Vasúti biztosítóberendezések formális specifikációjának formális modellre való automatizált transzformálásának lehetőségei

Lukács Gábor* Dr. Bartha Tamás**

**Budapesti Műszaki és Gazdaságtudományi Egyetem, 1111 Stoczek u. 2. (e-mail: lukacs.gabor@mail.bme.hu).*
***Budapesti Műszaki és Gazdaságtudományi Egyetem, 1111 Stoczek u. 2. (e-mail: bartha.tamas@mail.bme.hu).*

Absztrakt: A biztonságkritikus, beágyazott vasúti biztosítóberendezések fejlesztése során felhasználható formális módszerek gyakorlati alkalmazása egyre inkább a figyelem középpontjába kerül, melynek hátterében a formális módszerek matematikai logikán alapuló, precíz leíró képessége áll. A formális módszerek egyik előnye, hogy alkalmazásukkal jelentősen növelhető a valószínűsége annak, hogy a rendszer formális specifikációja helyes és teljes lesz, melyet például modellellenőrzéssel, mint verifikációs eljárással is megerősíthetünk. Kutatásunk célja egy olyan specifikációs környezet kialakítása, mely a biztosítóberendezési mérnökök számára megkönnyíti a formális specifikáció készítését anélkül, hogy a háttérben meghúzódó elméleti ismereteket el kellene sajátítaniuk. Jelen cikkükben a részletes specifikáció formális modellé való transzformálási kérdéseivel kapcsolatos tapasztalatainkat foglaljuk össze egy esettanulmány segítségével.

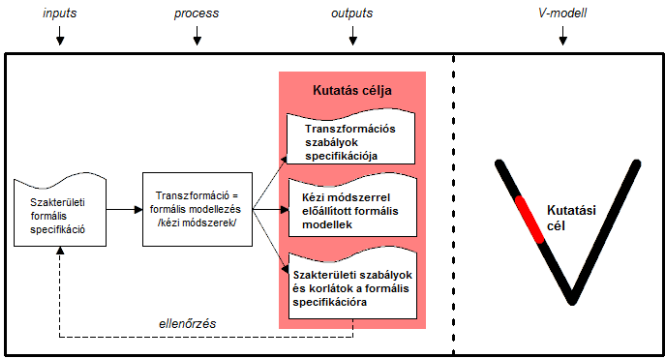
1. BEVEZETÉS

A formális módszerek alkalmazása a vasúti biztosító-berendezés fejlesztési terülten nagy múlttal rendelkezik [1 – 2]. Számos elméleti eredményt közlő publikáció jelenik meg [4], azonban a módszerek és technikák napi szintű mérnöki gyakorlatba való átültetése még várat magára. Az gyakorlati alkalmazási nehézségek mögött számos probléma húzódik, a teljesség igénye nélkül pl. a szükséges háttérismeretek hiánya a szakterületi mérnököknél (két tudományág alkalmazásának az igénye: számítástudomány és közlekedéstudomány), a hagyományos fejlesztési technikákhoz képesti többlet erőforrásigény, az egyszerű rendszerelemek integrációját követően adódó nagyméretű állapotterek kezelésének kérdésköre stb.

A formális módszereket a biztonságkritikus vasúti biztosító-berendezés fejlesztések esetében figyelembe veendő szabványok, pl. [8] is ajánlják. A szabványok a formális módszereket SIL3-SIL4 biztonságintegritási szinten HR (highly recommended) technikák közé sorolják.

Kutatásunk célja egy olyan specifikációs környezet előkészítése, mely a biztosítóberendezési mérnökök számára megkönnyíti a formális specifikáció készítését anélkül, hogy a háttérben meghúzódó elméleti ismereteket el kellene sajátítaniuk. Jelen írásunkban a részletes specifikáció formális modellé való transzformálási kérdéseivel kapcsolatos tapasztalatainkat foglaljuk össze egy esettanulmány segítségével. A transzformáció célja egy Petri-háló modell létrehozása, az ehhez szükséges transzformációs szabályok definiálása. Kutatásunk távoli célja, hogy a feltárt transzformációs szabályok alapján a folyamatot automatizáltan hajthassuk végre.

A transzformációs tevékenységet (process), valamint annak be- és kimeneteit (iputs és outputs) ill. a tevékenység elhelyezkedését az életciklus modellben a 1. ábrán foglaltuk össze.



1. ábra. A kutatás célja a fejlesztési életciklus modellben

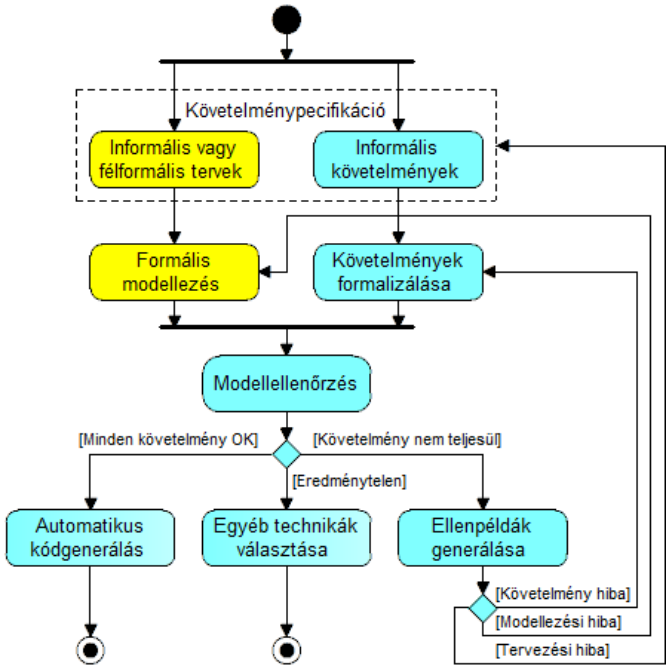
2. FORMÁLIS MODELLEZÉS

A formális modellezés [4 – 5] célja, hogy felhasználva a benne elért eredményeket azonosítsuk és eliminálják a specifikációs következetlenségeket, kétértelműségeket, hiányosságokat stb. Célja, hogy a rendszer tényleges elkészülése előtt képet kapjunk annak működéséről és ellenőrizhessük, hogy az általunk kívánt módon viselkedik-e (természetesen a modellezés korlátain belül).

A formális modellezés gyakorta grafikus megjelenítéssel párosul, amely segíti az átláthatóságot, érthetőséget. A grafikus megjelenítés lehetőséget teremt a specifikált viselkedés szimulációjára. A grafikus megjelenítés és szimuláció képessége azonban sok esetben a modellező eszköztől függ.

A formális modelleken formális verifikációt hajthatunk végre (pl. modellellenőrzés) és segítségükkel az automatikus kódgenerálás is előkészíthető.

A formális modellezés egy lehetséges gyakorlati felhasználására látunk példát a 2. ábrán [5 – 7].



