

## 5. Szkeleton tervezése

54 – *Override*

Konzulens:

dr. László Zoltán

### Csapattagok:

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

2011. március 13.

# Tartalomjegyzék

<b>5 Szkeleton tervezése</b>	<b>4</b>
5.1. Errata . . . . .	4
5.1.1. Objektumleírás: <b>Wire</b> . . . . .	4
5.1.2. Objektumleírás: <b>Node</b> . . . . .	4
5.1.3. Osztályleírás: <b>AbstractComponent</b> . . . . .	4
5.1.4. Osztályleírás: <b>Node</b> . . . . .	5
5.1.5. Osztályleírás: <b>Wire</b> . . . . .	5
5.1.6. Statikus struktúra diagramok . . . . .	6
5.2. A szkeleton modell valóságos use-case-ei . . . . .	7
5.2.1. Use-case diagram . . . . .	7
5.2.2. Use-case leírások . . . . .	7
5.3. Architektúra . . . . .	11
5.4. A szkeleton kezelői felületének terve, dialógusok . . . . .	11
5.5. Szekvencia diagramok a belső működésre . . . . .	12
5.6. Napló . . . . .	13

## Ábrák jegyzéke

5.1. Statikus struktúra nézet . . . . .	6
5.2. A szkeleton modell valóságos use-case-ei . . . . .	7
5.3. Áramkör inicializálása . . . . .	13

## 5. Szkeleton tervezése

### 5.1. Errata

Az előző fejezetben leírtak egy apró részletben megváltoztak. Az elemeket már nem közvetlenül kötjük össze, hanem *vezetékek* segítségével, melyeket egymással *csomópont*okkal lehet összekötni, ha szükséges. Így javítottuk a láthatósággal kapcsolatosan felmerült problémákat, ehhez fel kellett venni 2 új osztályt (Wire, Node), illetve az AbstractComponent módosítani, ezekhez tartozó objektum és osztályleírások alább olvashatóak, valamint mellékeljük a módosított statikus osztálydiagramot is. (Egy-két egyéb objektumleírás is módosult, de csak azért mert a kiértékelés logikája változott – nem hátulról megyünk, hanem az összes kiértékeli magát, ez nem szükséges a jelen fejezethez, hiszen magától értetődő)

#### 5.1.1. Objektumleírás: **Wire**

Vezeték, mely az áramköri komponensek ki és bemeneteit köti össze. Egy vezetéket egy darab kimenetet és egy darab bemenetet köt össze. A rajta lévő értéket le lehet tőle kérdezni, illetve be lehet azt állítani.

#### 5.1.2. Objektumleírás: **Node**

Csomópont, mely a bemenetén lévő értéket a kimeneteire adja. Segítségével lehet egy vezetéket „szétágaztatni”.

#### 5.1.3. Osztályleírás: **AbstractComponent**

Absztrakt osztály.

- Felelősség  
Egy komponens absztrakt megvalósítása, ebből származik az összes többi komponens. A közös logikát valósítja meg. A gyakran használt feladatokra ad alapértelmezett implementációt (pl. vezetékek bekötése). Tudja magáról, hogy a legutóbbi két kiértékelés között változtak-e a kimenetei.
- Ősosztályok: (nincs)
- Interfészek: (nincs)
- Attribútumok
  - `protected Wire[] inputs`: Bemeneteire kötött vezetékek.
  - `protected Wire[] outputs`: Kimeneteire kötött vezetékek.
- Metódusok
  - `addTo(Circuit c)`: Meghívja az áramkör `add(AbstractComponent ac)` metódusát.
  - `void evaluate()`: Komponens kimenetein lévő értékek kiszámolása a bemenetek alapján.
  - `boolean isChanged()`: Visszaadja, hogy a legutóbbi két kiértékelés között változtak-e a kimenetek.
  - `void setInput(int inputPin, Wire wire)`: Az adott bemeneti lábára rákötjük a megadott vezetéket.
  - `void setOutput(int outputPin, Wire wire)`: Az adott kimeneti lábára rákötjük a megadott vezetéket.

