TDK Dolgozat

Okostelefon-alapú keretrendszer a Jövő Internetének valós idejű, multiszenzoros mobil egészségügyi alkalmazásaihoz

Készítette: Varga Norbert

Konzulens: Bokor László

2014. október 22.

Tartalomjegyzék

1. BEVEZETÉS

Napjainkban a hálózati szolgáltatásokat igénybe vevő felhasználók száma egyre nagyobb, az Internet nélkülözhetetlen információs közművé vált. A teljes mobil Internet adatforgalom várhatóan több mint tízszeresére fog nőni a következő öt év folyamán. A növekvő teljesítményű okostelefonok és hálózatra kapcsolódó beágyazott és mobil eszközök (Internet of Things – IoT) rohamos terjedése olyan új kihívásokkal állítja szembe a hagyományos mobil Internet architektúrákat, melyekkel egyre nehezebben birkóznak meg [1].

A Jövő Internet mobil architektúráinak már körvonalazódó általános koncepciója lehet a megoldás a fentebbi problémákra [2]. A vízionált nagyszámú, heterogén, átlapolódó vezeték nélküli hálózat és az elosztott, jól skálázható hálózati struktúra megteremti a lehetőséget, hogy állandó, mindenhol jelenlévő szolgáltatásokat biztosítsunk az újgenerációs alkalmazások számára. Ezen új szolgáltatások táborában egyre jelentősebbé válik az e-egészségügy (eHealth) [3] területe. Itt fontos csoportot alkotnak a mobil egészségügyi (mHealth) [4] rendszerek, melyek a különböző hozzáférési hálózatok erőforrásait, az okostelefonokat és azokhoz csatlakoztatott szenzorokat kihasználva igyekeznek növelni az egészségügyi szolgáltatások hatékonyságán, például költséghatékony mobil távfelügyeletet, távdiagnosztikát, vagy éppen mobil telekonzultációt biztosítva [5]. Ezeknek az alkalmazásoknak, olyan speciális QoS és QoE igényeik vannak, melynek kielégítése nagy kihívást jelent.

Különösen fontos figyelmet igényelnek az mHealth terület valós-idejű szolgáltatásai (pl. egy orvosi konzílium által távolról irányított beavatkozások/vizsgálatok), melyek még speciálisabb igényekkel rendelkeznek (kevés csomagvesztés, gyors reakcióidő, tehát kis késleltetés és jitter stb.). Sok mHealth forgatókönyv támaszkodik különböző viselhető vagy beépített szenzorok használatára (pl. ECG, ultrahang, szívritmus monitor, nagy felbontású kamera stb.). Ilyen szenzorokkal integrált alkalmazások hálózati folyamai eltérő tulajdonságúak lehetnek.

A hatékony és állandó szolgáltatás nyújtás érdekében, olyan rendszert kell tervezni, mely elosztott, jól skálázható, kihasználja az aktuálisan rendelkezésre álló, eltérő rádiós technológiák által biztosított erőforrásokat, és a különböző tulajdonságú folyamoknak mindig optimális átvitelt biztosít, bármilyen mobilitási esemény esetén. Mindehhez megfelelő támogató keretrendszerre van szükség.

TDK dolgozatomban pontosan egy ilyen, a Jövő Internet heterogén architektúráiba illeszkedő, mobil egészségügyi támogató keretrendszert javaslok, és részletezem a tervezési és implementációs részleteket, valamint a teljesítményelemzés eredményeit. A javasolt megoldás központi része egy okostelefon-vezérelt, folyam alapú mobilitás-kezelési és döntési keretrendszer, mely a legkülönbözőbb szenzorinformációk QoS/QoE igényei és a rendelkezésre álló hálózatok aktuális állapota alapján képes döntést hozni az mHealth alkalmazás folyamainak és az elérhető hozzáférési hálózatoknak az optimális összerendeléséről.

A második fejezetben a vonatkozó irodalomkutatás olvasható. Ebben a fejezetben bemutatom a mobil egészségügy fogalmát, szolgáltatás típusait, fókuszálva a valós idejű, egészségügyi szenzor alapú mHealth forgatókönyvekre, továbbá az mHealth alkalmazások hatékony mobilitás-kezelésére található javaslatokra. Az irodalomkutatás során kiemelt figyelmet fordítottam arra, hogy melyek azok a források, amik nem csak elméleti javaslatot tesznek a probléma megoldására, hanem valós implementációt tartalmaznak, hiszen az én munkám is valós rendszerekre történő implementáción alapszik.

A harmadik fejezetben a javasolt keretrendszer általános bemutatása következik. A keretrendszer komponenseit, azok implementációs részleteit a negyedik fejezetben fejtem ki.

Az ötödik fejezetben pontos leírást adok a tervezett mérési tesztkörnyezetről, továbbá bemutatom és elemzem a mérési eredményeket. A hatodik fejezetben összefoglalom TDK dolgozatomat, majd kitérek a továbbfejlesztési lehetőségekre, és a témához kapcsolódó jövőbeli kutatási irányokra.

2. A VONATKOZÓ IRODALOM BEMUTATÁSA

2.1. MOBIL EGÉSZSÉGÜGYI SZOLGÁLTATÁSOK ÁTTEKINTÉSE

Napjainkban az egyre nagyobb hardveres teljesítményű, több interfésszel rendelkező mobil eszközök és a vezeték nélküli, heterogén hozzáférési technológiák terjedésének, valamint az egészségügyi szenzorok olcsóvá és nagy tömegben elérhetővé válásának köszönhetően egyre nagyobb figyelmet kapnak a mobil egészségügyi (mHealth) szolgáltatások [6]. Az mHealth az elektronikus egészségügy (eHealth) területének egy csoportja, melynek célja a hagyományos egészségügyi szolgáltatások kiterjesztése mobil eszközökön keresztül a kommunikációs technológiákat felhasználva [5]. Az mHealth számos forgatókönyvet foglal magába, ide tartoznak (a teljesség igénye nélkül) a valós-idejű távfelügyeleti, diagnosztikai, fitnesz és sport alkalalmazások, telekonzultációs, betegség nyomonkövetési, és az életmód tanácsadást biztosító szolgáltatások [7].

A mobil készülékek különféle viselhető, egészségügyi szenzorral kapcsolhatóak össze vezeték nélküli technológiákat alkalmazva, Body Area Network (BAN) hálózatot alkotva, mely ezáltal fontos szerepet játszik a mobil távfelügyelet, telekonzultáció és otthoni egészségügy (home healthcare) területeken. TDK dolgozatom során elsősorban a valós idejű, multi szenzor alapú mobil távfelügyeleti és telekonzultációs szolgáltatásokra fókuszáltam, mivel ezen esetek még speciálisabb igényekkel rendelkeznek, így kiemelten fontos, hogy egy hatékony, optimális átvitelt biztosító keretrendszert biztosítsunk a számukra.

Ezen speciális igények közé tartozik többek között a kevés csomagvesztés, a gyors reakcióidő, tehát a kis késleltetés és a jitter. Elsősorban valós idejű, vészhelyezeti forgatókönyveket támogató mHealth alkalmazások esetén fontosak különösen ezek a követelmények. Ilyen a valós idejű beteg monitorozás, az irányított diganosztika (pl. remote ultrasound) vagy irányított beavatkozás (pl. remote surgery). Ilyenkor a kórháznak/krízisközpontnak folyamatosan monitoroznia kell az egymáshoz viszonyítva kis időbélyegekkel érkező életjel információkat/orvosi multimédia-tartalmakat, melyeknél minden egyes értéknek vagy adatcsomagnak fontos szerepe lehet, így a csomagvesztés nem elfogadható. Mivel valós idejű támogatásról van szó, a támogatást nyújtó oldalnak azonnal reagálni kell a beteg állapotában bekövetkezett eseményekre, emiatt a kétirányú kommunikációban a késleltetés és jitter magas értéke sem megengedett.

Az átvitel QoS paramétereinek sokkal fontosabb szerepe van ezekben az mHealth szolgáltatásokban, mint egy általános Android alkalmazásban, így a biztosításuk kiemelt figyelmet és megfelelő hálózati megoldásokat kíván [8].

A mobil távfelügyelet az mHealth alkalmazások azon csoportja, mely viselhető, mobil készülékekhez csatlakoztatott egészségügyi szenzorok alapján méri a felhasználó különféle életjeleit (pl.: ECG, EMG, vérnyomás, testhőmérséklet, véroxigén, vércukor, légzési ritmus stb.), majd ezeket a mért adatokat továbbítja a kórházba/egészségügyi krízisközpontba a rendelkezésre álló hozzáférési hálózatokon keresztül. A [9] szerzői kategorizálják a mobil távfelügyeleti szolgáltatásokat, megpróbálják behatárolni a folyamat fő szereplőit és egy általános támogatási architektúrával szemben támasztott követelményeiket. Ők három kategóriába sorolják a mobil távfelügyeleti szolgáltatásokat: krónikus betegségeket, sürgős eseteket felügyelő rendszerek és járó beteg felügyeletet megvalósító szolgáltatások. A krónikus betegségeket felügyelő rendszerek a mobil távfelügyeleti rendszerek legegyszerűbb típusa. Célja, hogy a páciens készülékéről a kórházba/egészségügyi központba feldolgozás nélkül tovább küldött egészségügyi szenzorok (pl.: vérnyomásmérő, vércukormérő, légzés monirotozó stb.) adatait analizálja, és olyan eltéréseket keressen az adatokban, melyek ismert krónikus betegségek indikátorai lehetnek, ezáltal elősegítve a korai megelőzésüket. A sürgősségi távfelügyelet azokat a helyzeteket fedi le, amikor a páciens állapotában hirtelen változás lép fel, és azonnali beavatkozásra van szükség az orvosok/egészségügyi krízisközpont részéről, esetleg egy távvezérelt gyógyszeradagoló berendezés által. Ezek a szolgáltatások folyamatos, meghatározott QoS és QoE követelményű monitorozást igényelnek, így ezeknél az eseteknél különösen fontos a hatékony hálózatkezelési eljárások biztosítása, bármilyen mobilitási eseményről is legyen szó (pl. a páciens szállítása, gyors mozgása közben is a kívánt minőségű legyen a kapcsolat).

A telekonzultációs alkalmazások célja, hogy optimális környezetet biztosítsanak a páciens és orvosi/egészségügyi központ között zajló előzetes konzultációkhoz, irányított beavatkozásokhoz. A telekonzultáció során lehetőséget kell biztosítani az egészségügyi szenzorok által mért és egyéb media adatok (pl. MRI képek, ultrahang felvételek, CT képek, vizsgálati eredmények stb.) megosztására az elérhető hozzáférési hálózatok erőforrásait optimálisan kihasználva [10]. A telekonzultáció segítségével gyorsabb és költséghatékonyabb konzultációs lehetőséget teremthetünk meg a páciensek (akik földrajzilag elszigetelten vagy

nagy távolságban tartózkodhatanak a kórháztól) és az orvosok között. A [11] szerzői videókonferencia alapú telekonzultációs lehetőséget biztosítottak a pacienseknek. Azt vizsgálták, hogyan viszonyulnak a betegek a telekonzultáció koncepciójához. A felmérésekből az derült ki, hogy a résztvevők szívesen használják ezeket a lehetőségeket és elégedettek a videókonferencia nyújtotta környezettel. Az irányított beavatkozások esetében egy orvos/orvoscsoport nyújt távoli segítséget egy bevatakozáshoz/vizsgálathoz mobil eszközöket és infokommunikációs technológiákat felhasználva.

2.2. HATÉKONY, FOLYAM-SZINTŰ MOBILITÁS-KEZELÉS

Ahhoz, hogy optimális architektúrát és szolgáltatási környezet teremtsünk az mHealth alkalmazások számára, lehetőséget kell biztosítanunk az elérhető, heterogén rádiós hálózatok által szolgáltatott erőforrások hatékony kihasználására. A heterogén hálózatokban rejlő előnyök akkor használhatóak ki, hogyha a különböző típusú hozzáférési hálózatok között optimálisan kezeljük a mobilitást. Ezt úgy érhetjük el, ha az alkalmazások hálózati folyamait a megfelelő elérhető interfészhez rendeljük egy komplex döntési algoritmus által vezérelve. A döntési algoritmusnak részletesen fel kell térképeznie az aktuális alkalmazási és hálózati kontextust. A vertikális hívásátadás és a folyam alapú mobilitás-kezelési mechanizmus, integrálva rétegek-közötti (cross-layer) optimalizásiós sémával lehet az alapja az ilyen algoritmusoknak a Jövő Internet mobil architektúráiban. Az irodalomban fellelhető cikkek nagy része a folyam-alapú mobilitás-kezelés protokoll szintű magvalósítási lehetőségeit tárgyalja (pl.: [12], [13], [14]).

Az általam javasolt keretrendszerben a MIP6D-NG [15] megoldja a protokoll szintű mobilitási feladatokat, mely a mobil IPv6 protokoll család és kiegészítésein (pl.: Flow Bindings, Multiple Care-of Addresses Registration) alapszik. A MIP6D-NG működését a későbbi fejezetekben részletesen mutatom be. A [16] szerzői egy több-paramétert figyelembe vevő döntési logikával integrált vertikális handover mechanimust mutatnak be, mely képes kezelni a különböző hálózati folyamokat. A bemutatott mobilitási forgatókönyvben két különböző típusú folyamot (FTP és VoIP) definiálnak. Esetükben a mobilitást a hálózat menedzseli egy nagyrészt csak elméletben létező ún. tartalom központú hálózatban (Content Centric Network - CCN). Ezzel szemben az általam javasolt keretrendszerben egy valós, IPv6-alapú, kliens vezérelt mobilitási megoldás működik. Az első irodalomban publikált Flow Bindings implementációt [17] a cikk mutatja be. Ez az implementáció linux alapú disztribúciókra

készült, azonban csak NEMO környezetben képes működni, hagyományos mobil eszközök nem képesek hálózati folyamok regisztrálására, illetve interfészek közötti mozgatására.

Az mHealth keretrendszerenbe integrált mobilitás-kezelési mechanizmushoz a [18] cikkben olvasható megközelítés áll a legközelebb. A cikk szerzői egy mobilitás-kezelési koncepciót mutatnak be Android platformra. Ahhoz, hogy lehetővé tegyék az Android rendszerben található hálózati interfészek szimultán hazsnálatát, ők is egy egyedi ROM-ot készítettek. A cikkben tárgyalt koncepció alapja az IEEE 802.21 Medai Independent Handover (MIH) szabvány. Azonban ebből munkából teljesen hiányzik a folyam-szintű mobilitás kezelés és a komplex döntési algoritmus által vezérelt mobilitás egyaránt.

2.3. AZ mHEALTH SZOLGÁLTATÁSOK IMPLEMENTÁCIÓJA

Napjainkban számos mHealth szoftver tölthető le mobil készülékekre az egyes alkalmazás boltokból. Az mHealth terület rengeteg különböző forgatókönyvet fed le. Az alkalmazások között találunk fitnesz és training, kalória számláló, egészséges táplálkozást segítő, távoli betegmonitorozást megvalósító, alvás monitorozó, gyógyszerszedési emlékeztető, általános életviteli tanácsadó és telekonzultációs szoftvereket egyaránt. Az alkamazás boltokban elsősorban inkább a fitnesz és a életviteli tanácsadást nyújtó alkalmazások és hozzájuk tartozó felhőszolgáltatások a legelterjedtebbek. Az általam javasolt keretrendszer támogatja az összes mHealth forgatókönyvet, azonban a fent említett elterjedt use-case-ek általában nem igényelnek speciális QoS követelményeket, esetükben nem elsődleges szempont a valós idejű, késleltetés kritikus működés, például a vészhelyzetekben használt alkalmazásokkal ellentétben. Emiatt én elsősorban azokat a forgatókönyveket tartottam szem előtt, melyeknél elengedhetetlen a hatékony támogató keretrendszer megvalósítása (pl.: telekonzultáció, valós idejű távfelügyelet és irányított beavatkozás [19].

Az irodalomban több olyan cikkel is találkozunk, melyek okostelefon alapú távfelügyeleti és telekonzultációs szolgáltatásokat valósítanak meg. A [20] cikk szerzői egy olyan architektúrát mutatnak be, mely lehetővé teszi az egészségügyi adatok továbbítását a kórházba/egészségügyi központba a páciens mobil eszközén keresztül az elérhető vezeték nélküli hálózatokat felhasználva. A bemutatott rendszer egy teljes végfelhasználótól végfelhasználóig terjedő koncepciót mutat be. Hasonlóan ehhez, a [21] szerzői is egy olyan szoftver platformot javasolnak, mely lehetővé teszi az egészségügyi adatok (ECG és vérnyomás) átvitelét, feldolgozását, riportolását. A [22] cikk egy olyan távfelügyeleti

információs architektúrát mutat be, mely lehetővé teszi az orvosok számára, hogy mobil eszközük segítségével böngészhessék pacienseik egészségügyi adatait.

A fentebb bemutatott mHealth szolgáltatások implementációjának az alapját a hagyományos mobility-kezelési eljárások és hálózati protokollok képzik. Ezek a rendszerek a Jövő Internet mobil architektúráiban természetüknél fogva nem lehetnek képesek optimális átvitel biztosítására az olyan alkalmazások számára, melyek valós idejűek, magas QoS/QoE és hálózati erőforrás követelményekkel rendelkeznek.