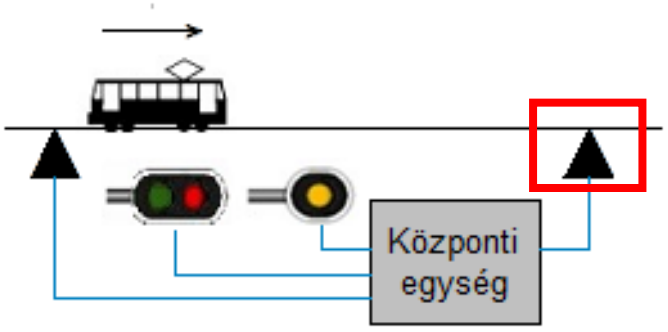
2. ábra. A formális modellezés egy lehetséges felhasználása

A 2. ábrán látható folyamat alapját a követelmény-specifikáció képezi. A követelményspecifikáció két fő leíró részből áll, melyek a rendszer viselkedését és tulajdonságait definiálják. A viselkedést leíró specifikációs részekből a formális modellezés során formális modelleket építhetünk, míg rendszer tulajdonságait leíró specifikációs részekből formalizált követelményeket állíthatunk elő.

A modellellenőrzés [5] egy olyan formális módszer, amely a vizsgálandó rendszer egy modelljéről (viselkedés leírás) és annak elvárt működését tartalmazó követelmény specifikációjáról (tulajdonság leírás) a rendszermodell teljes állapotterének szisztematikus bejárásával dönti el, hogy a rendszermodell a követelményspecifikációt teljesíti-e vagy sem. A modellellenőrzés kimenete (a felhasznált modellellenőrző eszköztől függően) a vizsgált követelmény nem teljesítése esetében egy ellenpélda. Az ellenpélda vizsgálatának eredményeképpen három lehetőség lehetséges: követelményhiba, modellezési hiba vagy tervezési hiba. Ha a modellellenőrzés eredménytelen, akkor egyéb technikát kell választani (pl. végtelen állapottér esetén), míg ha eredményes, akkor megfelelő eszköztámogatottság mellett lehetséges akár az automatikus kódgenerálás is.

3. ESETTANULMÁNY

Az esettanulmány keretében a BKV Zrt. [9] feltétfüzete által definiált utolérés kizáró rendszer, pontszerű foglaltság-érzékelési alrendszerének érzékelőelem objektumát használjuk fel. A 3. ábrán láthatjuk az utolérés kizáró rendszer főbb részeit egyvágányú pálya esetén (piros keretben az érzékelőelem).



3. ábra. Utolérés kizáró rendszer vázlata

Az utolérés kizáró rendszer célja, hogy megakadályozza a nem belátható pályarészeken az utoléréses balesetek bekövetkezését. A rendszer alapállapotában (nincs villamos a hatókörében) szabad jelzést ad a főjelzőjén (kétoptikás), az ellenőrzőjelző sötét (egy optikás). Ha a nyíllal jelölt közlekedési irányból egy villamos halad be a rendszer hatókörébe (a rendszer ezt a fekete háromszöggel jelzett érzékelőelemek segítségével érzékeli) a rendszer a főjelzőt vörösre kapcsolja, megtiltva ezzel a rendszer hatókörébe lépett villamost követő villamosok rendszerbe való behaladását. Ezt követően az egy optikából álló ellenőrzőjelzőn visszajelenti a rendszerbe érkezett villamosvezetőnek, hogy sikerül-e a főjelzőre kitennie a vörös fényt. Ha sikerült, akkor az ellenőrzőjelzőn sárga fénnyel világít. Ha a villamos kihalad a rendszer hatóköréből (érintve a piros kerettel jelölt érzékelőelemet), akkor a rendszer visszakerül alapállapotába.

3.1. Az érzékelőelem specifikációja

Megjegyzés: az itt közölt specifikáció a gyakorlati életből vett példa alapján készült egyszerűsített, nem teljes változat. Célja, hogy segítségével a modellezés célját (az érzékelőelem viselkedését megértési szintre fejlesszük, ill. a gyakorlati életben használt grafikus specifikációs elemekre (diagramokra) néhány példát mutassunk be. A példákat mintaként kell kezelni, nem feleltethetőek meg egy az egyben a valós leírásnak.

3.1.1. Az érzékelőelem célja

Az érzékelőelem célja a hatókörében tartózkodó vasúti jármű, vagy annak valamely részének detektálása.

3.1.2. Az érzékelőelem általános bemutatása

Megjegyzés: a soron következő leírásban, a zárójelben szereplő részek az érzékelőelem kimenetére vonatkoznak.

Az érzékelőelem alapállapotában (nem hibás, szabad) vasúti jármű nem tartózkodik a hatókörében. Ha az érzékelőelem hatókörébe vasúti jármű érkezik, akkor foglalttá válik (nem hibás, foglalt). Ha az érzékelőelem hatóköréből a vasúti jármű távozik, akkor az érzékelőelem felszabadul (nem hibás, szabad).

Az érzékelőelem foglalttá válásakor aktiválódik a TF foglaltsági időzítő.

Ha az érzékelőelem a foglalttá válását követően túl hamar (a valós körülményekhez képest „hihetetlenül” hamar) felszabadul, akkor az érzékelőelem hiba (hibás, foglalt) állapotba lép. A foglaltsági időnek a foglaltsági információ elfogadásához meg kell haladni a TAF alulfoglaltsági időt. A TAF értéke: 100 ms.

Ha az érzékelőelem a foglalttá válását követően túl sokáig (a valós körülményekhez képest „hihetetlenül” sokáig) nem szabadul fel, akkor az érzékelőelem hiba (hibás, foglalt) állapotba lép. A foglaltsági időnek a foglaltsági információ elfogadásához nem szabad meghaladnia a TTF túlfoglaltsági időt. A TTF értéke: 3000 ms.

Összefoglalva az érzékelőelem hatókörében közlekedő jármű által okozott érzékelőelem foglaltságot akkor tekintjük hihetőnek, ha TAF < TF < TTF összefüggés teljesül.

Az érzékelőelem foglaltsági bemenetét ponált/negált információ képezi. Ha a foglaltsági bemenetek valamely bemeneti esemény hatására megváltoznak, akkor egy TPN (P/N ellentmondás tolerancia) ideig a bemeneteken lévő azonos információtartalom megengedett. A TPN időn lejártát követően az érzékelőelem bemenete foglaltsági P/N ellentmondásos, melyek hatására az érzékelőelem hibás (hibás, foglalt) állapotba lép. A TPN értéke 200 ms.

Az érzékelőelem hiba bemenetét ponált/negált információ képezi. Ha a hibabemenetek közül bármely (legalább az egyik) nem hibás állapotról, hibás állapotra vált, az érzékelőelem azonnal hibás állapotba kerül (hibás, foglalt).

3.1.3. Konfigurálhatóság

Mindkét hibabemenet létezése konfigurálható. Ez pontosan 4 esetet jelent: egyik sem létezik, mindkettő létezik, egyik vagy másik létezik.

