Készítette

Fejér Norbert

Elekes Attila

2018.12.17

1. Bevezetés

1.1. A szoftver rövid leírása

Napjainkban egyre nagyobb teret hódítanak meg az okos eszközök, és számos lehetőséget nyújtanak mind a hétköznapi élet, mind pedig az IoT területén. Az egyik ilyen okoseszköz, amely a nap talán legtöbb részében van velünk, az okosóra, illetve okoskarkötő. Bátran elmondható, hogy ezek az egyik legelterjettebb okoseszközök.

Számos megoldást láthatunk ezek megvalósítására, projektünk során mi is egy okos karkötő megvalósítását tűztük ki célul. A projekt névválasztása (SmartBand) talán hűen tükrözi, hogy mire is hivatott az általunk összeállított eszköz. Számos megoldást tanulmányoztunk, annak érdekében, hogy projektünket kellő képpen tudjuk kivitelezni, viszont rengeteg olyan részt is tartalmaz, amely teljes egészében a mi munkásságunk fedi le.

1.2. Célkitűzés

Projektünk során meglevő alkatrészek felhasználásával egy olyan eszközt szerettünk volna összeállítani, ami képes segíteni a hétköznapi felhasználó munkáját, különböző adatokat gyűjt a felhasználó viselkedéséről, majd statisztikák segítségével segíti a felhasználó életminőségének a javításán.

Eszközünk jelen állapotában egy egyszerű, kezdetleges okosóraként használható, sajnos az alkatrészek nagy mérete miatt nem rendelkezik a piacon levő eszközök kicsiny méretével, viszont folyamatban van az energiafelhasználás optimizálása, illetve az alkatrészek lecserélése minél kisebb méretre.

Eszközünk két képernyő állapottal rendelkezik. A főképernyőn az aktuális idő, dátum, illetve nap együttesét láthatjuk, emellett lehetőségünk van követni, hogy az adott nap hány lépést tettünk meg. A képernyő felső sarkában a eszközünk kapcsolatának állapotát tekinthetjük meg, vagyis láthatjuk hogy csatlakoztatva van-e egy okostelefonhoz, esetlegesen táblagéphez. Két nyomógomb áll rendelkezésünkre ahhoz, hogy navigálni tudjuk a képernyő állapotai között, illetve különböző

műveleteket hajthassunk végre. Az eszköz második képernyőjén egy stopper óra áll rendelkezésünkre, amivel különböző aktivitásaink időtartamát mérhetjük.

Az eszköz mellé kifejlesztettünk egy Androidos applikációt is, mely segítségével a felhasznánló követheti aktivitásait, statisztikát nézhet meg a lépéseiről és új célokat állíthat fel. Az applikáció és eszköz közötti kommunikációt Bluetooth technológia segítségével valósítottuk meg.

2. Követelmény specifikáció

2.1. A szoftver rövid követelménye

Az általunk készített eszköz képes legyen adatokat gyűjteni és megjeleníteni a felhasználóról, illetve kommunikáció megvalósítása az eszköz és egy Android operációs rendszert futtató okostelefon között az általunk fejlesztett applikáció segítségével.

2.2. Kezdeti elvárások

- Hardware szinten:
  - Képes legyen az idő, dátum és nap megjelenítésére
  - Detektálni és számolni tudja a felhasználó lépéseit
  - Értesítéseket jelenítsen meg az applikáción keresztül
  - Stopper funkcióval rendelkezzen
  - Komminikációra legyen képes más eszközzel
- Software szinten:
  - Adatok fogadása és küldése az eszköznek
  - Értesítések, hívások, illetve üzenetjelzések továbbítása az eszköznek
  - Felhasználói statisztikák készítése és megjelenítése

2.3. Felhasználói követelmények

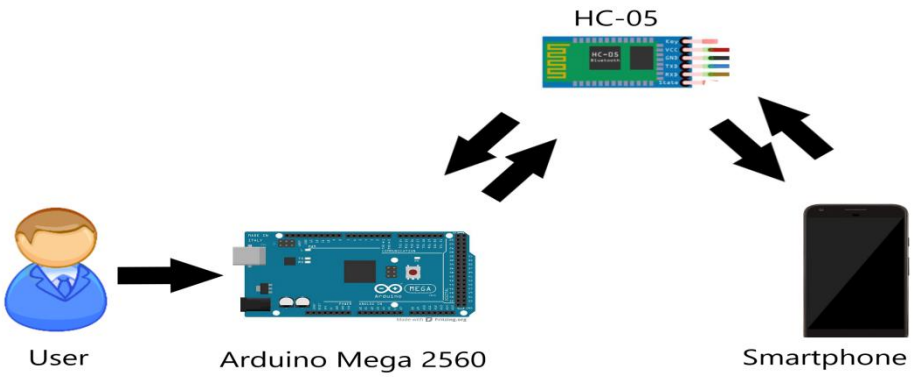
Hardware szinten:

