3. Analízis modell kidolgozása 1

54 – *Override*

Konzulens:
dr. László Zoltán

Csapattagok:

Kriván Bálint	CBVOEN	balint@krivan.hu
Jákli Gábor	ONZ5G1	j_gab666@hotmail.com
Dévényi Attila	L1YRH0	devenyiat@gmail.com
Apagyi Gábor	X8SG3T	apagyi.gabooo@gmail.com
Péter Tamás Pál	N5ZLEG	falconsaglevlist@gmail.com

2011. február 26.

Tartalomjegyzék

Ábrák jegyzéke

3. Analízis modell kidolgozása 1

3.1. Objektum katalógus

3.1.1. Parser

Áramkör értelmező objektum, feladata, hogy a paraméterként átadott, illetve fájlban elhelyezett komponenseket értelmezze, a kapcsolatokat feltérképezze, elvégezze az összeköttetéseket, és ezáltal felépítse az áramkört.

3.1.2. ConsoleView

Az áramkör karakteres megjelenítéséért, és a szimuláció során a változások megjelenítésének frissítéséért felelős objektum.

3.1.3. Simulation

Szimuláció objektum. A szimulációért felelős. Elindítja a jelgenerátor léptetőt, s utasítja az áramkört a kiértékelésre, és figyeli ha az áramkörben változás történt. Ha változás megadott lépésen belül nem történt, tájékoztatja a felhasználót, hogy nincs stacionárius állapot. Továbbá a megadott grafikai megjelenítőt frissíti.

3.1.4. Circuit

Az áramkör objektum. Ezen objektum feladata a jelgenerátor léptető kérésére a jelgenerátorok léptetése, az áramkörben található komponensek utasítása arra, hogy töröljék a "már kiértékelve" flaget, hogy ezáltal a következő kiértékelés kezdeményezésre továbbítsák azt bemeneteik számára is. Továbbá feladata a kiértékelés elindítása az összes kijelzőre, mert a rendszer kiértékelése a kijelzők kiértékelésével kezdődik.

3.1.5. SequenceGeneratorStepper

Jelgenerátor léptető objektum. Feladata, hogy a szimulációt utasítsa, hogy az áramkörben megtalálható jelgenerátorokat léptesse.

3.1.6. SequenceGenerator

Jelgenerátor, az áramkört felépítő egyik alapelem, kiértékelési kezdeményezés hatására az előre betáplált jelsorozatot soron következő elemét állítja be aktuális értékként, így azon komponensek melyek bemenetére a Jelgenerátor van kötve, elérik aktuális értékét. Bemenete nem komponens jellegű így nem kezel más komponenseket.

3.1.7. AndGate

ÉS kapu, az áramkör egyik alapeleme. Bemeneteire kötött komponensek kiértékelését kezdeményezi, s a kapott értékek logikai ÉS kapcsolatát valósítja meg, ezáltal a kimenetére kötött komponens eléri az aktuális értékét.Figyeli hogy ha már kiértékelődött akkor nem kezdeményezi a bemenetére kötött komponensek kiértékelését.

3.1.8. OrGate

VAGY kapu, az áramkör egyik alapeleme. Bemeneteire kötött komponensek kiértékelését kezdeményezi, s a kapott értékek logikai VAGY kapcsolatát valósítja meg, ezáltal a kimenetére kötött komponens eléri az aktuális értékét.Figyeli hogy ha már kiértékelődött akkor nem kezdeményezi a bemenetére kötött komponensek kiértékelését.

3.1.9. Inverter

Invertáló, az áramkör alapelemei közé tartozik. A bemenetére érkező jel logikai negáltját valósítja meg, így a kimenetén levő komponens eléri aktuális értékét.

3.1.10. Gnd

Föld, az áramkört felépítő egyik elem, aktuális értéke minden kiértékelési kérésre logikai hamis. Bemenete nem létezik, így nem kezdeményez további kiértékeléseket. Állandó értéke logikai hamis.

3.1.11. Vcc

Áramkör alapeleme, mely kiértékelési kezdeményezésre aktuális értékét logikai igaz ra állítja be. Állandó értéke logikai igaz.

3.1.12. Led

Egy kijelző az áramkör alapeleme, bemenetére kötött komponens kiértékelését kezdeményezi, és ezáltal az aktuális értékét egy a felhasználó számára érzékelhető módon kijelzi.