3.1.4. Interfész specifikáció

1. táblázat. Interfész specifikáció

Az.	Típus	Felvehető érték	Kezdőérték
FP	Bemenet	foglalt/szabad	-
FN	Bemenet	foglalt/szabad	-
HP	Bemenet	hibás/nem hibás	-
HN	Bemenet	hibás/nem hibás	-
F	Kimenet	foglalt/szabad	foglalt
H	Kimenet	hibás/nem hibás	hibás

3.1.5. Az érzékelőelem logikai viselkedése

2. táblázat. Logikai viselkedés

Áll. (régi)	Bemenet				Kimenet		Áll. (új)
B_H	BE_HP	BE_HN	BE_FP	BE_FN	KI_H	KI_F	B_H
nem hibás	hibás	-	-	-	hibás	foglalt	hibás
nem hibás	hibás	-	-	-	hibás	foglalt	hibás
nem hibás	-	hibás	-	-	hibás	foglalt	hibás
nem hibás	-	hibás	-	-	hibás	foglalt	hibás
nem hibás	nem hibás	nem hibás	szabad	foglalt	hibás	foglalt	hibás
nem hibás	nem hibás	nem hibás	szabad	foglalt	hibás	foglalt	hibás
nem hibás	nem hibás	nem hibás	foglalt	szabad	hibás	foglalt	hibás
nem hibás	nem hibás	nem hibás	foglalt	szabad	hibás	foglalt	hibás
nem hibás	nem hibás	nem hibás	foglalt	foglalt	nem hibás	foglalt	nem hibás
nem hibás	nem hibás	nem hibás	szabad	szabad	nem hibás	szabad	nem hibás
hibás	nem hibás	nem hibás	szabad	szabad	nem hibás	szabad	nem hibás

Megjegyzés: A táblázat nem tartalmazza a természetes nyelvű specifikációs leírásban definiált időzítéseket, azonban azokat is figyelembe kell venni. A táblázat a már statikusan beállt állapototokat és a rájuk vonatkozó reakciókat írja le, és nem foglalkozik a bemenet változásokat követő tolerancia idővel.

3.2. A kiindulási specifikáció értékelése

A 3.1. fejezetben felvázolt specifikációt és egyes elemeit a formalizáltság szintje és minősége alapján értékeljük.

A specifikáció vegyesen használ:

- formális leírást (pl. igazságtáblázat),
- félformális leírást (pl. táblázat),
- nem formális leírást (pl. természetes nyelvű leírás).

A rendszer teljes specifikációját is röviden értékeljük, mert a célunk, hogy a későbbiekben a rendszer teljes foglaltság-érzékelési alrendszerét modellezzük és modellellenőrizzük. A rendszer teljes specifikációja vegyesen használ:

- formális leírást (pl. igazságtáblázat),
- félformális leírást (pl. állapottérkép, idődiagram),
- nem formális leírást (pl. természetes nyelvű leírás).

Összefoglalva a rendszer teljes specifikációjának jellemzőit, a specifikációt véges számú elemkészlet jellemzi. A véges számú elemkészlet legtöbb tagja megfelel az UML szabványnak [10].

A 3.1. fejezetben bemutatott specifikáció minőségének értékelése néhány példán keresztül:

- nem egyértelmű (pl. 2. táblázathoz nincs magyarázat mellékelve, így az félreértelmezhető lehet, főleg a P/N információk kapcsán),
- bizonyíthatóan nem teljes (pl. pl. 2. táblázat nem a teljes igazságtáblázat, hanem egy egyszerűsített változata, csak a táblázat visszafejtésével látható be, hogy minden érvényes és érvénytelen bementre van specifikálva viselkedés; vagy a 3.1.5. fejezetben lévő megjegyzés ugyan utal az időzítések kezelésére, de pontosan nem adja meg a hozzájuk kapcsolódó viselkedéseket, azokra valamennyire 3.1.2. szöveges leírásból következtethetünk),
- inkonzisztens (pl. az 1. táblázatban definiált bemenet azonosítókat a 2 táblázatban nem ugyanazon a néven adja meg),
- ellenőrizhetősége nem minden esetben egyértelmű (pl. a nem formális elemek használata ezt jelentősen megnehezíti),
- módosíthatóság (pl. a követelmények redundánsan jelennek meg 3.1.2. szöveges leírásban és 3.1.5. fejezetben lévő logikai viselkedés leírásban, módosíthatóság szempontjából ez hordozhat veszélyeket, amennyiben az egyik rész nem javítják vissza megfelelően)
- követhetőség (pl. nem derül ki, hogy az érzékelőelem hol helyezkedik el a foglaltsági alrendszeren belül, milyen objektumokkal áll kapcsolatban, milyen hatások érik, milyen hatásokat okozhat; vagy nincsenek hivatkozások).

4. A RÉSZLETES SPECIFIKÁCIÓ

A 3.2. fejezet alapján megállapítottuk, hogy ilyen specifikációs tulajdonságok mellett nem érdemes azonnal a formális modellt készíteni. Hogy ezeket a specifikációs problémákat kiküszöböljük, először egy részletes specifikálást hajtunk végre, melynek során a specifikáció egyes elemei (melyek esetében ez lehetséges) formális leírást kapnak, illetve 3.2. fejezetben felsorolt minőségi tulajdonságok javulnak.

Mivel a teljes érzékelőelem vonatkozásában ez rendkívül terjedelmes lenne, ezért csak egy kiragadott részén mutatjuk be a 4. ábrán felvázolt folyamatot. A kiragadott specifikációs részlet a 2. táblázat szerinti logikai viselkedés (vagyis az igazságtáblázat). A specifikáció többi elemének részletes specifikációjára és a formális modellbe történő beépítésére csak utalásokat teszünk.

4. ábra. A formális modell előkészítésének folyamata

4.1. Jelölésrendszer bevezetése

A részletes specifikálás kezdeti fázisában bevezettünk egy egységes jelölésrendszert, amihez a kiindulási specifikáció kellően jó alapot adott. A jelölésrendszer igazságtáblázatra vonatkozó részletét a 3. táblázatban foglaltuk össze.

3. táblázat. Jelölésrendszer (igazságtábla értelmezéséhez)

Jelölés	Magyarázat	Lehetséges értékek			
BE_HP	Ponált hiba bemenet	0	nem hibás	1	hibás
BE_HN	Negált hiba bemenet	0	hibás	1	nem hibás
BE_FP	Ponált foglaltsági bemenet	0	szabad	1	foglalt
BE_FN	Negált foglaltsági bemenet	0	foglalt	1	szabad
B_H	Hiba - belső állapot	0	nem hibás	1	hibás
KI_H	Hiba kimenet	0	nem hibás	1	hibás
KI_F	Foglaltsági kimenet	0	szabad	1	foglalt
		-	Közömbös		

4.2. Az igazságtáblázat konstruálása

A 3. táblázat jelöléseivel képzett igazságtáblázatot a 4. táblázatban láthatjuk.

Megjegyzés: A kimenetek és az új belső állapot képzéséhez szükséges szabályokat itt nem részletezzük.

