



**TUDOMÁNYOS DIÁKKÖRI DOLGOZAT**

# **VEZETÉK NÉLKÜLI SZENZORHÁLÓZAT SZIMULÁCIÓJA**

**Szerző: Obermajer Ferenc**  
mérnök informatikus BSc. szak  
IV. évf.

**Konzulensek: Dr. Horváth Árpád**  
adjunktus

**Berényi Imre**  
mérnök informatikus

**Székesfehérvár, 2014.**

# Tartalomjegyzék

|                                                               |    |
|---------------------------------------------------------------|----|
| Bevezetés .....                                               | 4  |
| 1. WSN technológia ismertetése .....                          | 5  |
| 1.1. Egy valós rendszer ismertetése .....                     | 7  |
| 1.2. A rendszer elemei .....                                  | 8  |
| 1.2.1. Terepi vezérlőeszköz (BT) .....                        | 8  |
| 1.2.2. DAS távoli adatgyűjtő rendszer (TCP/IP) .....          | 8  |
| 1.2.3. Bázis állomás (BT, TCP/IP, WSN) .....                  | 8  |
| 1.2.4. Adatgyűjtő (WSN) .....                                 | 9  |
| 1.3. SensorWeb rendszer .....                                 | 9  |
| 1.3.1. Felépítés .....                                        | 9  |
| 1.3.1.1. MoteSystem .....                                     | 10 |
| 1.3.1.2. Adatbázis réteg .....                                | 10 |
| 1.3.1.3. Kliens réteg .....                                   | 11 |
| 1.3.2. Üzenetküldési folyamat .....                           | 11 |
| 2. Szimuláció elkészítése .....                               | 12 |
| 2.1. HardWare Simulation szoftver specifikáció .....          | 12 |
| 2.1.1. Általános áttekintés .....                             | 13 |
| 2.1.2. Feladat .....                                          | 14 |
| 2.1.3. Időterv .....                                          | 14 |
| 2.1.4. Minimális rendszerkövetelmény .....                    | 14 |
| 2.1.5. Többnyelvűség .....                                    | 15 |
| 2.1.6. Megjelenítési módok .....                              | 15 |
| 2.1.7. Fejlesztői környezet .....                             | 15 |
| 2.1.8. Jogosultsági körök .....                               | 15 |
| 2.2. HWS szoftver moduljai strukturális megközelítéssel ..... | 15 |
| 2.2.1. Kommunikációs csatornák .....                          | 16 |
| 2.2.1.1. WCF (Windows Communication Foundation) .....         | 17 |
| 2.2.1.2. Főbb szolgáltatások .....                            | 17 |
| 2.2.1.3. HWS API .....                                        | 19 |
| 2.3. HWS szoftver moduljai UI megközelítéssel .....           | 20 |
| 2.3.1. Bejelentkezés .....                                    | 20 |
| 2.3.2. Folyamatok panel .....                                 | 21 |
| 2.3.3. Projekt létrehozása .....                              | 22 |
| 2.3.4. Eszközök panel .....                                   | 23 |
| 2.3.5. Összegzés panel .....                                  | 25 |

|        |                                 |    |
|--------|---------------------------------|----|
| 2.3.6. | Mentett projekt betöltése ..... | 26 |
| 2.3.7. | Log üzenetek.....               | 27 |
| 2.4.   | Eredmények.....                 | 28 |
| 3.     | Konklúzió.....                  | 30 |
| 4.     | Összefoglalás.....              | 31 |
| 5.     | Ábrajegyzék.....                | 32 |
| 6.     | Irodalom jegyzék.....           | 33 |

# Bevezetés

A dolgozat részletesen ismerteti egy olyan vezeték nélküli szenzorhálózat szimulációját, amelyet virtuálisan hozunk létre annak érdekében, hogy megfelelő képet kapjunk a jövőben egy hasonló valóban kiépített rendszer működéséről.

A TDK dolgozat fő témája, hogy szisztematikusan leírja egy szimulációs szoftver fő lépéseit, valamint az ezekhez tartozó algoritmusokat, amelyek a szenzorhálózat előállítását segítik elő.

A Wireless Sensor Network (továbbiakban: WSN) egy olyan heterogén eszközökből álló hálózat, mely alacsony energiafelhasználással vezeték nélkül kommunikál egy bázissal, ami aztán továbbküldi az eredményeket bizonyos időközönként a központi feldolgozó szerverparkba. A szenzorok eszközökben helyezkednek el, amelyek összeépíthetők nagyobb egységekbe: szondákba valamint szondacsoportokba illetve telephelyekbe. A valós környezetben ezek az eszközök nyílt terepen végeznek méréseket, a szimulációs környezetben pedig előre meghatározott intervallumokon belüli értékek állítódnak elő.

A szoftver egy költséghatékony megoldást kínál, mivel nem szükséges fizikailag létrehozni és kihelyezni az eszközöket, mert az egész hálózat szimulációs módszerekkel is leképezhető.

Létrehozhatunk normál körülményeket utánozó WSN szimulációkat, de olyan helyzetet is előállíthatunk, amely szélsőséges mérési eredményeket közöl a bázissal. Az esetek variálásával tesztelhetjük a rendszer rugalmasságát, és stabilitását. A szoftver lehetővé teszi, hogy több ilyen hálózatot menedzselhessünk valós időben. Ezzel a szerverpark feldolgozó képessége is tesztelhetővé válik.

Időzíthető az, hogy mikor küldje a szimuláció a mérési eredményeit a szerverparkba, így gyakorlatilag összességében egy valós WSN bármely paraméterét le tudjuk utánozni.

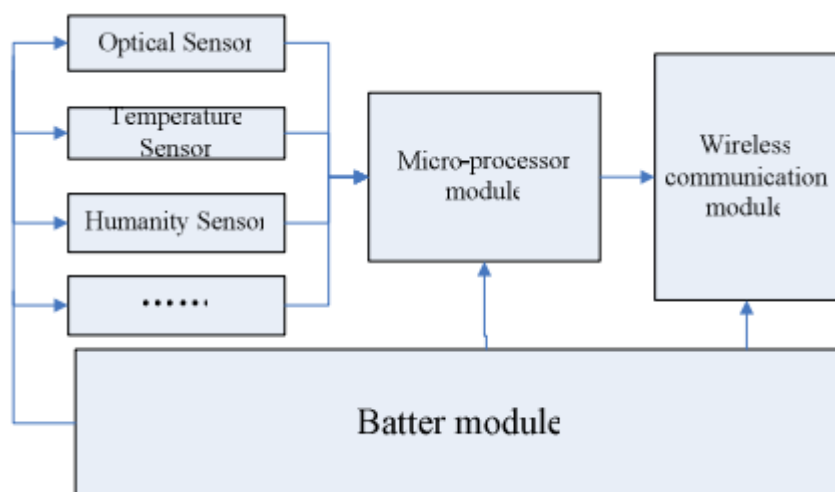
A TDK dolgozatban végül a gyakorlatban felhasznált szoftver eredményeit közlöm valamint az általa feltérképezett problémák tükrében a hasznosság mértékét is számításba veszem.

A szimulációs program egy olyan téma feldolgozása, melyet egy WSN technológiát alkalmazó cég ajánlott fel nekem.

# 1. WSN technológia ismertetése

A vezeték nélküli kommunikáció fejlődésével, valamint az elektronika és mikroszámítógépek területén elért eredmények hatására a vezeték nélküli szenzor kommunikáció az egész világon széleskörűen használatossá vált.[1]

A vezeték nélküli szenzor hálózatokban szereplő heterogén eszközöket node-oknak nevezzük. Több node képes önálló hálózatba szerveződni. A WSN (Wireless Sensor Network) node-ok belső felépítése változó lehet, de alapvetően mindegyik tartalmaz egy mikroprocesszort, egy vezeték nélküli kommunikációért felelős modult és az adott projekthez szükséges szenzorokat. Ezenkívül az áramellátásért felelős akkumulátor is helyet foglal benne. A technológiát meglehetősen preferálják az ipari szektorban valamint a tudományos intézetekben, mivel alacsony költségű, valamint meglehetősen minimális az energiafogyasztása is. Az eddigieken túl használatba állították már katonai-, egészségügyi-, agrárgazdasági-, környezetvédelmi-, és rengeteg más területen is.

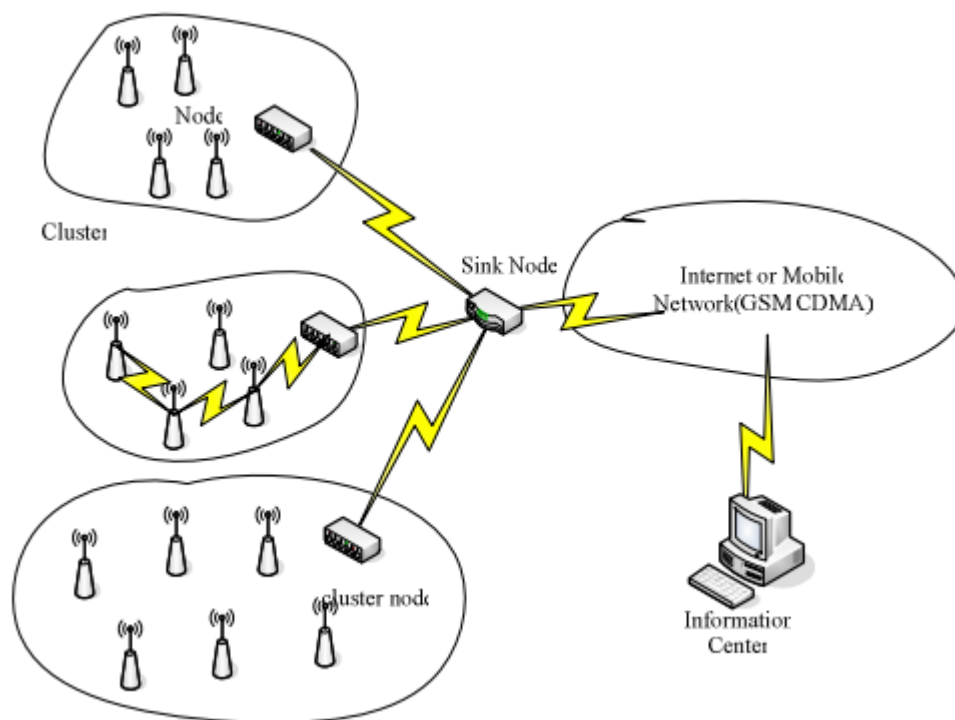


1. ábra: Egy node hardveres struktúrája [1]