1. Felhasználói felület: a kijelzőn (fekete-fehér képernyő) angol nyelvű előre definiált szöveget láthatunk, ami értesítések megjelenítésekor háttérvilágítást biztosít, hogy könnyebben vegyük észre az bejövő értesítéseket
2. Felhasználóbarát: a felülete intuitív, pillanatok alatt elsajátítható a kezelése a nyomógombok segítségével
3. Operációs rendszer: eszözünkön futó program jelenleg Arduino Mega 2560-ra van optimizálva, így az adott forráskód helyes működése érdekében ennek használatát ajánlnuk
4. Navigálás a képernyő állapotai között: egyszerűen végezhető a felszerelt két nyomógomb segítségével
5. Szűkséges erőforrás: eszközünk tartalmaz egy beépített erőforrást (külső hordozható akkumulátor), ez biztosítja a rendszer energia igénylésének biztosítását, melynek 3 naponta történő újrafeltöltése szűksége.

Szoftware szinten:

1. Felhasználói felület: alkalmazásunk egyetlen képernyőt tartalmaz, melyen egy statisztikát látatunk, az elmúlt időszakban történt aktivitásunkról, illetve napi célt tudunk felállítani magunknak és láthatjuk milyen mértékben sikerült ezt teljesítenünk
2. Felhasználóbarát: az alkalmazás felületének megalkotása során letisztult formákkal dolgoztunk, látványos vezérlőgomokat és kapcsólókat használva
3. Operációs rendszer: az applikáció telepítéséhez a felhasználó minimálisan Android 5.1.1-es operációs rendszerrel kell rendelkezzen
4. Navigálás: könnyedén történik, navigálást megkönnyítő elemekkel kiegészítve

2.4. Rendszerkövetelmények



Ábra 1

A fenti ábrán látható projektünk architektúrális felépítése. A felhasználó a kész eszközt (SmartBand) használva csatlakozhat okostelefonjához bluetooth kommunikáción keresztül. A kommunikáció bidirekcionális, vagyis küldhet adatot a telefonjának és fogathat is attól.

Funkcionális követelmények

Hardware szinten az alábbi eszközök(modulok) meglétele szükségeltetik:

a) Arduino Mega 2560 (teszteléseink ilyen eszközön történtek, viszont a felhasznált memória szempontjából nézve forráskózunk futtatható egyaránt Arduino Uno vagy Arduino Nano eszközön is) - a program futtatására és adatok feldolgozásához
b) 9 Axes Motion Shield - lépés detektálásra és számoláshoz
c) HC-05 Bluetooth Module - a kommunikáció kialakításához az eszköz és az applikáció között
d) 2 darab egyszerű nyomogómb - a navigálás megoldásához
e) LCD5110 kijelző - az adatok megjelenítéséhez és a felhasználóval való interakció kivitelezéséhez
f) Külső hordozható akkumulátor - az eszköz áramellátásának biztosításához

Software szinten az alábbiakra lesz szükségünk:

a) Legalább Android 5.1.1 (API level 22) operációs rendszert futtató okostelefon/ táblagép

