Budapesti Műszaki és Gazdaságtudományi Egyetem

Villamosmérnöki és Informatikai Kar

Automatizálási és Alkalmazott Informatikai Tanszék

Bodolai Dorottya, Gazdi László

FELÜGYELŐALKALMAZÁS FEJLESZTÉSE MOBIL OKTATÓJÁTÉKOKHOZ

KONZULENS

Dr. Forstner Bertalan

BUDAPEST, 2015

TARTALOMJEGYZÉK

Összefoglaló

Napjaink egyre növekvő problémája, hogy az általános tanítási és tanulási módszerek a mai rohanó, digitalizálódó világban egyre kevésbé testhezállóak, és nem képesek lekötni a gyermekek figyelmét. Ez hatványozottan igaz azokra a tanulókra, akik valamilyen tanulási nehézségben (diszlexia, diszgráfia, diszkalkulia, stb.) vagy figyelemzavarban szenvednek. Számos kutatás bizonyítja, hogy a megváltozott percepciós mintával rendelkező gyerekek sokkal jobban képesek figyelni és kommunikálni valamilyen elektronikai eszközzel. Szerencsére az utóbbi időben az informatika és különösen a mobilipar fejlődésével már szinte mindenhol megtalálhatóak ezek a készülékek. Elterjedésükkel párhuzamosan egyre többféle oktatószoftver jelenik meg a különböző platformokra, ezeken belül is a legnépszerűbbek, adottságaikból adódóan az érintőképernyős mobiltelefonok és a tabletek. Kutatások irányulnak arra, hogy ezek felhasználásával játékos formában, újra élménnyé tegyék a tanulást. További előnye az elektronikus formában tálalt feladatoknak, hogy könnyen újrafelhasználhatóak, amivel értékes pedagógusi munkaidőt spórolnak meg, és lehetővé teszik az önálló gyakorlást is. Azonban az önálló gyakorlásnak nagy veszélye, hogy a tanuló nem tudja jól meghatározni a számára megfelelő feladatot, ismétlésszámot és nehézséget.

A tanszéken futó AdaptEd projekt ezekre a problémákra nyújt megoldást. A projekt keretein belül fejlesztett keretrendszer képes mérni a tanulók vagy inkább játékosok mentális és érzelmi állapotát, majd ez alapján befolyásolni a játékmenetet. A rendszer figyel arra, hogy folyamatosan a megfelelő állapotban tartsa a játékost, és így a lehető leghatékonyabbá tegye a tanulási folyamatot. Azonban az ilyen rendszereknek is szüksége van valamilyen felügyeletre egy képzett tanár részéről.

Munkánk során az AdaptEd keretrendszerhez fejlesztünk egy felügyelő alkalmazást, amin keresztül a tanár figyelemmel tudja követni a diákjait. Az Androidos alkalmazásban a tanár láthatja az osztályteremben játszó összes tanulójának játékmenetét és fiziológiai adatait. A tanár a megfigyelés mellett interakcióba is léphet mind a keretrendszerrel, mind a játékokkal. A keretrendszer osztályozó algoritmusainak tanítóhalmazát is a felügyelő határozza meg a személyes tapasztalatai alapján, majd futás közben is befolyásolhatja annak működését. Szükség esetén közvetlenül változtathat a játékok nehézségi szintjein is.

A dolgozat a felügyelő alkalmazás fent leírt feladatkörein keresztül mutatja be a komplex rendszer tervezési és implementálási feladatait, amik lehetővé teszik az újszerű felhasználási módot. A dolgozat kitér a bemutatott felügyelő alkalmazás valós környezetben futó tesztjeire is.

Abstract

Nowadays in our fast and digitalized world, the general educational and learning methods are getting more and more outdated, they are less suitable for preoccupying the students. This is especially true to those pupils who suffer from some kind of learning disability (dyslexia, dysgraphia, dyscalculia, etc) or attention deficit disorder. Several studies show that those children, who have changed patterns of perception are performing much better at concentrating and communicating when electronic devices are used. Fortunately, the recent evolution in information technology and in particular the mobile industry, these devices are widely available. While these tools are being spread in the world, educational serious games are getting more popular, especially the ones developed for smartphones and tablets. There are many researches focusing on how to use these games to make studying more enjoyable again. These digitalized tasks can be easily re-used, which saves valuable pedagogic time and allows independent practice as well. Although individual exercising has a great benefit, it has disadvantages too, as it can be difficult to find the proper task, the number of repetition and the difficulty level.

The AdaptEd project, which is developed in our department, offers a solution to these problems. The Framework is able to measure the mental and emotional state of the students (players), and uses these data to manipulate their gameplay. The system takes care of keeping the player in proper condition and thus make the learning process as efficient as possible. However, such systems also need supervision of qualified teachers.

Our work focuses on a Supervisor software for the AdaptEd Framework, which can be used by teachers to monitor the activities of their students. This Android application helps the teacher observe gameplays and physiological data of all the students in her class. Besides the monitoring, the teacher can interact with the Framework and the games as well. The classification algorithm of the Framework gets its training set from the markers of the teacher, who uses her personal experience to set the proper values, and she is able to influence the calculation even at runtime. If needed, she is also able to directly modify the difficulty of the games.

Besides the already mentioned functions of the Supervisor application, the paper presents the complex system planning and implementation tasks, which makes the innovative uses possible. The paper covers real environment tests of the presented Supervisor application.

1. Bevezetés

A huszonegyedik században a digitális technika ez életünk szerves részévé vált. Ha kimegyünk az utcára vagy közösségi közlekedési eszközt veszünk igénybe, feltűnik, hogy talán nincs is olyan ember, aki ne hallgatna zenét, ne telefonálna, internetezne, vagy éppen csak játszana az okostelefonjával. Míg ezelőtt néhány évvel, évtizeddel, a járműveken a legtöbb utas a reggeli híreket olvasta valamelyik nyomtatott sajtótermékben, ma már a híreket szinte mindenki „böngészi". A nyomtatott sajtó túl lassú ehhez a felgyorsult világhoz, ahol mindenki folyamatosan online van, és a hírek azonnal elérik őket. A digitális eszközök már mindenkinek a táskájában, zsebében ott vannak, használatuk és kezelésük mindennapi rutinná vált.

Így van ez a gyermekeknél is, akik már szinte az anyatejjel szívják magukba a digitális világot. Gyakran előbb képesek mobiltelefont vagy táblagépet kezelni, mint járni vagy beszélni. Óvodás és kisiskolás korukra a digitális eszközöket készség szintjén használják, kezelésük egyértelmű számukra. Így hát nem is csoda, hogy amikor a rohanó, izgalmas, digitális világból bekerülnek az iskolapadba, és múlt századi tananyaggal és taneszközökkel találkoznak, szinte azonnal lelkesedésüket vesztik. Ezek a száraz, lassú, monoton módszerek nem képesek lekötni a gyermekek figyelmét, így nem, vagy csak lassabban érik el a kívánt hatást.

Különösen igaz ez akkor, amikor olyan gyermekek ülnek be az iskolapadba, akiknek valamilyen tanulási nehézségük van. Őket néhány évtizede még a legtöbb helyen egyszerűen lustának vagy butának titulálták, ha nem tudtak megfelelően olvasni vagy számolni, holott csak másfajta odafigyelésre és tanítási módszerekre lett volna szükségük. Szerencsére manapság már idejében felismerik, ha egy gyermek olvasási (diszlexia), írási (diszgráfia), számolási (diszkalkulia) vagy bármilyen egyéb tanulási, esetleg az úgynevezett figyelemhiányos hiperaktivitás-zavarban (Attention Deficit Hyperactivity Disorder, *ADHD*) szenved. Az ilyen gyermekek nem butábbak társaiknál, csak máshogyan működik az agyuk. Ha megkapják a megfelelő odafigyelést és bánásmódot egy képzett fejlesztő pedagógustól, akkor akár az egészséges tanulóknál jobb eredmény elérésére is képesek lehetnek.

Az ilyen fajta tanulási zavarok és a figyelemhiány a társadalom 5-10%-át érinti[1]. Ennyi gyermek képzéséhez és fejlesztéséhez nagyon sok gyógypedagógusra lenne szükség, azonban az ő számuk sajnos véges. Különösen igaz ez a kisebb városokban, falvakban. Míg egy nagyvárosban több olyan iskola és pedagógus is található, aki megfelelő szinten képes foglalkozni a tanulási zavaros gyerekekkel, a kisebb településeken élőknek nincsen ekkora szerencséjük, így gyakran nem kapják meg a megfelelő képzést.

A fent vázolt problémákra megoldást nyújthatnak az oktatószoftverek. A régi tankönyveket, feladatgyűjteményeket felválthatják a mobil eszközökön futó applikációk. Ilyen környezetben még a legegyszerűbb, megszokott feladatok is új tartalommal tölthetőek fel, és újra élménnyé tehetik a tanulást. A gyerekek az ilyen oktatószoftvereket sokkal szívesebben használják, különösen, ha az valamilyen játékos formában tanít. A tanulók így játék közben sajátíthatnak el valamilyen új tudást, vagy gyakorolhatják be a frissen tanult leckét.

A tanulási zavaros gyerekek is sokkal szívesebben tanulnak digitális eszközökkel[2], hiszen nem kell izgulniuk, egy gépnek mindig van türelme. A türelem és a gyakorlás pedig ezeknél a tanulóknál még sokkal fontosabb, mint egészséges társaiknál. Az oktatószoftverek képesek szinte a végtelenségig feladatokat adni egy témakörből, mindig valami újdonsággal, folyamatos jutalmazással. Ez lehet egy időnként megjelenő figura, kép, hanghatás, valamilyen gyűjthető tárgy vagy egy egyszerű pontszám. A játékos környezet, a változatosság, a folyamatos jutalmazás tényleg élménnyé, játékká teheti a tanulást. Így egy tanuló akár magától, önállóan is képes gyakorolni, feladatokat csinálni. Ezzel értékes pedagógusi idő szabadulhat fel, ami több gyereknél eredményezhet nagyobb odafigyelést.

Azonban az önálló feladatvégzésnek itt is megvannak a maga veszélyei. Amennyiben a tanuló önállóan választ feladattípust, szintet, ismétlésszámot, úgy előfordulhat, hogy nem a megfelelő mellett dönt. Könnyen megtörténhet, hogy a folyamatos sikerélmény elérése érdekében túl könnyű feladatokat választ, vagy a nehezebbeket inkább átugorja. Ezzel a tanulás nem hatékony, sőt, néhány esetben még kontra produktív is lehet. Tehát a speciális tanári irányítás és felügyelet soha sem hagyható ki a tanulási és tanítási folyamatból.

1.1. A cél

A dolgozat célja olyan hatékony felügyeleti módot találni, amellyel tanulási diszfunkcióban szenvedő gyermekek fejlődését segíthetjük elő digitális oktatójátékok használatával. A felügyelettel szemben támasztott legfőbb követelmény, hogy a tanuló számára ne legyen zavaró, ugyanakkor a tanár mégis képes legyen megfigyelni minden apró részletet, valamint szükség esetén képes legyen befolyásolni a játékmenetet.

Az ilyen tanulási zavarral küzdő tanulóknál különösen fontos, hogy nyugodtan, nyomás nélkül, a külön a számukra kialakított speciális feladatokkal, példákkal tanuljanak. Ezek nem csak az adott tantárgy elsajátításában lehetnek segítségükre, hanem olyan alap készségek fejlesztésében is, melyek az ő esetükben problémát okozhatnak.

A következőkben először bemutatjuk azt a nagyobb projektet, és fontosabb részeit, amibe szorosan illeszkedik jelen dolgozatunk tárgya, a felügyelőalkalmazás. A 2. fejezetben kitérünk az egyes komponensek közötti kommunikációra, különös tekintettel a felügyelő lehetőségeire. A 3. fejezetben bemutatjuk az elkészült alkalmazást, majd a 4. fejezetben kitérünk az éles környezetben végzett tesztekre is. Végül az 5. fejezetben összegezzük a tapasztalatainkat, és felvázoljuk a továbblépés irányait.

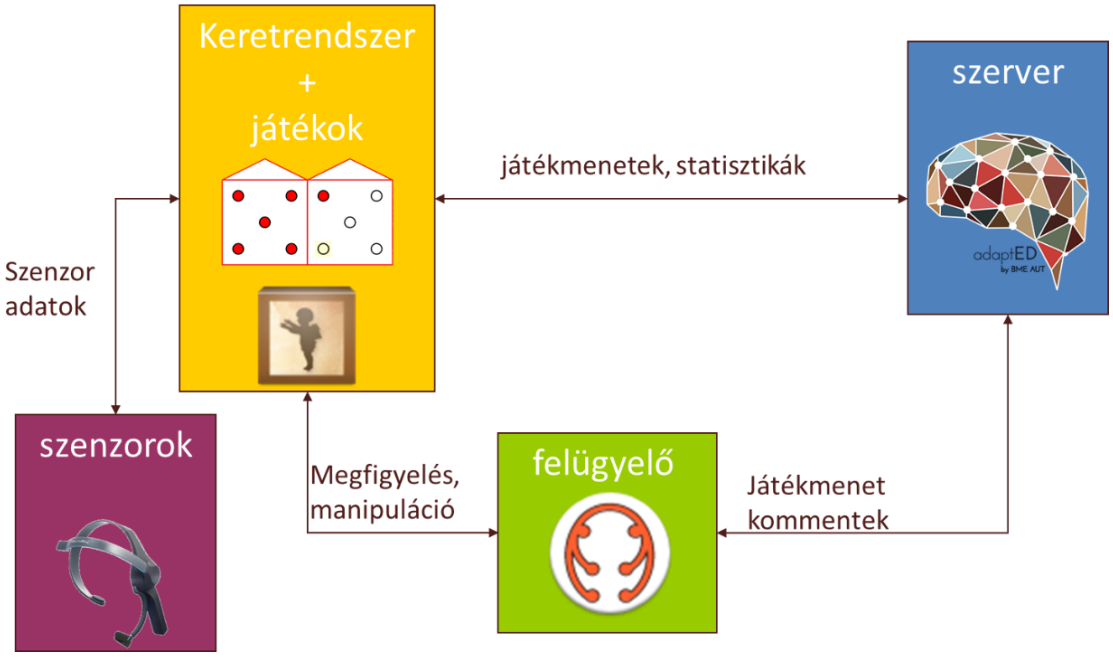
1.2. Az AdaptEd projekt

Az Automatizálási és Alkalmazott Informatikai Tanszéken fejlesztett AdaptEd projekt célja egy olyan oktatási rendszer kialakítása, mely nem csak speciális feladatokat, játékokat tartalmaz, hanem egy komplett keretrendszert. Ez a keretrendszer a játékok nehézségét képes adaptív módon állítani és az ehhez szükséges fiziológiai adatokat feldolgozni. A gyermekek haladásának megfigyelésében egy szerver alkalmazás és egy felügyelő applikáció van a pedagógusok segítségére.