4. táblázat. Az érzékelőelem igazságtáblázata

Ssz.	Igazságtábla							
	Állapot (régi)	Bemenet				Kimenet		Állapot (új)
	B_H	BE_HP	BE_HN	BE_FP	BE_FN	KI_H	KI_F	B_H
0	0	0	0	0	0	1	1	1
1	0	0	0	0	1	1	1	1
2	0	0	0	1	0	1	1	1
3	0	0	0	1	1	1	1	1
4	0	0	1	0	0	1	1	1
5	0	0	1	0	1	0	0	0
6	0	0	1	1	0	0	1	0
7	0	0	1	1	1	1	1	1
8	0	1	0	0	0	1	1	1
9	0	1	0	0	1	1	1	1
10	0	1	0	1	0	1	1	1
11	0	1	0	1	1	1	1	1
12	0	1	1	0	0	1	1	1
13	0	1	1	0	1	1	1	1
14	0	1	1	1	0	1	1	1
15	0	1	1	1	1	1	1	1
16	1	0	0	0	0	1	1	1
17	1	0	0	0	1	1	1	1
18	1	0	0	1	0	1	1	1
19	1	0	0	1	1	1	1	1
20	1	0	1	0	0	1	1	1
21	1	0	1	0	1	0	0	0
22	1	0	1	1	0	1	1	1
23	1	0	1	1	1	1	1	1
24	1	1	0	0	0	1	1	1
25	1	1	0	0	1	1	1	1
26	1	1	0	1	0	1	1	1
27	1	1	0	1	1	1	1	1
28	1	1	1	0	0	1	1	1
29	1	1	1	0	1	1	1	1
30	1	1	1	1	0	1	1	1
31	1	1	1	1	1	1	1	1

4.3. Az igazságtáblával megadott logikai függvény egyszerűsítése

A további lépések ismertetése előtt megjegyezzük, hogy egy szoftver architektúrális döntés hatással volt a következőkben ismertetett tervezés folyamatra. Ez a döntés a tervezett szoftveres megvalósításra vonatkozott, miszerint szinkron sorrendi hálózat alapelveinek megfelelően fog működni a szoftverrendszer. Ebből következik pl. az is, hogy nem kell számolnunk versenyhelyzetekkel.

Az érzékelőelem viselkedésének meghatározása így a digitális technikában jól ismert szinkron sorrendi hálózat tervezési feladatra vezethető vissza [12]. Az érzékelőelemnek van kezdeti állapota (melyet a kiindulási specifikáció alapján feltételeztünk). Az érzékelőelem belső állapota képviseli az ún. szekunder kombinációt, amely a rendszer előéltéről hordoz információt. Az érzékelőelem tervezését szinkron sorrendi hálózattal valósítottuk meg, a Mealy kimeneti modell alkalmazásával.

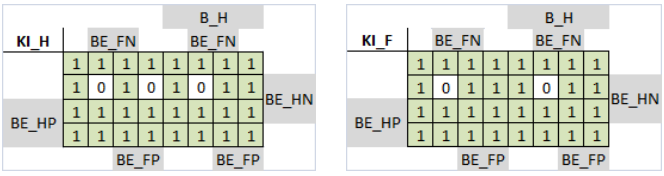
Az érzékelőelem kódolt állapottábláját (5. táblázat) a megadott igazságtáblázat alapján azonnal konstruálhatjuk.

5. táblázat. Az érzékelőelem kódolt állapottáblája

BE_ / B_H	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0	1/11	1/11	1/11	1/11	1/11	0/00	0/10	1/11	1/11	1/11	1/11	1/11	1/11	1/11	1/11	1/11
1	1/11	1/11	1/11	1/11	1/11	0/00	1/11	1/11	1/11	1/11	1/11	1/11	1/11	1/11	1/11	1/11

A 5. táblázatban láthatjuk, hogy állapotok összevonására nincs lehetőség, a specifikáció formalizáltsága miatt azonnal felírhattuk akár a kódolt állapottáblát (az előzetes és összevont állapottáblák lépéseket kihagyva).

A kódolt állapottábla alapján adódnak a kimeneti függvények, melyeket a Karnaugh-táblák felhasználásával írtunk fel mindkét kimenetre (lásd 5. ábra).



5. ábra. Az igazságtáblák Karnaugh-táblára való leképezése

A Karnaugh-táblák alapján felírt kimeneti függvények és negáltjaik (minterm alak):

KI_H = BE_HP + ⌐BE_HN + BE_FN × BE_FP + ⌐BE_FN × ⌐BE_FP + B_H × BE_FP

⌐KI_H = ⌐BE_HP × BE_HN × ⌐BE_FP × BE_FN + ⌐BE_HP × BE_HN × ⌐B_H × ⌐BE_FN × BE_FP

KI_F = BE_HP + ⌐BE_HN + BE_FP + ⌐BE_FN × ⌐BE_FP

⌐KI_F = ⌐BE_HP × BE_HN × ⌐BE_FP × BE_FN

Megjegyzés: KI_H = B_H és ⌐KI_H = ⌐B_H

Ezt követően a kimeneti függvényeket felhasználva újra felírtuk az immáron egyszerűsített igazságtáblázatot. A felírás lépéseit nem részletezve az egyszerűsített táblázatot a 6. táblázatban foglaltuk össze. Ez a táblázat képezi az alapját a Petri-hálós formális modell generálásának. A generálás lépéseit és a vonatkozó szabályokat az 5. fejezetben foglaltuk össze.

6. táblázat. Az egyszerűsített igazságtáblázat

Ssz.	Igazságtábla							
	Állapot (régi)	Bemenet				Kimenet		Állapot (új)
	B_H	BE_HP	BE_HN	BE_FP	BE_FN	KI_F	KI_H	B_H
0	-	1	-	-	-	1	1	1
1	-	-	0	-	-	1	1	1
2	-	-	-	1	1	1	1	1
3	-	-	-	0	0	1	1	1
4	1	-	-	1	-	1	1	1
5	-	0	1	0	1	0	0	0
6	0	0	1	1	0	1	0	0

Az összevont igazságtáblázat helyességét annak visszafejtésével igazoltuk.

5. A FORMÁLIS MODELL GENERÁLÁSA ÉS VIZSGÁLATA

A formális modell elkészítéséhez a vasúti biztosító-berendezési területen régóta alkalmazott Petri-hálókat [11] használtuk fel. A modell elkészítéséhez az egyszerűsített kimeneti függvényekből indultunk ki a jelen fejezetben leírt lépéseket végrehajtva. Célunk egyszerű Petri-háló (kiterjesztésektől mentes Petri-háló) készítése volt elősegítendő az automatizált Petri-háló generálását, ill. a Petri-háló modell elemezhetőségét, hordozhatóságát.

Felhasználva 6. táblázatot (lecserélve annak fejlécét) a 7. táblázatban látható hozzárendelés végezhető el. A 7. táblázat képezi az alapját a Petri-háló elemek meghatározásának.