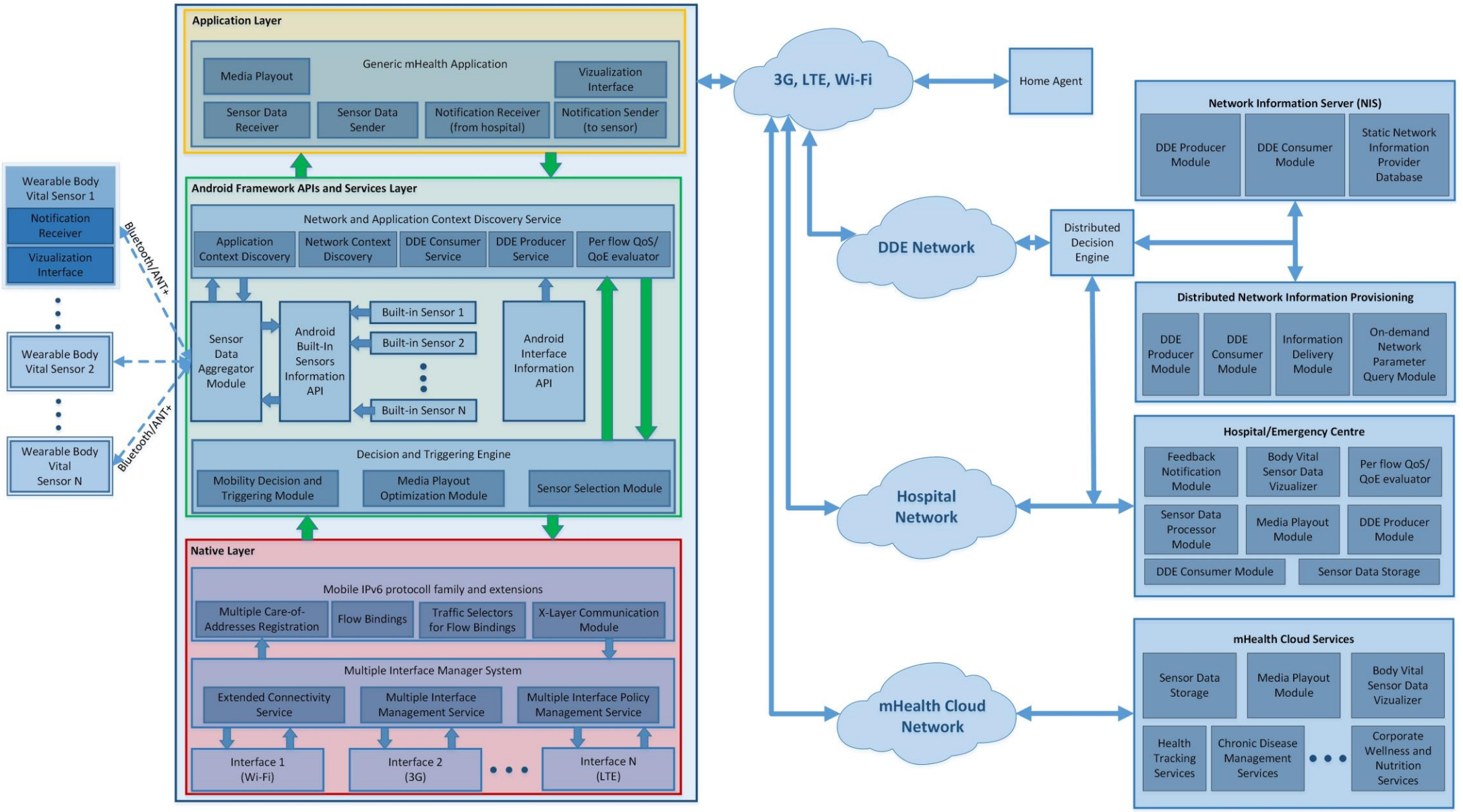
Az átfogó irodalomkutatás során arra a következtetésre jutottam, hogy jelenleg nem létezik olyan rendszer az irodalomban, mely valós implementációra épül, és folyam-alapú, cross-layer optimalizált keretrendszert biztosítana az alkalmazások számára. Ennek alapján TDK dolgozatomban egy, a Jövő Internet mobil architektúráiba illeszkedő, mobil egészségügyi támogató keretrendszert tervezek, implementálok és elemzek. A javasolt megoldás központi része egy okostelefon-vezérelt, folyam alapú mobilitás-kezelési és döntési keretrendszer, mely a legkülönbözőbb egészségügyi szenzorinformációk QoS/QoE igényei és a rendelkezésre álló hálózatok aktuális állapota alapján képes döntést hozni az alkalmazás folyamainak és az elérhető hozzáférési hálózatoknak az optimális összerendeléséről és a az orvosi multimédia-átvitel paramétereiről. A javasolt keretrendszer részleteit a következő fejezetekben mutatom be.

3. A JAVASOLT KERETRENDSZER ÁLTALÁNOS ARCHITEKTÚRÁJA

Ebben a fejezetben a Jövő Internet heterogén hozzáférési rendszereire és elosztott hálózati megoldásaira támaszkodó mobil egészségügyi alkalmazások számára javasolt támogató keretrendszer általános architektúráját mutatom be. A koncepció számos saját fejlesztésű elemet tartalmaz, és ezeken kívül is integrál több, az irodalomban már létező komponenst és modult. A komponensek a javasolt keretrendszerben definiált kommunikációs és együttműködési séma alapján dolgoznak össze, és együttesen egy elosztott, jól skálázható, az mHealth szolgáltatások számára optimalizált hálózati erőforrás kihasználást biztosító egységes architektúrát alkotnak. Az alfejezetekben bemutatom a rendszer fontosabb komponenseit és a közöttük lévő kapcsolatokat.

3.1. A KERETRENDSZER ÁLTALÁNOS SÉMÁJA

A javasolt keretrendszer általános architektúráját az 1. ábra mutatja be. A bemutatott rendszer egyik központi eleme az Android operációs rendszert futtató felhasználói végberendezés, okostelefon. Ehhez a készülékhez eltérő életjeleket és folyamatokat monitorozó egészségügyi és általános szenzorok tartoznak, melyek vagy külöböző vezeték nélküli technológiákkal (pl. Bluetooth, ANT+) csatlakoznak hozzá, vagy magába az okostelefonba vannak integrálva. A szenzorok által biztosított egészségügyi és média adatokat használják fel a mobil terminálon futó mHealth alkalmazások. A készülék egy olyan keretrendszert biztosít a futó alkalmazások számára, melynek segítségével azok optimálisan használhatják az aktuálisan elérhető hálózati erőforrásokat, kerüljenek bármilyen mobilitási eseménybe. A keretrendszer folyamatosan monitorozza az alkalmazások igényeit (elsősorban a használt szenzorok és átvitelre szánt adatfolyamaik QoS és QoE követelményei alapján) és feltérképezi az elérhető hozzáférési hálózatok paramétereit. Az utóbbi folyamatban a hálózati információk egy részét a mobil készüléken futtatott mérések szolgáltatják, másokat pedig az Eloszott Döntéstámogatási Motor (Distributed Decision Engine, DDE) szállítja igény szerint. A DDE az Elosztott Hálózati Információ-Szolgáltatási Alrendszertől és a Hálózati Információ Szervertől kapja a különféle hálózati paramétereket. Ezeket az alrendszereket a későbbi fejezetekben részletesen is bemutatom.



1. ábra A javasolt keretrendszer általános architektúrája

A készüléken futó mHealth alkalmazások egy krízisközpontba/kórházba vagy egy egészségügyi felhőszolgáltatásba továbbítják a mért adatokat az adott mHealth alkalmazás forgatókönyve alapján, ahol a kapott adatokat feldolgozzák, statisztikákat készítenek, megjelenítik és visszajelzéseket küldenek a felhasználók számára, esetleg beavatkozást inicializálnak. Ez utóbbi műveletcsoport az egyszerű információ megjelenítéstől és figyelmeztetések küldésétől a távoli és automatikus gyógyszeradagolásig terjedhet.

A keretrendszer egy másik fontos aspektusa a média folyamok optimalizálásának lehetősége. Ahhoz, hogy mindig a megfelelő minőségű média folyamot küldhessük a kórház/mobil terminál számára, figyelembe kell venni az aktuális hálózati kontextust, és az elérhető erőforrások függvényében kell meghatározni az átvitt média paramétereit. A küldés során változtathatjuk média kódolására használt kodekeket, azok paramétereit, választhatunk az elérhető média reprezentációk közül. Ezen a területen napjainkban az egyik elterjedt eljárást az Scalable Video Coding (SVC) szabvány biztosítja, mely skálázható, erőforrás hatékony megoldást nyújt a média folyamok optimalizálására [23]. Az SVC rétegezett kódolási technikát valósít meg. Ennek segítségével valós időben, a hálózati körülményeket figyelembe véve választhatjuk ki, hogy melyik rétegek kerüljenek továbbításra [24]. A multimédia folyamok optimalizálásának másik elterjedt megoldását az MPEG's Dynamic Adaptive Streaming over HTTP (MPEG-DASH) nyújtja [25]. Az MPEG-DASH koncepció alapja, hogy eltárolják a média tartalmak különböző reprezentációját, melyek közül a DASH kliensbe implementált algoritmus választja ki és kéri le a HTTP szervertől az aktuális környezet számára optimálisat [26].

Az SVC és DASH szabványokat figyelembe véve a javasolt keretrendszerbe egy megfelelő támogató architektúrát terveztem és implementáltam. A folyamatos hálózat monitorozási eredmények alapján képes változtatni a rendszer a média folyam paramétereit, ezáltal optimális átvitelt elérve.

Ez az általános koncepció megteremti a lehetőséget, hogy az Android alkalmazások (az mHealth területre fókuszálva) optimálisan kihasználhassák az környezetükben elérhető heterogén hozzáférési hálózatokat, ezáltal biztosítva számukra a folytonos, hatékony és nem utolsó sorban akár diagnosztikai minőséget képviselő szolgáltatás nyújtást és adaptív médiaátvitelt. A keretrendszer moduláris felépítésének köszönhetően a 3rd party mHelath alkalmazások implementálása és konfigurálása, a rendszer új szenzorokkal történő

kiegészítése gyorssá és hatékonnyá válik. Az architektúra egyes elemeinek részleteit a következő alfejezetek tárgyalják.

3.2. A KERETRENDSZER ELEMEINEK RÉSZLETEZÉSE

3.2.1. ANDROID-ALAPÚ FELHASZNÁLÓI TERMINÁL

A mobil egészségügyi rendszerek központi elemét képezik a napjainkban egyre nagyobb hardveres teljesítménnyel rendelkező okostelefonok/táblagépek és az azokhoz csatlakoztatható vagy beléjük integrált egészségügyi (esetleg általánosabb célú) szenzorok. Az általam tervezett rendszerben a mobil készülék szerepét Android-alapú okostelefonok töltik be. Ennek egyik legfőbb oka, hogy a TDK dolgozatom készítésekor az Android a legelterjedtebb mobil platform az okostelefonok és táblagépek piacán. Másrészről az Android nyílt forráskódjának, valamint a gazdag API készletének köszönhetően nem csak alkalmazásokat írhatunk a készülékekre, de testreszabhatjuk, módosíthatjuk a teljes platformot is, aminek a javasolt keretrendszerem implementációjában és így valós rendszeren történő validációjában elengedhetetlen szerepe van.

Ahhoz, hogy az mHealth alkalmazások a Jövő Internetének mindenütt elérhető mobil egészségügyi szolgáltatásaiban működve folyamatosan segíthessenek a felhasználóknak, a mobil végberendezéseken biztosítanunk kell a különböző mHealth adatfolyamok egymástól eltérő kommunikációs igényeinek kielégítését, a különböző típusú hozzáférési hálózatok kezelésén alapuló hatékony mobilitás-menedzsmentet, és az elérhető hálózati erőforrások optimális, adaptív viselkedést igénylő kihasználását. Ehhez több réteget kell módosítani/kiegészíteni az Android platform hagyományos architektúrájában, és több új modul integrálására is szükség van.

3.2.1.1 HATÉKONY MULTI-INTERFÉSZ MENEDZSMENT

Napjainkban az elérhető Android készülékek szinte kivétel nélkül több hálózati interfésszel rendelkeznek (pl. Wi-Fi, 3G vagy 4G), azonban a legújabb platform verziókban sem lehetséges ezen interfészek egyidejű használata, emiatt számos kihívás adódik ezen a területen. Az általános, több interfésszel rendelkező rendszerek működésének optimalizálása egy fontos kutatási területet jelent, számos RFC és egyéb publikáció foglalkozik vele (pl.: [27], [28], [29]).

Android platformon a hálózati interfészeket menedzselő logika igen egyszerű: ha csatlakozunk egy Wi-Fi hálózatra, a rendszer lekapcsolja a 3G/4G interfészt, ellenkező esetben csak a 3G/4G interfész aktív, a Wi-Fi interfész passzív állapotban várakozik egy új elérhető hálózathoz való kapcsolódásra. Ahhoz, hogy az mHealth alkalmazások számára elérhetővé tegyük az interfészek szimultán használatát, módosítani kell ezt a logikát. Ennek pontos folyamatára az implementációs részleteket tárgyaló fejezetekben térek majd ki.

3.2.1.2 FEJLETT MOBILITÁS-KEZELÉSI PROTOKOLL

Nem csak az interfész-menedzsment, de az elterjedt hálózat-kezelő mechanizmusok és a hagyományos TCP/IP stack sincs felkészítve több interfész egyidejű használatára. Emiatt új eljárások implementálása szükséges. A Mobil IPv6 (MIPv6) protokoll család [30], a hozzá tartozó kiegészítésekkel együtt (pl. Multiple Care-of-Addresses Registration [31], Flow Bindgins [32] és Traffic Selectors for Flow Bindgins [33]) megoldást jelent erre a problémára. A MIPv6 családtagok segítségével folyam szintű mobilitás-kezelést valósíthatunk meg. Ez azt jelenti, hogy az egyes alkalmazásokhoz különböző hálózati folyamokat definiálhatunk, melyeket megadott interfészen, eltérő szabályok szerint kezelhetünk. A folyam szintű mobilitás-kezelésnek köszönhetően a megfelelő interfészeket rendelhetjük az alkalmazásokhoz, azok QoS és QoE gényeit figyelembe véve. Az alkalmazások és hálózati interfészek optimális összerendeléséhez szükséges egy olyan entitás a rendszerben, mely részletesen feltérképezi a futó alkalmazások QoS és QoE igényeit, továbbá folyamatosan monitorozza az elérhető hozzáférési hálózatok különböző paramétereit.

3.2.1.3 ALKALMAZÁS ÉS HÁLÓZATI KONTEXTUS FELDERÍTÉS

Az Alkalmazás- és Hálózati Kontextus Felderítés (Network and Application Context Discovery - NACD) komponens a széleskörű környezet monitorozásért felelős az általam javasolt keretrendszerben.

A mobil készülék aktuális pozíciójában elérhető különböző típusú hozzáférési hálózatok részletes feltérképezése elengedhetetlen az mHealth alkalmazások optimális működéséhez (hálózati-kontextus felderítés). A hálózatok átfogó vizsgálatához a TCP/IP stack különböző rétegeiből származó információkat használok fel. Ezeket a paramétereket különböző entitások biztosítják. Egyfelől maga a mobil felhasználói készülék képes a beépített hálózat-menedzsment API-kon keresztül az adatkapcsolati és fizikai rétegből származó paraméterek

lekérdezésére. Ha az okostelefon csatlakozik egy adott hálózathoz, akkor hálózati rétegből származó információkhoz is hozzáférhetünk. Ebben az esetben a mérési folyamatokhoz, és a megfelelő hálózat kiválasztásához szükséges erőforrások a mobil készüléket terhelik. Másrészről a javasolt keretrendszer lehetőséget biztosít arra, hogy különböző hálózati entitásoktól kérdezzük le a szükséges paramétereket, így spórolva a mobil végberendezés erőforrásain, és olyan információkat beszerezését is biztosítva, melyekre egyébként nem lenne lehetőségünk. Ezt a célt szolgálja az Elosztott Döntéstámogatási Motor (Distributed Decision Engine - DDE), melyet finn szerzői a [34] cikkben publikáltak, én pedig beemeltem a javasolt mHealth szolgáltatási keretrendszerbe (pontos működését a későbbi fejezetekben részletezem).

Az elérhető hálózatok paraméterein kívül figyelembe kell vennünk az mHealth szolgáltatások QoS és QoE igényeit, aktuális forgalmi jellemzőit, felhasználói sajátosságokat, stb. (alkalmazás-kontextus felderítés). Ezeket a paramétereket az alkalmazási réteg biztosítja, és a javasolt keretrendszerben elsősorban a készülékhez csatlakoztatott vagy beépített szenzorok erőforrás igényei alapján határozzuk meg. Ehhez szükség van egy olyan modulra a rendszerben, mely kezeli a készülékhez tartozó szenzorokat és részletes információkat szolgáltat azokról és az általuk forgalmazott adatfolyamokról. Az előbbi feladatot hivatott ellátni a Szenzor Adat Aggregációs Modul (Sensor Data Aggregator Module – SDAM). Beépített szenzor esetén az Android platform beépített API-jait használja a paraméterek lekérdezésére. A készülékhez Bluetooth, ANT+ vagy egyéb vezeték nélküli technológia használatával csatlakoztatott viselhető, a test különböző életjeleit és folyamatait monitorozó szenzorok esetében általában gyártó specifikus API áll rendelkezésünkre a kívánt adatok kinyeréséhez.

A hálózati információk, valamint az mHealth alkalmazások igényei alapján a Döntési és Triggerelési Alrendszer (Decision and Triggering Engine - DTE) nevű komponens dönt a megfelelő interfész kiválasztásáról, a média tartalom optimalizáslásáról, továbbá kiválasztja és konfigurálja az aktuálisan használni kívánt szenzorokat. Ennek a modulnak az általános működését a következő alfejezet tárgyalja.

3.2.1.4 DÖNTÉSI ÉS TRIGGERELÉSI ALRENDSZER

A NACD komponens által biztosított adatok alapján ez a modul konfigurálja és adaptálja a média folyamokat, választja ki és konfigurálja be az adott alkalmazáshoz/szolgáltatáshoz

használandó szenzorokat, valamint döntést hoz a hálózat váltást/inicializálást érintő kérdésekben, és triggelreli a MIPv6 mobilitás-kezelési folyamatait. A triggerelés hatására a MIPv6 létrehozza vagy módosítja az alkalmazáshoz tartozó folyamokat és a döntés alapján a megfelelő interfészhez rendeli őket. A folyamatos monitorozásnak köszönhetően a rendszer mindig alkalmazkodik az aktuális hálózati körülményekhez. Az elérhető hálózati erőforrások függvényében képes adaptívan szabályozni a média stream olyan meghatározó paramétereit, mint az alkalmazott kodek, vagy a stream felbontása. Az adaptív média streamnek (pl. MPEG-DASH [25]) köszönhetően az alkalmazás mindig az adott pillanatban elérhető optimális konfigurációjú média tartalmat tudja továbbítani a megfelelő hálózati entitások számára.

3.2.1.5 ÁLTALÁNOS mHEALTH ALKALMAZÁS

Az Android készülék architektúrájának tetején az alkalmazási réteg foglal helyet. A javasolt keretrendszerben ez egy általános mHealth alkalmazás, mely különféle szolgáltatásokat nyújthat. Külső és beépített szenzorok által szolgáltatott adatokat továbbít a kórházba/speciális egészségügyi központba vagy egy mobil egészségügyi felhőszolgáltatásba a mHealth szolgáltatás forgatókönyvétől függően. Az alkamazás megjeleníti a szenzoroktól származó adatokat, és a kórháztól kapott értesítéseket. Az értesítéseket továbbíthatja kijelzővel rendelkező, viselhető szenzorokra (pl. szívritmus monitorral felszerelt okosórára). Az alkalmazás olyan mHealth use-case-eket valósíthat meg, mint például a valós-idejű, szenzor alapú beteg távfelügyelet, telekonzultáció, irányított beavatkozások, de kevésbé késleltetés kritikus, nem valós idejű szolgáltatások is ide tartoznak, úgy, mint a fitnesz és egészséges életmód tanácsodást biztosító alkalmazások. A javasolt keretrendszer rétegek-közötti (cross-layer) optimalizációs sémát biztosít az alkalmazás számára, aminek segítségével egyrészt az alkalmazás több rétegből származó adatok alapján dönthet a hálózati erőforrások optimáls kihasználásáról, másrészt a mobilitás-kezelési feladatok szükségességének észlelése, triggerelése és végrehajtása különböző rétegekben történik, ami tovább növeli a hatékonyságot.