1.2.1. Architektúra

A projekt a következő komponensekből épül fel:

- Oktató, fejlesztő játékok
- Keretrendszer
- Szenzorok
- Webes szerver
- Felügyelő alkalmazás

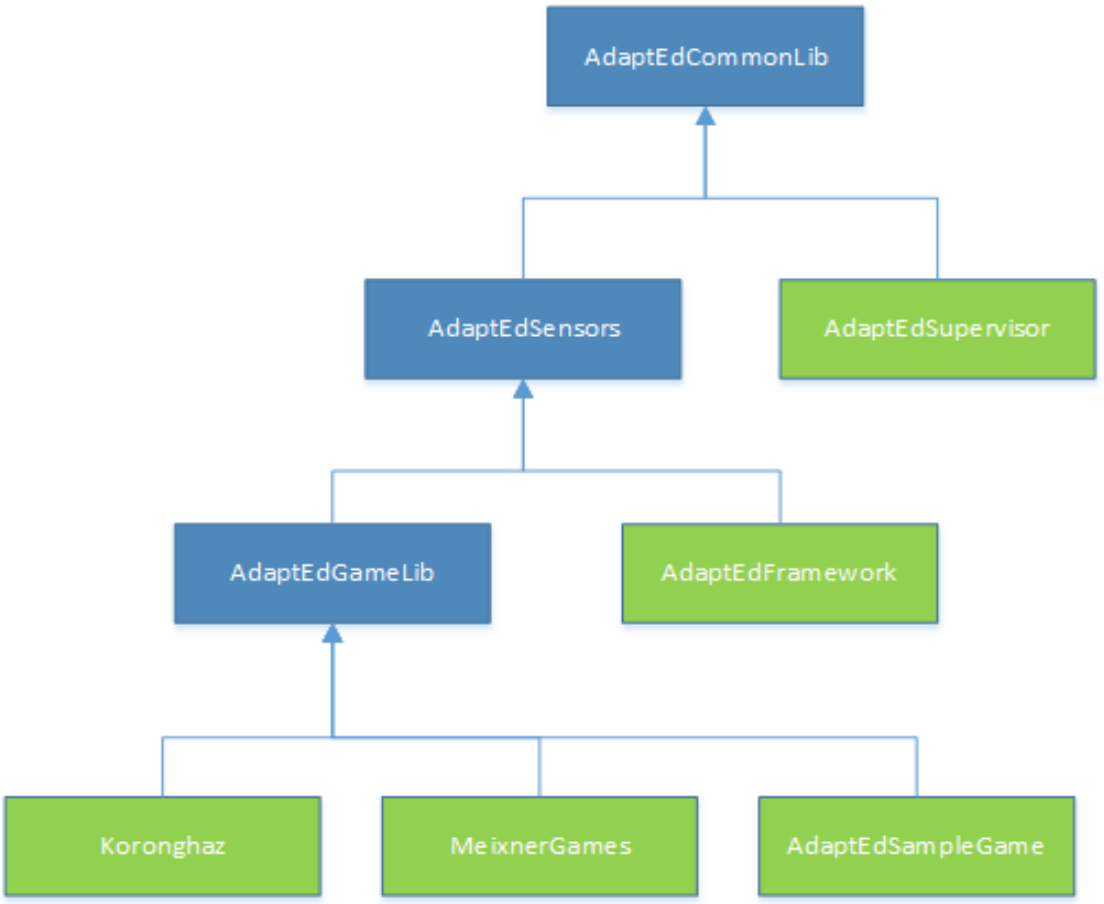


1. ábra: Az AdaptEd rendszer architektúrája

A keretrendszer és a játék azonos eszközön, a felhasználó táblagépén fut, ehhez kapcsolódnak a szenzorok. A szerver PC-n fut egy .NET-es szoftvercsomag, a felügyelő alkalmazás pedig a pedagógus táblagépén fut. Többféle fiziológiai szenzort használunk, illetve tervezünk használni. Ezek egyike egy olyan EEG berendezés, mely könnyű, olcsó és hordozható. Ez a játékos agyhullámainak megfigyelésére szolgál, erről szolgáltat adatokat a keretrendszernek. Másik fontos szenzorunk egy ugyancsak könnyen felhelyezhető, hordozható szívritmus mérő készülék. Ezek használata és működése a későbbi fejezetekben lesz bemutatva.

A 2. ábrán látható komponensek közül a kék színnel jelöltek az osztálykönyvtárak. Ezekre épülnek a zöld színnel jelölt konkrét különálló applikációk.

- AdaptEdCommonLib: Alkalmazások közti kommunikáció osztályai, konstansai, illetve olyan egyéb elemek, melyeket egyszerre többféle alkalmazás is használ (pl.: az itt található egységes bejelentkező képernyőt használják a játékok és a felügyelő is).
- AdaptEdSensors: A biofeedback szenzorokkal kommunikáló, azok mért adatait feldolgozó java illetve natív C++ komponenseket tartalmazó könyvtár.
- AdaptEdGameLib: Minden játék alapját képező könyvtár.



2. ábra: A projekt Androidos könyvtárai és alkalmazásai

1.2.1.1. Oktatójátékok

A jelenleg fejlesztés alatt álló speciális játékok a Meixner Alapítvánnyal[3] folyamatos szakmai együttműködésben készülnek. Az alkalmazások egy része elsősorban a tanulási nehézségekkel küzdő gyermekek számára készült az intézmény speciális módszerei alapján. Ezek a játékok a folyamatos gyakorlást helyezik előtérbe, ami a tanulási zavarral rendelkező gyermekek számára a legfontosabb annak érdekében, hogy ők is ugyanolyan természetesen tudják használni bizonyos képességeiket, mint egészséges társaik. A játékok másik része a mindennapi oktatásban használatos feladatokat helyezi át digitális platformra a tanulási élmény növelése és a tanár segítése céljából. Itt olyan egyszerű feladatokra kell gondolni, mint például a mondat-kiegészítés, a csoportosítás vagy egy egyszerű puzzle kirakása. Ha ezeket a tanár egyesével, minden gyermeknek saját kezűleg készíti el, rengeteg idejébe kerül. Ha a diákok csoportmunkában oldják meg, akkor pedig nem biztos, hogy mindenkinek ugyanannyira hasznára lesz a feladat. A digitális feladatok előnye, hogy elég egyszer elkészíteni őket, ami után valamennyi tanuló gyakorolhat vele.

A játékok úgy készülnek, hogy mindegyik megvalósítsa a keretrendszer által adott interfészt, melynek segítségével a keretrendszer képes a játékmenetekről információt gyűjteni, és azok felhasználásával befolyásolni tudja azt.

1.2.1.2. Keretrendszer

A játékok támogatására szolgál egy általános keretrendszer, ami egy háttérben futó Android Service. Ez segíti a többi komponens működését.

A keretrendszer folyamatosan gyűjti az információt a játékmenetről, és a felhasználó fiziológiai adatairól majd ezek alapján igyekszik a nehézséget úgy hangolni, hogy a gyermekek számára megfelelő szinten tudjon maradni, vagyis a játék közben a gyerekek mentális állapota a lehető legjobb legyen[4]. Ennek az állapotnak a definiálása nem könnyű dolog, mi erre a Csíkszentmihályi Mihály által megfogalmazott[5] flow élményt használjuk.

A keretrendszer az egyes mérések adatait továbbítja a szerver felé, ahol azok tárolásra kerülnek. Ezen adatok tárolása fontos, egyrészt egy későbbi esetleges analízis miatt, másrészt ezekből létrehozható egy speciális profil a felhasználóról. Ez a profil elengedhetetlen annak érdekében, hogy egy ismételt használatnál a keretrendszer megfelelően tudja osztályozni a tanuló adatait, és kikövetkeztesse a megfelelő mentális állapotot.

A keretrendszer ugyanis tartalmaz egy osztályozó algoritmust, amely a mért fiziológiai adatok alapján, a jelenlegi és az előző játékmenet alapján valamint a pedagógus visszajelzése alapján határozza meg a játékos mentális állapotát. A rendszer ennek fényében ajánlást tesz a játékok felé a szint változtatására, vagy mini-játékok indítására. A mini-játékok célja, hogy az unatkozó vagy esetleg elkalandozó figyelmű tanulókat visszaterelje a megfelelő mentális állapotba.

A keretrendszer a játék, a szenzorok és a szerver mellett kommunikál a felügyelő alkalmazással is. Továbbítja felé a mérési adatokat, játékmeneteket, képernyőképeket és fogadja a beérkező utasításokat, valamint az osztályozó algoritmus számára a mentális állapot tanítóhalmaz elemeit.

1.2.1.3. Biofeedback szenzorok

A keretrendszer egyik fő feladata, hogy kapcsolatot tartson a csatlakoztatott fiziológiai mérőműszerekkel, majd ezek adatainak analízise után meghatározza a felhasználó

mentális állapotát. A rendszer jelenleg két külső szenzort használ rendszeresen, de tervben van továbbiak bevonása, hogy a mért adatok kombinálásával még pontosabb eredményeket érhessünk el.

1.2.1.3.1. EEG

Néhány évtizeddel ezelőtt egy EEG készülék mindennapos használata elképzelhetetlen lett volna. Az ilyen készülékeket leginkább klinikákon vagy kutatóközpontokban lehetett megtalálni, és nagyon drágák voltak. Az utóbbi idők technológiai fejlődése azonban lehetővé tette, hogy elérhető áron hozzáférhetővé váljanak. Ma már nem óriási és kényelmetlen gépekre kell gondolni, amikor az agyhullámok vizsgálata szóba kerül. Az ár és méret csökkenésével természetesen a pontosság is csökkent, azonban az egy-kétszáz dolláros eszközök is képesek olyan adatokat szolgáltatni, amelyek elégségesek a jelen céljainkra.

Jelenleg a keretrendszerhez az Emotiv Epoc szenzora [6], valamint a NeuroSky Mindwave Mobile egycsatornás EEGje [7] van csatlakoztatva.

3. ábra: NeuroSky Mindwave Mobile

Ez a szenzor, a nyers adatok mellett, képes olyan származtatott értékek szolgáltatására is, mint a felhasználó figyelmének vagy nyugodtságának mértéke. Ezek az adatok nagy segítségére lehetnek a keretrendszerben futó algoritmusnak, és a megfigyelő pedagógusnak is a validálás során.

1.2.1.3.2. EKG

A másik, keretrendszerhez kapcsolt szenzor egy Zephyr HxM BT szívritmus mérő [8]. Egy orvos számára egy EKG diagram rengeteg információt hordoz a páciens egészségi állapotáról. Esetünkben a mérés lényege gyakorlatilag a szívritmus-variancia, amiből nagyon jól lehet következtetni a mentális állapotra.

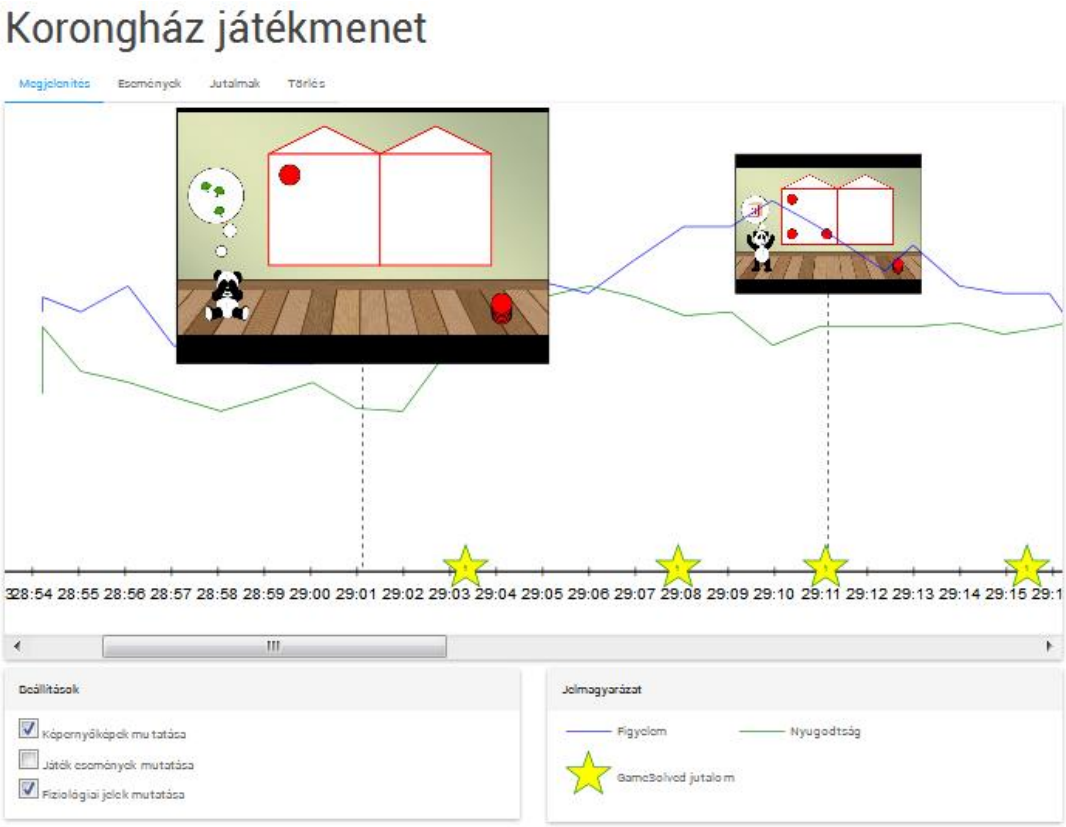
4. ábra: Zephyr HxM BT

Mindkét mérőeszközön látszik, hogy kisméretű, és egyszerűen használható. Ez azért jó, mert az őket használó gyermekeket nem zavarja vagy frusztrálja a viselésük. Ugyanis fontos, hogy a minél jobb hatásfok érdekében a tanulók nyugodt általános állapotban, mindenféle zavaró külső tényezőtől mentesen játszanak.

1.2.1.4. Szerver

A szerver végzi a játékos profilok nyilvántartását, a hozzájuk tartozó játékmenetekkel. A játékmenetekről tárolja az egyes eseményeket illetve a felhasználó játék közbeni fiziológiai jeleit. A későbbi esetleges elemzéseket segítendő webes felületen visszanézhetőek az adatok és játékmenetek, valamint megtekinthetőek a játékosprofilok. Az oktatójátékokkal játszó regisztrált felhasználók intézményekbe, azon belül csoportba vannak osztva, így segítve a könnyebb eligazodást.

Annak kiszűrésére, hogy egy-egy játékmenet során valóban a bejelentkezett felhasználó használta-e az alkalmazást és el tudjuk kerülni, hogy a játékos profilba fals adatok kerüljenek, egy, a szerveren futó csalásdetektáló is implementálásra kerül.



5. ábra Egy, a szerveren tárolt, játékmenet ábrázolása

1.2.1.5. Felügyelő

A felügyelő a pedagógusok segédalkalmazása. A helyi hálózaton kapcsolódik a keretrendszert futtató eszközökhöz, így a tanárok valós időben tudják megfigyelni, vagy akár manipulálni az egyes diákok játékmenetét és a hozzájuk kapcsolódó fiziológiai méréseket.

