Android-alustan haasteet tehokkaan käytönvalvontasovelluksen toteuttamisessa

Jukka Hell

Tampereen yliopisto
Informaatiotieteiden yksikkö
Tietojenkäsittelyoppi
Pro gradu -tutkielma
Ohjaaja: Martti Juhola
Kesäkuu 2015

Tampereen yliopisto
Informaatiotieteiden yksikkö
Tietojenkäsittelyoppi
HELL, JUKKA: Android-alustan haasteet tehokkaan käytönvalvonnan toteuttamisessa
Pro gradu -tutkielma, 46 sivua
Kesäkuu 2015

Lapset viettävät nykyään yhä enemmän aikaa mobiililaitteiden parissa. Aika kuluu muun muassa pelaamiseen, sosiaalisessa mediassa viestittelyyn sekä Internetin selailuun. Etenkin älypuhelinten yleistyttyä viimevuosina voimakkaasti, ruutuajan ja käytön valvonta voi jäädä joissain perheissä täysin huomiotta. Syy tähän on tutkimusten mukaan vanhempien vähäinen tietämys käytönvalvonta-teknologiaan. Vähäinen tietämys saattaa osaltaan johtua siitä, ettei mobiililaitteisiin juurikaan tarjota sisäänrakennettuna mahdollisuutta rajoittaa tai valvoa lasten laitteiden käyttöä. Tässä pro gradu -tutkielmassa pyritään selvittämään, minkälaiset mahdollisuudet kolmannen osapuolen kehittäjillä on rakentaa toimiva käytönvalvontaohjelmisto Android-käyttöjärjestelmiin.

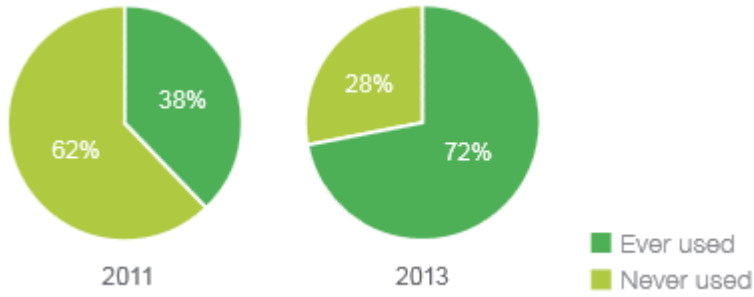
Kolmannen osapuolen kehittäjien käytönvalvonnan rakentamisen mahdollisuuksiin vaikuttavat ennen kaikkea Android-järjestelmän rajoitukset tietoturvan ja yksityisyyden varjelemisen vuoksi. Lisähaasteita tuo mobiililaitteiden käytönvalvonnan monesta eri osa-alueesta muodostuvat ominaisuudet, jotka voi toteuttaa monella eri tapaa. Esimerkiksi tähän päivään mennessä ei ole vielä onnistuttu luomaan ohjelmistoa, millä vanhemmat saisivat täydellisesti estettyä lapsia pääsemästä haitallisille Internet-sivustoille tai suodatettua lapsille sopimattoman median pois. Tämän vuoksi on kehitetty erilaisia menetelmiä ominaisuuden toteuttamiseksi. Lisäksi ihmisillä on monia erilaisia tarpeita käytönvalvonnan suhteen, mutta tärkeää jokaisessa ohjelmistossa on niiden yksinkertainen käyttö.

Avainsanat ja -sanonnat: Android, käytönvalvonta, mobiilisovellus, parental control.

Sisällys

1. Johdanto

Nykyään yhä useammassa perheessä myös pienillä lapsilla on oma mobiililaite, jolla pääsee Internettiin. Amerikassa 0 – 8-vuotiaista lapsista jopa 63 % omisti älypuhelimen vuonna 2013 [Common Sense Media, 2013]. Lasten moobililaitteiden käyttö on kasvanut räjähdysmäisesti vasta kuluvan vuosikymmenen aikana (Kuva 1), mikä mahdollisesti vaikuttaa siihen, ettei käytönvalvonta ole saanut vielä kaipaamaansa huomiota puhelinvalmistajien osalta.



Kuva 1. Mobiililaitteiden käyttömäärän kasvu 0 - 8-vuotialla. Lähde: [Common Sense Media, 2013].

Television ja tietokoneen käyttöä on helpompi valvoa ilman erityisiä käytönvalvontasovelluksia, sillä ne ovat yleensä kotona näkyvällä paikalla. Mobiililaitteita sen sijaan voi kuljettaa minne tahansa, jolloin käyttöä on käytännössä mahdoton seurata ja rajoittaa ilman toimivaa käytönvalvontasovellusta. Tietokoneiden käytössä valvonta kohdistuu usein vain peli-/käyttöaikaan ja Internet-liikenteeseen. Mobiililaitteissa voidaan näiden lisäksi rajoittaa myös muun muassa soittojen ja tekstiviestien määrää sekä seurata lapsen sijaintitietoja.

Android on ollut vuosina 2013 ja 2014 suosituin [W3Schools, 2014] mobiililaitteiden käyttöjärjestelmistä, eikä siihen ollut tehtynä vielä ennen heinäkuuta 2013 minkäänlaista sisäänrakennettua käytönvalvontaa. Heinäkuussa 2013 Google julkisti Androidin version 4.3, joka mahdollisti taulutietokoneisiin keveähkön käytönvalvonnan. Uusimmat Android-versiot eivät kuitenkaan ole välttämättä milloinkaan käytettävissä vanhemmissa Android-laitteissa. Android-puhelimiin ei ole vielä tätä kirjoittaessa tullut käytönvalvontaominaisuuksia. Käytönvalvontasovelluksen puutteen vuoksi monessa perheessä saattaa jäädä asia täysin huomiotta. Tutkimuksen mukaan mobiililaitteiden käytönvalvonnan käyttö on kohtalaisen heikkoa juuri vanhempien vähäisestä tietämyksestä käytönvalvonta-teknologiaan [Heart Research Associates, 2011]. Android-sovelluskeskuksesta kuitenkin löytyy useitakin eri vaihtoehtoja maksullisista ilmaisiin käytönvalvontasovelluksiin, mutta mikään niistä ei ole täydellinen ja sopivan löytäminen saattaa olla hankalaa. Turvallisuus- ja yksityisyysaspektien takia - joita käsitellään tarkemmin kappaleessa 3.3 - täydellistä kolmannen osapuolen käytönvalvontasovellusta on kuitenkin

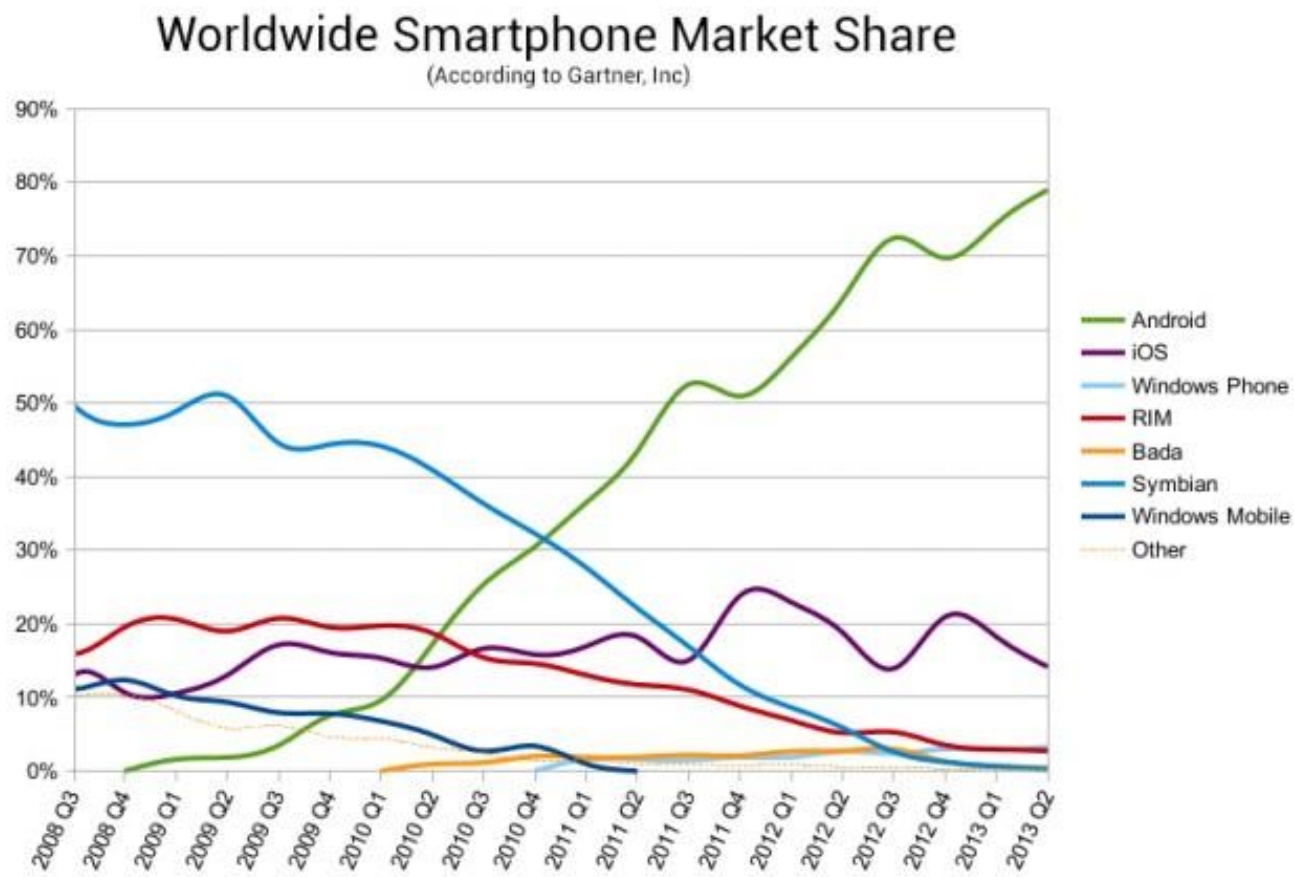
mahdotonta tehdä tällä hetkellä. Ainoastaan esiasennetuilla sovelluksilla on järjestelmätason oikeudet, joilla pystyy esimerkiksi estämään sovelluksen poistaminen lapsen toimesta. Tässä pro gradu -tutkielmassa (jatkossa ”tutkielma”) pyritään selvittämään, kuinka Android-alustalle on mahdollista tehdä mahdollisimman tehokas kolmannen osapuolen käytönvalvontasovellus ja minkälaisia vaihtoehtoja ongelmien ratkaisemiseksi voisi olla. Tutkielmassa pohditaan myös eri ratkaisuvaihtoehtojen hyviä ja huonoja puolia. Tutkimusaineistona käytetään pääosin toteuttamaani käytönvalvontasovelluksen prototyyppiä (jatkossa ”prototyyppi”) ja sen tekemisen yhteydessä kirjaamiani havaintoja erilaisista ongelmista ja ratkaisuvaihtoehdoista.

Tutkielman kolme ensimmäistä lukua ovat johdannollisia, joissa käydään tarkemmin läpi aiheita, joihin tämä tutkielma pohjautuu. Neljäs luku käsittelee tutkielman tarkemmat tavoitteet. Viidennessä luvussa eritellään käytönvalvontasovelluksen sisältämät ominaisuudet ja tutkimusmenetelmät. Lopuksi esitellään tutkimuksen tulokset ja yhteenveto.

2. Android-käyttöjärjestelmä

Android on ensimmäinen vapaaseen lähdekoodiin perustuva mobiililaitteille kehitetty sovelluskehitysalusta, jonka ensimmäinen SDK (Sofware Development Kit) julkaistiin vuonna 2007 [Ableson et al., 2011]. Androidin ydin pohjautuu Linux-käyttöjärjestelmään ja se on suunniteltu ensisijaisesti kosketusnäytöllisiin mobiililaitteisiin, kuten älypuhelimiin ja taulutietokoneisiin (tabletteihin). Android on ilmestynyt viime vuosina myös muun muassa rannekelloihin ja älylaseihin.

Sovelluksista Androidin mukana tulee ainoastaan muutama oleellisin sovellus; muut tarvitut sovellukset käyttäjä voi ladata esimerkiksi Google Play -mobiiliverkkokaupasta, josta löytyi heinäkuuhun 2014 mennessä yli 1,3 miljoonaa sovellusta [Statista, 2014]. Koska Android on vapaan lähdekoodin projekti, monet puhelinvalmistajista ovat muokanneet Androidista oman versionsa, joka on käännetty puhelimen laitteistoon sopivaksi. Tämä antaa puhelinvalmistajille samalla mahdollisuuden tehdä ja esiasentaa omia allekirjoitettuja sovelluksia puhelimiinsa [Amadeo, 2013]. Toisaalta tämä myös aiheuttaa sen, etteivät kaikki käyttäjät saa uusinta Androidin versiota samanaikaisesti käyttöönsä. Vanhemmissa puhelinmalleissa uusin Androidin versio ei välttämättä edes toimi. Siten markkinoilla on tälläkin hetkellä Androidin eri versioita 2.2-versiosta uusimpaan 5.1-versioon. Tällä hetkellä Android on eniten käytetyin [W3Schools, 2014] ja eniten myydyin käyttöjärjestelmä (Kuva 2). Laitteita, joihin Android on asennettu, on myyty enemmän kuin Windows, iOS ja Mac OS X -laitteita yhteensä [Gartner, 2014]. Koska käyttäjiä on paljon, myös sovelluskehittäjiä ja puhelinmalleja löytyy paljon: Visionmobilen [2013] kehittäjäkyselyn mukaan 71 % mobiilisovelluksien kehittäjistä tekevät sovelluksia Android-alustalle ja eri laitteita oli OpenSignal-yhtiön [2014] mukaan vuonna 2014 jopa 18796. Laitteiden ja versioiden suuri määrä on tällä hetkellä suuri ongelma sovelluskehittäjien kannalta, sillä sovelluksen toimivuuden varmistus kaikissa eri laitteissa ja versioissa on käytännössä mahdotonta.



Kuva 2. Älypuhelimien markkinaosuus vuosina 2006 – 2013. Lähde: [Amadeo, 2013].

2.1. Tietoturva ja yksityisyys

Android-sovellukset ajetaan eristetyssä tilassa, josta niillä ei ole pääsyä systeemin muihin resursseihin, ellei käyttäjä anna erikseen niihin lupaa sovellusta asennettaessa. Käytännössä jokainen sovellus omaa Android-käyttöjärjestelmän näkökulmasta katsottuna oman sovelluskohtaisen käyttäjätilin. Jokaista sovellusta myös ajetaan omassa virtuaalikoneessa (virtual machine), jolloin se on eristettynä muista laitteen sovelluksista. Käyttäjä näkee jo ennen asennusta sovelluskeskuksessa kaikki sovelluksen vaatimat oikeudet. Oikeuksia on useita erilaisia, kuten esimerkiksi täydellinen verkkoyhteys, mahdollisuus muokata tietoja muistikortilla tai lukea käynnissä olevien ohjelmien tietoja. Käyttäjä voi asennuksen yhteydessä joko sallia tai hylätä sovelluksen vaatimat luvat; sovellus asentuu ainoastaan, jos käyttäjä hyväksyy ne. Eristäminen ja lupakäytännöt vähentävät sovelluksen aiheuttamia riskitekijöitä ja virheiden mahdollisuuksia.