A U.S. business magazin a vezeték nélküli szenzorhálózatot jósolta az egyik legjelentősebb jövőbeni high-tech iparágnak.[1]

A hálózat tehát a nagymennyiségű hasonló építőelemből, úgynevezett node-okból épül fel. Ezek lehetőség szerint szétszórva helyezkednek el egy bizonyos területen. Képesek egymással kommunikálni a saját csatornáikon és mindegyik ellát forgalomirányítási szerepet is. Jellegzetes ad-hoc struktúrája van. Rendkívül jól bővíthető a hálózat új eszközök beállításával. Bővíthetősége azonban korlátozott mivel hardver korlátai révén nem képes egy bizonyos számú

eszköz mennyiséget már kezelni. A bővíthetőségi határ jellemzően 100, 1000 vagy akár több ezer node-ot is jelenthet, ennek anyagi-, és az előbb említett fizikai korlátai vannak.



2. ábra: WSN rendszer hálózati struktúrája [1]

Az eszközök klaszterekbe tagolódnak és akár egymáson keresztül is kommunikálva egy központi klaszter node-ba küldik a méréseiket, ahonnan vagy közvetlenül, vagy közvetetten egy több klasztert egybefogó node segítségével az adatgyűjtő állomásra irányítódik, jellemzően GSM vagy CMDA hálózaton keresztül.

A telephelyen képesek vagyunk monitorozni és kiértékelni a méréseket. Lehetőség van a szenzorok által mért eredmények felhasználásával gyorsabban jelezni az általuk lefedett területen előforduló szélsőséges vagy nem várt eseményeket. A manuális módszer ezzel ellentétben, sokkal lassabban történik és emberi figyelmetlenség tényezőjét is bele kell számolnunk.

A technológia alkalmazásával képesek vagyunk egy olyan rendszert kialakítani, amely automatikusan reagál és közbelép, ha a normálistól eltérő, esetleg nem várt esemény történik. Ezáltal olyan fokú biztonságot teremthetünk meg a vezeték nélküli szenzorhálózat által lefedett területen, amely védelmet nyújt az azt monitorozó szenzorok által érzékelt eseményektől.

Jelenleg hazánkban egyre jelentősebb mértékben kezdik alkalmazni a technológiát. Megannyi kis-, közép-, és nagyvállalkozás próbálja az itthoni piacon népszerűsíteni és értékesíteni a saját fejlesztéseiket, több-kevesebb sikerrel, mert bár a technológia a praktikusságáról, olcsóságáról és könnyen telepíthetőségéről lett híres, az eredmények

folyamatos kiértékelése és adminisztrálása megnöveli a költségeket, így olyan területen ahol nagymértékű profit nem keletkezik, ezt nem látják kifizetődőnek.

### 1.1. Egy valós rendszer ismertetése

A technológia alkalmazása a lokális méretektől kezdve akár a több telephely lefedettségének megteremtésére is kialakítható. A különböző projektek határozzák meg, hogy miféle elrendezésben rakjuk le a terepen, milyen fizikai formációba helyezzük az eszközt, valamint hogy az adott eszköz milyen szenzorral rendelkezik. Készíthető olyan node is, amely több szenzort tartalmaz.

Lehetőségem nyílt egy olyan vállalat (KKV) projektjébe beletekinteni, amely a vezeték nélküli szenzorhálózatot főleg a mezőgazdaság által megteremtett lehetőségeket próbálja kiaknázni. Gondolhatunk itt magtárolókra, szántóföldekre vagy akár erdőőrökre is.

A vállalati környezetben a hierarchia a következő:

***Vállalat > Telephely > Terület > Bázis > Node***

Tehát például, ha egy mezőgazdaságban használt magtárolóban (siló) szeretnénk felépíteni egy rendszert, akkor egyes eszközöket erre a célra megtervezett csövekben helyezik el, amely könnyen beleállítható a mérendő gabona sokaságába. Egy ilyen rúdban több szenzor is található, esetünkben szükség van minimum egy hőmérsékletmérő- valamint egy páratartalom mérő szenzorra. A rúdban az eszköz közvetlenül a borkupak mögött található, ennek a nyakján vannak elhelyezve a szenzorok.

Ezen elemek kihelyezésével és együttes használatával jön létre a telephely, amely egy olyan zárt területnek számít, ahonnan az adatok egy (vagy több) kitüntetett bázison keresztül irányítódnak a központi szerverpark felé, ahol a beérkező adatokat úgy kezelik, mint egy önálló állomás adatait. Ez azért fontos, hogy fizikailag lokalizálni lehessen egy esetlegesen előforduló nem várt mérés helyét.

A node-ok vezeték nélkül kommunikálnak így a fizikai telepítés nem tart sokáig az egyes telephelyeken. A vállalat a rendszer elhelyezése után a beérkező adatokat menedzseli és karbantartja, így a megrendelőnek nem kell törődnie a hálózaton felmerülő esetleges problémákkal.

A felhasználó egy online elérhető grafikus monitorozó szoftver segítségével ellenőrizheti a rendszerét és az egyes node-ok állapotát. Ezenkívül egyeztetett információk alapján bizonyos

időközönként, valamint nem várt események történésekor üzenetet küld a felhasználó telefonjára, e-mail címére vagy egyéb más kommunikációs eszközre.

A vállalat a nem várt eseményből adódó problémákat, ha csak az nem az egyik eszközt érinti, riasztásos üzenet formájában küldi, nem feladata biztosítani a területet, viszont a figyelt hely szirénáját például működésbe hozhatja. Tervben van, hogy tűz esetén, olyan projekteknél ahol ez megengedhető, a locsolórendszer beindítását is alkalmazhatja.

Jelenleg számos olyan terület van ahol a cég vezeték nélküli szenzorhálózatos szolgáltatást biztosít. A beérkező adatok feldolgozására egy olyan rendszert készítettek, amely képes a különböző projektek méréseit egy közös helyen kezelni.

## **1.2. A rendszer elemei**

A rendszer működésében szerepet játszó alapelemek a következők:

- Terepi vezérlőeszköz (BT)
- DAS távoli adatgyűjtő rendszer (TCP/IP)
- Bázisállomás (BT, TCP/IP, WSN)
- Adatgyűjtő (WSN)

### **1.2.1. Terepi vezérlőeszköz (BT)**

Az eszköz alapvetően egy Android rendszerű, BT interfésszel rendelkező hardver. Az ENCO terepi vezérlőszoftverrel képes beavatkozni a mérőrendszerbe, valamint annak állapotát monitorozni.

### **1.2.2. DAS távoli adatgyűjtő rendszer (TCP/IP)**

Két interfésszel rendelkező mérésadatgyűjtő rendszer. Egyik oldalról egy vagy több bázisállomás számára ad TCP/IP alapú csatlakozási felületet, melyen keresztül a bázisállomás a rendszer mérési adatait feltölti. Másik oldalról a felhasználó WEB (http) interfészen keresztül szervezett formában tekintheti meg az adatokat.

### **1.2.3. Bázisállomás (BT, TCP/IP, WSN)**

A bázisállomás alapvető funkciója a GPRS modemén keresztül a mérési adatok eljuttatása a DAS távoli adatgyűjtő rendszerre, valamint BT interfészén interfészt biztosít egyszerre egy eszköz számára. Erről a BT hardver gondoskodik.

Továbbá a bázisállomás adatgyűjtő is egyben, tehát a Terepi eszközön a bázisállomásnak is meg kell jelennie, mint egy speciális adatgyűjtő eszköz. Mérési adata pl. a GSM térerősség, valamint a rá csatlakoztatott szenzorok adatai.

A bázisállomás harmadik szerepköre a WSN interfészen keresztüli kommunikáció az Adatgyűjtőkkel, amelyek routing protokolljának “gyökér” csomópontjaként szerepel.

#### **1.2.4. Adatgyűjtő (WSN)**

Az adatgyűjtők feladata a csatlakoztatott szenzorok kezelése, és a mérési adatok továbbítása a beállított konfiguráció szerint.

### **1.3. SensorWeb rendszer**

A SensorWeb rendszer az adatgyűjtő szoftveres megvalósítása. A komponensei segítségével elvégzi az adott projekt során alkalmazott vezeték nélküli érzékelők kezelését és az ezekről érkező adatok megjelenítését.

Az érzékelők rendszerbe illesztése GPRS, vagy Ethernet kapcsolaton keresztül végezhető. A beérkező adatokat a rendszer feldolgozza, azokat mérési eredményé alakítja. Teszi ezt oly módon, hogy a kliens oldali alkalmazásoknak már könnyen értelmezhető legyen.