7. táblázat. Az igazságtáblázat Petri-hálóra való leképezése

Ssz.	Hely								Tranzíció
	Előfeltételek					Utófeltételek			
	B_H	BE_HP	BE_HN	BE_FP	BE_FN	KI_F	KI_H	B_H	
0	-	1	-	-	-	1	1	1	T1
1	-	-	0	-	-	1	1	1	T2
2	-	-	-	1	1	1	1	1	T3
3	-	-	-	0	0	1	1	1	T4
4	1	-	-	1	-	1	1	1	T5
5	-	0	1	0	1	0	0	0	T6
6	0	0	1	1	0	1	0	0	T7

A következő fejezetekben részletesen tárgyaljuk a Petri-háló generálásához szükséges szabályokat és lépésről lépésre bemutatjuk a háló felépítését. A Petri-háló konstruálásához a PDN alkalmazást használtuk fel [13], mert alkalmas modellellenőrzésre.

5.1. A Petri-háló helyeinek leképezése

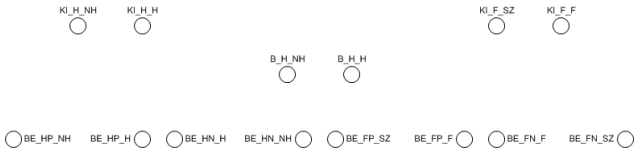
A Petri-háló helyei az igazságtábla oszlopaiból származtathatóak (bemenetek, kimenetek és belső állapot). Ezeket a 7. táblázat, mint elő- és utófeltételek nevesítettük. Minden bemenethez és kimenethez az állapotainak megfelelő számú helyet kell rendelni, vagyis a 4 bemenetből és a 2 kimenetből összesen 12 helyet kell létrehozni, míg az 1 belső állapotból 2 helyet. Erre azért van szükség, mert vizsgálva a 7. táblázat oszlopait megállapítottuk, hogy az egyes feltételek mindkét állapában való tartózkodásáról szükségünk van információra. Ezt az információt a tokenek hordozzák olyan módon, hogy az adott helyen a Petri-háló valamely állapotában van token avagy nincs. A tokenszám vonatkozásában triviális szabály az egyes feltételek vonatkozásában, hogy a feltételhez tartozó helypárnak egyidejűleg csak az egyik helyén lehet token. (Ilyen módon a háló korlátossága is biztosított lesz, természetesen ez függvénye az élek helyes szerkesztésének is.)

Mivel a helyek száma megkétszereződött, be kell vezetnünk egy jelölésrendszer, amely a 8. táblázatban lévő helyekből leszármaztatott új helyeket jelölésére is alkalmas (lásd 8. táblázat). Összhangban a 3. táblázattal NH jelöli a nem hibás, H a hibás, F a foglalt, SZ pedig a szabad állapotot.

8. táblázat. A Petri-háló helyeinek jelölésrendszere

Bemenet				Állapot	Kimenet	
BE_HP	BE_HN	BE_FP	BE_FN	B_H	KI_H	KI_F
BE_HP_NH	BE_HN_NH	BE_FP_SZ	BE_FN_SZ	B_H_NH	KI_H_NH	KI_F_SZ
BE_HP_H	BE_HN_H	BE_FP_F	BE_FN_F	B_H_H	KI_H_H	KI_F_F

A 6. ábrán láthatjuk az eddigi információk alapján megszerkesztett Petri-hálót részletet.



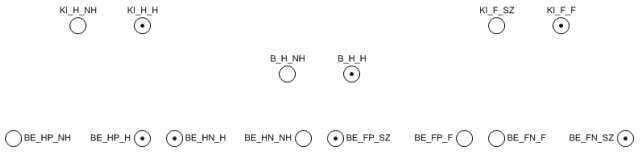
6. ábra. Petri-háló modell, a helyek felvétele

A helyeket a további lépésekben összevontan tartjuk, tehát nem használjuk (_NH, _H _SZ és _F utótagokat).

5.2. A kezdeti tokeneloszlás meghatározása

A kezdeti tokeneloszlás meghatározása a kiindulási specifikáció 1. táblázata alapján lehetséges. Megjegyezzük, hogy a kiindulási specifikáció nem ad információt a belső állapot kezdőállapotáról, de arra kimenetek alapján vissza lehet következtetni: a belső állapot hibás.

A 7. ábrán láthatjuk a felvett kezdeti token eloszlást. Megjegyzés, a bemeneti helyek állapota tetszőleges lehet, a belső állapot és a kimenetek állapota viszont jól definiált.



7. ábra. Petri-háló modell, a kezdeti tokeneloszlás

5.3. A Petri-háló tranzícióinak leképezése

A 7. táblázatban már jeleztük, hogy a tranzíciók (T1..T7) sorokhoz fognak kötődni. A tranzíciók leképezése azonban nem triviális. Két vizsgálatot kell végrehajtani adott sorrendben:

1. Hatástalanság-vizsgálat
2. Lehetséges aktuális állapot vizsgálat

Mindkét vizsgálat Petri-háló szerkesztési sajátosságokhoz köthető. Az alapproblémák felvetése után mindkét vizsgálatot egy-egy példán keresztül szemléltetjük.

Online: ISBN 978-963-88875-3-5
CD: ISBN 978-963-88875-2-8

Editor: Dr. Péter Tamás

A *hatástalanság-vizsgálat* célja megtalálni a már egyszerűsített igazságtáblában azokat a belső állapotokat, melyek a kimenetre nincsenek hatással (bárhogy változnak is mellettük a bemenetek). Fogalmazhatunk úgy is, hogy a hatástalanság-vizsgálattal kiszűrjük azokat az állapotátmeneteket melyek kiindulási és célállapota megegyezik, és az állapotátlépés közben a rendszeren semmilyen hatást nem gyakorolnak, így viselkedésük modellezésére nincs szükség. A hatástalanság-vizsgálat végrehajtása tranzíció szám csökkenést okoz a Petri-háló modellben.

Példa1: Tekintsük a 7. táblázat 0. sorát. A 0. sor jelentése természetes nyelven megfogalmazva: Ha a ponált hibabemenet aktív, akkor az érzékelőelem új belső állapota hibás, kimenete pedig hibás és foglalt lesz. Ha a ponált hibabemenet aktív lesz (BE_HP = 1, vagyis BE_HP_H helyen van egy token), akkor T1 tranzíció engedélyezetté válik és tüzelhet. Tüzelésekor tokent kell tennie a KI_F_F, KI_H_H és B_H_H helyekre, vagyis az érzékelőelem hibás állapotba kerül. Ez az eset csak az érzékelőelem nem hibás belső állapota esetén igaz. Amennyiben az érzékelőelem hibás belső állapota esetén történik mindez, akkor a belső állapot és a kimenetek már az elvártnak megfelelő állapotban vannak, így azok állapotát nem szükséges módosítani. Ez igaz az 7. táblázat első sorában minden ’–’ jelölt elemre (és a 4. sorra is), kivéve az 5. sort. Az 5. sorban egyszerre két funkció van elrejtve az oldás és a szabadulás. Hibás belső állapot esetén oldásra, nem hibás belső állapot esetén pedig szabadulásra van szükség az alapállapot eléréséhez. Mindkét eset hatásos állapotátlépést eredményez. Megegyezzük, hogy hatásos állapotátlépésnek tekintjük azt is, ha a belső állapot nem változik meg, viszont a kimenet módosul (pl. foglaltról szabadra).