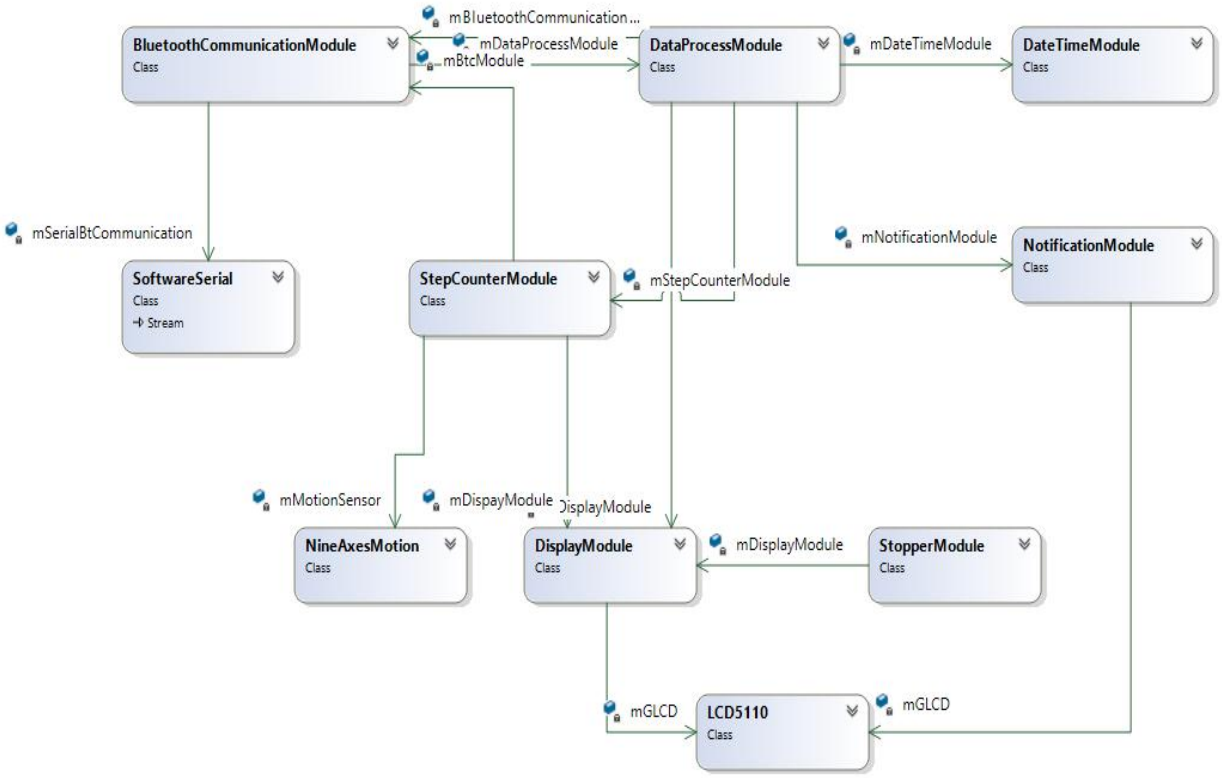
Nem funkcionális követelmények

1. Válaszidő: rendkivüli hangsúlyt fektettünk, hogy alkalmazásunk és eszközünk külön-külön, illetve együttes használat esetén is hatékony és gyors legyen, a felhasználó kényelme érdekében
2. Biztonság: az adatok helyes célbaérkezését és helytelen adatok kiszűrését külső libray-k segítségével oldottuk meg
3. Értesítések: értesíteni szeretnénk a felhasználót a bejövő hívásról, érkezett új üzenetről vagy esetlegesen kitűzött napi céljaik eléréséről, adott statisztikákról
4. Használt programozási nyelvek:
   a) Az eszközön futtatott programrészlet Android Studio környezetében megírt C++ alapú kódot tartalmaz, az osztályok és libray-k fejlesztése Sublime Text szövegszerkesztő segítségével történt, GCC c++14 kompilátort használva
   b) Az eszközhöz tartozó applikáció megírása Android Studio fejlesztőkörnyezetben történt, Java 8 technológiára alapozva
5. Használt külső libray-k az eszközön futó program megírása során:
   a) <Arduino.h> - az Arduino sajátos függvényeinek hívása végett(pinek olvasása, Serial monitorra való kiírás hibakeresés céljából)
   b) <LCD5110_Basic.h> - képernyőre való kiírást kezelő függvények gyűjteménye
   c) <Thread.h> - párhuzamosítás végett használtunk, habár eszközünk nem futtat való operációs rendszert, szükségünk volt több folyamat egyszerre végbemenő látsztát kelteni, így ez a libray volt a segítségünkre, mely a millis() beépített függvényt használj az adott folyamatok párhuzaosításának szimulálását.
   d) <SimpleTimer.h> - segítségével adott függvény adott időközönként való ötemezését állíthattuk be
   e) <NineAxesMotion.h>

f) <Wire.h> - az imént említett két függvény segítségével tudtuk a Motion Sensor által szolgáltatott adatokat megjeleníteni, majd feldolgozni