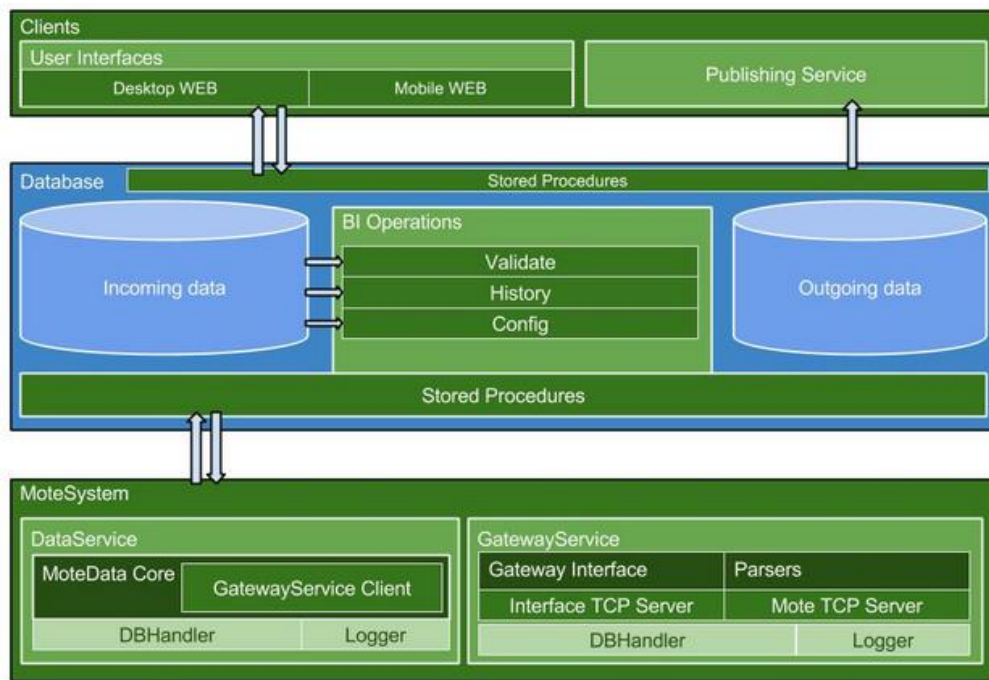
A feldolgozott adatok elérésére a rendszer több megoldást támogat. Kapcsolódhat hozzá felhasználói felület, de egy jól definiált, az adott alkalmazáshoz igazított interfészen keresztül is elvégezhető a lekérdezések.

A rendszer ipari körülmények között is alkalmazható, megfelelően az ipar által támasztott elvárásoknak.

A szoftverek szolgáltatásként futnak olyan felhasználói jogosultságokkal, melyek a feladat végrehajtásához éppen szükségesek.

#### **1.3.1. Felépítés**

A rendszer működését három jól elkülöníthető réteg biztosítja. Ezzel elérhető a terhelés megfelelő elosztása és a különböző feladatok szétválasztása, ezek egymástól független karbantartása.



3. ábra: SensorWeb rendszer struktúrája

#### 1.3.1.1. MoteSystem

A MoteGateway szoftver két szervizalkalmazás, melyek együtt a megnyitott TCP/IP kapcsolaton keresztül fogadják az érzékelők által küldött adatokat, majd elvégzik ezek feldolgozását és tárolását. A rendszer szempontjából egy réteget képviselnek, de a közöttük kialakított TCP/IP kapcsolatnak köszönhetően akár külön szerverre is telepíthetők. Ennek további előnye, hogy a különböző szenzorokról érkező adatok értelmezésére használt algoritmusok külön frissíthetők, a teljes rendszer leállítása nélkül.

#### 1.3.1.2. Adatbázis réteg

A tárolás az adatbázis szerveren található tárolt eljárásokon keresztül történik, az ezek hatására kiváltott események pedig további feladatokat látnak el. Ezek közé tartozik az adatok ellenőrzése és kimenő adattáblákba írása, illetve minden, az adatgyűjtő rendszer által megvalósított üzleti logikai elemet is ez a réteg tartalmaz.

A MoteSystem nem foglalkozik az adatok ellenőrzésével, vagy a megfelelő struktúrába rendezésével. Egyszerűen átadja a csomagot az adatbázis oldali tárolt eljárásnak, ami elvégzi azok beírását a megfelelő adattáblákba.

Új sor beillesztése esetén lefutnak az adattábla megfelelő triggerei, melyek elvégzik az üzleti logika által megkívánt műveleteket. Ezek közé tartozik az adatok ellenőrzése, tárolása. Az ellenőrzésen átjutott sorok bekerülnek azokba az adattáblákba, melyek a kliens oldal gyorsabb

kiszolgálásra alkalmasak. Ezek a lekérdezések számára könnyen kiválasztható struktúrában kerülnek kiírásra, szintén erre a célra létrehozott tárolt eljárások segítségével.

#### **1.3.1.3. Kliens réteg**

A felső réteg biztosítja a gyűjtött adatok elérését. A webes felülettel rendelkező kliensszoftverek mindig az adott projektnél várt formában és stílusban jelenítik meg az adatokat, de elérhető egy webes szolgáltatás is, melyből szintén a feladathoz igazított formában kérhetők le adatok.

#### **1.3.2. Üzenetküldési folyamat**

A bázis küld egy tranzakció kezdete jelet, majd elkezdi küldeni az üzeneteket. Ha végzett, küld egy tranzakció vége jelet a kontrol csatornán. A szerver ekkor szintén küld egy startot a vezérlőn, majd elküldi a konfigurációt megfelelően becsomagolva. Ha befejezte, vagy küld egy WAIT jelet a kontrol csatornán, ahol jelzi a várakozás hosszát, vagy küld egy tranzakció vége jelet a kommunikáció lezárásához.

## 2. Szimuláció elkészítése

Gyakornoki időszakom alatt lehetőségem nyílt a vállalat különböző projektjeibe betekintést nyerni.

Ezalatt az idő alatt lehetőséget kaptam, hogy elkészítsek egy szimulációs rendszert, amely képes virtuálisan leképezni egy vezetékek nélküli szenzorhálózatot. Ennek a virtualizációnak ugyanazon protokollok betartásával kell kommunikálnia az adatgyűjtő rendszerrel, mint az eredeti rendszerek.

Egy szimuláció elkészítése több célt is szolgálhat a tudomány bármely területén. Az efféle modellezés nagyon fontos lehet az azt elkészítő vállalat vagy kutatóintézetek számára.

A szimuláció valamely vizsgálandó jelenség vagy folyamat modell szintjén történő utánzása, aminek a célja, hogy az eredetihez mérten ugyanolyan hatást keltsen az őt alkalmazó felhasználó számára, mintha az eredeti környezetben tette volna ezt meg. [2]

Ha a szimulációt a szimulálandó rendszer előtt készítjük el, akkor:

- Lehetőség nyílik egy előrejelzést kapni az éles rendszer működéséről.
- Rálátást kapunk az esetlegesen a későbbiekben előforduló leggyakoribb logikai korlátokra.
- Megvizsgálhatjuk, hogy a létrehozandó rendszerhez kapcsolódó már létező rendszerekkel milyen mértékű összeegyeztetést mutat.

Ha a szimulálandó rendszer elkészülte után készül róla szimuláció:

- A már meglévő rendszer hibáinak könnyebb feltérképezhetőségét szolgálhatja.
- A meglévő rendszer bővíthetőségének kiderítése céljából létrehozott emulációként, vagy.
- az adatfeldolgozó rendszer stabilitását tesztelő szimulációként szolgálhat.

### 2.1. HardWare Simulation szoftver specifikáció

A vezetékek nélküli szenzorhálózatot szimuláló szoftver egy már létrehozott valós WSN technológiát alkalmazó rendszert kíván szimulálni. A projekt a HardWare Simulation nevet kapta, amelyet a továbbiakban HWS névvel is fogom említeni a dolgozatban.

A HWS létrejöttének céljai:

- Rendszer stabilitásának vizsgálata
- Költségek csökkentése

- Hibák javítása
- Bővíthetőség tesztelése
- Biztos, ellenőrizhető mérési értékekkel való tesztelés
- Tesztelés céljából felépített valós vezeték nélküli szenzorhálózat kiváltása
- Szélsőséges vagy hibás mérési eredmények szimulálása
- Valós rendszer normál eredményeinek szimulálása

Mivel a szoftver már egy létező rendszert modellez, ezért illeszkednie kell az adott protokollokhoz is, amely a fejlesztők által meghatározott normatíváknak felel meg.

A vállalat követelményei a szoftver kompatibilitásával kapcsolatban:

- A SensorWeb rendszer dolgozza fel a bejövő méréseket, tehát a szimulációnak is erre kell küldeni azokat.
- Adatgyűjtő TCP/IP kapcsolattal kommunikál, ezért a HWS-nek is ezt a csatornát kell használnia.
- A HWS részeit önálló egységekre kell bontani melyek szolgáltatásként futnak a háttérben a szerveren.
- A felhasználó UI felületen menedzseli a szoftvert.
- A UI meghatározott API kérésekkel kommunikál a szimuláció szolgáltatásaival és a SensorWeb rendszerrel.
- Vállalat által működtetett és karbantartott jogosultság kezelés a HWS-ben is alkalmazandó.

A szoftver jelenleg tesztelési státuszban van. Az alábbiakban ismertetem a felépítését, valamint a működését. Az áttekintés segítségével a projekt általános tulajdonságait taglalom.

### **2.1.1. Általános áttekintés**

Mint minden hivatalos projektnek, így ennek is az első lépése a megfelelő specifikáció elkészítése. A specifikáció célja, hogy a projektben szereplő részütemekhez iránymutatást adjon, illetve definiálja azokat mind szakmailag, mind pedig a megrendelő számára is közérthető módon. A szoftver fejlesztése során a specifikációban foglaltakat a fejlesztőnek, valamint a tesztelőknek ismernie kell. A fejlesztés és tesztelés alapja ezen dokumentum. Az elkészítés során a specifikációban definiált feladatok valósulhatnak meg, eltérni szigorúan tilos vagy csak pontos egyeztetés mellett lehet.

A tesztelőknek ezenkívül még törekedniük kell arra, hogy a specifikáció által definiált minden egyes feladatrészt a kvalitásának megfelelően, a tőle elvárt szakmai alapossággal, tesztesetekkel lefedjen.

### **2.1.2. Feladat**

Egy olyan szimulációs környezet megteremtése, amely segíti a szenzorhálózatos termékek tesztelését, a különböző környezeti és egyéb lehetőségek előállításával.