1.3. Egy tantermi szituáció

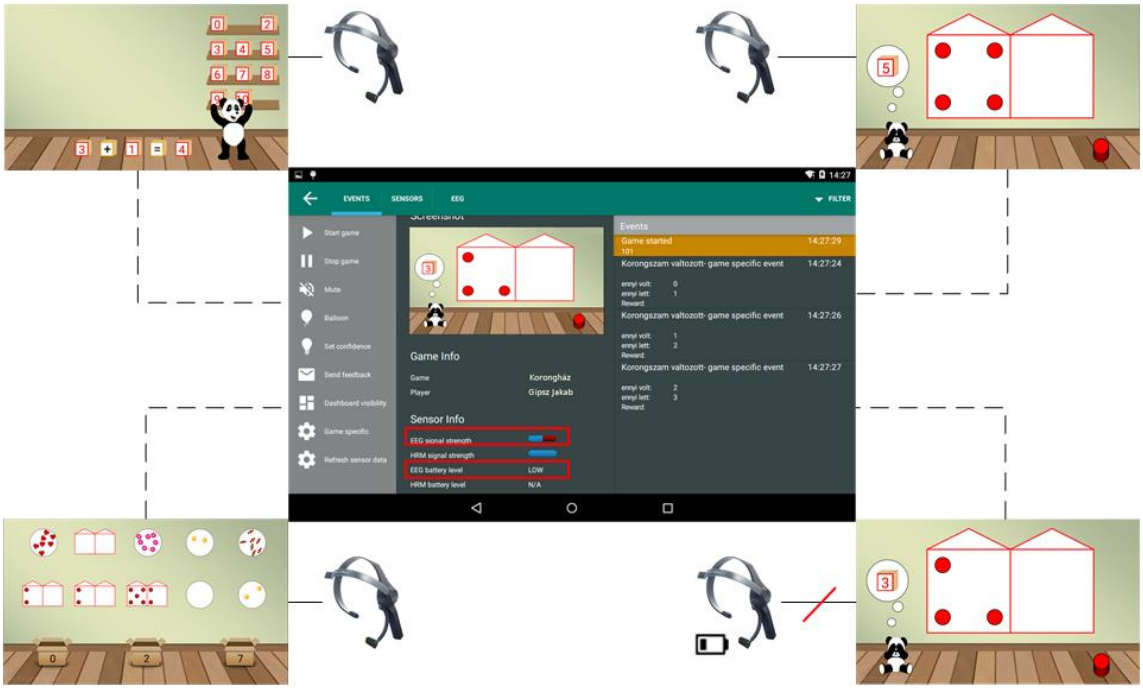
Tantermi környezetben fontos, hogy az egyes komponensek között megbízható kommunikációt tudjunk kiépíteni. A kommunikációval a 2. fejezet foglalkozik részletesebben, de előbb lássunk egy példát arra, amikor egy egyszerű játékot használ egy osztály.

A matematika tanteremben több gyermek ül, előttük táblagépek. Kezdetét veszi a játékos feladatmegoldás. Minden gyermek elindítja a Korongház játékot, ami egy diszkalkuliás tanulók számfogalmának kialakulását elősegítő oktatószoftver, amely szintén a projekt keretein belül készült el. A tanító pedig az asztalok között lavírozva indítja el mindenkinek a megfelelő számkör feladatait, mivel a gyermekek nem tudják meghatározni, hogy éppen mit is kéne gyakorolniuk. A tanító figyelmét folyamatosan megosztja

a gyermekek között és igyekszik mindenki haladását figyelni, úgy hogy közben ne zavarjon meg senkit. Eközben lemaradhat fontos információkról, például a próbálkozások számáról vagy a hibák jellegéről. Ezekről később csak akkor fog bármit megtudni, ha a gyermekek részletesen emlékeznek az egyes feladatokra adott megoldásaikra. Előfordulhat játék közben, hogy valaki nem a feladattal foglalkozik, elterelődik a figyelme, de erről sem értesül azonnal a pedagógus, pedig tudna segíteni.

Most lássuk ugyanezt a példát úgy, hogy nem csak a játék áll az osztály rendelkezésére, hanem a hozzá tartozó keretrendszer, biofeedback szenzorok, a szerver és a felügyelő alkalmazás.

A matematika tanteremben több gyermek ül, előttük táblagépek, amikhez bluetooth kapcsolaton keresztül csatlakoztatva van 1-1 EEG és szívritmus mérő. Kezdetét veszi a játékos feladatmegoldás. Minden gyermek bejelentkezik a Korongház játékba, a tanító pedig a felügyelő alkalmazáson keresztül elindítja a feladatokat. Játék közben az egész osztályról teljes képet kap, látja mindenkinek a haladását. A keretrendszer figyeli a gyermekek eredményeit és a feladatokat ennek megfelelően nehezíti vagy könnyíti. A pedagógus mindeközben távolról, a diákok megzavarása nélkül nézheti a játékmeneteket. Az egyik gyermek a fiziológiai jelei alapján frusztrált. A keretrendszer azonnal értesíti a felügyelő alkalmazást, ami pedig jelez a pedagógusnak. Ő erre egy merőben más típusú feladatot választ a tanulónak. Egy másik gyermek fején elcsúszott az EEG és lassan le is fog merülni, így nem megfelelő adatokat mér az eszköz, de a felügyelő ezt is rögtön látja, mivel értesítette az alkalmazása. A tanár a nap végén vissza szeretné nézni az osztály egyes tanulóinak játékmenetét és a hozzájuk tartozó fiziológiai méréseket, hogy tudja kinek mit kéne másnap gyakorolni, ezért belép a szerver webes felületére, és elkezdi böngészni az adatokat.

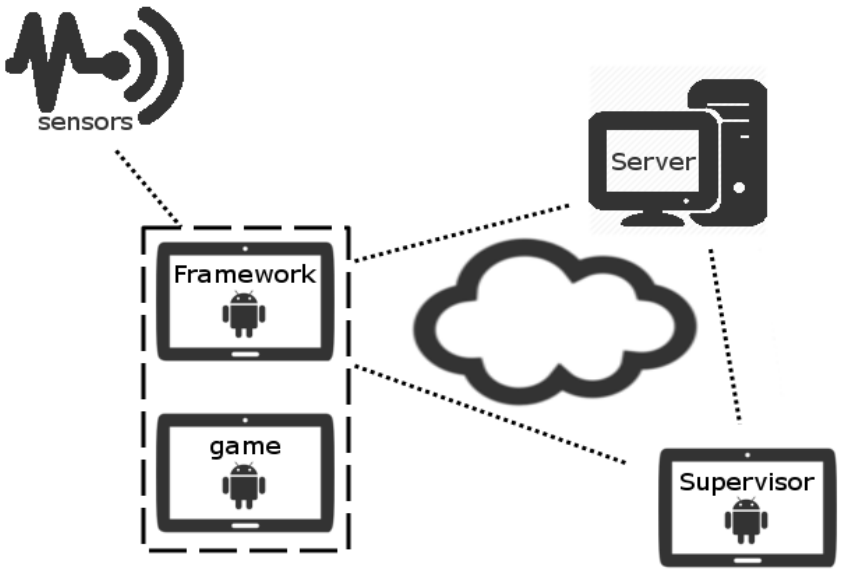


6. ábra Az osztálytermi szituáció. A felügyelő képernyőjén látható a hibás EEG adatokról szóló értesítés

2. Kommunikáció

Az 1.3. fejezet példájából is látszik, hogy a komponensek közti kommunikáció fontos szerepet játszik az egész projektben. Ennek is legfőbb eleme a keretrendszer, amely a hozzá csatolt szenzorok, a felügyelő, a szerver és a játékok között végez üzenet-továbbítást, illetve a saját méréseit, számításait is képes a többi elem felé eljuttatni. Mivel a három folyamat eltérő logikával rendelkezik, más-más csatornákat használnak, időzítési szempontból is különböznek, illetve a továbbítandó adattípusok is másak, fontos, hogy a keretrendszeren belül elkülönítsük az ezen funkciókat megvalósító modulokat.

Az üzenetek alapegysége minden irányban egy-egy esemény, melyek a különböző megoldásokban különböző előre definiált módon alakítva kerülnek elküldésre.



7. ábra A kommunikáció sematikus rajza

2.1. Keretrendszer

A keretrendszer, a játékoktól fogadott adatokat, a saját komponensei között esemény alapú kommunikációval továbbítja, amely egy központi eseménykezelőn keresztül folyik. Az egyes modulok csak azokat az eseményeket kapják meg, melyekre szükségük van. Például a figyelmi szint belső feldolgozására szolgáló *AttentionChangedEvent*-et nem szükséges a szerverrel kommunikáló *ServerProxy* modulba továbbítani, csak a már

feldogozott változatát, az *AttentionChangedGameEvent*-et. Ennek megvalósítása a *Visitor* modell alkalmazásával történt.

2.2. Szenzorok

A legtöbb biofeedback eszköz bluetooth kapcsolaton keresztül csatlakozik a keretrendszert futtató okostelefonhoz vagy táblagéphez. A mért fiziológiai adatokat is így továbbítják a mobil eszköz felé, amelyen a futó keretrendszer a szenzorok API-ját használva elkapja és feldolgozza azokat.

A szenzorok saját, előre meghatározott üzenettípusokkal rendelkeznek. Mindegyikre igaz, hogy jól definiált fejléc és hozzá meghatározott típusú payload szerkezetűek. Ezek feldolgozását a keretrendszer megfelelő szenzor moduljai végzik, melyek az így kapott adatokat eseményekké alakítják és továbbítják a többi modul felé.

2.3. Szerver

A keretrendszer a szerver felé HTTP POST kérésekben küldi az egyes eseményeket. Ezek lehetnek a játékmenethez kapcsolódó események, szenzor adatok, vagy akár felügyelő által küldött utasítás.

Az események JSON formában kerülnek felküldésre, hogy szerver oldalon könnyen, gyorsan feldolgozhatóak legyenek.

Az események szerver felé küldésére csak bizonyos időközönként kerül sor így a szerveren nem követhetőek valós időben az adatok, viszont utólagos feldolgozásra kifejezetten alkalmasak az adatsorok. A kapott játékmenetek tárolásra kerülnek, így bármikor visszanézhetőek, elemezhetőek.

2.4. Játékok

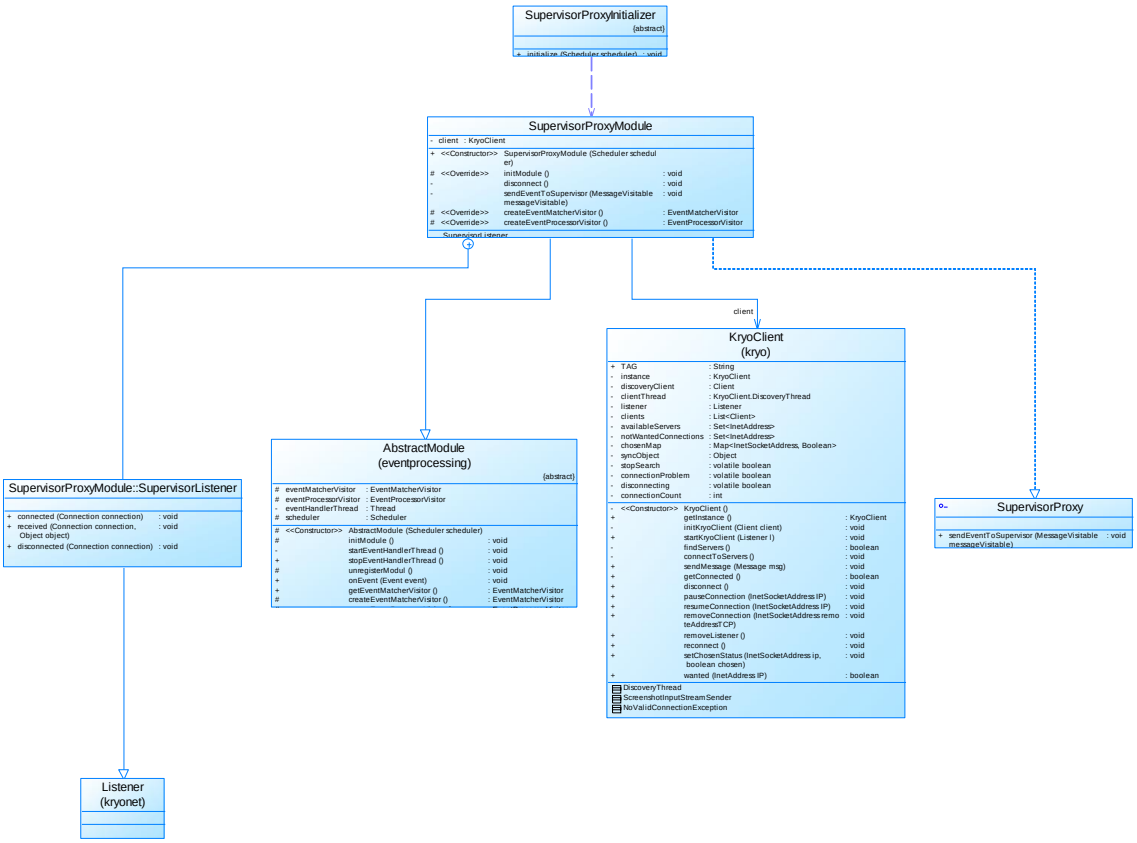
A játékok egy közös, a keretrendszer által definiált interfész segítségével (*FrameworkProxy*), a beépített Android-os *Messenger*-ek illetve Broadcast üzenetek használatával képesek a keretrendszerrel kommunikálni. A játékok a saját eseményeiket a Framework felé az interfész `void sendGameEvent(GameEvent gameEvent)` metódusával, egy *Messenger* használatával továbbítják. A keretrendszer a felügyelőtől kapott utasításokat illetve a saját eseményeit ugyancsak egy *Messenger* használatával küldi. Mindkét oldalon *Handler*-ek segítségével dolgozzák fel az üzeneteket.

2.5. Felügyelő

Felügyeleti szempontból ez a kétirányú kommunikáció a legfontosabb, hiszen itt valós időben követheti a pedagógus a gyermekek munkáját, és akár interakcióba is léphet velük.

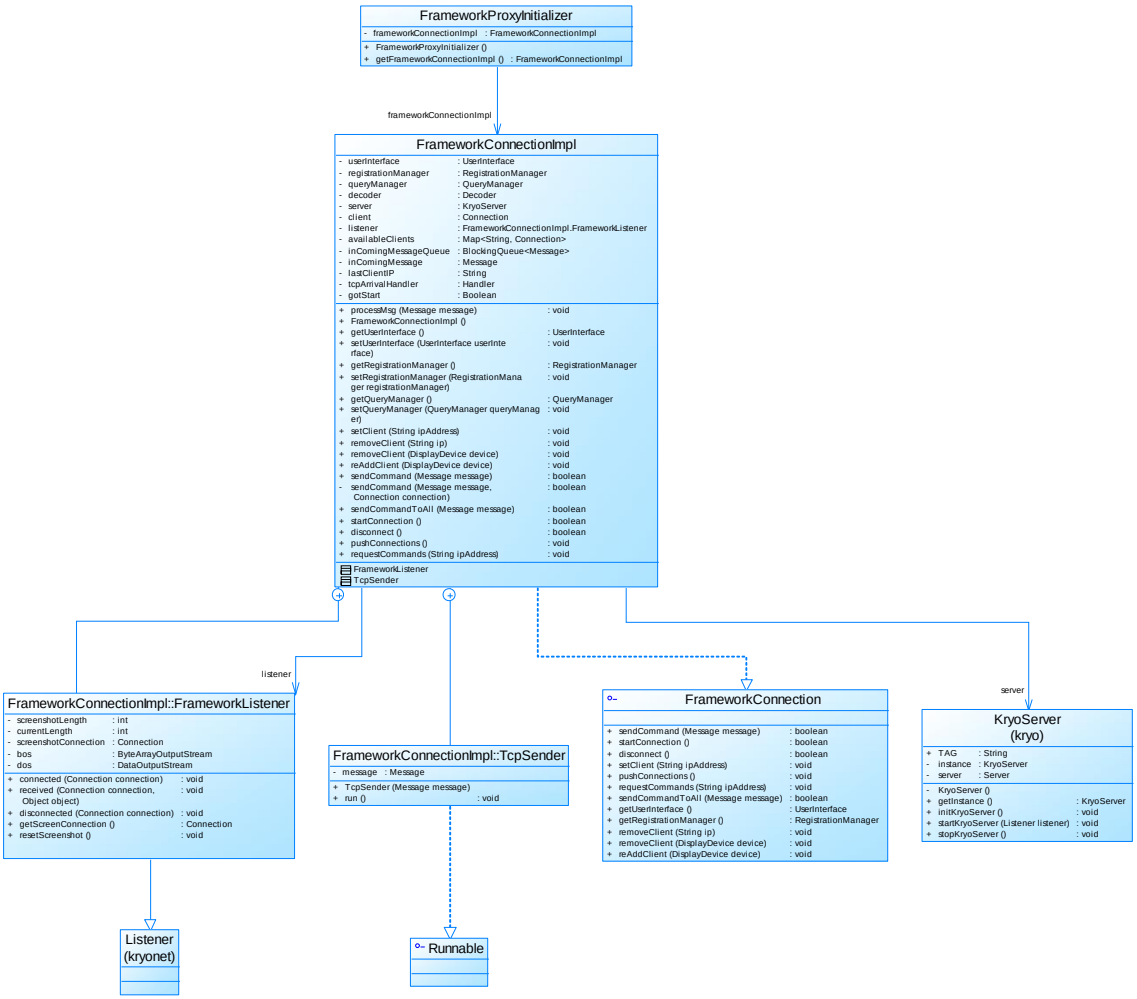
A kommunikációt a KryoNet [9] könyvtár segítségével valósítottuk meg, amely hatékony kapcsolatot és üzenetváltást biztosít. Legfőbb előnye, hogy egész objektumokat küldhetünk át a segítségével, így nincs szükség különösebb konverziókra vagy szerializálásra.

