Okos otthon – Távfelügyeleti rendszer fejlesztése

1 A projekt leírása

A cél egy olyan felhő alapú szolgáltatás fejlesztése, amely képes szabályozni az épületek belső hőmérsékletét a kazánok és klímák működésének vezérlése által. A felhasználó előfizet a szolgáltatásra és definiálja egy napon belüli időszakokra az elvárt szobahőmérsékletet. A manapság elérhető okos eszközök segítségével a rendszer pedig képes lekérdezni az otthonok aktuális hőmérsékletét, majd szükség esetén elindítja, illetve leállítja a fűtő vagy hűtő berendezéseket.

A fejlesztés során a cél egy általános vezérlő egység prototípus fejlesztése, amely illeszkedik a jelenlegi környezetbe, de lehetővé teszi a későbbi gyors integrációt más rendszerekkel, illetve további okos eszközök kezelésének integrálását.

2 Az elvárt működés

A távfelügyeleti rendszer által megvalósított működés:

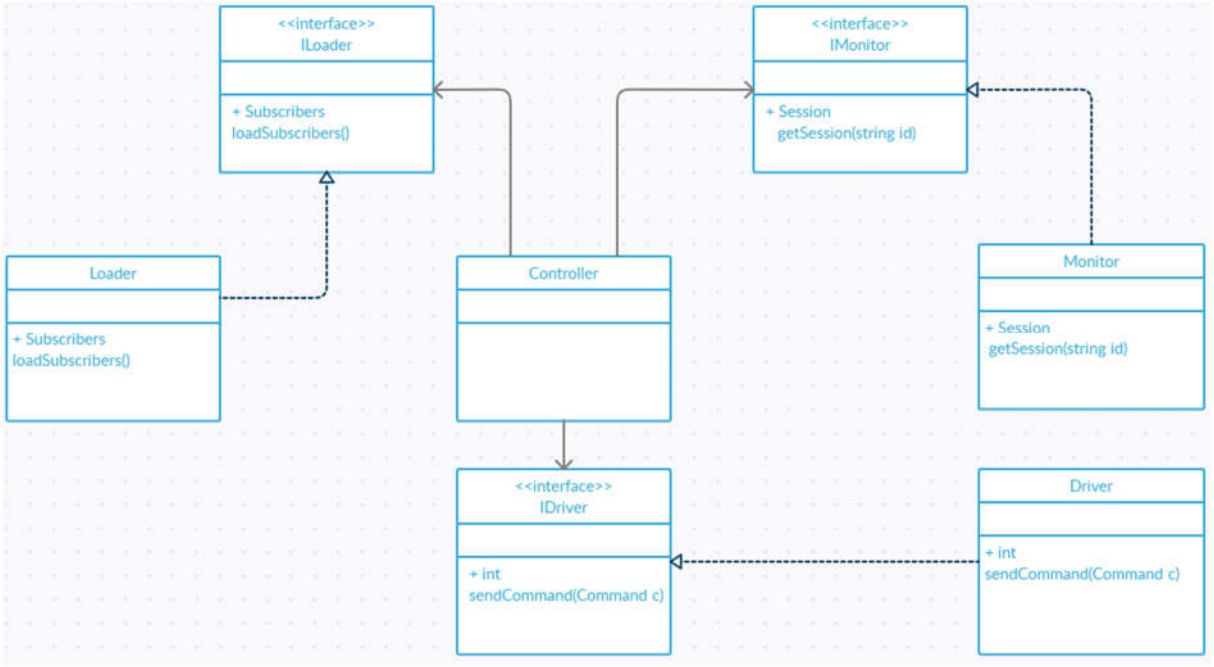
- A felhasználók egy webes rendszerbe regisztrálnak, amely tárolja a felhasználói információkat, valamint a szolgáltatásra vonatkozó adatokat. Az otthonok vezérlését megvalósító rendszer ettől függetlenül kerül megvalósításra, a végleges kialakítás során a kommunikáció a megfelelő API-kon keresztül fog történni, de jelenleg az adatok betöltését egy JSON/XML/CSV fájl alapján kell megvalósítani.

- A vezérlőegység a működése során betölti a felhasználókat, azok azonosítóját, a rendelkezésre álló eszközök típusát, valamint az aktuális időpontnak megfelelő elvárt hőmérsékletet. A következő lépésben a felhőn keresztül (*MonitorService*) egy azonosító alapján lekérdezi az aktuális szobahőmérsékletet, majd döntést hoz, hogy milyen beavatkozásra van szükség.

- A vezérlőegység a beérkezett adatok alapján döntést hoz, hogy szükséges-e a beavatkozás. Ha a hőmérséklet magasabb vagy alacsonyabb az elvártnál, akkor a felhőn keresztül (*ControllerService*) küld üzenetet a megfelelő eszköznek.

- Ha a vezérlőegység hibát észlel, vagyis az aktuális hőmérséklet nagymértékben eltér az elvárt hőmérséklettől, ekkor a rendszer egy bejegyzést készít egy naplófájlba.

A rendszer osztálydiagramja (nem teljes):



3 Modulok

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

3.1 Loader modul

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

3.2 Monitor modul

A modul feladata, hogy a megkapott adatok alapján lekérje a felhőből (*MonitorService*) az azonosítónak megfelelő otthon belső hőmérsékletét (Celsius fok) valamint a fűtő/hűtő eszközök állapotát (fűt, illetve hűt-e a lekérdezés pillanatában). Az azonosító egy 10 karakteres alfanumerikus kód, amelyet az előfizetői adatok tartalmaznak. A modul a megkapott információt csak 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 az egyes kazán és klíma típusok 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 (*ControllerService*) küldi ki a vezérlő jeleket.