### **2.1.3. Időterv**

A specifikációban meg kell említeni, hogy a projekt előreláthatólag mikor kerül be a legfontosabb fejlesztési szakaszokba. Ezt a projekt nehézségének felmérésével és a megfelelő egyeztetés után állapítjuk meg.

- A projekt első fejlesztési ütemének befejezési időpontja: **2014.09.01**
- A tesztelési idő belső rendszerben: **2014.09.05 - 2014.10.31**
- A tesztelési idő nyilvános környezetben: **2014.11.03 – 2014.12.01**
- A Szoftver 1.0 kiadása, 100%-os készültség: **2015.01.01**

### **2.1.4. Minimális rendszerkövetelmény**

A specifikációnak tartalmazni kell egy minimális rendszerkövetelményt, amely leírja, hogy legalább milyen erőforrásoknak kell megfelelniük egy adott számítógépnek, hogy a szoftver zökkenőmentesen tudjon rajta működni.

- Processzor: 1 gigahertzes (GHz) vagy gyorsabb, PAE-, NX és SSE2-támogatással
- RAM: 1 gigabájt (GB) (32 bites) vagy 2 GB (64 bites)
- Merevlemez terület: 16 GB (32 bites rendszerhez) vagy 20 GB (64 bites rendszerhez)
- Videokártya: a Microsoft DirectX 9-es verzióját támogató grafikus eszköz
- WDDM -illesztőprogrammal
- CD-ROM- vagy DVD-ROM meghajtó
- Billentyűzet és Microsoft egér vagy azzal kompatibilis mutatóeszköz
- Monitor (1024 x 768) vagy nagyobb felbontású
- Windows XP/7/8/8.1 operációs rendszer

### **2.1.5. Többnyelvűség**

Bár a tudományos munka és a specifikáció magyar nyelven íródott a szoftvernek az angol nyelvet kell támogatnia, mivel ez a sztenderd fejlesztői nyelv.

### **2.1.6. Megjelenítési módok**

Tartalmaznia kell a fejlesztési dokumentumnak a minimális képernyő felbontási igényt is. A szoftver által támogatott alap képernyőfelbontás: 1024 x 768.

### **2.1.7. Fejlesztői környezet**

Az előírás egyik legalapvetőbb témája a használt fejlesztői környezet. Meghatározó szerepe van a projektben, hiszen ez kínálja fel nekünk a program megvalósításának elemi eszközeit.

A szoftver C# nyelven íródott. A fejlesztői környezet Microsoft Visual Studio 2012.

### **2.1.8. Jogosultsági körök**

A felhasználói kezelést egy már használatban lévő, szoftvermodul fogja biztosítani. Ez a SensorWeb rendszer része, amelyre az API-n keresztül kapcsolódunk. Az összes vállalati projekt ezt használja, így ennek is tartalmaznia kell.

Jogosultsági körök:

- Adminisztrátor
- Felső vezetők
- Felhasználók

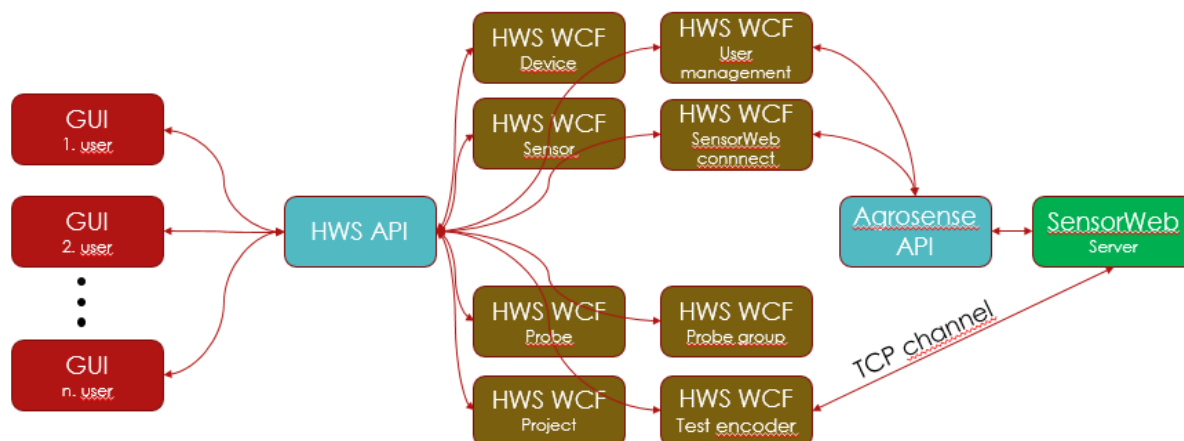
## **2.2. HWS szoftvermoduljai strukturális megközelítéssel**

A szoftvert a felépítése szempontjából több modulra lehet osztani. A modulok felosztása történhet felhasználói hozzáférés menetének szempontjából, vagy a program struktúrájának figyelembe vételével.

Az átláthatóság kedvéért először a szoftver strukturális felépítését, majd a felhasználó által bejárható UI felületet térképezzük fel.

### 2.2.1. Kommunikációs csatornák

A szoftver megfelelő megértése érdekében elsődleges cél, hogy a kommunikációs csatornák felvázolásával képet kapjunk a felépítéséről. Fent említett kritérium volt az, hogy a szimuláció lehetőleg minél szétagoltabban helyezkedjen el a szerveren, és erre a megfelelő megoldás, ha önálló szolgáltatásokra bontjuk a független részeket.



4. ábra: HWS szoftver építőelemei és kommunikációs struktúrája

Az ábrán a HWS szoftver teljes kommunikációs csatorna térképe látható. A felhasználók a HWS API-t használva kommunikálnak a szolgáltatásokkal valamint azokon keresztül a SensorWeb rendszerrel.

A jogosultsági és egyéb vállalati információk megszerzése érdekében a cég biztosított nekem egy az „Agrosense” mezőgazdasági projektben használt API –t, amelyen át kapcsolatba léphet a szoftver a megfelelő adatokkal.

Az ábrán jól kivehető, hogy a Test Encoder nevezetű szolgáltatás önálló TCP kapcsolatot épített ki a SensorWeb rendszerrel. Ez az a szolgáltatás, amely a szoftverben létrehozott majd elindított projektet a megfelelő formába alakítja és a csatornán keresztül az elküldi a méréseket.

A szolgáltatások létrehozásában megoldást nyújt a Visual Studioban is elérhető WCF (Windows Communication Foundation).

### **2.2.1.1. WCF (Windows Communication Foundation)**

A WCF egy a framework részeként települő osztály- és szolgáltatásgyűjtemény. Egy szervizorientált architektúrát valósít meg, amely olyan önálló egységekbe szervez alkalmazásokat, melyeknek viselkedésük, működésük jellemzői miatt szolgáltatásként lehet őket hívni.[3]

A szolgáltatásban futó metódusok, függvények a felhasználó számára transzparensnek, nem érzékel belőle semmit. Ezt kihasználva valósul meg a felhasználói felület (UI) és a logika szétválasztása egy adott programban.

A HWS szoftverben a szolgáltatások a szerveren futnak egymástól függetlenül. A Felhasználói felülettel, valamint egymással, API segítségével kommunikálnak. Számos előnnyel jár, ha szolgáltatásként fut egy szoftver logikája, a legfontosabb talán, hogy így csökkenthető a szoftver hardverigénye, mivel a kliensen csak UI fut, a szolgáltatások pedig a szerveren helyezkednek el.

A szolgáltatások használata azért is fontos, hogy a szoftver képes legyen működni akkor is, ha bármelyik része kiesik, illetve ne kelljen az egészet frissíteni, hanem elég legyen csak azt az egy modult, ami szükséges.

A szolgáltatások telepítését a command prompt –ban az „installutil + szolgáltatás neve” paranccsal lehet elvégezni, ami aztán bekerül a Windows szolgáltatásai közé.

### **2.2.1.2. Főbb szolgáltatások**

Az ábrán a több szolgáltatást is megfigyelhetünk. Ezek közül a legfontosabbak:

- HWS WCF Device
- HWS WCF User Management
- HWS WCF Project
- HWS WCF Test Encoder
- HWS WCF Sensor
- HWS WCF Sensor Web Connect

Ezekkel kommunikál a felhasználó közvetetten a UI felületről. Egyes szolgáltatások segítségével létrehozuk a projekteket, másokkal futtatjuk őket. A szolgáltatások mindegyike a függvények definiálására interfészosztályt alkalmaz.

Példa erre a HWS WCF User Management interfész osztálya:

```

namespace HardwareSimulation.UserManagementService
{
    /// <summary>
    /// User management interface service class
    /// </summary>
    [ServiceContract]
    public interface IUserManagement
    {
        /// <summary>
        /// Login interface function
        /// </summary>
        /// <param name="userName">user name</param>
        /// <param name="password">password</param>
        /// <returns>token and statuscode->LoginOk</returns>
        [OperationContract]
        HWSResultMessage Login(string userName, string password);

        /// <summary>
        /// Logout interface function
        /// </summary>
        /// <param name="token">user ID</param>
        /// <param name="userName">user name</param>
        /// <returns>statuscode->LogoutOk</returns>
        [OperationContract]
        HWSResultMessage Logout(string token, string userName);
    }
}

```

5. ábra: Részlet a UserManagement szolgáltatás interfész osztályából

Az ábrán látható, hogy ebben a szervizben csupán kettő darab függvény található, amelyek a szoftver be- és kijelentkezésének függvénydefiníciói. A bemeneti értékeket tartalmazó változók string típusúak. A login függvény bemenetként a felhasználónevet valamint a jelszót kéri. Ennek helyes megadásakor egy úgynevezett tokenel tér vissza a függvény valamint egy státuszkóddal. A token egy generált azonosító, a státuszkód pedig egy számösszeg. Ha a státuszkód megfelelő, akkor a hívás LoginOK értékkel tér vissza, így a bejelentkezés sikeres volt és a megkapott azonosítóval beléphet a rendszerbe a felhasználó. Amennyiben a felhasználónév-jelszó páros nem volt megfelelő, a státuszkód a LoginNOk –nak megfelelő üzenetet küld vissza és a nem kap a felhasználó sem tokenet sem hozzáférést a rendszerhez. A státuszkódok számsorozatának megfelelőjét egy enum tartalmazza, ez alapján történik az értékvi visszanyerés.

