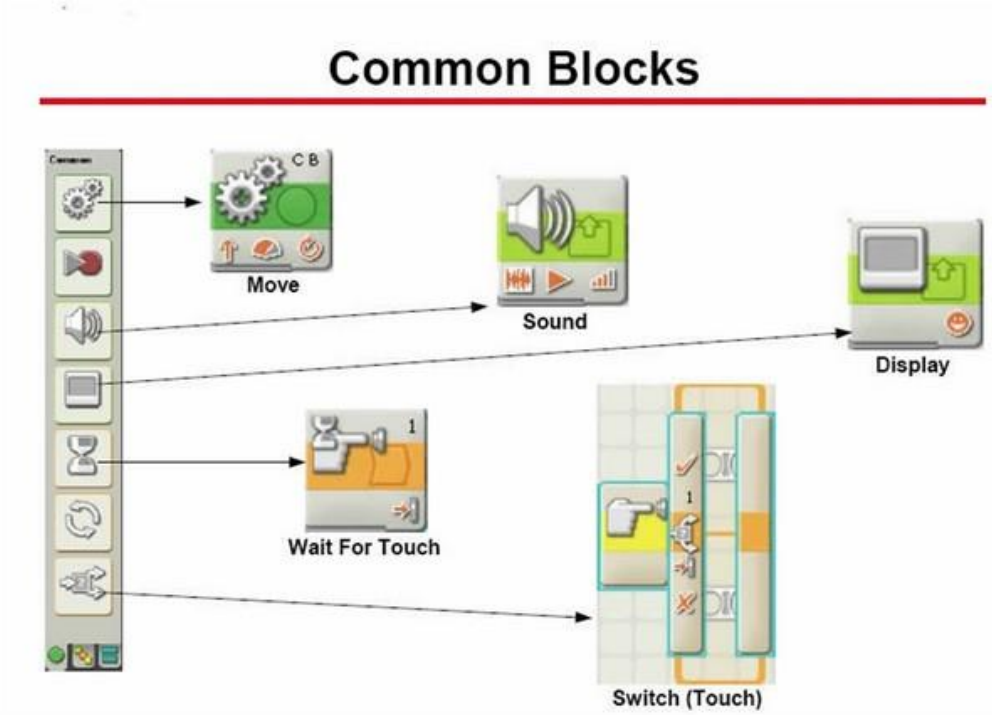
AZ UML-EN TÚL

- UML – “svájci bicska”: mindenre jó… egy kicsit

- Nekünk inkább egy ládányi célszerszám kellene

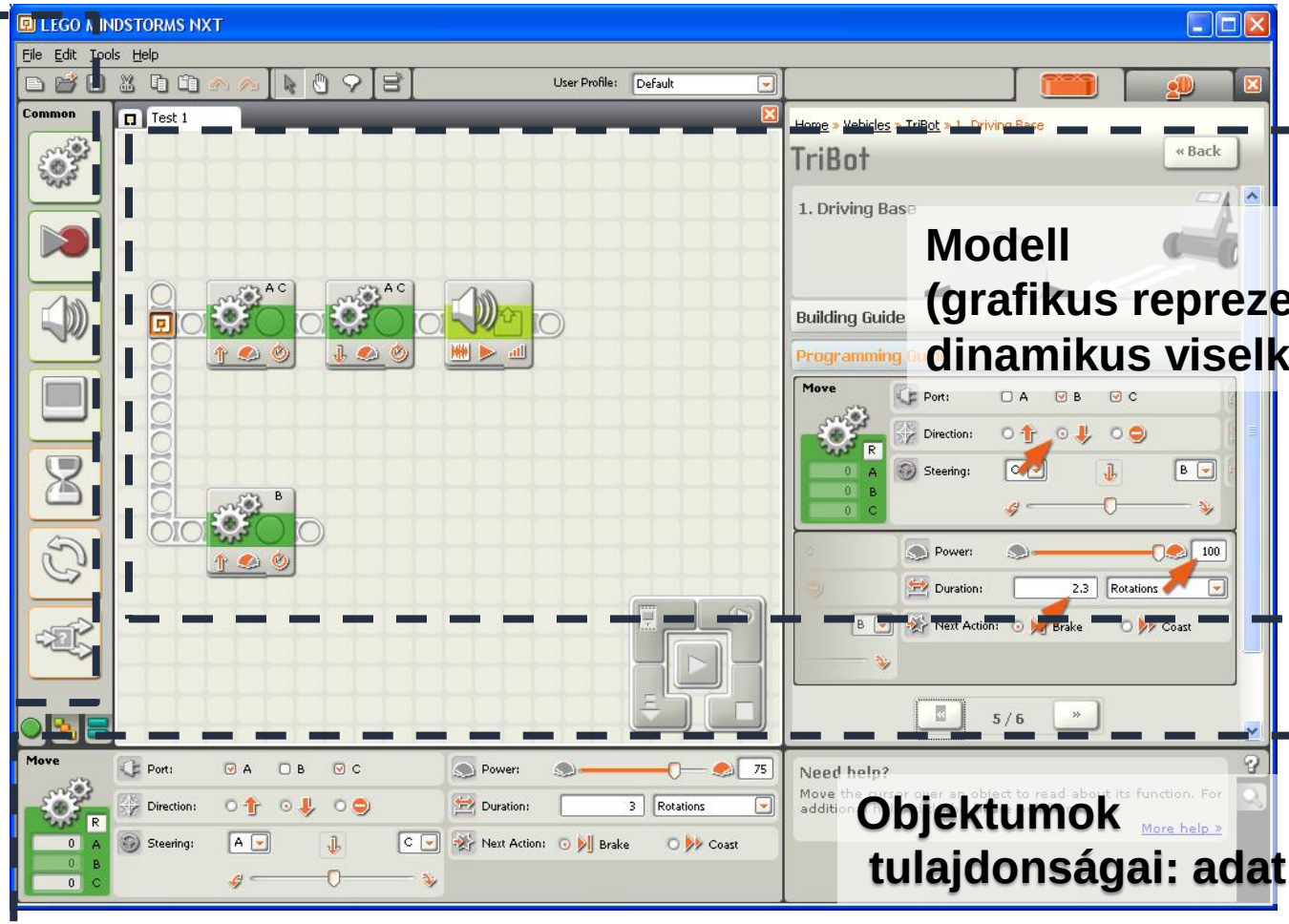
MINDSTORMS – NO CODE GRAFIKUS SZERKESZTŐ





MINDSTORMS – PROGRAMOZÁSI KÖRNYEZET

Modellezési Primitívek



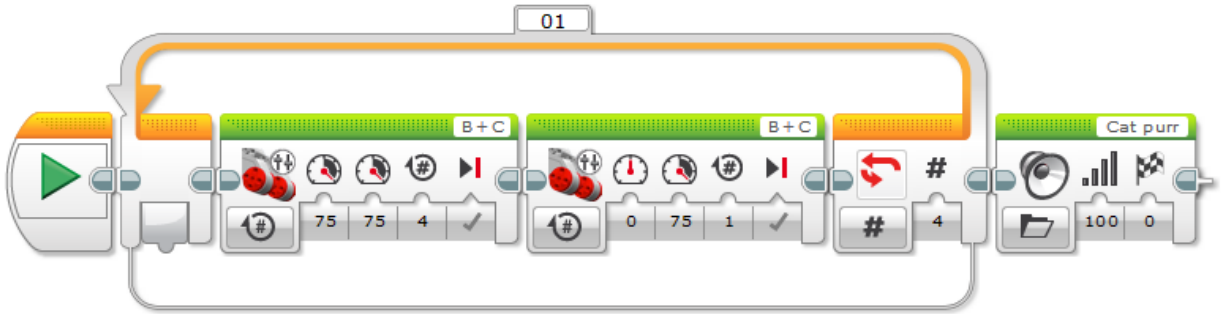
Modell (grafikus reprezentáció): dinamikus viselkedés

