Okos Üvegházakhoz – Vezérlőrendszer fejlesztése

1 A projekt leírása

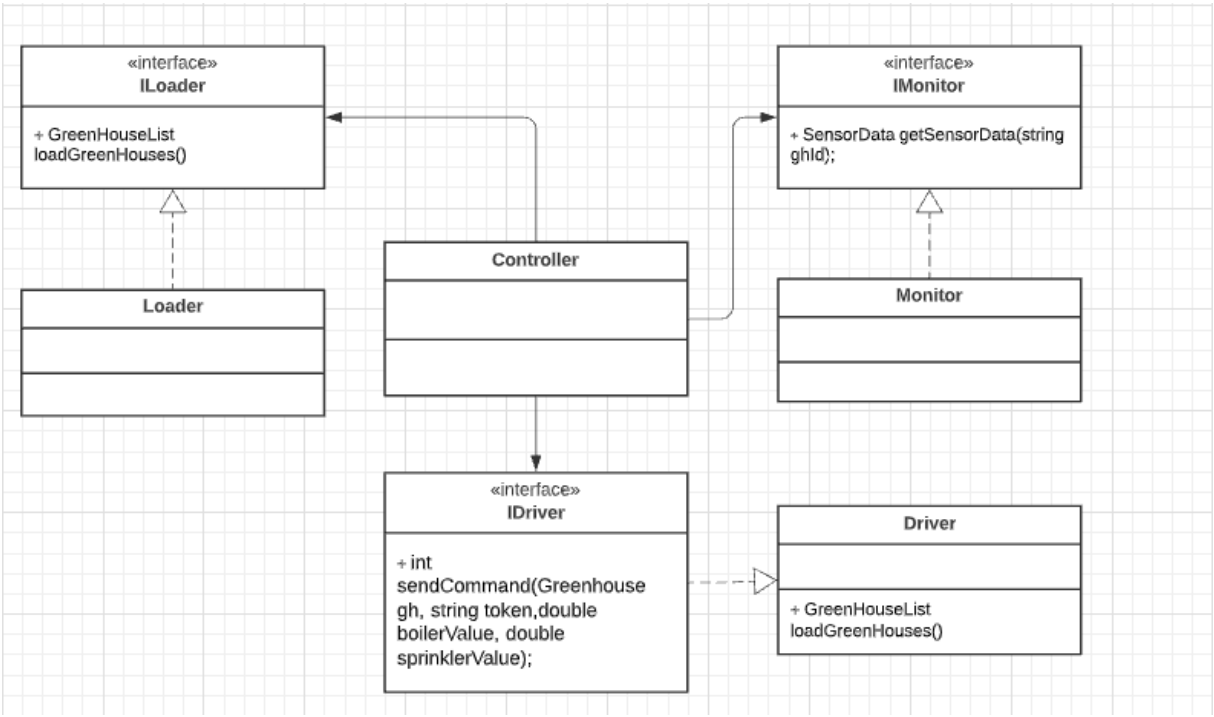
A projekt során a cél egy olyan felhő alapú vezérlőrendszer fejlesztése, amely képes automatikusan szabályozni az okos üvegházak hőmérsékletét és páratartalmát a fűtés és a locsoló berendezések vezérlésével. A rendszer képes egyszerre több üvegház kezelésére, amelyeknél a termesztett növény igényeinek függvényében előre definiálva van az elvárt hőmérséklet és minimális páratartalom. A rendszer a telepített szenzorok által mért adatokat a felhőn keresztül kéri le, amelyek alapján képes önállóan döntést hozni, hogy szükséges-e valamilyen beavatkozás.

2 Az elvárt működés

A távfelügyeleti rendszer által megvalósított működés:

- Az egyes üvegházakra vonatkozó információkat egy JSON/XML fájlból lehet betölteni a rendszer indulásakor.
- A vezérlő az indulás után a betöltött üvegházak listája alapján dolgozik és mindegyiknek sorra lekéri egy azonosító segítségével annak állapotát (MonitorService), vagyis az aktuális hőmérsékletet és páratartalmat, amelyek alapján döntést hoz, hogy milyen beavatkozásra van szükség.
- A vezérlőegység a beérkezett adatok alapján eldönti, hogy szükséges-e valamilyen beavatkozás, és ha igen, akkor a kazánnak hány fokot kell emelni a hőmérsékleten, illetve a hőmérsékletváltozás után szükséges-e újabb locsolás a páratartalom megfelelő szinten tartása érdekében. A döntés eredményét a felhőn keresztül küldi el (ControllerSevice) a megfelelő eszköz számára.
- Ha rendszer hibát észlel, vagyis túl nagy a hőmérséklet vagy páratartalom eltérése az elvárttól, akkor az eszközök meghibásodását kell feltételeznünk, és erről egy bejegyzés készül egy naplófájlba.