A kommunikáció olyan szerver-kliens architektúrának feleltethető meg, melyben a szerver szerepét a felügyelő tölti be, a kliensek pedig az egyes játékosok eszközein futó keretrendszerek. Az üzenetváltásért Framework oldalon a *SupervisorProxy*, Supervisor oldalon pedig a *FramewokProxy* felel. Az alkalmazások indításakor mindkét oldalon elindulnak a kommunikációs modulok. A kapcsolathoz használt végpontokat kezelő osztályok megvalósítása a közös *AdaptEdCommonLib*-ben található.



8. ábra Framework SupervisorProxy moduljának osztálydiagramja

A 8. ábra mutatja be a Framework kommunikációs modulját. A *SupervisorProxyModule* abból az *AbstractModule* osztályból származik, melyből az alkalmazás többi, a belső üzenetcserében résztvevő, modulja is. Láthatjuk továbbá, hogy referenciával rendelkezik a kommunikációért felelős *KryoClient Singleton* osztályra is.



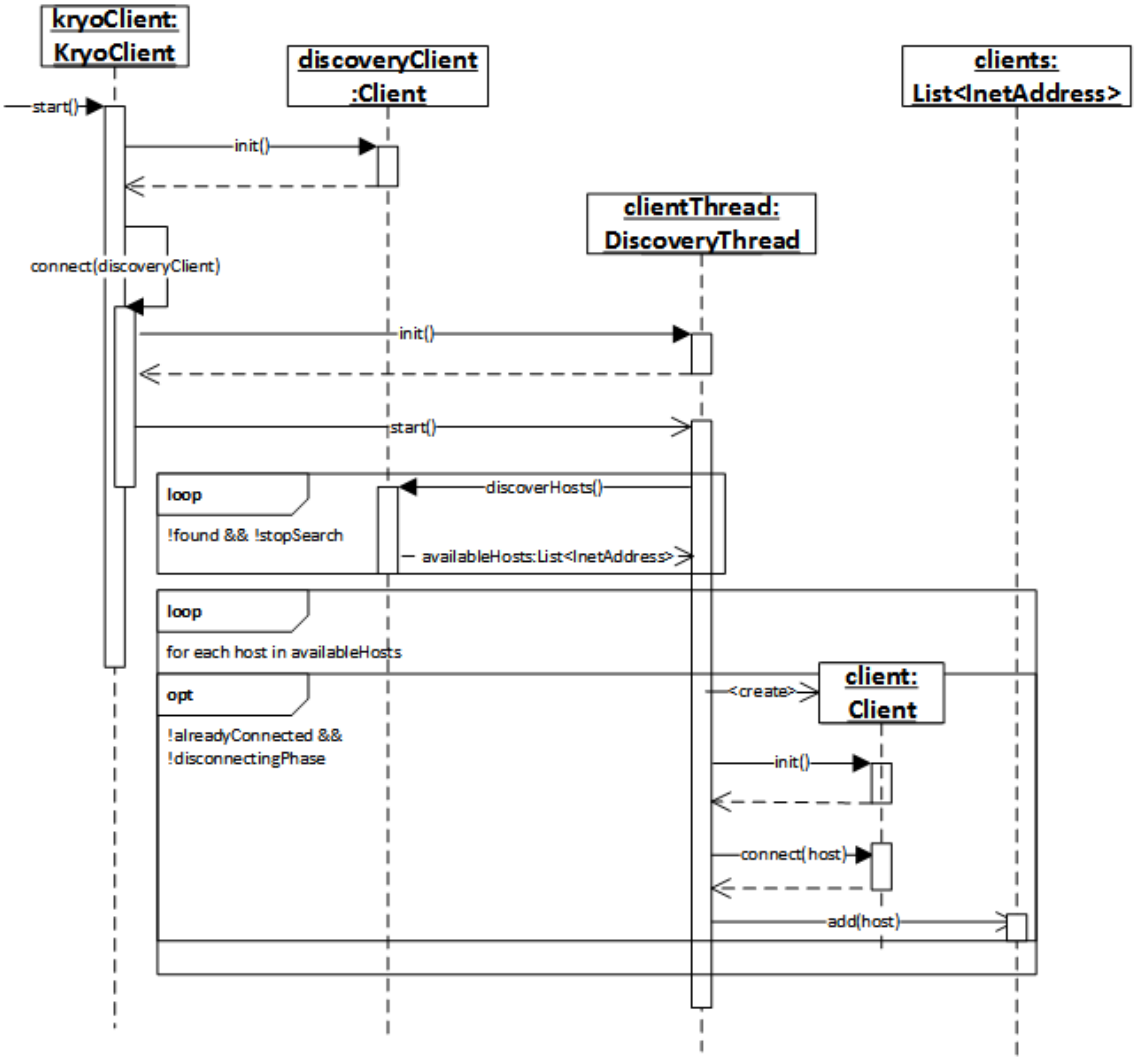
9. ábra Supervisor FrameworkProxy moduljának osztálydiagramja

A 9. ábra a Supervisor keretrendszerrel való kapcsolatért felelős *FrameworkProxy* modulját mutatja. A *FrameworkConnection* interfészű *FrameworkConnectionImpl* osztály tartalmaz egy *Sigleton* megvalósítású, *KryoServer* referenciát, ami a kommunikációt végzi.

2.5.1. Kapcsolódás

A Supervisor egy-egy megadott TCP illetve UDP porton várakozik a kliensekre. A kliens kapcsolódási folyamatát a 10. ábra mutatja be. A *kryonet* kliens `List<InetAddress> discoverHosts (int udpPort, int timeoutMillis)` metódusa

a megadott UDP porton, adott várakozási timeout-tal keres a hálózaton hosztokat. Mivel nincs arra garancia, hogy a Framework indításakor már van elérhető felügyelő, így ez a keresés adott esetben akár a játék végéig is tarthat. Amint a keresés talált felügyelő(ke)t, átlép a csatlakozási fázisba. Itt a keretrendszer minden megtalált felügyelőhöz az IP címe alapján, a megadott TCP porton keresztül csatlakozik.



10. ábra A Framework felügyelő kereső logikájának szekvencia diagramja

2.5.2. Kapcsolatok kezelése, üzenetfogadás

A kapcsolatok kezelésére a *KryoNet* a *Listener* osztályok alkalmazását támogatja. Ennek megfelelően a Framework *SupervisorProxy* modulja és a Supervisor *FrameworkProxy* modulja is tartalmaz egy-egy ilyen *Listener* leszármazottat. A keret-rendszer több kapcsolatát reprezentáló *Client* példányokhoz tartozik egy közös *Listener*

példány, a Felügyelőben pedig az egyetlen *Server* példányhoz tartozik egy ilyen *Listener* példány. Ezek a következő interfésszel rendelkeznek:

- `public void connected(Connection connection)`

  A kapcsolat felépülésekor hívódik meg. A *SupervisorProxy* ilyenkor egy *SupervisorConnectionEvent* küldésével jelzi a keretrendszer többi modulja felé, hogy felügyelőhöz kapcsolódott. Ezt a játékkal való kommunikációért felelős *GameProxy* modul továbbítja a futó játék felé.

  A *FrameworkProxy*-ban lévő Listener új kliens csatlakozásakor eltárolja a kapcsolatot reprezentáló, paraméterként kapott *Connection* példányt, valamint értesíti a többi modult az új kapcsolatról.

- `public void disconnected(Connection connection)`

  Kapcsolat bontásakor hívódik meg. A *SupervisorProxy* ilyen esetben újra kapcsolódni próbál egy felügyelőre. Erre akkor van szükség, ha a felügyelő távolról bontja a kapcsolatot. Ez akkor történhet meg, ha a felügyelő kilép, vagy tiltólistára helyezi az eszközt. Mivel ez akkor is meghívódik, ha a kliens maga bontja a kapcsolatot, így ebben az esetben először a *Listener*-t el kell távolítani és csak utána lehet elvégezni a bontást.

  A *FrameworkProxy* ilyenkor eltávolítja az elérhető kliensek listájából a paraméterben kapott *Connection*-t és jelez a többi modul felé.

- `public void received(Connection connection, java.lang.Object object)`

  Üzenet fogadása a paraméterben megadott *Connection*-től. Az üzenetváltás mindkét irányban saját *Message* osztály segítségével történik. Ez byte formában tartalmazza az üzenet típusát, a küldött objektumot (*java.lang.Object*) illetve opcionálisan a küldő fél IP címét is. A felügyelő-keretrendszer kommunikáció szempontjából releváns különböző üzenettípusainak listája a függelékek között található (2. táblázat).

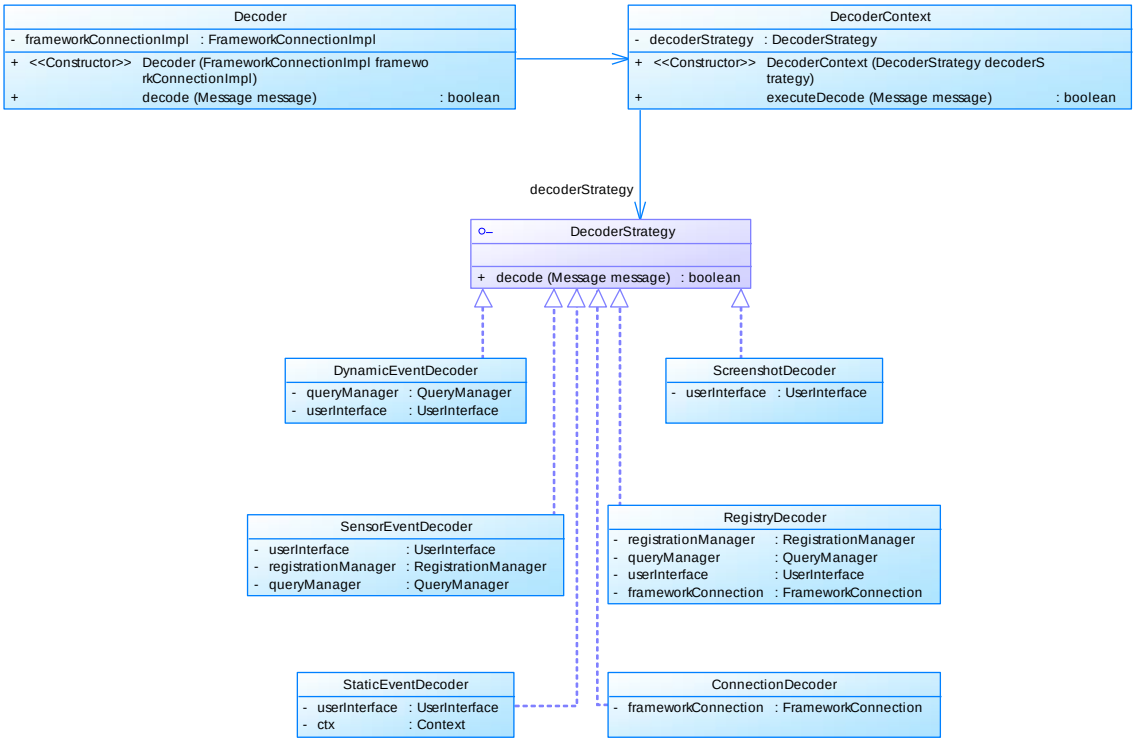
  A *SupervisorProxy* a beérkezett objektumot először üzenetté alakítja. Amennyiben tiltólistára vagy részletes megfigyelésre vonatkozik a fejléce, úgy ezt továbbítja a kapcsolatokat kezelő *kryoClient* felé. Egyébként a kapott üzenetet a *Message*, `void generateObject(Message message, MessageListener messageListener)` statikus metódusa megfelelő eseménnyé alakítja, és a kapott

MessageListener segítségével továbbítja a keretrendszer további részének (a felhasznált példány az eseményeket a többi modul felé továbbítja).

```
public interface MessageListener {

   public      void      onStaticCommandEventReceived(StaticCommandEvent
staticCommandEvent);

   public void onCommandEventReceived(CommandEvent commandEvent);

}
```

Az egyes események mind az *AdaptEdCommonLib Event* osztályából származnak.

A *FrameworkProxy* a beérkező objektumokat *Message*-ként dolgozza fel (hacsak nem screenshot darab érkezett), a feldolgozásra váró üzenetek listájába helyezi, majd a *Strategy* mintát követő *Decoder* segítségével feldolgozza. A képernyőképek, esetleges nagy méretük miatt, darabolva érkeznek. Ezt összevárja, majd így küldi feldolgozásra.



11. ábra A dekódolás során alkalmazott Strategy minta adaptációja

2.5.3. Üzenetküldés

A Supervisor az eltárolt kapcsolati példányok `int sendTCP(Object object)` metódusaival küldi az üzeneteket a keretrendszerek felé. A Framework ehhez hasonlóan a benne lévő kapcsolatonként fenntartott *Client* példányok *sendTCP* metódusát használja.

Ahhoz, hogy egy esemény elküldhető legyen, meg kell valósítania a *MessageVisitable* interfészt. Ennek a `Message generateMessage()` metódusa minden implementációban egy üzenetet generáltat. Ezt a *MessageGenerator* osztály statikus tagfüggvényeivel teszi, melyek minden egyes ilyen eseményosztályra a megfelelő fejlécű és tartalmú üzenetet generálják.

A képernyőképeket külön kell kezelni, mivel ezek nagy méretük miatt egyben nem küldhetőek át. Ennek a problémának a megoldására a *kryonet InputStreamSender* osztályát használjuk fel. Ez alapértelmezésben egy *InputStream*-et és egy alap csomagméretet kap paraméterként, majd ezek felhasználásával csomagméretnyi darabokban küldi az adatot. A *ScreenshotInputStreamSender* osztály az első üzenettel elküldi a byte tömbbé alakított képernyőkép esemény méretét a felügyelőnek, majd utána a megfelelő üzenet fejléccel a további darabokat is továbbítja az adott kapcsolaton keresztül.