Android on kaikista alustoista joustavuutensa ja suosionsa takia myös haittaohjelmien suosiossa. F-Secure [2014] raportoi vuonna 2014 yhteensä 277 erityyppistä uhkaa mobiilialustoille, mistä Android-alustalle oli kohdistunut 275. Haittaohjelmista yleisimpiä Android-laitteissa on niin

sanotut “premium-palvelu väärinkäyttäjät”, jotka lähettävät viestejä maksullisiin palveluihin ja täten aiheuttavat kuluja käyttäjille. Muut haittaohjelmat voivat esimerkiksi näyttää käyttäjälle ei-toivottuja mainoksia tai lähettää käyttäjän arkaluonteisia tietoja ulkopuolisille tahoille. Erityisesti lapset ovat alttiita asentamaan haittaohjelmia sovelluskeskuksesta, sillä lapset eivät välttämättä ymmärrä asennusvaiheessa tapahtuvaa lupien sallimisvaihetta: suurin osa peleistä esimerkiksi ei tarvitse tekstiviestinlähetys-mahdollisuutta.

Google on kehittänyt Google Play -kauppaansa automatisoidun antivirus-järjestelmän: Google Bouncer. Androidin tekniikan varatoimitusjohtaja Hiroshi Lockheimer [2012] ilmoitti Google Bouncerin vähentäneen järjestelmän käyttöönoton alkuvaiheissa jopa 40 % potentiaalisista haittaohjelmista. Tämän lisäksi Android 4.2-käyttöjärjestelmän mukana julkaistiin uusi turvallisuusjärjestelmä, joka skannaa kaikista laitteeseen asennetuista sovelluksista mahdollisia haitallisia ominaisuuksia [Raphael, 2012]. Turvallisuusjärjestelmä osaa myös varoittaa käyttäjää maksullisiin palveluihin lähetettävistä viesteistä, jolloin niiden lähettämisen voi estää salassa käyttäjältä. Nämä takaavat käyttäjälle suhteellisen hyvän turvan mahdollisilta haittaohjelmilta vaikka pienet lapset itsenäisesti niitä asentaisivatkin. Ne eivät silti takaa sitä, että asennetun sovelluksen sisältö olisi soveliasta lapselle, eivätkä myöskään kata monia muita käytönvalvontaan liittyviä osa-alueita.

Toukokuussa 2013 Google julkisti Android Device Manager (ADM) -ohjelmiston, joka sallii käyttäjien paikantaa ja seurata puhelinta sekä pyyhkiä puhelimen muisti etäkäyttöliittymän kautta [Poiesz, 2013]. Täten Android-laitteet on jossain määrin turvattu myös silloin, jos laite sattuu katoamaan tai se varastetaan.

Androidin avoin lähdekoodi sallii puhelinvalmistajien tekevän laitteista entistä turvallisempia. Esimerkiksi Samsung on yhdessä General Dynamicsin kanssa uudelleenkirjoittanut Androidin, mahdollistaen luottamuksellisten tai muuten arkaluoteisten tiedostojen salauksen Knox-ohjelmistollaan. Knox:lla voi pitää esimerkiksi työtiedostoja erillään henkilökohtaisista tiedoista. Puhelimen joutuessa vääriin käsiin kaikki Knox:ssa olevat tiedostot ovat salauksen takana turvassa.

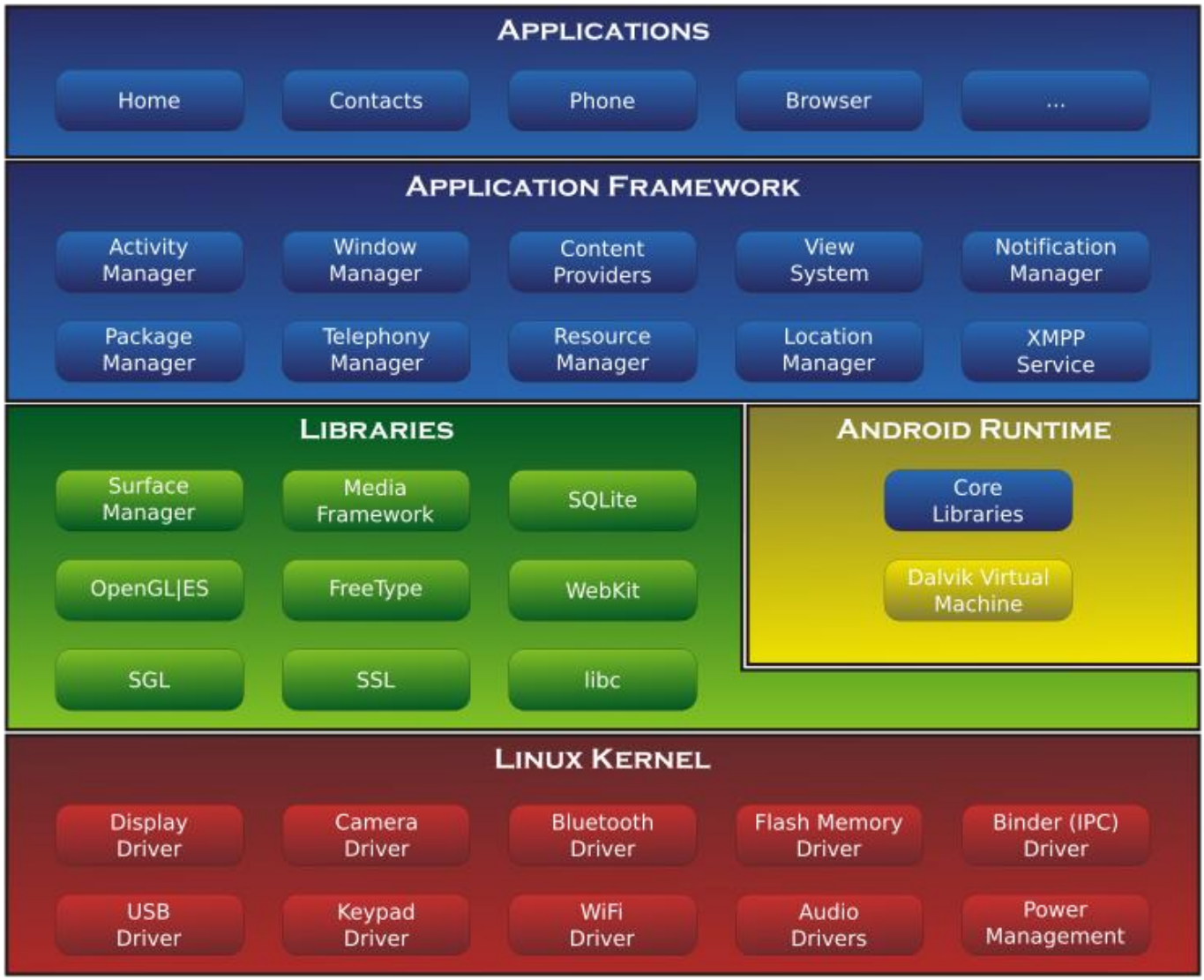
Vuonna 2013 salaisten asiakirjojen vuotojen yhteydessä selvisi, että Amerikan ja Britannian tiedusteluvirastoilla: National Security Agency (NSA) ja Government Communications Headquarters (GCHQ), on ollut pääsy lähes kaikkiin käyttäjän tietoihin iPhone, BlackBerry ja Android -laitteissa. Heillä on todistettavasti ollut pääsy lukea lähes kaikkien laitteiden käyttäjien tekstiviestejä, sijaintitietoja, sähköposteja ja muistiinpanoja [Spiegel, 2013]. Vuotaneet dokumentit paljastivat tiedusteluvirastojen tarkkailevan myös laitteista tehtyjä Google-karttojen hakuja kerätäkseen sijaintitietoja [The Guardian, 2014].

2.2. Androidin Linux-ydin

Androidin ydin pohjautuu yhteen Linux-ytimen pitkäaikaisen tuen (long-term support) haaroista, mutta Android ei kuitenkaan ole Linux. Linuxista poiketen Android-käyttöjärjestelmä ei muun muassa sisällä lainkaan ikkunointijärjestelmää. Huhtikuusta 2014 lähtien Android-laitteet käyttävät

pääosin Linux-ytimen versioita 3.4 tai 3.10. Androidissa on tähän asti kuitenkin ollut useampia Linux-ytimiä versiosta 2.6.25 lähtien, joka oli käytössä Android 1.0:ssa.

Linuxin ytimestä kehitetty Androidin Linux-ydin sisältää arkkitehtuurillisia muutoksia ja useita Androidille kohdennettuja komponentteja, joiden kehitys pidetään erillään Linux ytimen omasta kehityssyklistä Googlen toimesta [Brady, 2008]. Linux valittiin Androidin alustan pohjaksi muun muassa hyvän muistin- ja prosessinhallinnan, avoimen lähdekoodin, jaettujen kirjastojen sekä oikeuksiin pohjautuvan tietoturvamallin vuoksi [Brady, 2008]. Androidin arkkitehtuuri on jaettu viiteen eri kerrokseen: sovellukset, sovelluskehys, kirjastot, ajonaikaiset kirjastot ja Linux-ydin (Kuva 3).

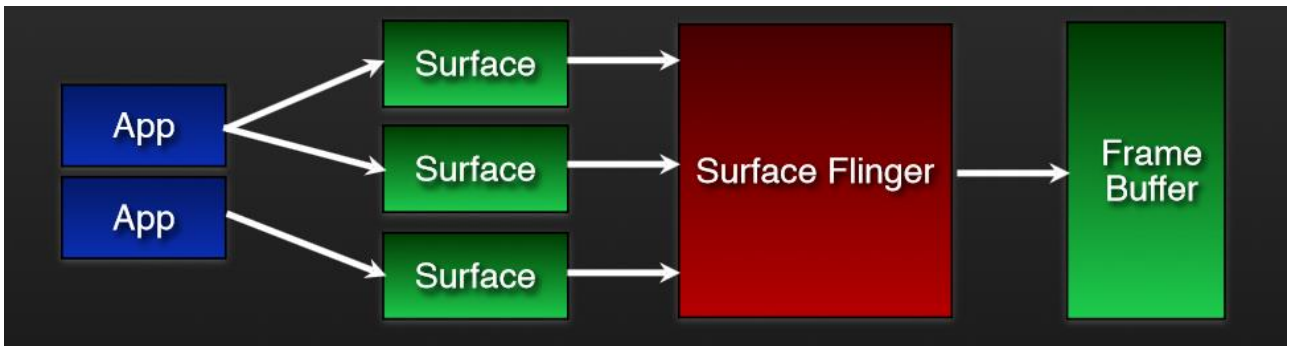


Kuva 3. Androidin anatomia [Brady, 2008].

Linux-ydin toimii Android-käyttöjärjestelmän alimpana kerroksena. Linux-ytimen komponentteihin on tehty useita eri parannuksia, jotka soveltuvat paremmin mobiilialustoille. Yksi uudelleen kirjoitetuista komponenteista on Binder (Inter Process Communication)-ajuri. Binderin tehtävänä on jakaa tietoa ja mahdollistaa kommunikointi eri sovellusten ja palveluiden välillä tehokkaasti ja turvallisesti. Toinen paranneltu komponentti on virranhallinta-komponentti. Virranhallinta on mobiililaitteissa erittäin kriittinen osa-alue, joten se on kirjoitettu toimimaan paljon agressiivisemmin Androidin Linux-ytimessä. Virranhallinnan kautta eri komponenttien on mahdollista kutsua herätyslukkoja (wake locks), joilla voi hallita esimerkiksi näytön ja prosessorin päälläoloa. [Brady, 2008].

Linux-ytimen päällä toimii natiivit kirjastot. Androidin Linux-haaraan on kirjoitettu uudestaan muun muassa koko libc-kirjasto (C-kielen oletuskirjasto). Libc-kirjaston uudelleenkirjoittamiseen oli moniakin syitä. Koska libc-kirjasto latautuu jokaisen prosessin yhteydessä, siitä haluttiin mahdollisimman pieni kooltaan. Mobiililaitteiden heikomman suorituskyvyn vuoksi libc-kirjaston on oltava myös nopea. Libc-kirjastoon haluttiin lisätä myös monia Android-alustalle oleellisia ominaisuuksia, kuten järjestelmäominaisuuksien lukemista ja lokitiedostoon kirjoittamista. Uudellenkirjoitettu libc ei ole yhteensopiva alkuperäisen libc-kirjaston kanssa.

Kuvaa ja ääntä useista eri lähteistä hallitsee niitä varten tehdyt pinnanheittäjä (surface flinger) ja äänenheittäjä (audio flinger). Heittäjät vastaanottavat kuva- tai äänivirrat Binderin kautta ja osavat yhdistää useammastakin lähteestä tulevat virrat näytölle tai äänen ulostuloon (Kuva 4). [Brady, 2008].



Kuva 4. Pintaheittäjän toimintakaavio [Brady, 2008].

Muita Androidille tehtyjä kirjastoja ovat muun muassa vapaaseen lähdekoodiin perustuva WebKit-selain, useiden eri formaatissa olevien kuva-, video- ja äänitiedostojen toistot mahdollistava media-kehys, sekä tietokantaoperaatiot mahdollistava kevyt SQLite-kirjasto. Lisäksi natiiveihin kirjastoihin sisältyy laitteiston abstrahointi-kerros (Hardware Abstraction Layer – HAL). HAL:n tehtävä on määritellä rajapinta, minkä Android vaatii kaikkien laitteistoajurien toteuttavan. [Brady, 2008].

Samalla tasolla natiivien kirjastojen tasolla toimii Androidin ajonaikainen virtuaalikone Dalvik sekä ydinkirjastot. Dalvik mahdollistaa sovellusten toimivuuden useissa eri laitteissa sekä ajonaikaisen yhtenäisyyden. Dalvik kääntää kaikki Android-sovellusten käännetyt Java-luokat (class-tiedostot) dex-formaattiin. Dex-formaatti on optimoitu toimimaan äärimmäisen tehokkaasti Dalvikin tavukooditulkissa. Dalvik tukee myös useiden virtuaalikoneiden yhtäaikaista prosessointia. Ydinkirjastot sisältävät kaikki Java-kielen tutuimmat ja oleellisimmat rajapinnat, muun muassa: tietorakenteet, tiedostojen hallinnan, verkko-operaatiot, grafiikat ja apuohjelmat. [Brady, 2008].

Heti sovellusten alapuolelta löytyy sovelluskehys-taso (Application framework). Kehys sisältää palveluita, jotka ovat olennaisia Android-alustassa. Ne toimivat yleensä taustalla ilman, että sovelluskehittäjien tarvitsee niitä suoraan käyttää. Niiden kautta hallitaan muun muassa ikkunointia, ilmoituksia, sovellusnäkymiä, sovelluspaketteja ja resurssitiedostoja. Esimerkiksi ikkunoinninhallinta hoitaa automaattisesti Android-laitteessa päällä olevien sovellusten järjestystä ja animointia sovellusta avatessa tai suljettaessa sekä näyttöä käännettäessä. Sovelluskehys tarjoaa myös sovelluskehittäjille pääsyn alemman tason laitteistorajapintoihin. [Brady, 2008].

Päällimmäiseksi arkkitehtuurissa rakennetaan kaikki kolmansien osapuolten sovellukset ja Androidiin sisäänrakennetut sovellukset, jotka ovat käyttäjille näkyviä rajapintoja mobiililaitteen laitteistoon.