A rendszer osztálydiagrammja (nem teljes):



3 Modulok

A rendszer megvalósításához négy fő modul fejlesztése szükséges, amelyek a következő fejezetben leírt interfészeken keresztül kommunikálnak.

3.1 Loader modul

A modul feladata az üvegházakra vonatkozó adatok betöltése a rendszer indulása során, amelyet jelenleg a JSON/XML fájl feldolgozásával tud megtenni. A betöltés után az ILoader interfész által definiált formátumban kell az adatokat átadni a Controller modul számára, amely megkezdi az üvegházak monitorozását.

3.2 Monitor modul

A modul feladata, hogy a megkapott adatok alapján lekérdezze a felhőből (MonitorService) az azonosítónak megfelelő üvegház belső hőmérsékletét (Celsius fok) valamint páratartalmát (%), továbbá információt kapunk arról is, hogy a kazán vagy a locsolórendszer parancsot hajt-e végre, tehát működik-e a lekérés pillanatában. Az azonosító egy 10 karakteres alfanumerikus kód, amely az induláskor betöltött állományban található meg. A modul a szolgáltatás által visszaadott információkat továbbítja a Controller modul számára az IMonitor interfész által definiált struktúrában.

3.3 Driver modul

A Driver modul végzi a Controller által meghatározott beavatkozások fordítását az aktuális eszköznek megfelelő parancsokra. A Driver tartalmazza a kazán és locsoló rendszer vezérlő parancsait, ezek külön fájlból való betöltésére nincs lehetőség, a driver frissítése során a teljes modul lecserélésre kerül. A Driver az IDriver interfészen keresztül kapja az utasításokat a Controllertől, majd a megfelelő mappelés után a felhőn keresztül (CommandService) küldi ki a vezérlő jeleket.

3.4 Controller modul

A Controller modul fő feladata, hogy az aktuális hőmérséklet, páratartalom, az eszközök állapota és a beállított elvárt hőmérséklet és páratartalom alapján meghatározza, hogy szükség van-e valamilyen beavatkozásra, amely alapján a következő esetek lehetségesek:

1. Ha valamelyik eszköz parancsot hajt végre, akkor a korábbi parancsot nem írjuk felül, feltételezzük, hogy megfelelően került meghatározásra, így semelyik eszköz nem kaphat új parancsot. Ebben az esetben egy üres parancsot (üres stringet) kell küldeni a mindkét eszköz számára.
2. Ha semelyik eszköz nem hajt végre parancsot, akkor a rendelkezésre álló információk alapján határozzuk meg, hogy melyiknek kell működnie:
    - Minden érték a megadott sávban van, nem szükséges a beavatkozás, üres parancsot küldünk a felhőn keresztül.

    - A hőmérséklet nem megfelelő: kiadunk egy parancsot, amelyben megadjuk, hogy a kazánnak hány fokkal kell növelnie a hőmérsékletet az optimális hőmérséklet eléréséhez
        - Megfelelő kazán parancs kiadásával (lásd vezérlő parancsok rész) Példa: aktuális hőmérséklet 25, minimálisan megengedett 27, optimális 28 → mivel 25 < 27, ezért szükséges a beavatkozás, vagyis emeljük a hőmérsékletet az optimális szintig, ami 28-25=3 Celsius fokot jelent, így a kiadandó parancs: **bup3c**

    - Ha az előző pont alapján szükséges a hőmérséklet emelés, akkor a cél hőmérséklete alapján meghatározzuk a várható páratartalmat, ha nem, akkor az eredetit vesszük figyelembe, amely, ha elmarad az elvárttól, akkor 1%-os párolgást feltételezve megadjuk a locsoló rendszer számára a szükséges kilocsolandó vízmennyiséget (l).
        - Példa: Tudjuk az aktuális hőmérsékletet pl.: 25 Celsius, és a páratartalmat pl.: 70%. A következő táblázat alapján meghatározható, hogy ez m3-enként hány g vizet jelent a levegőben, ami 25 celsiusnál maximálisan 23,3g, ami 70%-os páratartalom mellett 23,3*0,7 = 16,31g/m3. A cél legyen például 30 celsius, ahol már 30.3 g vizet képes tárolni köbmétereként, így már 16,31/30.3, azaz 53,82%-os lenne csak a páratartalom, amelyet tegyük fel, hogy legalább 60%-on kell tartanunk.