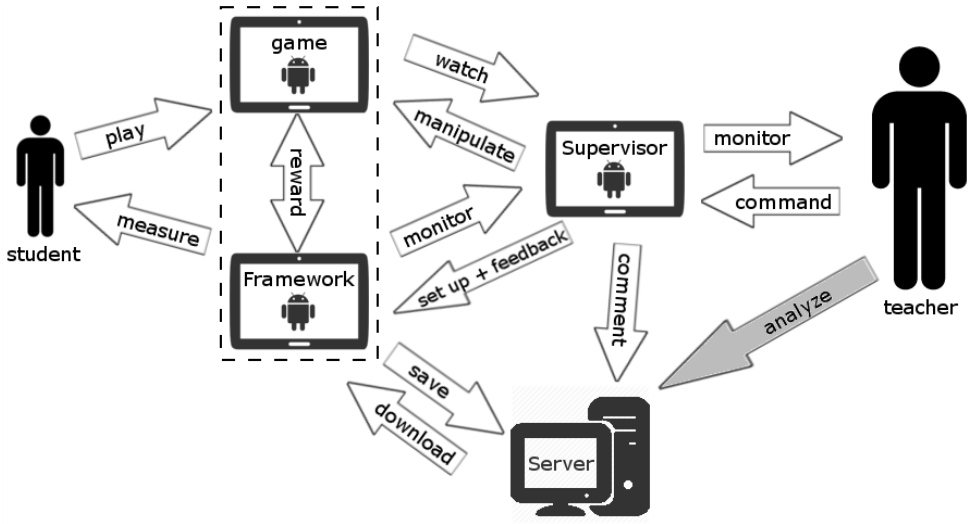
```
class ScreenshotInputStreamSender extends InputStreamSender {

    private int length;
    private Client client;

    public ScreenshotInputStreamSender(InputStream arg0, int arg1,
                                       int length, Client client) {
        super(arg0, arg1);
        this.length = length;
        this.client = client;
    }

    protected Object next(byte[] bytes) {
        return new Message(MessageCodes.Static.SCREENSHOT_PART,
                           bytes);
    }

    @Override
    protected void start() {
        super.start();
        Message m=new Message(MessageCodes.Static.SCREENSHOT_START,
                              length);
        synchronized (syncObject) {
            client.sendTCP(m);
        }
    }
}
```

2.5.4. Tiltólista

A felügyelőnek lehetősége van az olyan, hozzá kapcsolódó eszközök tiltólistára helyezésére, melyeket nem kíván megfigyelni. Ilyen eset előfordulhat például azért, mert több felügyelő is dolgozik egy adott csoporton, vagy a tanárok különböző csoportokat felügyelnek. Az ilyen eszközök számára a felügyelő egy *NOT_YOU* fejlécű üzenetet küld, melynek hatására a kliens a további üzenetküldésből kihagyja az üzenetet küldő Supervisort, de a kapcsolatot nem bontja vele. Ha a későbbiekben a felügyelő mégis szeretné követni a tiltólistára helyezett játék folyását, egy *YOU_AGAIN* fejlécű üzenettel jelezheti szándékát a keretrendszer felé.

3. A felügyelőalkalmazás

A felügyelőalkalmazásnak képesnek kell lennie a játékmenet megfigyelésére és manipulálása, valamint a fiziológiás adatok megfigyelésére.



12. ábra Tipikus felhasználói folyamat

3.1. Megfigyelés

A felügyelő alkalmazás egyik fő feladata a pedagógus segítése, hogy az könnyedén tudja a gyermekeket távolról, közvetlen fizikai kapcsolat nélkül figyelni.

A megfigyelt adatok a keretrendszer által küldött események, melyeket a felügyelő alkalmazás különböző módokon jelenít meg. Ezek az események többféle csoportba sorolhatóak. A minden játék során felmerülő azonos eseményeket statikus eseményeknek nevezzük. Ide tartoznak a játékmenettel kapcsolatos általános események, a játék és feladat indítása, megállítása és a képernyőképek. A játékos és játék leíróinak regisztrációja is statikus eseménynek tekinthető. A szenzorok is játék függetlenek, ezek eseményeit is ide soroljuk.

Dinamikus, játék specifikus eseménynek nevezzük azokat, melyek csak bizonyos játékok során merülnek fel. Ezek lehetnek akár csak egyszerű játékmenet információval szolgáló üzenetek, vagy különleges jutalmak, melyeket csak az adott játékban oszthat ki a keretrendszer. A dinamikus események leíróit az első használat során egy regisztrációs

üzenetben továbbítja a keretrendszer a felügyelő felé. Egy leíró (*TypeDescriptor*) tartalmazza a leírandó osztály nevét, az esemény kijelzésre szánt nevét és azon attribútumainak leíróit, melyek a felügyelő illetve a szerver számára elküldendőek. Egy attribútum leírója (*ParamType*) a paraméter nevéből, típusából, minimum és maximum értékéből, valamit a kijelzésre szánt nevéből áll. Ezek a leírók az eseményekről és a paraméterekről automatikusan keletkeznek, az egyes tulajdonságokat annotációk segítségével lehet megadni. A beérkező regisztrációs üzeneteket a Supervisor *RegistryDecoder*-e dolgozza fel majd a kapott dinamikus események (és a nagyon hasonló jutalmak) leíróit a *RegistryModule*-nak továbbítja, ahol név-leíró párosokként kerülnek tárolásra.

3.2. Interakció

A felügyelő alkalmazás másik fő feladata a megfigyelés mellett, hogy lehetővé tegye a tanár számára a távolról történő manipulációt. Ez jelenthet akár indítást, megállítást, újrakezdést, szintmódosítást vagy mini-játék indítást.

3.2.1. Parancsok

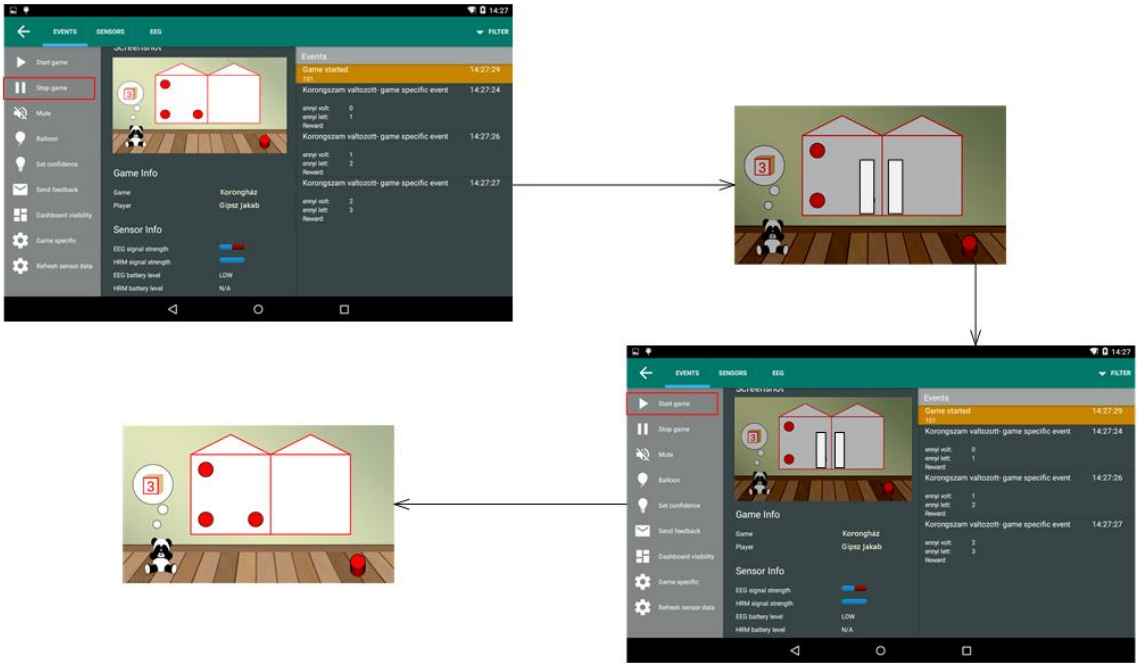
Az eseményekhez hasonlóan a parancsoknál is kétféle csoportot definiálhatunk. A minden játékban azonos statikus és a játékok saját, egyéni, dinamikus parancsait. Bár a statikus parancsok kezelésében az egyes játékok kis mértékben eltérhetnek, mivel minden játék másfajta elven működik, így megeshet, hogy például egy pillanat stop esetén különböző adatokat el kell menteni, időzítőt kell leállítani, ami a többi játékban nincs.

Az egyes parancsok a keretrendszerhez futnak be, majd az továbbítja azokat a játék felé, ami feldolgozza őket.

3.2.1.1. Statikus parancsok

Fontos, hogy a felügyelő a játékokat egyszerűen és gyorsan tudja manipulálni. Statikus parancsokkal a játékmenet szimpla irányítása lehetséges, valamint szolgáltathat adatot a keretrendszer számára, illetve megfogalmazhat egyszerű kéréseket a különböző bejövő információk minőségéről.

A játékmenetet manipuláló parancsok közé tartozik a játék indítása, a pillanat stop, illetve a pálya újraindítása. Ezek például akkor hasznosak, ha a felügyelő élőszóban szeretne a tanulóval beszélni ezért addig leállítja a játékot, vagy a diák nagyon elveszett, és egyszerűbb újrakezdetni vele a feladatot.



13. ábra Játék megállítása majd elindítása

A játékok általános viselkedését is statikus parancsokkal irányíthatjuk. Ide tartozik az egyes játékok hangos súgóinak, hangeffektjeinek hangerő szabályozása. Ezek a súgók nagy segítséget jelentenek a gyermekek számára, amikor azok egyedül vagy kis csoportban gyakorolnak a játékokkal. Ilyenkor egyszerű utasításokkal közli a játék a feladatot, hogy akár az olvasásban gyenge gyermekek is segítség nélkül tudják élvezni a őket. Az egyes játékok különböző hangjelzéseket is ki tudnak bocsátani ezzel is jelezve a gyermekek felé a megoldás sikerességét. Akár ezekhez a hangokhoz is kapcsolódhatnak feladatok. Ez nagyobb csoportban, egy osztályteremben viszont ez zavaró lehet, mert a tanulók nem egyforma sebességgel haladnak a feladatok megoldásával, így szinte folyamatos zajforrást jelentene. Erre a problémára ad egy megoldást maga a keretrendszer a hangerő szabályozásával. Ezt a tanuló táblagépén is lehet konfigurálni, illetve ha menet közben mégis szükség van a segítségre, a pedagógus távolról is ki-/bekapcsolhatja ezt.

A játékokhoz beépítve tartozik egy úgynevezett dashboard is, melyen a játékos teljesítménye és fiziológiai jelei követhetőek valós időben. Ez főleg a tesztelés fázisában segíthet, hogy a játék futása közben, lokálisan láthassuk a keretrendszer által mért értékeket, illetve a szenzorok kapcsolódási állapotát. Ez távolról, a felügyelőalkalmazáson keresztül is megjeleníthető, eltüntethető a játék képernyőjéről.

Fontos, hogy a felügyelő a csatlakoztatott szenzorok állapotáról manuálisan is kérhessen frissítést, ha úgy látja, hogy a kapott értékek eltérnek a korábbiaktól és szenzor hibára gyanakszik.

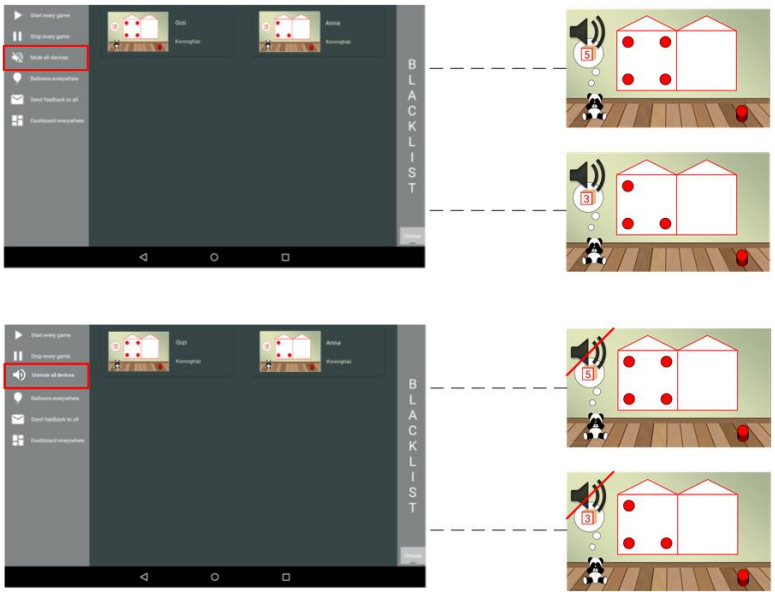
A képernyőképekhez is kapcsolódnia kell statikus parancsnak. Alap esetben, a nagy overhead elkerülése miatt, a képek kis méretben kerülnek elküldésre, hogy a felügyelő egy gyors pillantással átfogó képet kapjon a játék menetéről. Azonban ha gyermek elakad, vagy lassan halad, a segítségnyújtáshoz elengedhetetlen, hogy a tanár részletesen, nagy méretben lássa a gyermek játékát. Erre is lehetőséget nyújt az alkalmazás. Ilyenkor egy kérést küld a Supervisor a Framework felé, amely arra irányul, hogy az nagyobb méretben és felbontásban küldje a képeket. A nézet bezárásával újabb kérést küld, amivel visszaállnak az alap screenshot méretek.

Ezeken felül több statikus kategóriába sorolható parancs is elérhető, melyekről a későbbiekben bővebben lesz szó.

3.2.1.1.1. Egyéni

A felügyelő részleges megfigyelés közben az aktuálisan megfigyelt gyermek játékát tudja statikus illetve dinamikus parancsokkal irányítani. Ez hasznos, mivel ilyenkor a tanár részletes információval rendelkezik a gyermekről, akinek így hatékony segítséget tud nyújtani, illetve a keretrendszer számára is megfelelő információval tud szolgálni.

3.2.1.1.2. Csoportos



14. ábra Csoportos némítás

A felügyelőnek lehetősége van a statikus parancsokat akár az egész osztályterem számára kiadni. Nagyban segítheti munkáját, ha minden tanuló egyszerre kezdi el a feladatot, illetve ha nem kell minden egyes diák játékát egyesével leállítani. Erre mutat példát a 14. ábra.

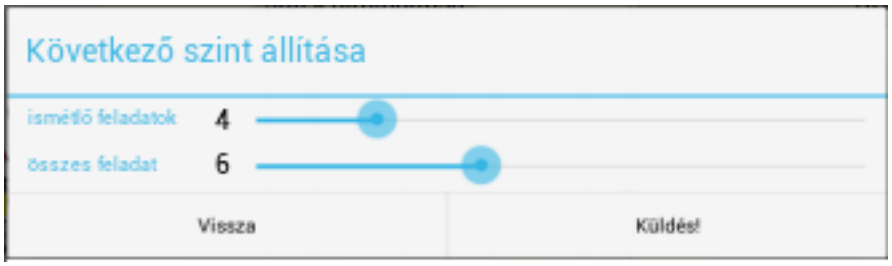
3.2.1.2. Dinamikus

Mivel minden játék különböző, így vannak olyan parancsok, melyek csak bizonyos játékokhoz értelmezhetőek. Az ilyenek segítik abban a pedagógust, hogy ne csak alap parancsokkal tudja a játékmenetet manipulálni, hanem specifikusan is segítségére lehessen a gyermekeknek. Ilyen lehet az adott játékban az ismétlő, gyakorló feladatok száma az egyes témakörökben, a feladat által feltett kérdések száma vagy akár a témakör kiválasztása is.

A dinamikus parancsok a dinamikus eseményekhez hasonlóan egy leíróval rendelkeznek (*TypeDescriptor*). Ezen leírókat játék szerint csoportosítva tárolja a Supervisor *RegistryModul* modulja. Az eseményeket a Framework számára mindig be kell regisztrálni, így a felügyelő is megkaphat egy leírót több, ugyanolyan játékot futtató eszköztől. Ez a parancsoknál nem igaz. Egy játék csak akkor küldi el a hozzá tartozó parancsokat, ha a kapcsolódás után a Supervisor észleli, hogy az adott játéktípustól még nincsenek regisztrált parancsai és ezt egy kérésben (*REQUEST_COMMAND*) jelzi a keretrendszer felé. Ekkor a játékban lévő megfelelően felannotált metódusok alapján létrejönnek a leírók és elküldésre kerülnek a felügyelő felé. A *@CommandMethod* annotáció jelzi, hogy az adott metódus dinamikus parancsként is használható, míg a paramétereit *@CommandParameterValue*-val kell ellátni. Egy ilyen paraméternek meg kell adni a nevét, valamint a minimum és a maximum értékét. A paraméterek típusa int, float, double (és ezek boxolt változatai) vagy String lehet. Ahhoz, hogy a parancs neve megjelenítésre kerüljön, azt is annotálni kell a *@DisplayName(name=String)* formátummal (az események és azok paraméterei is ilyen annotációval kaphatnak kijelzésre szánt elnevezést). Az egyes parancsokat a hozzájuk tartozó metódus nevével azonosítjuk.