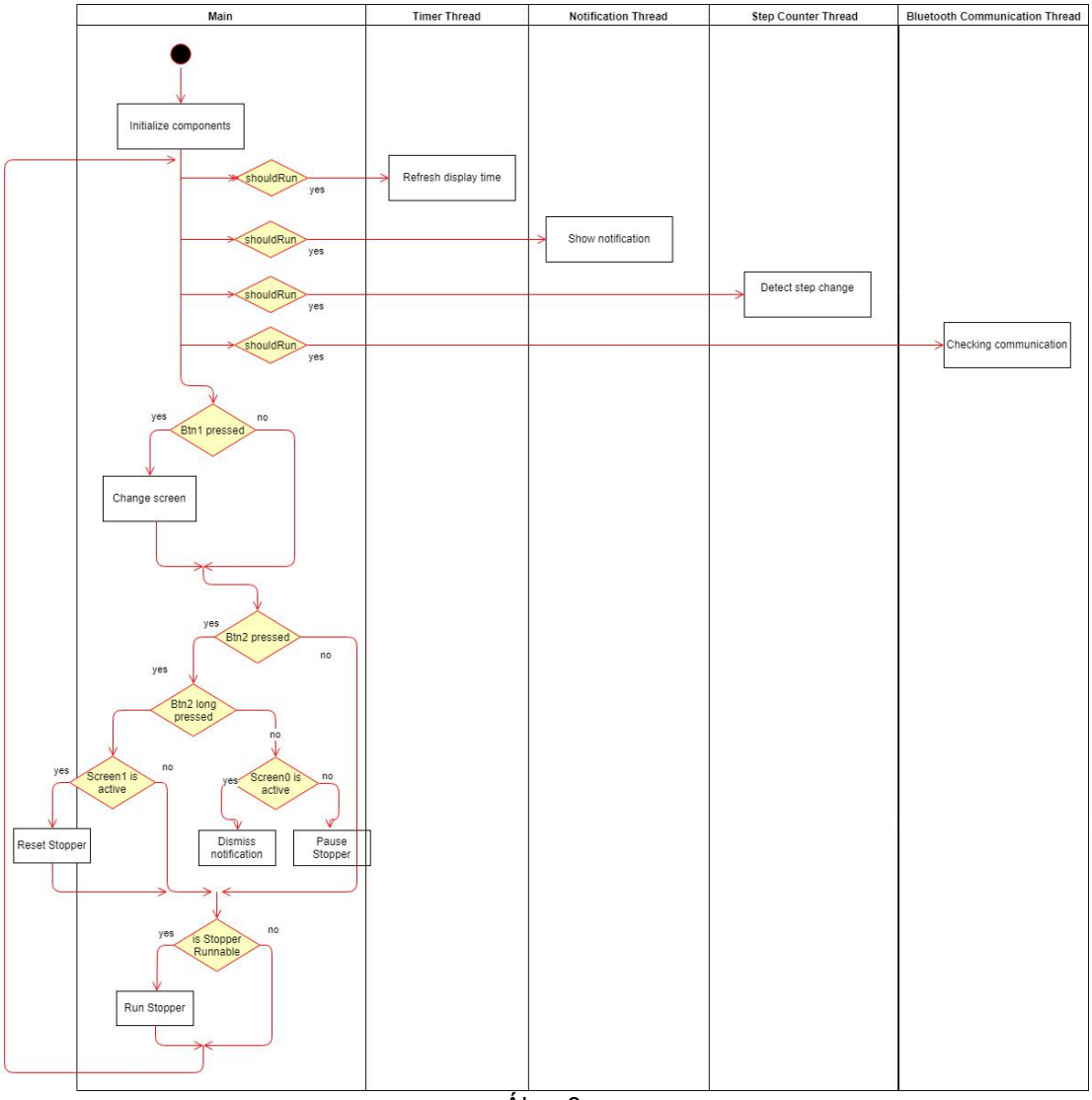
3. UML diagramok (Arduino)