3.2.1.6 SZENZOR AGGREGÁTOR

A szenzor aggregátor (Sensor Aggregator) modul célja az okostelefonhoz csatlakoztatható, valamint a készülékbe integrált egészségügyi és esetleg általánosabb célú (pl. giroszkóp, GPS,

beépített nagy felbontású kamera, stb.) szenzorok menedzselése. A modul feladatai közé tartozik az okostelefon és a szenzorok közötti hálózati kapcsolat kezelése, a szenzorok adatainak valós-idejű, folyamatos lekérdezése, az egészségügyi szenzor adatok átvitelének optimizálása, továbbá ezen adatok megfelelő formátumú továbbítása az mHealth alkalmazásnak. Ahhoz, hogy az adatátvitel az adott mHealth forgatókönyv számára hatékony legyen, az aggregátor modul változtathatja az adatcsomagok méretét és küldési gyakoriságát az adott hálózati kontextust figyelembe véve. Az adatátvitel optimalizálás egyik lépése, hogy ha az adott mHealth alkalmazás forgatókönyve lehetővé teszi, akár több szenzor adatot küldjünk át egy csomag fejléc alatt. Ez késleltetést eredményezhet a csomagok vételekor, azonban sokkal kevesebb hálózati többlet forgalommal jár [35].

3.2.2. ELOSZTOTT DÖNTÉSTÁMOGATÁSI MOTOR

A tervezett keretrendszer hatékonyságának egyik kulcsa a mobil eszköz környezetében elérhető hozzáférési hálózatok tulajdonságainak minél pontosabb felderítése. Ez a folyamat történhet a mobil terminálban is a beépített API-k és segéd-alkalmazások/natív binárisok használatával, azonban ez erőforrás igényes feladat, mely az okostelefont/táblagépet sokszor feleslegesen terhelheti (ugyanakkor az így megszerzett az információ azonnal rendelkezésre áll, és pontos képet ad az eszköz környezetéről). Ezeket az információkat szolgáltathatja azonban a hálózat is, és ennek az információ-szolgáltatási és feldolgozási folyamat-rendszernek az irányítója az Elosztott Döntéstámogatási Motor, melyet angolul Distributed Decision Engine-nek (DDE) hívunk.

A DDE motort finn kollégák tervezték [34]. A DDE alapvetően az előfizető-feliratkozó tervezési mintára (Publish-Subscribe Design Pattern) épül. A DDE Consumer modulon keresztül feliratkozhatunk a számunkra releváns DDE eseményekre. A DDE Producer modulon keresztül mi regisztrálhatunk be különféle eseményeket a rendszerbe, melyre mások feliratkozhatnak. Az események tárolásáért és karbantartásáért az Event Cache modul a felelős. A finn partnerek által implementált DDE-t az mHealth szolgáltatási keretrendszerbe integrálva a mobil alkalmazást hordozó végberendezés mentesül az erőforrásigényes mérési folyamatok alól, továbbá olyan speciális, a hálózati operátorok által biztosított paraméterekhez is hozzájuthat, melyeket a mobil eszközön nem is tudnánk mérni, mégis meghatározóak lehetnek (pl. a hálózat használati költsége, linkek terheltségi szintje, stb.). Fontos, hogy a mobil terminál abban az esetben tud egy adott link hálózati szintű

paramétereiről is adatokat gyűjteni (pl. csomagvesztés, késleltetés stb.) ha csatlakozva van az adott hálózathoz. A DDE segítségével ezekhez az információkhoz könnyen, akár a kérdéses hálózattól eltérő hálózatra csatlakozva is hozzájuthatunk, így még gyorsabban hozhatunk optimális döntést. A DDE a javasolt keretrendszerben két különböző forrásból nyerheti az információkat, melyeket a alábbi két alfejezetben részletezek.

3.2.3. ELOSZTOTT HÁLÓZATI INFORMÁCIÓ-SZOLGÁLTATÁSI ALRENDSZER

Az Elosztott Hálózati Információ-Szolgáltatási Alrendszer (Distributed Network Information Provisioning - DNIP) alrendszer a WLAN hálózatokban szolgáltat dinamikus információkat a linkek és hozzáférési pontok állapotáról (pl. aktuálisan elérhető sávszélesség, link kihasználtság, jelerősség az adott pontban, csomagvesztési ráta a linken, késleltetés, felhasználók aktuális száma, stb.). Eddig ilyen adatok csak gyártóspecifikus módon voltak elérhetők, most a javasolt keretrendszerben a DDE-t alkalmazva egyszerű API-val hozzáférhetők. A DNIP a DDE Producer alrendszeren keresztül elküldi a mért adatokat a DDE-nek. Erre az eseményre az Android készülék a DDE Consumer modult használva feliratkozhat, ezáltal folyamatos hálózati információ nyújtást biztosítva a készüléknek.

3.2.4. HÁLÓZATI INFORMÁCIÓS SZERVER

A Hálózati Információs Szerver (Network Information Server - NIS) statikus információkat bocsát a készülék rendelkezésére a különböző típusú hozzáférési hálózatokról (pl. az adott használat hálózati költsége, a rádiós cella/WLAN hozzáférési pont pontos helyzete, lefedettségi területének alakja és nagysága stb.). Mind a mobil eszköz, mind pedig az előbbi két modul a DDE Producer és DDE Consumer komponenst használja a DDE-vel való kommunikációra. A DDE Producer modul segítségével különböző adatokat küldhetünk a DDE számára (pl. a mobil készülék az aktuális geografikus pozícióját), illetve a NIS és a DNIP alrendszerek is ezt használják az adatok DDE rendszerbe történő feltöltésére. A DDE Consumer a DDE-ből érkező adatok fogadását menedzseli.

3.2.5. A KÓRHÁZ/KRÍZISKÖZPONT HÁLÓZATA

A mHealth forgatókönyvek a nagy részében (pl. valós idejű távfelügyelet, irányított beavatkozás, telekonzultáció) fontos szerepe van a kórháznak vagy egyéb speciális egészségügyi- és krízisközpontnak. A mobil készülék ide továbbítja a páciens által viselt

szenzorok adatait, ahol feldolgozzák az eredményeket (sok esetben valós időben), majd azok alapján utasításokat adhatnak a betegek számára. A javasolt keretrendszerben a kórház, mint alrendszer is több modulból áll. Képes feldolgozni, megjeleníteni a szenzorok adatait, lejátszani és optimalizálni a mobil készülék által továbbított média tartalmakat, továbbá szükség esetén értesítéseket küld a felhasználónak a feldolgozott eredmények tükrében.

3.2.6. mHEALTH FELHŐSZOLGÁTATÁSOK

A hatékony mobilitás-kezelési architektúra tervezésekor elsősorban a valós-idejű, szenzor alapú, gyors beavatkozást igénylő mHealth szolgáltatások forgatókönyveihez tartozó rendszer megteremtését tartottam szem előtt. Azonban a javasolt keretrendszer támogatja a mHealth szolgáltatások széles tárházát, felhő alapú szolgáltatási architektúrákon keresztül. A felhasználó feltöltheti a szenzorokól származó vagy egyéb adatokat az mHealth felhőszolgáltatásba. Az előzőekhez hasonlóan ez az alrendszer is képes feldolgozni és megjeleníteni a beküldött adatokat, statisztikákat készíteni és visszajelzéseket küldeni a felhasználók számára.

3.2.7. HOME AGENT

Mobil IPv6 szabvány szerint minden mobil eszköznek (Mobile Node – MN) van egy otthoni ügynöke (Home Agent – HA). A HA hálózati entitás egy állandó, a MN globális elérhetőségét biztosító címet, az ún. otthoni címet (Home Address – HoA) rendeli a hozzá tartozó mobil terminálokhoz. A mobil eszköz a HoA címen kívül mindig kap egy ideiglenes címet is abból a hálózatból, amelyben aktuálisan tartózkodik. Ezt a címet nevezzük ideiglenes címnek, vagyis Care-of-Addressnek (CoA). Az otthoni ügynök feladata a CoA és a HoA közötti kötés folyamatos menedzselése. A HA a periodikus és eseményvezérelt kötési üzeneteknek (Binding Update – BU) köszönhetően, mindig tudja, hogy éppen hol tartózkodik a mobil eszköz, így el tudja juttatni hozzá címezett adatcsomagokat [30]. A Mobil IPv6 szabványnak számos olyan kiegészítése van mely lehetővé teszi a folyam alapú mobilitás-kezelési eljárások tervezését és implementálását. Ezek közül kiemelném a Multiple Care-of Addresses Registration [31] (MCoA), Flow Bindings [32] és Traffic Selectors for Flow Bindings [33] szabványokat. Az MCoA lehetővé teszi, hogy egyszerre több ideiglenes címet rendeljünk a mobil terminálhoz, ezáltal biztosítva akár több interfészes rendszerek hatékony működését. Ezeket az ideiglenes címeket a Home Agent menedzseli. A Flow Bindings szabvány

segítségével különböző hálózati folyamokat definiálhatunk az alkalmazásokhoz, és a folyamokat különböző interfészekhez köthetjük. Egy általános hálózati folyamot öt paraméterrel írhatunk le: cél cím/port, forrás cím/port és a folyamhoz tartozó protokoll.

3.2.8. MEDIA PLAYOUT OPTIMALIZÁLÁS

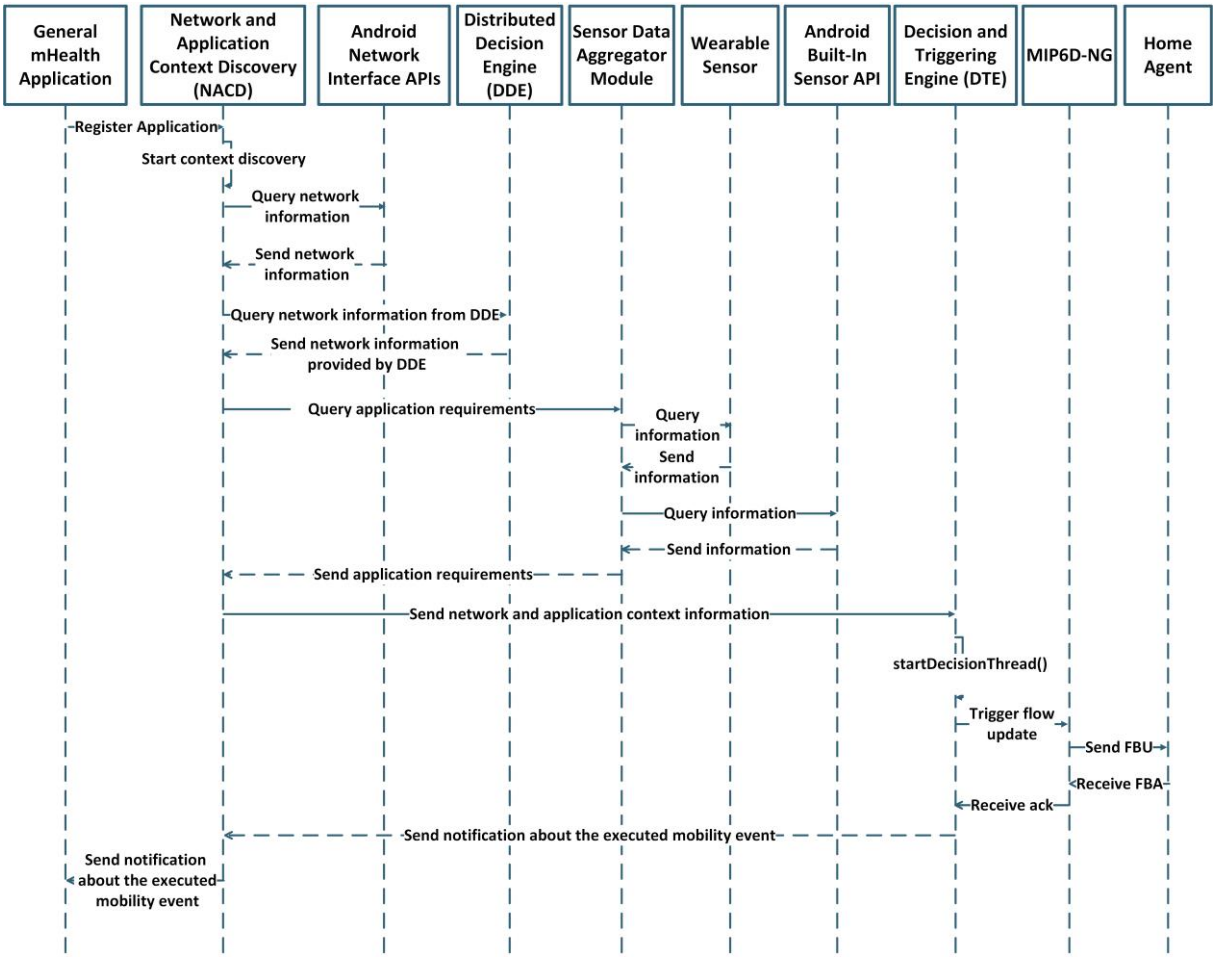
A javasolt keretrendszerben a média playout optimalizálás alapját az MPEG-DASH rendszerek képezik. A bemutatott architektúrában az általános mHealth alkalmazás tartalmaz egy Media Playout (MP) nevű alrendszert, mely egy DASH lejátszót reprezentál, mely MPEG-DASH tartalmak megjelenítéséért, és az elérhető média reprezentációk lekérdezéséért felelős. A reprezentációkat a DASH szervertől tudja lekérdezni HTTP alapú „request-response" kommunikációs sémát használva [26]. Ez a modul a DTE-ben található média playout optimalizálásáért felelős ún. Media Playout Optimalization (MPO) alrendszerrel kommunikál. Az MPO lekérdezi az alkalmazás MP moduljától az elérhető reprezentációkat, továbbá a DTE rendelkezik a jelenleg elérhető részletes hálózati és alkalmazási kontextus adataival. Ezek alapján a döntési algoritmus ki tudja választani az optimális média reprezentációt az elérhetőek közül. A keretrendszer azonnal értesül a mobilitási eseményekről, környezeti változásokról, így valós időben, adaptívan tudja szabályozni az átvitt média tartalmak minőségét, hatékonnyá téve ezáltal az átvitelt.

3.3. CROSS-LAYER KOMMUNIKÁCIÓS ÉS DÖNTÉSI SÉMA

3.3.1. CROSS-LAYER MECHANIZMUS ÁTTEKINTÉSE

A javasolt keretrendszer rétegek-közötti (cross-layer) optimalizációs sémát biztosít az mHealth alkalmazások számára, így lehetővé válik, hogy az architektúra különböző rétegei által biztosított információk alapján optimális döntést hozzon az algoritmus a hálózati erőforrások kihasználásáról és az orvosi multimédia-átvitel egyéb részleteiről. Ezen felül a cross-layer kommunikációs sémát alkalmazva a mobilitás-kezelési fontosabb metódusai (pl.: inizializálás, triggerelés, végrehatjás) különböző rétegekben történhet. A javasolt keretrendszer ennek segítségével olyan döntést tud hozni, mely a mobil terminál adott alkalmazási és hálózati kontextusát figyelembe véve optimális. A javasolt keretrendszerben 4 fő komponens vesz részt a cross-layer kommunikációban.

A NACD modul, a DTE modul és az mHealth alkalmazás közötti kommunikáció eseményvezérelt architektúrán alapszik, előfizető-feliratkozó (publish-subscribe) tervezési mintára épülve. Az egyes modulok eseményeket definiálhatnak, melyekre mások feliratkozhatnak. A DTE és a mobilitás-kezelési modul (MIP6D-NG) TCP alapú socketen keresztül képesek kommunikálni egymással. Amint egy mHealth alkalmazás elindul, értesíti erről a NACD modult. A NACD modul részletesen feltérképezi az aktuális alkalmazási és hálózati kontextust. Ez a modul továbbítja a szenzorokból érkező adatokat az alkalmazás számára.



2. ábra A javasolt keretrendszer komponenseinek kommunikációs diagramja

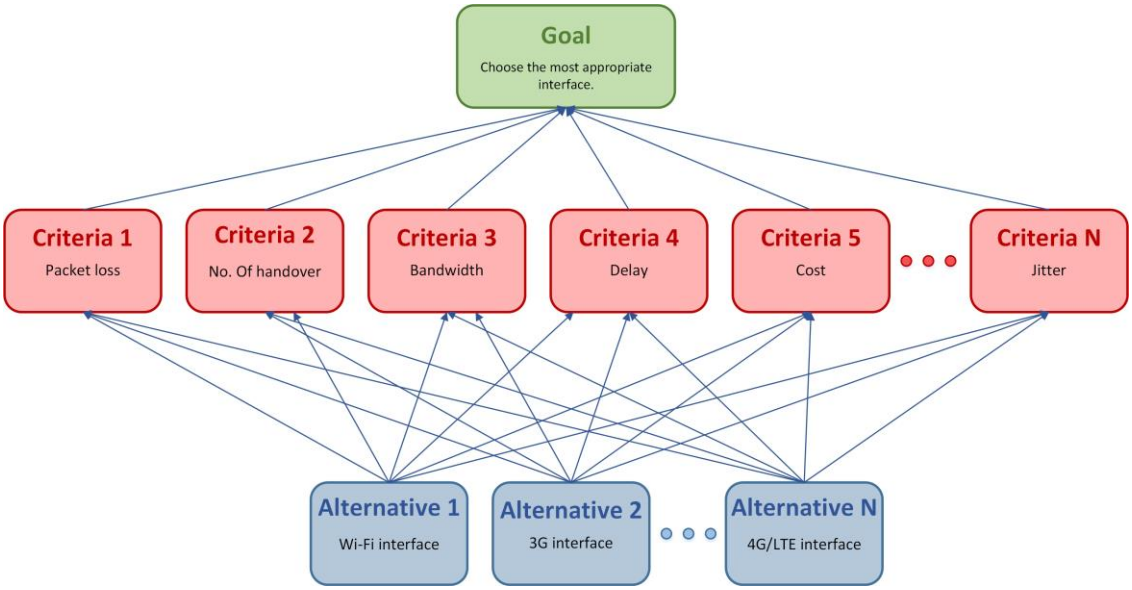
A mért/kapott kontextus adatokat pedig elküldi a DTE modulnak, mely ezek alapján döntést hoz a szükséges mobilitás-kezelési feladatról. Ha van elvégzendő mobilitás-kezelési feladat, a DTE socketen keresztül elküldi a megfelelő üzenetet a mobilitás-kezelési modulnak, aki nyugát küld vissza a DTE modulnak a végrehajtott feladat sikerességéről. A cross-layer

kommunikációval tehát elkülönül a monitorozás, döntés, triggerelés és végrehajtás metódusa.

Az egyes mondulok a javasolt keretrendszerben definiált kommunikációs és együttműködési séma alapján vannak integrálva. A komponensek közti interakciók és kommunikációs folyamatok általános működését az 2. ábra mutatja be.