A parancsleírók alapján a Supervisor dinamikusan felépített párbeszédablakokat generál, melyek a paraméterek bevitelére és a parancs elküldésére szolgálnak. A parancs *DYNAMIC_COMMAND* fejlécű, *CommandEvent* tartalmú üzenetként kerül elküldésre,

melyben szerepel a metódus neve és a paraméterek név-érték párosai (szöveges formában). A párbeszédablakok a paraméterek típusától függően beviteli mezőkből és csúszkákból állhatnak. Előbbibe szöveget utóbbiba pedig számadatot vihet be a felhasználó.



15. ábra: Korongház játék szint beállításai

A 15. ábrán látható egy ilyen dialógus ablak, melyen a Korongház játék következő számkörének paraméterei állíthatóak. A játék adott számú különböző típusfeladat után ismétlő feladatokat ad fel a gyermeknek. Ezek számát a felügyelő távolról is állíthatja.

3.2.2. Megjegyzés hozzáfűzése

A felügyelőnek lehetősége van megjegyzéseket fűzni az egyes játékmenetekhez. Ezek a kommentek a szerver irányába továbbításra kerülnek, és a későbbi analízist segíthetik. Ha például a felhasználót játék közben valami külső hatás érte, akkor ezt a tanár jelezni tudja, így az esetleges kiugró értékekre magyarázattal tud szolgálni, ezzel is segítve a kutatást és a játékos profilok tényleges valós adatokkal való feltöltését.

3.2.3. Algoritmus betanítása

A keretrendszerben, az adaptív szintváltoztatáshoz egy osztályozó algoritmus fut. Ez az algoritmus a rendelkezésére álló adatok alapján (a jelenlegi játékmenet, a mért fiziológiai adatok, az előző játékmenetek, valamint a pedagógus visszajelzése) osztályozza a felhasználó mentális állapotát, és változtatja a szinteket.

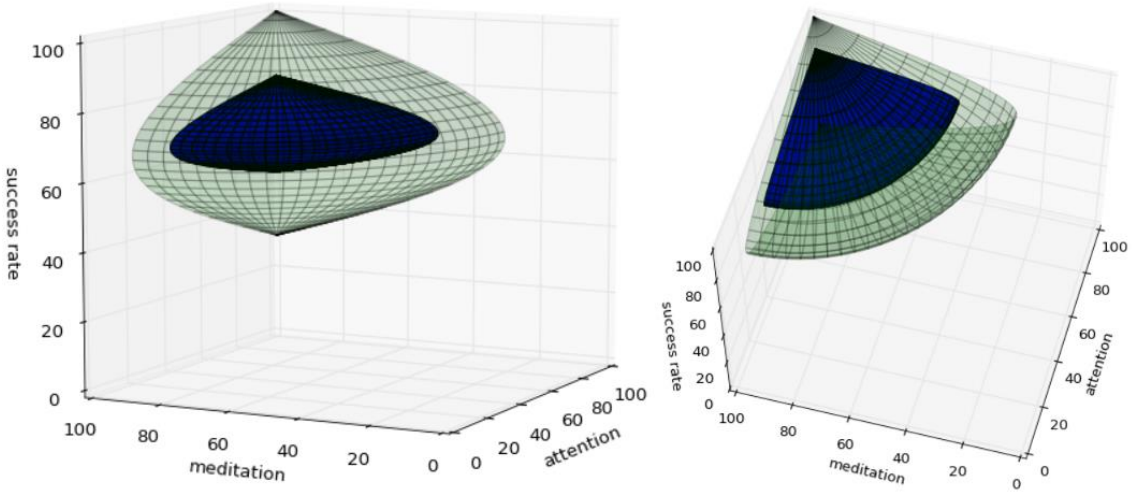
3.2.3.1. Az osztályozásról röviden

A keretrendszer által meghatározott mentális állapot három féle lehet: figyelem, fáradtság vagy unalom. Bizonyos esetben egy negyedik, null értéket is felvehet, amikor az azonosítás sikertelen volt, mert pontosan egyforma eséllyel tartozik a jelenlegi állapot két (vagy több) kategóriába.

Jelenleg többféle osztályozó algoritmus fejlesztése is folyik, hogy később ezekből ki tudjuk választani azt, ami a leginkább megfelelő az elvárásainknak, azaz nem használ túl sok erőforrást, de hatékonyan határozza meg az állapotokat. Ezt legegyszerűbben úgy oldhatjuk meg, ha közös interfészt alakítunk ki az ilyen jellegű moduloknak, hogy azok könnyen összehasonlíthatóak és cserélhetőek legyenek.

3.2.3.1.1. Nehézségi javaslat számító algoritmus

A keretrendszer tartalmaz többek között egy, a feladatmegoldások sikerességére és az EEG által számított értékekre épülő nehézség kalkuláló algoritmust [10]. Ez az aktuálisan mért értékek alapján igyekszik az optimális tanulást lehetővé tevő határok között tartani a felhasználót. Ezt ábrázolja a 16. ábra. Ez az algoritmus ugyanakkor nem nyilatkozik a felhasználó mentális állapotáról.

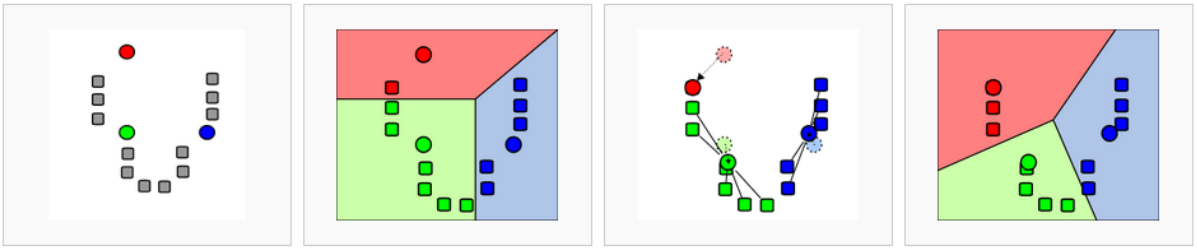


16. ábra Az optimális tanulást lehetővé tevő pontok

3.2.3.1.2. K-means algoritmus

A mentális állapot meghatározásához a K-means klaszterező algoritmus [11] egy módosított változatát használjuk. Ez az állapotvektorokat pozíciójuk alapján három klaszterbe (figyelem, fáradtság és unalom) sorolja. A hozzárendelés geometriai távolság alapján történik. Ehhez azonban először biztosítanunk kell az algoritmus számára legalább 1-1 állapotvektort, hogy az tudja, melyik kategória melyik mentális állapotnak felel meg. Ezután már többségi döntéssel képes meghatározni, hogy melyik klaszter melyik mentális állapothoz rendelhető, így képes önállóan is osztályozni az állapotokat. Az algoritmus

minden egyes új vektor beérkezésekor kiszámolja, hogy az melyik klaszter középpontjához van a legközelebb. Az új pont, osztályának meghatározása után, bekerül a klaszterbe, így módosítva a középpont helyét. Annak érdekében, hogy a nagyszámú régi vektor ne tegye statikussá a középpontot, az egy bizonyos időnél régebbi pontok mindig törlésre kerülnek. Az algoritmus így folyamatosan hangolja magát, és képes lekövetni a hosszú idő alatt végbemenő változásokat is.



17. ábra A három szín reprezentálja a három mentális állapotot

Ez a módszer tehát alkalmas arra, hogy a játékmenet során érkező mérési adatokat, melyek több szenzorból illetve forrásból jönnek (EEG számított és nyers adatok, szívritmus mérő adatok), egy-egy kategóriába soroljunk. Az adatok által képzett vektor komponensei azért tartoznak össze (és alkothatnak vektort), mert egy időben érkeztek és mind a játékos aktuális állapotát írják le. Ez a módszer már elméleti működéséből fakadóan is támogatja az esetleges szenzor- vagy adatforrások kiesését. Ilyen esetekben egyszerűen folyamatosan konstans értékkel – például 0-val – populálva a vektorok megfelelő komponensét a K-Means algoritmus szempontjából lényegtelenné válik az adott komponens és nem befolyásolja majd a döntést.

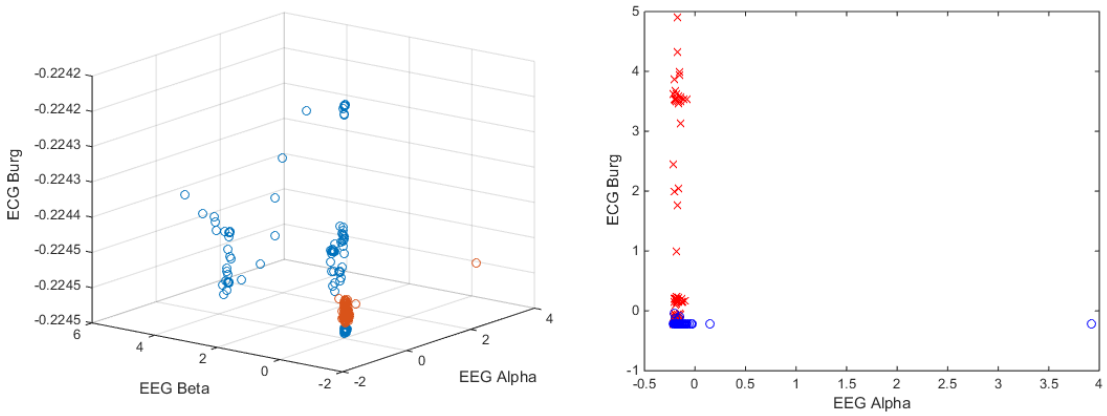
Az algoritmusnak a megfelelő működéshez szüksége van egy tanítóhalmazra. Ezt a tanítóhalmazt a keretrendszer be tudja szerezni a szerverről, ha voltak már korábbi játékmenetek és felhasználó-specifikus adatok.

Azonban, ha ilyenek nincsenek, akkor más külső forrásból kell beszereznie a tanítóhalmazt. A felügyelőalkalmazásnak erre is képesnek kell lennie. A tanár ezen keresztül tudja megadni az általa megfigyelt állapotot. Amikor a felhasználó elkezd játszani, a pedagógus figyeli a hangulatát, és a Supervisor alkalmazáson keresztül jelöli a gyermek mentális állapotát. Körülbelül 10-20 db vektorból álló tanítóhalmaz kell ahhoz, hogy az algoritmus jól működjön. Fontos, hogy a felügyelői javaslatok helyesek legyenek, mert ezek a játékmenet során nem kerülnek törlésre, tehát a hibás javaslat az egész játékmenetet eltorzíthatja.

Ezek alapján a visszajelzések alapján tudja elkezdeni az osztályozást a keretrendszer. Visszajelzést azonban nem csak a mérés megkezdésekor adhat a pedagógus. Ha az algoritmus már átvette az irányítást, és folyamatosan számol, a tanárnak akkor is lehetősége van befolyásolni azt. Közben is küldhet konfidencia adatokat, amelyekkel módosíthatja a már futó algoritmust.

3.2.3.1.3. SVM

Ezeken felül fejlesztés alatt áll egy lineáris SVM modell [12] is, amely később beépítésre kerülhet a keretrendszerbe. Ez az EEG és a szívritmusmérő adatait használja a mentális állapotok kiszámításához. A 18. ábra a modell működését mutatja egy mérés során. A kétdimenziós ábrán a kékek a relaxáció másodperceinek szenzoradatai, a pirosak azok a másodpercek, amikor 1-back (mentális terhelés) feladatot kapott az alany. A háromdimenziós képen a sárga a relaxáció, míg a kék pontok a 1-back állapot pillanatai. A képeken látható, hogy a különböző állapotok egész jól elkülöníthetőek egy síkkal/egyenessel.



18. ábra SVM modell által számított értékek 1-1 mérés során

3.2.4. Mini-játékok

Az egyes játékokhoz tartozhatnak úgynevezett mini-játékok. Ezek arra szolgálnak, hogy egy, az alapfeladattól teljesen független, és merőben más típusú játék elindításával a játékost kizökkentsék a jelenlegi állapotából, és visszavezessék egy normál,

relaxált állapotba. Az ilyen mini-játékok elindítását maga a keretrendszer is kezdeményezheti, ha a mért adatok alapján úgy ítéli meg, de a pedagógusnak is lehetősége van arra, hogy elindítson egy ilyet, ha szükségesnek érzi.

19. ábra Lufik a Korongház játékban

Egy ilyen mini-játék a lufi pukkasztás, melyet a 19. ábra szemléltet. Ennek elindítása után a játékos képernyőjén lufik kezdenek szállni, amiket neki ki kell pukkasztania. Ez segíti az elkalandozott figyelmű vagy éppen unatkozó gyermekek koncentrációját visszairányítani a játékra.

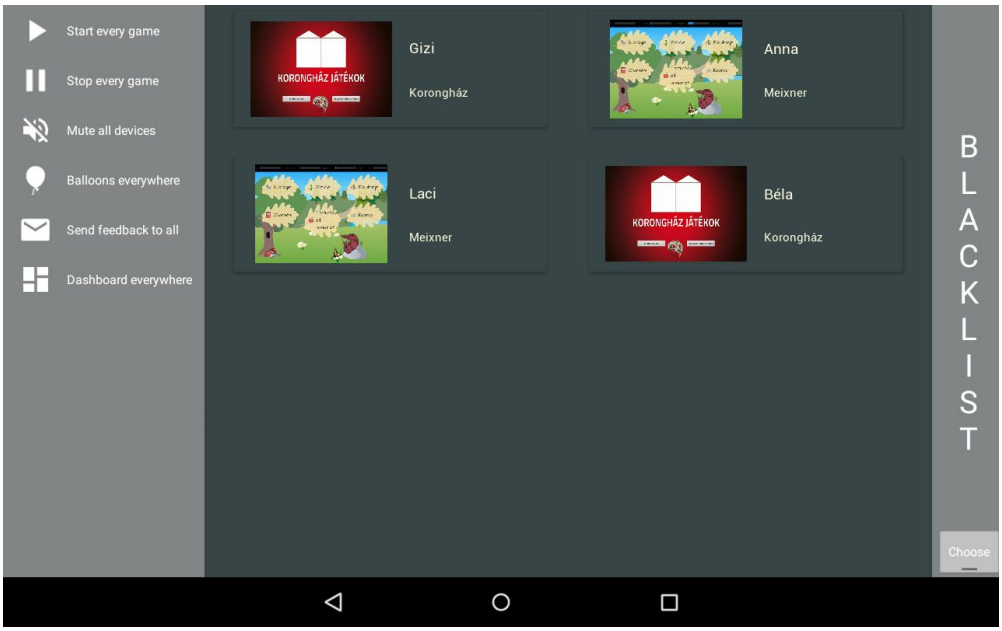
3.3. Felhasználói felület

A következőkben, képekben is bemutatjuk az általunk készített alkalmazást, amely a fent említett funkciók mindegyikére képes.

