**Formális módszerek**

**Webshopban történő vásárlás folyamatának szemléltetése UPPAAL-ban**

**Turcsa Zoltán**

**QTHHWZ**

Tartalom

[A feladat rövid bemutatása 3](#_Toc503294033)

[Az automata template-jei 3](#_Toc503294034)

[Vásárló template 3](#_Toc503294035)

[Kosár template 4](#_Toc503294036)

[Szallito template 4](#_Toc503294037)

[Termek templtate 4](#_Toc503294038)

[Globális csatornák és változók 5](#_Toc503294039)

[Egy lehetséges szimuláció 6](#_Toc503294040)

# A feladat rövid bemutatása

A Formális módszerek tárgyhoz készült beadandó feladatomban egy webshopban történő vásárlás vázaltos folyamatát szeretném bemutatni automatával.

A folyamatban érintett résztvevők az alábbiak:

* vásárló
* szállító
* a termék

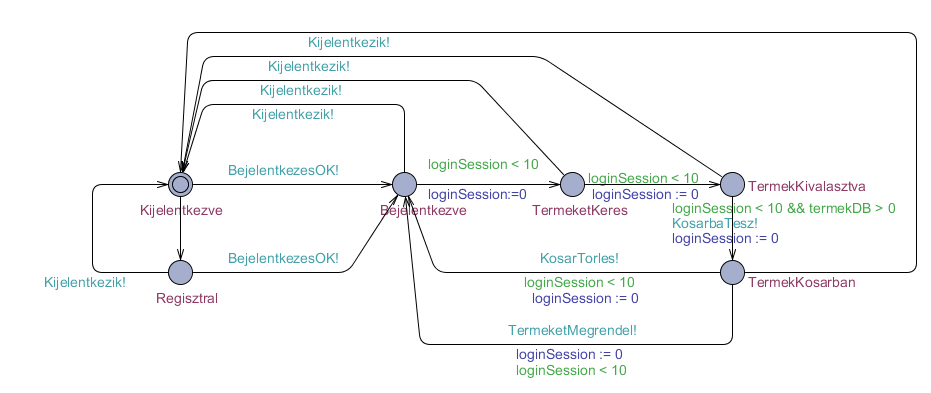
A fentiek mindegyike külön template-be került, szemléltetve azt, hogy külön feladatot látnak el, valamint a vásárlóhoz kapcsolódóan létre lett hozva egy kosár template is, amely a kosár életútját mutatja be a vásárlás során.

A következő fejezetben ezeket a templateket, valamint a hozzájuk kapcsolódó változókat, függvényeket szeretném bemutatni.

# Az automata template-jei

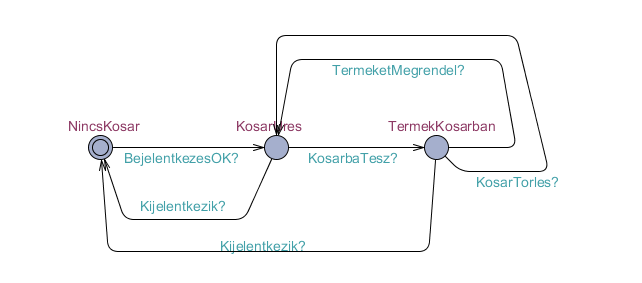
## Vásárló template

Ez a template talán a feladat legfontosabb eleme, hiszen a folyamatok nagy részében ő a kezdeményező. Felépítését tekintve az következő elemekből áll:



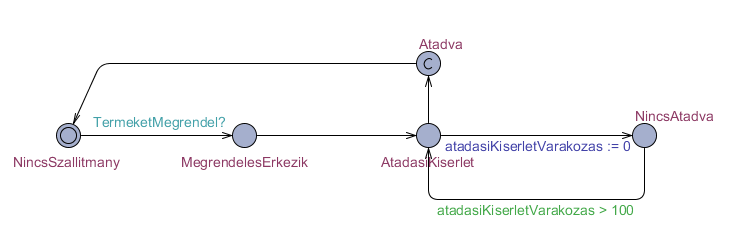
Kezdetben Kijelentkezett állapotban található, melyből átléphetünk a Regisztrál-ba. Mind a Kijelentkezett és a Regisztrál state-ben lehetőség van a bejelentkezésre. Amennyiben ez sikeres, úgy a BejelentkezesOK! channel-en üzenet kerül kiküldésre. Innentől kezdődően, amíg a Vásárló bejelentkezett állapotban van, minden lépésben akkor mehet tovább, ha a loginSession clock kisebb, mint 10, szemléltetve a böngészők időkorlátját. Emellett állapotváltozás esetén ez az óra nullázásra kerül. Ezután két állapot következik: a TermekKeres, majd ebből a TermekKivalasztas. Amennyiben a TermekKivalasztas sikeres, úgy a kosárba tehető, azzal a feltétellel, hogy a termekDB változó száma nagyobb, mint 0. Ezek után lehetőség van a kosár tartalmának törlésére, valamint a kosárban lévő termék megrendelésére. Megrendelés és törlés után is Bejelentkezett állapotba kerül a vásárló, ahonnan elölről kezdődhet a vásárlás folyamata.

## Kosár template



A fenti képen láthatóak a Kosár állapotai. Kezdetben, amíg a felhasználó ki van jelentkezve, nincs számára létrehozva kosár. A BejelentkezesOK csatornát figyelve jövet létre egy üres bevásárlókocsi, mely később átválthat TermekKosarban állapotra, ha a felhasználónak sikerült kiválasztania a terméket. Ha a Vásárló törli a kosár tartalmát, akkor visszakerül KosarUres állapotba, ha pedig Kijelentkezik, akkor a kezdő, NincsKosar-ba.

