**1. Követelmények**

54 – Override

Konzulens:

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 |

2011.02.16.

# Követelmények leírása

## A követelmények leírása

### Digitális rendszer modellezése

Egy digitális rendszer (áramkör) alapvető építőelemekből áll. Az építőelemek be- és/vagy kimenetekkel rendelkeznek. A kimenetek értéke a bemenetek értéke alapján alakul ki. Egy kimenetre tetszőleges számú bemenet csatlakoztatható, amely hatására az adott bemenetek felveszik a kimenet által adott logikai (igaz vagy hamis) szintet. Több kimenet összekapcsolása nem megengedett. Üresen hagyott (semmihez sem kapcsolódó) bemenet nem megengedett.

Az alapvető építőelemek:

* INVERTER - az inverter egy bemenettel és egy kimenettel rendelkezik, a kimenet értéke a bemenet logikai negáltja.
* ÉS kapu - az és kapu két vagy több bemenettel és egy kimenettel rendelkezik. A kimenet értéke a bemenetek logikai és kapcsolata.
* VAGY - a vagy kapu két vagy több bemenettel és egy kimenettel rendelkezik. A kimenet értéke a bemenetek logikai vagy kapcsolata.
* Kapcsoló - a kapcsoló jelforrás, egy kimenete van. "Be" állásban a kimenete igaz, "ki" állásban pedig hamis.
* Jelgenerátor - a jelgenerátor jelforrás, egy kimenettel. A kimenet előre definiált jelsorozatot ismétel ciklikusan.
* Kijelző - a kijelző a jel értékét megjelenítő elem, egy bemenete van. Bemenetének igaz estén világít, hamis esetében pedig nem világít.

Későbbiekben új építőelemek megjelenése is várható.

Az építőelemekből tetszőleges logikai hálózat felépíthető, amely a jelforrások, mint bemeneti értékek hatására működést produkál, amely a kijelzőkön keresztül megfigyelhető.

# Követelmény, projekt funkcionalitás

## Követelmény definíció

[A 2.1. és 2.2. dokumentumok tartalomjegyzéke a Szoftvertechnológia c. tárgy fóliáin megtalálható. De követhetők a Software Engineering tankönyvekben közölt tartalomjegyzékek és egy korábbi [minta](http://kfarkas.iit.bme.hu/%7Eszoftlab4/mintapelda.pdf) is. Mindenképpen specifikálni kell a szoftver fordításával, futtatásával és telepítésével kapcsolatos rendszer(ek) paramétereit.]

### A program célja, alapvető feladata

### A fejlesztőkörnyezet

A fejlesztéshez Netbeans 6.9.1 szoftvert választottuk. Modellezéshez a NetBeans-hez letölthető Use Case Model plugint használjuk, mely képes az objektum-diagramból Java forráskódot generálni. Közben ügyelünk arra, hogy a program kompatibilis legyen a Sun Java 1.6-os szabványával. Természetesen a cél az, hogy a modellező program a Hallgatói Számítógép Központban rendszeresített JDK alatt fordítható és futtatható legyen. A dokumentumokat Microsoft Office és Open Office programokkal, DOCX formátumban készítjük. A unit-tesztekre a JUnit csomagot fogjuk használni.

### A futtatáshoz szükséges környezet

Java Runtime Environment, illetve az a számítógép, mely ezt futtatni képes. (A Sun ajánlásai PC-re: Pentium 166Mhz vagy gyorsabb processzor és 32Mb memória.) A játék használatához grafikus képernyő és egér szükséges.

### A felhasználói felület

A játékprogram végső változata grafikus felhasználói felülettel rendelkezik. A programot a felhasználó az egér segítségével vezérelheti.

### Minőségi tényezők

**Teljesítmény**: A cél az, hogy a digitális rendszermodellező szoftver használható legyen a fentebb meghatározott minimális rendszeren. A grafikus felületnél törekedni fogunk a folyamatos szimuláció megjelenítése.

**Újrafelhasználhatóság**: A cél az, hogy a grafikus felhasználói felületet a program többi részétől teljesen különválasszuk, így lehetővé téve azt, hogy később a grafikus felület egyszerűen és gyorsan változtatható legyen.