3.3.1. Áttekintő képernyő

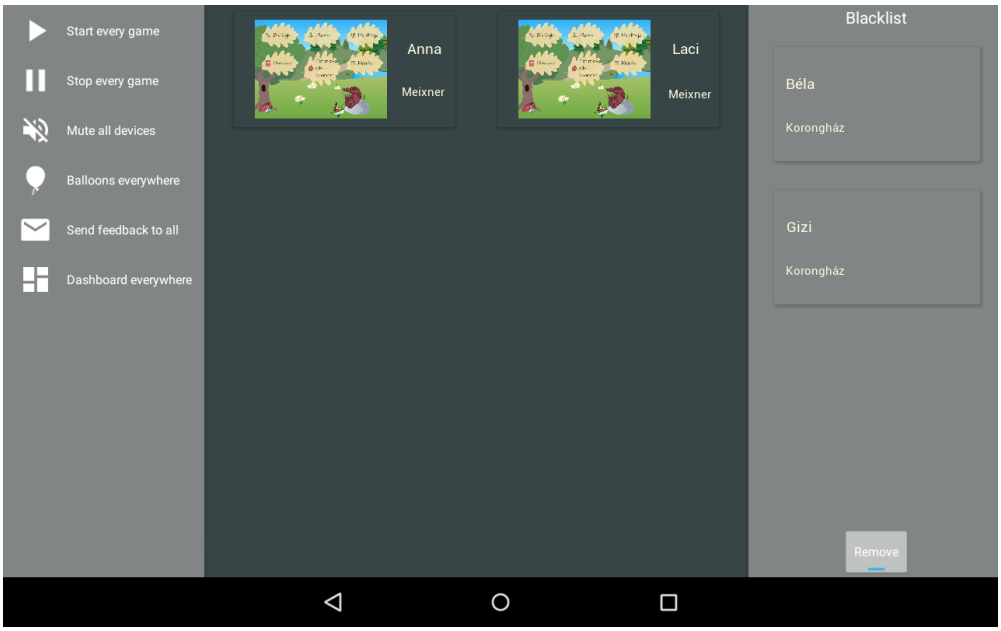
A felügyelő alkalmazás elindítása után a tanár a 20. ábra szerinti képet látja. Itt megjelennek az azonos WLAN hálózaton lévő eszközök, amelyeken fut a keretrendszer. Az egyes játékok egy képernyőképpel, a bejelentkezett felhasználó nevével, valamint a játék nevével vannak azonosítva. Amennyiben a játékokhoz szenzorok is csatlakoztatva vannak, úgy azok állapota is megjelenik. Az eszközöket reprezentáló kártyák zöldek, ha

a mérőműszer megfelelően van csatlakoztatva, pirosak, ha valamilyen hiba van. Így a felügyelőtanár könnyen áttekintő képet kaphat a teljes osztály szenzorjainak állapotáról, és rögtön értesül, ha valami nem megfelelően működik. Akkor is azonnali értesítést kap, ha a szenzorok nem hibásak, csak valamilyen kiugróan rossz értéket mérnek.



20. ábra Az eszközválasztó képernyő

Egy iskolában könnyen előfordulhat, hogy egyszerre több órán is használják a játékokat, így egyszerre többen jelennek meg a hálózaton, mint amennyi tanulót a tanár felügyelni szeretne. Ez nem csak átláthatatlanná teszi a képernyőt, de a felügyeletet is zavarja. Erre az eshetőségre a megoldásunk az, hogy a pedagógus feketelistára helyezheti azokat az eszközöket, amelyeket nem akar a továbbiakban figyelni. Ezt mutatja a 21. ábra. A tanár egyesével is válogathat, de szűrhet feladattípus vagy a felhasználói csoportok szerint is.



21. ábra A felhasználók szűrése

Az eszközválasztó képernyő bal oldalán is találhatóak parancsok. Az itt kiadott utasításokat minden felügyelt eszköz megkapja. Ezek a csoportos parancsok hasznosak lehetnek egy óra elején, végén, illetve akár egy számonkérés alkalmával is. Az eszközök elnémítására is gyakran szükség lehet egy osztályteremben, hogy a hangos súgók ne legyenek zavaróak.

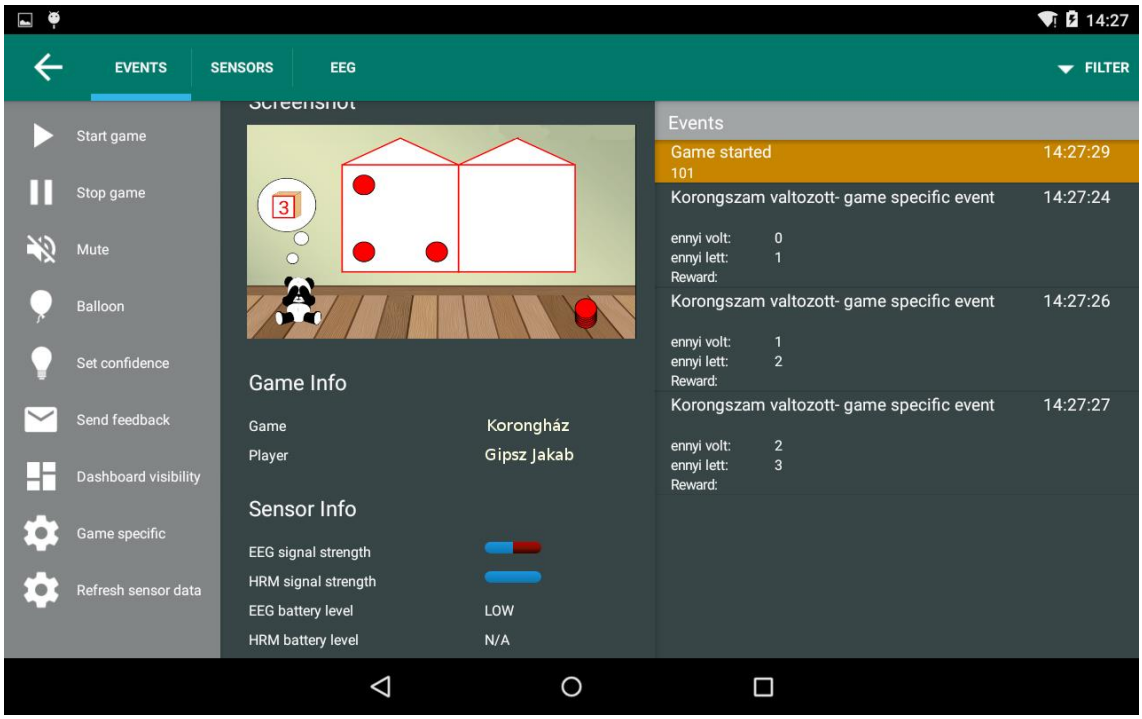
3.3.2. Főképernyő

Amennyiben a felügyelő egy tanuló részletes játékmenetére és szenzoradataira kíváncsi, úgy a kártyájára koppintva megnyithatja a főképernyőt. A főképernyő (22. ábra) 4 elkülöníthető részből áll. A fölső címsor bal oldalán található egy vissza navigációs gomb, amely az eszközválasztó képernyőre visz vissza. E mellett fülek találhatók, amelyekkel a jobboldali szegmensben megjelenő tartalmat változtathatjuk. A jobboldali fragmens dinamikusan van fölcsatolva a felületre, így a fülek által megjelenített felületek könnyen változtathatóak és bővíthetőek. A középső szegmens az a rész, ahol az általános adatok jelennek meg, ez mindig látható, a felület statikus része. A baloldalon található a parancsok fragmense, amely szintén dinamikusan van fölcsatolva, így mindig az aktuális parancsokat jeleníti meg. Ez a panel egy oldal irányú mozdulattal be is csukható, ekkor csak az ikonok láthatóak, a gombok azonban így is működnek. A középső elem fix szélessége miatt ekkor a jobboldali fragmens növekszik meg, ami segíti az azon megjelenő információk láthatóságát.

3.3.2.1. Események megjelenítése

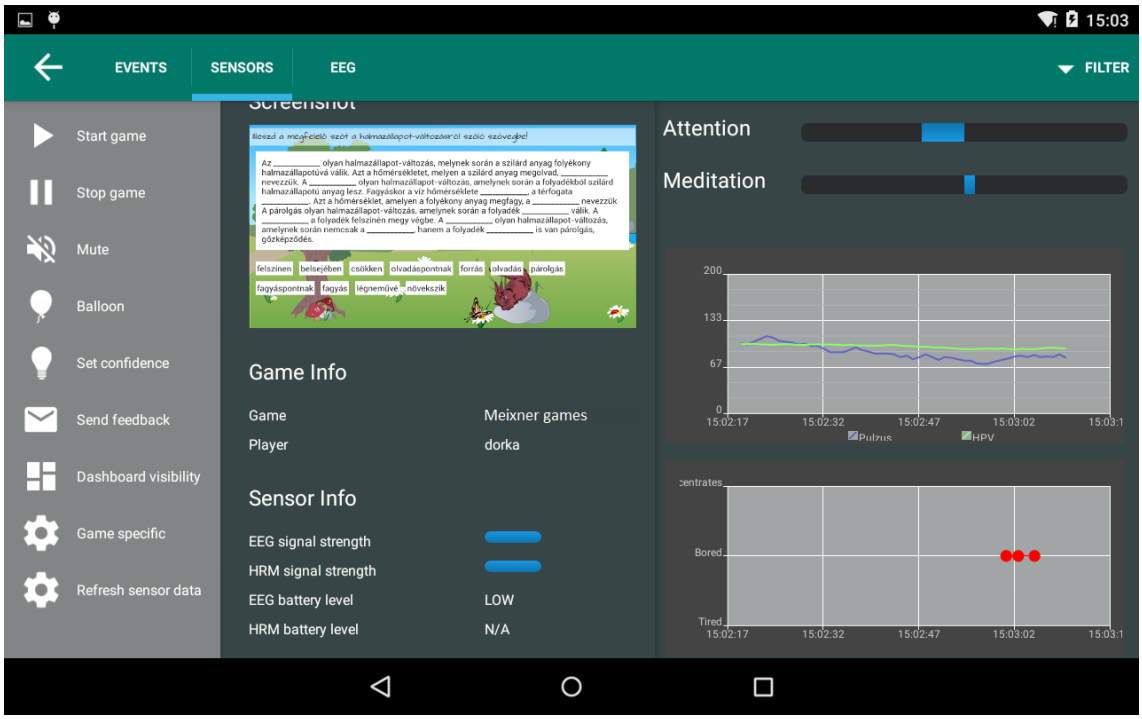
A részletesen megfigyelt tanuló játékeseményeit a felügyelő valós időben követheti végig. Az egyes eseménytípusok különböző színkódokkal jelennek meg, így a tanár könnyen elkülönítheti a számára fontos eseményeket. A 22. ábra jobb oldalán láthatjuk, hogy a játékos egymás után 3 korongot helyezett el a házon, de a feladatot még nem fejezte be a feladatot, mivel az erre vonatkozó eseményt még nem kapta meg a Supervisor. A beérkező események megjelenítése szűrhető is az alapján, hogy a pedagógus milyen típusokat szeretne látni.

Középen, a játék és a játékos adatai fölött helyezkedik el a képernyőkép, mivel ez az egyik legfontosabb követési módja a játékmenetnek. A pedagógus itt 10 másodpercenként frissülő képet kaphat a diák aktuális játékmenetéről.



22. ábra Korongház játék megfigyelése

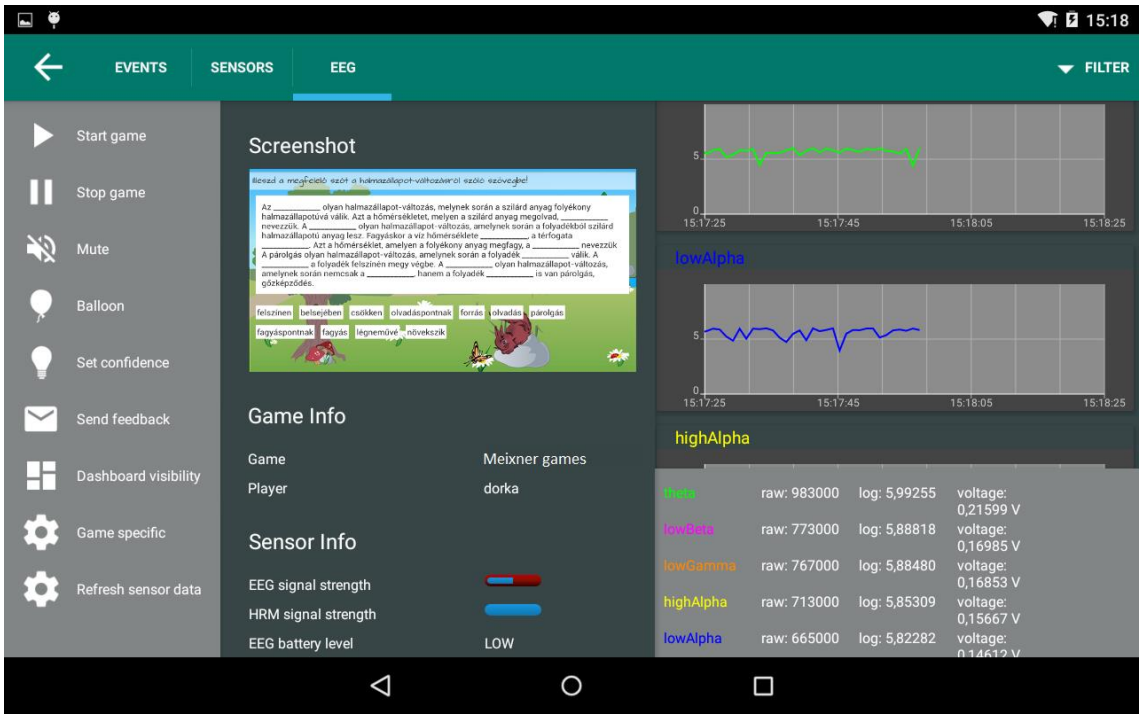
3.3.2.2. Fiziológiai adatok megjelenítése



23. ábra A főképernyőn a főbb fiziológiai adatok is megjelennek

A keretrendszer a hozzá csatolt szenzorok segítségével többféle fiziológiai adatot is szolgáltat. Ezek a felügyelő számára folyamatos képet biztosítanak a gyermekek állapotáról. Az egyes ilyen adatok grafikonokon és csúszkákon vannak ábrázolva. A következtetett figyelmi és nyugalmi állapotok egy 0-100-as skálán helyezkednek el, ahol a 0 érték az adott állapot hiányát jelenti, míg a 100 a teljes meglétét. Emiatt itt az 50-es értéktől való eltérést érdemes ábrázolni, ehhez két-két *ProgressBar*-t használunk, melyek egymáshoz vannak illesztve. A grafikonok kirajzolásához az *Androidplot*[13] könyvtárat vesszük igénybe. A szívritmus adatok historikus ábráján a pulzus és a szívritmus variancia jelenik meg. A keretrendszer által következtetett mentális állapot is grafikonon van ábrázolva. Ezen megjelennek a felügyelő által javasolt értékek is.

A 23. ábra középső részén láthatóak kiemelt szenzor információt. Ezek az egyes szenzorok jelerősségét illetve akkumulátor adatait jelenítik meg. Ilyen módon a felügyelő közvetlenül értesül, ha valamelyik szenzor hibásan működik és időben korrigálhatja azt.