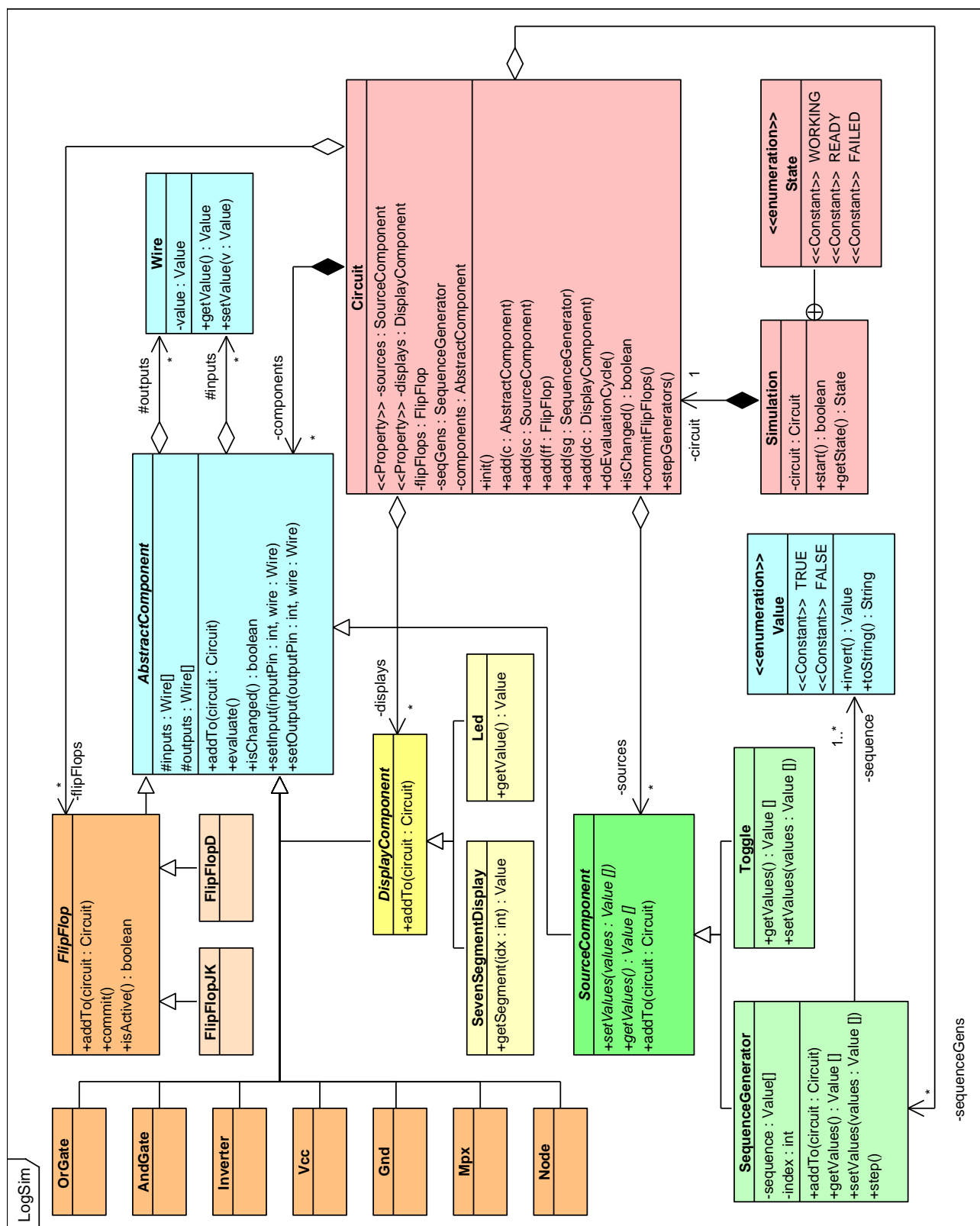
#### 5.1.4. Osztályleírás: **Node**

- Felelősség  
Csomópont, mely a bemenetén lévő értéket a kimeneteire adja. Segítségével lehet egy vezetékot „szét-ágaztatni”.
- Ősosztályok: AbstractComponent.
- Interfészek: (nincs)
- Attribútumok
  - (nincs)
- Metódusok
  - (nincs)

#### 5.1.5. Osztályleírás: **Wire**

- Felelősség  
Vezeték, mely az áramköri komponensek ki és bemeneteit köti össze. Egy vezeték egy darab kimenetet és egy darab bemenetet köt össze. A rajta lévő értéket le lehet tőle kérdezni, illetve be lehet azt állítani.
- Ősosztályok: AbstractComponent.
- Interfészek: (nincs)
- Attribútumok
  - `private Value value`: Vezetéken lévő érték
- Metódusok
  - `Value getValue()`: Visszaadja a vezetéken lévő értéket.
  - `void setValue(Value v)`: Beállítja a vezetéken lévő értéket.