A *lehetséges aktuális állapot vizsgálat* célja, hogy megkeressük (ha léteznek) azokat a kimeneteket, melyek aktuális állapota befolyásolhatja az előálló új kimenetet és új belső állapotot. (Az aktuális belső állapotot már az eddigiekben is figyelembe vettük, ezért azt külön nem kell vizsgálni.) A lehetséges aktuális állapot vizsgálat a kimeneti kombinációk felvételéből és egy hatástalanság-vizsgálatból áll. A lehetséges aktuális állapot vizsgálat tranzíció szám növekedést okoz a Petri-háló modellben.

Megjegyzés: A lehetséges aktuális állapot vizsgálat analóg a Sequential Circuit Test Generation elméletében ismert Time Frame Expansion technikával [14]. Egy időbeli kihajtogatást jelent, melyet jelen esettanulmányban elegendő volt csak az 1. szintig (következő állapotra) kiterjeszteni.

Példa2: Tekintsük 7. táblázat 0. sorát. (A 0. sor értelmezését Példa1-ben megadtuk.) Tételezzük fel, hogy már elvégeztük a hatástalanság-vizsgálatot, aminek megfelelően a 7. táblázat 0. sorában a B_H oszlop alatt már csak egy 0 érték szerepel. Ez azt jelenti, hogy csak nem hibás belső állapot esetén kerülhet a rendszer hibás belső állapotba. Ekkor a kimenetet hibásra és foglaltra, az új belső állapotban pedig hibásra kell állítani, vagyis az utófeltételeket a táblázatnak megfelelő állapotba kell hozni. Tekintsük a KI_F kimenetet. Két lehetőség adódik:

– KI_F foglalt,
– KI_F szabad.

Ha KI_F foglalt esetén következik be a 7. táblázat 0. sorában szerepeltet hibajelenség, akkor a foglaltsági kimenetet esetében a hatástalanság-vizsgálathoz hasonló eset történik, ugyanis a foglalt állapotban lévő kimenetet már nem kell újra foglaltba állítani. Ha KI_F kimenet szabad, akkor viszont KI_F kimenetet foglaltra kell állítani. Ez két különböző esetnek számít, ami Petri-háló modell esetében úgy képezhető le, hogy a 7. táblázat 0. sorában lévő előfeltételt felbővítjük az aktuális állapotban fennálló kimenet állapotokkal.

A példában leírt vizsgálatok elvégzését követően előállítható a 9. táblázat, melyben megjelenítettük a releváns aktuális kimenteket is. Megjegyzés: a lehetséges aktuális állapot vizsgálat a hibakimenet jelen példában nem érintette. (Nem lehetséges olyan eset, hogy a kimenet hibás legyen akkor, amikor a belső állapot nem hibás.) A 9. táblázat a vizsgálatok eredményeit ismerteti, a folyamatát nem (valamint csak a modellezésre használt sorok és oszlopok szerepelnek benne).

9. táblázat. Az igazságtáblázat Petri-hálóra való leképezésének pontosítása

Hely										
Előfeltételek							Utófeltételek			
KI_F (t)	KI_H (t)	B_H (t)	BE_HP (t+1)	BE_HN (t+1)	BE_FP (t+1)	BE_FN (t+1)	KI_F (t+1)	KI_H (t+1)	B_H (t+1)	Tranzíció
0	0	0	1	-	-	-	1	1	1	PHIB1
1	0	0	1	-	-	-	1	1	1	PHIB2
0	0	0	-	0	-	-	1	1	1	NHIB1
1	0	0	-	0	-	-	1	1	1	NHIB2
0	0	0	-	-	1	1	1	1	1	APNHIB1
1	0	0	-	-	1	1	1	1	1	APNHIB2
0	0	0	-	-	0	0	1	1	1	BPNHIB1
1	0	0	-	-	0	0	1	1	1	BPNHIB2
1	0	0	0	1	0	1	0	0	0	SZAB
1	1	1	0	1	0	1	0	0	0	OLD
0	0	0	0	1	1	0	1	0	0	FOGL

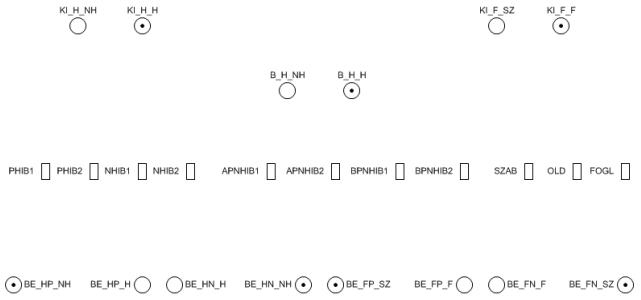
Megjegyezzük, hogy jelen esettanulmány nem adott lehetőséget több belső változóval rendelkező objektum vizsgálatára, azonban az automatizált Petri-háló generáláshoz szükséges szabályok konstruálásához – az eljárás általánosítása céljából – ez a téma további vizsgálatokat igényel.

A tranzíciók elnevezése a hozzájuk kapcsolódó funkcionalitás alapján történt. A tranzícók végén szereplő szám az aktuális állapotra utal vissza. A rövidítések természetes nyelvű megfogalmazása:

– PHIB: ponált hiba bemenet hiba
– NHIB: negált hiba bemenet hiba

- PNHIB: a foglaltsági bemenet P/N ellentmondást (kétféle hiba lehet, B: 00 és A: 11)
- SZAB: érzékelőelem felszabadul (nem hibás állapotban)
- OLD: az érzékelőelem oldódik (hibás állapotból)
- FOGL: az érzékelőelem foglalt lesz (nem hibás állapotban)

A 8. ábrán láthatjuk az eddigi információk alapján megszerkesztett Petri-hálót részletet.



8. ábra. Petri-háló modell, tranzíciók felvétele

5.4. A Petri-háló éleinek leképezése

A Petri-háló éleit 9. táblázat eddig még nem felhasznált részeiből képezhetjük le. A leképezés lényege, hogy meghatározzuk a szomszédossági mátrixot (W ill. W^- és W^+).

A W^- mátrix határozza meg azokat az éleket (az élek számát/élsúlyt), melyek egy tetszőlegesen kiválasztott tranzíció a hálóban szereplő helyek felől a tranzíció felé mutatnak (vagyis azt határozza meg, hogy az adott tranzíció az egyes helyekről mennyi tokent vesz el. Ha nem vesz el tokent egy adott helyről, azt a W^- mátrixban 0-val jelöljük, ekkor él sem létezik. A W^+ mátrix esetében hasonló a helyzet, azonban ott azt határozzuk meg, hogy az egyes tranzíciók mennyi tokent szállítanak az egyes helyekre (a token szállításához él szükséges).

Kéttípusú élet kell megkülönböztetnünk:

- az egyik az érzékelőelem állapotának (pl. bemenetek) ellenőrzésére szolgál,
- a másik pedig az érzékelőelem állapotának ill. kimeneteinek a megváltoztatását végzi.

A két éltípus közti különbség, hogy előbbi „nem okoz” token mozgást. Ezeket teszt vagy ellenőrző él néven is emlegetik [15].