2.3. Sovelluskehitys

Sovelluksilla voi laajentaa Android-laitteen ominaisuuksia monin eri tavoin. Androidille tehtävät sovellukset kirjoitetaan pääosin Java-ohjelmointikielellä käyttäen hyödyksi Androidin tarjoamaa sovelluskehitysalustaa (SDK). SDK sisältää monia hyödyllisiä kehitystyökaluja muun muassa vianmääritystilan (debugger), ohjelmistokirjastoja, puhelin-emulaattorin, dokumentaation, esimerkkiohjelmia ja opetusohjelmia (tutorial). SDK:n mukana tulevan hallintatyökalun kautta sovelluskehittäjä voi asentaa tuen uusimmille Android-alustan rajapintatasoille (API level). Tämän hetken uusin rajapintataso on 22, joka tukee Android 5.1 -alustan mukana esiteltyjä ominaisuuksia. Android-sovelluskehityksessä on tärkeää tukea myös vanhempia rajapintatasoja, jotta sovellus toimisi mahdollisimman monessa vanhemmassakin puhelimessa. Uusien rajapintatasojen myötä joitakin vanhoista ominaisuuksista saatetaan hylätä (deprecate), jolloin niitä ominaisuuksia käyttävä sovellus on päivitettävä uusien käytäntöjen mukaan. Usein hylätyn ominaisuuden tilalle tulee vaihtoehtoinen tapa tehdä asia, mutta joskus ominaisuus saatetaan hylätä kokonaan esimerkiksi paremman tietoturvan tai yksityisyyden saavuttamiseksi. Yhtenä käytönvalvontasovellusten kehitystä entisestään hankaloittavana esimerkkinä, rajapintataso 21 (Android 5.0) hylkäsi erittäin tärkeän ominaisuuden sovellusten käytönvalvontaan liittyen, josta kerrotaan lisää kappaleessa 3.3.

Monet ohjelmointiympäristöt (IDE) sisältävät tuen Android-sovelluskehitykseen. Android tarjoaa ohjelmointiympäristö IntelliJ IDEA:n päälle kehitettyä Android Studio:ta ilmaiseksi kaikille Android-kehittäjille [Android Studio, 2015]. Android Studio tarjoaa muun muassa Gradle-käännöstyökalun, automaattisen APK-pakettien kääntämisen, sovellusten optimoinnin,

minimoinnin ja allekirjoituksen -hallinnan, sisäänrakennetut Google-palvelut ja näkymien rakennustyökalun. Gradle-käännöstyökalulla ohjelmistokehittäjien on mahdollista automatisoida sovelluskehityksen eri vaiheita testaamisesta sovelluksen paketointiin sekä hallita sovelluksen riippuvuuksia ulkopuolisiin kirjastoihin. Kun sovellus halutaan julkaista, on se allekirjoitettava sovelluskehittäjän omalla sertifikaatilla ja paketoitava oikeantyyppiseksi APK-tiedostomuodoksi. Tämän lisäksi sovellusten käännetty tavukoodi yleensä optimoidaan erilaisilla työkaluilla, joita myös Android Studion paketinrakennus-työkalu tukee. Optimoinnin tarkoituksena on pienentää julkaistavan paketin kokoa, nopeuttaa ajettavan ohjelman toimintoja sekä hankaloittaa käännetyn sovelluksen takaisinmallinnusta (reverse engineering).

Android-projekti voi sisältää neljää erityyppistä komponenttia: aktiviteetit (activity), palvelut (service), sisällöntoimittajia (content provider) ja lähetyksen vastaanottajat (broadcast receiver). Jokainen komponentti voi toimia tietynlaisena sisäänkäyntinä sovelluksen käynnistämiseksi ja jokaisella on oma sovelluksensisäinen elinkaarensa. Aktiviteetti kuvastaa yksittäistä käyttäjälle näkyvää näkymää, jonka kautta käyttäjä voi olla sovelluksen kanssa vuorovaikutuksessa. Yksi näkymä voi olla esimerkiksi käyttäjän luontilomake, johon syötetään uuden käyttäjän tiedot ja lopuksi painetaan tallenna-painiketta. Palvelut ovat käyttäjältä piilossa työskenteleviä pitkään kestäviä operaatioita. Palvelut voivat olla jatkuvasti päällä tai ne voidaan käynnistää toisen komponentin pyynnöstä. Käyttäjän luonnin yhteydessä uusi käyttäjä saatetaan haluta tallentaa taustajärjestelmään, mikä saattaa olla pitkäkestoinen operaatio. Jotta käyttäjän käyttökokemus saadaan pidettyä mahdollisimman sujuvana, ilman pitkiä latausaikoja, halutaan taustajärjestelmään tallennusoperaatiot yleensä suorittaa omassa palvelussaan. Palveluiden kautta voi ohjata käyttäjälle näytettävää aktiviteettia, riippuen esimerkiksi tallennuksen onnistumisesta. Sisällöntoimittajan kautta voi hallita tiedon tallentamista ja lukemista Android-järjestelmässä. Tallennus voi tapahtua esimerkiksi SQLite-tietokantaan, tiedostojärjestelmään, Internetiin tai mihin tahansa muuhun tallennusjärjestelmään. Android-järjestelmä tarjoaa esimerkiksi sisällöntoimittajan, jonka kautta hallitaan laitteen käyttäjän yhteystietoja. Sen kautta kaikki riittävät oikeudet omaavat muut sovellukset voivat hakea yhteystietoja tai jopa tallentaa uusia.

Android-järjestelmän eräs yksilöllisistä ominaisuuksista on, että siinä mikä tahansa sovellus voi käynnistää toisen sovelluksen komponentteja. Ohjelmistokehittäjän ei tarvitse toteuttaa omaa kamerasovellusta, vaikka sovelluksessa pitää pystyä ottamaan valokuvia. Sen sijaan, sovellus pystyy kutsumaan suoraan puhelimeen asennetun kamerasovelluksen aktiviteettia, joka palauttaa otetun kuvan sovelluksen käyttöön. Käyttäjän näkökulmasta kuitenkin näyttää siltä, että kamera olisi osa käytettyä sovellusta. Kun järjestelmä käynnistää komponentin, se käynnistää prosessin komponentin omistavalle sovellukselle ja alustaa luokat, joita komponentti tarvitsee. Toisin kuin useimmissa muissa järjestelmissä, Android-sovelluksilla ei siis ole vain yhtä käynnistyspistettä, josta sovelluksen komponentteja voi käynnistää. Koska Androidissa kaikki sovellukset ajetaan omina prosesseinaan ja omilla oikeuksillaan, eivät toiset sovellukset voi suoraan kutsua toisen sovelluksen komponenttia. Android-järjestelmä kuitenkin voi. Sovelluksen halutessa aktivoida toisen sovelluksen komponentti on ensimmäisen sovelluksen välitettävä tarkoitus-viesti (intent

message) Android-järjestelmälle, millä kerrotaan, että on tarkoitus käynnistää toisen sovelluksen tietty komponentti. Neljästä erityyppistä komponentista kolmea komponenttia (aktiviteetti, palvelu ja lähetyksen vastaanottaja) voi aktivoida käyttöön asynkronisilla tarkoitus-viesteillä. Tarkoitus-viestit kytkevät kaikki yksittäiset komponentit toisiinsa ajonaikaisesti huolimatta siitä, kuuluuko komponentti omaan sovellukseesi vai toisen sovellukseen. Tarkoitus-viestin voi määrittää joko eksplisiittiseksi tai implisiittiseksi. Eksplisiittinen viesti kertoo tarkalleen, mikä komponentti halutaan aktivoida. Oman sovelluksen sisällä olevia komponentteja usein kutsutaan eksplisiittisillä tarkoitus-viesteillä. Implisiittisellä viestillä voi määritellä, minkä tyyppinen komponentti halutaan aktivoida. Kolmannen osapuolen tarvitessa kamerasovellusta lähetetään yleensä implisiittinen tarkoitus-viesti, jolla kerrotaan, että halutaan saada kamera-komponentti aktivoitua. Tämän saatuaan Android-järjestelmä katsoo onko laitteeseen rekisteröitynä tarkoitus-viestin tarkoittamia komponentteja. Mikäli komponentteja löytyy, Android käynnistää niistä käyttäjän määrittämän oletussovelluksen tai oletuksen puuttuessa antaa käyttäjälle mahdollisuuden valita haluamansa sovellus. Tarkoitus-viestillä aktiviteettia tai palvelua kutsuttaessa, pitää määritellä haluttu toimenpide. Lisäksi viestillä voi välittää resurssin osoitteen (URI), minkä pohjalta toiminto halutaan suorittaa. Osoite voi määrittää esimerkiksi kuvan sijainnin, yhteyshenkilön tai Internet-osoitteen. Joissakin tapauksissa voidaan haluta vastauksena tarkoitus-viesti, joka voi sisältää esimerkiksi osoitteen puhelinluettelosovelluksesta valittuun yhteyshenkilöön. Sisällöntoimittajat ovat ainoita komponenttityyppejä, joita ei voida aktivoida tarkoitus-viesteillä. Sen sijaan, sisällöntoimittaja-komponentti aktivoituu saatuaan pyynnön ContentResolver-luokalta. Siten sisällöntoimittajia ei tarvitse ikinä kutsua suoraan, vaan ContentResolver toimii abstrahointi-kerroksena sisällöntoimittajan ja sen käyttäjän välillä parantaen samalla tietoturvaa.

Ennen kuin Android-järjestelmä voi käynnistää sovelluksen yhtäkään komponenttia, on jokaisen sovelluksella oltava tietynlainen manifesti-tiedosto. Manifesti on XML-muodossa ja siinä voi määritellä useita sovelluksen erityispiirteitä, joista jotkin ovat pakollisia jotkin vapaaehtoisia. Manifestin tulee aina sisältää lista jokaisesta sovelluksen komponentista seuraavalla tavalla:

```
<activity android:name=”com.example.project.ExampleActivity”
        android:label=”@string/example_label” …>
</activity>.                                                    (2.1)
```

Jokaiselle komponentille voi myös määrittää monenlaisia erityispiirteitä suoraan manifestiin. Erityispiirteitä on esimerkiksi aktiviteetin värimaailma tai teema, takaisinpainikkeen käyttäytymismalli ja käynnistymistila. Lisäksi manifesti voi pitää sisällään muun muassa kaikki sovelluksen tarvitsemat oikeudet, minimirajapintatason, laitteistovaatimukset, ulkopuoleiset kirjastoriippuvuudet ja monia muita määrityksiä. Sovellus, joka tarvitsee toimiakseen kameran ja Android-version 2.1 tai suuremman, määriteltäisiin seuraavalla tavalla:

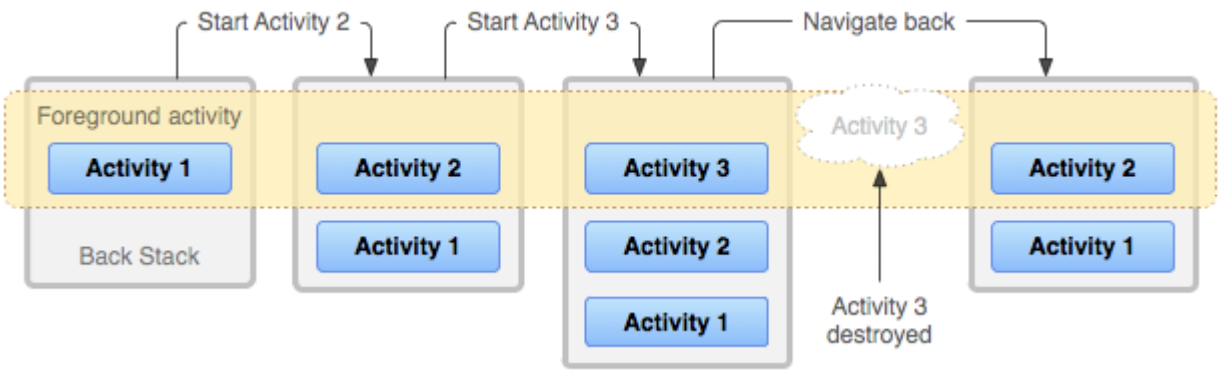
```
<uses-feature android:name=”android.hardware.camera.any”
        android:required=”true” />
```

</uses-sdk android:minSdkVersion=”7” android:targetSdkVersion=”21” />. (2.2)

Android-sovellusten yksi erityispiirteistä työpöytäsovelluksiin verrattuna on lopetus-painikkeen puuttuminen. Lopetuspainikkeen saa kyllä toteutettua mihin tahansa Android-sovellukseen, mutta yleinen käytäntö on, ettei sitä tarvita. Koska Android-järjestelmässä ei ole ikkunointia, on täytynyt kehitellä toisenlainen tapa hallita sovellusten näyttämistä käyttäjälle.

Sovellus sisältää yleensä useita aktiviteetteja. Jokainen aktiviteetti pitää suunnitella siten, että siirtymät muihin aktiviteetteihin ovat loogisia. Esimerkiksi sähköposti-sovelluksen yksi aktiviteetti näyttää kaikki saapuneet sähköpostit. Kun käyttäjä valitsee yhden sähköposti-viesteistä, avataan uusi aktiviteetti, joka näyttää valitun viestin sisällön. Kuten aikaisemmin mainittiin, aktiviteetti voi avata aktiviteetteja myös toisista sovelluksista. Vaikka aktiviteetteja voidaan näyttää useista eri sovelluksista, Android näyttää siirtymät sulavasti ja pitämällä aktiviteetit saman työn (task) sisällä. Työ on kokoelma aktiviteetteja, joiden kanssa käyttäjä toimii tehdessään jotain tiettyä tehtävää. Työn sisällä olevat aktiviteetit järjestetään pinoon (back stack) siinä järjestyksessä, missä käyttäjä on niitä tarkastellut.

Laitteen kotinäkymä on aloituspiste useimmille töille. Kun käyttäjä koskettaa sovelluksen käynnistyskuvaketta, kyseisen sovelluksen työ tulee etualalle. Kyseisellä sovelluksella ei välttämättä ole olemassa olevaa työtä, mikäli sovellusta ei ole käytetty hetkeen. Tällöin luodaan uusi työ ja käyttäjälle näytetään sovelluksen pää-aktiviteetti. Kun aktiviteetista käynnistetään toinen aktiviteetti, uusin tulee pinon päällimmäiseksi ja näkymä näytetään käyttäjälle. Edellinenkin aktiviteetti pysyy pinossa, mutta sen toiminta on pysäytetty. Järjestelmä pitää aktiviteetin nykyisen tilan tallessa vaikka se olisi pysäytettynä. Käyttäjän painaessa takaisin-painiketta, päällimmäisin aktiviteetti tuhotaan ja aiempi aktiviteetti jatkaa siitä mihin se oli jäänyt. Pinossa olevia aktiviteetteja ei ikinä järjestetä uudelleen; ainoastaan poistetaan tai lisätään pinon päältä. Pino siis toimii viimeksi sisään, ensiksi ulos -periaatteella (Last In First Out). Kuva 5 visualisoi tätä yhden työn sisällä tapahtuvaa aktiviteettien välistä käyttäytymismallia ajan eri hetkillä.



Kuva 5. Aktiviteettipinon toimintakaavio. [Android Back Stack, 2015].

Jos takaisin-painiketta painetaan niin kauan, että työn aktiviteettipinosta lähtee viimeinenkin aktiviteetti, näytetään Androidin kotinäkymä tai se työ, josta edellinen työ käynnistettiin. Jos sovellus mahdollistaa saman aktiviteetin käynnistämisen kahdesta eri aktiviteetista, luodaan kyseisestä aktiviteetista oletuksena aina uusi ilmentymä. Samaa aktiviteettia ei tuoda pinon alta päällimmäiseksi, koska aktiviteetteja ei ikinä järjestetä uudelleen pinon sisällä. Täten samasta aktiviteetista voi olla useita eri ilmentymiä saman työn tai useammankin työn sisällä. Tällöin takaisin navigoitaessa sama aktiviteetti saattaa tulla eri tilassa useaan kertaan käyttäjälle näkyviin. Tätä toimintamallia voi kuitenkin muokata esimerkiksi manifesti-tiedostoon määritettävään komponenttien erityispiirteistä löytyvään käynnistymistila-muuttujaan. Tämän hallitseminen on erittäin tärkeää, mutta myös monimutkaista, Android-sovelluksen siirtymien toteuttamisessa loogisesti.