3.3.2. A DÖNTÉSI ALGORITMUS ÁTTEKINTÉSE

A javasolt döntési algoritmus formalizálásához az Analytical Hierachy Process (AHP) technikáját választottam. Az AHP [36] széles körben használt módszere a több paramétert számításba vevő döntési rendszereknél, melyekre MCDM (Multiple-criteria Decision Making), vagy gyakran MCDA (Multiple-criteria Decision Analysis) rendszerként hivatkozunk. Az AHP egy jól struktúrált analízis technika, olyan komplex döntési algoritmusok számára, melyek számos döntési paramétert vesznek figyelembe az optimális döntés meghozatala során.



3. ábra A javasolt döntési algoritmus AHP reprezentációja

Az algoritmus első lépése az döntési probléma dekompozíciója, melynek során hierarchikus strutúrába képezzük le az eredeti döntési problémát. A hierarchia három mélységű. A legalsó szinten ábrázolhatjuk a lehetséges döntési alternatívákat. A második szinthez a döntés során figyelembe vehető kritériumok tartoznak. Végül a legfelső szint az elérni kívánt célt reprezentálja. A 3. ábra a döntési algoritmusom AHP struktúra megfeleltetését ábrázolja. A felső szint a cél, vagyis megtalálni az adott mHealth alkalmazáshoz tartozó hálózati folyam

számára a legmegfelelőbb interfészt az elérhetőek közül. A második szint a döntés során figyelembe vett kritériumokat tartalmazza.

Ahhoz, hogy megtaláljuk a megfelelő interfészt és biztosítsuk az alkalamzások QoS és QoE igényeit, figyelembe kell venni az adott linken elérhető sávszélességet, késleltetést, jitter, csomagvesztési rátát, a hálózati link használatának a költségét valamint, hogy az adott folyam mennyire tolerálja a frekventált hívásátadást. Természetesen a kritériumok listája tovább bővíthető, finomítva ezzel a döntési algoritmust. A legalsó szinten találjuk a lehetséges alternatívákat, tehát jelen esetben az igénybe vehető hálózati interfészek listáját.

A felépített struktúra alapján az AHP három lépésben meghatározza az eredményt:

- Minden kritérium párhoz kiszámolja a páronkénti összehasonlítási mátrixokat (pairwise comparision matrices) és a kritériumokhoz tartozó súlyvektorokat.
- Kiszámolja az alternatívákhoz tartozó összehasonlítási mátrixokat (matrix of alternative option scores).
- Sorrendezi az alternatívákat.

Az első lépés tehát a páronkénti összehasonlítási mátrixok meghatározása. Ez a mátrix azt mutatja, hogy egy adott kritérium mennyivel relevánsabb egy másik kritériumnál egy meghatározott hálózati folyam esetében. A mátrix minden elemére a következő kifejezésnek teljesülnie kell:

$$(1) \qquad a_{ij} \cdot a_{ji} = 1$$

Ebből a mátrixból a következő képlet segítségével el tudjuk készíteni a normalizált összehasonlítási mátrixokat:

$$(2) \quad \overline{a}_{ij} = \frac{a_{ij}}{\sum_{k=1}^{m} a_{kj}}$$

A mátrix oszlopainak számtani átlagolásával megkapjuk a mátrixokhoz tartozó, a kritérium súlyozását jelző vektorokat.

a_{ij} *értéke*	*interpretáció*
1	i és j kritériumok egyformán
3	i kis mértékben fontosabb j
5	i kritérium fontosabb j
7	i sokkal fontosabb j

Példaként elkészítettem három különböző alkalmazás folyamhoz tartozó páronkénti összehasonlítási mátrixot. A példa két videó folyam (magas felbontású és alacsony felbontású video stream) és egy egészségügyi szenzor adat továbbításához használt folyam mátrixát mutatja be. Ahhoz, hogy számszerűen meghatározzuk, hogy egy kritérium mennyivel fontosabb egy másiknál, szükség van valamilyen interpretációra, mely a fontosság mértékét egy számmal fejezi ki. Az irodalomban gyakran az alábbi (vagy ahhoz nagyon hasonló) interpretációt alkalmazzák. Ez alapján a példa folyamokhoz előállított páronkénti összehasonlítási mátrixokat a 4. ábra mutatja. A mátrixban szereplő kritériumok rendre: a=csomagvesztési ráta, b=késleltetés, c=hívásátadások száma, d=költség, e=sávszélesség, f=jitter.

$$M_1 = \begin{bmatrix} & a & b & c & d & e & f \\ a & 1 & 1/5 & 5 & 5 & 1/7 & 1/5 \\ b & 5 & 1 & 1/3 & 5 & 1/7 & 1/5 \\ c & 1/5 & 3 & 1 & 5 & 1/7 & 1/3 \\ d & 1/5 & 1/5 & 1/5 & 1 & 1/7 & 1/5 \\ e & 7 & 7 & 7 & 7 & 1 & 5 \\ f & 5 & 5 & 3 & 5 & 1/5 & 1 \end{bmatrix} \quad M_2 = \begin{bmatrix} & a & b & c & d & e & f \\ a & 1 & 1/5 & 5 & 5 & 1/5 & 1/3 \\ b & 5 & 1 & 1/3 & 5 & 1/5 & 1/5 \\ c & 1/5 & 3 & 1 & 5 & 1/5 & 1/3 \\ d & 1/5 & 1/5 & 1/5 & 1 & 1/5 & 1/3 \\ e & 5 & 5 & 5 & 5 & 1 & 3 \\ f & 3 & 5 & 3 & 3 & 1/3 & 1 \end{bmatrix}$$

$$M_3 = \begin{bmatrix} & a & b & c & d & e & f \\ a & 1 & 5 & 1/3 & 7 & 1/5 & 5 \\ b & 1/5 & 1 & 1/7 & 7 & 5 & 5 \\ c & 3 & 7 & 1 & 7 & 5 & 5 \\ d & 1/7 & 1/7 & 1/7 & 1 & 1/5 & 1/5 \\ e & 5 & 1/5 & 1/5 & 5 & 1 & 1/3 \\ f & 1/5 & 1/3 & 1/5 & 5 & 3 & 1 \end{bmatrix}$$

4. ábra Kritériumok páronkénti összehasonlítási mátrixai (magas felbontású videó stream [M_1], alacsony felbontású videó stream [M_3], egészségügyi szenzor adat folyama [M_2])

A kritériumokhoz tartozó súlyvektorok az alábbi képlettel számíthatók ki:

$$(3)\ w_i = \frac{\sum_{k=1}^{m} \overline{a}_{ik}}{m}$$

Második lépésben az AHP algoritmusa meghatározza a az alternatívákhoz tartozó páronkénti összehasonlítási mátrixokat. Ezekre a mátrixokra is az (1) és (2) képlet teljesül, illetve az 1. táblázat szerinti reprezentációt használjuk a relevánsság mértékének a számszerűsítésére.

A példához tartozó mátrixokat a következő ábra szemlélteti. A példa esetében az egyszerüsítés kedvéért éltünk azzal a feltételezéssel, hogy a készülékben összesen két hálózati interfész található (Wi-Fi és 3G). A mátrixok rendre a következő kritériumokhoz tartoznak: a=csomag vesztési ráta, b=késleltetés, c=hívásátadások száma, d=költség, e=sávszélesség, f=jitter.

$$A_1 = \begin{bmatrix} a & 3G & WiFi \\ 3G & 1 & 5 \\ WiFi & 1/5 & 1 \end{bmatrix} \quad A_2 = \begin{bmatrix} b & 3G & WiFi \\ 3G & 1 & 1/5 \\ WiFi & 5 & 1 \end{bmatrix} \quad A_3 = \begin{bmatrix} c & 3G & WiFi \\ 3G & 1 & 3 \\ WiFi & 1/3 & 1 \end{bmatrix}$$

$$A_4 = \begin{bmatrix} d & 3G & WiFi \\ 3G & 1 & 3 \\ WiFi & 1/3 & 1 \end{bmatrix} \quad A_5 = \begin{bmatrix} e & 3G & WiFi \\ 3G & 1 & 1/7 \\ WiFi & 7 & 1 \end{bmatrix} \quad A_6 = \begin{bmatrix} f & 3G & WiFi \\ 3G & 1 & 3 \\ WiFi & 1/3 & 1 \end{bmatrix}$$

5. ábra Az alternatívákhoz tartozó páronkénti összehasonlítási mátrixok

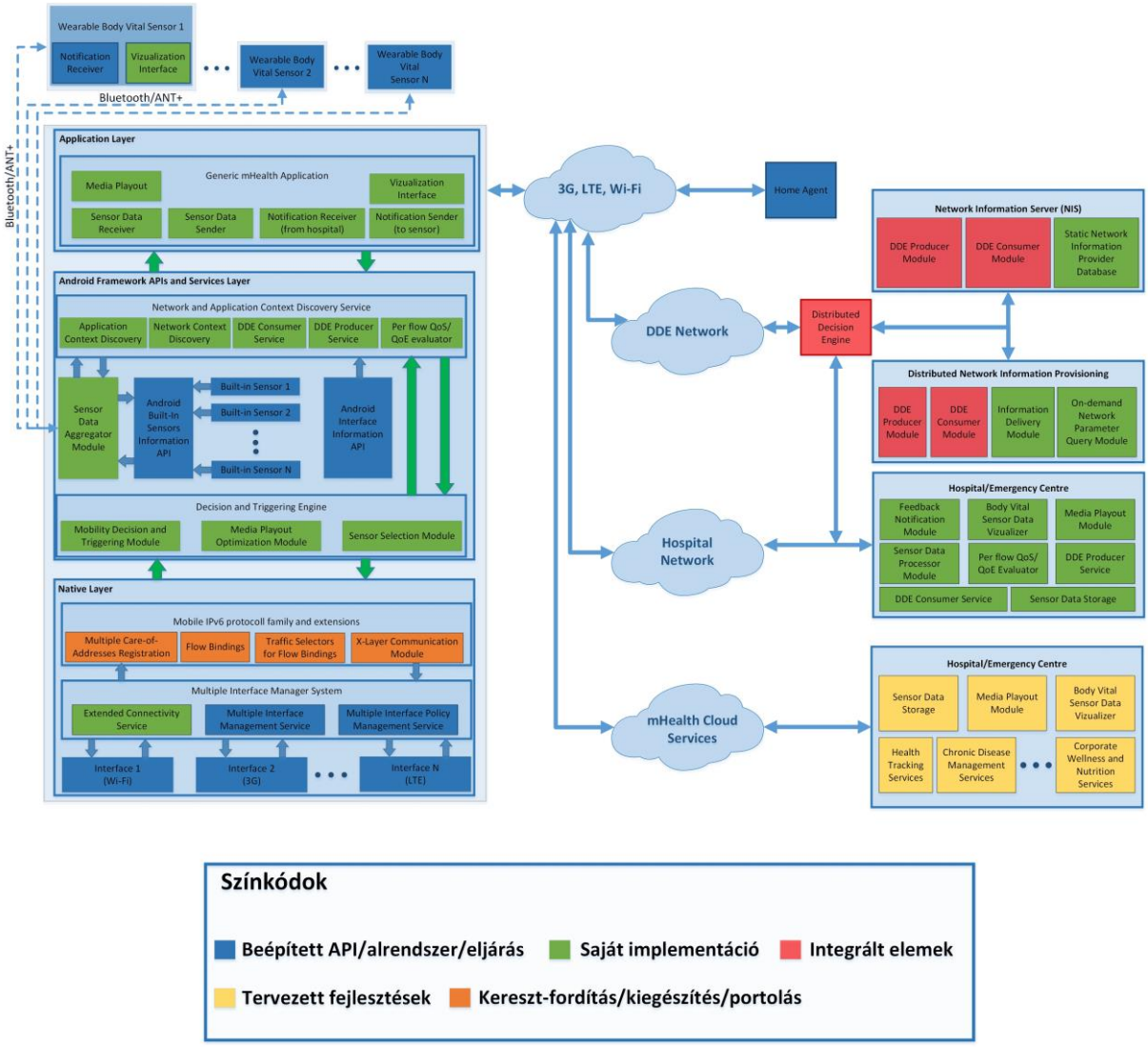
Az alternatívákhoz tartozó súlyvektrokat (**s** vektor) a mátrixok oszlopainak átlaga adja meg. Az algoritmus végső lépéseként az i. alternatívákhoz tartozó globális pontszám meghatározása a következő képlet segíségével történik:

$$(4)\ V_i = S_i \times W_i$$

A legnagyobb globális pontszámmal rendelkező alternative (vagyis rendelkezésre álló interfész) lesz az optimális döntés az adott hálózati folyam számára.

4. A JAVASOLT KERETRENDSZER IMPLEMENTÁCIÓS RÉSZLETEI

Ebben a fejezetben a javasolt keretrendszer elemeinek részleteit mutatom be. Ismertetem a fontosabb implementációs megfontolásokat, az általános Android architektúra módosításához szükséges eljárásokat és magukat a módosításokat, továbbá a külső rendszerek integrálásához általam tervezett mechanizmusokat. A koncepció nagy része saját fejlesztésű komponenseket foglal magában, de ezeken kívül több, az irodalomban már létező elemet és modult integrál.



6. ábra A javasolt keretrendszer komponenseinek fejlesztési státusza

Az egyes komponensek a javasolt keretrendszerben definiált kommunikációs és együttműködési séma alapján dolgoznak össze, és együttesen egy elosztott, jól skálázható, a felsőbb rétegbeli alkalmazások számára optimalizált hálózati erőforrás-kihasználást biztosító egységes architektúrát alkotnak. A 6. ábra az egyes komponenseknek a javasolt

keretrendszerben betöltött fejlesztési státuszát (saját fejlesztés, integrált elem, beépített komponens, tervezett fejlesztés) mutatja be.

4.1. AZ ANDROID-ALAPÚ FELHASZNÁLÓI VÉGBERENDEZÉS IMPLEMENTÁCIÓJA

Az architektúra központi eleme a nagy mértékben módosított Android platformot használó okostelefon. Ez az entitás felelős az mHealth alkalmazások futtatásáért, az alkalmazási és hálózati kontextus részletes feltérképzéséért a készülék által mért, illetve a DDE-től kapott méréseket felhasználva (NACD modul). A NACD által kapott adatok alapján a DTE modul döntést hoz és triggeleri a különböző mobilitási eseményeket, optimalizálja a média folyamokat, kiválasztja a használni kívánt szenzorokat. A kiváltott trigger események alapján a MIPv6 szabványra épülő, több hálózati interfész szimultán használatát lehetővé tévő natív modulok pedig elvégzik a mobilitás-kezelési feladatokat.

Az okostelefonhoz különféle egészségügyi szenzorok csatlakozhatnak vezeték nélküli technológiák (pl.: Bluetooth, ANT+ stb.) segítségével, de napjainban egyre eltrejedtebbek a készülékbe integrált egészségügyi és egyéb általános célú szenzorok is. Ezen szenzorok adatainak segítségével implementálhatóvá válnak az mHealth terület különböző forgatókönyeit támogató alkalmazások. A következő alfejezetekben a javasolt keretrendszer Andorid-alapú felhasználói végberendezésének alrendszereit mutatom be, melyek lehetővé teszik a fentebb leírt funkciók megvalósítását. A komponensek bemutatása során az architektúrában lentről felfelé (“bottom-up” megközeletítés) haladok.

4.1.1. MULTI-INTERFÉSZ MENEDZSMENT

Napjainkban az okostelefonok/táblagépek túlnyomó többsége egynél több fizikai (pl. Wi-Fi, 3G, 4G/LTE) vagy virtuális (pl. Wi-Fi Direct) hálózati interfésszel (Multiple Interface – MIF) vannak felvértezve. Ezeken az interfészeken keresztül a mobil készülék képes csatlakozni a különböző típusú hozzáférési hálózatokhoz. A heterogén, átlapolódó vezeték nélküli hálózatok szimultán használata megteremti a lehetőséget az újgerenációs alkalmazások minőségének a növelésére azáltal, hogy képesek kihasználni a környezetükben elérhető, eltérő rádiós technológiák által biztosított hálózati erőforrásokat.

Azonban jelenleg a különféle hálózati eszközökben alkalmazott hálózatkezelési mechanizmusok, mobilitás-kezelési eljárások nem támogatják közvetlenül a több interfész egyidejű használatát. Ezen a területen számos kihívás adódik emiatt, a multi-interfész rendszerek kezelése egy fontos kutatási terület napjainkban, számos RFC draft foglalkozik a MIF rendszerek problématárával, követelményeivel, megoldási lehetőségeivel. Ez a problémakör még a legújabb Android platformok esetében sem megoldott, az Android alapú eszközök nem tudják egyszerre használni a különböző típusú hálózati interfészeiket. A javasolt keretrendszer hatékony működése érdekében módosítottam az Android hálózatkezelési mechanizmusát, hogy lehetőségem legyen az interfészek egyidejű használatára. A keretrendszerben a Multi-interfész Menedzser (Multi Interface Manager System) jelent megoldást a problémára. Egy általános MIF rendszerben számos kérdést kell megoldani a hatékony működés érdekében:

- A rendszer milyen módon rendelje össze a létező fizikai interfészeket és az elérhető hozzáférési hálózatokat
- A heterogén hálózatoktól kapott eltérő konfigurációs információkat hogyan egyeztesse össze, anélkül hogy azok konfliktusba kerülnének egymással
- Fontos kérdés a hálózatokhoz tartozó policy beállítások hatékony menedzsmentje
- A felsőbb rétegek milyen mechanizmusokkal kezeljék a szimultán érkező hálózati kapcsolatokat

Az Android platform esetében elsősorban erőforrás takarékossági szempontok miatt nem engedélyezett a hálózati interfészek egyidejű használata, de szerencsére a natív, Linux alapú alrendszerekek fel vannak készítve a multi-interfész kezelésére lehetőséget adó protokoll szintű mechanizmusokra. Emiatt nekem a hálózatkezelési mechanizmus implementációját kellett módosítanom, melyet az ábrán Kiterjesztett Csatlakozási Menedzsernek (Extended Connectivity Service – ECS) nevezetem el. Az alrendszerhez tartozó másik két modul, a Multi-interfész Menedzser (Multiple Interface Management Service) és a Multi-interfész Policy Menedzser (Multiple Interface Policy Management Service) funkcióit az Android platform natív rétegei ellátják.

A ECS hatékony működése tehát az Android magasabb szintű kapcsolatkezelési mechanizmusának a módosítását követlei meg. Ez a logika az Android *Connectivity Service* moduljában található meg. A *ConnectivityService* osztályban található a hálózati kapcsolatok

menedzselésének állapotgépe. Ezt az állapotgépet saját monitorozó függvényekkel kiegészítve részletesen feltérképeztem. Ezután kiegészítettem, hogy ha egy prioritásos hálózat már definiált a rendszerben, és csatlakozni kíván egy új hálózathoz is, akkor a rendszer ne tegye meg az új hálózatot a prioritásosnak (lekapcsolva ezáltal az eredeti hálózatot), hanem mindkettőt prioritásos hálózatnak definiálja. Ezzel a kiegészítéssel a natív rétegek már tudják kezelni a több interfész szimultán használatát, mivel ezekben a rétegekben fel van készítve az Android erre. Tehát a *ConnectivityService* általam elvégzett módosításával lehetőség nyílik több interfész egyidejű használatára. A különböző Android verziókban a fent említett hálózatkezelési állapotgép eltérő, én a fejlesztéseket kezdetben Android 4.0, Android 4.1 rendszereken kezdtem, majd frissítettem az implementáció készítésének idején elérhető legfrissebb, Android 4.4 platformra. A módosítások végrehajtása az Android forráskódjának újrafordítását igényli. A fejlesztés során az CyanogenMod[1] nyílt forráskódú disztribúcióit használtam fel.