3.1.13. Toggle

Kapcsoló, az áramkört felépítő elem, felhasználói interakciót követően, az aktuális értékét lehet állítani. Komponens bemenete nincs, így nem kezel további komponenseket.

3.2. Osztályok leírása

[Az előző alfejezetben tárgyalt objektumok felelősségének formalizálása attribútumokká, metódusokká. Csak publikus metódusok szerepelhetnek. Ebben az alfejezetben megjelennek az interfészek, az öröklés, az absztrakt osztályok. Segédosztályokra még mindig nincs szükség. Az osztályok ABC sorrendben kövessék egymást. Interfészek esetén az Interfészek, Attribútumok pontok kimaradnak.]

3.2.1. Circuit

- Felelősség
  Áramkört reprezentál, melyhez komponeseket lehet adni, és kiértékelési ciklusokat lehet futtatni, utóbbi a `Simulation` feladata.
- Ősosztályok Object → Circuit.
- Interfészek (nincs)
- Attribútumok
  - `private HashMap componentMap`
  - `private Simulation simulation`
  - `private boolean stable`
- Metódusok
  - `public Component addComponent(Component component)`: Komponens hozzáadása az áramkörhöz.
  - `public void doEvaluationCycle()`: Egy kiértékelési ciklus lefuttatása. Az áramkörtől ezután lekérdezhető, hogy stabil (nem változott semelyik komponens kimenete az utolsó futtatás óta) vagy instabil állapotban van-e.

  - `public Component getComponentByName(String name)`: Lekérünk egy komponenst az áramkörtől a neve alapján.
  - `public List getDisplays()`: Megjelenítő típusú komponeseket adja vissza.
  - `public List getSources()`: Jelforrás típusú komponenseket adja vissza.
  - `public boolean isStable()`: Áramkör stacionárius állapotának lekérdezése.
  - `public void setSimulation(Simulation simulation)`: Szimuláció beállítása.
  - `public void setStable(boolean stable)`: Áramkör stabilitásának beállítása.
  - `public void simulationRefreshRequired()`: Jelzi a szimuláció felé, hogy új ciklust kell indítani. Ezt egy jelforrás beállítása után hívjuk meg.
  - `public void stepGenerators()`: Jelgenerátorok a szimuláció szemszögéből nézve, egyszerre történő léptetése.

3.2.2. LogSim

- Felelősség

- Ősosztályok Object → LogSim.
- Interfészek (nincs)
- Attribútumok
  - (nincs)
- Metódusok
  - `public static void main(String[] args):`

3.2.3. SequenceGeneratorStepper

- Felelősség

- Ősosztályok Object → Thread → SequenceGeneratorStepper.
- Interfészek (nincs)
- Attribútumok
  - `private long pause`
  - `private boolean shouldRun`
  - `private Simulation simulation`
- Metódusok
  - `public void run():`

3.2.4. Value

- Felelősség
  Az áramkörben előfordulható érték
- Ősosztályok Object → Enum → Value.
- Interfészek (nincs)
- Attribútumok
  - `public static final Value FALSE`
  - `public static final Value TRUE`
- Metódusok
  - `public Value invert():`
  - `public static Value valueOf(String name):`
  - `public static Value[] values():`

3.2.5. Component

Absztrakt osztály.

- Felelősség
- Ősosztályok Object → Component.
- Interfészek (nincs)
- Attribútumok
  - `protected boolean alreadyEvaluated`
  - `protected Circuit circuit`
  - `protected Value[] currentValue`
  - `protected int[] indices`
  - `protected Component[] inputs`
  - `protected Value[] lastValue`
  - `protected String name`
- Metódusok
  - `public void clearEvaluatedFlag():`
  - `public Value evaluate():`
  - `public Value evaluate(int outputPin)`: Számolás:
  - `public String getName():`
  - `public Value getValue():`
  - `public Value getValue(int idx):`
  - `public void setCircuit(Circuit parent):`
  - `public void setInput(int inputSlot, Component component):`

  - `public void setInput(int inputPin, Component component, int outputPin)`: Beállítunk egy bemenetet.
  - `public void setInputPinsCount(int inputPinsCount)`:
  - `public void setName(String name)`:

3.2.6. IsDisplay

Interfész.

