Intelligens elosztott rendszerek
BMEVIMIAC02
2017

Fazekas Bálint (ASQLN0)
Juhász Stefánia (GLNE8V)
Tűzérzékelő és riasztó

Tartalomjegyzék:

A feladat leírása

A házi feladat során egy tűzérzékelőt és egy ahhoz kapcsolódó riasztó rendszert valósítottunk meg.

Megoldásunkban három szoba áll vizsgálat alatt. Ezekben a szobákban a falban hőmérséklet érzékelők vannak beépítve, valamint a plafonon pedig füstérzékelők találhatóak. Ezek képesek egy mással kommunikálni, kiértékelni a mért értékeket. Ezen kívül képes üzenni a riasztó rendszernek.

A riasztó rendszer feladata, hogy bizonyos protokollok alapján végrehajtson feladatokat. Ilyenek a tűzoltók, rendőrök, mentők értesítése, az ajtók nyitása az épületben és a garázsban valamint a lift leállítása.

A GUI egy kezelőfelület, amivel lehet tűzesetet szimulálni a szobánkénti hőmérséklet és füst állításával. Emellett van egy konzol rajta, amin megjelennek az ágensek üzenetei. Felül látható a hívás, ajtók, garázs ajtó, lift és a hangosbemondó állapota. Frissíteni a Refresh gombbal lehet, a konzolt törölni pedig a Clear gomb segítségével.

A megoldás összefoglalása

Tűzfigyelő ágens (TÁ):

Két fő funkciója van:

- Feladata az épület folyamatos vizsgálata füst és hőmérséklet alapján minden egyes helyiségben, külön-külön. Az ágens eszközei: füst érzékelők több helyen a plafonon és hőmérséklet érzékelők ugyancsak több helyen a falban. Ha a két szenzor értéke egyszerre tér el a normális tartománytól, akkor a helységben tűzet feltételez.
- A második feladata a tűz eset kihírdetése. Ha magasabb hőmérsékletet és füstöt érzékel az egyik helyiségben akkor ismét rákérdez, ha ez még mindig eltér a normálistól akkor kihírdeti a tűz jelenlétét, valamint a körülvevő helységekre is extra figyelmet fordít, hogy a tűz továbbterjedését időben észrevegye.

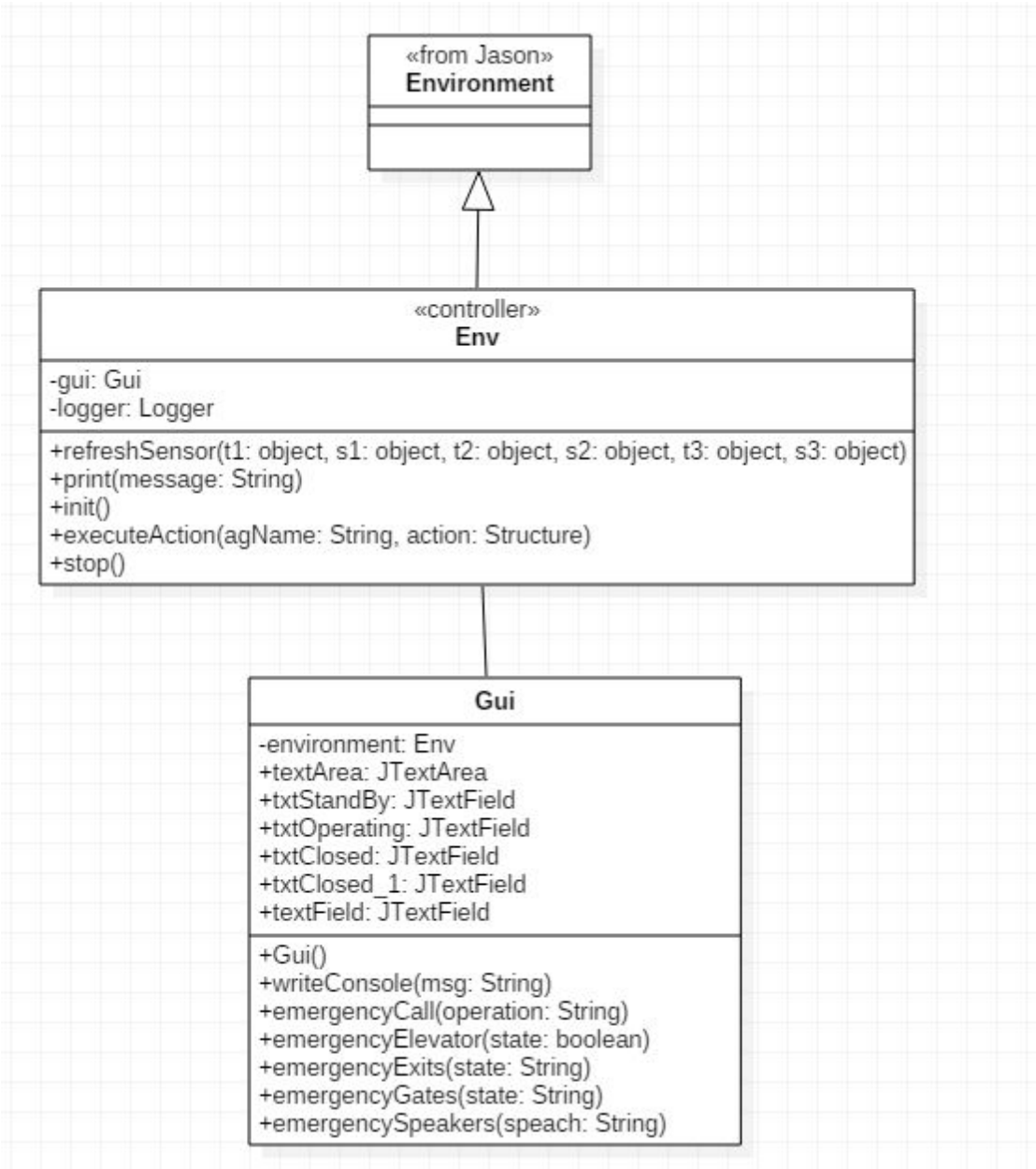
Biztonságért felelős ágens (BÁ):