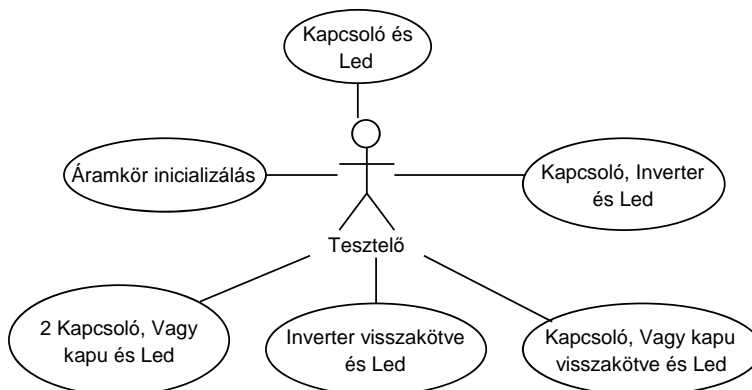
## 5.1.6. Statikus struktúra diagramok



5.1. ábra. Statikus struktúra nézet

## 5.2. A szkeleton modell valóságos use-case-ei

### 5.2.1. Use-case diagram



5.2. ábra. A szkeleton modell valóságos use-case-ei

### 5.2.2. Use-case leírások

Use-case neve	Áramkör inicializálása
Rövid leírás	Ez a usecase egy áramkör és a hozzá tartozó szimuláció inicializálását mutatja be, hogyan jönnek létre a komponensek és a közöttük lévő összeköttetés. Jelen példa egy Kapcsoló és egy Led összeköttetését prezentálja.
Aktorok	Tesztelő
Forgatókönyv	<ul style="list-style-type: none"> <li>• szimuláció létrehozása</li> <li>• áramkör létrehozása</li> <li>• áramkör bejegyzése a szimulációba</li> <li>• áramkör inicializálása               <ul style="list-style-type: none"> <li>– kapcsoló létrehozása</li> <li>– vezeték létrehozása</li> <li>– kapcsoló kimenetére vezeték kötése</li> <li>– led létrehozása</li> <li>– led bemenetére vezeték kötése</li> <li>– kapcsoló áramkörbe regisztrálása</li> <li>– led áramkörbe regisztrálása</li> </ul> </li> </ul>

Use-case neve	Kapcsoló és Led
Rövid leírás	Ez a usecase egy olyan áramkör tesztelését mutatja be, amely egy kapcsolóból és rá kötött ledből áll.
Aktorok	Tesztelő

Forgatókönyv	<ul style="list-style-type: none"> <li>• Áramkör és komponensek létrehozása</li> <li>• kapcsoló értékének beállítása (<b>megkérdezi a tesztelőt</b>)</li> <li>• szimuláció indítása             <ul style="list-style-type: none"> <li>– hálózat kiértékelés indítása                 <ul style="list-style-type: none"> <li>* kapcsoló kiértékelése (állapotának kijelzése)</li> <li>* led kiértékelése (világít/nem világít kijelzése)</li> </ul> </li> <li>– áramkör változásának vizsgálata</li> <li>– stacionárius állapot, szimuláció vége</li> </ul> </li> </ul>
--------------	---

Use-case neve	Kapcsoló, Inverter és Led
Rövid leírás	Ez a usecase egy olyan áramkör tesztelését mutatja be, amely egy kapcsolóból egy rá kötött inverterből és egy arra kötött ledből áll.
Aktorok	Tesztelő
Forgatókönyv	<ul style="list-style-type: none"> <li>• Áramkör és komponensek létrehozása</li> <li>• kapcsoló értékének beállítása (<b>megkérdezi a tesztelőt</b>)</li> <li>• szimuláció indítása             <ul style="list-style-type: none"> <li>– hálózat kiértékelés indítása (2x)                 <ul style="list-style-type: none"> <li>* kapcsoló kiértékelése (állapotának kijelzése)</li> <li>* inverter kiértékelése                     <ul style="list-style-type: none"> <li>· bemenetén lévő érték lekérése</li> <li>· kimenetére kötött érték kiszámolása és kiadása</li> </ul> </li> <li>* led kiértékelése (világít/nem világít kijelzése)</li> </ul> </li> <li>– áramkör változásának vizsgálata</li> <li>– két lépés alatt stacionárius állapot<sup>1</sup>, szimuláció vége</li> </ul> </li> </ul>