3. Käytönvalvonta

3.1. Menetelmät

Käytönvalvonnan kohteita ja käyttötapoja on monia. Käytönvalvontaa voidaan tarvita kaupoissa esilläoleviin puhelimiin ja taulutietokoneisiin, koulujen tietokoneisiin tai perheiden medialaitteisiin. Tämä tutkielma keskittyy ainoastaan perheen käytössä olevien Android-käyttöjärjestelmällä toimivien puhelinten ja taulutietokoneiden käytönvalvontaan. Myös menetelmiä käytönvalvonnan toteuttamiseksi on useita. Mobiililaitteissa oleellisimpia ovat käyttöajan ja sovellusten rajaaminen, Internet-suodatus, sijaintitietojen tallennus sekä puheajan ja tekstiviestien rajoittaminen.

Lapsen ruutuajan rajoittaminen on varmasti monelle vanhemmalle ongelmallista. Television ja tietokoneiden ruutuaikaa on suhteellisen helppoa valvoa, mutta lapsen voi silti olla vaikea irrottautua pelin tai elokuvan otteesta. Puhelinta ja taulutietokonetta voi kuljettaa mukanaan paikkaan, jossa vanhemmat eivät pääse valvomaan sen käyttöä. Jotkin sovelluksista voivat olla sellaisia, joita lapsi ei saa esimerkiksi ikänsä puolesta käyttää, kun toisiin sovelluksiin halutaan asettaa aikarajoitus.

Internet-suodatuksella halutaan varjella lapsia pääsemästä sivustoille, jotka sisältävät lapsille sopimatonta tai muuten haitallista sisältöä. Sopimaton sisältö voi olla esimerkiksi alastonkuvat, joihin saattaa törmätä haluamattaan hyvinkin helposti nykyaikana. Internet sisältää myös monia haitallisia sivustoja, jotka saastuttavat käyttäjän tietokoneen viruksilla.

Sijaintitietojen tallennus on hyödyllinen ominaisuus monellakin tapaa. Hävinneen puhelimen voi löytää helpommin siitä huolimatta, että puhelin olisi esimerkiksi pudotessaan sammunut. Vanhempi voi mennä paikanpäälle hakemaan lasta, mikäli häntä ei jostain syystä ole saanut tavoitettua puhelimella. Hienostuneimmissa sovelluksissa voisi olla myös ominaisuus, joka antaa hälytyksen mikäli lapsi menee määritetyn alueen ulko- tai sisäpuolelle.

Puheaikaa, tekstiviestien määrää ja tiedonsiirtomääriä vanhemmat voivat haluta tarkkailla, jotta puhelinlaskut eivät kasvaisi liian suuriksi. Nykyisten viestinohjelmien ja sosiaalisen median avulla on hyvin helppoa valikoida ihminen kenelle lähettää viestiä huolimatta siitä, tunteeko viestin lähettäjä vastaanottajaa. Näitä menetelmiä hyödyntäen seksuaalirikoksia tekevät voivat houkutella uhrin luokseen. Vanhemmat voivat siis halutessaan saada ilmoituksen aina, kun tuntemattomasta numerosta tulee soitto tai tekstiviesti. Jotkut käytönvalvontasovellukset saattavat jopa lähettää tekstiviestien sisällöt vanhemmille luettavaksi. Erillisten viestinohjelmien keskustelujen seuraaminen kolmannen osapuolen sovellukselta voi olla haastavaa tai jopa mahdoton toteuttaa.

3.2. Nykytila

Käytönvalvonnan tuki missään medialaitteessa on vielä hyvin suppeaa. Uusimmista televisioista löytyy usein mahdollisuus lukita televisio numerokoodin taakse, jolloin ainoastaan koodin tietävä saa käytettyä televisiota. Microsoft kehitti Windows-käyttöjärjestelmäänsä käytönvalvonnan tuen vasta Windows Vistaan, joka julkaistiin vuonna 2007 [Windows Vista, 2007].

Android-taulutietokoneet saivat tuen rajoitetuille käyttäjätileille heinäkuussa 2013 julkaistussa versiossa 4.3 [Android 4.3, 2013]. Rajoitetut käyttäjätilit sisältävät mahdollisuuden asettaa suppeat käytönvalvonnan ominaisuudet uusille laitteeseen lisättäville käyttäjille. Jokainen rajoitettu tili tarjoaa eristetyn ja turvallisen tilan omilla tallennustilalla, kotinäkymillä ja asetuksilla. Jokainen profiili luodaan pääkäyttäjän toimesta ja rajoitetut profiilit sisältävät samat sovellukset, mitä pääkäyttäjällä on asennettuna. Pääkäyttäjällä on kuitenkin mahdollisuus hallita yksitellen, mitä asennettuja sovelluksia on mahdollista käyttää ja pääsy pääkäyttäjän tiliin on oletuksena estetty. Sovellukset, jotka tarvitsevat pääkäyttäjän tiliä esimerkiksi kirjautumiseen, voi pääkäyttäjä erikseen sallia ne sovellus- ja käyttäjäkohtaisesti. Ohjelmistokehittäjille on tarjottu mahdollisuus toteuttaa omia sisältö- ja ominaisuusrajoituksia, jotka näytetään pääkäyttäjälle rajoitettujen käyttäjätilien asetusnäkymässä. Sovelluskohtaiset rajoitusasetukset voivat olla esimerkiksi valintaruutuja tai alasvetovalikoita. Sovellusta ajettaessa rajoitetussa tilassa, ohjelman on mahdollista tarkistaa pääkäyttäjän profiilille asettamia rajoituksia ja käyttäytyä sen mukaisesti. Esimerkiksi videopalveluun voisi olla mahdollisuus asettaa ikärajoitus, jolloin videoista tarjotaan vain valitulle iälle sopivia videoita. Mikäli sovellus ei toteuta rajoitusasetuksia, pääkäyttäjän on mahdollista estää tai sallia sovelluksen käyttö rajoitetuissa profiileissa. [Android 4.3, 2013]. Ominaisuus on askel oikeaan suuntaan, mutta ihmisten monet eri käyttötavat huomioiden se ei ole vielä tarpeeksi joustava ja monipuolinen. Ohjelmistokehittäjille annettu mahdollisuus tarjota asetukset sisällön rajaamiseksi rajoitetuille käyttäjille mahdollistaa kuitenkin jo paljon.

Android-puhelimet saivat tuen luoda uusia käyttäjiä vasta vuonna 2014 marraskuussa julkaistun Android Lollipop:n myötä [Android 5.0, 2014], mutta rajoitettuja käyttäjätilejä ei ole vielä tähän päivään mennessä julkaistu. Samsungin Android-variantit eivät ole saaneet uusien käyttäjien luontimahdollisuuttakaan vielä. Uusimmista Android-versioista kuitenkaan harvoin tehdään omaa versiota vanhemmille puhelinmalleille, joihin on asennettuna valmistajakohtainen Android-variantti.

Play-kauppapaikasta löytyy tällä hetkellä 97 käytönvalvontaan liittyvää kolmannen osapuolen kehittämää sovellusta. Suurin osa sovelluksista keskittyy ainoastaan yhteen käytönvalvonnan osa-alueeseen, mutta joistakin löytyy useampia ominaisuuksia. Haittapuolena esimerkiksi turvalliseen Internet-selaamiseen tehdyssä sovelluksessa on sen helppo kierrettävyys. Android-laitteissa ei ole mahdollista tällä hetkellä poistaa esiasennettuja sovelluksia, joihin sisältyy muun muassa Internet-selain. Lapsi voi siis halutessaan käyttää selainta, joka sallii pääsyn kaikille sivustoille ilman, että vanhempi saa koskaan tietää asiasta.

3.3. Haasteet

Useimmat Androidille toteutettavan käytönvalvontasovelluksen haasteista johtuvat alustan puutteista tai rajoituksista, mutta osa myös yleisten standardien puutteesta. Monet rajoitukset johtuvat siitä, että käyttäjien yksityisyyttä pyritään suojelemaan. Esimerkiksi rajapintatasoa 21 edeltävät rajapintatasot mahdollistivat kolmansien osapuolten sovellusten hakea listan käyttäjän viimeisimmistä avoinna olleista sovelluksista. Rajapintatason 21 myötä ominaisuus ei ole enää

käytössä, koska se saattaa vuotaa sitä käyttävälle sovellukselle henkilökohtaista tietoa käyttäjästä [Android 5.0, 2015]. Android 5.0:n [2015] muutoksista kertovalla sivustolla mainitaan kuitenkin, että tilalle tuli toinen menetelmä, jolla sovelluskehittäjä voi hakea suppeamman listan viimeisimmistä käytössä olleista sovelluksista. Tämän ominaisuuden hylkäämisen myötä kolmannen osapuolen käytönvalvontasovelluksiin käyttö- ja sovellusrajoitusten toteutus on käytännössä mahdotonta. Käyttö- ja sovellusrajoituksilla vanhempi voisi määrittää lapselle rajat siihen, mitä sovelluksia lapsi voi käyttää ja kuinka kauan. Kun käytönvalvontasovellus ei voi lukea viimeisintä aktiivisena olevaa sovellusta, on mahdoton mitata sen käyttöaikaa tai tietää, onko se sallittujen sovellusten listalla. Vastaavanlaisia haasteita on myös muissakin puhelinalustoissa, mutta Windows-käyttöjärjestelmä mahdollistaa ominaisuuden työpöytäsovelluksille [MSDN, 2015].

Yksi haastavimmista ominaisuuksista, mille tahansa laitteelle, on toteuttaa toimiva Internet-suodatus. Monilla vanhemmilla voi olla erilaisia näkemyksiä siitä, mitä eivät halua lastensa löytävän Internetistä. Yleisimpiä aihealueita, joita halutaan rajoittaa, on muun muassa väkivalta, kiroilu ja alastomuus. Myös huumeita, aseita, uhkapelejä ja monia muita aiheita käsitteleviä sivustoja halutaan rajoittaa lapsilta. Haitallisten sivustojen tunnistaminen täydellä varmuudella on kuitenkin mahdotonta, vaikka suuri osa haitallisista sivustoista olisikin onnistuttu suodattamaan. Monet Internet-suodatus -ohjelmistot on toteutettu käyttämällä laajoja tietokantoja haitallisista sivustoista. Tietokanta sisältää sivuston osoitteen sekä aihealueen ja ohjelma estää sivustolle pääsyn, mikäli käyttäjä on valinnut sivuston aihealueen haitalliseksi. Tämän menetelmän haasteena on saada pidettyä tietokanta tarpeeksi kattavana ja ajantasaisena. Tietokanta saattaa myös joutua kovalle kuormalle, mikä puolestaan näkyy käyttäjälle hidastuneen käyttökokemuksena. Toinen lähestymistapa on määrittää lista kiellettyjä sanoja, joiden esiintymisiä ladattavilta sivustoilta tarkkaillaan. Sivun lataaminen estetään, mikäli kielletty sana löytyy sieltä. Tämän menetelmän huono puoli on, että se saattaa herkästi estää sallittujakin sivustoja. Pääkäyttäjillä (vanhemmilla) on kuitenkin usein mahdollisuus lisätä yksittäisiä sivustoja ”sallitut sivustot”-listalle, joita ei erikseen tarkisteta. Kolmas tapa, joka on myös prototyypissä käytössä, on jättää suodatus täysin vanhempien vastuulle. Vanhemmat voivat joko oletuksena estää kaikki sivut ja määritellä erikseen sallitut sivustot tai sallia kaikki ja erikseen estää kielletyt sivustot. Mobiililaitteissa Internet-suodatuksen toteutusta hankaloittaa kolmansien osapuolien sovellusten rajalliset oikeudet tarkastella Internet-liikennettä. Vanhemmat saattavat haluta sallia Google Play -mobiiliverkkokaupan käytön lapsillensa, mutta haluavat rajoittaa/tarkastella asennettavia sovelluksia. Kun lapsi pyrkii asentamaan sovellusta, siitä voisi lähteä pyyntö vanhemman puhelimeen, jolloin vanhempi voi joko sallia tai hylätä toimenpiteen. Tämänlaisen ominaisuuden toteuttaminen nykyisellään ei ole mahdollista, koska Google Play -sovelluksen Internet-liikennettä ei voi lukea.

Family Online Safety instituutin (FOSI) 90-luvun lopulla perustama ICRA (Internet Content Rating Association) [2015] on sisällönkuvausjärjestelmä, jolla pyrittiin standardisoimaan Internet-sivujen sisällön luokituksen määrittäminen. Sen tarkoitus oli sallia ohjelmoijille mahdollisuuden luokitella tekemiensä sivustojen sisällön kategorian. Kategorioita oli muun muassa alastomuus, seksi, väkivalta, sosiaalinen media ja muita haitalliseksi koettuja aiheita. Sisällön tuottajan lisäksi

sivustolla kävijät voivat arvioida sivuston sisällön laadun. Tämän myötä käytönvalvontasovellusten olisi ollut helppo tarkistaa kaikkien sivustojen luokitus ja tehdä ratkaisu suodatuksen tarpeesta sen perusteella. Vuonna 2008 projektin rahoittaja FOSI kuitenkin päätti lakkauttaa ICRA:n toiminnan [Archer, 2009]. ICRA-järjestelmän kehittäjiin kuuluva Phil Archer [2009] julkaisi kirjoituksen siitä, miksi järjestelmä ei ollut menestys. Kaikkein suurin syy menestymisen esteenä Archerin [2009] mukaan oli se, ettei kukaan halunnut lisätä luokituksia omille sivustoilleen, koska siitä ei ollut mitään hyötyä. Vaikka luokituksia olisi lisättykin, niin niiden lukeminen ja sivustojen suodatus luokitusten perusteella oli hankalaa vanhemmille. Lisäksi, vaikka puolet koko Internetin sisällöstä olisi luokiteltukin ja suodattimia olisi käytetty, ei järjestelmä siltikään olisi ollut täydellinen suoja lapsille haitallisista sivustoista.

Käytönvalvontaa tarvitaan monen eri-ikäisen lapsen laitteissa, mikä tuottaa erinäisiä haasteita. Vanhemmilla lapsilla saattaa olla taitoja poistaa käytönvalvontasovellus käytöstä tai kiertää se muilla keinoilla. Sovellus pitää pystyä suojaamaan siten, että sen poistaminen on mahdollista vain riittävillä oikeuksilla, sovelluksen on käynnistyttävä aina järjestelmän käynnistymisen yhteydessä ja mahdollisen kaatumisen jälkeen automaattisesti. Pienemmät lapset saattavat erehdyksessä soittaa jollekin tai saavat poistettua tärkeitä tietoja näppäillessään sattumanvaraisesti laitetta. Jotkin laitteet saattavat olla useamman henkilön käytettävissä, jolloin rajoitukset pitää pystyä joustavasti säätämään käyttäjän mukaan. Tämän kaltainen toiminnallisuus vaatii mahdollisuuden luoda käyttäjätilejä laitteeseen. Laite tulee olla lukittuna, kunnes käyttäjätili on valittu ja tarvittava käytönvalvontataso asetettu toimimaan. Android 5.0:aa [2014] edeltävissä versioissa käyttäjätilien luonti ei ole tuettuna, joten se pitäisi toteuttaa kolmannen osapuolen sovellukseen. Täten lukitusruutukin pitää tulla kolmannen osapuolen sovelluksesta, joka sotii Androidin sovelluskehittäjien sopimusehtoja vastaan [Android Terms and Conditions, 2012]. Androidin sopimusehdoissa [Android Terms and Conditions, 2012] mainitaan muun muassa, etteivät sovellukset saa estää muita sovelluksia toimimasta tai puuttua toisten sovellusten verkkoliikenteeseen. Tästä huolimatta Googlen Play -mobiiliverkkokaupasta löytyy käytönvalvontaan liittyviä sovelluksia jotka näin tekevät.