- Felelősség
- Ősosztályok IsDisplay.
- Interfészek (nincs)
- Metódusok
  - (nincs)

3.2.7. IsSource

Interfész.

- Felelősség
- Ősosztályok IsSource.
- Interfészek (nincs)
- Metódusok
  - `public void setValues(Value[] values)`: Beállítjuk a jelforrás értékét.

3.2.8. AndGate

- Felelősség
- Ősosztályok Object $\rightarrow$ Component $\rightarrow$ AndGate.
- Interfészek (nincs)
- Attribútumok
  - (nincs)
- Metódusok
  - (nincs)

3.2.9. Gnd

- Felelősség

- Ősosztályok Object → Component → Gnd.
- Interfészek (nincs)
- Attribútumok
  - (nincs)
- Metódusok
  - (nincs)

3.2.10. Inverter

- Felelősség

- Ősosztályok Object → Component → Inverter.
- Interfészek (nincs)
- Attribútumok
  - (nincs)
- Metódusok
  - (nincs)

3.2.11. Led

- Felelősség

- Ősosztályok Object → Component → Led.
- Interfészek IsDisplay.
- Attribútumok
  - (nincs)
- Metódusok
  - (nincs)

3.2.12. OrGate

- Felelősség

- Ősosztályok Object → Component → OrGate.
- Interfészek (nincs)
- Attribútumok

  - (nincs)

- Metódusok
  - (nincs)

3.2.13. SequenceGenerator

- Felelősség

- Ősosztályok Object → Component → SequenceGenerator.
- Interfészek IsSource.
- Attribútumok
  - `private int idx`
  - `private Value[] sequence`
- Metódusok
  - `public void setValues(Value[] values):`
  - `public void step():`

3.2.14. Toggle

- Felelősség

- Ősosztályok Object → Component → Toggle.
- Interfészek IsSource.
- Attribútumok
  - (nincs)
- Metódusok
  - `public void setValues(Value[] values):`
  - `public void toggle()`: Kapcsoló állapotát megváltoztatjuk

3.2.15. Vcc

- Felelősség

- Ősosztályok Object → Component → Vcc.
- Interfészek (nincs)
- Attribútumok
  - (nincs)
- Metódusok
  - (nincs)

3.2.16. Controller

Interfész.

- Felelősség
- Ősosztályok Controller.
- Interfészek (nincs)
- Metódusok
  - `public void start():`
  - `public void stop():`

3.2.17. Simulation

- Felelősség
- Ősosztályok Object → Thread → Simulation.
- Interfészek Controller.
- Attribútumok
  - `private Circuit circuit`
  - `private AtomicInteger counter`
  - `private static final int cycleLimit`
  - `private final Object lock`
  - `private final SequenceGeneratorStepper seqGenStepper`
  - `private boolean shouldRun`
  - `private final Object synchObj`
  - `private final View view`
- Metódusok
  - `public Circuit getCircuit():`
  - `public Object getLock():`
  - `public void run():`
  - `public void setCircuit(Circuit circuit):`
  - `public void sourcesChanged():` Megváltozott valamelyik jelforrás, szimuláció mehet újból

3.2.18. Parser

- Felelősség

- Ősosztályok Object → Parser.
- Interfészek (nincs)
- Attribútumok
  - `private static final HashMap availableComponents`
  - `private Circuit circuit`
  - `private static Pattern componentPattern`
  - `private int constComps`
  - `private static Pattern inputPattern`
  - `private HashMap inputs`
- Metódusok
  - `public Circuit parse(File file)`: Létrehoz egy áramkört a megadott fájlból
  - `public Circuit parse(String[] content)`: Létrehoz egy áramkört az argumentumokban megadott komponensekből

3.2.19. Osztály1

- Felelősség
  [Mi az osztály felelőssége. Kb 1 bekezdés.]
- Ősosztályok
  [Mely osztályokból származik (öröklési hierarchia)
  Legősebb osztály → Ősosztály2 → Ősosztály3...]
- Interfészek
  [Mely interfészeket valósítja meg.]
- Attribútumok
  [Milyen attribútumai vannak]
  - attribútum1: attribútum jellemzése: mire való
  - attribútum2: attribútum jellemzése: mire való
- Metódusok
  [Milyen publikus metódusokkal rendelkezik. Metódusonként egy-három mondat arról, hogy a metódus mit csinál.]
  - int foo(Osztály3 o1, Osztály4 o2): metódus leírása
  - int bar(Osztály5 o1): metódus leírása