3.1. Struktúrált diagrammok - osztálydiagram



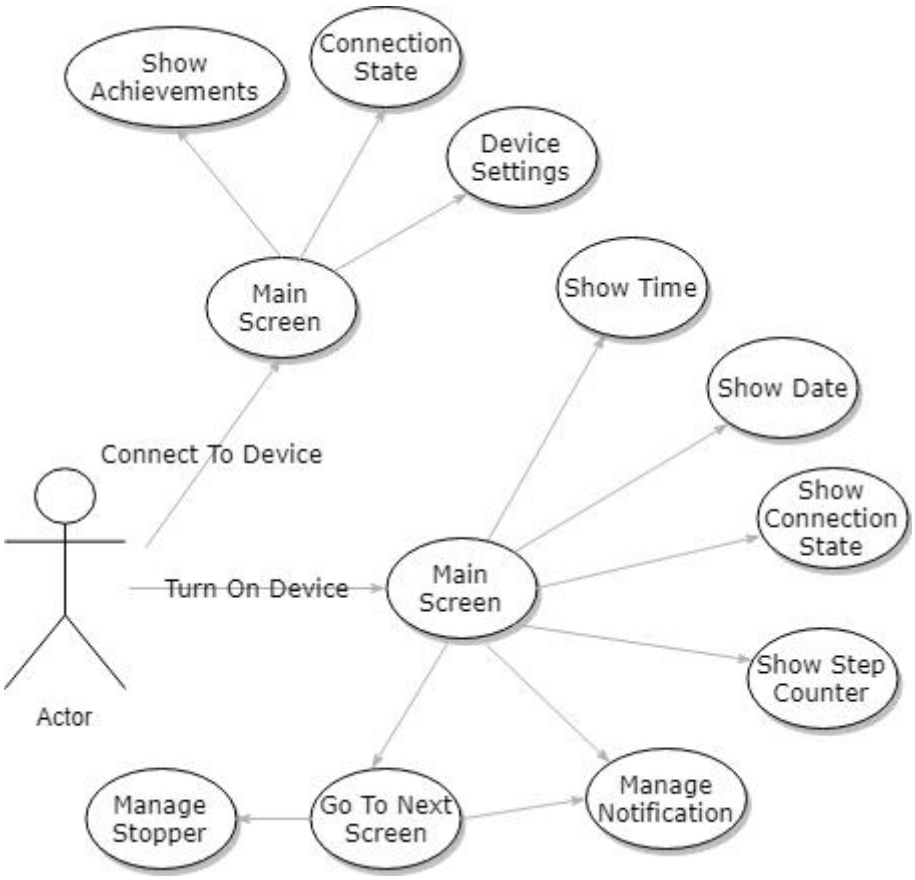
Ábra 2

3.2. Activity Diagram



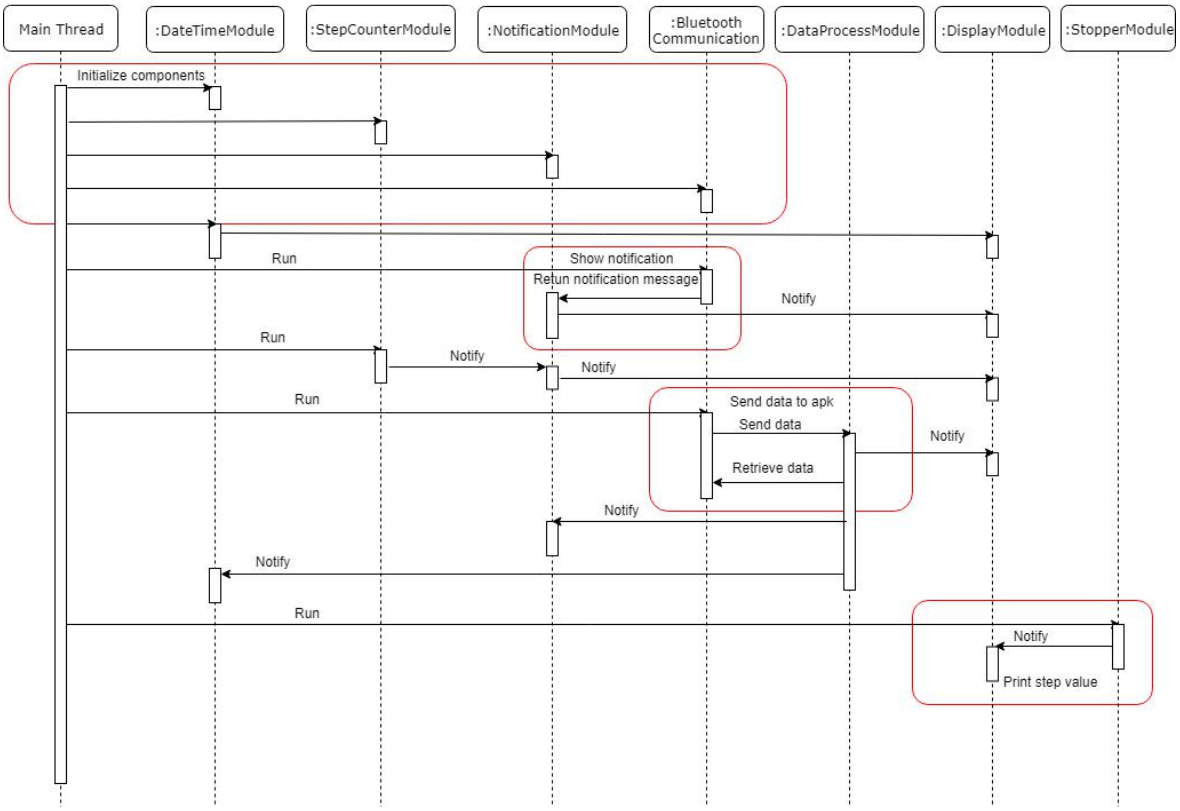
Ábra 3

3.3. Viselkedés (Use-case) diagram



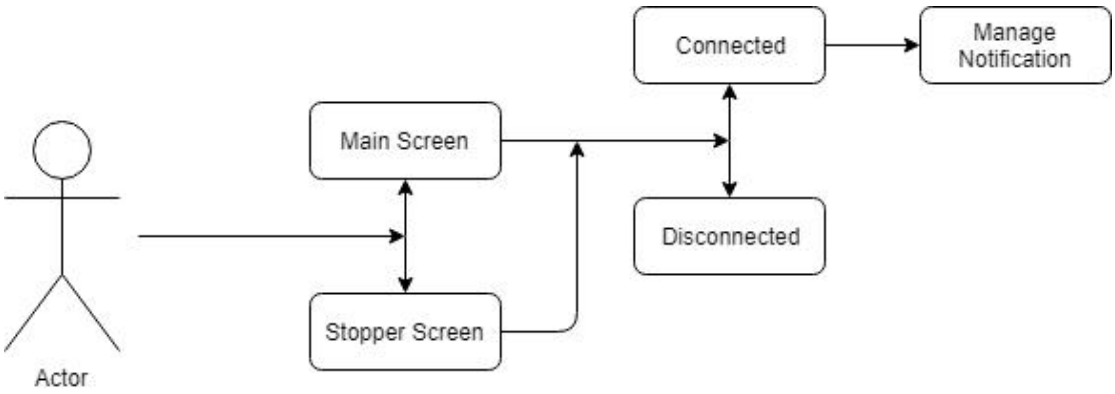
Ábra 4

3.4. Szekvencia (Sequence) diagram



Ábra 5

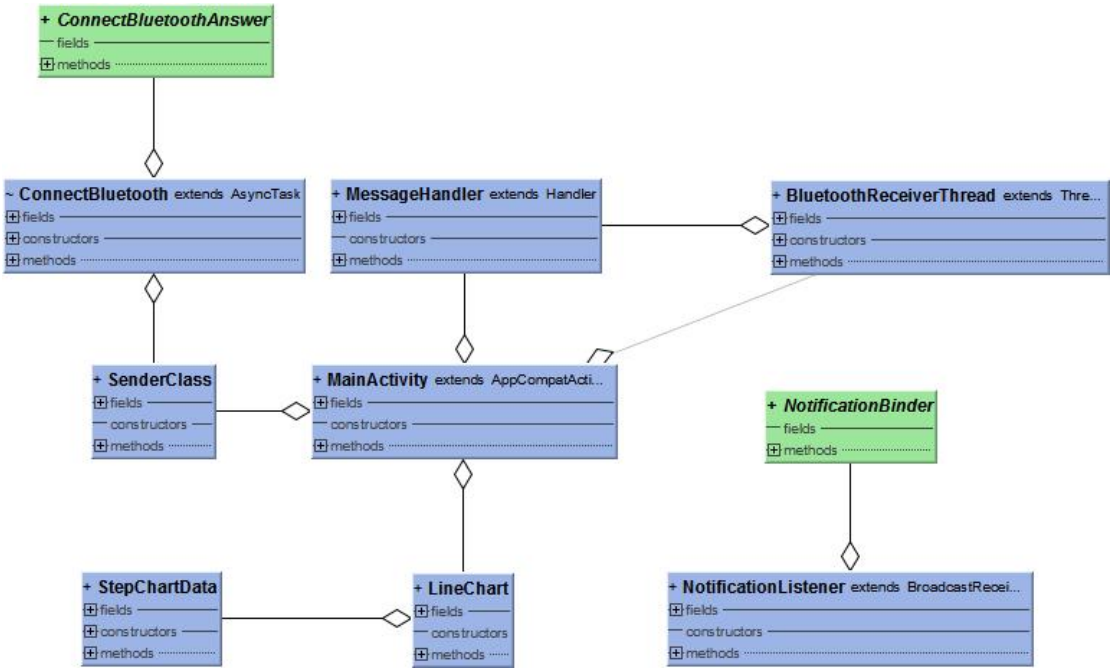
3.5. Állapot (State) diagram



Ábra 6

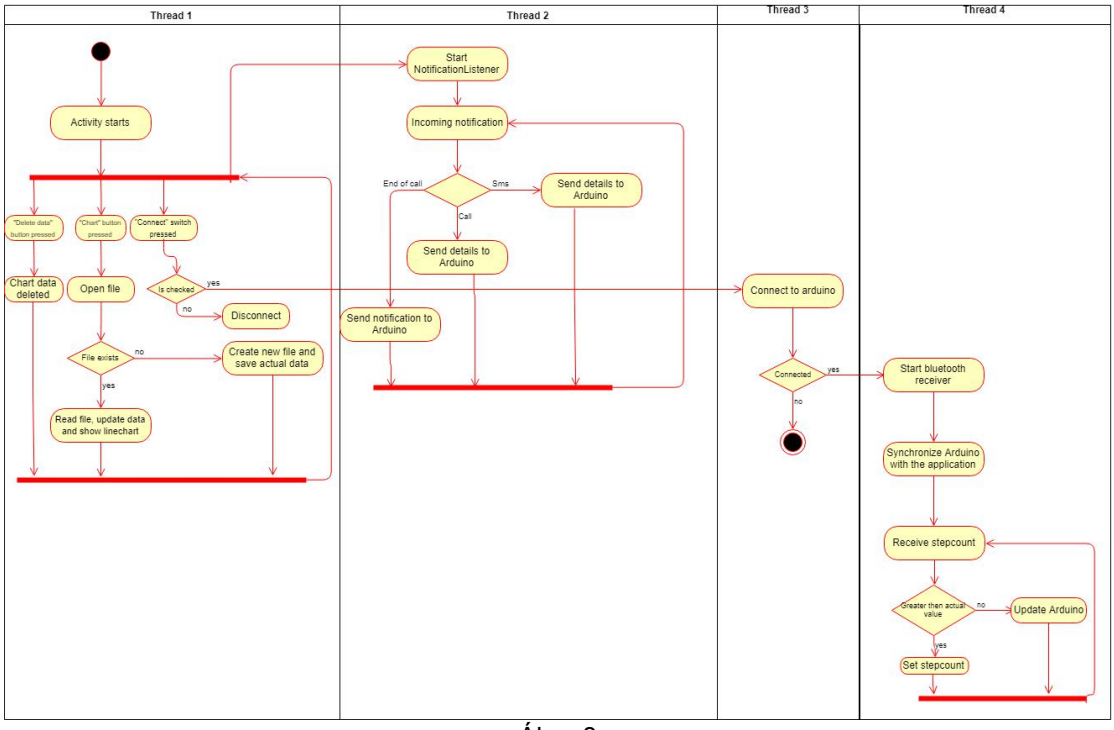
4. UML diagrammok (Android)

4.1. Osztálydiagram



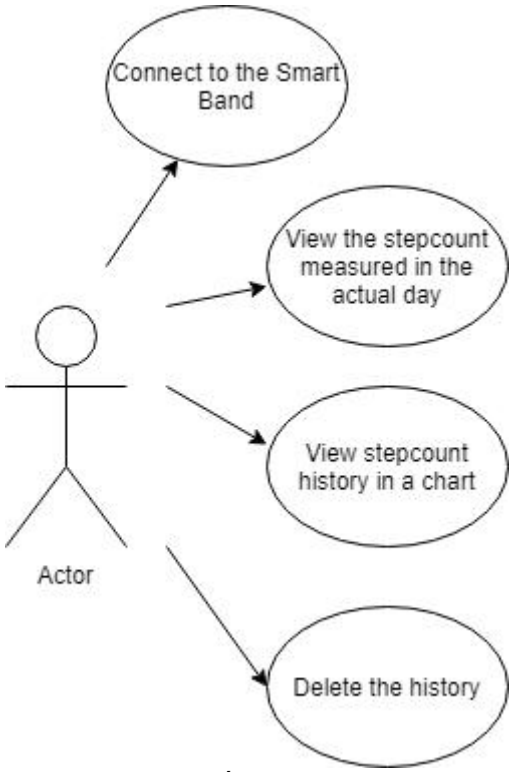
Ábra 7

4.2. Activity diagram



Ábra 8

4.3. Use-Case diagram



Ábra 9

5. Implementáció

5.1. Bevezető