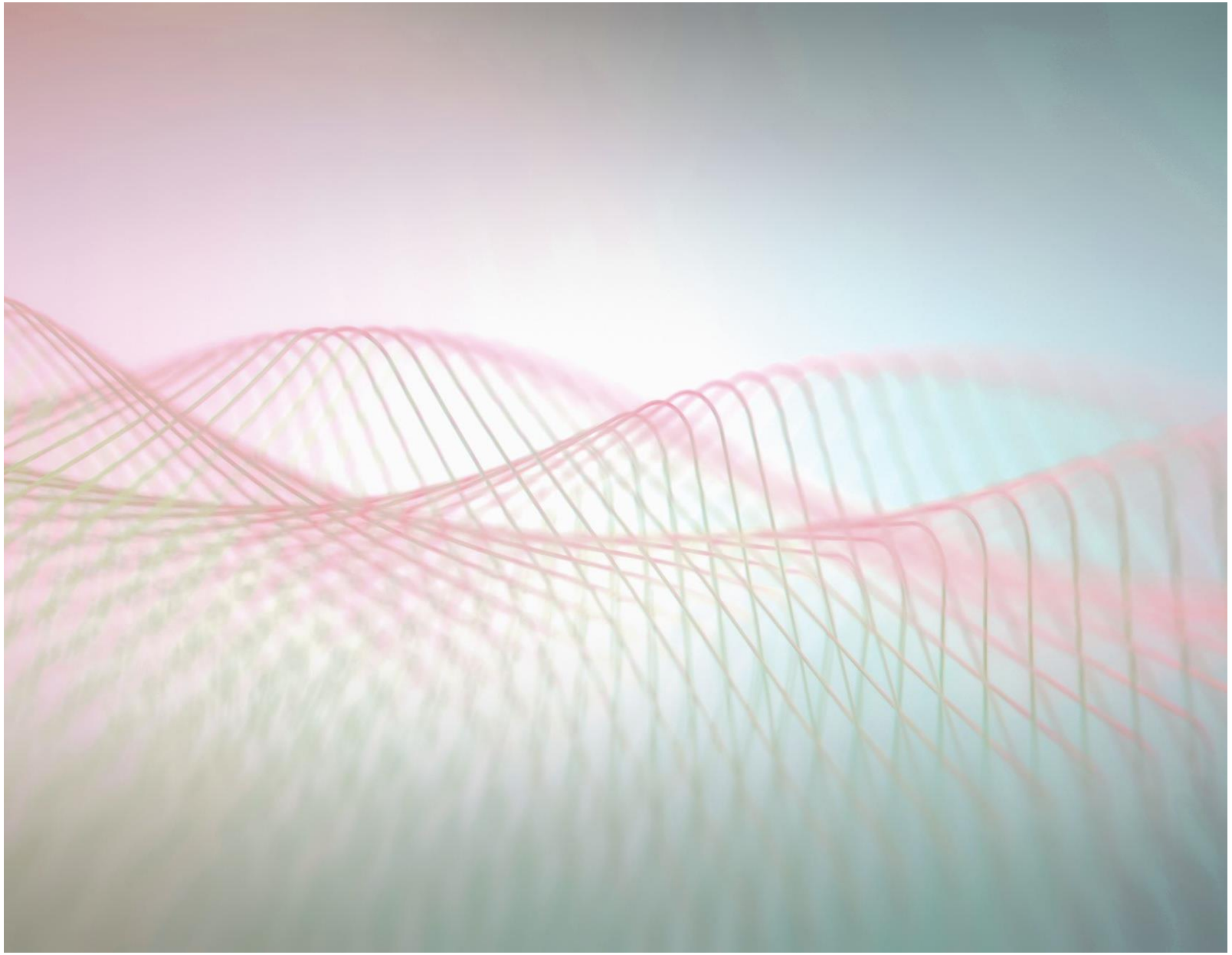
Objektumok tulajdonságai: adat

MINDSTORMS – PÉLDÁK

Program 1 – Menjen előre 1 másodpercig

Program 2 – Négyzet alakban megy, majd hangot ad ki

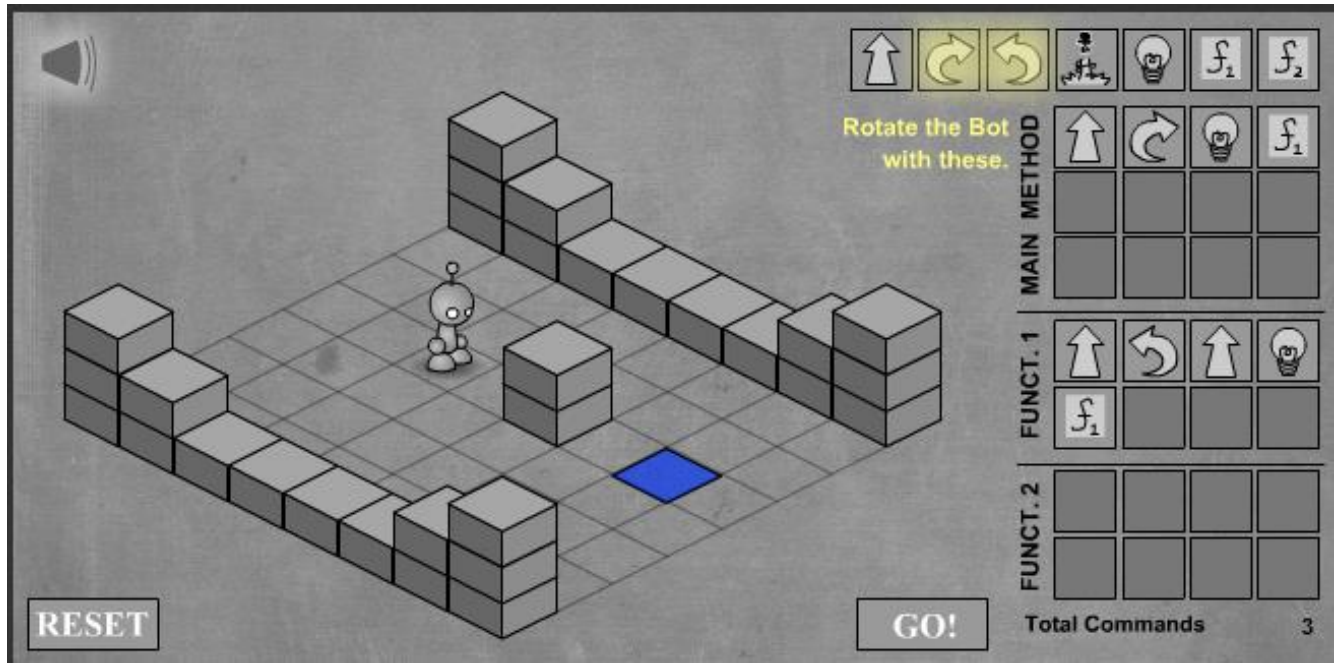




MODELLEK A NAGYVILÁGBÓL

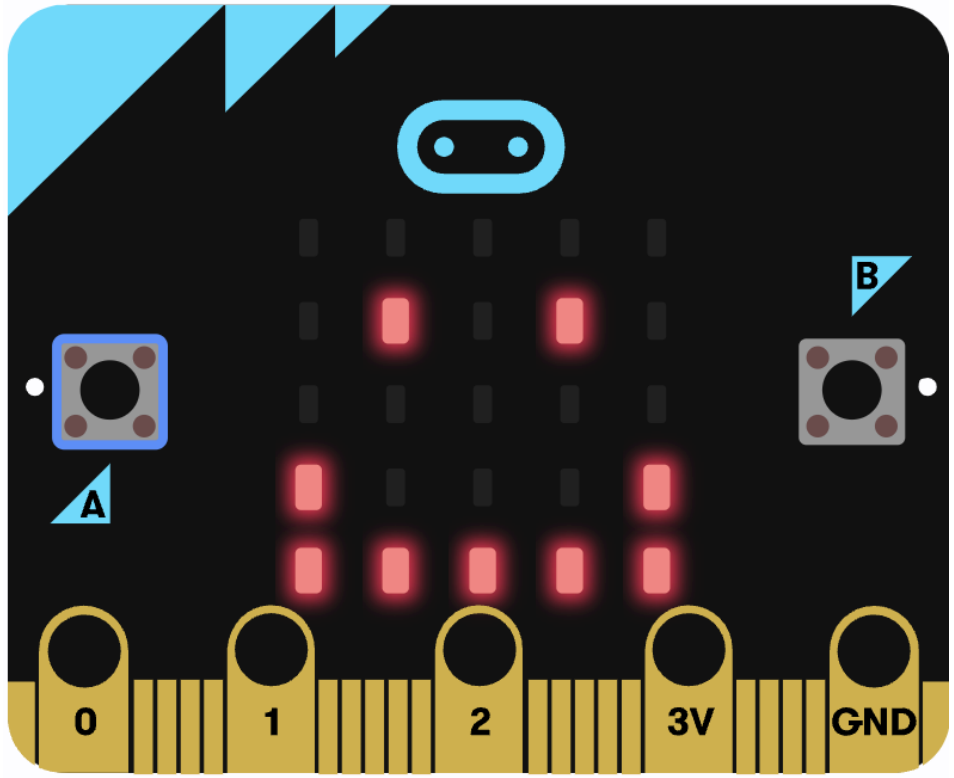
TOVÁBBI PÉLDÁK – LIGHTBOT

- Robotirányítás néhány paranccsal
  - Grafikus programozási felület
  - Grafikus "debugger"