Különböző vészhelyzetek esetén más-más protokoll szerint jár el. Például, baleset esetén mentők értesítése, betörés esetén rendőrök illetve az intézmény vezetőjének értesítése. A mi esetünkben csak a tűzzel kapcsolatos teendőit fogjuk kidolgozni.

- Menekülő út biztosítása, ajtók nyitása motor karok segítségével
- Lift(ek) megállítása a legközelebbi emeletnél, liftet használók értesítése a tűzesetről hangszórók segítségével (lépcsők használatára kötelez)
- Mélygarázs kinyitása a könnyebb távozás érdekében.
- Légkeverés és szellőztetős rendszerek leállítása tűzeset területén és környezetében.
- Megfelelő segítség kérés (mentő, tűzoltó)
- Figyelmeztetés hangszórón keresztül a tűzesetről (“Mindenki fegyelmezetten hagyja el az épületet!”, sziréna)
- Megfelelő személyzet figyelmének felhívása a megfelelő kiürítés megtörténtének ellenőrzésére (mindenki kijutott-e)

A két ágens között kapcsolat van. A Tűzfigyelő ágens a kiértékeli a mért adatokat, majd amit ebből leszűrt továbbítja a Biztonságért felelős ágens felé. Így elmondható, hogy a Biztonságért felelős ágens működése függ attól, hogy kap-e üzenetet Tűzfigyelő ágenstől.

A modell rajza:



A fejlesztés összefoglalása

Felhasznált eszközök:

Jason 2.1
Java JDK 1.8
Eclipse
WindowBuilder 1.9.0
StarUML

Az elkészült dolgok ASL szinten:

Maguk az ágensek, azok teljes működése. Tehát a kapott adatok kiértékelését az ágensek végzik és a következtetést levonják belőle. Ezen kívül meg van írva a kommunikáció köztük.

Az elkészült dolgok Java szinten:

Az egész rendszert egy környezete foglalja össze. Ennek a környezetnek része egy GUI, ami egy JFrame. Ezen a felhasználónk lehetősége van tesztelni a rendszert. A GUI-ban történik a nyomógombok hatásainak kezelés. A GUI-ban a Refresh gomb lenyomására a környezet segítségével bekerül a rendszerbe egy új hiedelem, ami a ciklus miatt a felhasználás után törlésre is kerül. Ezen kívül a Jason belső konzolára semmi sem kerül kiírásra, minden hatást a környezet és a GUI kezel le, tehát a GUI konzolára írunk és az Effects részen kerülnek jelzésre az éppen futó folyamatok (például hangosbemondó).

Az éppen futó folyamatok átvételét a környezet executeAction metódusa végzi el. Itt az action functora alapján az elvett adat továbbküldésre kerük. A functor lehet: print, emCall, emEl, emExits, emSpk, emGate. Switch-case segítségével bontásra került, hogy melyik esetén milyen metódud hivás történik.