4. Tavoitteet

Tutkielman tavoitteena on eritellä käytönvalvontaohjelmiston oleellisimmat osa-alueet ja niiden mahdollistaminen Android-laitteissa mahdollisimman tehokkaasti. Tehokkuudella tarkoitetaan tässä tutkielmassa kolmea asiaa: toimivuus, suorituskyky ja joustavuus. Tärkein tehokkuuden mittari tämän tutkielman tavoitteista on toimivuus. Esimerkiksi toimiva Internet-suodatus suoriutuisi tehtävästä niin hyvin, ettei lapsi pääsisi ikinä vahingossakaan haitallisille Internet-sivuille. Täysin toimivan Internet-suodatuksen teon mahdottomuudesta kerrottiin enemmän kappaleessa 3.3. Suorituskyvyltään hyvän sovelluksen virrankulutus on pientä eikä se hidasta muiden sovellusten suoritusta ollessaan päällä taustalla. Joustavuudella tarkoitetaan sitä, että sovellusta pystyy säätämään asetuksilla moneen eri tarpeeseen. Esimerkiksi hyvä käytönvalvontasovellus olisi säädettävissä toimivaksi pienemmille ja isommille lapsille sekä laitteissa, joita käyttää koko perhe.

Ominaisuuksien mahdollistamisen lisäksi pyritään vertailemaan eri toteutustapojen hyviä ja huonoja puolia. Jotkin käytönvalvontasovelluksen ominaisuuksista on mahdollista toteuttaa useammalla eri tavalla, joista on pyritty löytämään tehokkuuden lisäksi käyttäjäystävällinen tapa toteuttaa se. Mikäli tehokkuus kärsii käyttäjäystävällisyydestä, valinta kallistuu tehokkuuteen.

Lopuksi pohditaan myös, millä tavoin eri osa-alueissa kohdatut haasteet saisi tulevaisuudessa mahdollistettua kolmannen osapuolen ohjelmistokehittäjille vaarantamatta käyttäjän tietoturvaa tai yksityisyyttä.

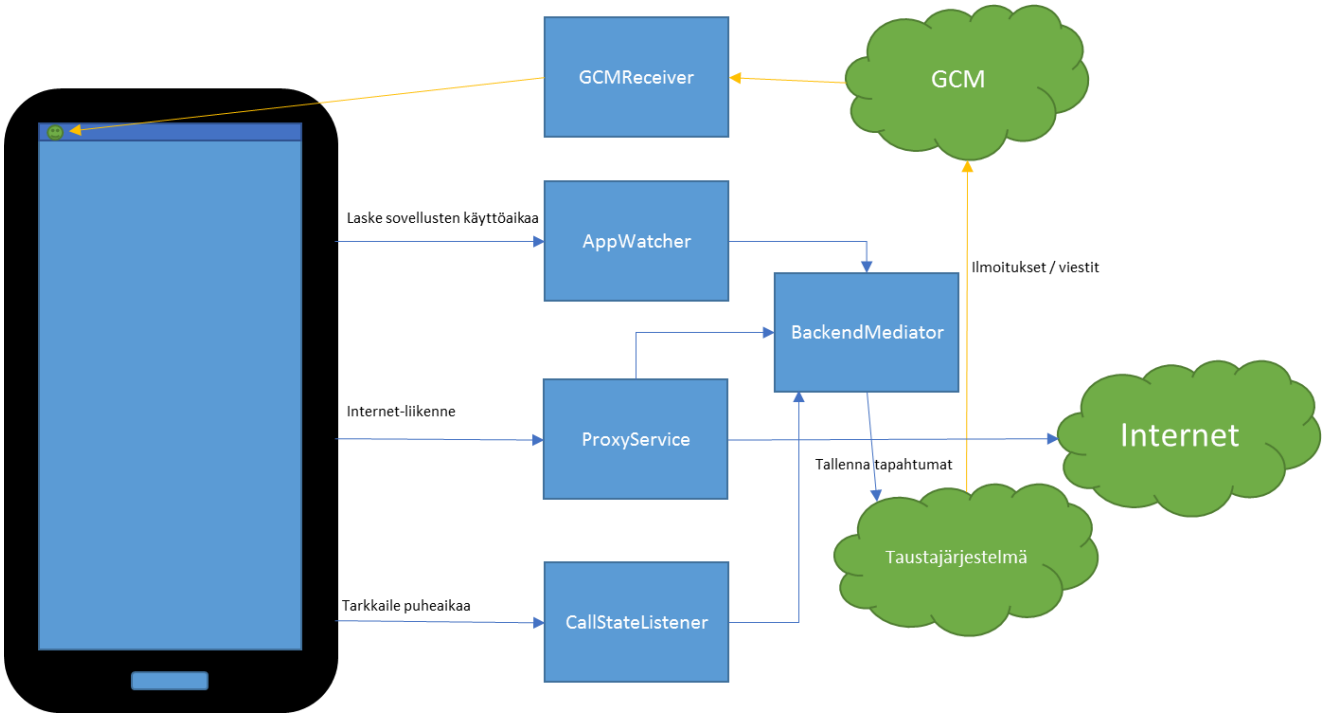
5. Prototyypin toteutus

Käytönvalvontasovelluksen prototyyppi on toteutettu käyttämällä IntelliJ Idea ohjelmointiympäristöä ja testattu toimivaksi Androidin versioilla 4.2 – 4.4. Kolmannen osapuolen kirjastoja prototyypissä on käytössä kolme: Gson [2015], LittleProxy [2011] ja Acra [Gaudin, 2015]. Gson-kirjaston avulla JSON-formaatissa olevasta tekstistä saa muunnettua helposti Java-olioita ja päinvastoin. Toisin kuin web-sovelluksilla, mobiilisovelluksissa tapahtuvia virhelokeja ei oletuksena pysty seuraamaan. Acra-kirjaston avulla sovellus lähettää virheraportteja poikkeustilanteissa taustajärjestelmään, josta virheiden havaitseminen tulee helpommaksi. Välityspalvelin on toteutettu käyttämällä kolmannen osapuolen Java-projektia LittleProxy. Sisäänrakennettua välityspalvelinta käytetään prototyypin Internet-historian tallentamiseen ja Internet-liikenteen suodattamiseen.

Prototyyppi on mobiilisovellus, joka keskustelee taustajärjestelmän kanssa. Taustajärjestelmä on Grails-kehyksellä ohjelmoitu verkkopalvelu, joka tarjoaa mobiilisovellukselle REST-rajapinnan (Representational state transfer) tietojen vastaanottamiseksi. Taustajärjestelmälle on myös mahdollista tehdä käyttöliittymä, josta voisi hallita mobiilisovellusten asetuksia ja valvoa laitteiden käyttöä. Taustajärjestelmän pääasiallisena tehtävänä on vastaanottaa kaikki mobiilisovelluksiin syötetyt ja sen keräämät tiedot. Tiedon koonnille taustajärjestelmään on useita syitä. Kun kaikki tieto löytyy verkosta, on vanhempien mahdollista seurata ja selata lasten mobiililaitteiden käyttötietoja lähes reaaliaikaisesti suoraan taustajärjestelmän omasta käyttöliittymästä. Mobiililaitteiden tallennuskapasiteetti on paljon rajallisempi, kuin palvelinten. Kun tieto siirretään mobiililaitteesta palvelimelle, voidaan mobiililaitteessa säilyttää vain viimeaikainen tieto ja tarpeen vaatiessa hakea palvelimelta aikaisempaa tilastoa. Kun perheeseen kuuluu useita eri mobiililaitteita, voi olla tarpeellista tietää lapsen käyttöaikaa muissa laitteissa, jotta eivät vanhempien asettamat käyttöajat pääse ylittymään.

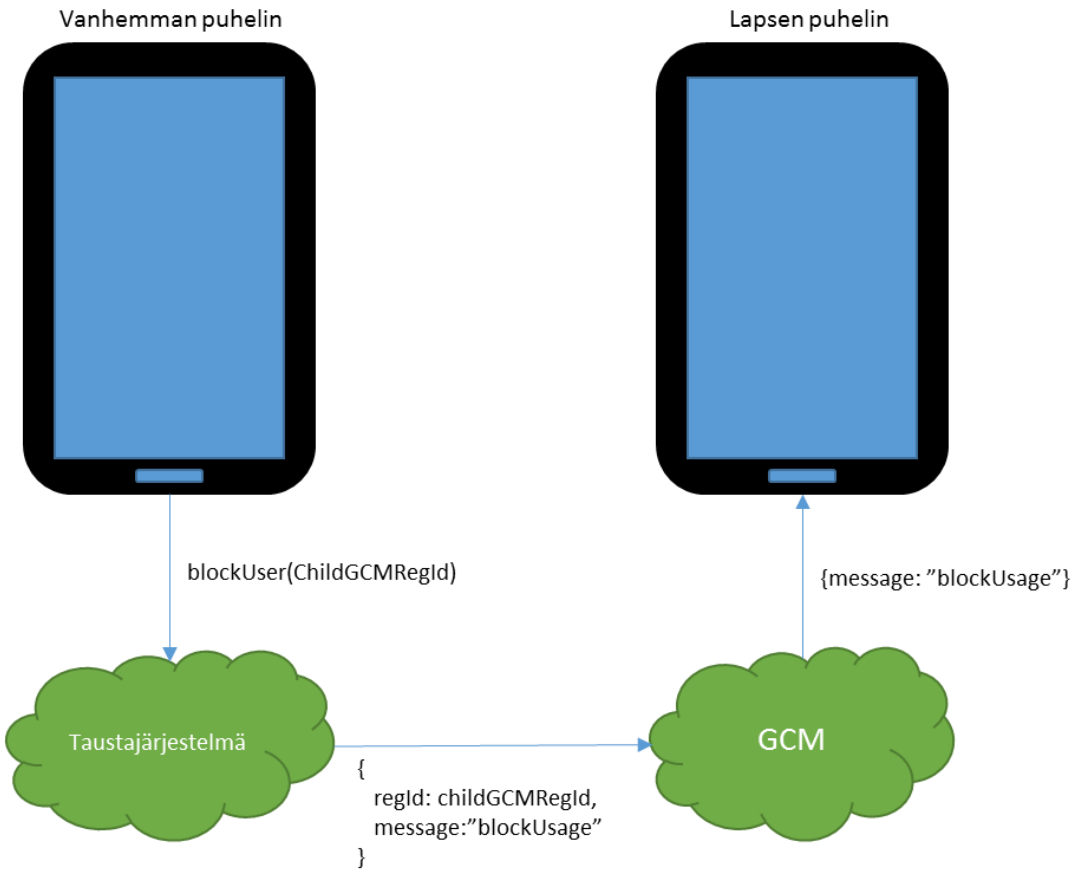
5.1. Arkkitehtuuri

Android-sovelluskehityksessä arkkitehtuuri on lähes aina pakotettu tekemään MVC-mallin (Model-View-Controller) mukaan. Näiden komponenttien lisäksi sovellusten oleellisimpia osia ovat taustapalvelut (background services). Palvelut voidaan joko herättää toimintaan tiettyjen tapahtumien seurauksena tai ne voivat olla jatkuvasti päällä puhelimen taustalla. Prototyyppi sisältää useamman taustapalvelun, joista osa on jatkuvasti päällä ja osa herätettäviä. Kuva 6 näkyy osa taustapalveluista ja niiden roolista sovelluksessa.



Kuva 6. Prototyypin korkean tason arkkitehtuurikaavio.

GCM:n (Google Cloud Messaging) kautta taustajärjestelmä voi lähettää lyhyehköjä sanomia suoraan puhelimeen. Tämä antaa vanhemmille esimerkiksi mahdollisuuden lopettaa lapsen puhelimen käytön etänä suoraan omasta puhelimestaan. Kuva 7 esittää GCM-viestin kulun vanhemman puhelimesta lapsen puhelimeen. Ensin vanhemman puhelimesta lähtee tavallinen http-pyyntö taustajärjestelmään. Pyyntö sisältää lapsen puhelimen GCM-tunnisteen, jonka taustajärjestelmä välittää GCM-palveluun. Prototyyppi rekisteröi puhelimen aina käynnistyessään GCM-palveluun, minkä avulla GCM osaa lähettää viestin oikeaan puhelimeen ja oikealle sovellukselle. Prototyypissä on toteutettuna erillinen lähetysten vastaanottaja (Broadcast Receiver), joka on manifestissa määritetty käsittelemään muun muassa GCM_RECEIVED_ACTION-tyyppisiä tarkoitus-viestejä, minkä kautta kaikki GCM-viestit tulevat.



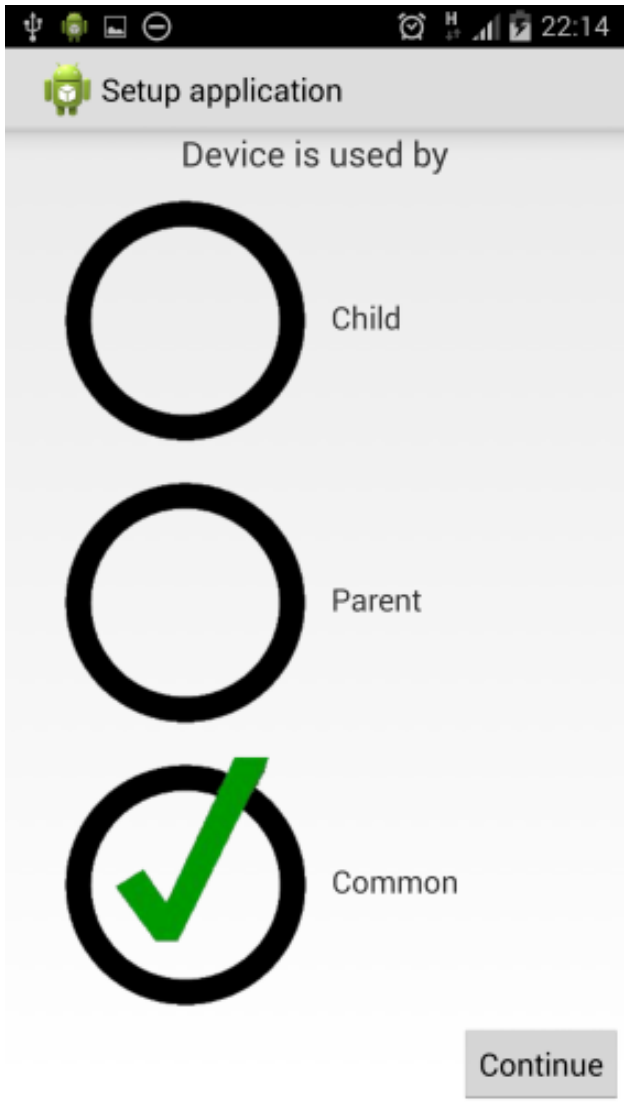
Kuva 7. GCM-viestin kulku vanhemman puhelimesta lapsen puhelimeen.

Jokaiselle prototyypin tietomallille on luotu omat Java-luokkansa. Tiedot tallennetaan laitteen muistiin ja välitetään verkon yli taustajärjestelmään aina JSON-muodossa. Tallennuksessa käytetään Androidin avain-arvo-pari -tyyppistä tallennusmuotoa SQL:n sijaan. Lista Java-olioita muunnetaan Gson-kirjastoa hyödyntämällä JSON-muotoiseksi tekstiksi, joka tallennetaan arvoksi sopivaksi katsotun avaimen taakse. Esimerkiksi käyttäjälista tallennetaan ”users”-avaimen taakse. Ratkaisun toimivuutta käydään tarkemmin läpi kappaleessa 6.1, jossa tarkastellaan sovelluksen ja ratkaisujen toimivuutta.