          Így legalább 30.3*0,6 = 18,18g/m3-nek kell lennie a levegőben, tehát 18,18-16,31 = 1,89 g/m3-t kell még a levegőbe juttatni. Az 1%-os párolgással számolva: 1,87 / 0.01 = 187 g fizet kell elszórni köbmétereként. Ezt kell megszorozni az üvegház méretével, pl.: 1200 m3, így megkapjuk, hogy 224,4 liter vizet kell kilocsolni a megfelelő páratartalom eléréséhez. Így a kiküldendő parancs (kerekített értékkel): **son224l**
        -

Hőmérséklet	Max g/m3
20	17,3
21	18,5
22	19,7
23	20,9
24	22,1
25	23,3
26	24,7
27	26,1
28	27,5
29	28,9
30	30,3
31	31,9
32	33,5
33	35,1
34	36,7
35	38,3

3. Hiba észlelése, ha az elvárt szinttől nagy mértékben, vagyis legalább 5 fokkal különbözik az aktuális hőmérséklet, illetve 20%-kal aktuális páratartalom, akkor hibát enged feltételezni, amelyet fájlba kell logolni.

A vezérlő parancsok:

- Kazán: **bup*{x}*c** → hőmérséklet megemelése x fokkal
- Locsoló rendszer: **son*{x}*l** → x liter víz locsolásának megkezdése

A vezérlő a működése során iteratív végrehajtást fogunk feltételezni, de a jelenlegi prototípus során elég egyszer végig ellenőrizni az egyes üvegházakat.

4 Felhő szolgáltatások

A távfelügyeleti rendszer működéséhez két szolgáltatás használatára van szükség:

4.1 MonitorService

A szolgáltatás elérése: http://193.6.19.58:8181/greenhouse/{id}

A szolgáltatás paraméterként át kell adni az üvegház azonosítóját (id)pl.: (KFI3EW45RD), amelyre válaszul a következő adatokat kapjuk:

Paraméter	**Típus**	**Leírás**
ghId	string	Az üvegház azonosítója
token	string	Biztonsági token, amely mentése szükséges a további kommunikációhoz
temperature_act	double	Az üvegház aktuális hőmérséklete Celsius fokban
humidity_act	double	Az üvegház aktuális relatív páratartalma %-ban
boiler_on	bool	A kazán parancsot hajt-e végre?
sprinkler_on	bool	A locsoló parancsot hajt-e végre?

Példák:

- request:
  - http://193.6.19.58:8181/greenhouse/KFI3EW45RD
- response:

```
{ "ghId":"KFI3EW45RD",
  "token":"12454",
  "temperature_act":25.0,
  "humidity_act":70.0,
  "boiler_on":false,
  "sprinkler_on":false
}
```

4.2 ControllerService

A szolgáltatás elérése: http://193.6.19.58:8181/greenhouse/{token}

A szolgáltatás paraméterébe át kell adni a MonitorService által visszadott tokent, valamint body-ban a megadott JSON struktúrát a következő paraméterekkel:

Paraméter	**Típus**	**Leírás**
ghId	int	Az üvegház azonosítója
boilerCommand	string	A kazán működését befolyásoló parancs
sprinklerCommand	string	Az locsoló működését befolyásoló parancs

- request (POST)
  - http://193.6.19.58:8181/greenhouse/12454
  - Headerbe:
    - `content-type:`text/plain;
  - `Bodyba:`
    - ```
      {
            "ghId": "KFI3EW45RD",
            "boilerCommand":"bup5c",
            "sprinklerCommand":"son224l"
      }
      ```

A szolgáltatás egy hibakódot ad vissza, amely a következő értékeket veheti fel:

Hibakód	Leírás
100	A parancs végrehajtásra került!
101	Hibás kalkuláció!
102	Parancs került kiküldésre egy éppen parancsot végrehajtó eszköznek!
103	Hibás parancs került kiküldésre a kazánnak!
104	Hibás parancs került kiküldésre a locsolónak!
105	Az üzenetben lévő token nem érvényes!
106	Az üzenetben szereplő üvegház nem található!
107	Általános üzenet feldolgozási hiba!