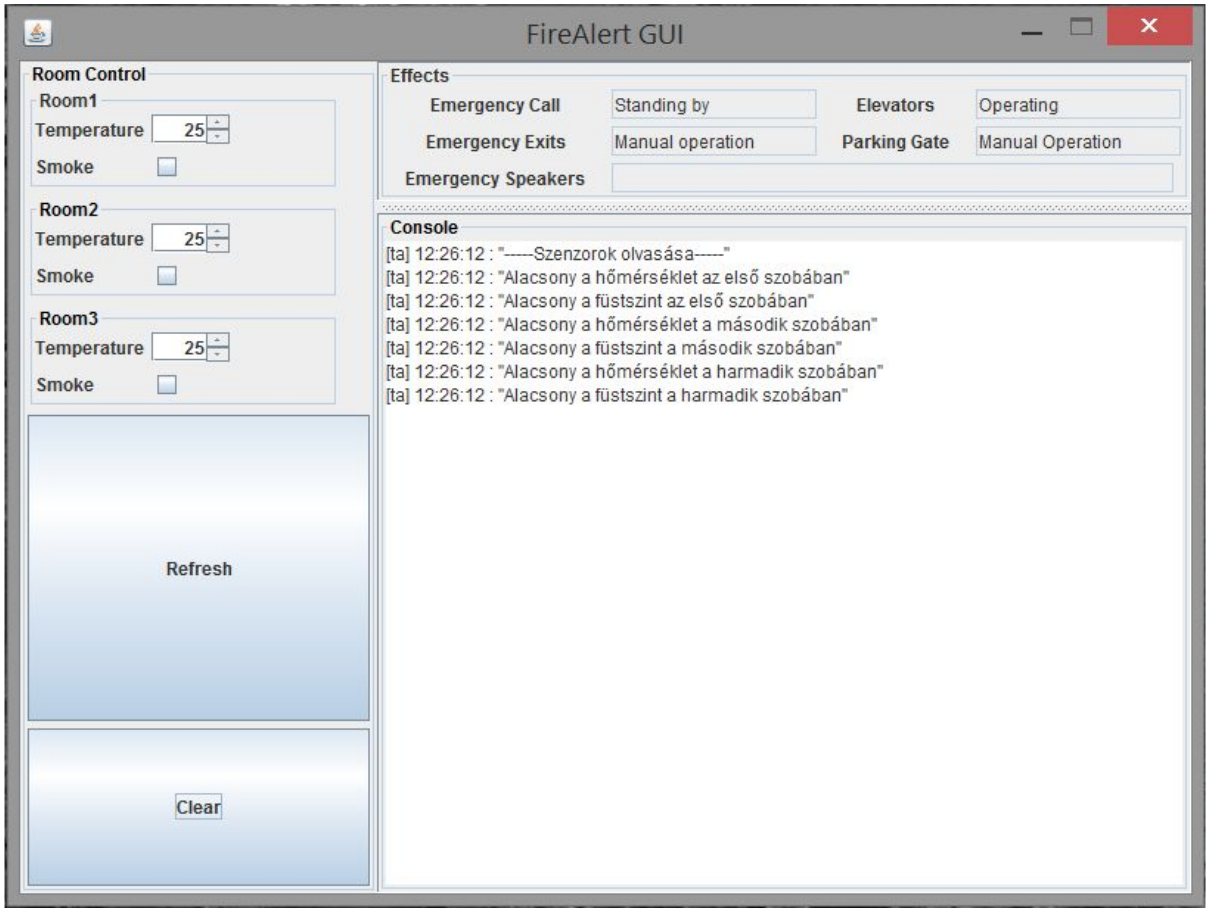
emCall: emergencyCall(msg), GUI-n Emergency call-nál megjelenik, hogy kit hív
emEl: emergencyElevator(operation), GUI-n Elevators-nál kiírásra kerül, hogy működik-e a lift
emExits: emergencyExits(state), GUI-n Emergency exits-nél kiírásra kerül, hogy milyen az aktuális működése
emGate: emergencyGates(state), GUI-n Parking Gate-nél kiírásra kerül, hogy milyen az aktuális működése
emSpk: emergencySpeakers(speach), GUI-n az Emergency Speaker-nél megjelenika szöveg, ami bemondásra kerül

Részek kidolgozása:

Az egyes feladat részeket nem osztottuk fel egymás között, minden közös munkával készült el.

A kifejlesztett program ismertetése

Az alkalmazáshoz elkészített kezelő felület képe:



Bal felső sarokban található Room Control-ban a három szobában lévő szenzorokat lehet beállítani tetszőleges értékekre. A hőmérséklet az alkalmazás indulásakor 25C°-on áll. Ez tetszőlegesen átírható vagy 0.5C°-onként csökkenthető vagy növelhető. A füst alap értelmezetten nincs, a checkbox használatával ez is tetszőlegesen állítható

Az így megadott értékeket a Refresh gomb lenyomásával tudjuk eljuttatni a rendszerünkhöz, melynek hatására frissül a Tűzfigyelő ágensünk (TÁ) hiedelmi állapota. A változás nem azonnali, meg kell várni a kiértékelési ciklust, a látható változáshoz. Ez tűz jelenlétében 5 másodpercet jelent minta vételek között, ennek hiányában 10 másodperc.