Az eszközön futó program C++ nyelv alatt íródott, külön osztályokat hoztunk létre a különböző modulok kezelésére, mint például a DisplayModule, ami a képernyő kezelését valósítja meg, vagy a StepCounterModule ami a felhaszánló lépései számolja, illetve kezeli az adott adat megjelenítését a DisplayModule segítségéve. Számos Design Patternt használtunk a forráskód szerkesztése során, a könnyed átláthatóság, illetve későbbi bővíthetőség érdekében. Ezek közül megemlíteném az Adaptert, amelyet az adataink feldolgozása és továbbítása során használunk fel és az Observer Design Pattern-t, amely segítségével a kijelzőnknek értesítjük arról, hogy

bizonyos adat megváltozott, amit frissíteni kellen a képernyőn, ezzel alakítva ki a felhasználó és eszköz közti interakciót.

Az applikációt, mely segítségével eszközünk és telefonunk közti kommunikációt tudunk megvalósítani Androidban fejlesztettük.

5.2. Implementálás

Projektünk első lépése az eszköz megvalósításához szűkséges eszközök kiválasztása és ezek beszerzése volt. Nagyobb méret mellett döntöttünk, az energiagazdálkodás kihasználás végett, emelett az Arduino Mega 2560 által nyújtott belső memória is teljes mértékben elégségesnek tűnt, hogy programunkat kivitelezhessük és helyesen működtethessük. Később, mikor már a projekt tesztelés fázisába értünk realizáltuk, hogy forráskódunk nagyszerűen működik Arduino Nano alatt is, így sokkal kisebb helyre lehet összesűríteni a kívánt eszközöket.