A két éltípus egyértelműen azonosítható a 9. táblázatban, ezt követően 3. táblázat szerinti értelmezések figyelembevételével hozhatóak létre W^- és W^+ mátrixok:

- A 9. táblázat bemenetre vonatkozó előfeltételei (BE_ kezdetű oszlopok) esetében minden olyan helyre kell létesíteni élpárt, amely az adott sor tranzíciója és a bemeneti hely 3. táblázat szerinti, azaz 0 vagy 1 értelemben vett állapota között van. Egyszerű Petri-háló révén nem teszt élet, hanem egy-egy a hely és a tranzíció közt lévő ellentétes irányú élet hozunk létre), amely teszt élként funkcionál. Megjegyzés: ezek az élek a szomszédossági mátrixban nem fognak megjelenni (csak W^- és W^+ mátrixokban). Ezzel az adott tranzícióra vonatkozóan a releváns aktuális bemenetek (előfeltételek kerülnek meghatározásra.)
- A belső állapotok és kimenetek helyeivel kapcsolatos éleket a 9. táblázat alapján úgy kell meghatározni, hogy vizsgálni kell az előző (értsd t-edik) ill. az új (értsd t+1-edik) belső állapotot ill. kimenetet. Amennyiben t-hez képest t+1 megváltozott (pl. t-ben 0 t+1 pedig 1 állapotváltás történt), akkor ez azt jelenti, hogy az éppen vizsgált tranzíció az adott helyhez dedikált két hely között tokent léptett át egyik helyről a másikra (vagyis állapotváltás történt).

A szabályok felhasználásával azonnal képezhető W^- és W^+ mátrixok, melyekből előállítható a szomszédossági mátrix is ($W = W^+ - W^-$).

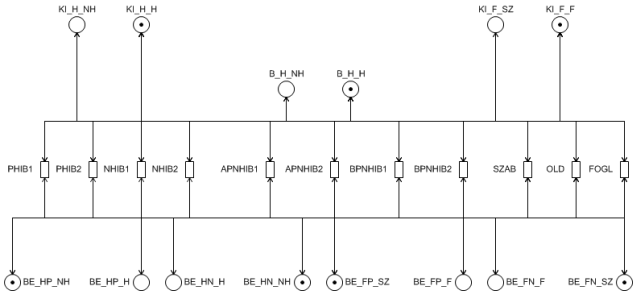
Terjedelmi korlátok miatt csak a szomszédossági mátrixot közöljük, az egyértelműség kedvéért táblázatos formában (jelölve a tranzíciókat és a helyeket is)

10. táblázat. Szomszédossági mátrix

W	BE_HP_NH	BE_HP_H	BE_HN_NH	BE_HN_H	BE_FP_SZ	BE_FP_F	BE_FN_SZ	BE_FN_F	KI_F_SZ	KI_F_F	KI_H_NH	KI_H_H	B_H_NH	B_H_H
PHIB1	0	0	0	0	0	0	0	0	-1	1	-1	1	-1	1
PHIB2	0	0	0	0	0	0	0	0	0	0	-1	1	-1	1
NHIB1	0	0	0	0	0	0	0	0	-1	1	-1	1	-1	1
NHIB2	0	0	0	0	0	0	0	0	0	0	-1	1	-1	1
APNHIB1	0	0	0	0	0	0	0	0	-1	1	-1	1	-1	1
APNHIB2	0	0	0	0	0	0	0	0	0	0	-1	1	-1	1
BPNHIB1	0	0	0	0	0	0	0	0	-1	1	-1	1	-1	1
BPNHIB2	0	0	0	0	0	0	0	0	0	0	-1	1	-1	1
SZAB	0	0	0	0	0	0	0	0	1	-1	0	0	0	0
OLD	0	0	0	0	0	0	0	0	1	-1	1	-1	1	-1
FOGL	0	0	0	0	0	0	0	0	-1	1	0	0	0	0

Az élek generálása azonban W^- és W^+ mátrixok felhasználásával célszerű, mivel a szomszédossági mátrix elrejti pl. a teszt éleket. Az érzékelőelem konstruált Petri háló modelljét a 9. ábrán láthatjuk.

Megjegyezzük, hogy elvesztettük a Petri-hálók egy előnyös tulajdonságát, az olvashatóságot és értelmezhetőséget. Ez nem is volt célunk, hiszen automatizált Petri-háló generálás esetében ennél az esettanulmánynál csak jóval bonyolultabb példákra számítunk, így célként nem az átláthatóság, hanem az elemezhetőség kerül előtérbe.

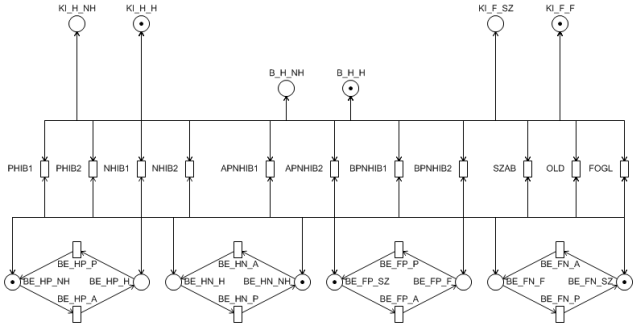


9. ábra. Az érzékelőelem Petri-háló modellje

5.5. A generált Petri-háló bemeneteinek gerjesztése

A 9. ábrán lévő Petri háló elérhetőségi gráfja alapján holtpontos (mindössze két állapota van). Ennek oka, hogy az OLD tranzíció tüzelését követően a bemenetek nem képesek állapotot váltani. (Az OLD tranzíció is csak azért tüzelhet, mert a háló kezdeti állapota teljesíti a tüzelési feltételeket.)

EA holtpont kiküszöbölhető megfelelő bemeneti gerjesztéssel, melyet a modellezési cél függvényében akár több alternatív módon is megvalósíthatunk. Mi most az egyik legegyszerűbb megoldást mutatjuk be (10. ábra)



10. ábra. Az érzékelőelem Petri-háló modellje

Minden bemenetet képező helypár közé egy-egy tranzíciót tettünk (_P végződés jelzi a bemenet aktív >> inaktív átmenetét, _A végződés jelzi a bemenet passzív >> aktív átmenetét).

A háló dinamikus tulajdonságait ilyen módon már van értelme vizsgálni.

Megjegyezzük, hogy célszerű lehet a bemeneteket (és a kimeneteket is) alhálóba szervezni (hierarchikus Petri-háló építése). Ezzel azonban nem foglalkoztunk, mert nem minden Petri-háló elemzésre képes eszköz tudja kezelni a hierarchikus Petri-hálókat.

A 10. ábrán bemutatott Petri-háló modell már alkalmas a tokenjáték szimulálására is.

5.6. A generált Petri-háló tulajdonságainak összefoglalása

Egy Petri-háló elemzésekor általában a statikus és dinamikus tulajdonságokat értékelik. A teljeskörű elemzés bemutatását itt nem célozzuk meg, néhány fontosabb tulajdonságot emelünk ki a PDN eszköz felhasználásával, összefoglaló jelleggel.