Use-case neve	2 Kapcsoló, Vagy kapu és Led
Rövid leírás	Ez a usecase egy olyan áramkör tesztelését mutatja be, amely egy vagy kapura kötött két kapcsolóból és a vagy kapu kimenetére kötött ledből áll.
Aktorok	Tesztelő

<sup>1</sup>amennyiben a kapcsoló logikai igazra van állítva, akkor egy lépés is elég, de két lépés biztosan, így ezt ábrázoljuk diagramon  
2011. március 13.



Forgatókönyv	<ul style="list-style-type: none"> <li>• Áramkör és komponensek létrehozása</li> <li>• egyik kapcsoló értékének beállítása (<b>megkérdezi a tesztelőt</b>)</li> <li>• másik kapcsoló értékének beállítása (<b>megkérdezi a tesztelőt</b>)</li> <li>• szimuláció indítása             <ul style="list-style-type: none"> <li>– hálózat kiértékelés indítása (2x)                 <ul style="list-style-type: none"> <li>* egyik kapcsoló kiértékelése (állapotának kijelzése)</li> <li>* másik kapcsoló kiértékelése (állapotának kijelzése)</li> <li>* VAGY kapu kiértékelése                     <ul style="list-style-type: none"> <li>· bemenetén lévő értékek lekérése</li> <li>· kimenetére kötött érték kiszámolása és kiadása</li> </ul> </li> <li>* led kiértékelése (világít/nem világít kijelzése)</li> </ul> </li> <li>– áramkör változásának vizsgálata</li> <li>– második lépés után stacionárius állapot<sup>2</sup>, szimuláció vége</li> </ul> </li> </ul>
--------------	--

<sup>2</sup>amennyiben mindkét kapcsoló 0-ás állapotban van, egy lépés alatt stabil lesz a hálózat, hiszen a VAGY kapu végig hamis állapotot ad ki, itt és a szekvencia diagramon úgy vesszük, mintha legalább az egyik kapcsoló 1-esbe lenne állítva.

Use-case neve	Inverter visszakötve és Led
Rövid leírás	Ez a usecase egy olyan áramkör tesztelését mutatja be, amely egy inverterből, amelynek kimenete egy ledbe illetve saját bemenetére van kötve. Oszcillálni fog, ezért a szimuláció rövid időn belül leáll.
Aktorok	Tesztelő
Forgatókönyv	<ul style="list-style-type: none"> <li>• Áramkör és komponensek létrehozása</li> <li>• szimuláció indítása <ul style="list-style-type: none"> <li>– hálózat kiértékelés indítása (3x) <ul style="list-style-type: none"> <li>* inverter kiértékelése <ul style="list-style-type: none"> <li>· bemenetén lévő értékek lekérése</li> <li>· kimenetére kötött érték kiszámolása és kiadása</li> </ul> </li> <li>* csomópont kiértékelése <ul style="list-style-type: none"> <li>· bemenetén lévő érték lekérése</li> <li>· kimeneteire az érték kiadása</li> </ul> </li> <li>* led kiértékelése (világít/nem világít kijelzése)</li> </ul> </li> <li>– áramkör változásának vizsgálata</li> <li>– harmadik lépés után sincs stacionárius állapot, szimuláció vége</li> </ul> </li> </ul>

Use-case neve	Kapcsoló, Vagy kapu visszakötve és Led
Rövid leírás	Ez a usecase egy olyan áramkör tesztelését mutatja be, amely egy kapcsolóból, egy VAGY kapuból, melynek egyik bemenetére a kapcsoló, másik bemenetére a saját kimenete van kötve és egy ledből, melyre szintén a VAGY kapu kimenetét kötöttük. Ez egy olyan visszakötéses hálózat, mely stabil állapotban van.
Aktorok	Tesztelő