5.2. Ensiasennus

Sovellusta ensimmäistä kertaa käynnistäessä siihen pitää määrittää muutamia pakollisia asetuksia. Ensimmäisenä käyttäjän tulee valita laitteen käyttötapa (Kuva 8). Laite voi olla joko lapsen käytössä, vanhemman käytössä tai yleisessä käytössä. Sovelluksen toimintatapa riippuu tästä valinnasta huomattavan paljon. Lasten käyttöön asetetut sovellukset pitävät käytönvalvonnan jatkuvasti päällä. Vanhemman käyttöön asetetut sovellukset antavat vanhemmalle mahdollisuuden hallita lastensa käytössä olevien laitteiden asetuksia suoraan omasta laitteestaan. Yleiseen käyttöön

asetetut sovellukset voivat sisältää useita käyttäjiä eri oikeuksilla. Tällöin sovellus asettaa laitteeseen myös näyttölukituksen, jolloin laitetta ei saa auki ilman valitun käyttäjän tunnuksia, mikäli tällaiset on asetettu.



Kuva 8. Prototyypin käyttötavan valinta.

Käyttötavan valinnan jälkeen vanhemman tulee aina luoda ensimmäisenä oma käyttäjätilinsä riippumatta käyttötavasta. Lapsen käyttöön valittu sovellus pyytää lisäksi käyttäjätilin luomisen lapsen käyttöön. Yleiseen käyttöön valittuun sovellukseen voi luoda niin monta käyttäjää, kuin on tarve.

Yleiseen käyttöön valitut sovellukset vaativat aina näyttölukituksen, josta pitää valita käyttäjä, kenelle käyttötilastot kirjataan. Näyttölukitusta ei luonnollisesti tule voida kiertää esimerkiksi painalla puhelimen koti-painiketta. Jotta prototyyppi saadaan valittua uudeksi oletus

kotisovellukseksi, tulee yhdelle prototyypin aktiviteeteista antaa tarkoitussuodatin (intent-filter) seuraavalla tavalla:

```
<intent-filter>
   <action android:name="android.intent.action.MAIN"/>
   <category android:name="android.intent.category.DEFAULT" />
   <category android:name="android.intent.category.HOME" />
</intent-filter>
```
(5.1)

Tarkoitussuodattimen kategoria android.intent.category.HOME kertoo Android-järjestelmälle, että kyseinen aktiviteetti tarjoaa uuden vaihtoehdon käyttäjälle kotipainiketta painettaessa (Kuva 9). Prototyyppi tulisi valita uudeksi kotisovellukseksi, jolloin tunnistautumaton käyttäjä ei pääse lukitusruudusta puhelimen alkuperäiseen kotinäkymään painamalla koti-painiketta. Alkuperäinen kotinäyttö halutaan kuitenkin näyttää kotipainiketta painettaessa, mikäli käyttäjä on tunnistautunut. Alkuperäinen kotisovelluksen paketti ja luokka on luettavissa ja tallennettava muistiin, jotta se saadaan tarpeen mukaan näytettyä käyttäjälle.

Kuva 9. Prototyypin valinta kotipainikkeen oletus-sovellukseksi.

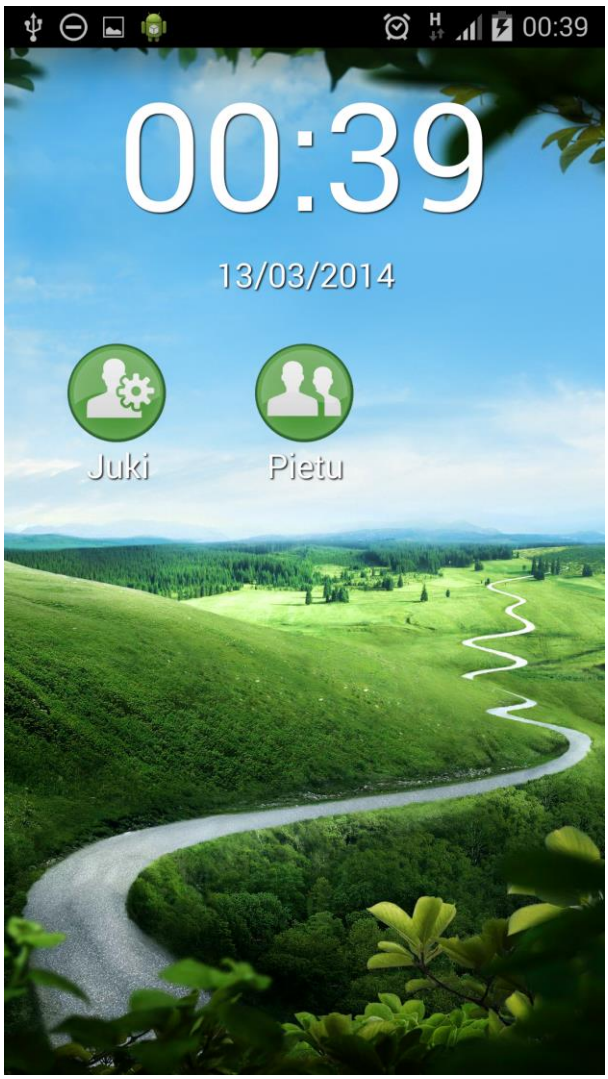
Viimeiseksi käyttäjän on asetettava sovellus laiteylläpitäjä-tilaan (device admin), joka estää käyttäjää poistamasta tai sammuttamasta sovellusta. Laiteylläpitäjä-tilassa olevia sovelluksia pystyy kuitenkin hallitsemaan Android-järjestelmän omien asetusten kautta. Täten on käyttäjä ohjeistettava luomaan profiili asetus-sovelluksen käytön estolle kaikille lapsikäyttäjille, jotta sovellusta ei pysty poistamaan tai sulkemaan.

5.3. Käyttäjätilit

Käyttäjätilit sallivat laitteella olevan useita käyttäjiä erilaisilla rajoituksilla ja asetuksilla. Etenkin taulutietokoneiden käytössä on yleistä, että sitä voi käyttää kaikki perheenjäsenet ja mahdollisesti myös vieraat. Käyttäjätili edellyttää, että pääkäyttäjä luo sovellukseen uuden käyttäjän ja asettaa tälle halutut rajoitukset ja valvonnat päälle. Pienemmille lapsille on esimerkiksi mahdollista valita

heille soveltuvat pelit, jolloin lapsi ei näe eikä pysty käynnistämään mitään muita sovelluksia, kuin hänelle valitut. Vierailija-tilillä sitä vastoin saisi laitteen virallisen kotinäkymän esiin, mutta esimerkiksi asetuksiin ei voisi päästä.

Tuki käyttäjätileille edellytti toteutettavaksi näyttölukon prototyyppiin. Näyttölukko on mobiililaitteen päällimmäisenä oleva sovellus, joka ilmestyy pienellä viiveellä aina, kun näyttö pimenee. Sovellukseen on luotava asennusvaiheessa pääkäyttäjä, kenellä on oikeus lisätä uusia käyttäjiä milloin tahansa. Näyttölukko näyttää kellon, päivämäärän ja jokaisen sovellukseen syötetyn käyttäjän kuvan ja nimen (Kuva 10). Kun käyttäjä valitsee yhden käyttäjistä, sovellus pyytää valitun käyttäjän salasanaa tai kuviolukitusta, minkä jälkeen käyttäjällä on mahdollisuus päästä käsiksi laitteen sovelluksiin.



Kuva 10. Näyttölukitus-näkymä kahdella käyttäjällä.

Kustomoidun näyttölukituksen toteutus ei ole suoraan tuettuna Android-alustassa, joten sellaisen tekeminen ei ole täysin suoraviivaista. Sovelluksen eri näkymille voi asettaa erilaisia lippuja, joista yksi mahdollistaa sen, että näkymä tulee aina näkyviin näyttölukonkin päälle:

```
getWindow().addFlags(
    WindowManager.LayoutParams.FLAG_DISMISS_KEYGUARD +
    WindowManager.LayoutParams.FLAG_SHOW_WHEN_LOCKED
);.
```
(5.2)

Näyttölukituksen kiertämiseksi tuli tehdä useita toimenpiteitä. Edellisessä kappaleessa (5.2) kerrottiin kuinka prototyypin asettamisella kotisovellukseksi saatiin lukitusnäytön kiertäminen koti-painiketta painamalla estettyä. Android-puhelimissa on myös takaisin-painike, jonka toiminnallisuuden ylikirjoittaminen on suoraan tuettuna kaikille sovelluksille toteuttamalla metodi: onBackPressed. Jättämällä metodin sisällön tyhjäksi, takaisin-painikkeesta ei tapahdu mitään. Kotipainiketta painettaessa pitkään pohjaan avautuu näkymä viimeisimmistä käytetyistä sovelluksista. Sovellusten käyttöoikeuksia tarkastelevaan taustajärjestelmään oli tehtävä tarkistus, jolla estettiin sovelluksien avaaminen ilman käyttöoikeuksia. Tästä kerrotaan lisää kappaleessa 5.6. Näiden lisäksi Android-järjestelmissä on yläreunassa ilmoituspalkki, jonka alas vetämällä pääsee näkemään asetus- ja ilmoitusnäkymän. Näkymästä on mahdollista päästä Androidin asetuksiin, josta on muun muassa mahdollista poistaa tai sammuttaa sovelluksia. Ilmoituspalkin alas vetämisen estämiseksi Android ei tarjoa mitään ominaisuuksia. Täten ainoa keino estää sovelluksen poistaminen on, että käyttäjä lisää asetusnäkymän aina kielletyksi sovellukseksi. Kolmansien osapuolinen sovelluskehittäjillä on mahdollisuus antaa käyttäjälle vaihtoehto määrittää sovellus laiteylläpitäjä-tilaan, jolloin sovelluksen poisto- tai sammuta-painiketta ei voi käyttää. Laiteylläpito-tila tulee asettaa sovelluksen sisällä itsessään pääkäyttäjän toimesta ennen kuin muita käytönvalvonta-asetuksia pystyy määrittämään. Laiteylläpito-tilan voi kuitenkin ottaa pois käytöstä myös laitteen omista asetuksista, minkä jälkeen sovelluksen poisto on jälleen mahdollista.

Lapsi-käyttäjille on mahdollisuus valita hiekkalaatikko-tila tai alkuperäinen-tila. Hiekkalaatikko-tilassa kyseisen lapsen kirjautuessa laitteelle prototyyppi lukee asetuksista ja käyttötilastoista sovellukset, joita lapsen on mahdollisuus käyttää juuri sillä hetkellä. Tarkistusta tehdessä lasketaan ensimmäisenä jokaisen sovelluksen käyttöaika. Tämän jälkeen haetaan lista kaikista laitteeseen asennetuista sovelluksista ja ne käydään läpi. Jokaisen sovelluksen kohdalla tarkistetaan onko, käyttöaika ylittynyt tai kuuluuko sovellus kielletyiden tai sallittujen sovellusten listaan. Jos tarkistushetkellä on aktiivisena käyttöprofiili, mihin valitut sovellukset on merkitty sallitulle listalle, kaikki tähän listaan kuulumattomat sovellukset eivät ole sallittuja käyttää. Myös aikuisen on mahdollista kirjautua lapsi-asetuksilla toimivaan laitteeseen. Prototyyppi-sovelluksen käyttöliittymää käynnistäessä sovellus pyytää käyttäjän salasanaa. Vanhemman tunnuksilla kirjautuessa laite antaa pääkäyttäjä-oikeudet laitteen kaikkiin sovelluksiin, jolloin vanhempi voi

esimerkiksi määritellä Androidin omia asetuksia. Pääkäyttäjäoikeudet lähtevät pois, kun näyttö menee pois päältä tai ilmoitusalueelle ilmestynyttä ”pääkäyttäjä-tila” -ilmoitusta painaa sormella.

Prototyypin näyttölukitukseen ei ole toteutettu mahdollisuutta soittaa edes hätäpuheluita, mikä olisi kuitenkin tarpeellinen ja hyvin tärkeä ominaisuus. Puheluiden vastaanottaminen on kuitenkin mahdollista vaikka näyttölukko olisi päällä.

5.4. Internet-historia ja -suodatus

Sovelluksen toteutuksen hankalin osa-alue oli tehdä luotettava ja selkeä Internet-historiatietojen tallennus ja suodatus. Prototyypin Internet-liikenne ohjataan sovellukseen sisällytetyn välityspalvelin-kirjaston (LittleProxy) kautta, millä pystyy lukemaan lähes kaikkien sovellusten http-pyyntöjen osoitteet ja tarvittaessa estämään pääsyn sivulle. Useat sovellukset käyttävät Android-järjestelmälle määrättyä välityspalvelinta, mikäli tällainen on käyttäjän toimesta asetettu. Välityspalvelinten asetuksia ei voi asettaa automaattisesti sovelluksen sisältä käyttäjän puolesta tietoturvasyistä, vaan käyttäjä on ohjeistettava asettamaan välityspalvelin-asetukset itse. Jos mikä tahansa kolmannen osapuolen sovellus pystyisi asettamaan laitteen välityspalvelin-asetukset käyttäjän tietämättä, olisi niiden mahdollista tarkkailla salaa käyttäjän koko Internet-liikennettä. Salaamattomilta osoitteilta olisi myös mahdollista lukea sivustojen sisältöä ja pyyntöjen parametreja. Tästä syystä käyttäjän on aina itse määritettävä käytettävä välityspalvelin, johon he varmasti voivat luottaa.

Koska prototyyppi sisältää välityspalvelimen puhelimessa itsessään, pyörii se aina osoitteessa 127.0.0.1, mikä saattaa näyttää monelle käyttäjälle jopa pelottavalta. Välityspalvelimen asettamisen jälkeen lähes kaikki Internet-liikenne kulkee kolmannen osapuolen sovelluksen (prototyypin) läpi, mikä saattaa olla monelle käyttäjälle myös pelottava asia. Välityspalvelin pystyy lukemaan verkkoliikenteestä jopa pyyntöjen sisältöä, mikäli se ei mene salattua yhteyttä pitkin (esimerkiksi https tai ssh). Jotkin Android-versiot sisältävät salaisen välityspalvelinten asetussivun, mutta toisissa versioissa välityspalvelimet pitää käydä asettamassa erikseen langattomalle (WLAN) ja mobiiverkolle. Tämän ohjeistaminen käyttäjälle on erittäin haasteellista. Sovelluksen sisältä on kuitenkin mahdollista tuoda esiin Android-järjestelmän omiakin aktiviteetteja, mikä osaltaan helpottaa käyttäjän viemistä oikeaan asetusnäkymään.

Kun välityspalvelinasetukset on saatu asetettua oikein, alkaa sovellus automaattisesti tallentamaan kaikkien välityspalvelinasetuksia noudattavien sovellusten Internet-liikennettä. Näistä pyynnöistä suurin osa on kuitenkin epäolennaista vanhemmille. Esimerkiksi yksi pyyntö Googlen etusivulle (http://google.fi) tekee taustalla 26 erillistä pyyntöä resursseja hakeakseen. Tällöin kaikki 27 pyyntöä tallentuvat vanhemman tarkasteltavaksi, mutta vain yksi niistä on tarpeellinen. Kuva 11 näyttää erään päivän Internet-historiaa. Kuvassa näkyy LinkedIn-sivuston lataus, jolloin historiaan on tallentunut myös yksi resurssi-tiedosto ja Google-mainoksen lataus. Sivuston nimen oikealla puolella näkyy, kuinka monta kertaa kyseiseen juuriosoitteeseen on tehty pyyntöjä.

Kuva 11. Internet-historian tarkastelunäkymä.