**Rugalmasság**: A rugalmasságot a fejlesztőkörnyezet biztosítja, a modellező szoftvernek ugyanis minden olyan környezetben futtathatónak kell lennie, melyben létezik megfelelő Java futtatókörnyezet.

**Felhasználhatóság**: A használat különösebb tanítást nem igényel, alapfokú számítástechnikai tudással akár a felhasználói kézikönyv elolvasása nélkül is használható.

### A software minősítése

A kifejlesztett software akkor megfelelő, ha minél pontosabban megegyezik a fentebb leírtakkal. Ezt ellenőrizni lehet a játék futtatásával és kipróbálásával, illetve a forráskód és a modell összevetésével.

### A kibocsátás

A program kibocsátása először a forráskóddal együtt a konzulens felé fog történni.

## Projekt terv

### Fejlesztői csapat

|  |  |
| --- | --- |
| **Csapattag neve** | **Feladatköre** |
| Kriván Bálint | csapatvezető, kód, dokumentáció, szervezés |
| Jákli Gábor | kód, dokumentáció, ticket koordinálás |
| Dévényi Attila | kód, dokumentáció |
| Apagyi Gábor | kód, dokumentáció |
| Péter Tamás Pál | kód, dokumentáció |

### Életciklus modell

A feladat először a program megtervezése, mely a dinamikus- és objektummodelleket foglalja magába. Ha ez készen van, elkezdhető a skeleton implementálása. Ez a lépés már teljesen meghatározott, nem merülhet fel semmilyen komplikáció, ha a modellek megfelelőek voltak.

A következő feladat a prototípus elkészítése. A programnak ebben az állapotban könnyen tesztelhetőnek kell lennie, hogy a programozási és funkcionalitási logikai hibák könnyen felismerhetők legyenek. A könnyű tesztelhetőség azt jelenti, hogy a bemenetet és a kimenetet is lehet állományból illetve állományba generálni, hogy ennek kiértékelése is egyszerű legyen.

Ha már a prototípus is megfelelő, akkor kezdődhet a grafikus felület megvalósítása. Itt is fontosa tesztelés és a kiértékelés, mert a jó megjelenés sokat számít a modellező használhatóságában. Ha ennek kifejlesztése is sikeres, készen van a program első teljes változata. A kötelező feladat csak eddig tart. Ezt a változatot kell leadni a dokumentációval és a forráskóddal együtt.

### Szervezési struktúra

A csapat öt emberből áll. A feladat szempontjából a tudásunk nem azonos, mindenki más-más területet érez a magáénak, illetve a feladat eltérő részinek megoldásához van nagyobb kedvünk. Azt a felépítést választottuk, hogy mindenki az érdeklődésének és tudásának legmegfelelőbb részt kapja az egész feladatból. A feladatok szétosztását találkozókon beszéljük meg, ahol az egyéni kívánságok mellett ügyelünk arra, hogy minden feladat kiosztásra kerüljön, valamint a csapattagok az egész feladat megoldásából nagyjából egyenlő mértékben vegyék ki a részüket. A találkozók keretében, mivel a szétosztott feladatok nagy mértékben függnek egymástól, javaslatokat teszünk egymásnak a feladat megoldásának körülményeit és a határidőt illetően. Ezt követően a csapattagok elvégzik a rájuk kiosztott feladatot, melyek a következő találkozón való egyeztetés után tehetők közzé.

Hogy a fejlesztés minél hatékonyabb és zökkenő mentesebb legyen, a következő eszközöket alkalmazzuk:

**E-mail**: Az egymás számára fontos anyagokat, melyeket a találkozókon előzetesen megbeszéltünk, levélben küldjük el.

**Msn**: Felvettük egymást a Microsoft Messenger-be, hogy szükség esetén egymástól is segítséget tudjunk kérni kisebb technikai problémák megoldásában. Természetesen ezek a feladat lényegét, a projectről hozott döntéseket nem érinthetik, de kivételes helyzetben akár az Interneten is tarthatunk találkozót.