TOVÁBBI PÉLDÁK – MICRO:BIT

- Beágyazott programozás néhány blokkal



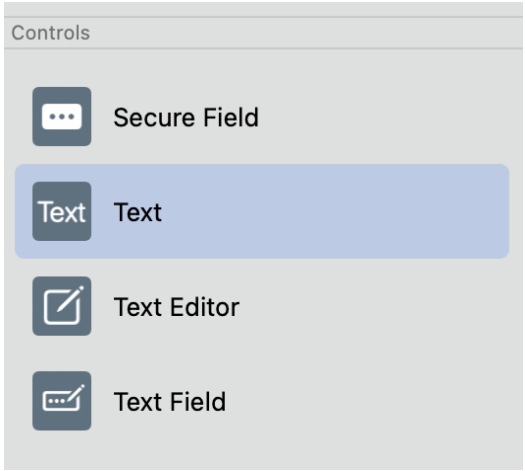
TOVÁBBI PÉLDÁK – CCG

- Gyűjtögetős kártyajáték - szabályrendszer

CARD {
Name: "Kvatch Solder";
Type: Creature;
Attack: 2;
Health: 3;
Cost: 3;
Guard: true;
}

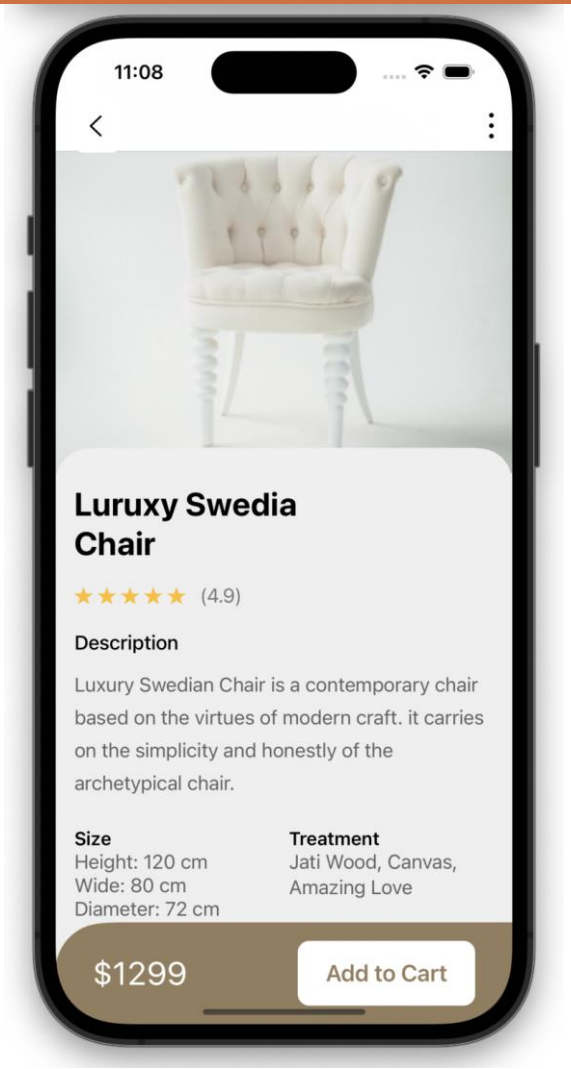
TOVÁBBI PÉLDÁK – SWIFTUI

- Low code natív alkalmazásfejlesztés iOS-re




```
HStack {
    Text("$1299")
        .font(.title)
        .foregroundColor(.white)
    Spacer()
```

Text Text



TOVÁBBI PÉLDÁK – SQL / NOSQL

SQL: általános célú relációs adatbázis definíció és lekérdezésre specializált deklaratív **nyelv**

- Adatbázistól megvalósítástól független
- Programozás helyett használható
- Absztrakciós réteg

```
SELECT Book.title,
        count(*) AS Authors
FROM Book
        JOIN Book_author ON
Book.isbn = Book_author.isbn
        GROUP BY Book.title;
```

SQL query

NoSQL: Specializált adatbázisok és lekérdezési nyelvek

- Új keresőalgoritmusok használatára új nyelvek
- A nyelvek kiemeli az algoritmusok erősségeit

```
(:Person {name: string})
-[:ACTED_IN {roles: [string]}]->
(:Movie {title: string, released: number})
```

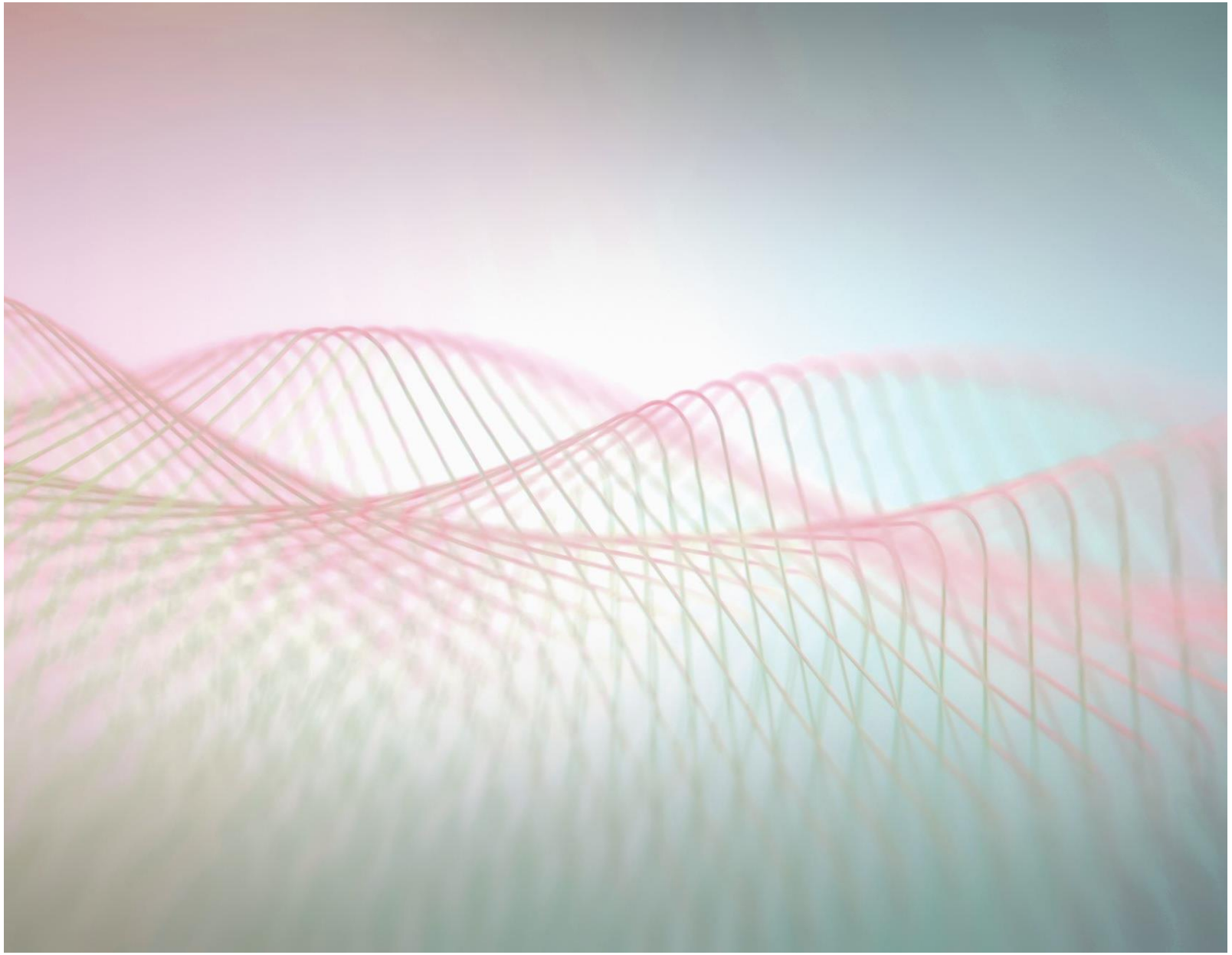
Cypher query

A Neo4J hatékony **join** algoritmust ígér, ezért ezt **nyelvi elemként** használja:

```
(x) -[ ]-> (y)
```

TOVÁBBI PÉLDÁK

- Markup nyelvek: HTML, CSS, Latex
- Programozás tanulása: Logo, Scratch
- Játék engine programozás: UnrealScript
- Hardver leírás: VHDL, Verilog
- Pénzügyi szoftverek: HR szabályrendszer, Drools
- Beágyazott rendszerek: Yakindu, AUTOSAR



A SZAKTERÜLETI NYELVEK

SZAKTERÜLETI NYELV

- Szakterületi nyelv (Domain-Specific Language, DSL)

  Egy jól definiált programozási vagy specifikációs nyelv egy szakterületre specializálva, korlátozott kifejező képességgel.