3.4 Controller modul

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

- Ha a hőmérséklet megfelel az adott időszakban elvárt hőmérsékletnek és az eszközök nem aktívak, akkor nincs szükség beavatkozásra, a feldolgozás folytatódhat a következő otthon vizsgálatával.
- Ha a hőmérséklet megfelel az adott időszakban elvárt hőmérsékletnek, de valamelyik eszköz aktív, akkor utasítás kell küldeni a Driver számára, hogy az eszközt állítsa le.
- Ha a hőmérséklet nem megfelelő, tehát a megadott érték felett/alatt van 0.2 fokkal, akkor meg kell határozni, hogy fűtésre vagy hűtésre van-e szükség, amely alapján jelzést kell küldeni a drivernek, amely kiküldi a megfelelő vezérlő üzenet a felhőn keresztül (*ControllerService*).
- Hiba észlelése, ha az elvárt szinttől nagy mértékben, vagyis legalább 20%-al eltér az aktuális hőmérséklet, akkor hibát enged feltételezni, amelyet fájlba kell logolni.

A vezérlő a működése során iteratívan végzi el az ellenőrzést, vagyis 5 percenként indul a feldolgozás, amely során minden előfizetett ügyfél lakása ellenőrzésre kerül.

4 Felhő szolgáltatások

A távfelügyeleti rendszer működéséhez két szolgáltatás használatára van szükség:

4.1 MonitorService

A szolgáltatás elérése: http://193.6.19.58/8182/smarthome/{homeId}

A szolgáltatás paraméterként át kell adni az otthon azonosítóját (homeId)pl.: (KD34AF24DS), amelyre válaszul a következő adatokat kapjuk:

Paraméter	**Típus**	**Leírás**
temperature	double	A szoba hőmérséklete Celsius fokban
boilerState	bool	Igaz esetén a kazán fűz, hamis esetén kikapcsolt állapotban van
airConditionerState	bool	Igaz esetén a klíma hűt, hamis esetén kikapcsolt állapotban van

Példák:

- request (GET)
  - http://193.6.19.58:8182/smarthome/KD34AF24DS

- response

{
  "temperature":22.5,

```
  "boilerState":false,
  "airConditionerState":false
}
```

4.2 ControllerService

A szolgáltatás elérése: http://193.6.19.58:8182/smarthome/{homeId}

A szolgáltatás paraméterébe át kell adni a MonitorService által visszaadott *HomeId*-t, valamint body-ban a megadott JSON struktúrát a következő paraméterekkel:

Paraméter	Típus	Leírás
homeId	string	Az otthon azonosítója
boilerCommand	string	A kazán működését befolyásoló parancs
airConditionerCommand	string	A klíma működését befolyásoló parancs

Példa:

- request (POST)
  http://193.6.19.58:8182/smarthome/FKR34DGT23
- rersponse

```
{
  "homeId":"KR323DE24W",
  "boilerCommand":"bX1232",
  "airConditionerCommand":"cX3452"
}
```

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

Hibakód	Leírás
100	A parancs végrehajtásra került
101	Hibás parancs
102	Nincs ilyen eszköz

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 előfizetők listáját és a kapcsolódó adatokat. Ezek az adatok rendelkezésre állnak JSON/XML és CSV 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 Subscribers loadSubscribers();
}
```

Objektum leírás:

```
public class Temperature    {
   public string period { get; set; }
   public double temperature { get; set; }
}

public class Subscriber    {
   public string subscriber { get; set; }
   public string homeId { get; set; }
   public string boilerType { get; set; }
   public string airConditionerType { get; set; }
   public List<Temperature> temperatures { get; set; }
}

public class Subscribers    {
   public List<Subscriber> subscribers { get; set; }
}
```

Bemeneti formátum JSON:

```
{
  "subscribers":[
    {
      "subscriber":"John Smith",
      "homeId":"KR323DE24W",
      "boilerType":"Boiler 1200W",
      "airConditionerType":"Air p5600",
      "temperatures":[
        {
          "period":"00-08",
          "temperature":20
        },
        {
          "period":"08-12",
          "temperature":21.5
        },
        {
```

```
          "period":"12-16",
          "temperature":22
        },
        {
          "period":"16-24",
          "temperature":23
        }
      ]
    }
  ]
}
```


Bemeneti formátum XML:


```
<?xml version="1.0" encoding="UTF-8" ?>
<root>
 <subscribers>
  <subscriber>John Smith</subscriber>
  <homeId>KR323DE24W</homeId>
  <boilerType>Boiler 1200W</boilerType>
  <airConditionerType>Air p5600</airConditioner>
  <temperatures>
   <period>00-08</period>
   <temperature>20</temperature>
  </temperatures>
  <temperatures>
   <period>08-12</period>
   <temperature>21</temperature>
  </temperatures>
  <temperatures>
   <period>12-16</period>
   <temperature>22</temperature>
  </temperatures>
  <temperatures>
   <period>16-24</period>
   <temperature>23</temperature>
  </temperatures>
 </subscribers>
</root>
```


Bemeneti formátum CSV:


```
subscriber;homeId;boilerType;airConditionerType;Temperatures;
John Smith;KR323DE24W;Boiler 1200W;00-08,20,08-12,21.5,12-16,22,16-24,23;
```

5.2 IMonitor interfész

Az IMonitor interfész feladata a megadott azonosító alapján az aktuális hőmérsékletnek 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 Session getSession(string homeId);
}
```

Objektum leírás:

```
public class Session
{
   public string sessionId;
   public double temperature;
   public bool boilerState;
   public bool airConditionerState;
}
```

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. A megfelelő parancs kiküldéséhez a következő intefészt kell alkalmazni:

Interfész leírás:

```
interface IDriver
{
   public int sendCommand(Subscriber subs, bool boiler, bool ac)
}
```