Minden modul tesztelésére külön időt szántunk, így készítve elő azokat a projekt teljeskörű beillesztésére. A már működő modulok összerakása és az osztályok helyes megtervezése jelentette a következő nagy problémát, viszont a megfelelő diagramok elkészítésével ezek mind megoldásra kerültek.

Ezzel párhuzamosan folytattuk az alkalmazás fejlesztését is okostelefonra, hisz azt szerettük volna, ha a felhasználó valós időben tudna statisztikákat látni az aktivitásáról, illetve különböző értesítéseket megjeleníteni az eszközön, így ha nem vagyunk a telefonunk közelében akkor is láthatjuk, hogy éppen mi történik a telefonunk képernyőén, esetlegesen ki akar kapcsolatba lépni velünk.

Az applikáció főképernyőjén látható az elmúlt időszak statisztikai adatai, a kitűzött célok jelzéséül szolgáló állapotjelző sáv, az elégetett kalória mennyisége, a napi lépésszám, emelett egy kapcsoló, ahol váltani tudunk, hogy applikációnk az eszköz összekapcsolásával vagy anélkül funkcionáljon. Lehetőségünk van továbbá törölni az eddig összegyűjtött adatokat.

A kapcsolat kialakítására az eszköz és applikáció között Bluetooth kommunikációt használtunk. Az adatok küldését, fogadását és ezek sikeres megérkezésének vizsgálatát egy külső libray segítségével valósítottuk meg. Egy saját protokollt vezettünk be, annak vizsgálatára, hogy az érkezett adat milyen csoportba tartozik, kinek szól, milyen formában kell továbbküldődjön/ megérkezzen, illetve hogyan kell feldolgozni. Az általunk használt protokollkészlet az alábbi szerkezetet

mutatja: a legelső egy bájt tartalmazza az üzenet típusát, vagyis hogy kinek szól és lényegében milyen üzenetet tartalmaz. A applikáció által küldött és eszköz által fogadott kódszavak a következő képpen néznek ki:

s|400 - jelzi, hogy a StepCounter értéket 400-ra kell beállítanom a programban

t|1930 - az idő beállítása, ami 19 óra 30 percet jelent

d|12031 - a dátum beállítása, 12.-ik hónap 3.-a, kedd

m|TestUser - új üzenet érkezett a Test User nevű felhasználótól

p|TestUser - bejövő hívás a Test User nevű felhasználótól

i|0 - a kapcsolat állapotát jelzi, ebben az esetben a 0 a kapcsolat hiányát mutatja

n|end - az end kúlcsszó segítségével fejezzük be az értesítés mutatását a képernyőn

c|achieved - ha elértük a napi kitűzött célunkat, akkor ezt az üzenet küldi el az applikáció

Az eszköz által applikáció felé küldött adat:

s|551 - 551 lépést tettünk meg a mai nap folyamán

6. Következtetések

A projekt megvalósítása nem volt egy egyszerű feladat, rengeteg új kihívást jelentett számunkra az új technológák megtanulása és minél optimálisabb, célravezetőbb és leghatékonyabb használata. A fejlesztés két párhuzamos fázisát tudjuk elülöníteni, az eszköz fejlesztését és a rá illő forráskód megírását, illetve az applikáció fejlesztésének fázisát. Minden összevetve nem okozott gondot számunkra csapatmunka, hiszen a Git verziókövető rendszert használtuk projektjeink koordában tartására, illetve használtuk a Trello nevű alkalmazást, amely segített a feladatok leosztásában, nyilvántartásában és nem utolsó sorban a határidők betartására.

Következtetésképpen elmondható, hogy sikerült egy igencsak megbízható eszközt létrehozni, amely használhat akár a mindennapok során is kisebb tevékenységek felhasználásához, illetve az Android applikáció segítségével bármikor

nyomonkövethetjük az elmúlt tevékenységeink adatait, esetleg az aktuálisakat listázhatjuk.

7. Lehetséges bővítések

1. Többnyelvűség beállításának lehetősége
2. Felhasználó statisztikáinak javítása személyes adatok megadása által
3. Elérhető eszközök listázása, majd az adott listából való kiválasztás a csatlakozáshoz
4. Firebase Realtime adatbázissal való összekapcsolás, felhasználói fiók létrehozásával, a rugalmasabb adatelérés és feldolgozás érdekében
5. Az eszkök kis méretre csökkentése, a hordozhatóság érdekében
6. Lépésszámláló mért értékeinek pontosabb meghatározása
7. Hívás elfogadás/elutasítás nyomogómb segítségével
8. Rezgő üzemmód beépítése az eszközbe
9. Időmérő pontosítása küldő modul segítségével

8. Ábrajegyzék