3.2.20. Osztály2

- Felelősség
  [Mi az osztály felelőssége. Kb 1 bekezdés.]
- Ősosztályok
  [Mely osztályokból származik (öröklési hierarchia)
  Legősebb osztály → Ősosztály2 → Ősosztály3...]
- Interfészek
  [Mely interfészeket valósítja meg.]
- Attribútumok
  [Milyen attribútumai vannak]
  - attribútum1: attribútum jellemzése: mire való
  - attribútum2: attribútum jellemzése: mire való
- Metódusok
  [Milyen publikus metódusokkal rendelkezik. Metódusonként egy-három mondat arról, hogy a metódus mit csinál.]
  - int foo(Osztály3 o1, Osztály4 o2): metódus leírása
  - int bar(Osztály5 o1): metódus leírása

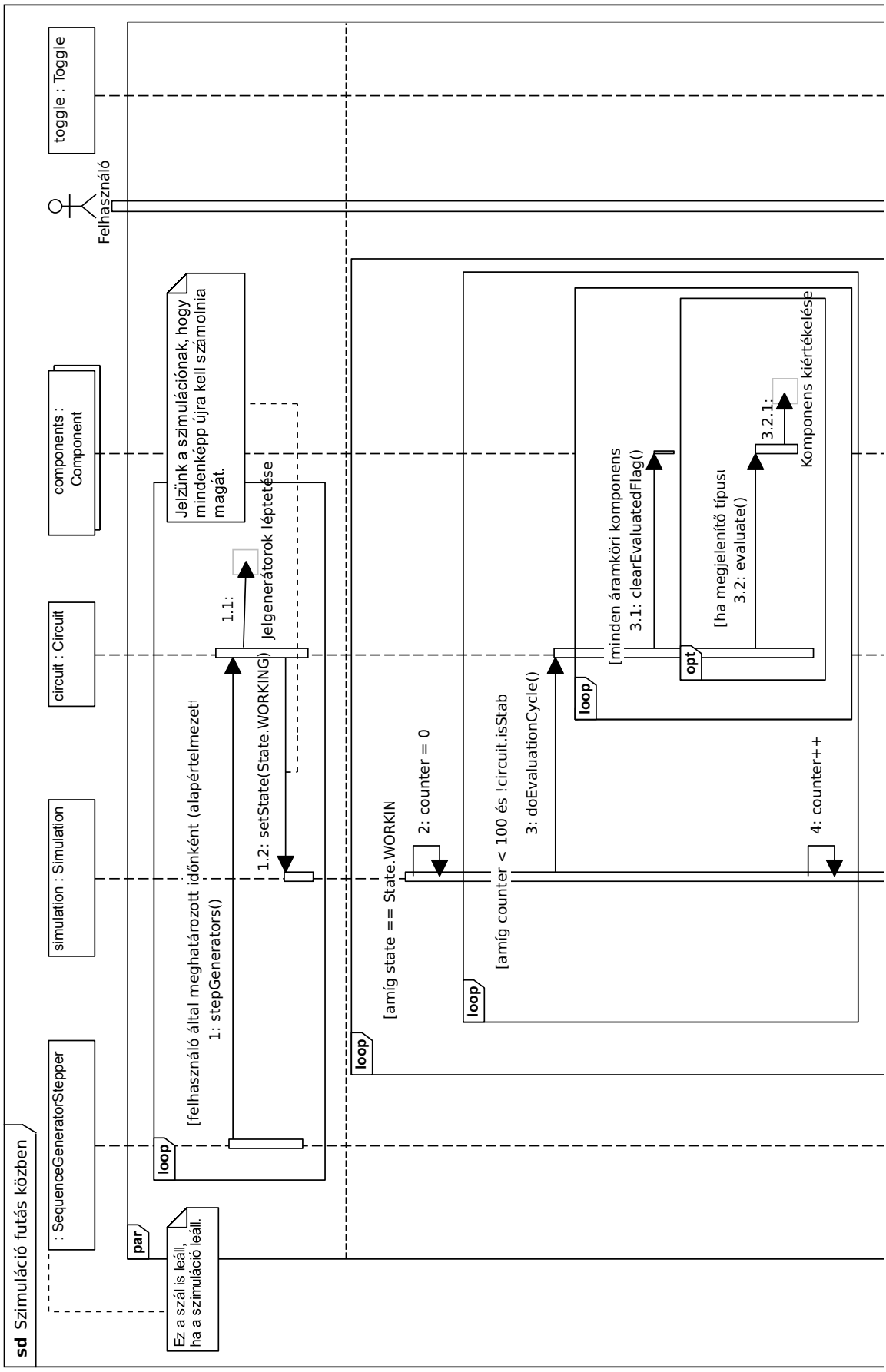
3.3. Statikus struktúra diagramok

[Az előző alfejezet osztályainak kapcsolatait és publikus metódusait bemutató osztálydiagram(ok). Tipikus hibalehetőségek: csillag-topológia, szigetek.]