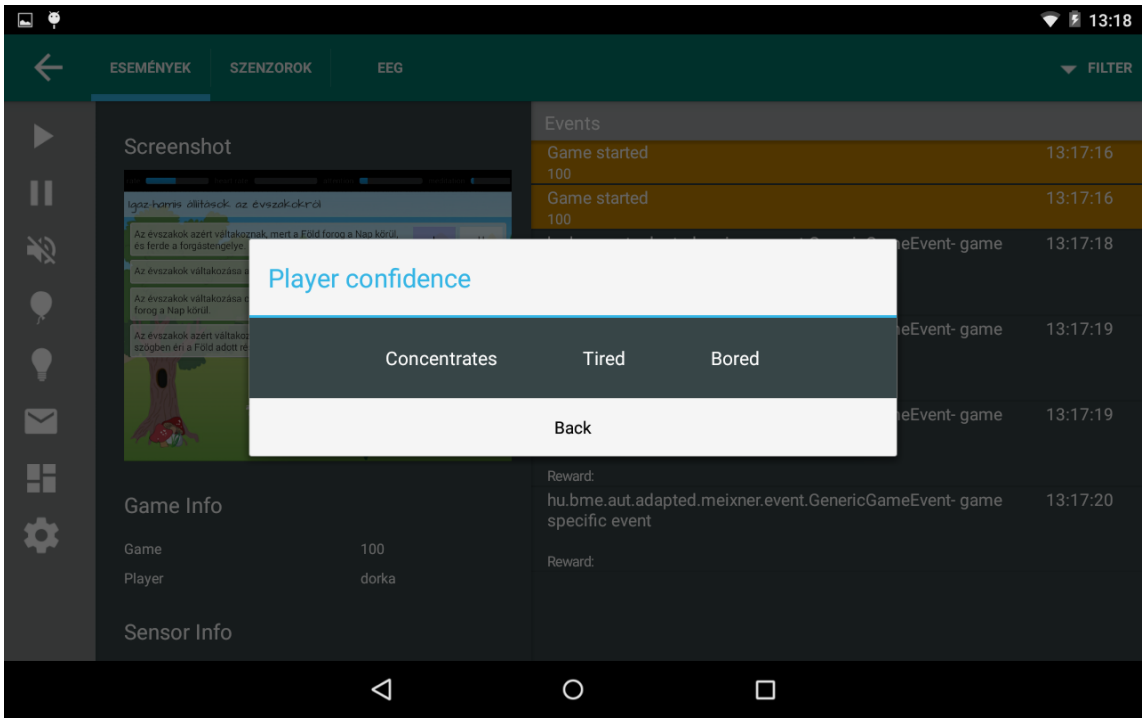


24. ábra Az EEG csatornáit külön-külön is megtekintheti a felügyelő

A 24. ábra azt mutatja, hogy az EEG fül kiválasztásával, a jobb oldali panelen a kilenc EEG csatorna külön is megjeleníthető, a belőlük származó részletes adatokkal együtt.

3.3.2.3. Parancsok megjelenítése

A baloldali szürke panel a parancsok helye. A fentebb már említett statikus parancsok (játék indítása, megállítása, némítás) közvetlenül jelennek meg a felületen, míg a dinamikus parancsok csak egy összefoglaló gombot kaptak, amelyet megérintve a konkrét parancsok egy felugró ablakban láthatóak. Ez a lépés helytakarékossági okokból szükséges. A visszajelzés küldése is egy külön felugró dialógusablakban történik, ezt mutatja a 25. ábra. Itt a felügyelő megadhatja, hogy a tanuló koncentrál, fáradt vagy unatkozik. Az ábrán az is látszik, hogy a parancsok panelja összecsukható, és így több hely marad a lényegesebb adatok megjelenítésére.



25. ábra Visszajelzés küldése a keretrendszernek

A felügyelő alkalmazás rendelkezik még egy egyszerű felülettel, ahol a tanár a teljes felbontású képernyőképet tekintheti meg. Erre a felületre a kis képernyőkép megnyomásával navigálhat át.

4. Tesztek

Az oktató játékok, és ezzel együtt a felügyelő alkalmazás, valós környezeti tesztjére több lehetőségünk is volt a Meixner Általános Iskola tanulóinak és pedagógusainak részvételével. Ezen alkalmak során lehetőségünk nyílt visszajelzést kapni az alkalmazások használhatóságáról és az esetleges hibáiról. A visszajelzések pozitívak voltak, élőben is megtapasztalhattuk, hogy a tanulási zavarral küzdő gyerekek mennyivel szívesebben használnak például egy olyan szoftvert, ahol egy pandamaci jelzi a feladat sikerességét. A tanároknak több továbbfejlesztési javaslatát is fontolóra vettük, ezek főleg az egyes játékok specifikus parancsaira vonatkoztak.

26. ábra Meixner játékok használata egy tanórán

A táblagépek jól beépíthetőek a tanórába, könnyedén használták mind a gyermekek, mind a felnőttek. A 26. ábra is egy ilyen természetismeret órán készült, amikor a gyerekek a digitális feladatokon gyakorolhatták a halmazállapot-változásról szerzett ismereteiket.

A 27. ábra egy fiatalabb gyerekekből álló csoport matematika gyakorlását mutatja be. A tanulók szinte természetes szinten kezelték a táblagépeket. A bal oldali gyermek többféle fiziológiai mérőeszközzel is meg volt figyelve, ám ezek sem zavarták a feladatmegoldásban, ami biztató a projekt további haladásának szempontjából.

27. ábra Diszkalkuliás gyermekek a Korongház használata közben

Természetesen szükség van további tesztelésekre is, ezek folyamatban vannak. A következő intézmény, amelyet bevontunk, a Budapesti Fazekas Mihály Gyakorló Általános Iskola és Gimnázium[14]. Emellett ezt a rendszert a Nemzeti Tehetségprogram egyik ága kapcsán végzett felmérések és gyakorlások során is alkalmazni fogják.

5. Összefoglalás

5.1. Eredmények, tapasztalatok

A projekt során láthattuk, hogy hogyan valósul meg az együttműködés több tudományterület között. Az informatika, a pszichológia, a pedagógia, a jelfeldolgozás valamilyen mértékben mind megjelenik a kutatás és a fejlesztés során. Ezen felül betekintést nyerhettünk egy valós projekt fejlesztési folyamatába.

Bekapcsolódásunk óta több komponens tervezésében és továbbfejlesztésében vettünk részt. Ezek közül a legfontosabb a felügyelőalkalmazás, melynek segítségével az oktatók könnyen és hatékonyan tudják a játékokat felügyelni. Ahogy azt a tesztek is mutatják a projekt jó irányba halad a legfontosabb kitűzött cél, a gyermekek számára is élvezetes és hatékony oktatás felé. Egy olyan szakaszba léptünk, ahol több alkalmazás is közel áll ahhoz az állapothoz, hogy azokat éles környezetben is használni lehessen.

5.2. Továbbfejlesztés

A továbbfejlesztés iránya már nagyban függ a tesztek során velünk együttműködő tanárok kéréseitől és észrevételeitől. A következő lépés a fejlesztés folyamán a tiltólista funkcióinak kibővítése. A felügyelő alkalmazásban jelenleg manuálisan lehet feketelistára helyezni a megfigyelni nem kívánt eszközöket. Ezt szeretnénk kibővíteni úgy, hogy akár a tanár órarendje alapján az alkalmazás képes legyen felismerni a megfelelő tanulócsoportot és azokhoz az eszközökhöz kapcsolódjon. Ezzel együtt mindenképpen meg kell hagyni a jelenlegi tiltólistás megoldást is azokra az esetekre, amikor a csoport összetételében előre nem látott változás következik be, vagy a pedagógus tanórán kívül is felügyelné a játékosokat.

A továbbiakban is folytatjuk a folyamatos fejlesztést, mivel már az osztálytermi tesztelés során láttuk, milyen fontos a korszerű eszközök felhasználásával a legtöbbet kihozni a bevált oktatási módszertanokból. Hisszük, hogy a jövő generációjának a kutatásaink és erőfeszítéseink eredményeképpen élvezetesebb és hatékonyabb lehet a tanulás. A pozitív visszajelzések bátorítanak minket arra, hogy tovább végezzük munkánkat ezen az izgalmas területen.

Ábrák jegyzéke

Irodalomjegyzék

[1] Lyon, G. Reid. "Learning disabilities." *The future of children* (1996): 54-76.

[2] Álvaro Fernández-López, María José Rodríguez-Fórtiz, María Luisa Rodríguez-Almendros, María José Martínez-Segura, "*Mobile learning technology based on iOS devices to support students with special education needs.*" Computers & Education 61 (2013): 77-90.

[3] Meixner Alapítvány honlapja (2015. október), http://meixner.hu/alapytvanyunk/

[4] L. Szegletes, M. Koles and B. Forstner, '*The design of a biofeedback Framework for dynamic difficulty adjustment in games*', 2014 5th IEEE Conference on Cognitive Infocommunications (CogInfoCom), 2014.

[5] Csíkszentmihályi Mihaly. *Flow – Az áramlat: A tökéletes élmény pszichológiája (Flow: The psychology of optimal experience. Vol. 41.)*
New York: HarperPerennial, 1991.

[6] Emotiv Epoc (2015. október), https://emotiv.com/epoc.php

[7] NeuroSky Mindwave Mobile (2015. október),
http://store.neurosky.com/products/mindwave-mobile

[8] Zephyr HxM BT (2015. október),
http://www.zephyranywhere.com/products/hxm-bluetooth-heart-rate-monitor

[9] KryoNet (2015. október): https://github.com/EsotericSoftware/kryonet

[10] Deáki Soma (Forstner Bertalan Dr.): *Androidos oktatójáték fejlesztése biofeedback alapú keretrendszerre*
Villamosmérnöki és Informatikai Kar 2014 TDK konferencia, Mobil alkalmazások szekció

[11] K-means klaszterező algoritmus, Kanungo Tapas, Mount D.M., Netanyahu N.S., Piatko C.D., Silverman R., Wu, A.Y., "*An efficient k-means clustering algorithm: Analysis and implementation.*" Pattern Analysis and Machine Intelligence, IEEE Transactions on 24.7 (2002): 881-892.

[12] SVM modell, Cortes, Corinna, and Vladimir Vapnik, "*Support-vector networks*." Machine learning 20.3 (1995): 273-297.

[13] Androidplot (2015. október): http://androidplot.com/

[14] Budapesti Fazekas Mihály Gyakorló Általános Iskola és Gimnázium (2015. október), https://www.fazekas.hu/

Függelék

csoport név	kód	leírás
CATEGORY_COMMUNICATION	0x00	kommunikáció (belső illetve Framework-Supervisor)
CATEGORY_STATIC	0x10	statikus események és hozzájuk tartozó parancsok
CATEGORY_REGISTRY	0x20	dinamikus adatok regisztrálása
CATEGORY_EVENT	0x30	dinamikus események
CATEGORY_PROTOCOL_ERROR	0x40	protokoll hiba
CATEGORY_UNKNOWN_ERROR	0x50	ismeretlen hiba
CATEGORY_SENSOR	0x60	szenzor események
CATEGORY_COMMAND	0x70	parancsok

1. táblázat üzenet típusok kategóriái

típus	kategória	kód	objektum típusa	használat
CHOSEN	CATEGORY_COMMUNICATION	0x06	boolean	részletes megfigyelésre kiválasztás
NOT_YOU	CATEGORY_COMMUNICATION	0x07	null	tiltólistára helyezés
YOU_AGAIN	CATEGORY_COMMUNICATION	0x08	null	tiltólistáról levétel
EVENT	CATEGORY_REGISTRY	0x00	GameEventTypeRegisteredEvent	játékspecifikus esemény leírójának regisztrációja
REWARD	CATEGORY_REGISTRY	0x01	RewardTypeRegisteredEvent	játékspecifikus jutalom leírójának regisztrációja
COMMAND	CATEGORY_REGISTRY	0x02	CommandTypeRegisteredEvent	játékspecifikus parancs leírójának regisztrációja
USER_INFO	CATEGORY_REGISTRY	0x03	UserRegisteredEvent	játékos információ regisztrálása
GAME_INFO	CATEGORY_REGISTRY	0x04	GameInfo	játék információk regisztrálása
REQUEST_COMMAND	CATEGORY_REGISTRY	0x06	null	kérés játékspecifikus parancsok leíróinak regisztrációjára
SCREENSHOT	CATEGORY_STATIC	0x00	ScreenshotEvent	képernyőkép küldésre (keretrendszeren belüli használatra)
GAME_START	CATEGORY_STATIC	0x01	GamePlayStartEvent/ null	játék kezdetét jelzi/játék távoli indítása

GAME_PAUSE	CATEGORY_STATIC	0x02	null	távoli pillanat stop
GAME_RESTART	CATEGORY_STATIC	0x03	null	távoli játék újraindítás
GAME_FINISHED	CATEGORY_STATIC	0x04	GamePlayStopEvent/ null	játék végét jelzi/játék távoli megállítása
LEVEL_RESTART	CATEGORY_STATIC	0x05	null	távoli feladat újrakezdés
MESSAGE	CATEGORY_STATIC	0x06	SupervisorFeedback	felügyelő megjegyzése a játékmenethez
CONFIDENCE	CATEGORY_STATIC	0x07	MentalState	felügyelő mentális állapot javaslata
SCREENSHOT_START	CATEGORY_STATIC	0x08	int	screenshot darabok összesített mérete
SCREENSHOT_PART	CATEGORY_STATIC	0x09	byte[]	screenshot darab
DIFFICULTY_CHANGED	CATEGORY_STATIC	0x0A	double	megváltozott nehézség
LEVEL_FINISHED	CATEGORY_STATIC	0x0B	long	pálya vége (időbélyeg)
LEVEL_START	CATEGORY_STATIC	0x0C	long	pálya kezdete (időbélyeg)
SENSOR_EMOTION	CATEGORY_SENSOR	0x00	MentalStateCalculatedEvent	kikövetkeztetett mentális állapot
SENSOR_SIGNAL_STRENGTH	CATEGORY_SENSOR	0x02	SignalStrength	Emotiv EPOC eeg jelerősség
SENSOR_BETA	CATEGORY_SENSOR	0x03	FrequencyAnalysisResult	Emotiv EPOC eredmény
SENSOR_NEURAL_LOAD_INDEX	CATEGORY_SENSOR	0x04	NeuralLoadIndexEvent	egyszerű mentális állapot
SENSOR_NEUROSKY	CATEGORY_SENSOR	0x05	AttentionChangedGameEvent	figyelmi szint
SENSOR_HEART_RATE	CATEGORY_SENSOR	0x06	HeartRateChangedEvent	szívritmus adatok
SENSOR_MEDITATION	CATEGORY_SENSOR	0x07	MeditationChangedGameEvent	nyugalmi szint
NEUROSKY_SIGNAL_STRENGTH	CATEGORY_SENSOR	0x08	NeuroskySignalStrengthEvent	neurosky eeg jelerősség
LOW_BATTERY	CATEGORY_SENSOR	0x09	LowBatteryEvent	akkumulátor szint
SENSOR_POWER	CATEGORY_SENSOR	0x0A	EEGPowerEvent	Neurosky powerspektrum
SENSOR_STATUS	CATEGORY_SENSOR	0x0B	SensorStatusEvent	szenzor állapot

DYNAMIC_EVENT	CATEGORY_EVENT	0x00	GameDynamicEvent	dinamikus esemény
REWARD_EVENT	CATEGORY_EVENT	0x01	RewardGivenEvent	jutalom
DYNAMIC_COMMAND	CATEGORY_COMMAND	0x00	CommandEvent	játékspecifikus parancs
SCREENSHOT_MODIFY	CATEGORY_COMMAND	0x01	boolean	screenshot méret változtatás
MUTE_HELP	CATEGORY_COMMAND	0x03	boolean	hangos súgó némítás/hangosítás
DASHBOARD_VISIBLE	CATEGORY_COMMAND	0x04	null	játék dashboard láthatóság
DISPLAY_BALLOONS	CATEGORY_COMMAND	0x05	null	minijáték indítás
REFRESH_SENSOR_STATUS	CATEGORY_COMMAND	0x06	null	szenzor állapot lekérdezés

2. táblázat üzenet típusok