A HWS WCF UserManagement kommunikál az AgroSense API-val mivel a bejelentkezéshez szükséges jogosultságokat azon keresztül érheti el.

### 2.2.1.3. HWS API

A HWS API kezel minden kommunikációt a szolgáltatások, felhasználói felület és az AgroSense API között. Ha bármilyen kommunikáció megindul két szolgáltatás között az a szimulátor API-ján keresztül megy.

Ennek a megvalósításnak az oka az, hogy a későbbi fejlesztések során tervben van a UI webes megoldása.

Az alábbi ábrán egy részlete látható az algoritmusnak, amely azért felelős, hogy felismerje és elkapja azokat az URL-eket, amik az adott függvényt működésbe hozzák.

```
namespace HardWareSimulation.API.Controllers
{
    /// <summary>
    /// User management API Controller.
    /// </summary>
    public class UsersController : ApiController
    {
        /// <summary>
        /// API login
        /// </summary>
        /// <param name="userName">username</param>
        /// <param name="passWord">password</param>
        /// <returns>LoginOk(statusCode) and Token(Data)</returns>
        [System.Web.Http.HttpGet]
        public HWSResultMessage Login(string userName, string passWord)
        {
            return UserModule.Login(userName, passWord);
        }
    }
}
```

6. ábra: Részlet az API Controller osztályából

Itt megint csak a bejelentkezés látható. Ha olyan URL megy ki a megfelelő paraméterekkel, ami a login függvényt hívja, akkor az API működésbe lép. Elkapja az URL-t és a megfelelő irányba, függvények feltöltésével irányítja tovább.

<http://localhost:62661/HardWareSimulation.API/usermanagement/Login?username=&password=>

7. ábra: Példa egy API hívásos URL-re

Feljebb egy tipikus bejelentkezéskor elküldött URL található. Ezen jól látható a függvény neve valamint a változók neve. A bejelentkezés GET metódust alkalmaz, ami nem túl biztonságos, de tekintve hogy a vállalaton belül kerül felhasználásra, így vállalható rizikót jelent.

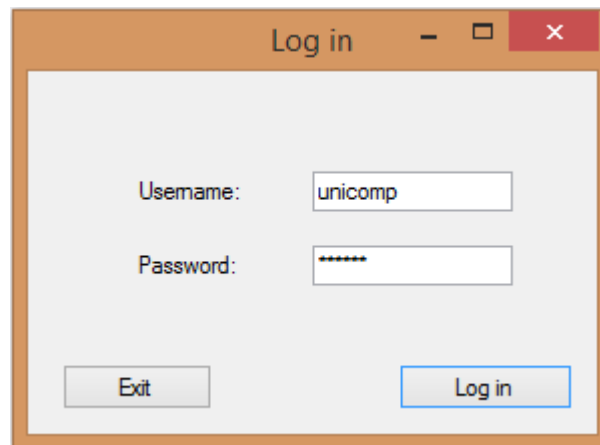
## 2.3. HWS szoftver moduljai UI megközelítéssel

Egy másik megközelítéssel a UI felületen pásztázhatunk végig. Így a következő lehetséges modulokat tudjuk megkülönböztetni egymástól:

- Bejelentkezés
- Folyamatok panel
- Elérhető szenzorok megtekintése
- Projekt létrehozása
- Projekt összegzése
- Mentett projekt betöltése
- Log üzenetek kezelése

### 2.3.1. Bejelentkezés

A program elindításával a bejelentkező ablak fogad minket elsőként. A bejelentkezéshez meg kell adni a megfelelő felhasználónevet és a hozzá tartozó jelszót. Sikeres bejelentkezés esetén a főablakhoz jutunk, ami a „folyamatok panel” nevet viseli. Sikertelen bejelentkezés esetén hibaüzenet jelenik meg a megfelelő státuszkóddal. Ha a hálózati vagy egyéb probléma adódik a megfelelő státuszkóddal rendelkező üzenet jelenik meg. A jelszót és a felhasználónevet a központi jogosultságkezelő adatbázisban keresi a rendszer és ez alapján értékeli ki.



8. ábra: Bejelentkező ablak

### 2.3.2. Folyamatok panel

A sikeres bejelentkezés után a folyamatok panelre lépünk. Itt egy listában az aktuális már létrehozott tesztek találjuk, amelyekről meg tudjuk állapítani:

- A projekt nevét
- A létrehozó felhasználó nevét
- A létrehozás dátumát
- A projekt futásának státuszát

A lista alatt található gombok segítségével menedzselhetjük a kiválasztott tesztek. A gombokon jellegzetes alakzatok elhelyezésével tudatja a program a felhasználóval a lehetséges opciókat. Lehetőségünk van:

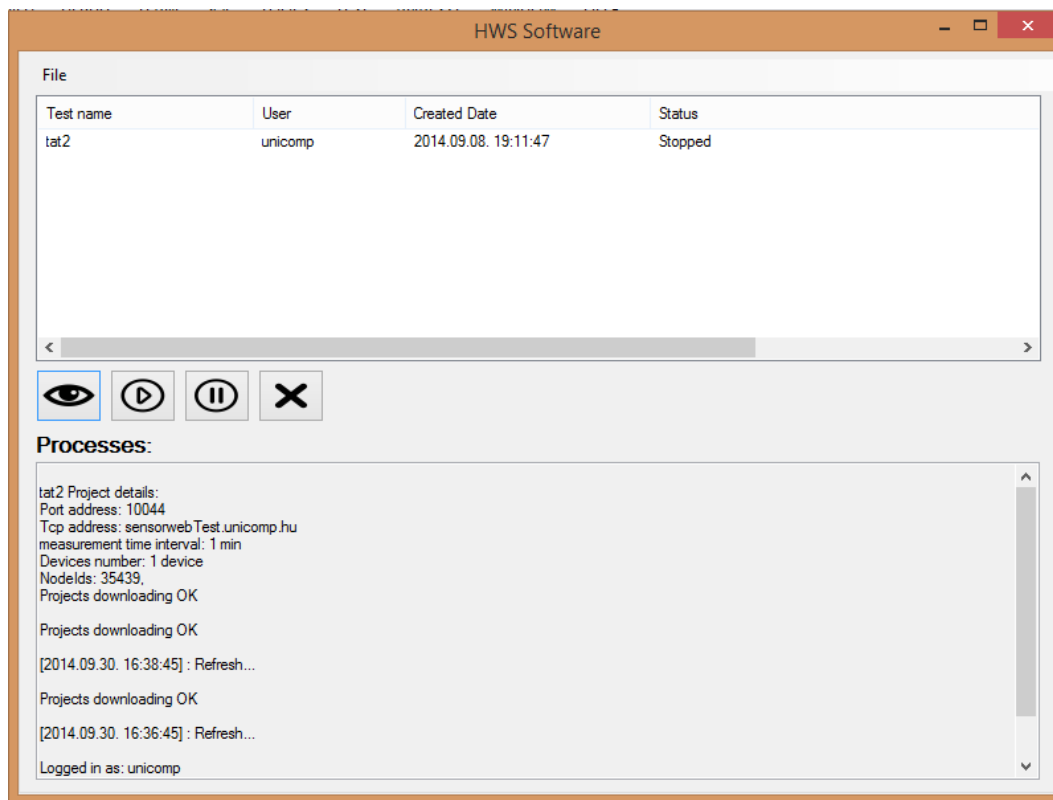
- Megtekinteni az adott teszt tulajdonságait
- Törölni az adott tesztet
- Elindítani az adott tesztet
- Megállítani az adott tesztet

A gombok alatt egy rendszerüzeneteknek és állapotjelentéseknek fenntartott felület található.

A szoftver UI felülete egy úgynevezett vékonykliensként működik. Több felhasználó is bejelentkezhet egy időben külön-külön UI felülettel. Ilyenkor az aktuális projektek listában ugyanazokat a tesztek figyelhetjük meg, hiszen a szoftver a szerveren lévő szolgáltatásokból frissíti bizonyos időközönként a listát. Az idegen tesztek más felhasználó leállíthatja, de le nem törölheti azokat, amelyek nem ő hozott létre.

A kiválasztott projektet abban az esetben indíthatjuk csak el, ha az időzítője megegyezik a többi már elindított projektek időzítőivel. Az időzítő szabja meg, hogy a tesztet milyen intervallumban küldje el a méréseit.

Ha nem tudjuk, hogy a kiválasztott projekt milyen paraméterekkel van felruházva, egyszerűen ki kell jelölni és a megtekintés (szem) gombra kattintani. Az itt található információk alapján eldönthetjük, hogy szükség van-e rá.



9. ábra: Folyamatok ablak

### 2.3.3. Projekt létrehozása

A legördülő menüre való kattintás után, ha új projektet szeretnénk létrehozni, akkor kiválaszthatjuk a „Create New Test” gombot. Az első panel, ami megjelenik ilyenkor az a szenzorpanel, ahol kiválaszthatjuk, hogy a szimulálandó eszközünk milyen érzékelőket tartalmazzon.