4.1.2. MOBILE IPV6 IMPLEMENTÁCIÓ

Nem csak az interfész-menedzsment, de a jelenleg elterjedt mobilitás-kezelési eljárások és a hagyományos TCP/IP stack sincs felkészítve több interfész egyidejű használatára. Emiatt új eljárások implementálása szükséges. Én a Mobil IPv6 (MIPv6) protokoll családot és az erre épülő kiegészítéseket választottam a hatékony mobilitás-kezelés megvalósítására. A keretrendszer elkészítése során a Next Generation Mobile IPv6 for Linux (MIP6D-NG) [15] implementációt használtam fel. Ez az első mobil IPv6 implementáció (az UMIP[2] továbbfejlesztésének, új alapokra helyezésének tekinthető), mely elosztott, moduláris és olyan kiegészítéseket tartalmaz, mint a Flow Bindigns [32], Multiple Care-of Addresses Registration [31], NEMO [37] vagy HMIPv6 [38]. A MIP6D-NG-t Takács András és Bokor László tervezte és implementálta. A részletes bemutatását a [15] cikkük tárgyalja. Én az Android platformra történő kompatibilitási fejlesztéseket végeztem, továbbá a MIP6D-NG portolásához szükséges keresztfordító eszköztárat készítettem el.

A MIP6D-NG portolása nem triviális feladat, mivel számos, Linux disztribúciókon elérhető, a MIP6D-NG által igényelt könyvtár és forrásfájl hiányzik az Android rendszerekről, vagy ha

[1] CyanogenMod github: https://github.com/CyanogenMod

[2] UMIP: http://www.umip.org/

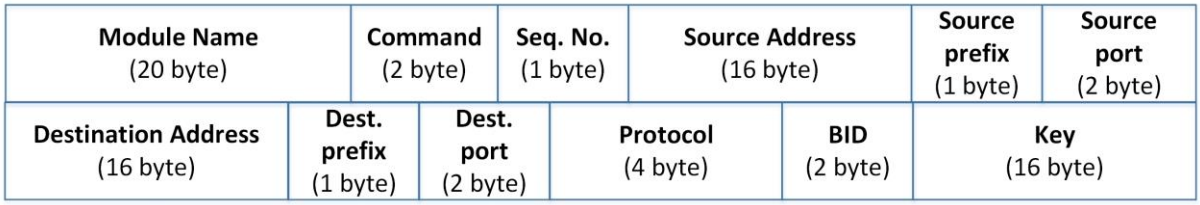
létezik is, akkor gyakran különbözik a Linux verzióktól. Emiatt egy olyan kiegészített keresztfordító eszköztárat kellett készítenem, az Android NDK eszköztárra[3] alapozva, mely tartlamazza a MIP6D-NG által megkövetelt könyvtárak Android kompatibilis verzióját. A kompatibilitáshoz szükséges fejlesztéseket, valamint a keresztfordító eszköztár részletesebb leírását a [39] és [40] cikkeinkben tárgyalom. Ahhoz, hogy a MIP6D-NG futtatható legyen Android platformon, három fő követelményt kell kielégitenie a rendszerünknek. Egyrészt biztosítani kell a több interfész szimultán használatát, továbbá minden szükséges, Linux disztribúción megtalálható könyvtárnak léteznie kell Androidon is. Ezt a két követelményt a fentebb tárgyalt módszereket alkalmazva már teljesíti a rendszerünk. A harmadik feltétel a speciális kernel futtatásának igénye. A MIP6D-NG speciális kernelt igényel, mely teljeskörű Mobile IPv6 támogatással rendelkezik és tartalmazza a MIP6D-NG kernel modulokat is. Ennek a követelmények a kielégítése végett módosítottam az Android kernel forráskódját. A forráskód alapjául az Andromadus és CyanogenMod nyílt forráskódjait használtam. Mivel az Android kernel és OS forráskód is több rétegű módosítást igényel, ezért a teljes forráskód újrafordítása szükséges feladat volt. Így egy olyan saját ROM-ot készítettem, amely lehetőséget nyújt több interfész egyidejű használatára, továbbá a mobilitás-kezelési feladatokat a MIP6D-NG látja el, a Mobile IPv6 protokoll családban definiált szabványoknak megfelelően.

4.1.3. DÖNTÉSI ÉS TRIGGERELÉSI ALRENDSZER IMPLEMENTÁCIÓJA

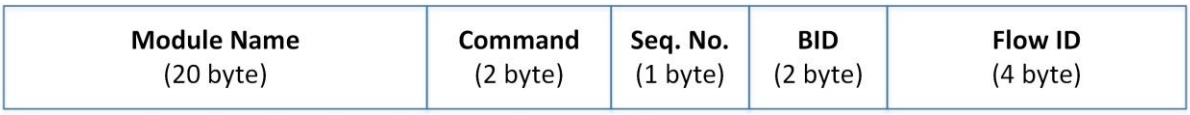
A döntési és triggerelési alrendszer három fő komponenst tartalmaz. A mobilitás-kezelés döntési és triggerelési modulja (Mobility Decision and Triggering Module – MDT) alapvetően két fő feladatot lát el. Ez a modul tartalmazza az AHP alapú döntési algoritmus implementációját. Az AHP a szükséges QoS paramétereket a NACD modultól kapja meg. Az AHP algoritmus eredménye alapján a modul triggereli a különféle mobilitás-kezelési feladatokat. Ezt a MIP6D-NG által biztosított TCP socket alapú API-jának segítségével teheti meg. Az API-n keresztül a DTE modul menedzselheti a MIP6D-NG-hez tartozó modulokat. A részletes API parancsokat és modulokat a cikk tárgyalja [15]. Ebben a fejezetben a folyam-szintű mobilitási-kezeléséhez szükséges *Flow Register* és *Flow Update* API parancsokat mutatom be részletesen. Ezen üzenetek segítségével a DTE új folyamot regisztrálhat be a

[3] Android NDK toolchain: https://developer.android.com/tools/sdk/ndk/index.html

rendszerbe, és frissítheti a folyamokat. A frissítési metódust használva a folyamokat átmozgathatjuk az elérhető hálózati interfészek között. A folyam szintű menedzseléséhez szükséges *Flow Register* és *Flow Update* cross-layer jelzési üzenetek formátumát a 7. ábra és a 8. ábra tükrözi.



7. ábra A Flow Register API parancs formátuma



8. ábra A Flow Update API parancs formátuma

A *Flow Register* parancs egy modul névvel kezdődik. Itt az API által meghívott MIP6D-NG modul nevét kell megadni, ezen üzenet esetében ez a FLOB modul. A parancs fontos eleme a hálózati folyamhoz tartozó cél- és forráscím, valamint az ezekhez tartozó prefix és port. Meg kell adni a folyam protokolljának típusát (pl. TCP, UDP, RTP stb.), továbbá a BID értéket, ami a hálózati interfész azonosítója. Így létre tudunk hozni egy meghatározott protokollra, cél- és forráscímre illesztkedő folyamot, amit a megadott hálózati interfészhez rendelhetünk. A *Flow Update* parancs esetében a frissíteni kívánt folyam egyedi azonosítóját és az interfész BID értékét kell megadni.

A DTE második modulja a Média Folyam Optimalizáló Modul (Media Playout Optimization Module – MPO). A javasolt keretrendszer ezen almodulja biztosítja a hatékony média átvitelt. Az SVC és MPEG-DASH szabványokhoz hasonlóan a DTE modul Media Playout alrendszere a hálózati kontextust információk alapján képes változtatni az átvitt média folyam paramétereit. A statikus és dinamikus hálózati információkat egyrészt a mobil készüléken végzett mérési metódusok, másrészt a DDE szolgáltatja. A DDE-től való információ kérés módja lehet periodikus vagy esemény vezérelt. Periodus lekérdezés esetén a mobil eszköz megadott intervallumokban küld kérést a DDE-nek. Esemény vezérelt koncepcióban a mobil készülék által detektált QoS/QoE változások triggerelik a lekérdezés folyamatát.

Az esemény vezérelt működést az általam tervezett és implementált folyam szintű QoS és QoE mérőrendszer, az ún. Per flow QoS/QoE Evaluator (FlowEval) valósítja meg. A FlowEval különböző hálózati méréseket hajt végre a készüléken az Android Network Interface API-jait, továbbá különféle monitorozó natív binárisokat felhasználva. Ha ezek meghatározott küszöbértékeket meghaladó változást detektálnak, a FlowEval lekérdezi a DDE-ből az aktuális hálózati kontextusokat, melynek hatására a média folyam optimalizásiója, és a szükséges mobilitás-kezelési események tirggerelése megtörténik. A FlowEval sokkal gyorsabban képes reagálni a környezeti változásokra, mint a periodikus mechanizmus, növelve ezzel a rendszer hatékonyságát.

A média optimalizálás számos mHealth forgatókönyv esetében elengedhetlen. Az optimalizálás igénye többféle irányból érkezhet. Például egy telekonzultációs rendszer esetében a küldő oldal módosítja a média folyam paramétereit, de egy irányított beavatkozás, távsegítség esetében az orvosok is küldhetnek kérést a kívánt média paraméterekről.

A jelenlegi keretrendszerben kétféle média folyam került implementálásra. Az egyik egy UDP alapú, az Android *DatagramSocket* API-ját használó szoftver komponens, melyet elsősorban QoS mérési célokra terveztem. Az MPO dinamikusan változatatja a küldendő média csomagok nagyságának és küldési gyakoriságának az értékét az aktuális hálózat NACD modul által mért paraméterei alapján. Ez az implemnetáció megkönnyíti a mérések azonos paraméterekkel történő ismétlését, egyszerű reprodukáltahóságát.

A másik média folyam implementáció az olyan mHealth forgatókönyvek számára készült, melyben fontos szerepe van a valós-idejű kamera kép továbbításának. Az implementáció során az Android LibStreaming [4] külső könyvtárat vettem alapul. Ez a könyvtár lehetőséget biztosít az Android beépített kamera képének a streamelésére. A stream H.264 kodeket használ, és a továbbítás UDP feletti RTP protokoll segítségével történik. Az MPO modul képes változtatni a továbbított kamera kép felbontását (az adott készülék kamerájának maximális felbontásáig) az aktuális hálózati kontextust figyelembe véve. A jövőbeni fejlesztések során MPEG-DASH alapú média streamet szeretnénk biztosítani az egészségügyi alkalmazások számára, ahol dinamikusan kiválasztható a használni kívánt média reprezentáció, továbbá a kodekek kiválasztása is a kontextus függvényében történik.

[4] LibStreaming könyvtár: https://github.com/fyhertz/libstreaming

A DTE harmadik komponense a Szenzor Kiválasztó Modul (Sensor Selection Module – SSM), mely a használt szenzorok kiválasztásáért, konfigurációjáért felelős. A NACD modul elküldi a DTE részére az elérhető, az alkalmazás által használni kívánt szenzorok listáját. Minden szenzornak megvannak a saját hálózati erőforrás, QoS/QoE igényei. Ezen igények alapján az SSM eldönti, hogy az adott hálózati paraméterek mellett melyik szenzor használható, illetve felkonfigurája őket az aktuális hálózati kontextushoz optimális paraméterekkel.

4.1.4. ALKALMAZÁSI ÉS HÁLÓZATI KONTEXTUS FELDERÍTŐ MODUL IMPLEMENTÁCIÓJA

Az Alkalmazási és Hálózati Kontextus Felderítő modult egy háttérben futó Android szolgáltatásként (Android service) implementáltam. A NACD négy fő alrendszerből áll: az Alkalmazási Kontextus Felderítés (Application Context Discovery – ACD), a Hálózati Kontextus Felderítés (Network Context Discovery – NCD), a DDE információ nyelő (DDE Consumer Module – DDEC) és a DDE információ forrás (DDE Producer Module – DDEP) modulból. A hálózati kontextus felderítés két különböző forrás alapján történik. Egyrész a javasolt keretrendszer saját maga által is végez méréseket az adott hálózati paraméterekről. A csomagvesztési rátát a ping bináris, sávszélességet a bwping bináris segítségével. A ping bináris elérhető Android platformokon, a bwping[5] binárist az általam készített keresztfordító eszköztár segítségével portoltam. Ahhoz, hogy Android alkalmazásból vagy szolgáltatásból tudjunk futtatni natív binárisokat, és képesek legyünk feldolgozni az általuk szolgáltatott adatokat, egy külső könyvtárat, a RootCommands[6] könyvtárat kellett használnom. A többi hálózati információt a DDE szolgáltatja. A NACD modul a DDEC és DDEP alrendszerein keresztül kommunikál a DDE-vel, TCP alapú socket kommunikációs sémát használva. A DDEP komponensen keresztül küldhetünk adatokat a DDE számára, és a DDEC modult használva fogadhatjuk azokat DDE-ből érkező eseményeket, melyekre feliratkoztunk. A DDE egy speciális adatreprezentációt követel meg a kommunikáció során, melyet XDR-nek (External Data Representation) neveznek [41]. A fejlesztés során a finn partnerektől kapott Java alapú XDR könyvtárat használtam fel.

[5] BWPing bináris: http://bwping.sourceforge.net/

[6] RootCommands könyvtár: https://github.com/dschuermann/superuser-commands

A keretrendszer jelenlegi implementációja alapján az Android terminál az aktuális geografikus pozícióját küldi el a DDE-t használva, és lekérezi a DNIP-től és a NIS-től a készülék földrajzi pozíciójában elérhető hálózatok dinamikus és statikus információit. Ezeket a paramétereket a DDEC alrendszeren keresztül kapjuk meg.

4.1.5. MHEALTH ALKALMAZÁS

A javasolt keretrendszer architektúrájának legfelső szintje az alkalmazási réteg, itt futnak az Android platform által menedzselt alkalmazások. Ebben a fejezetben egy általam tervezett valós-idjű, a mobil távfelügyeleti mHealth forgatókönyvet lefedő alkalmazást részletezek. Az alkalmazás a jelen implementációt tekintve hat részre bontható, de a moduláris fejlesztési koncepciónak köszöhetően ez később az adott igényeknek megfelelően bővíthető.

A szenzor adatokat fogadó modulon (Sensor Data Receiver - SDR) keresztül kapja meg az alkalmazás a készülékhez csatlakoztatott egészségügyi szenzorok által mért adatokat. Ez a modul van feliratkozva a Sensor Data Aggragator alrendszer eseményeire, melyeken keresztül megkapja a szenzorból érkező, a SDA által feldolgozott adatokat. A szenzor adatokat küldő modul (Sensor Data Sender) továbbítja a mért egészségügyi információkat a kórházba/egészségügyi krízis központba vagy egy általános mHealth felhőszolgáltatásba TCP/UDP socket alapú kommunikációs sémát használva. A kommunikáció kétirányú, a kórház, a felhő szolgáltatás is küldhet a TCP/UDP socketen visszajelzéseket a feldolgozott egészségügyi adatok alapján.

A visszajelzéseket az Értesítés fogadó, angolul Notification Receiver (NR) kezeli. Az alkalmazás lehetőséget biztosít, hogy ezeket a visszajelzéseket továbbítsuk a kijelzővel rendelkező szenzorok (pl.: okosóra) felé. Ezt a funkcionalitást az Értesítés küldő (Notification Sender - NS) valósítja meg. A szenzorra küldött értesítéseknél célszerű a gyártóspecifikus API-kat használni, az általam használt Samsung API-t a szenzorokat tárgyaló fejezetben fogom részletezni.

Ezeken felül az alkalmazás tartalmaz még egy vizualizációs modult, mely a szenzor információk, értesítések, média adatok grafikus felületen történő megjelenítésért felelős. Az alkalmazás a jelenlegi implementációs fázis alapján képes grafikonon megjeleníteni a szenzorok által mért, a kórház felé továbbított szívritmus adatokat, továbbá mutatja az eszköz kamerájának streamelését az Android beépített SurfaceView komponensét

felhasználva. A grafikonok készítésére egy külső könyvtárat, a graphview-t[7] használtam, mely lehetővé teszi az adatok valós-idejű megjelenítését.

4.2. SZENZOROK

Napjainkban számos, okostelefonokhoz csatlakozatható egészségügyi szenzor segíti az mHealth alkalmazásokban rejlő lehetőségek kibontakozását. Ezek a szenzorok különböző vezeték nélküli technológiákat (pl. Bluetooth, ANT+, Wi-Fi) használva kapcsolódnak a mobil készülékekhez. Ezen felül az egészségügyi alkalmazások (elsősorban a telekonzultáció, irányított/vezetett beavatkozás use-case-ek) gyakran használják a készülékekbe beépített szenzorokat (pl. nagy felbontású kamera) média tartalom küldésére a kórház/krízisközpont felé. Az általam jelenleg használt tesztrendszerben a készülékbe beépített kamerát, és két viselhető egészségügyi szenzort, egy Samsung Gear Fit okosórát és egy Zephyr HxD szívritmus monitort alkalmaztam. Ezek adatai a Sensor Aggregator modul dolgozza fel. A javasolt keretrendszer modularitásának köszönhetően könnyen csatlakoztathatunk új szenzort a rendszerbe. Az általam jelenleg használt szenzorok részletes leírását a következő alfejezetek tárgyalják.

4.2.1. ZEPHYR HXM

A Zephyr HxM egy okostelefonokhoz csatlakoztatható szívritmus monitorozó eszköz. Bluetooth protokollt használva kapcsolódik a mobil készülékhez.

9. ábra Zephyr HxM szívritmus szenzor

Képes mérni a szívritmust, a megtett távolságot, sebességet és az RR értéket, ami a szívritmus inverz paramétere. A javasolt keretrendszerben a Sensor Aggregator modul

[7] GraphView könyvtár: http://android-graphview.org/

lekérdezi ezeket a paramétereket a Zephyr-től és a mobil eszközön keresztül továbbítja a kórház/krízisközpont/mHealth felhő számára. Az Android telefon és a Zephry közötti kommunikáció implementálásához a gyártó által biztosított Zephry SDK-t használtam. Az eszközök Bluetoothon keresztüli párosításához az Android SDK-ban található *BluetoothDevice* osztályt használtam fel. A párosítás után a Zephyr SDK a *ZephryProtocol* és *HRSpeedDistPacketInfo* osztályok segítségével lehetővé teszi a mért adatok lekérdezését a csatlakoztatott készüléktől.