A Console ablakon figyelhetjük meg a TÁ világában észlelt állapotokat. Amennyiben tűz jelentkezik, ugyan itt jelenik meg a Biztonságért felelős ágens (BÁ) nyugtázó válasza a TÁ által átküldött üzenet hatására. Amennyiben a tűzeset megszűnik, újabb nyugtázó üzenet jelenik meg, ezúttal a baleset megszűnéséről.

Clear gomb segítségével törölhetjük ennek a kiírásnak a tartalmát, helyet adva a frissen érkező szövegeknek. Ez csak egy kényelmi funkció, a sok görgetés elkerülése érdekében egy hosszabb futás esetén.

Amennyiben baleset következett be akkor a megfelelő protokoll szerint jár el a Biztonságért felelős ágens (BÁ). Ezt az eljárást illetve hatásait figyelhetjük meg a felső Effects ablakban. Ami számot ad az automatikus hívó rendszer állapotáról, liftek üzemeléséről, ajtók és garázs kapuk helyzetéről valamint a hangos bemondóban kiadott figyelmeztetésről.

Ágensprogramok rövid összefoglalása:

Tűzfigyelő Ágens (TÁ):

Beliefes:

`hassmoke_N`$_{[source(self)]}$. - N-edik szoba füst hiedelme

`hightemperature_N`$_{[source(self)]}$. - N-edik szobahőmérséklet hiedelme

`refresh(A,B,C,D,E,F)`$_{[source(percept)]}$. - Szenzorok értékeinek hiedelme

Plans:

```
+!start <-  !!run.
+!run : refresh(A,B,C,D,E,F) <-
          print("-----Szenzorok olvasása-----"); ... !!end.
+!end : (hightemperature_N & hassmoke_N)<-
          .send(ba,untell,firealert); wait(5000); !!start.
+!end  <- .send(ba,untell,firealert); .wait(10000); !!start.
```

Ez terv együtt alkotnak egy megfigyelési hurkot, ahol a start feladata az előfeltételek megteremtése (ha kell), run feladata a szenzorok leolvasása és a fenti füst és hőmérséklet hiedelmi állapotok beállítása. Végül az end feladata a ezek kiértékelése és a BÁ ágens értesítése, valamint új ciklus elkezdése ennek függvényében.

A run során használt kiértékelő tervek:

```
+!checkT_*(N) : N > 45 <-
          +hightemperature_*; print("Magas a hőmérséklet...").
+!checkT_*(N) : N <= 45 <-
          -hightemperature_*; print("Alacsony a hőmérs...").
```

```
+!checkS_*(N) : N == true <-
          +hassmoke_*; print("Magas a füstszin..").
+!checkS_*(N) : N == false <-
          -hassmoke_*; print("Alacsony a füstszint ..").
```
(*-helyére a megfelelő szoba száma kerül)

Biztonságért felelős ágens (BÁ)

Beliefes:

`catastrophe`$_{[source(self)]}$. - kapott üzenet alapján feltételezett katasztrófa hiedelem

`firealert`$_{[source(ta)]}$. - TÁ ágenstől kapott hiedelem

`accident`$_{[source(*)]}$. -más ágenstől kapható hiedelmek
`robbery`$_{[source(*)]}$.

Plans:

```
+!run : catastrophe & accident <- emCall(ambulance).
+!run : catastrophe & robbery <- emCall(police);
emExit(close); emeGate(close).

+!run : catastrophe & firealert <- emCall(firefighters);
emExit(open); emGate(open); emEle(false);  emSpk("Mindeki
fegyelmezetten hagyja el az épületet!").

+!run : not catasrophe <- emCall(standby); emExit(manual);
emGate(manual); emEle(true); emSpk(" ").
```

Ezek a tervek hajtódnak végre különböző katasztrófa állapotok szerint. A végrehajtáshoz meghívja a megfelelő utasítást a környezetben(Environment).

```
+firealert[source(A)]: true <- print("-----Kapott üzenet
kiértékelése-----"); print("Tüz elleni protokol indítása");
+catastrophe; !run.

-firealert[source(A)]: true <- print("-----Kapott üzenet
kiértékelése-----"); print("Minden rendben"); -catastrophe; !run.
```

TÁ-tól kapott üzenetek kezelése, katasztrófa hiedelem elrendelése vagy megszüntetése, ellenlépések megkezdése.

Felvett videó

A kifejlesztett program működéséről felvett videó tárolási URL-je:
https://www.youtube.com/watch?v=PcDzpZDExos