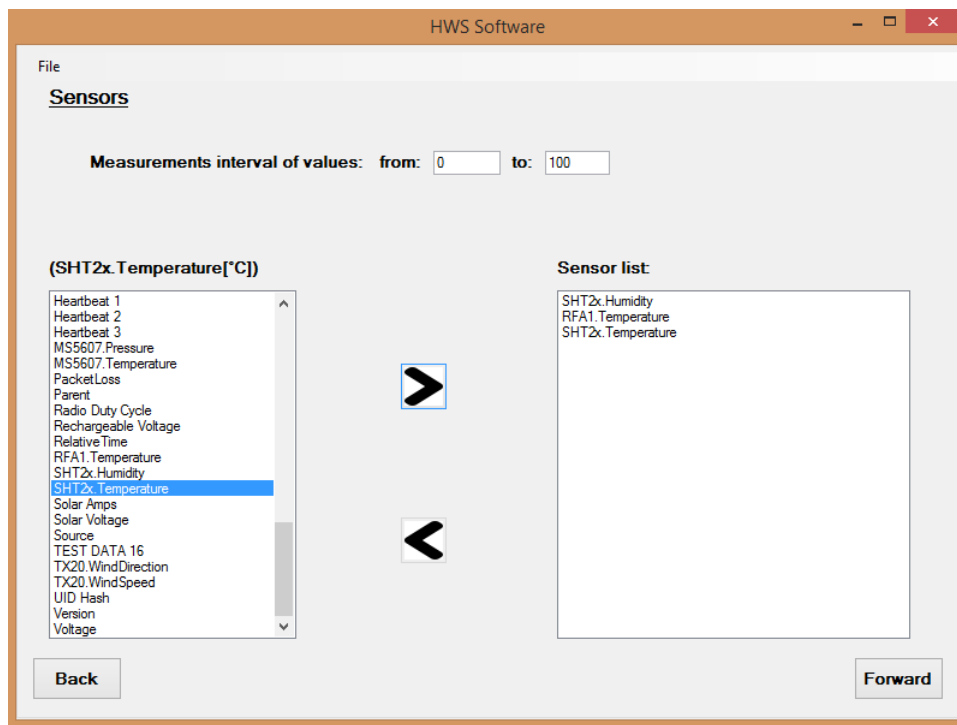
A vállalat a különböző projektek során megannyi szenzort készített el. Jelenleg valamivel több, mint százféle érzékelőt jegyeztek be. Mindegyikhez tartozik leírás. A leírásban a következő tulajdonságok szerepelnek:

- Szenzor név
- Szenzor azonosító
- Szenzor típus
- Szenzor típus azonosító
- Szenzor maximális mérési értéke

A tulajdonságok adatainak felhasználásával építhető be az eszközbe a kijelölt érzékelő. Gyakorlatilag nincs megszabva, hogy mennyi szenzort telepíthetünk egy node-ba, de általában a valós projektekhez mérve van összerakva. Lehetőségünk van ugyanazon szenzorból akár többet

is belerakni az eszközbe, hiszen ha például egy rúdra tesszük az eszközt és aztán a rúd végét a földbe helyezzük, akkor képesek vagyunk a földfelszín alatt és felett is hőmérsékletet mérni.

Az alábbi ábrán látható a szenzorok listája, valamint az a lista ahova áthelyezhetjük egy kiválasztott érzékelő másolatát, ezenkívül itt tudjuk megadni a kiválasztott eszköz mérési intervallumát egyenként. A jövőben az elindított projekt szereplői ezen a beállított intervallumon belül fognak egy véletlenszerű mérési értéket generálni.



10. ábra: Szenzor panel

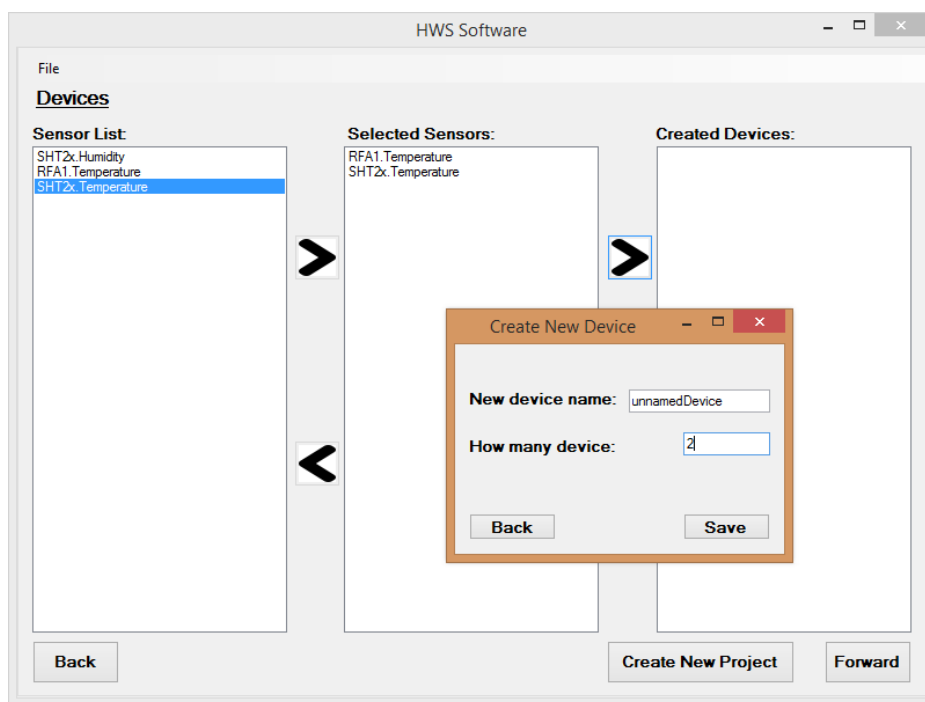
A „Back” gombbal visszajutunk a folyamatok panelra, de akkor minden eddigi munkánk elveszik. A „Forward” gombra kattintva az eszközök panelra jutunk, ahol további beállítási lehetőségeink vannak.

#### 2.3.4. Eszközök panel

Az eszközök panelen, ahogy a nevéből is következtethetünk rá, elhelyezhetjük a kiválasztott szenzorokat az eszközökben. Az alábbi ábrán három lista látható. A baloldali oszlopban láthatóak azok az érzékelők, amiket beválogattunk a listából az előző panelon. A középső oszlopba helyezhetjük azokat a szenzorokat, amelyeket tartalmaznia kell annak az eszköznek, amit létre akarunk hozni. A jobboldali oszlopban a létrehozott eszközök listája található. Az oszlopok közötti gombok segítségével menedzselhetjük a kiválasztott elemek felvételét és törlését. Amennyiben a jobb felső gombra, vagyis az „eszköz hozzáadásá” –ra kattintunk, előugrik egy ablak, ahol felkínálja nekünk a program, hogy elnevezzük az eszközt,

valamint, hogy hány darab ilyen felépítésű eszközt szeretnénk létrehozni. A megfelelő adatok bevitelével és a „Save” gombra kattintva megjelennek az elnevezett eszközök a jobb oldali listában. Ha több ugyanolyan eszközt hozunk létre, akkor az eszközök neve után automatikusan egy sorszám kerül megkülönböztetésük végett. Ezek az eszközök az alábbi tulajdonságokkal rendelkeznek egyenként:

- Eszköz név
- Eszköz típusa
- Eszköz azonosítója
- Eszközben lévő szenzorok száma
- Eszközben lévő szenzorok listája



11. ábra: Eszközök panel

A „Back” gomb lenyomására visszakerülünk a szenzor panelre, ahol további szenzorokat vehetünk fel a listába. A „Create New Project” gomb lenyomásával az összegzés panelhez jutunk, ahol menthetjük a létrehozott eszközeinket ezáltal új projektet hozunk létre.

A „Forward” gomb jelenleg még nem üzemel, mivel az egy későbbi továbbfejlesztés eleme. A jövőben képesek leszünk nagyobb csoportokba rendezni a létrehozott eszközeinket, ezáltal áttekinthetőbb és adminisztrálhatóbb lesz a rendszer.

### 2.3.5. Összegzés panel

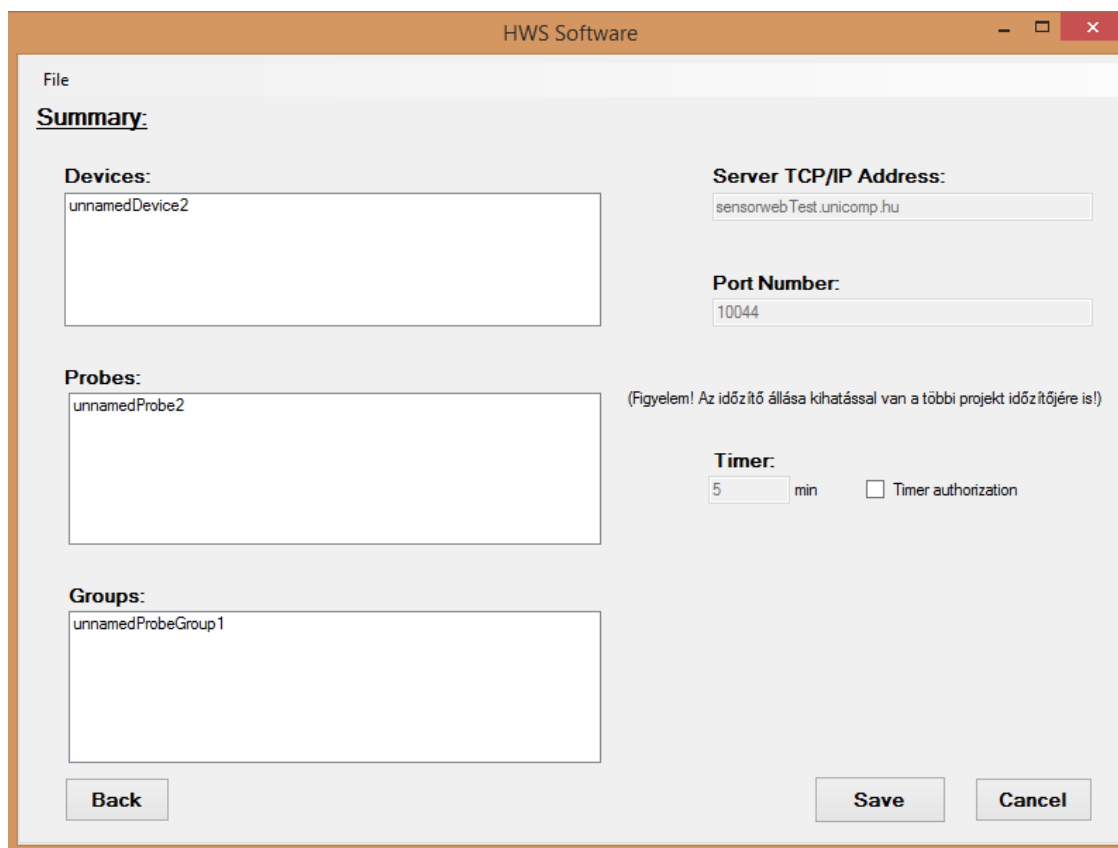
Az következő oldal, amire jutunk az az összegzés panel. Az alábbi ábrán, a baloldalon láthatóak a létrehozott eszközök listája, valamint a jövőben kialakítható „Szondák” és „Szondacsoportok” listája. Ezen a felületen adhatjuk meg azt a címet és port-számot, amelyre a létrehozott projekt küldeni fogja a méréseket.