Internet-suodatuksen toteuttamiseen olisi ollut myös vaihtoehtoisia tapaa. Yksi niistä olisi laittanut laitteen omat selaimet kiellettyjen sovellusten listalle ja selain olisi sisällytetty prototyyppiin itseensä. Androidin rajapinta mahdollistaa kolmansien osapuolien sovellusten sisällyttää aktiviteettiin WebView-komponentin [Android WebView, 2015], joka osaa piirtää Internet-sivuston näkyviin. WebView-komponentti käyttää WebKit -piirtomoottoria (rendering engine) sivuston mallintamisessa. WebView-komponentti tukee myös JavaScript-skriptejä ja keksien (cookies) tallentamista. Komponentti olisi siis ominaisuuksiensa puolesta ollut oikein hyvä vaihtoehto suodatuksen toteuttamiseen. WebView-komponenttiin voi asettaa oman WebViewClient-toteutuksen, jolla on mahdollista tarkkailla avattavien sivustojen osoitetta ja tarvittaessa kieltää pääsy. Tätä lähestymistapaa käyttämällä olisi välttynyt monimutkaisten

välityspalvelinasetusten vaivalta. WebView-komponentti ei kuitenkaan tarjoa kaikkia samoja ominaisuuksia, mitä oikeat selaimet. Selainnäkymän avaaminen prototyypin sisältä olisi ollut vaivalloisempaa. Lisäksi pääkäyttäjän tulisi varmistaa, ettei Google Play -sovellus - tai muut sovellusverkkokaupat - ole käytettävissä olevissa sovelluksissa, ettei niiden kautta pääse asentamaan valvomattomia selainohjelmia. Välityspalvelinta käyttämällä pystyy valvomaan myös kaikkien kolmansienkin osapuolten sovellusten - muidenkin, kuin selainten - liikennettä, mikäli ne vain kunnioittavat Android-järjestelmään asetettuja välityspalvelin-asetuksia. Toinen vaihtoehto olisi ollut käyttää sovelluksen ulkopuoleista välityspalvelinta. Tämä vaihtoehto olisi kuitenkin vaatinut samat välityspalvelin-asetusten määrittämisen ja ulkoinen välityspalvelin kuormittuisi helposti suuresta käyttäjämäärästä. Välityspalvelimen tulisi olla tietoinen suodatin-asetuksista ja kyvykkyys lähettää Internet-historia oikeisiin laitteisiin tarkasteltavaksi. Monet muut suodatustavat olisi vaatinut laitteen asettamista pääkäyttäjä-tilaan (rooting), jolloin käyttäjä saisi laitteeseen pääkäyttäjän oikeudet samaan tapaan kuin Linux-järjestelmissä. Pääkäyttäjän oikeuksilla laitteen nimipalvelinasetukset (DNS) olisi voinut asettaa käyttämään omaa nimipalvelinta, jolle on mahdollista asettaa kaikki kielletyt sivustot. Prototyyppi on kuitenkin kehitetty sillä ajatuksella, että sovellus olisi mahdollista laittaa jakoon esimerkiksi Google Play -verkkokauppaan. Kaikilla verkkokauppojen asiakkailla ei todennäköisesti ole laitteet asetettuna pääkäyttäjä-tilaan.

5.5. Sijaintitietojen tallennus

Sijaintitietojen tallennus vaatii käyttäjältä paikannustietojen sallimisen sovelluksen asentamisen yhteydessä. Yleisistä asetuksista pääkäyttäjä voi valita kaikki käyttäjät, keiden sijaintitietoja halutaan paikantaa ja kuinka usein. Lasten sijaintitietoja tarkasteltaessa tallennetut sijainnit näkyvät punaisena pallona Google Maps -karttapalvelun päällä (Kuva 12). Mitä kauemmin laite on pysynyt samalla alueella, sen isompi pallo paikassa näkyy. Google Maps -rajapinta tukee erilaisten kuvioiden piirtämisen kartalle helposti myös Android laitteissa [Google Maps, 2015].

Myöhemmissä versioissa pääkäyttäjälle voisi antaa mahdollisuus rajata kartasta alueita ja määritellä ne joko kielletyksi tai sallituksi alueeksi. Mikäli laite poistuu tai saapuu määritellylle alueelle, pääkäyttäjälle tulisi tästä ilmoitus.

Kuva 12. Sijaintitietojen tarkasteleminen lasten koulupäivästä vanhemman laitteessa.

5.6. Sovellusten käytön rajaus

Jotta sovellusten käyttöä voidaan rajoittaa joko kieltämällä sovelluksen käyttö kokonaan tai asettamalla käyttöaikaa, prototyypillä pitää olla kyvykkyys lukea kaikkien puhelimeen asennettujen sovellusten tiedot. Sen lisäksi prototyypin pitää pystyä näkemään, mitä sovellusta ollaan aikeissa käynnistää sekä aktiivisena oleva sovellus. Kuten kappaleessa 3.3, käytönvalvonnan haasteet kerrottiin, aktiivisten sovellusten lukemiseksi tehty metodi on poistettu Android 5.0 -versiosta lähtien. Tämän päivityksen myötä sovellusten käyttöä ei voi kolmannen osapuolen sovelluskehittäjien toimesta valvoa tai rajoittaa. Ennen versiota 5.0 tämä oli kuitenkin mahdollista.

Yksi prototyypin taustapalveluista tarkkailee jatkuvasti aktiivisena olevaa sovellusta silloin, kun laitteen näyttö on päällä (Algoritmi 1). Käyttöoikeuksien tarkistus pyörii ikuisessa silmukan sisällä

omassa taustaprosessissaan. Silmukka odottaa jokaisen kierroksen välissä kolme sekuntia ennen uuden tarkistuskierroksen aloittamista niin kauan, kuin laitteen näyttö on päällä. Silmukan alussa haetaan aina sillä hetkellä aktiivisena oleva sovellus. Aktiivinen käyttäjä tarkistetaan aina siltä varalta, että sovellus on asennettu toimimaan yleisessä käytössä. Näyttö lukitaan heti, jos aktiivista käyttäjää ei ole asetettuna. Käyttöoikeuksien ja -aikojen tarkistus tehdään ainoastaan lapsiksi määritellyille käyttäjille. Mikäli sovellus on juuri käynnistetty, tarkistetaan, oliko ennen käynnistettyä sovellusta jokin sovellus käynnissä ja tallennetaan sen käyttöaika. Asetetaan juuri käynnistetty sovellus väliaikaiseen luokkamuuttujaan, jotta sovellusten vaihdosta voidaan tarkastella tätä muuttujaa vasten. Nollataan samalla käyttöaikalaskuri. Mikäli sovellus ei ole muuttunut, kasvatetaan sen käyttöaikaa algoritmin rivillä 15 odotetun ajan verran. Ajan kasvatuksen jälkeen voidaan tarkistella onko sovellus ylipäänsä sallitulla listalla tai onko käyttöaika ylittynyt sallitusta. Aktiivisen sovelluksen käyttö estetään tuomalla prototyypin varoitusnäkymä päällimmäiseksi sovellukseksi, jos jompikumpi edellisistä ehdoista on tosi. Uusi tarkistuskierros aloitetaan pienen viiveen jälkeen. Algoritmin lopussa tarkistetaan vielä, onko laitteen näyttö mennyt juuri kiinni. Ehdon täyttyessä aktiivisen sovelluksen käyttöaika tallennetaan ja aktiivinen sovellus sekä aikalaskuri nollataan. Algoritmi jää odottamaan säielukon avaamista siihen asti, kunnes näyttö menee jälleen päälle.

Algoritmi 1 Sovellusten käyttöoikeuksia ja -aikaa tarkastelevan taustapalvelun toiminta-algoritmi.

```
1: while sovellus on päällä do
2:   aktiivinenSovellus ← hae tällä hetkellä aktiivinen sovellus
3:   if aktiivinenSovellus on löytynyt then
4:     if aktiivista käyttäjää ei ole valittuna then
5:       lukitse näyttö
6:     if käyttäjä on lapsi then
7:       if aktiivinenSovellus on muuttunut then
8:         tallenna edellisen sovelluksen käyttöaika
9:         tallenna aktiivinenSovellus väliaikaiseen luokkamuuttujaan
10:        aktiivisenSovelluksenKäyttöaika ← 0
11:      else
12:        kasvata aktiivisenSovelluksenKäyttöaika
13:      if aktiivinenSovellus käyttö on kielletty tai käyttöaika on ummessa then
14:        estä aktiivinenSovellus käyttö
15:  odota hetki ennen uutta tarkistuskierrosta
16:  if näyttö meni kiinni then
17:    tallenna aktiivinenSovellus käyttöaika
18:    tyhjennä aktiivinenSovellus
```

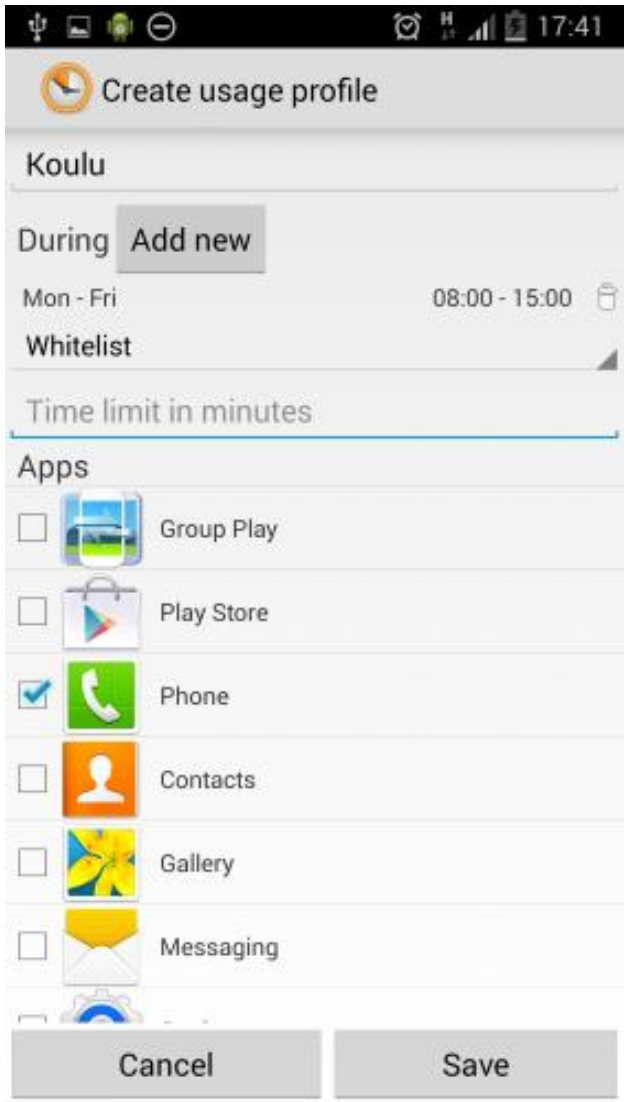
19: *aktiivisenSovelluksenKäyttöaika* ← 0
20: odota näytön menemistä uudelleen päälle

Kolmen sekunnin viive uuden tarkastuskierroksen tekemiksi antaa siis rajoitetulle käyttäjälle mahdollisuuden avata kielletty sovellus hetkeksi. Lisäksi taustaprosessien aktiivisuuden priorisointi Android-järjestelmässä saattaa vaihdella, riippuen laitteen sen hetkisestä kuormittumisesta. Tämä tarkoittaa sitä, että sovellusten valvontakierrosten välissä oleva kolmen sekunnin viive saattaa kasvaa jopa viiteen sekuntiin. Taustaprosessien säikeiden priorisointia on mahdollista kasvattaa hieman asettamalla seuraava asetus:

```
Thread.setPriority(Thread.MAX_PRIORITY);
```
(5.3)

Tästä huolimatta tarkistus viive saattaa yltää ajoittain viiteen sekuntiin. Tämä mahdollistaa käyttäjän pääsemään esimerkiksi Androidin omiin asetuksiin, mikä antaa käyttäjälle mahdollisuuden ottaa käytönvalvontasovellus pois käytöstä. Tätä aukkoa varten sovellus on hyvä suojata laitepääkäyttäjä-asetuksella.

Sovellusten käytön rajaukseen määriteltävien profiilien luonti on pyritty tekemään prototyyppiin mahdollisimman joustavaksi. Joustavuuden parantuessa käytettävyys hankaloitui jonkin verran, mistä kerrotaan lisää luvussa 6.1. Jokaiselle profiilille määritellään nimi, ajanjaksot, rajoitustyyppi, mahdollinen aikaraja sekä sovellukset, joihin profiili vaikuttaa (Kuva 13). Ajanjaksoilla määritellään viikonpäivät ja kellonajat, milloin profiilin rajoitukset ovat voimassa. Rajoitustyyppi voi olla joko sallitut sovellukset (whitelist) tai kielletyt sovellukset (blacklist). Sallituille sovelluksille voi lisäksi antaa aikarajan, kuinka kauan päivän aikana profiiliin valittuja sovelluksia voi käyttää. Yleisessä käytössä olevissa laitteissa valitaan lisäksi lapset keille luotu profiili halutaan ottaa käyttöön.



Kuva 13. Käyttöprofiilin luonti.

5.7. Puheluiden ja viestien valvonta

Joskus vanhemmilla voi olla tarve rajoittaa tai valvoa puhelimesta lähteviä ja saapuvia puheluita ja viestejä. Toisinaan tarve voi olla vähentää puhelinlaskujen summaa ja toisinaan valvoa, kenen kanssa lapsi on tekemisissä. Android-alustassa on mahdollista tarkkailla saapuvien puheluiden soittajien puhelinnumeroita ja katsoa löytyykö puhelinnumero tallennettuna laitteen yhteystietoihin.

Prototyypin puheluiden valvonta on mahdollista asettaa päälle, jolloin kaikki saapuvat puhelut estetään, jos numero ei löydy tallennetuista yhteystiedoista. Soitosta lähtee ilmoitus vanhemman puhelimeen, jolloin vanhempi pystyy halutessaan tarkistamaan soittajan henkilöllisyyden ja asian. Android mahdollistaa myös lähtevien puheluiden tarkkailun. Lähteviin puheluihin voi rekisteröidä lähetyksenvastaanottajan, jota kutsutaan aina puhelua tehtäessä ja puhelun katketessa. Tapahtumien välinen aika on laskettavissa, jolloin olisi helppo antaa vanhemmille mahdollisuus määrittää aika, kuinka paljon puhelimella voi soittaa tietyn ajan sisällä.