  (Martin Flower)

SZAKTERÜLETI NYELV

- Speciális nyelv egy adott szakterületre
- Korlátozott elemkészlet
- Erősen specializált szabályok és jelölés
- Egy adott termék(családhoz) készül
- Tudunk kódot generálni belőle!
- Low code – No code felfogás

ELŐNYÖK: PRODUKTIVITÁS ÉS MEGBÍZHATÓSÁG

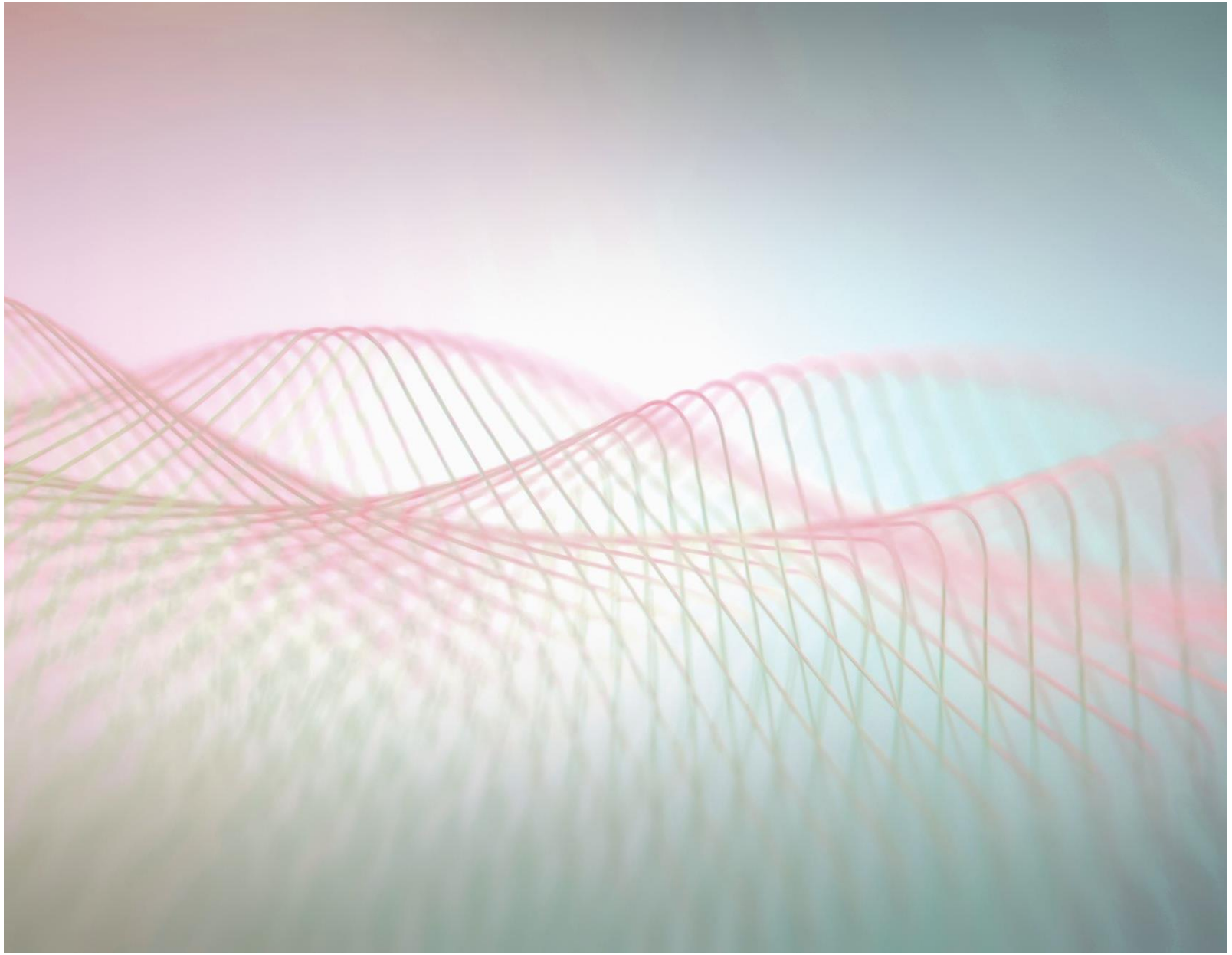
- Produktivitás és magas minőség
  - Ismerős nyelvi elemek és fogalmak a felhasználóknak
  - Kisebb változtatások fejlesztők nélkül
  - Kizárólag szakterületi szabályok mentén
  - Elrejti a lényegtelen részeket (magas absztrakciós szint)
  - Célzott matematikai analízis
- De: a kezdeti költség nagy lehet!
  - Saját nyelv és eszköz fejlesztése és karbantartása

ELŐNYÖK: TESTRESZABHATÓSÁG

- Testreszabhatóság szintjei
  - Parancssori paraméterek
  - Konfigurációs fájl (pl. XML)
  - Modolási lehetőség (pl. Skyrim)
  - Szakterületi nyelv

ELŐNYÖK: MULTIPLATFORM FEJLESZTÉS

- Több platform támogatása
  - Párhuzamos natív alkalmazások
  - Crossplatform fejlesztőeszközök
    (pl. Xamarin/.NET MAUI, Flutter, React Native, Ionic)
  - Szakterületi modellezés + kódgenerátorok
    - Egységes modell, konzisztencia
    - Platformonként egy kódgenerátor



MODELLEZÉS AZ IPARBAN

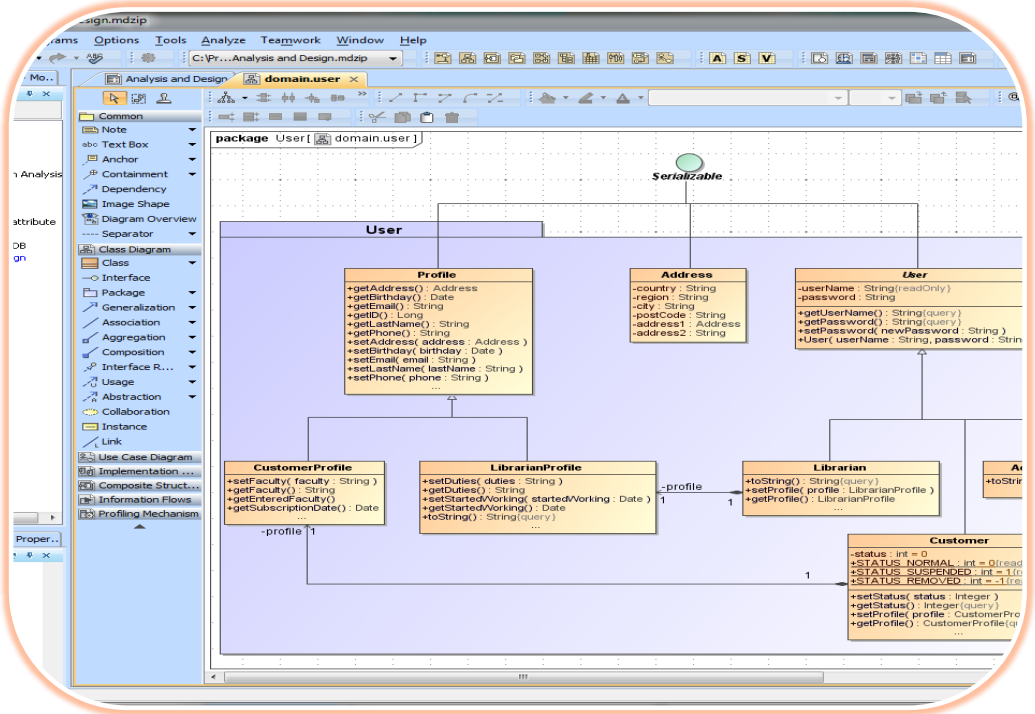
IPARI MODELLEZŐESZKÖZÖK

- Számos területen egyeduralkodók a modellezési nyelvek és modellezőeszközök
  - Fejlesztők kizárólag ezeken az eszközökön dolgoznak
  - Szabvány írja elő az eszközök használatát

DO-178C, Software Considerations in Airborne Systems and Equipment Certification. *SG4: Model Based Development and Verification*

- Az eszközökhöz illeszkedni kell!
  - A mögöttes módszerek/technológiák – ebben a tárgyban

Példák:



SysML: MagicDraw

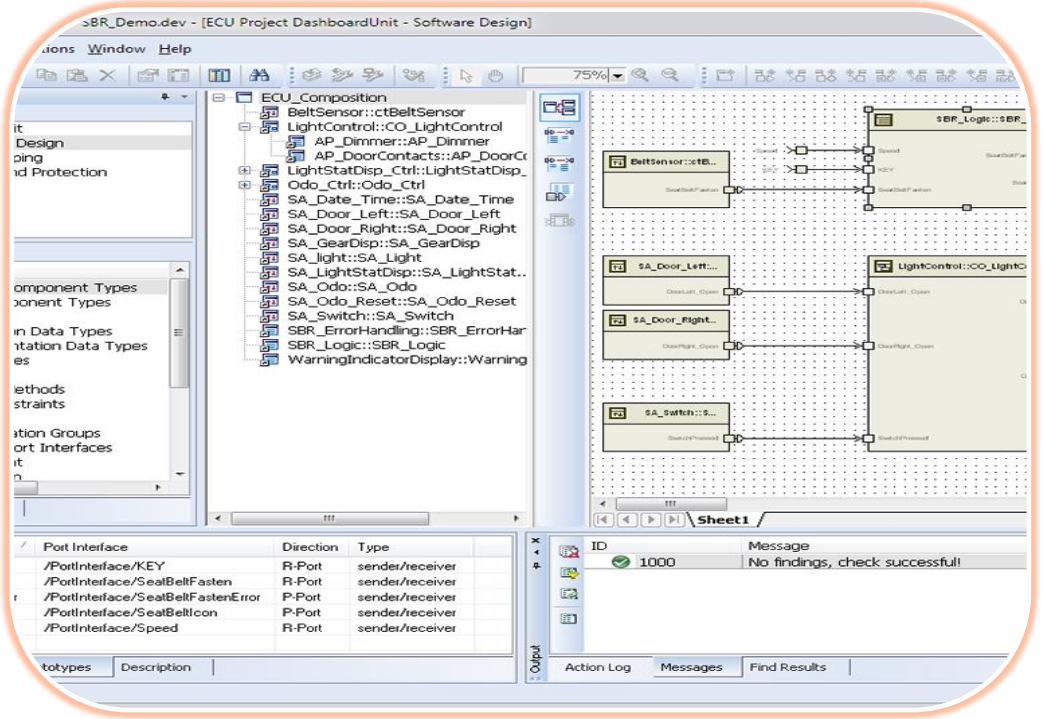
IPARI MODELLEZŐESZKÖZÖK

- Számos területen egyeduralkodók a modellezési nyelvek és modellezőeszközök
  - Fejlesztők kizárólag ezeken az eszközökön dolgoznak
  - Szabvány írja elő az eszközök használatát

DO-178C, Software Considerations in Airborne Systems and Equipment Certification. *SG4: Model Based Development and Verification*

- Az eszközökhöz illeszkedni kell!
  - A mögöttes módszerek/technológiák – ebben a tárgyban

Példák:



AUTOSAR:
Vector DaVinci Developer

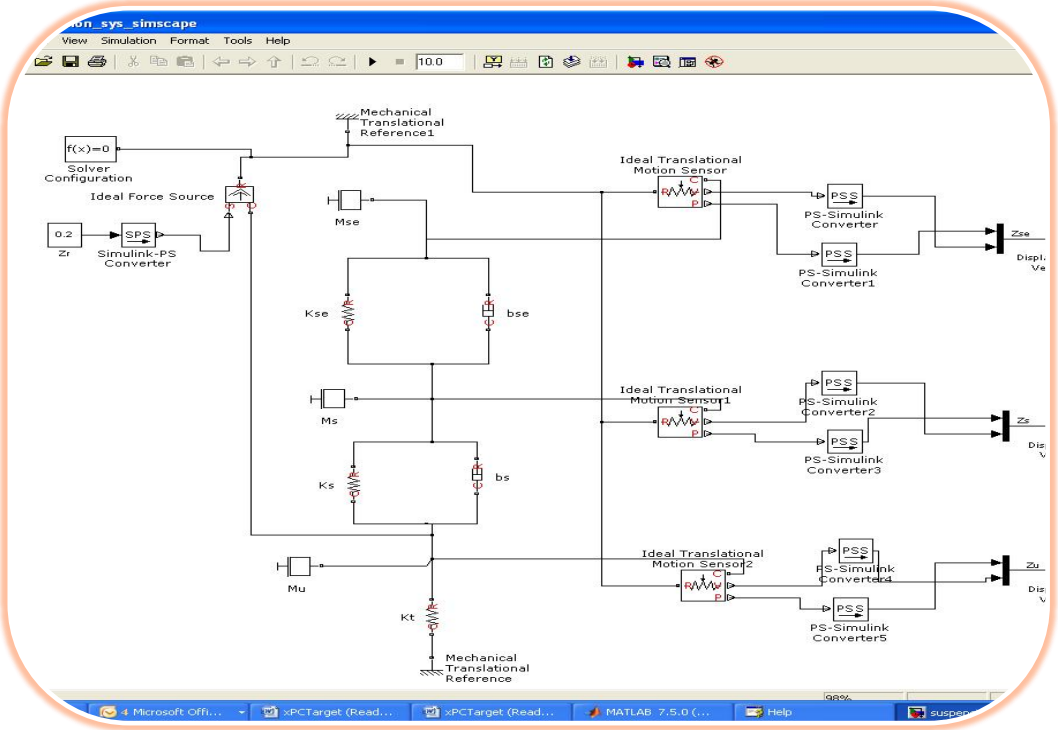
IPARI MODELLEZŐESZKÖZÖK

- Számos területen egyeduralkodók a modellezési nyelvek és modellezőeszközök
  - Fejlesztők kizárólag ezeken az eszközökön dolgoznak
  - Szabvány írja elő az eszközök használatát

DO-178C, Software Considerations in Airborne Systems and Equipment Certification. *SG4: Model Based Development and Verification*

- Az eszközökhöz illeszkedni kell!
  - A mögöttes módszerek/technológiák – ebben a tárgyban

Példák:



Matlab Simulink

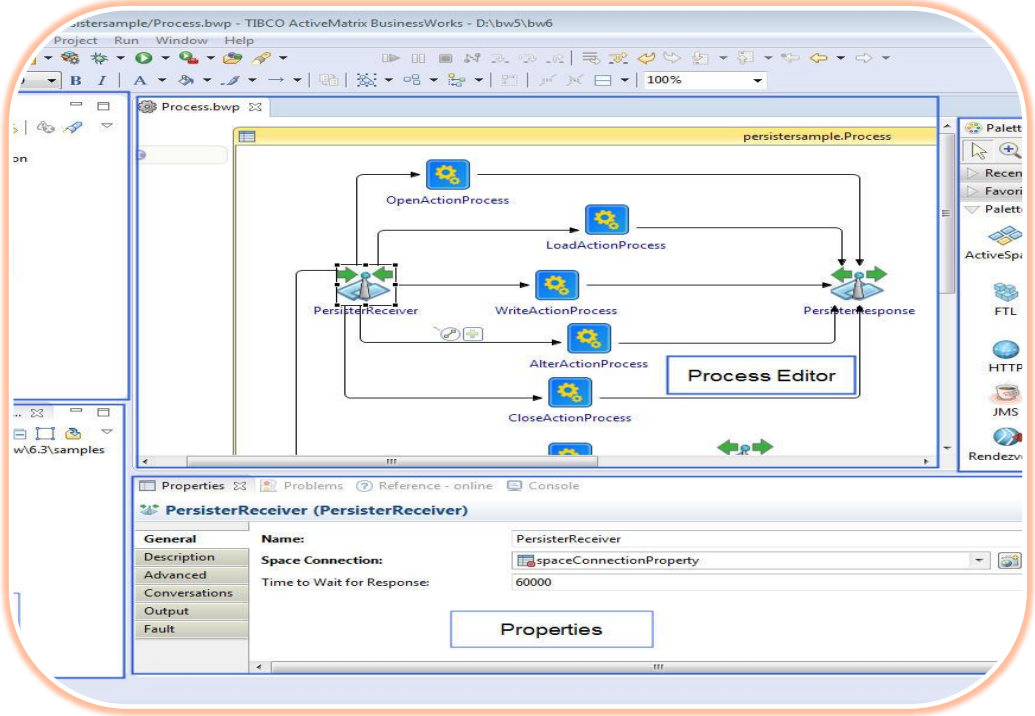
IPARI MODELLEZŐESZKÖZÖK

- Számos területen egyeduralkodók a modellezési nyelvek és modellezőeszközök
  - Fejlesztők kizárólag ezeken az eszközökön dolgoznak
  - Szabvány írja elő az eszközök használatát

DO-178C, Software Considerations in Airborne Systems and Equipment Certification. *SG4: Model Based Development and Verification*

- Az eszközökhöz illeszkedni kell!
  - A mögöttes módszerek/technológiák – ebben a tárgyban

Példák:



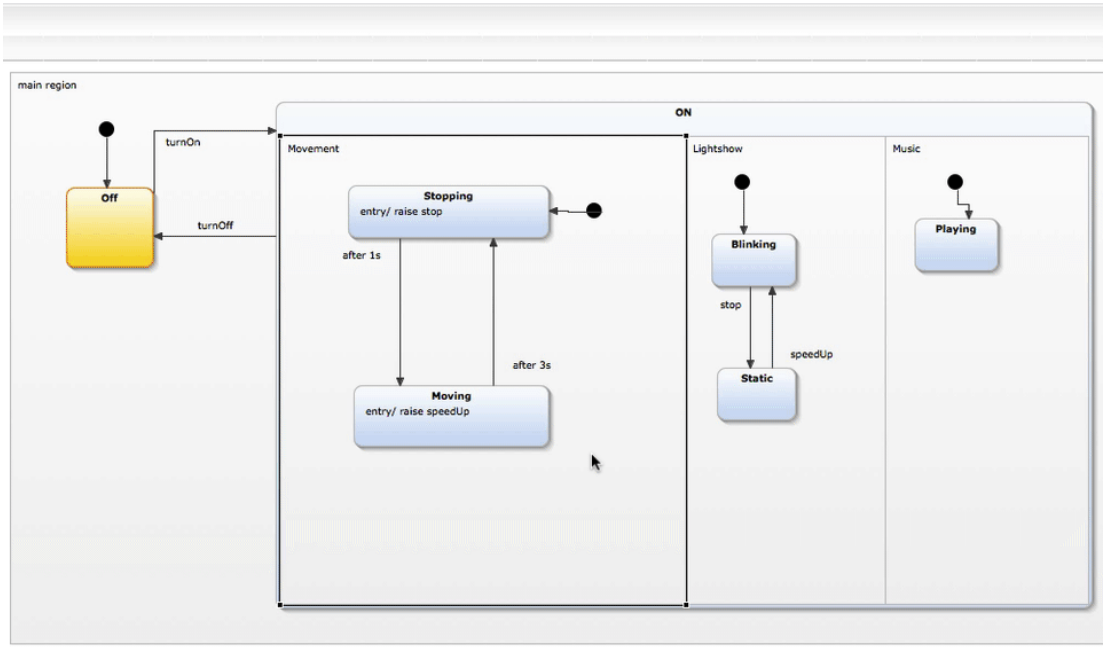
TIBCO Business Studio

SZAKTERÜLETI NYELV != MODELLEZÉS

- Egy szakterületi nyelv önmagában nem mindig elég!
  - Szerkesztő környezet
  - Debugger / szimulátor
  - Kódgenerátor
  - Kiegészítő funkciók (pl. helyesség ellenőrzés)

PÉLDA - YAKINDU: RENDSZERMODELLEZÉS HF

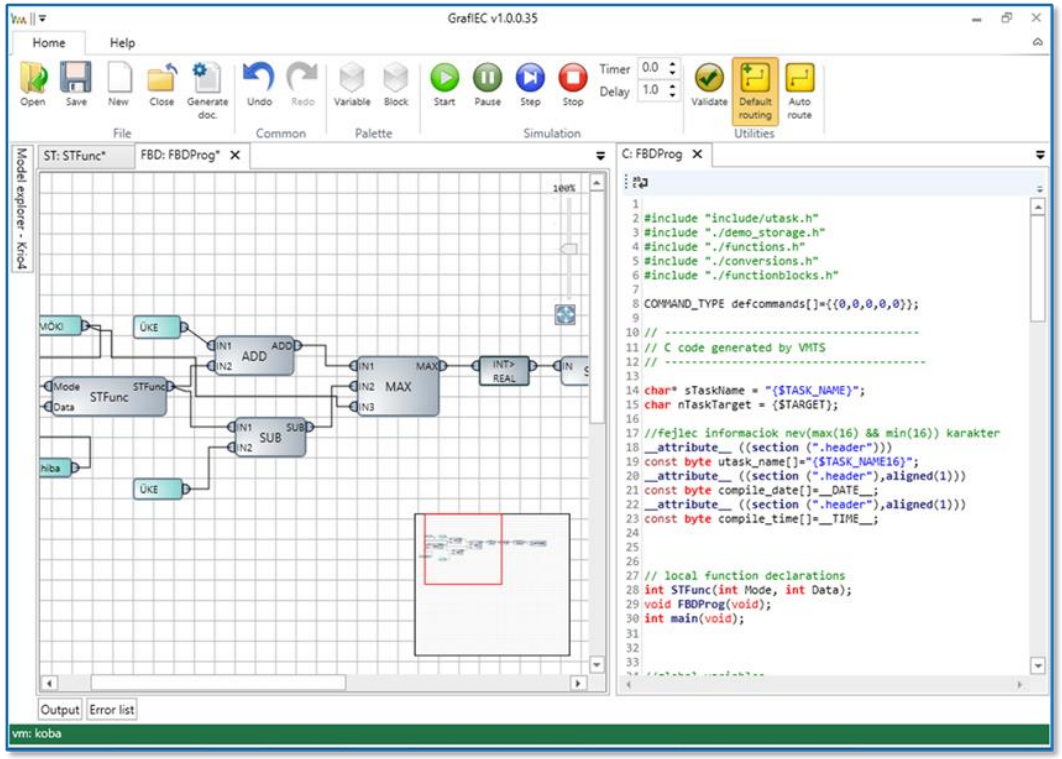
- Modellezőkörnyezet
- Szimulátor
- Kódgenerátor több nyelvre
- Matematikai helyességellenőrzés

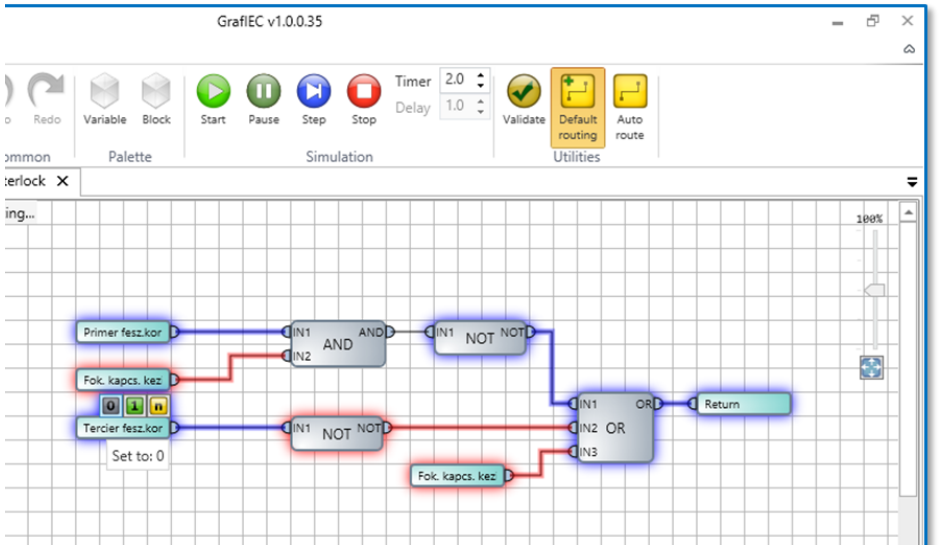


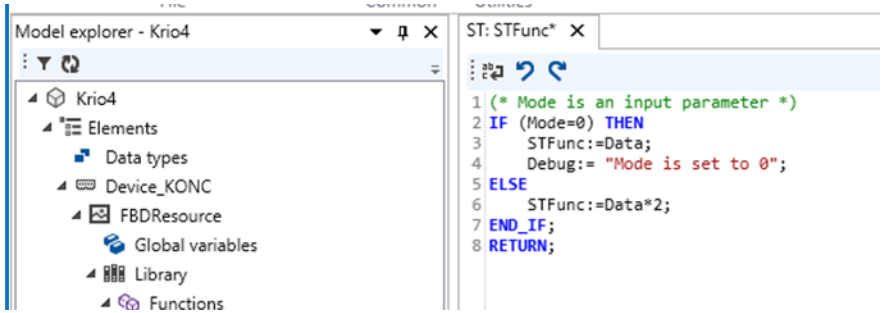
https://blogs.itemis.com/en/how-to-simulate-a-statechart-model
https://github.com/ftsrg/gamma