További konfigurációs lépés lehet az alapértelmezett időzítő értékének a megváltoztatása. Az időzítő határozza meg hogy a létrehozott projekt, milyen intervallumonként küldjön méréseket a SensorWeb rendszerbe.

Az idő, célcím és port-szám egy előre beállított adatot tartalmaz, amely a leggyakoribb megvalósítás, de megváltoztatható.

A „Cancel” gombra kattintva törölhetjük az eddigi létrehozásainkat és visszakerülünk a folyamatok panelre. A „Back” gomb lenyomásával az eszköz beállításokhoz jutunk.

A „Save” gomb megnyomására egy előugró ablak jelenik meg, ahol nevet adhatunk a projektnek és elmenthetjük egy meghatározott mappába, ahonnan a későbbiekben is betölthető lesz. A tárolás után automatikusan visszakerülünk a folyamatok panelre ahol már az aktuális projektek listában megtalálható a létrehozott munka.



12. ábra: Összegzés panel

A mentéskor megkapják az eszközök az azonosítóikat, amelyeket a vállalat adatbázisa generál és oszt ki. Ennek az azonosítónak a használatával ismeri fel a SensorWeb rendszer az eszközöket, és ez alapján adminisztrálja a beérkező méréseket.

A dolgozat elején szó esett a vezetékek nélküli hálózatok működéséről. A hálózat klaszteresen épül fel. A klaszterben helyet foglal egy bázis is, ami ugyanúgy szenzoros eszköz azzal a különbséggel, hogy egy kitüntetett szerepet kapott a többi mellé. Az egyedi feladat nem más, minthogy begyűjtse a hálózatban szereplő eszközök méréseit és bizonyos időközönként az adatgyűjtőbe irányítsa azt.

A bázis kiválasztása úgy történik, hogy a projekt mentésekor, az eszközazonosítók csatolásakor automatikusan a legelső létrehozott eszköz lesz a szimulációban.

### 2.3.6. Mentett projekt betöltése

A mentett fájl „.json” kiterjesztésű, ami könnyen feldolgozható. Az alábbiakban egy ilyen mentés tartalma látható.

```
[{"projectName": "vegso",
{"deviceName": "unnamedDevice4", "deviceTypeBase": false, "nodeId": "35597",
"selectedDevice": "null", "deviceSensorsNumber": "2", "encodedMeasurement": null,
"sensorsList": [
{"descName": "RFA1.Temperature", "descId": null, "descFieldUnit": null,
"descFieldLength": "2", "sensorValueMin": "0", "sensorValueMax": "100", "descFieldId": "7",
"selected": null},
{"descName": "SHT2x.Humidity", "descId": null, "descFieldUnit": null,
"descFieldLength": "2", "sensorValueMin": "0", "sensorValueMax": "100", "descFieldId": "3",
"selected": null}]]], "selectedProbe": "null"}],
"dateTime": "2014-10-22T16:53:37.7350246+02:00",
"userName": "unicomp", "nodeIdValue": "4", "projectStatus": false,
"serverAddress": "ucwsn.unicomp.hu", "portAddress": "10045", "timerValue": "1"}]
```

13. ábra: Mentett projekt tartalma

A legördülő menüben található „Open Test” nevű gombra kattintva megjelennek egy ablakban az eddigi elmentett projektek. A betöltött projekt az aktuális projekteknek fenntartott listába töltődnek fel.

### 2.3.7. Log üzenetek

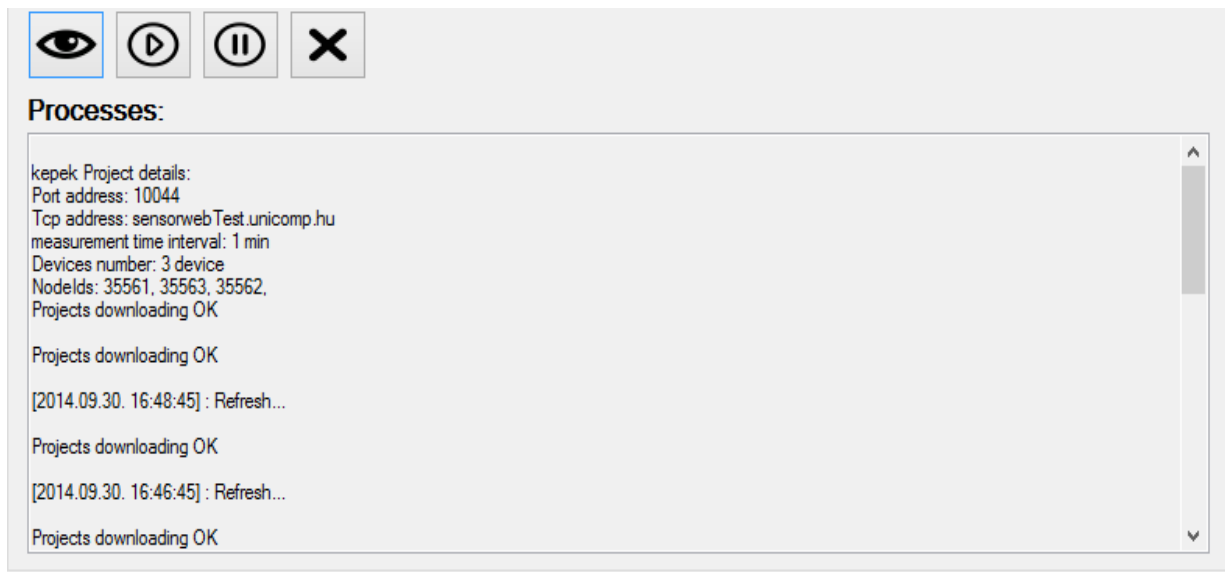
A log üzenetek nagyon fontos részei a szoftvernek. Általuk betekintést nyerhetünk a folyamatok mögé és a hibák feltérképezhetőségét is megkönnyíti. Két külön log van jelen a szoftverben. Az egyik a folyamatok felületen is látható rendszerüzeneteknek fenntartott log. A másik egy fájlba írásos log a folyamatok minden lépésének naplózására.

A Microsoft Visual Studio C#-os környezete lehetővé teszi a szálkezelést.[4]

A folyamatok panelon működtetett log, a párhuzamos megvalósítás által bizonyos időközönként frissíti a rendszert, és ha bármiféle változás történt, legyen az hálózati vagy szolgáltatások által okozott üzemzavar, ez kiírásra kerül és a szükséges módosítások, így például a futó projektek leállítása, automatikusan végrehajtnak.

Ebben a listában kap helyet az az információ is, amelyet a projekt megtekintése (szem) gombbal listázhatunk ki és a projekt tartalmát sorolja fel.

Amikor megszakad a kapcsolat a szolgáltatások és a UI felület között, letiltódnak az olyan műveletek és lehetőségek amelyek a hálózatot igénybe veszi. Az alábbiakban látható a folyamatok panelen jelenlévő log lista.



14. ábra: Rendszerüzenetek log lista

A Szolgáltatások által használt log a szerveren történik kiírásra egy fájlba. Itt is a log párhuzamos megvalósítása lévén egy saját időzítő működik minden szolgáltatásban, amely félpercenként kiírja az üzeneteket fájlba.

A fájl neve tartalmazza mindig az adott dátumot. A fájlba olyan üzenetek kerülnek kiírásra, amikkel vissza lehet ellenőrizni a projektek ütemezését és a mérések értékét. Hiba esetén ide bekerülnek a hibaüzenetek, a leállások és az újraindítások is.