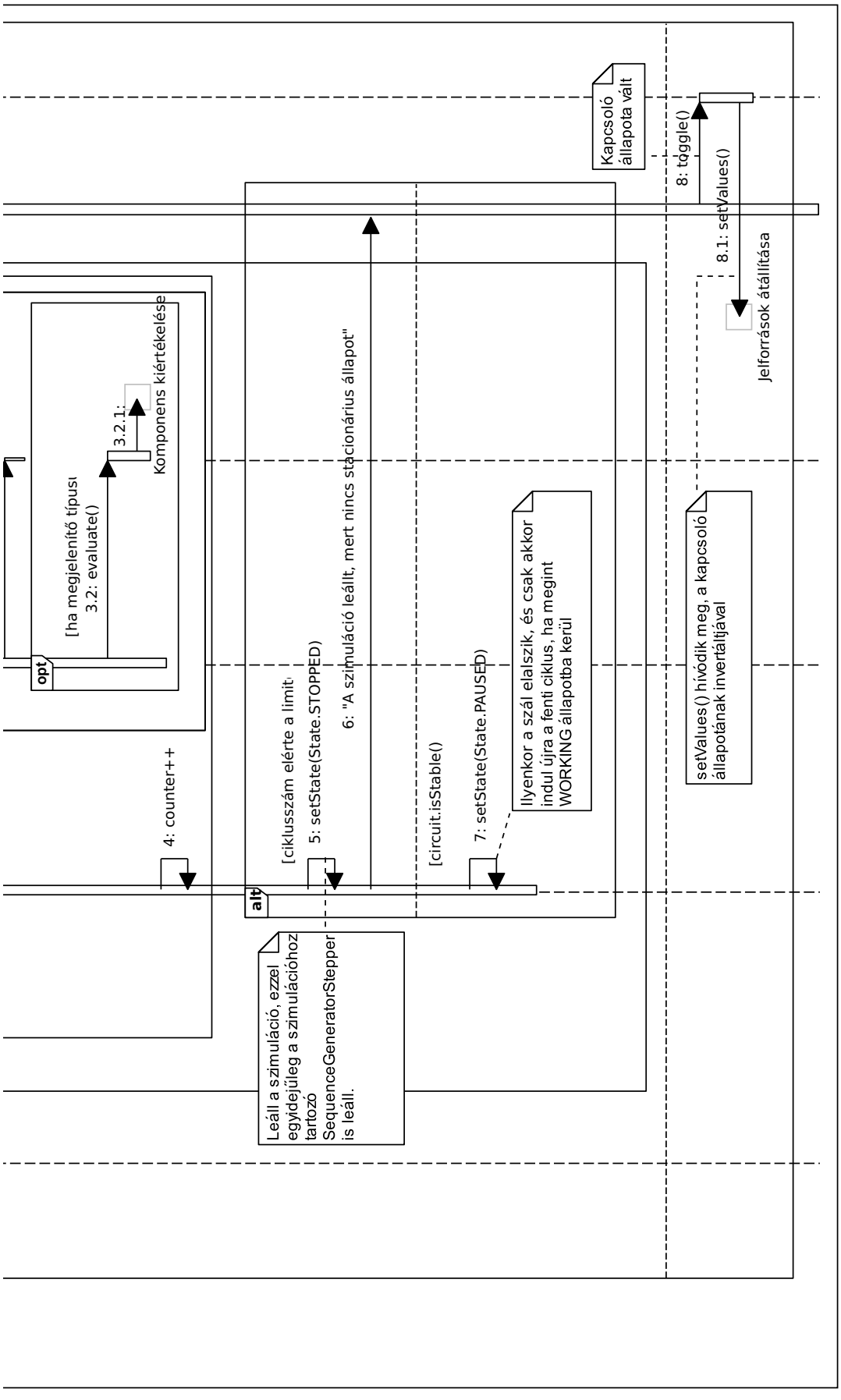
3.1. ábra. x

3.4. Szekvencia diagramok

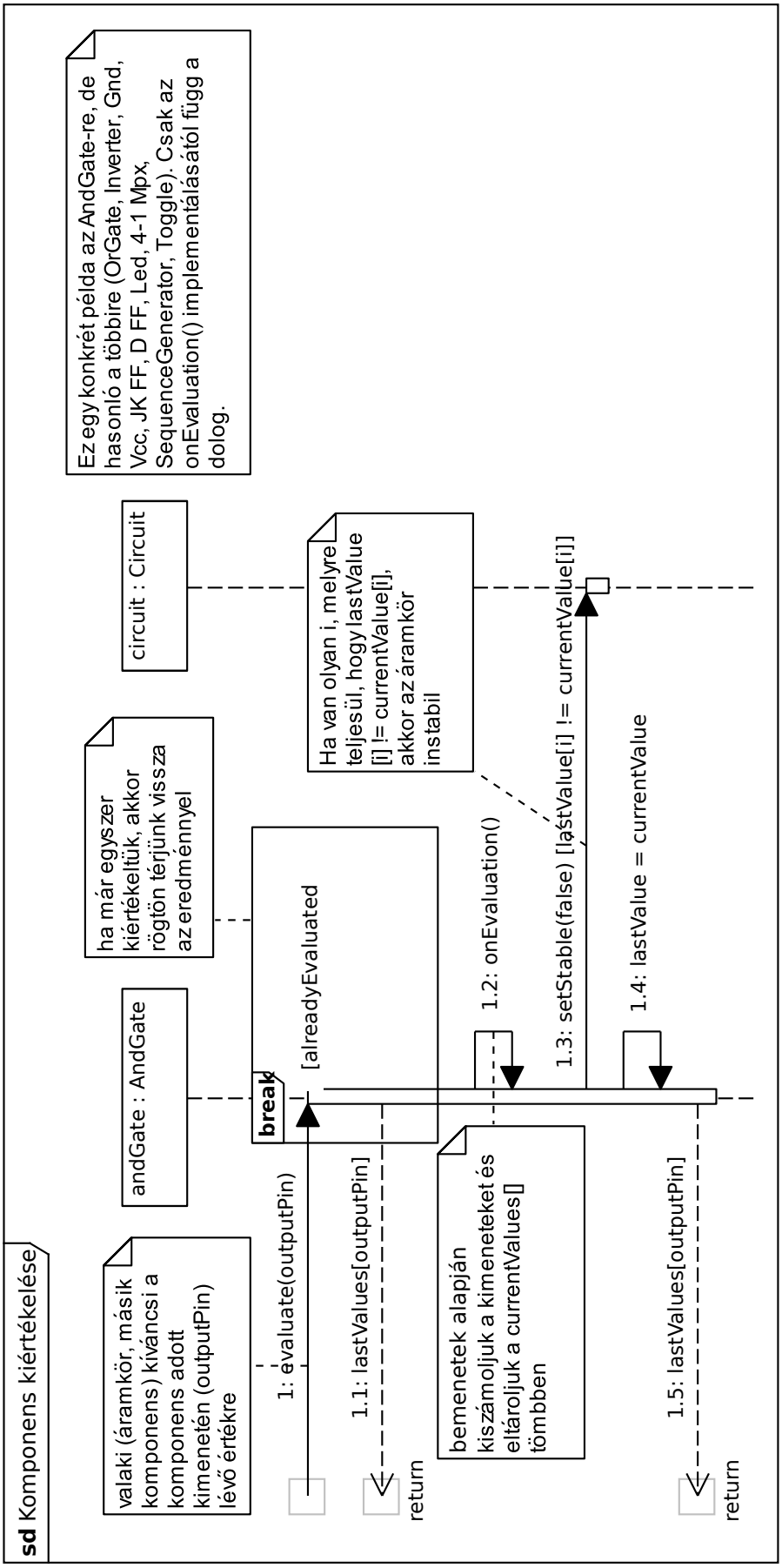
[Inicializálásra, use-case-ekre, belső működésre. Konzisztens kell legyen az előző alfejezettel. Minden metódus, ami ott szerepel, fel kell tűnjön valamelyik szekvenciában. Minden metódusnak, ami szekvenciában szerepel, szereplnie kell a valamelyik osztálydiagramon.]