PÉLDA – GRAF IEC

- IEC 61131 ipari szabvány

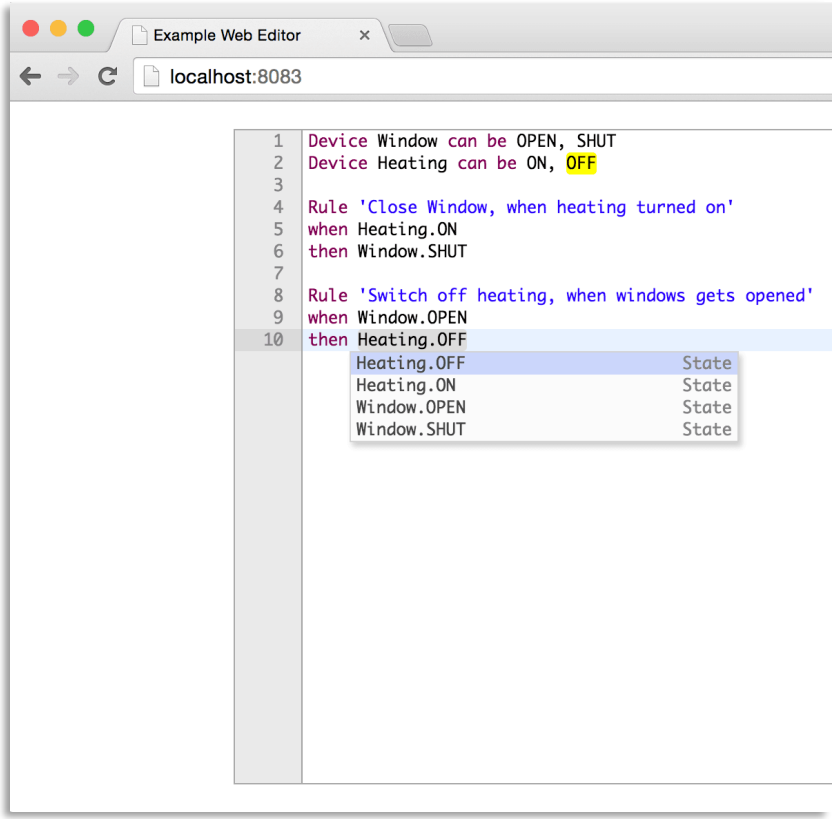


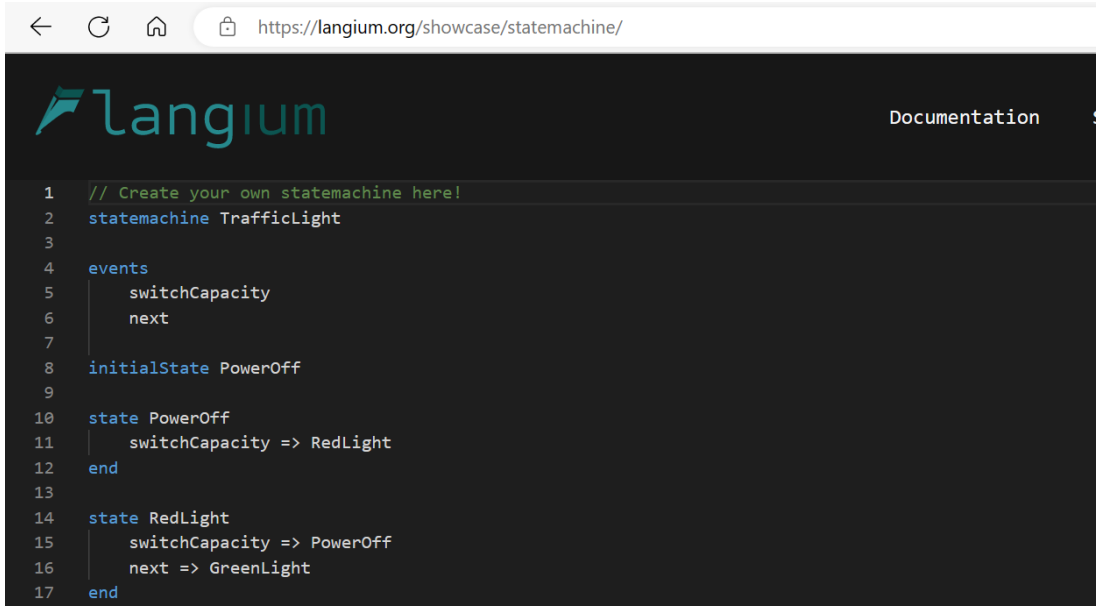




MODELLEZÉS A WEBEN

- Webes eszközök – modellezési környezet telepítés nélkül

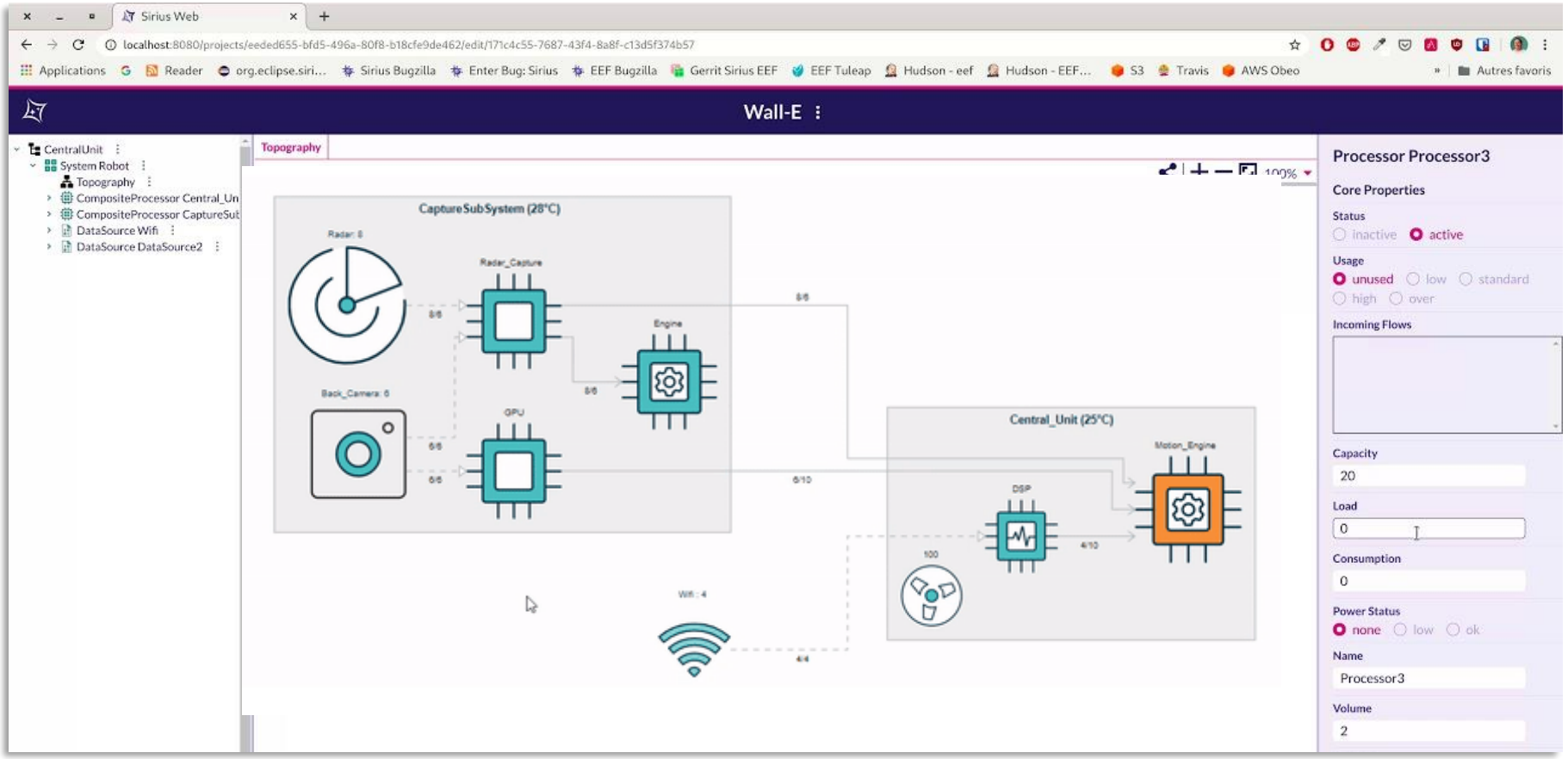




https://langium.org/

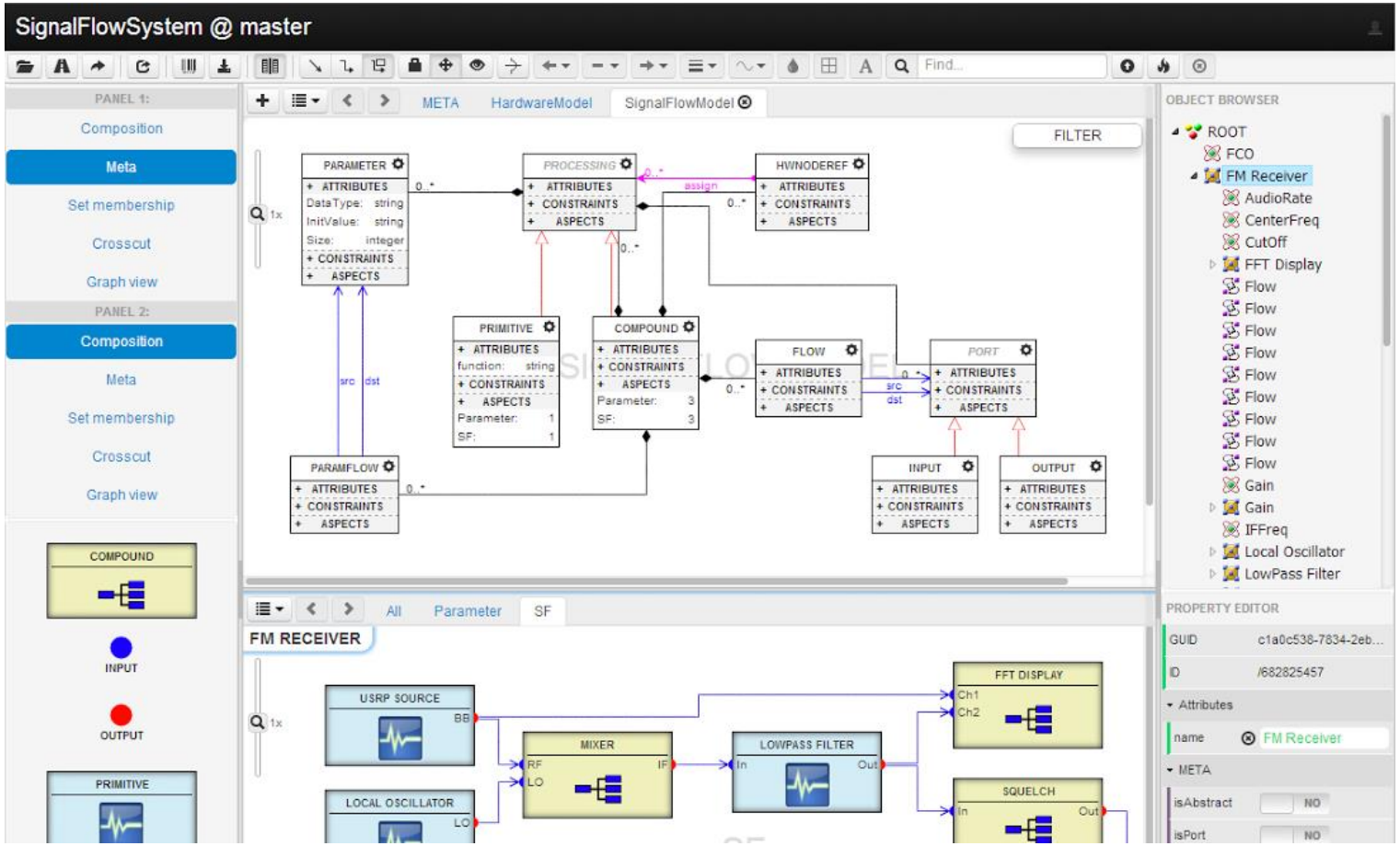
https://www.eclipse.org/Xtext/documentation/330_web_support.html