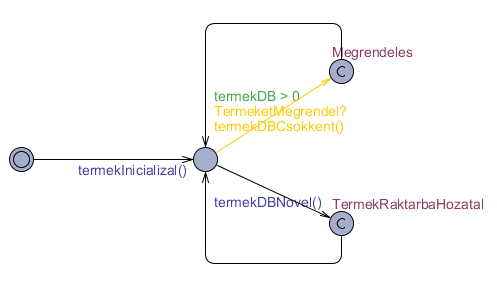
## Szallito template



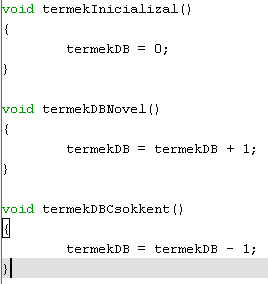
A Szallito a futárt igyekszik szimulálni, ahogyan a terméket továbbítja a megrendelőhöz. Kezdeti állapotában, ahogy a neve is tükrözi, nincs termék, amit ki kellene szállítania. Amennyiben azonban a vásárló megrendel egy terméket, a MegrendelesErkezik állapotba kerül. Innen történhet az AtadasiKiserlet state. Ahogyan a valóságban is, előfordulhat, hogy a kísérlet során nem tudjuk átvenni a csomagot, ezután a futár 1-2 nappal újra próbálkozik, ezt próbálja szemléltetni az itt elhelyezett clock: amennyiben a NincsAtadva állapotba lépünk, a megadott feltétel szerint csak akkor léphetünk tovább, ha az óra értéke legalább 100, ekkor újra próbálkozhatunk. Ha sikeresen megtörténik a tranzakció, az Atadva committed állapotba lépünk, innen pedig mindenképp tovább kell lépnünk (emiatt lett comitted), így visszakerül NincsSzallitmány kiindulási állapotba.

## Termek templtate

Ez a template rendkívül egyszerű felépítésű, csupán azért felel, hogy a megrendelések ás árubeérkezéseknek megfelelően növelje és csökkentse a készleten lévő darabszámot. Az alábbi állapotokkal rendelkezik:



A templateben az állapotátmeneteknél függvényhívások történnek, ennek pedig rendkívül egyszerű oka van: a termék készleten lévő darabszámát nehéz lenne állapotokkal ábrázolni, emiatt egy belső globális változót használtam ehhez, amely megrendeléskor csökkenti, áru beérkezésekor pedig növeli az értéket. A kezdeti állapotból egy termekInicializal() fuggvényhívás történik, hogy a kezdő érték beállításra kerüljön.



A fenti programrészleten láthatóak az értékmódosítást végző rövid függvények.

## Globális csatornák és változók

A folyamatok megfelelő működéséhez több globális csatornát is kíszítenem kellett, ezek az alábbiak:

* KosarTorles
* KosarbaTesz
* TermekValasztas
* TermekKereses
* Kijelentkezik
* BejelentkezesOK
* TermeketMegrendel

Ezek rendre a Vasarlo template állapotátmeneteikor történnek meg, ezek hatására zajlanak átmenetek a többi template-ben.

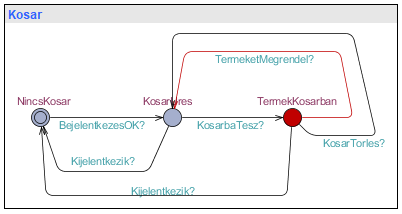
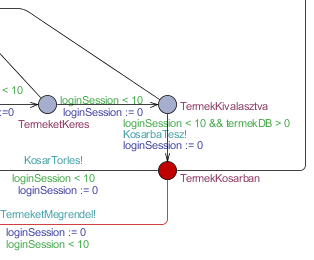
Itt fontosnak tartanám megjegyezni, hogy a TermeketMegrendel csatorna broadcast jellegű, mivel több template-ben is kivált állapotváltozást.

Globális változóként a termekDB szerepel, mivel azt több helyen is el kell érni: a Vasarlo megrendeléskor figyeli az értéket, illetve a Termek-ben történik az érték módosítása.

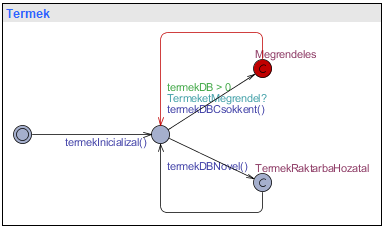
# Egy lehetséges szimuláció

Kezdetben a termék raktáron lévő darabszámát kell növelni, hogy a vásárló tudjon megrendelni, ekkor egy committed state-be kerülönk, mindenképp innen kell továbblépnünk.

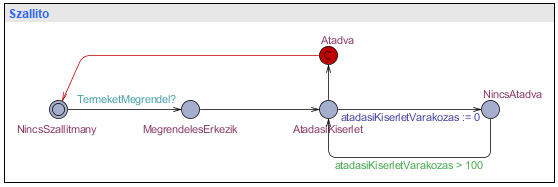
Ezután a Vásárlót sikeres bejelentkezés után egyenes úton a TermekKivalasztva állapotba juttathatjuk, ahonnan a TermekKosarban állapotba juttatva a Kosar automata is állapotot vált:



Ezután, ha a terméket megrendeljük, a Kosar visszaáll üres állapotra, míg a Szállító átmegy MegrendelesErkezik state-be. Ilyen esetben a Megrendeles comitted állapotú lesz, mindenképp ebből kell továbblépnünk, ekkor csökken a készlet darabszáma.



Ezután a szállító állapotai között léphetünk tovább, amíg el nem jutunk az Atadva statuszba.



Ezután újraindulhat a vásárlás folyamata, amennyiben azonban a loginSession eléri a 10-es értéket, előbb újra be kell jelentkeznünk.