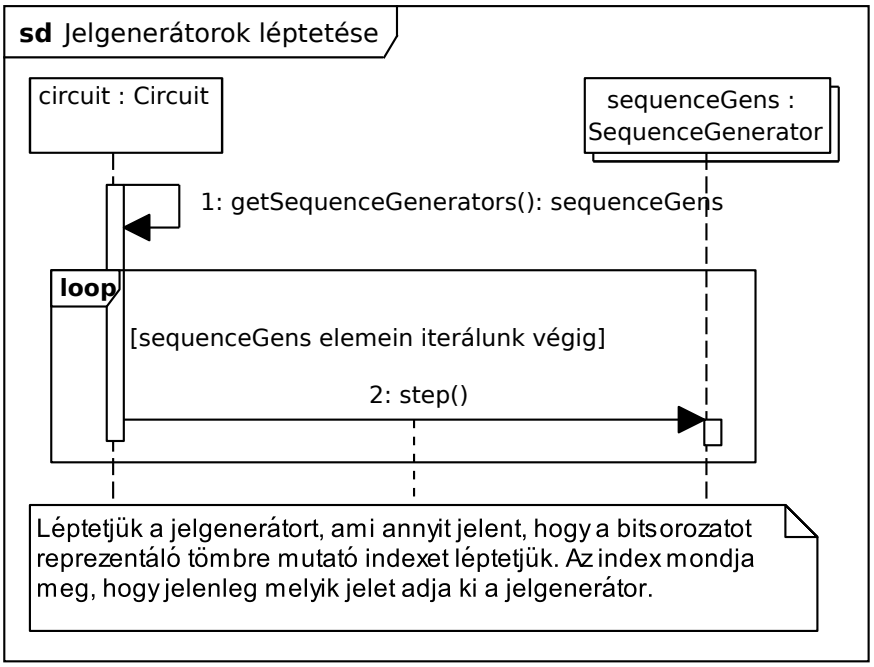
3.2. ábra. Szimuláció futás közben 1. rész



3.3. ábra. Szimuláció futás közben 2. rész



3.4. ábra. Komponens kiértékelése



3.5. ábra. Jelgenerátorok léptetése

3.5. State-chartok

[Csak azokhoz az osztályokhoz, ahol van értelme. Egyetlen állapotból álló state-chartok ne szerepeljenek. A játék működését bemutató state-chart-ot készíteni tilos.]

3.6. ábra. x

3.6. Napló

Kezdet	Időtartam	Résztvevők	Leírás
2011.02.23. 10:00	30 perc	**Kriván B.** **Dévényi A.** **Péter T.** **Apagyi G.** **Jákli G.**	Analízis modell kezdeti lépéseinek megbeszélése
2011.02.23. 15:00	30 perc	**Péter T.**	LaTex és Visual Paradigm for UML szoftverek beállítása.
2011.02.25. 22:00	2 óra	**Péter T.**	Objektum katalógus kitöltése.
2011.02.26. 15:00	1 óra	**Apagyi G.** **Péter T.**	Objektum katalógus elemeinek átbeszélése
2011.02.26. 16:00	1 óra	**Apagyi G.**	Átállás a megbeszélt programokra (Latex, VP)
2011.02.26. 17:00	2 óra	**Apagyi G.**	Sequence diagram (start/stop) szerkesztése
2011.02.26. 18:00	2 óra	**Apagyi G.**	FF és MPX implementálása
...	...	**...**	...