5 Interfészek

A modulok előre definiált interfészeken keresztül kommunikálnak, ezzel biztosítható a rendszer gyors és egyszerű továbbfejlesztése.

5.1 ILoader interfész

Mivel a szolgáltatás menedzsment rendszer fejlesztése késik, ezért a prototípus jelenleg fájlból tölti be az üvegházakra vonatkozó adatokat. Ezek az adatok rendelkezésre állnak JSON és XML és formátumban is, így a megvalósítás során szabadon megválasztható a használt bemenet típusa. A szolgáltatás menedzsment rendszer gyorsabb integrálásának biztosítására, a kontrollerrel való kommunikáció a következő interfész segítségével történik:

Interfész leírás:

```
interface ILoader
{
    public GreenHouseList loadGreenHouses();
}
```

Objektum leírás:

```
public class Greenhouse
{
   public string ghId { get; set; }
   public string description { get; set; }
   public int temperature_min { get; set; }
```
```

```
    public int temperature_opt { get; set; }
    public int humidity_min { get; set; }
    public int volume { get; set; }
  }

  public class GreenHouseList
  {
    public List<Greenhouse> greenhouseList { get; set; }
  }
```

Input lekérése API-ról (a service által kezelt aktuális változat, ezért érdemes többször ellenőrizni a félév során):

- http://193.6.19.58:8181/greenhouse

JSON input példa:

```
{
 "greenhouseList": [
  {
   "ghId": "KFI3EW45RD",
   "description": "Habanero",
   "temperature_min": 27,
   "temperature_opt": 30,
   "humidity_min": 60,
   "volume": 1200
  },
  {
   "ghId": "LDIE5695WW",
   "description": "Carolina Reaper",
   "temperature_min": 29,
   "temperature_opt": 31,
   "humidity_min": 65,
   "volume": 1200
  },
  {
   "ghId": "EORIEJS9WD",
   "description": "Moruga Scorpion",
   "temperature_min": 26,
   "temperature_opt": 28,
   "humidity_min": 70,
   "volume": 1200
  }
 ]
}
```

XML input példa:

```
<?xml version="1.0" encoding="UTF-8" ?>
<root>
 <greenhouseList>
  <ghId>KFI3EW45RD</ghId>
```

```
  <description>Habanero</description>
  <temperature_min>27</temperature_min>
  <temperature_opt>29</temperature_opt>
  <humidity_min>60</humidity_min>
  <volume>1200</volume>
 </greenhouseList>
 <greenhouseList>
  <ghId>LDIE5695WW</ghId>
  <description>Carolina Reaper</description>
  <temperature_min>29</temperature_min>
  <temperature_opt>31</temperature_opt>
  <humidity_min>65</humidity_min>
  <volume>1200</volume>
 </greenhouseList>
 <greenhouseList>
  <ghId>EORIEJS9WD</ghId>
  <description>Moruga Scorpion</description>
  <temperature_min>26</temperature_min>
  <temperature_opt>28</temperature_opt>
  <humidity_min>70</humidity_min>
  <volume>1200</volume>
 </greenhouseList>
</root>
```

5.2 IMonitor interfész

Az IMonitor interfész feladata a megadott azonosító alapján az aktuális hőmérsékletnek, páratartalomnak és az eszközök állapotának lekérdezése a MonitorService használatával. A Controller és a Monitor modul között a következő interfészen történik az adatcsere:

Interfész leírás:

```
interface IMonitor
{
    public SensorData getSensorData(string ghId);
}
```

Objektum leírás:

```
public class SensorData
{
      public string ghId;
      public string token;
      public double temperature_act;
      public double humidity_act;
      public bool boiler_on;
      public bool sprinkler_on;
}
```

5.3 IDriver interfész

A IDriver interfészen keresztül történik a parancsok elküldése a CommandService használatával a konkrét eszközök felé. A Driver modul fő feladata a parancsok megfelelő mappelése, ezért a Controller nem tartalmaz semmilyen eszköz specifikus információt, így csak a parancs előállításához szükséges értékek kerülnek átadásra. A megfelelő parancs kiküldéséhez a következő intefrészt kell alkalmazni:

Interfész leírás:

```
interface IDriver
{
    public int sendCommand(Greenhouse gh, string token,
                                    double boilerValue, double sprinklerValue);
}
```

Objektum leírás:

```
public class Command
{
    public string ghId;
    public string boilerCommand;
    public string sprinklerCommand;
  }
```