5.6.1. Statikus tulajdonságok

- Helyek száma: 14 db
- Tranzíciók száma: 19 db
- Élek száma 130 db
- Tokenszám: 7 db (megjegyzés: állandó)
- p-invariánsok száma: 13 db
  pl.: {1 × BE_FP_SZ, 1 × BE_FP_F}
- t-invariánsok száma: 26 db
  pl.: {PHIB1, OLD}

5.6.2. Dinamikus tulajdonságok

A dinamikus tulajdonságok az elérhetőségi gráf vizsgálata az elérhetőségi gráf képezi az alapját. (A gráfot mérete és olvashatatlansága miatt nem közöljük.)

Összefoglalva az érzékelőelem Petri-hálójának néhány dinamikus tulajdonsága:

- Állapotok száma: 48 db,
- Korlátosság: 1 korlátos (biztos),
- Holtpontmentes,
- Megfordítható,
- Nem tiszta.

5.7. A kiindulási specifikáció további elemeinek modellezési lehetőségei

A kiindulási specifikáció 3.1. fejezetben részletezett kiindulási specifikáció még számos modellezhető elemet tartalmaz (pl. konfigurálhatóság, időzítések, stb.). Ebben a fejezetben két példát mutatunk be ezek viselkedés-modellbe való beépítési lehetőségeire.

5.7.1. Időzítések modellezése

Az időzítések modellezéséhez célszerű felhasználni az időzített Petri-hálókat [16]. (Megjegyezzük, hogy az időzítés megoldató egyszerű Petri-hálóval is.)

Az időzítések kétféleképpen építhetőek be a modellben:

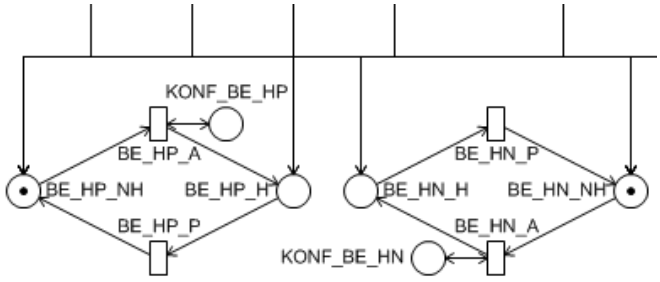
- a megfelelő, már meglévő tranzíció felhasználásával,
- új tranzíció definiálásával (erre akkor van szükség, ha nincs olyan tranzíció a meglévő Petri hálóban, melynek elő- és utófeltételi megegyeznek az időzítés elő- és utófeltételivel).

Előbbire példa a kiindulási specifikációból a 3.1.2. fejezetéből a TPN időzítés, míg utóbbira példák a TAF és TTF időzítések.

Megjegyezzük azonban, hogy ezen terület még kutatás alatt van, így jelen fejezetben közölt információk még kiegészülhetnek.

5.7.2. Konfigurációs lehetőségek modellezése

Az konfigurációs lehetőségek közül a kiindulási specifikáció egy egyszerű példát hoz, a hibabemenetek létezésének konfigurálhatóságát (3.1.3. fejezet). A kiindulási specifikáció által említett 4 konfigurálási lehetőség megvalósítható bemenetenként 2 újabb helyek és a helyek felől a megfelelő tranzíció felé- és visszahúzott élekkel (tehát egy teszt éllel). A helyekre egy-egy tokent helyezve lehet beállítani a bemenet létezését (valójában a token nem létezése miatt sosem lehet aktív egyik hibabemenet sem). A 11. ábrán egy olyan konfigurációs esetet láthatunk, ahol egyik hibabement sem létezik.



11. ábra. Az érzékelőelem Petri-háló modelljének konfigurálhatósága

6. ÖSSZEFOGLALÁS

Jelen írásunkban bemutattunk a vasúti biztosítóberendezések fejlesztésben használt formális specifikáció egy részletének formális modellé történő transzformálását. Egy célszerű esettanulmányból kiragadott példák segítségével ismertettük a transzformáció lépéseit. Kutatásunk, hogy létrehozzunk egy olyan szakterületi specifikus platformot, amely megteremeti a lehetőséget a vasúti mérnökök számára a formális módszerek matematika leírásait elrejtve a rendszer tulajdonságainak és viselkedésének ellenőrzését. Jelen írásunkban olyan tapasztalatokat gyűjtöttünk össze, melyek jelentősen befolyásolhatják a platform kialakítását.

7. HIVATKOZÁSOK

[1] Sághi B. Formális módszerek alkalmazása a vasútbiztosító technikában, PhD értekezés, Budapest, 2003.

[2] A. v. Lamsweerde: Formal Specification: a Roadmap, 2000

[3] A. Bonacchi, et al. Validation of Railway Interlocking Systems by Formal Verification, A Case Study, Internal Conference on Software Engineering and Fornmal Methods, pp 237-252, 2013.

[4] Pataricza A., et al. Formális módszerek az informatikában, ISBN 978-963-9548-90-9, 2006.

[5] Ésik Z. [et al.] Hardver- és szoftverrendszerek verifikációja, Typotex, 2011, ISBN 978-963-279-497-6, http://www.tankonyvtar.hu/hu/tartalom/tamop425/0008_esikgombasnemeth/Esik_Gombas_Nemeth_Hardver_1_1.html

[6] Farkas B., Lukács G., Dr. Bartha T. Formális modellezés alkalmazásának lehetőségei a vasúti biztosítóberendezések területén – 1. rész, Vasúti vezetékvilág XXII. 2017/2

[7] Farkas B., Lukács G., Dr. Bartha T. Formális modellezés alkalmazásának lehetőségei a vasúti biztosítóberendezések területén – 2. rész, Vasúti vezetékvilág XXII. 2017/3

[8] MSZ EN 50128:2011 Vasúti alkalmazások. Távközlési, biztosítóberendezési és adatfeldolgozó rendszerek. Szoftverek vasúti vezérlő- és védelmi rendszerekhez

[9] Feltétfüzet közúti vasutak (villamosok) forgalomirányításához szükséges biztonságtechnikai elemek és berendezések számára, 2011.

[10] Unified Modeling Language, http://www.omg.org/spec/UML/

[11] MURATA, Tadao Petri- Nets: Properties, Analysis and Applications, *Proceedings of the IEEE*. Vol. 77, No. 4. (1989) pp. 542-559.

[12] Tarnai G., et al. Irányítástechnika I. Egyetemi tananyag, 2011.

[13] VÖRÖS András [et al.] PetriDotNet 1.5 User Manual. Budapest, BME MIT. https://inf.mit.bme.hu/research/tools/petridotnet

[14] Mohammadi, Sequential Circuit ATPG, Time-Fram Expansion, Lecture 13, 2001.

[15] Hybrid functional Petri net includes hybrid Petri net, hybrid dynamic net, and functional Petri net, genome.ib.sci.yamaguchi-u.ac.jp/~gon/HFPNincludes.htm

[16] Bartha T., Pataricza A. Petri hálók: alapfogalmak, kiterjesztések, BME, MIT előadásvázlat