**Git-repo**: A feladatok megoldása közben keletkezett anyagokat egy, kizárólag a csapat tagjai által hozzáférhető helyen tároljuk. Így mindig elérhető a fejlesztések legfrissebb változata. Csapatunk a [www.codaset.com](http://www.codaset.com) oldal által nyújtott ingyenes git-repo-t használja.

**Ticket**: A fejlesztés során felmerülő problémákat, kérdéseket ticket formájában megírjuk egymásnak, amit később a kijelölt felelős személy megold, ha szükséges, akkor együtt konzultálunk a megoldás módjáról, menetéről.

### Fejlesztési ütemterv

A program fejlesztésének három fő lépcsőfoka van. Ezek a következők:

**Skeleton**: A cél az, hogy mind a dinamikus, mind az objektum modell jól legyen kitalálva. Ha ezek elkészültek, akkor a fejlesztés szempontjából sikeresen leraktuk az alapokat.

**Prototípus**: Ez már szinte a teljes változat, csak a grafikus felület elemei hiányoznak. Ez a változat tökéletesen megfelelő arra, hogy az objektumok, rutinok, függvények szemantikai helyességét vizsgáljuk.

**Grafikus változat:** A program teljes változata. Tulajdonképpen a prototípus a grafikus felülettel

kiegészítve, esetleg kismértékben továbbfejlesztve.

### Határidők

|  |  |
| --- | --- |
| febr. 11. | Csapat regisztráció |
| febr. 21. | Követelmény, projekt, funkcionalitás |
| febr. 28. | Analízis modell kidolgozása 1. |
| márc. 7. | Analízis modell kidolgozása 2. |
| márc. 14. | Szkeleton tervezése |
| márc. 21. | Szkeleton |
| márc. 28. | Prototípus koncepciója |
| ápr.  4. | Részletes tervek |
| ápr. 18. | Prototípus |
| ápr. 26. | Grafikus felület specifikációja |
| máj. 9. | Grafikus változat |
| máj. 13. | Összefoglalás |

## Feladatleírás

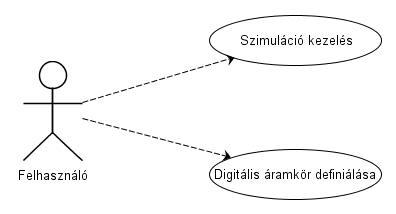
[A feladat kb. 4000 karakteres (kb 1,5 oldal) részletezettségű magyar nyelvű leírása. Nem szerepelhetnek informatikai kifejezések.]

## Szótár

[A szótár a részletes leírás alapján készítendő dokumentum. Egy szótári bejegyzés definiálásához csak más szótári bejegyzések és köznapi – a feladattól független – fogalmak használhatók fel. A szótár mérete kb. 1-2 oldal legyen.]

## Essential use-case-ek

### Use-case diagram



1. Use-case diagram

### Use-case leírások

**Use Case:** Szimuláció kezelés

**Actor:** Felhasználó

**Leírás:** A felhasználó új szimulációt indít vagy leállít

**Use Case:** Digitális áramkör definiálása

**Actor:** Felhasználó

**Leírás:** A felhasználó digitális áramkört betölti, szerkeszti

## Napló

|  |  |  |  |
| --- | --- | --- | --- |
| **Kezdet** | **Időtartam** | **Résztvevők** | **Leírás** |
| 2011.02.16. 09:00 | 2 óra | Kriván B.  Dévényi A.  Apagyi G.  Péter T. | Értekezlet.  Megbeszéltük a projekt szervezését, struktúráját; kommunikáció formáit. Algoritmusokról is konzultáltunk. |
| 2011.02.16. 15:00 | 1 óra | Jákli G. | Projektterv írás |
| 2011.02.16. 20:00 | 1 óra | Jákli G. | Projektterv írás folytatása |
| 2011.02.16. 22:00 | 20 perc | Kriván B. | Formázás és számozás javítása |
| 2011.02.17. 16:00 | 15 perc | Apagyi G. | Use-case diagram |