Az alábbiakban látható egy részlet a szolgáltatások által üzemeltetett log fájlból.

```
[2014.11.03. 11:33:23] : vegso project started...
[2014.11.03. 11:33:23] : Connecting to sensorweb...
[2014.11.03. 11:33:23] : Connecting successful...
[2014.11.03. 11:33:23] : vegso project sending...
[2014.11.03. 11:33:23] : 35594(base) creating..
[2014.11.03. 11:33:23] : RFA1.Temperature: 8500
[2014.11.03. 11:33:23] : SHT2x.Humidity: 1800
[2014.11.03. 11:33:23] : 35596 creating..
[2014.11.03. 11:33:23] : RFA1.Temperature: 6000
[2014.11.03. 11:33:23] : SHT2x.Humidity: 2300
```

15. ábra: Szolgáltatások log üzenetei fájlban

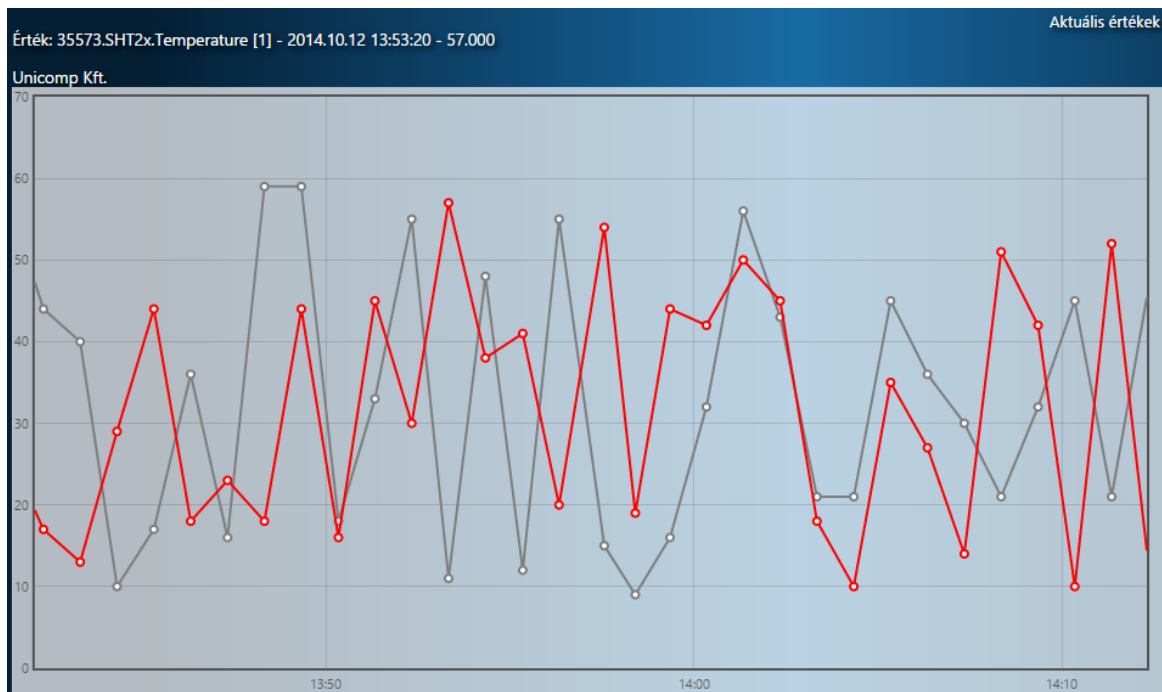
### 2.3.8. Kijelentkezés

A szoftverből úgy lehet kilépni, ha a legördülő menüben található „Log out” gombra kattintunk. Ilyenkor bezárul a felhasználó UI felülete. A szolgáltatások a szerveren a háttérben futnak, így a felhasználó kijelentkezése után sem áll le a szoftver ezen része. A projektek ezután is ugyanúgy futnak a szerveren.

Bejelentkezéskor az aktuális projektek is betöltődnek.

### 2.4. Eredmények

A szoftver elsődleges fejlesztési időszaka 2014.09.05 –én lezárult és a rendszer a belső rendszereken megkezdte a működését. Az alábbi ábrán a SensorWeb rendszer felhasználói felületén megtekinthető HWS által készített mérési eredményeket figyelhetjük meg.



16. ábra: Egy eszköz hőmérsékleti szenzorjainak idő függvénybeli értéke

Az felső ábrán két szenzor segítségével végrehajtott hőmérsékletmérés eredményei láthatóak az idő függvényében. Az eszköz, amely küldte az adatokat a 35573-s azonosító nevet viseli. A projekt elkészítésekor a hőmérsékleti szenzoroknak be lett állítva, hogy 10 és 60 fok között egyenletes eloszlással keletkezzenek.

A mérések percenként kerülnek továbbításra.

A következő ábrákon a szerver különböző teljesítmény grafikonjait lehet látni. Ezek az értékek akkor születtek, mikor a szimuláció éppen nagymennyiségű mérési adatot közölt periodikus időosztással az adatgyűjtő szerverrel.



17. ábra: CPU-kihasználtság a szerveren



18. ábra: Memóriakihasználtság a szerveren

A fenti ábrákon jól kivehető a szimuláció működésének üteme. Amikor magas a CPU- és memóriakihasználtság, olyankor kommunikál a szerverrel a node bázis.

A szerverrel való kommunikáció nagymennyiségű adatoknál a belső rendszer tesztelésekor nagymértékű hardver ráfordítást igényelt szerveroldalon.

### 3. Konklúzió

A HWS szoftver kifejlesztésének megbízásakor számos addigi problémára vártak megoldást a vállalat körében. A fejlesztés alatt sok új ötlet és megkötés is hozzáadódott az addigi felépítéshez. Ezek egy része implementálásra került, más része egy ezután elkészítendő verzió alapjait képezik majd.

A szoftver teszt-üzemmódban való futásakor olyan problémák kerültek elő a Sensorweb-el kapcsolatban, amelyekről eddig nem volt tudomása a vállalatnak. A 17.-ik és 18.-ik ábrán fellelhető szélsőséges értékek a rendszer felülvizsgálatát eredményezték.

Egy többnapos teszt üzem mód során kiderült, hogy volt a SensorWeb rendszernek egy olyan holtidőszaka, amikor nem fogadta a beérkező adatokat, pedig a log üzenetek alapján ezek sikeresen átmentek.

A szoftver mindazonáltal nem képes igazán természetes értékeket mérni, így sohasem lehet összetéveszteni a valós környezettel.

A programban létrehozott projekt méretezhetősége korlátos. A korlátok átlépése üzemzavart okozhat mind a szoftver mind pedig a SensorWeb rendszerében, ezért a mesterségesen beállított határértékeket átlépni nem lehet.

A HWS létrehozásának a célja, hogy szélsőséges, a normálistól eltérő mérési eredmények kerüljenek közlésre. A program a megadott feltételeknek megfelelni bizonyult.

A vállalatnak több olyan céges projektje van ahol ezt szoftvert tesztelésre lehet használni. A könnyen ellenőrizhető, megbízható értékek kielemezésének segítségével az adott céges projektek tesztelési időszaka megrövidülhet.

## 4. Összefoglalás

A HWS szoftver fejlesztési ideje összesen 5 hónapot ölelt fel. Ez alatt az idő alatt sikerült implementálni a kitűzött feladatokat a rendszerbe. Az éles rendszer tesztelése 2014 december elsejével befejeződik.

A tesztelés alatt is rengeteg problémára derített fényt eddig a program. A szoftvert teszt üzemmódban is lehetett már fontos projektek ellenőrzésére használni.

Segítségével költséghatékonyabb megoldást kínál a vállalatok számára a vezeték nélküli hálózatok tesztelésében, valamint az eredményeket feldolgozó rendszer stabilitásának vizsgálatában.

Jelenleg függőben vannak olyan jövőbeli fejlesztések, amelyek most már csak a hivatalos verzió után egy következő továbbfejlesztés részei lehetnek csak. A tesztelés során előkerülő ötletek a következők:

- Az eszközök nagyobb mértékű csoportosíthatósága.
- Báziseszköz kijelölése manuális módszerrel.
- Mérési értékek előállítása nem csak véletlen-, hanem előre beállított módon is.
- Eszközazonosítók karbantartása, újrafelhasználása, mert ha például törölünk egy projektet, akkor az eszköz azonosítók kihasználatlanul maradnak.

Bár a szoftver egy bizonyos vállalat rendszerére lett optimalizálva, mégis pár hónapos fejlesztéssel akár más céges környezetbe áttelepíthető, ha a szükséges információk segítségével.

A fejlesztés ideje alatt részt vettem egy vállalat szoftverfejlesztő részlegének munkálataiban. Megfigyelhettem egy csapat stílusát, stratégiáját. Lehetőséget kaptam annyi tudást és tapasztalatot gyűjteni, amennyit csak ezen idő alatt lehetett.

## 5. Ábrajegyzék

|                                                                               |    |
|-------------------------------------------------------------------------------|----|
| 1. ábra: Egy node hardveres struktúrája .....                                 | 5  |
| 2. ábra: WSN rendszer hálózati struktúrája .....                              | 6  |
| 3. ábra: SensorWeb rendszer struktúrája .....                                 | 10 |
| 4. ábra: HWS szoftver építőelemei és kommunikációs struktúrája.....           | 16 |
| 5. ábra: Részlet a UserManagement szolgáltatás interfész osztályából.....     | 18 |
| 6. ábra: Részlet az API Controller osztályából.....                           | 19 |
| 7. ábra: Példa egy API hívásos URL –re .....                                  | 19 |
| 8. ábra: Bejelentkező ablak.....                                              | 20 |
| 9. ábra: Folyamatok ablak .....                                               | 22 |
| 10. ábra: Szenzor panel.....                                                  | 23 |
| 11. ábra: Eszközök panel .....                                                | 24 |
| 12. ábra: Összegzés panel .....                                               | 25 |
| 13. ábra: Mentett projekt tartalma .....                                      | 26 |
| 14. ábra: Rendszerüzenetek log lista .....                                    | 27 |
| 15. ábra: Szolgáltatások log üzenetei fájlban .....                           | 28 |
| 16. ábra: Egy eszköz hőmérsékleti szenzorjainak idő függvénybeli értéke ..... | 28 |
| 17. ábra: CPU kihasználtság a szerveren.....                                  | 29 |
| 18. ábra: Memóriakihasználtság a szerveren .....                              | 29 |

## 6. Irodalomjegyzék

- [1] **WANG Weihong, LIN Yubing:** *The Application of Wireless Sensor Network on Regional Environmental Protection*, College of Software, Zhejiang University of Technology, Hangzhou, P.R.China, 310032, [Wwh@zjut.edu.cn](mailto:Wwh@zjut.edu.cn)
- [2] *What is Simulation?* [online] [Hivatkozva: 2014. november 6.]  
<http://www.goldsim.com/Web/Introduction/Simulation/>
- [3] *Communication Foundation – SOA* [online] [Hivatkozva: 2014. november 6.]  
<http://aries.ektf.hu/~hz/pdf-tamop/pdf-03/html/ch11.html>
- [4] **István, Reiter:** *C# programozás lépésről lépésre*. Budapest: Jedlik Oktatási Stúdió Kft., 2012.