MODELLEZÉS A WEBEN



https://www.eclipse.org/sirius/sirius-web.html

MODELLEZÉS A WEBEN



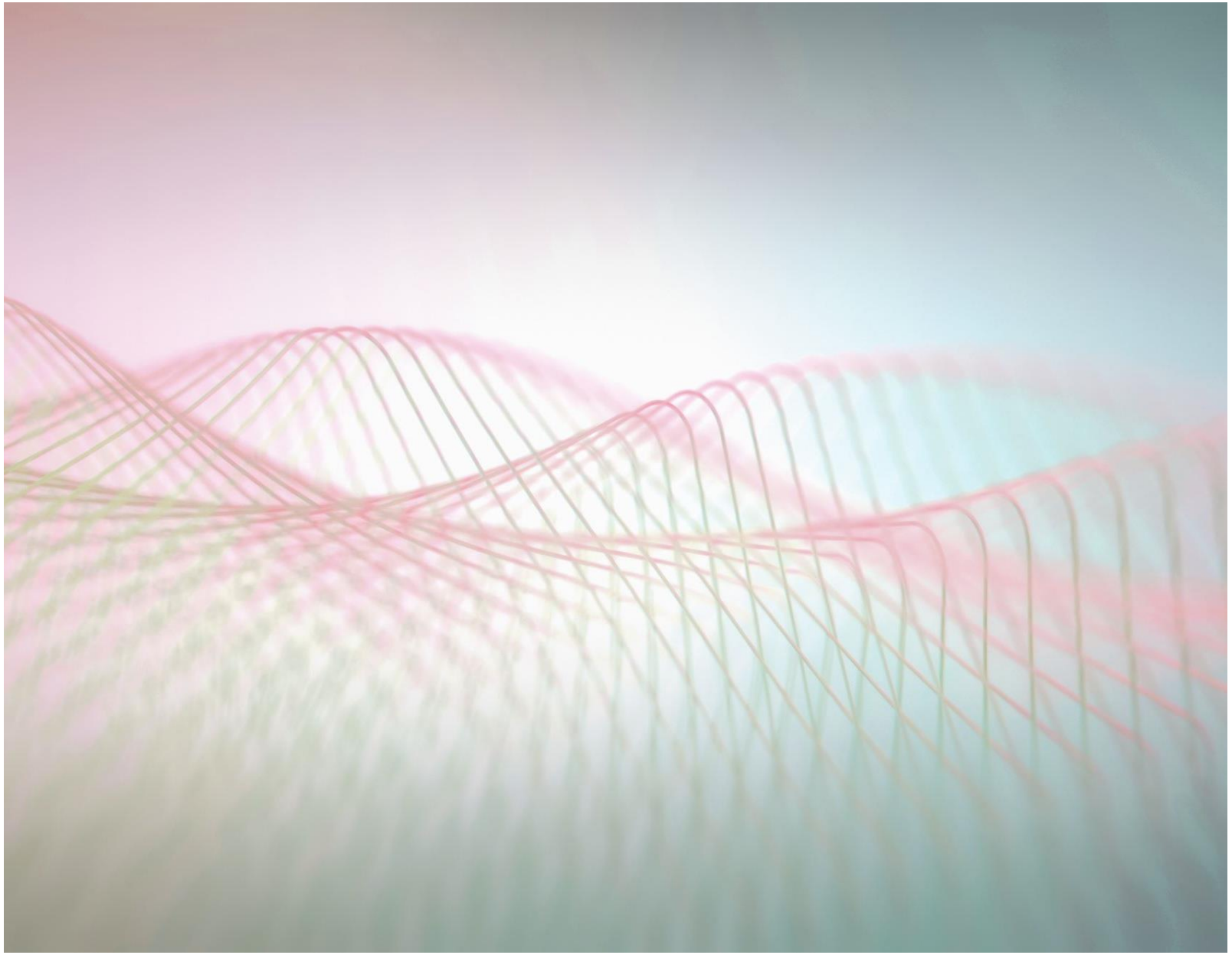
https://webgme.org/

A MAI ELŐADÁS

I. fejezet Miért?

II. fejezet Miről?

III. fejezet Hogyan?



PÁR SZÓBAN A TÁRGYRÓL

A TÁRGYRÓL

- Adatlap: https://portal.vik.bme.hu/kepzes/targyak/VIAUMA22
- Oktatás – három tanszék kooperációja (AUT – IIT – MIT)
  - AUT: Mezei Gergely, Somogyi Ferenc
  - IIT: Simon Balázs
  - MIT: Semeráth Oszkár
- Előadás – minden héten
  - Elméleti ismeretek
- Gyakorlat – minden második héten
  - Az elmélet szemléltetése a gyakorlatban
  - Demók, gyakorlati példák, esettanulmányok




SZÁMONKÉRÉSEK

- 6 db kis ZH
  - Moodle teszt (online)
  - A gyakorlatok anyagából
  - 3 sikeres kis ZH kell az aláíráshoz
- ZH (április közepe-vége)
- Vizsga
  - Írásbeli
  - A félévközi eredmények nem számítanak bele

1

Szöveges modellezés

Fordítóprogramok,
Nyelvfeldolgozás lépései.
Kódgenerálás,
Interpreterek

2

Grafikus modellezés

Szerkezet + megjelenítés,
Blockly, UML Profile,
Metamodellezés,
Szemantika

3

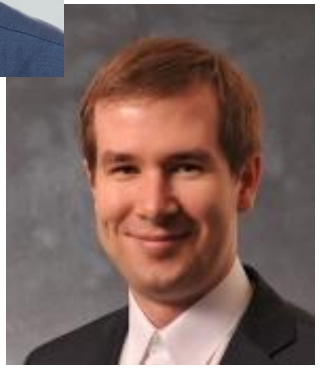
Modellfeldolgozás

Modellfeldolgozás,
Kódgenerálás,
Gráftranszformáció,
Modellalapú fejlesztés

MIRŐL LESZ SZÓ?

A KÖVETKEZŐ RÉSZ TARTALMÁBÓL…

- Gyakorlat – három történet az iparból
- **Alois Zoitl** (JKU, LIT Cyber-Physical Systems Lab):
  Modeling distributed production control systems with IEC 61499 and Eclipse 4diac – Taming the control software dragon!
- **Ráth István** (IncQuery Labs):
  Modellezéssel Budapestről a Marsig, avagy hogyan lehet egy BME spinoff a NASA JPL beszállítója
- **Simon Balázs** (BME, IIT):
  Szöveges DSL-ek az iparban – Vasúti biztosítóberendezések és elosztott szolgáltatások



KÖSZÖNÖM A FIGYELMET!

SZOFTVERFEJLESZTŐ SZEMINÁRIUM

- Qualysoft ACADEMY - Ipari technológiák és tapasztalatok testközelből
  - Szoftverfejlesztői szeminárium (nem modellezés)
  - Ipari résztvevők a tapasztalataikról
  - 2023 tavaszi félév (03.08. - 05.17. hetente)
- https://autacademy4.wixsite.com/autacademy/qualysoft