4.2.2. SAMSUNG GEAR FIT

A Samsung Gear Fit egy okosóra és egy fitnesz/egészségügyi monitorozó eszköz keveréke. A Gear Fit lehetőséget biztosít a viselője szívritmusának, napi sport aktivitásainak (pl. futás, biciklizés, gyaloglás) és a gyakorlatok alatt nyomonkövetett életjel adatok (pl. szívverés, elégetett kalória, megtett távolság, sebesség) monitorozására és mentésére. Az okosóra képes fogadni a Bluetoothon keresztül hozzá csatlakoztatott okostelefontól/tablettől érkező értesítéseket.

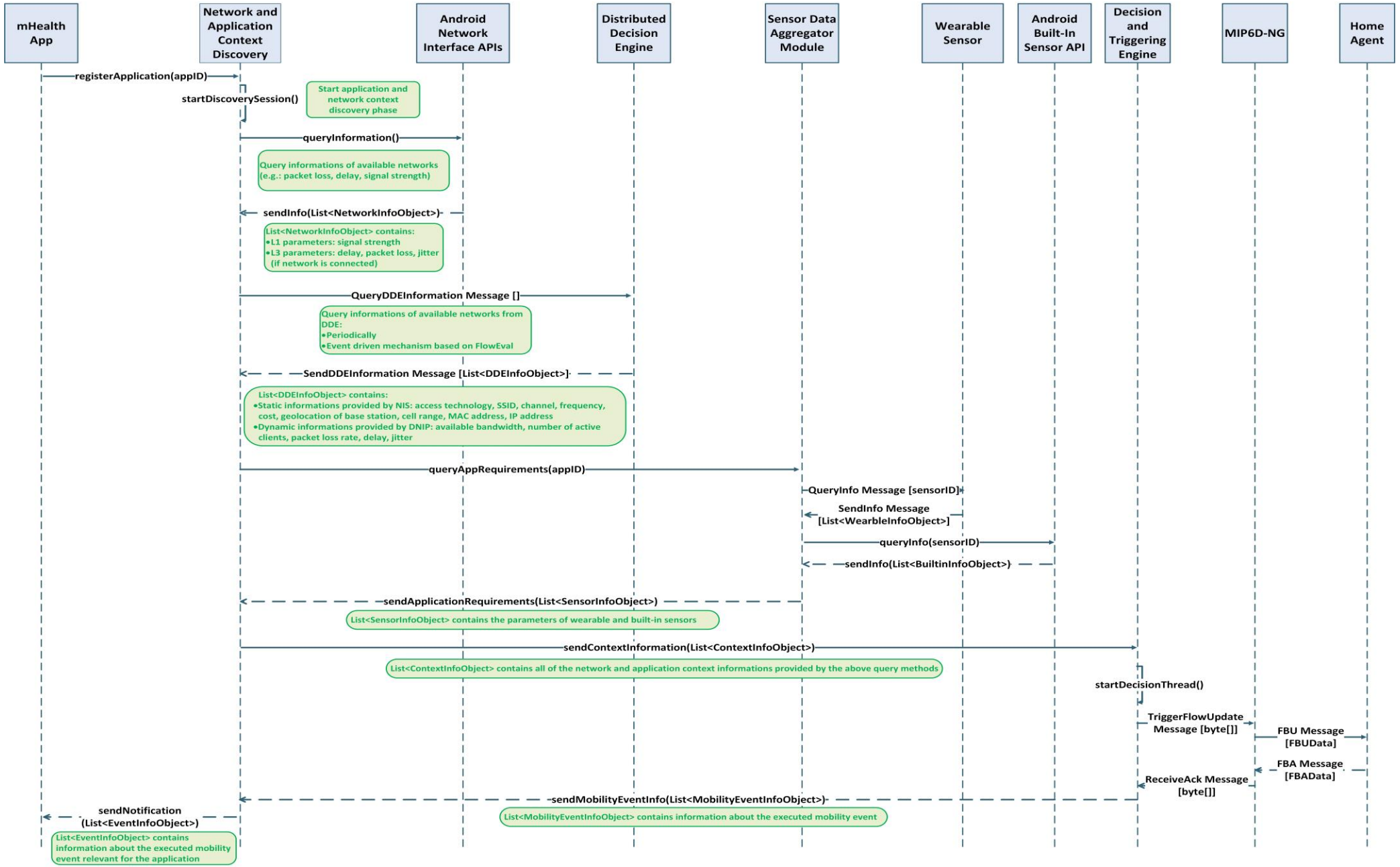
A Samsung a fejlesztők számra egy ún. Mobile SDK-t biztosít, mely lehetővé teszi az okosóra egyes funkcióinak menedzselését a telefonon futó alkalmazásból. Az SDK 16 független könyvtárat tartalmaz a fejlesztéshez, melyből én a Remote Sensor Package-t használtam, melynek segítségével lekérdezhetem a jelenlegi aktivitási státuszt, a beépített lépésszámláló adatait, valamint az okosóra viseleti státuszát. A fejlesztés ideje alatt a Remote Sensor Package béta verzióban volt, és nem biztosított hozzáférést a szívritmus szenzor adatainak a manipulálásához.

10. ábra Samsung Gear Fit szenzor

Az okosóra megjeleníti a kórházból/krízisközpontból/mHealth felhőszolgáltatásból érkező értesítéseket a kijelzőjén. Az értesítés először a telefonhoz érkezik be, majd a Notification Sender modul küldi tovább a Gear Fit részére az értesítést Bluetoothon keresztül.

4.2.3. ANDROID BEÉPÍTETT KAMERA

Az Android alapú okostelefonok szinte kivétel nélkül rendelkeznek beépített kamerával és a készülékek nagy részébe szerelt kamera minősége már megfelelő különféle mHealth szolgáltatások támogatására. Az olyan mHealth forgatókönyvek számára, melyekben fontos szerepe van a valós-idejű kamera kép továbbításának, a beépített kamera képének streamelése jó megoldást jelenthet. Az streamelés implementációja során az Android rendszerekhez írt LibStreaming külső könyvtárat vettem alapul. Ez a könyvtár lehetőséget biztosít az Android beépített kamera képének a streamelésére. A stream H.264 kodeket használ, és a továbbítás UDP feletti RTP protokoll segítségével történik. A könyvtár lehetőséget ad a továbbított kamera kép felbontásának változtatására, több beépített kamera esetében (pl. előlapi és hátoldali kamerák) a forrás kiválasztására.



11. ábra A javasolt keretrendszer funkcionális diagramja

4.3. A JAVASOLT KERETRENDSZER FUNKCIONLIS ÖSSZEFOGLALÁSA

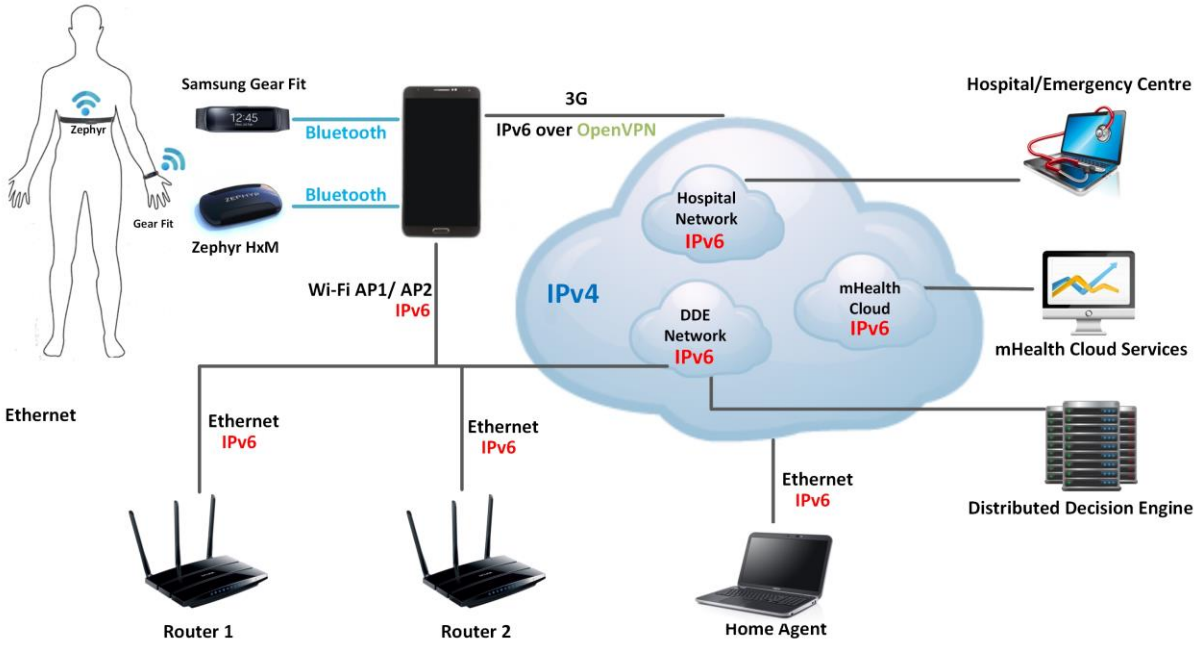
A javasolt keretrendszer funkcionális diagramját a 11. ábra mutatja. Az ábra tartalmazza a bemutatott architektúra fő moduljait. A modulok között kétféle kommunikációs sémát definiáltam. Általában az azonos fizikai eszközön elhelyezkedő alrendszer eseményvezérelt módon, függvényhívásokon keresztül kommunikálnak egymással. Ez az sémát a következő formátumban jelölöm a diagramon: *functionName(paramterList)*. A többi komponens hálózati interfészüket használva küldenek egymásnak üzeneteket (a javasolt rendszerben legtöbbször TCP alapú socket implementációt használva), melyeket *FuncionName Message [parameterList]* formában jelölök az ábrán. A funkcionális diagram továbbá tartalmazza a komponensek közötti üzenetváltások során átvitt adatokat. A keretrendszer egyes komponenseit az előző fejezetekben részletesen bemutattam, ebben az alfejezetben a keretrendszer működésének rövid összefoglalása olvasható.

Egy általános mHealth alkalmazás esetében az első lépés a keretrendszer felé történő regisztráció, melynek hatására a keretrendszer elkezdi az alkalmazás menedzselését. A javasolt rendszer először lekérdezi az aktuális hálózati kontextust az Android interfész kezelő API-jait, valamint a DDE rendszert felhasználva. Ezen komponensek részletes információkat szolgáltatnak az elérhető hálózatokról. Az információk különböző TCP/IP stack rétegéből származnak, például a fizikai rétegből (pl. jelerősség) vagy hálózati rétegből (pl. csomagvesztés, aktuálisan elérhető sávszélesség, késleltetés), mely hatékony feltérképezést tesz lehetővé. A keretrendszer következő lépése az mHealth szolgáltatás által használt készülékbe integrált vagy ahhoz vezeték nélküli technológiával kapcsolódó szenzorok paramétereinek a lekérdezése. Az alkalmazási és hálózati kontextus adatokat elküldve a DTE modulnak, a komplex döntési algoritmus triggereli a megfelelő mobilitás-kezelési feladatokat, valamint optimalizálja a média és szenzor folyamokat. A DTE modul a MIP6D-NG API-ján keresztül képes triggerelni a szükséges feladatokat. A natív MIP6D-NG elvégzi az optimális hálózatkezeléshez szükséges protokoll szintű feladatokat. A végrehajtásról értesíti a NACD modult, ami pedig tovább küldi az alkalmazásnak a releváns információkat. A keretrendszer mindig az optimális kontextust rendeli hozzá az alkalmazáshoz, a folyamatos valós-idejű monitorozási alrendszereknek köszönhetően.

5. A TESZTKÖRNYEZET ÉS MÉRÉSI EREDÉNYEK BEMUTATÁSA

5.1. TESZTRENDSZER BEMUTATÁSA

A javasolt keretrendszer hatékonyságát a 12. ábra által bemutatott tesztrendszer segítségével validáltam. A tesztrendszer központi eleme az Android alapú okostelefon, mely jelen esetben egy Samsung Note 3 készülék. A készülék támogatja a legfontosabb és legújabb vezeték nélküli kommunikációs technológiákat: 3G, 4G/LTE, Wi-Fi, NFC, Wi-Fi Direct, Bluetooth 4.0. A Note 3 az egyik legerősebb hardveres teljesítménnyel (FullHD felbontású kijelző, 3GB memória, Qualcomm Snapdragon 800 Quad-core 2.3 GHz processzor, 13 MP hátoldali és 2MP előlapi kamera) rendelkezik a jelenlegi okostelefon/táblagép piacon. A Note 3 az előbbi fejezetben bemutatott, általam készített ROM-ot futtatja, tehát képes kommunikálni 3G/4G és Wi-Fi interfészen keresztül egyidejűleg, speciális kernellel rendelkezik és tartalmazza a MIP6D-NG ARM architektúrára portolt binárisait és könyvtárait (a tesztek során a 3G interfészt használtam). A tesztek során három szenzor szolgáltat adatokat a kórház/krízis központ számára: a beépített kamera szenzor, a Samsung Gear Fit okosóra, valamint a Zephyr HxM szívritmus monitor.



12. ábra A mérési tesztkörnyezet

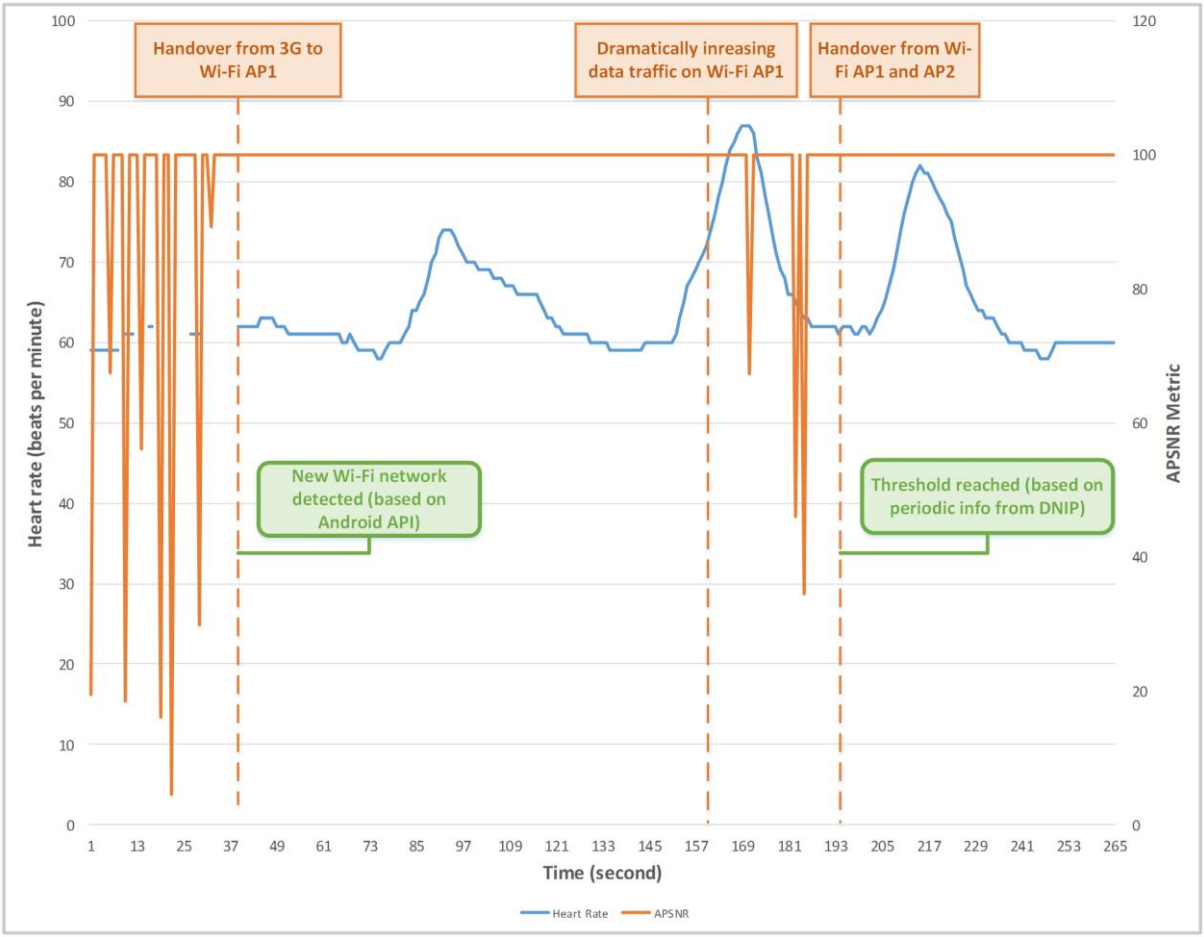
A Gear Fit és a Zephyr BT 4.0-n keresztül kommunikál az okostelefonnal. A Note 3 OpenVPN segítségével képes IPv6 alapokon kommunikálni a 3G interfészen. A tesztkörnyezetben két, változtatható paraméterű Wi-Fi hozzáférési pont és a folyamatosan elérhető 3G hálózat

biztosítja a csatlakozást a kórház és egyéb kommunikációs partnerek felé. A routerek szerepét TP-Link WDR3400 v1.7 eszközök töltik be. A routereken OpenWRT operációs rendszer (Barrier Breaker 14.07 verzió) fut. Az otthoni ügynök entitás egy MIP6D-NG-t futtató (MIP6D-NG támogatású, speciális kernellel fordított) egy Dell Inspiron 7720 notebook valósítja meg.

5.2. MÉRÉSI FORGATÓKÖNYVEK

5.2.1. A JAVASOLT KERETRENDSZER FUNKCIONÁLIS TESZTELÉSE

Az első teszt forgatókönyvben a mobiltól a kórház felé továbbított videó és szívritmus adatok minőségét vizsgáljuk különböző mobilitási események közben. Az eredményt a 13. ábra szemlélteti. Két folyamot definiáltam a mérés során: egy UDP alapú video küldést és egy szintén UDP alapú szenzor adatokat továbbító folyamot.