Forgatókönyv	<ul style="list-style-type: none"> <li>• Áramkör és komponensek létrehozása</li> <li>• szimuláció indítása <ul style="list-style-type: none"> <li>– hálózat kiértékelés indítása (2x) <ul style="list-style-type: none"> <li>* kapcsoló kiértékelése (állapotának kijelzése)</li> <li>* VAGY kapu kiértékelése <ul style="list-style-type: none"> <li>· bemenetén lévő értékek lekérése</li> <li>· kimenetére kötött érték kiszámolása és kiadása</li> </ul> </li> <li>* csomópont kiértékelése <ul style="list-style-type: none"> <li>· bemenetén lévő érték lekérése</li> <li>· kimeneteire az érték kiadása</li> </ul> </li> <li>* led kiértékelése (világít/nem világít kijelzése)</li> </ul> </li> <li>– áramkör változásának vizsgálata</li> <li>– második lépés után stacionárius állapot<sup>3</sup>, szimuláció vége</li> </ul> </li> </ul>
--------------	--

### 5.3. Architektúra

### 5.4. A szkeleton kezelői felületének terve, dialógusok

Az általunk elkészített szkeleton egy program váz melynek felülete egy egyszerű konzolos megjelenítési felület, amely alkalmas arra, hogy a use case-k által leírt teszteseteket bemutassuk. Az egyes tesztesetek a neki megfelelő use case sorszámaival van elnevezve, így program indítás után egy szám bevitelét követően a kiválasztott teszteset lefut. A teszteset futása közben kiír minden objektumot amin metódust hív, illetve kiírja a metódus nevét a paraméterekkel együtt, majd a visszatérési értéket. Ez azért lehetséges, mert a szkeleton már tartalmazza az elkészítendő szoftver összes fontos osztályát és metódusát, azonban az üzleti logikát még nem. Így könnyen eldönthető, hogy a use case-nek megfelelően viselkedik a program és továbbiakban képes lesz-e megfelelően működni. A tesztelési folyamat során döntési helyzet léphet fel. Ilyenkor a program felteszi a kérdést, majd a kapott válasz alapján folytatja a további futást. Ezzel csökkentjük a tesztesetek számát, anélkül, hogy bizonyos esetek kimaradnának a tesztelés alól. Futás közben megjegyzés formájában a program tájékoztat néhány elem belső állapotáról (például kapcsoló értéke) vagy bizonyos fontosabb lépésekről (például inicializálás). Az elvárás, hogy a szkeleton a szekvenciadiagramok által leírt működést mutassa. A program egyszerű és könnyen összehasonlítható formában írja ki a működését, amelyet könnyen összevethetjük a szekvencia diagrammokkal.

Egy metódushívás és visszatérés esetén kiírt adatok a következők:

- Metódushívás esetén a CALL szót, konstruktorhívás esetén CREATE szót, míg visszatéréskor a RETURN szót
- Objektum neve
- A metódus neve és a metódus paramétereinek értékét
- Visszatérés esetén a visszatérési értéket

Egy döntési helyzetben a kiírt adatok a következők:

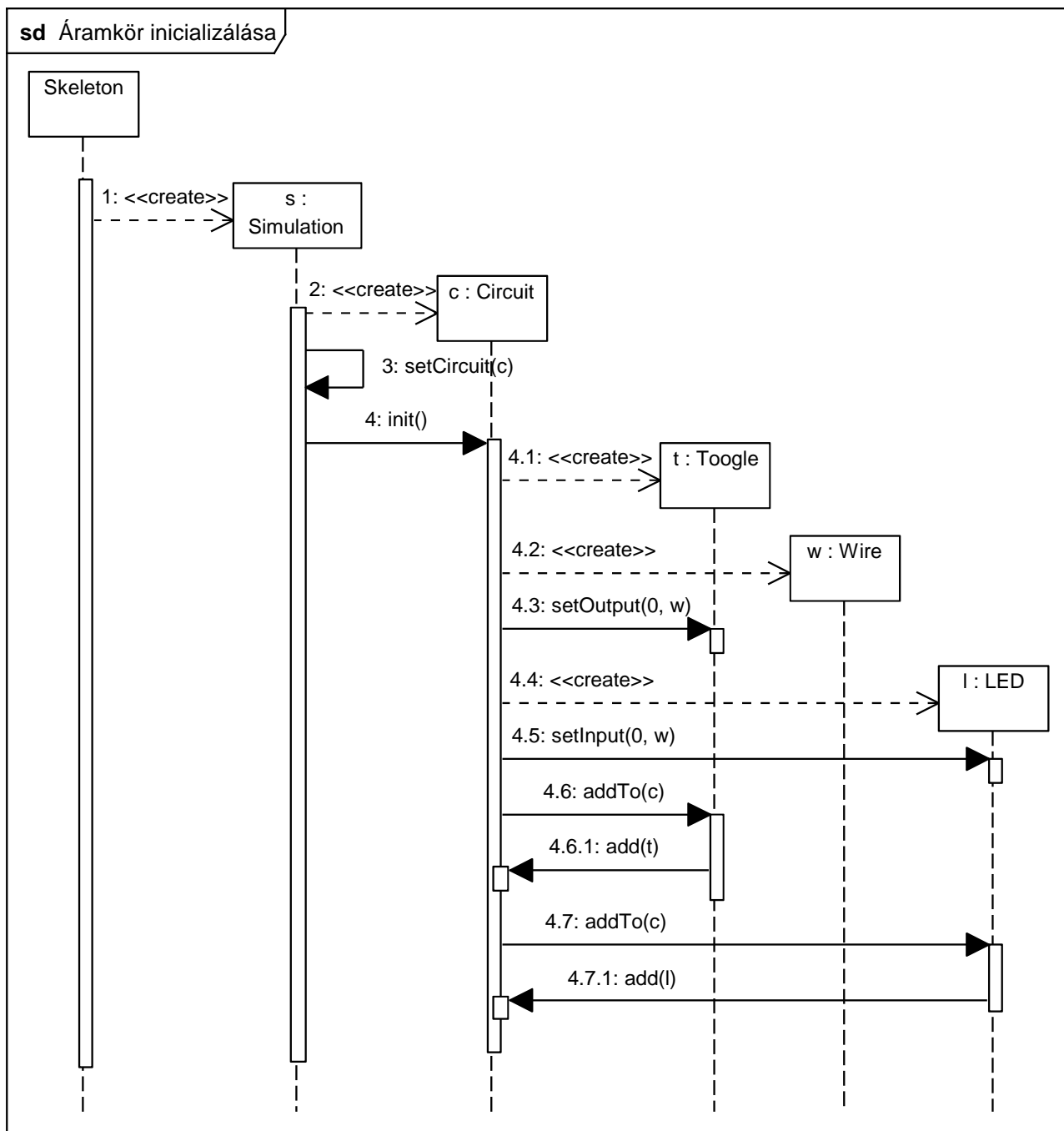
- QUESTION szó

<sup>3</sup>ha a kapcsoló 0-ás állapotban van, akkor egy lépés alatt bekövetkezik, de érdekesebb szituáció, amikor 1-es állapotban van, ezt ábrázoljuk diagramon

- objektum neve
- Egy rövid magyarázó szöveg
- Szögletes zárójelben a lehetséges válaszok

### 5.5. Szekvencia diagramok a belső működésre

*[A szkeletonban implementált szekvenciadiagramok. Tipikusan egy use-case egy diagram. Ezek megegyezhetnek a korábban specifikált diagramokkal, de az egyes élvonalakat (lifeline) egyértelműen a szkeletonban példányosított objektumokhoz kell tudni kötni. Azt kell megjeleníteni, hogy a szkeletonban létrehozott objektumok egymással hogyan fognak kommunikálni.]*



5.3. ábra. Áramkör inicializálása

## 5.6. Napló

Kezdet	Időtartam	Résztevők	Leírás
2010.03.12. 14:00	1,5 óra	<b>Kriván B.</b>	Javaolt módosítások elvégzése az előző fejezetben, rövid errate készítése jelen fejezet elé.
2010.03.13. 00:00	2 óra	<b>Péter T.</b>	Use-casek leírása szöveges formátumban
2010.03.13. 09:30	30 perc	<b>Kriván B.</b>	Use-case diagram megrajzolása

Kezdet	Időtartam	Résztevők	Leírás
2010.03.13. 10:00	2 óra	Kriván B.	Use-casek leírásának $\text{\LaTeX}$ formátumra való alakítása, apróbb finomítások
...	...	...	...