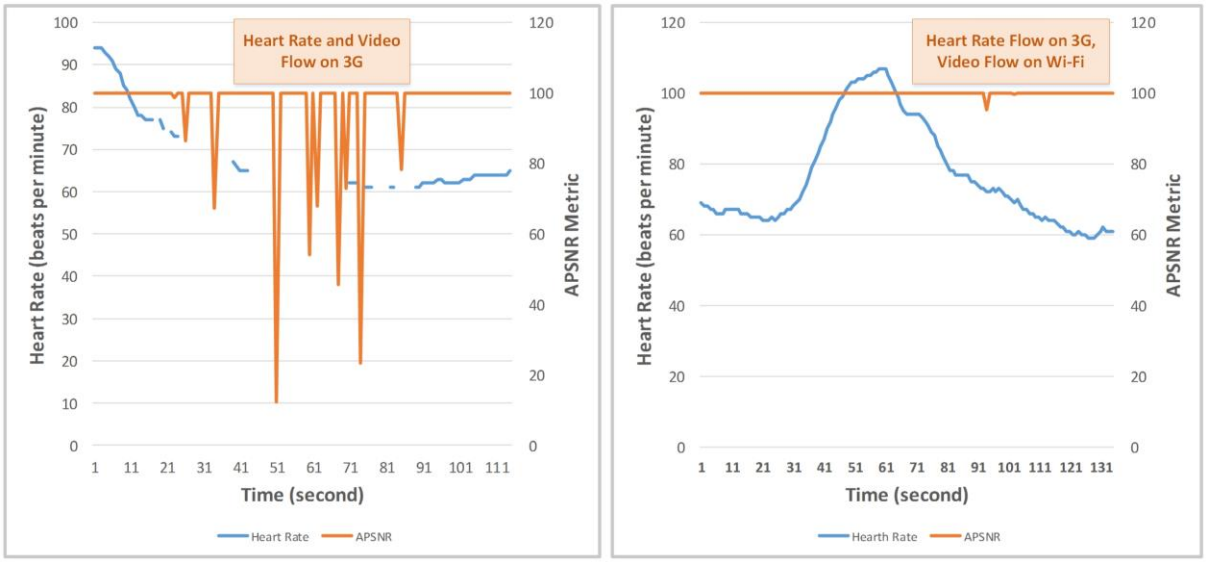
13. ábra Videó és szenzor adatok folyamainak vizsgálata heterogén környezetben

A mérés első fázisában mindkét folyam az egyetlen elérhető vezeték nélküli hálózatra, a 3G interfészhez van rendelve. A 3G hálózati erőforrásai nem elegendőek egy magas felbontású videó és szenzor adatok szimultán továbbítására, emiatt mind két folyam esetében nagy lesz a csomagvesztési ráta. A videó folyamnál ezt a hozzá tartozó APSNR objektív QoE metrikával [42] szemléltetem. Egy tökéletesen átvitt videó keret APSNR értéke 100. Az ábra jól mutatja, hogy a csomagvesztésnek köszönhetően számos hibás videó keret keletkezik, melynek hatására a videó minősége leromlik. A szívritmus folyam esetében a csomagvesztést mutatja az ábra (szaggatottá váló grafikon). A UDP alapú implementációba beépítettem egy csomag sorszámozási eljárást, melynek segítségével ellenőrizni tudom, hogy az adott sorszámú csomag megérkezett-e és ha igen, akkor milyen sorrendben a vételi oldalon. Az első fázisban azt látjuk, hogy számos szenzor adat elveszlett, ami egyes mHealth forgatókönyveknél megengedhetetlen.

Ezután megjelenik két Wi-Fi hálózat, melyet a javasolt keretrendszer a megfelelő API-k segítségével detektál. Ezekről a hálózatokról információt kér a DDE-től, és ez alapján csatlakozik a jobb paraméterekkel rendelkező Wi-Fi hálózathoz. A keretrendszer ennek hatására, átteszi a videó folyamot a Wi-Fi interfészre. A szenzor adatfolyam marad a 3G-n, mivel a szívritmus szenzorfolyam profilja számára elegendőek a 3G erőforrásai, továbbá a 3G biztonságosabb átvitelt nyújthat a biztonságkritikus egészségügyi adatok számára. Látható, hogy ebben az esetben később már nem történik csomagvesztés. Azonban a Wi-Fi hálózaton megnövekszik az adatforgalom (pl. szomszédos mobiltelefonok/laptopok sávszélesség-igényes letöltéseket indítanak). A csökkenő hálózati erőforrás hatására ismét csomagvesztés következik be a videó folyamon, ami romló QoE-t eredményez a vételkor. A DDE-től periodikusan lekért adatok segítségével a keretrendszer észleli a fogyó erőforrásokat az adott linken. Mivel elérhető egy másik, a videó folyam QoS igényeinek megfelelő paraméterekkel rendelkező, kevésbé terhelt Wi-Fi hálózat is, így átmozgatja azt az egyik Wi-Fi AP-ról a másikra. Az ezt követő fázisban mind a videó, mind a szenzor adat csomagvesztés nélkül kerül továbbításra.

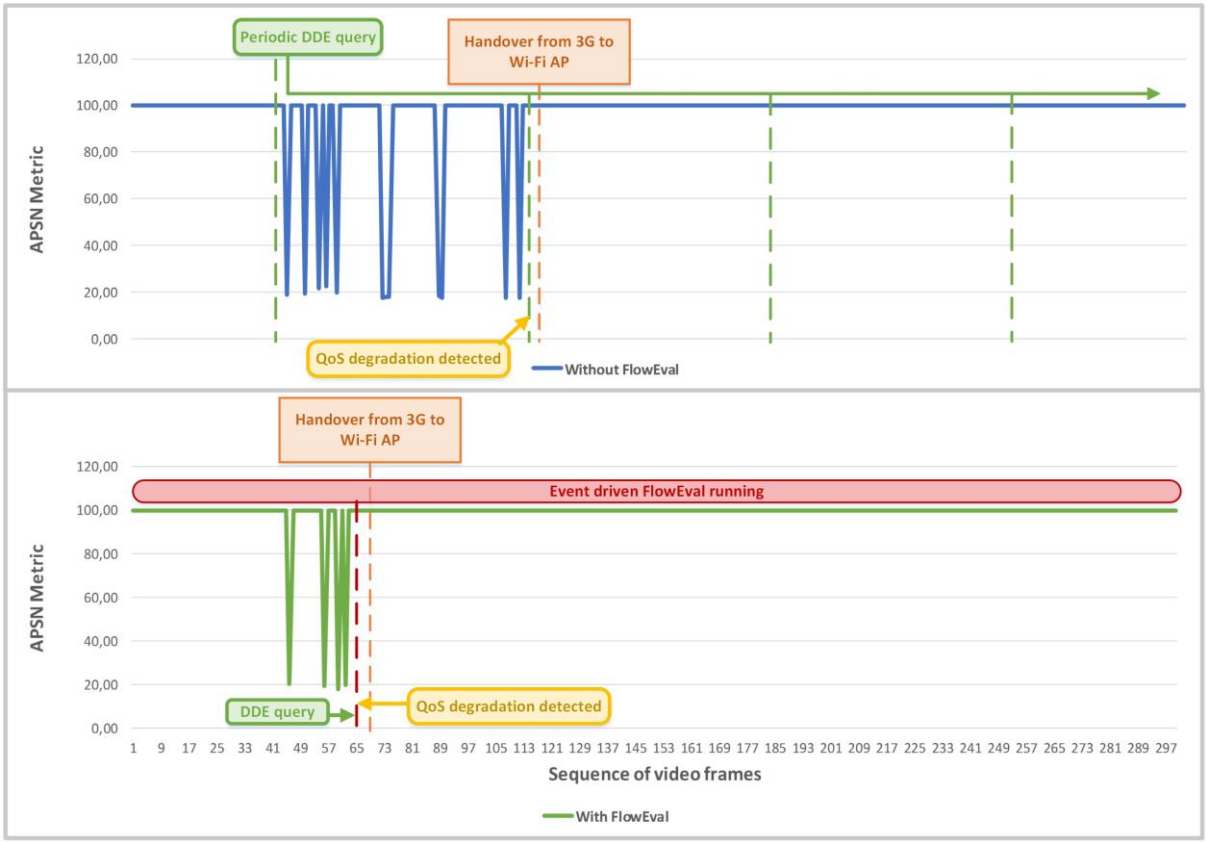
Ehhez hasonló mérési forgatókönyvet mutat be a 14. ábra is. Az első grafikon esetében mindkét folyam az egyetlen elérhető 3G interfészre van inicializálva, melynek erőforrásai nem elegendőek mindkét folyam számára, így csomagvesztés történik, ami a videó minőségének romlását, valamint számos szenzor adat elvesztését eredményezi. A második

grafikon azt az esetet mutatja, amikor 3G és Wi-Fi interfész is elérhető, így már a továbbítás megkezdésekor külön interfészhez rendelhetjük a folyamokat a keretrendszerünk segítségével.



14. ábra Videó és szenzor adatok folyamainak inicializálása hetergoén környezetben

A következő mérési forgatókönyv a periodikus és esemény vezérelt információ gyűjtési séma hatékonysága közötti különbségeket mutatja be. A hálózati kontextus alapos felderítését a DDE végzi a NIS és DNIP alrendszerek segítségével. Az információ lekérdezés a DDE-ből történhet periodikus és esemény-vezérelt módon. A periodikus lekérdezés implementációja alapján a rendszer megadott időközönként küld egy kérést a DDE felé. A FlowEval az okostelefonon futatott belső mérési metódusok alapján képes detektálni a hálózati paraméterekben bekövetkezett változásokat, melynek hatására kérést küld a DDE felé. A mért eredményeket a 15. ábra mutatja. A FlowEval az esetek többségében gyorsabban képes reagálni a környezeti változásokra, mint a periodikus mechanizmus, növelve ezzel a rendszer hatékonyságát. Azokban az esetekben amikor a hálózati kontextusban bekövetkezett változás a mérési periodusok között történik meg, akkor a rendszer csak a következő mérési ciklusban fogja detektálni az eseményt. A hálózati erőforrás takarékosságot figyelembe véve a mérési periodust nem csökkenthetjük elég kicsire ahhoz, hogy az esemény vezérelt mechanizmusnál jobb hatásfokot érjen el. Az ábra egy olyan esetet mutat, ahol a periodikus lekérdezi sémát tekintve az adott hálózat romlása pont egy adott mérési ciklusa után következik be. E degradációt csak a következő periodusában fogja detektálni, azonban addig a videó minősége (APSNR értéke) jelentősen romlik.



15. ábra Periodikus és esemény vezérelt (FlowEval) triggerelési sémák összehasonlítása

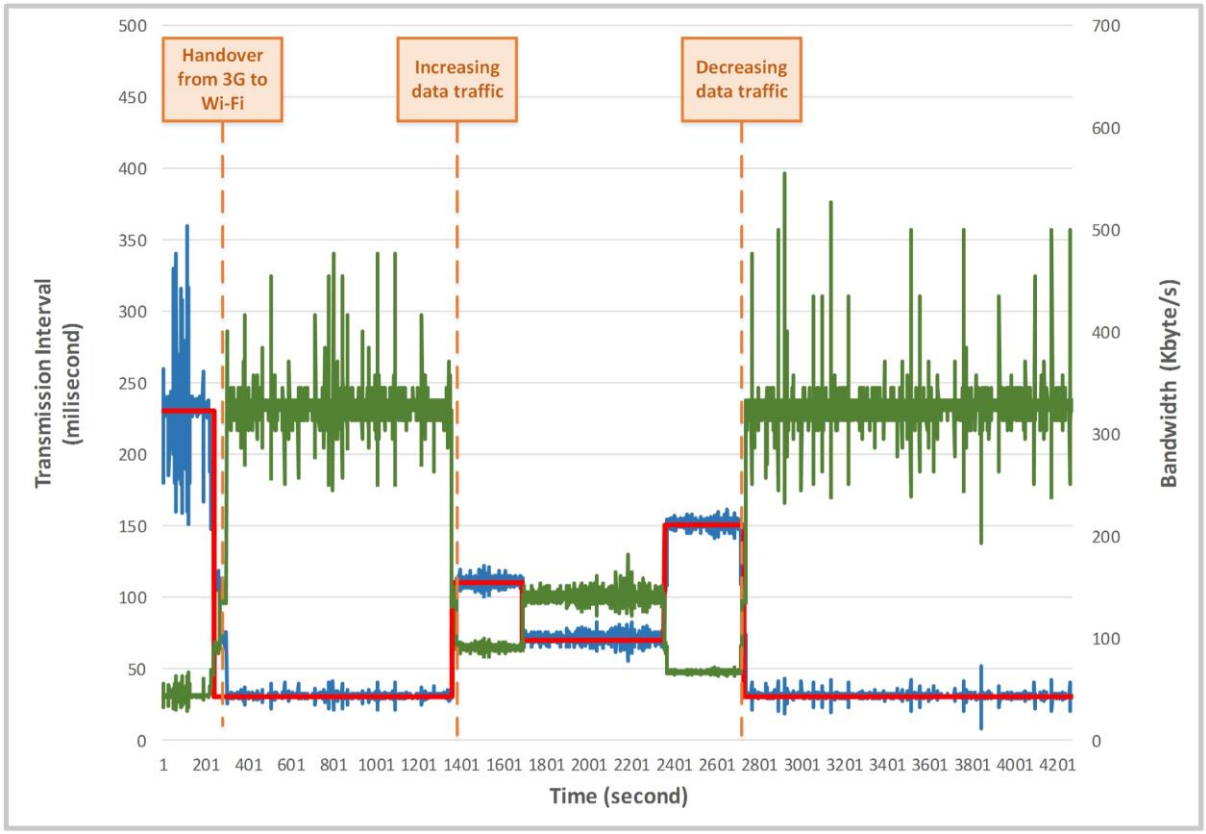
Látható, hogy a FlowEval kihasználja a telefonon futó mérési eredményeket, és jóval hamarabb detektálja és triggereli a szükséges mobilitás-kezelési funkciókat, vagyis áthelyezi a videó folyamot a leromlott paraméterekkel rendelkező 3G interfészről, a DDE-ből érkező adatok alapján, a videó folyam számára megfelelő QoS értékeket nyújtó Wi-Fi interfészre.

5.2.2. MÉDIA FOLYAM OPTIMALIZÁCIÓS TESZTEK

Ebben a mérési forgatókönyvben a javasolt keretrendszer adaptív média optimalizálást végző funkcióját teszteltem. A rendszer képes változtatni a média folyam paramétereit (csomag méret és csomagküldési intervallum) az aktuális hálózati kontextus függvényében, ezzel biztosítva az elérhető hálózati erőforrások feltétele melleti optimális átvitelt.

Az ábrán a piros vonal mutatja a Java szintű implementációban rögzített csomagküldési frekvenciák értékét (ms felbontásban), míg a kék vonal a vételi oldalon mért csomagok közötti intervallumot. A zöld vonal az aktuális küldési sebességet prezentálja kbyte/s bontásban. A keretrendszer periodikusan lekérdezi a DDE-től az elérhető hálózatok paramétereit, melyek a döntési algoritmus alapjul szolgálnak. A mérés során a küldési

intervallumot változtatja adaptívan a rendszer, ezzel emulálva az SVC és MPED-DASH algoritmusokhoz hasonló viselkedését. Öt különböző gyakoriságot (vagyis MPEG-DASH reprezentációt) definiáltam. Az első fázisban a videó folyam a 3G interfészre van inicializálva, melynek sávszélessége a tesztrendszeremben jóval kisebb, mint a Wi-Fi hálózatoké, így ehhez a legkisebb csomagküldési frekvenciát rendeli a rendszer. Ez azt jelenti, hogy a használt beállítások szerint 230 ms-ként küld ki egy csomagot.

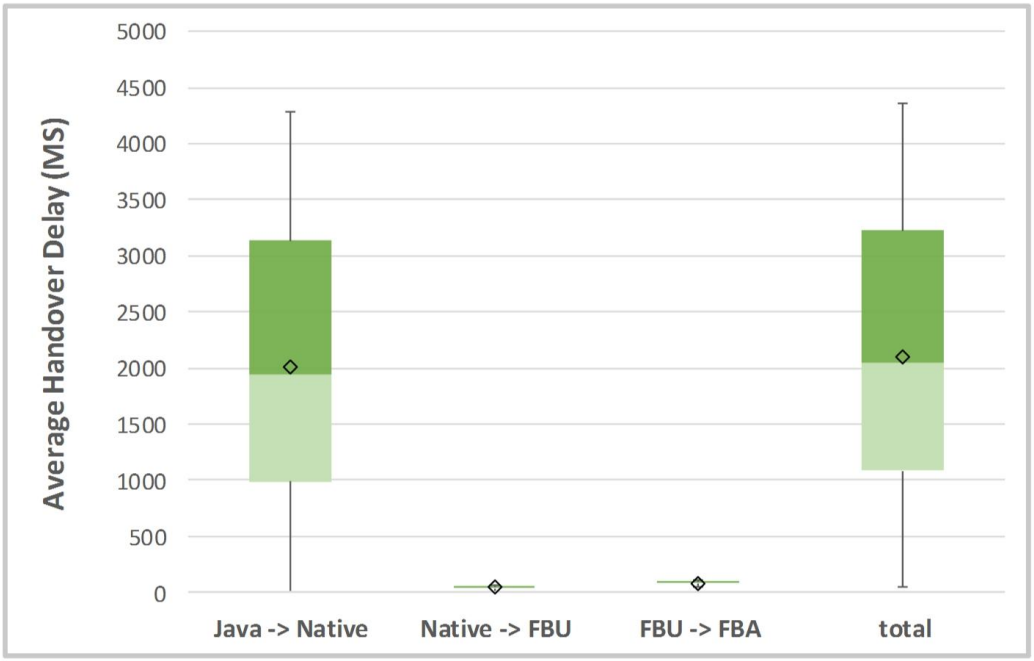


16. ábra Média folyam optimalizálás

Amikor elérhetővé válik egy Wi-Fi hálózat, a rendszer átmozgatja a folyamot az új hálózatra. A DDE által küldött adatok alapján, az elérhető hálózati paraméterek tökéletesnek bizonyulnak, így a legnagyobb küldési frekvenciát társítja a rendszer ehhez a fázishoz (30 ms-ként küld csomagot). Ez azt jelenti, hogy egy MPEG-DASH implementáció esetén a legnagyobb felbontású reprezentációt választanánk. Ezután különböző mértékben terhelem a Wi-Fi hálózatot, így az adott hálózati kontextusnak megfelelően választ a rendszer a küldési frekvenciák (MPEG-DASH reprezentációk) között. A terhelés megszüntetése után újra a legalacsonyabb küldési frekvenciát (vagyis legnagyobb felbontású, így legnagyobb sávszélességet igénylő MPEG-DASH reprezentációt) rendeli a keretrendszer a videó folyamhoz.

5.2.3. FLOW ALAPÚ HANDOVER TELJESÍTMÉNY TESZTEK

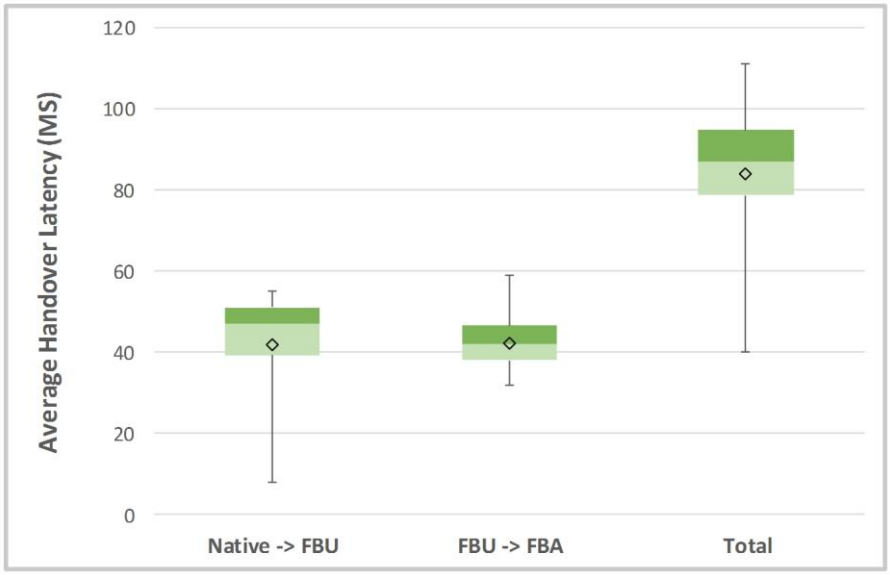
Ebben a fejezetben a javasolt keretrendszer folyam szintű mobilitás-kezélési mechanizmusának a teljesítmény elemzését mutatom be. A 17. ábra a teljes folyam szintű hívásátadás késletetésének egy komponenseit mutatja be.



17. ábra Folyam mobiltási teljesítmény teszt

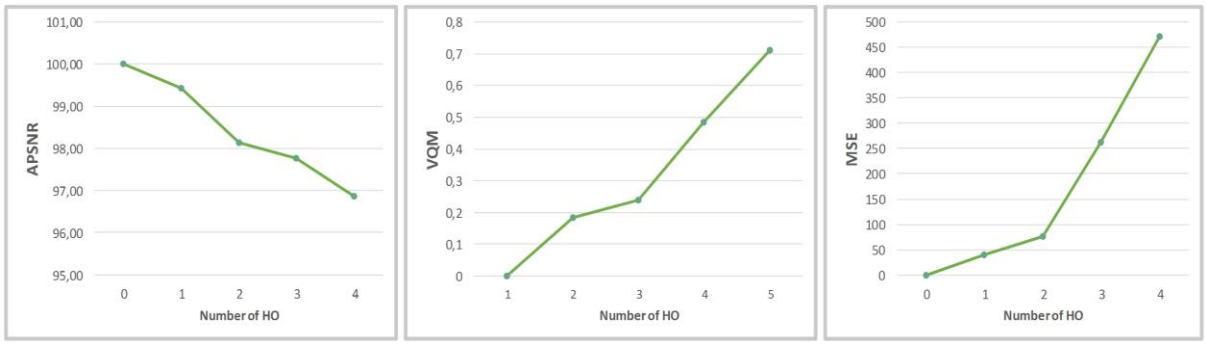
A késleltetés három komponensből tevődik össze. Az első komponens a Java szint és natív C szint közötti késleltetés. Ez a komponens azt az időtartamot mutatja, mely a Java implementációs szinten érkezett hálózatváltási trigger és a natív szinten futó MIP6D-NG API parancs lefutása között telik el. A Java szintű esemény hatására a keretrendszer feldolgozza a triggert, majd ennek alapján összeállítja, felparaméterezi, és elküldi a megfelelő mobilitás-kezelési üzenetet a MIP6D-NG API-jához nyitott TCP alapú socketen keresztül. A második komponens az API parancs MIP6D-NG-hez való beérkezése, feldolgozása és ennek hatására kiküldött Flow Binding Update üzenet között eltelt időtartalmat mutatja. A harmadik komponens tartalmazza a MIP6D-NG által kiküldött Flow Binding Update (FBU) üzenet és a Home Agent-től kapott Flow Binding Acknowledgement (FBA) üzenet között eltelt idő. A működés szempontjából ez a legfontosabb komponens, mivel ennek hatására történik meg a tényleges interfész váltás, az első két komponens ideje alatt a rendszer még az eredeti interfészen továbbítja a csomagokat. Az első komponens átlaga 2000 ms, a második és harmadik összetevő átlaga pedig körülbelül 40 ms. A 18. ábra a késleltetés natív

komponenseit, tehát a MIP6D-NG API üzenetének fogadása és az FBA megérkezése között eltelt időt ábrázolja. Ennek átlaga 80 ms körüli, tehát elhanyagolható. A mérések alapján látható, hogy a cross-layer optimalizálási séma biztosította mobilitás-kezelés optimális erőforrás támogatást nyújt az mHealth szolgáltatások számára.



18. ábra Folyam mobiltási natív teljesítmény teszt

A 19. ábra az átvitt videó minőségét mutatja a hívásátadások számának a függvényében. Három különböző, a videók minőségét jellemző metrikát (APSNR, VQM, MSE [42]) mértem 1-4 hálózatváltás esetében. A minőségi metrikák meghatározásához az MSU Video Quality Measurement Tool[8]-t használtam. A teszt során egy 4:2:0 mintavételű, QCIF (176x144) felbontású YUV formátumú videófájlt streameltem 90 másodpercig. Látható, hogy az optimális mobilitás-kezelésnek köszönhetően, akár rövid időn belül két hálózatváltás esetében sem romlik le drasztikusan az átvitt videó minősége.



19. ábra Média folyam minőség elemzése

[8] MSU VQMT: http://compression.ru/video/quality_measure/video_measurement_tool_en.html

6. ÖSSZEFOGLALÁS

Napjainkban az egyre nagyobb hardveres teljesítményű, több interfésszel rendelkező mobil eszközök és a vezeték nélküli, heterogén hozzáférési technológiák terjedésének, valamint az egészségügyi szenzorok olcsóvá és nagy tömegben elérhetővé válásának köszönhetően kiemelt figyelmet kapnak a mobil egészségügyi (mHealth) szolgáltatások.

TDK dolgozatomban a Jövő Internet heterogén architektúráiba illeszkedő, mobil egészségügyi támogató keretrendszert javasoltam és részleteztem a tervezési és implementációs megfontolásokat, valamint a teljesítményelemzés eredményeit. A javasolt megoldás központi része egy okostelefon-vezérelt, folyam alapú mobilitás-kezelési és döntési keretrendszer, mely a legkülönbözőbb szenzorinformációk QoS/QoE igényei és a rendelkezésre álló hálózatok aktuális állapota alapján képes döntést hozni az mHealth alkalmazás folyamainak és az elérhető hozzáférési hálózatoknak az optimális összerendeléséről.

A teljesítmény elemzésre és a javasolt keretrendszer funkcionalitásának tesztelésére készített mérési forgatókönyvek jól tükrözik, hogy a tervezett és implementált okostelefon-vezérelt, folyam alapú mobilitás-kezelési és döntési keretrendszer jól skálázható, kihasználja az aktuálisan rendelkezésre álló, eltérő rádiós technológiák által biztosított erőforrásokat, és a különböző tulajdonságú folyamoknak mindig optimális átvitelt biztosít, bármilyen mobilitási eseményről is legyen szó. A rendszer nagyrésze saját fejlesztésű, de tartalmaz az irodalomban már létező, integrált komponenseket is. Ezek a komponensek a javasolt keretrendszerben definiált kommunikációs és együttműködési séma alapján dolgoznak össze, egy egységes rendszert alkotva.

A javasolt keretrendszer legtöbb komponense egy önmagában is komplex döntési algoritmust, eljárást tartalmaz, melyeket a jövőbeni fejlesztések során tovább lehet finomítani, pontosítani a keretrendszer hatékonyságának további növelésének céljából. A jelenlegi média stream implementáció továbbfejlesztése is egy fontos aspektusa lesz a későbbi fejlesztéseknek, mely az MPEG-DASH szabványait kielégítő módosításokat tartalmazza majd. Továbbá a fejlesztési tervek közé tatozik az implementált mHealth alkalmazás és a vételi oldal (kórház és mHealth felhőszolgáltatás) funkcióinak a bővítése is.

7. ÁBRAJEGYZÉK

8. RÖVIDÍTÉSJEGYZÉK

3GPP	3rd Generation Partnership Project
802.21 MIH	802.21 Media Independent Handovers
ACD	Application Context Discovery
AHP	Analytical Hierachy Process
ANDSF	Access Network Discovery and Selection Function
AP	Access Point
APSNR	Average Peak-to-peak Signal-to-Noise Ratio
AR	Access Router
BID	Binding Update
BT	Bluetooth
BU	Binding Update
CCN	Content Centric Network
CN	Correspondent Node
CoA	Care-of-Address
CT	Computed Tomography
DDE	Distributed Decision Engine
DDEC	DDE Consumer Module
DDEP	DDE Producer Module
DNIP	Distributed Network Information Provisioning
DTE	Decision and Triggering Engine
ECG	Electrocardiography
ECS	Extended Connectivity Service
eHealth	Electronic Health
EMG	Electromyography
GUI	Graphical User Interface
HA	Home Agent
HTTP	HyperText Transfer Protocol
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force

IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
M2M	Machine-to-Machine
MAC	Medium Access Control
MCDM	Multiple-criteria Decision Making
MCoA	Multiple Care-of Addresses Registration
MDT	Mobility Decision and Triggering Module
mHealth	Mobile Health
MIF	Multiple Interface
MIMS	Multi Interface Manager System
MIP6D-NG	Mobile IPv6 Daemon – Next Generation
MIPv4	Mobile IPv4
MIPv6	Mobile IPv6
MN	Mobile Node
MPEG-DASH	MPEG's Dynamic Adaptive Streaming over HTTP
MPO	Media Playout Optimalization
MSE	Mean Squared Error
NACD	Network and Application Context Discovery
ND	Neighbor Discovery
NIS	Network Information Server
NR	Notification Receiver
QCIF	Quarter Common Intermediate Format
QoE	Quality of Experience
QoS	Quality of Service
RA	Router Advertisement
RFC	Requests for Comments
RTP	Real-time Transport Protocol
RTT	Round Trip Time
SDK	Software Development Kit
SDS	Sensor Data Sender

SNR	Signal-to-Noise Ratio
SSM	Sensor Selection Module
SVC	Scalable Video Coding
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
UDP	User Datagram Protocol
UMTS	Universal Mobile Telecommunications System
VoIP	Voice over IP
VQM	Video Quality Metric
Wi-Fi	Wireless Fidelity
WiMAX	Worldwide Interoperability for Microwave Access
WLAN	Wireless Local Area Network
XDR	External Data Representation
XML	Extensible Markup Language

9. HIVATKOZÁSOK

[1] Cisco Visual Networking Index, “Global Mobile Data Traffic Forecast Update, 2013–2018.” 05-Feb-2014.

[2] L. Bokor, Z. Faigl, and S. Imre, “Flat Architectures: Towards Scalable Future Internet Mobility,” in *The Future Internet*, vol. 6656, J. Domingue, A. Galis, A. Gavras, T. Zahariadis, D. Lambert, F. Cleary, P. Daras, S. Krco, H. Müller, M.-S. Li, H. Schaffers, V. Lotz, F. Alvarez, B. Stiller, S. Karnouskos, S. Avessta, and M. Nilsson, Eds. Springer Berlin Heidelberg, 2011, pp. 35–50.

[3] G. Eysenbach, “What is e-health?,” vol. J. Med. Internet Res., 2001.

[4] “mHealth - New horizons for health through mobile technologies,” *World Health Organ.*

[5] H. Greenspun and S. Coughlin, “mHealth in an mWorld, How mobile technology is transforming health care.” Deloitte Center for Health Solutions, 2012.

[6] PWC, “Touching lives through mobile health, Assessment of the global market opportunity,” Feb-2012. [Online]. Available: www.pwc.com/mhealth.

[7] PWC, “Emerging mHealth: Paths for growth.” 2012.

[8] V. Ghini, S. Ferretti, and F. Panzieri, “M-Hippocrates: Enabling Reliable and Interactive Mobile Health Services,” *IT Prof.*, vol. 14, no. 3, pp. 29–35, 2012.

[9] T. Bratan and M. Clarke, “Towards the Design of a Generic Systems Architecture for Remote Patient Monitoring,” in *Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the*, 2005, pp. 106–109.

[10] W. D. Bradford, “Telemedicine and Telehealth: Principles, Policies, Performance and Pitfalls by Adam W. Darkins and Margaret A. Cary. Free Association Books, London, 2000. No. of pages 316. ISBN 1-853-43518-X,” *Health Econ.*, vol. 10, no. 7, pp. 681–682, 2001.

[11] J. Williams and Ellis, “Primary care patients in psychiatric clinical trials: a pilot study using videoconferencing.” .

[12] A. De La Oliva, C. J. Bernardos, M. Calderon, T. Melia, and J. C. Zuniga, “IP flow mobility: smart traffic offload for future wireless networks,” *Commun. Mag. IEEE*, vol. 49, no. 10, pp. 124–132, Oct. 2011.

[13] J. Kim, Y. Morioka, and J. Hagiwara, “An optimized seamless IP flow mobility management architecture for traffic offloading,” in *Network Operations and Management Symposium (NOMS), 2012 IEEE*, 2012, pp. 229–236.

[14] Y.-N. Lin, W. Chen, S.-C. Tsai, and Y.-B. Lin, “Design and Implementation of an Offloading Technology for 3.5G Networks,” in *Vehicular Technology Conference (VTC 2010-Spring), 2010 IEEE 71st*, 2010, pp. 1–5.

[15] A. Takács and L. Bokor, “A Distributed Dynamic Mobility Architecture with Integral Cross-Layered and Context-Aware Interface for Reliable Provision of High Bitrate mHealth Services,” in *Wireless Mobile Communication and Healthcare*, vol. 61, B. Godara and K. Nikita, Eds. Springer Berlin Heidelberg, 2013, pp. 369–379.

[16] R. Haw and C. S. Hong, “A seamless content delivery scheme for flow mobility in Content Centric Network,” in *Network Operations and Management Symposium (APNOMS), 2012 14th Asia-Pacific*, 2012, pp. 1–5.

[17] T. Ropitault and N. Montavont, “Implementation of Flow Binding Mechanism,” in *Pervasive Computing and Communications, 2008. PerCom 2008. Sixth Annual IEEE International Conference on*, 2008, pp. 342–347.

[18] R. Silva, P. Carvalho, P. Sousa, and P. Neves, “Enabling Heterogeneous Mobility in Android Devices,” *Mob. Netw. Appl.*, vol. 16, no. 4, pp. 518–528, 2011.

[19] N. Varga, L. Bokor, and A. Takacs, “Context-aware IPv6 Flow Mobility for Multi-Sensor based Mobile Patient Monitoring and Tele-consultation,” in *2014 First International Workshop On Wireless Solutions For Healthcare Applications (Concerto 2014)*, Rome, Italy, 2014.

[20] K. Wac, R. Bults, B. Van Beijnum, I. Widya, V. Jones, D. Konstantas, M. Vollenbroek-Hutten, and H. Hermens, “Mobile patient monitoring: The MobiHealth system,” in *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*, 2009, pp. 1238–1241.

[21] M. V. Ramesh, S. Anand, and P. Rekha, “A mobile software for health professionals to monitor remote patients,” in *Wireless and Optical Communications Networks (WOCN), 2012 Ninth International Conference on*, 2012, pp. 1–4.

[22] D. Niyato, E. Hossain, and S. Camorlinga, “Remote patient monitoring service using heterogeneous wireless access networks: architecture and optimization,” *Sel. Areas Commun. IEEE J. On*, vol. 27, no. 4, pp. 412–423, May 2009.

[23] H. Schwarz, D. Marpe, and T. Wiegand, “Overview of the Scalable Video Coding Extension of the H.264/AVC Standard,” *Circuits Syst. Video Technol. IEEE Trans. On*, vol. 17, no. 9, pp. 1103–1120, Sep. 2007.

[24] N. Zotos, G. Xilouris, B. Shao, D. Renzi, and A. Kourtis, “Performance evaluation of H264/SVC streaming system featuring real-time in-network adaptation,” in *Quality of Service (IWQoS), 2011 IEEE 19th International Workshop on*, 2011, pp. 1–3.

[25] I. Sodagar, “The MPEG-DASH Standard for Multimedia Streaming Over the Internet,” *Multimed. IEEE*, vol. 18, no. 4, pp. 62–67, Apr. 2011.

[26] L. Pozueco, X. Pañeda, R. García, D. Melendi, S. Cabrero, and G. Orueta, “Adaptation engine for a streaming service based on MPEG-DASH,” *Multimed. Tools Appl.*, pp. 1–20, 2014.

[27] P. Yang, “Requirements on multiple Interface (MIF) of simple IP,” 27-Feb-2009. [Online]. Available: http://tools.ietf.org/id/draft-yang-mif-req-00.txt.

[28] M. Wasserman, *Current Practices for Multiple-Interface Hosts*, November 2011. .

[29] J. Yang, “Multi-interface Connection Manager Implementation and Requirements,” 04-Mar-2009. [Online]. Available: http://tools.ietf.org/id/draft-yang-mif-connection-manager-impl-req-00.txt.

[30] D. Johnson, C. Perkins, and J. Arkko, *Mobility Support in IPv6*. IETF, 2004.

[31] R. Wakikawa, V. Devarapalli, G. Tsirtsis, T. Ernst, and K. Nagami, *Multiple Care-of Addresses Registration*. IETF, 2009.

[32] G. Tsirtsis, H. Soliman, N. Montavont, G. Giaretta, and K. Kuladinithi, *Flow Bindings in Mobile IPv6 and Network Mobility (NEMO) Basic Support*. IETF, 2011.

[33] G. Tsirtsis, G. Giarreta, H. Soliman, and N. Montavont, *Traffic Selectors for Flow Bindings*. IETF, 2011.

[34] J. Mäkelä, M. Luoto, T. Sutinen, and K. Pentikousis, “Distributed Information Service Architecture for Overlapping Multiaccess Networks,” *Multimed. Tools Appl*, vol. 55, no. 2, pp. 289–306, Nov. 2011.

[35] Z. Kanizsai, L. Bokor, and G. Jeney, “An anycast based feedback aggregation scheme for efficient network transparency in cross-layer design,” vol. Vol 55, no. No 1–2 (2011), pp. pp. 45–52, Nov. 2013.

[36] E. Triantaphyllou and S. H. Mann, “Using the Analytic Hierarchy Process For Decision Making in Engineering Applications: Some Challenges,” *J. Ind. Eng. Appl. Pract.*, vol. 2, no. 1, pp. 35–44, 1995.

[37] V. Devarapalli, R. Wakikawa, A. Petrescu, and P. Thubert, *Network Mobility (NEMO) Basic Support Protocol*. IETF, 2005.

[38] H. Soliman, C. Castelluccia, K. ElMalki, and L. Bellier, *Hierarchical Mobile IPv6 (HMIPv6) Mobility Management*. IETF, 2008.

[39] N. Varga, L. Bokor, S. Bouroz, B. Lecroart, and A. Takács, “Client-based and Cross-Layer Optimized Flow Mobility for Android Devices in Heterogeneous Femtocell/Wi-Fi Networks,” *Int. Conf. Sel. Top. Mob. Wirel. Netw.*, Sep. 2014.

[40] N. Varga, L. Bokor, and A. Takács, “Android-based Testbed and Demonstration Environment for Cross-layer Optimized Flow Mobility,” in *9th International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities (TridentCom’2014), May 5–7, 2014 Guangzhou, People’s Republic of China*, 2014.

[41] M. Eisler, *XDR: External Data Representation Standard*. IETF, 2006.

[42] Y. Wang, “Survey of Objective Video Quality Measurements.” Worcester Polytechnic Institute, USA, 2006.