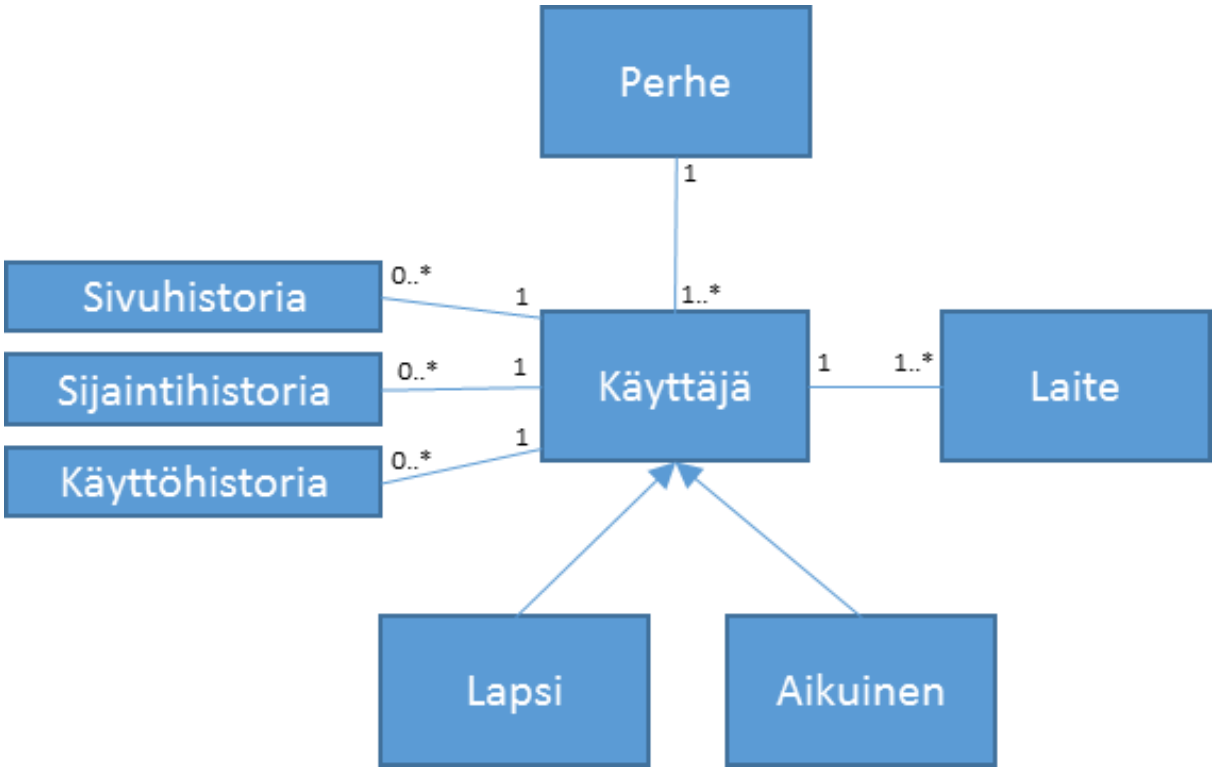
Androidin sisällöntoimittajia hyödyntämällä kolmansien osapuolten sovellusten on mahdollista lukea myös kaikki laitteeseen saapuvat ja laitteesta lähtevät tekstiviestit. Jotkut vanhemmat saattavat haluta tarkkailla lastensa tekstiviesti-liikennettä suojellakseen näitä joutumasta seksuaalirikollisten uhriksi. Tekstiviestien luku, kuten myös puheluiden tarkkailu, vaativat oikeuksien määrittämisen manifesti-tiedostoon, joka näytetään käyttäjälle sovellusta asennettaessa. Prototyyppiin ei ole toteutettuna tekstiviestin luku-ominaisuutta.

Nykyään nuoret käyttävät kuitenkin hyvin paljon muitakin viestintäsovelluksia, kuin puheluita ja tekstiviestejä. Kolmansien osapuolten viestintäsovelluksia on käytännössä mahdotonta valvoa. Monet viestintäsovellukset mahdollistavat viestin lähetyksen vastaanottajalle, mikäli heillä on vastaanottajan puhelinnumero tiedossa, joissakin ei tarvitse edes puhelinnumeroa.

5.8. Hallinta

Koko perheen lasten käytönvalvonnan hallinta voi tapahtua kolmella eri tapaa, joista kaksi on toteutettuna prototyyppiin. Vanhemman on mahdollista hallita lapsen asetuksia asettamalla ne suoraan laitteisiin, joita lapsi käyttää. Sovellus on myös mahdollista asentaa vanhemman omaan laitteeseen, josta kaikkien lasten asetuksia voi myös hallita. Kolmas vaihtoehto olisi tehdä taustajärjestelmään käyttöliittymä, josta vanhempi voisi hallita ja valvoa lasten puhelinten tai taulutietokoneiden käyttöä. Hallinta sisältää sekä käytönvalvonta-asetusten määrittämisen, että tapahtumien seurannan.

Koska perheessä voi olla useampia laitteita, joita lapset käyttävät, on tarpeellista saada liitettyä sovellukset yhden perheen alle sovellustasolla. Tämä tarkoittaa sitä, että tietomalliin luodaan perhe-malli, jonka alle käyttäjät lisätään (Kuva 14).



Kuva 14. Prototyypin tietomallit ja mallien relaatiosuhteet.

Ensimmäinen laitteeseen lisätty käyttäjä on aina pääkäyttäjä, jonka sähköpostiosoite toimii perheen yksilöivänä tunnisteena. Tästä syystä pääkäyttäjän sähköposti pitää olla kaikissa laitteissa sama, jotta käyttäjät voidaan yhdistää oikean perheen alle. Koska on mahdollista, että käyttäjä kirjoittaa sähköpostinsa väärin erehdyksessä tai tahallaan, piti varmistaa, että sähköposti on varmasti oikein. Käyttäjien on mahdollista kytkeä Android-järjestelmään oma Google-tilinsä, joka sisältää aina sähköpostiosoitteen. Kaikki lisätyt käyttäjät on mahdollista hakea kolmansien osapuolien sovelluksen sisältä, josta käyttäjä voi valita oikean tilin luodessaan käyttäjää prototyyppiin. Kaikki käyttäjät lähetetään heti luonnin jälkeen myös taustajärjestelmään, jossa käyttäjät liitetään oikean perheen alle.

Sovellus hyödyntää Google Cloud Messaging (GCM) -palvelua lähettäessään viestejä mobiililaitteesta toiseen. Tähän tarkoitukseen laitteen on rekisteröitävä itsensä GCM-palveluun aina käynnistyessään, mistä jokainen laite saa yksilöllisen tunnisteen. Tämä tunniste lähetetään taustajärjestelmään ja assosioidaan sen hetkiselle käyttäjälle. Vanhemman tunnistautuessa laitteeseensa, sovellus hakee taustajärjestelmästä kaikki perheeseen kuuluvat lapsi-käyttäjät. Jos vanhempi haluaa lähettää lapselleen lisää käyttöaikaa, lähtee viesti tavallisella http-post-pyynnöllä ensin taustajärjestelmään. Taustajärjestelmä katsoo kyseisellä lapsella sillä hetkellä aktiivisena olevan laitteen GCM-tunnisteen ja lähettää GCM:ään lisäaikaa myöntävän sanoman. GCM lähettää

edelleen sanoman eteenpäin lapsen laitteeseen, jossa prototyypin taustapalvelu käsittelee sen ja antaa lapselle lisää käyttöaikaa, vastaavasti kuten Kuva 7. Mikäli yksikään lapsen laitteista ei sillä hetkellä ole taustajärjestelmässä aktiivisessa tilassa, sanoma jää odottamaan kunnes jokin laitteista aktivoituu. Aktivoitumisen yhteydessä tarkistetaan ja välitetään mahdolliset odottavat viestit, uudet asetukset ja käyttöhistoria.

Kuten edellä todettiin, laite määritellään taustajärjestelmään aktiiviseksi aina, kun lapsi avaa laitteensa näytön tai kirjautuu yleisessä käytössä olevan laitteen sisään omilla tunnuksillaan. Laitteen näytön sammuessa laite otetaan pois aktiivinen-tilasta. Näytön tilan tarkastelemista varten Android tarjoaa kehittäjille mahdollisuuden määrittää manifesti-tiedostoon vastaanottaja-palvelun, jonne tulee ilmoitukset tarkoitus-viesteinä myös näytön tilan muutoksista. Käyttäjän aktivoinnin yhteydessä taustajärjestelmästä haetaan myös uudet käytönvalvonta-asetukset, joita vanhempi on saattanut oman laitteensa kautta asettaa. Sen lisäksi haetaan lapsen käyttötilastot siltä varalta, että lapsi on mahdollisesti käyttänyt aikarajoitettuja sovelluksia muissa laitteissa, jolloin niiden käyttö voidaan estää muissakin laitteissa. Näytön sammumisen yhteydessä sovellus lähettää käyttötilastot taustajärjestelmään.

Kaikki sanomat liikkuvat laitteiden ja taustajärjestelmän välillä REST-arkkitehtuurimallilla JSON-muodossa. Prototyyppiin on rakennettu erillinen välittäjä-palvelu (BackendMediator), mikä hoitaa kaiken keskustelun taustajärjestelmän kanssa. Tällä tavoin taustajärjestelmän osoitteiden tai rajapintojen muuttuessa on yksinkertaista päivittää vain kyseisen palvelun metodit. Jotta kuka tahansa ei voi hakea taustajärjestelmästä muiden käyttäjien tietoja, piti rajapinta suojata. Suojaus on toteutettu siten, että taustajärjestelmästä pyydetään aina käyttöoikeustunniste (access token), millä laite tunnistautuu aina tietoja pyytäessään tai tallentaessaan. Aihetta käydään vielä lyhyesti läpi kappaleessa 6.2.

6. Tulokset

6.1. Sovelluksen toimivuus

Prototyyppi on testattu toimivaksi laitteissa, joissa on Androidin versio 3.1 – 4.4. Toimivalla tarkoitetaan tässä tapauksessa sitä, että sovellus käynnistyy ja sillä pystyy rajoittamaan ja tarkkailemaan laitteen sovellusten käyttöä. Monien ominaisuuksien toteutus vaati kuitenkin toimiakseen erilaisia ongelmien kiertotapoja, jotka aiheuttavat yleensä sovelluksen epävakautta, ominaisuuden kiertomahdollisuutta ja käytettävyyden heikkenemistä.

Internet-suodatuksen käyttöönotto vaatii käyttäjältä monimutkaisten ja teknisten välityspalvelin-asetusten määrittämistä, mikä heikentää huomattavan paljon sovelluksen käytettävyyttä. Internet-suodatus itsessään on jätetty prototyypissä täysin vanhempien vastuulle, mikä vaatii vanhemmilta jatkuvaa valvontaa ja asetusten määrittämistä. Mikäli vanhemmat valitsevat lähestymistavan, jossa määritellään sivustot, joihin lapsen on mahdollista päästä, pitää lapsen aina erikseen pyytää pääsyä uusille sivustoille. Tämän voisi automatisoida puoliksi, jolloin estetyn sivuston osoite lähetettäisiin vanhemman puhelimeen ilmoituksena. Ilmoitus näyttäisi sivuston, mihin kyseinen lapsi on yrittänyt päästä ja sen alla kaksi painiketta: salli ja estä. Painiketta painettaessa lähtee ilmoitus GCM-palvelun kautta viiveettä lapsen puhelimeen johon uusi asetus tallentuu. Toinen lähestymistapa on määritellä erillinen estolista sivustoista joihin pääsy halutaan estää. Tämä vaatii vanhemmilta jatkuvaa tarkkailua lapsen Internet-historiasta, josta saattaa paljastua sivustoja, jotka halutaan jatkossa estää. Tätäkin prosessia voisi hieman helpottaa siten, että lisää Internet-historia-listaan jokaisen sivuston viereen painikkeen, josta sivuston voi suoraan lisätä estolistalle. Näiden lisäksi prototyyppistä on hankala tarkastella Internet-historiaa. Internet-liikennettä tarkkaileva palvelu välittää jokaisen selaimen tekemän pyynnön, jolloin sieltä on hankala poimia vanhemmille oleellisimmat osoitteet. Monet sivustot tekevät pyyntöjä muun muassa omille resurssi-palvelimilleen, jotka toimivat eri osoitteiden alla, jolloin niistä muodostuu historiaan aina oma rivinsä. Vanhempien on usein hankala tunnistaa näitä osoitteita, jolloin valvomisesta tulee reilusti työläämpää, jos jokainen osoite tarvitsee tarkastaa erikseen. Yleensä ei välttämättä edes riitä, että tallennetaan pelkkä sivuston pääosoite. Esimerkiksi monet kuvapalvelut voivat tarjota lapsille niin soveliaita, kuin sopimattomiakin kuvia. Tästä syystä vanhemmilla voi olla tarve pystyä näkemään jokainen sivupyyntö pelkän sivuston pääosoitteen sijaan, jolloin listasta saattaa kertyä hyvinkin pitkä.

Näyttölukon toteuttamiseksi piti tehdä myös useille ongelmille erilaisia kiertoja. Yksi niistä oli koti-painikkeen ylikirjoittaminen. Koti-painikkeelle on asetettu Android-järjestelmän kautta aina oma oletussovellus, mikä käynnistetään kun painiketta painaa. Jos oletussovellukseksi ei aseta prototyyppiä, käyttäjä pääsisi koti-painiketta painamalla Androidin omaan kotisovellukseen. Sovellus on rekisteröitävä koti-kategoriaan, jolloin Android tunnistaa koti-painiketta painettaessa, että käyttäjälle pitää antaa mahdollisuus valita uusi oletus-kotisovellus. Toimenpiteen sai automatisoitua lähes täysin. Ainoa vaadittu toimenpide käyttäjältä on valita

käytönvalvontaprototyyppi uudeksi kotisovellukseksi ensimmäisen käynnistyksen yhteydessä. Toinen toimenpide näyttölukon toteuttamiselle oli estää käyttäjää pääsemästä asetuksiin ilmoituspaneelin kautta. Ilmoituspaneelia ei ole mahdollista estää toimimasta kolmansien osapuolten sovellusten toimesta. Siksi taustapalvelun sovellustentarkkailijaan oli tehtävä tarkistus, jolla estetään muita sovelluksia käynnistymästä aina, kun näyttölukkoa ei ole avattu oikeaoppisesti. Tällä toimenpiteellä käyttäjä ei voi myöskään päästä muihin sovelluksiin viimeaikaisten sovellusten valikosta painamalla koti-painiketta pitkään pohjassa.

Sovellusten käytön estäminen näyttölukon ollessa voimassa tai vanhemman määrittelemän estolistan tai aikarajojen mukaisesti ei kuitenkaan ollut mahdollista tehdä täysin vedenpitäväksi. Sovelluksia ei pysty sulkemaan tai estämään toimimasta kolmansien osapuolten sovellusten toimesta. Ainoa mahdollisuus estää sovellusten käyttö oli tuoda taustapalvelun kautta prototyypin oma aktiviteetti estetyn sovelluksen päälle. Menetelmän heikkona puolena on se, että kielletyt sovelluksen on kuitenkin mahdollista avata muutamaksi sekunniksi kerrallaan. Se antaa käyttäjille pienen mahdollisuuden esimerkiksi poistaa koko sovellus laitteesta. Sovellusten tarkkailu ja käyttöajan laskeminen on mahdollista Android-käyttöjärjestelmän versioon 4.4 asti, minkä jälkeen aktiivisia sovelluksia ei pysty näkemään kolmansien osapuolten sovellusten toimesta.

Käyttöprofiilien määrittäminen käyttäjän toimesta on käytettävyydeltään hankala, mutta monipuolinen. Käyttäjän on mahdollista luoda useita eri käyttöprofiileita, joilla on eri voimassaoloajat, kohdennetut sovellukset ja käyttötapa. Yksi profiili voi olla esimerkiksi yö-profiili, joka on voimassa kello 22:00 – 07:00, minkä ajan on sallittuna ainoastaan herätyskello-sovellus. Toinen profiili voi olla yleinen estolista, joka määrittää, mitä sovelluksia ei saa ikinä käyttää. Kolmantena profiilina voisi olla vielä peli-profiili, joka määrittää pelit, mitä voi pelata kello 15:00 – 18:00 välillä tunnin ajan joka toinen päivä. Profiilien kertyessä käyttäjän voi olla hyvin hankala seurata, milloin mikäkin profiili on voimassa ja mikä on kenelläkin voimassa. Tämä altistaa virheille, jolloin lapsen voi olla mahdollista esimerkiksi pystyä pelaamaan ajankohtana, mikä ei pitäisi olla sallittu. Parannuksena tähän olisi hyvä tehdä vanhemmille saataville viikkonäkymä, joka näyttäisi kuhunkin lapseen kohdistuvat profiilit värikoodattuina.

Puheluiden ja sijaintitietojen tarkkailu oli toteutuksen kannalta täysin tuettuja Androidin puolesta. Huono puoli sijaintitietojen tarkkailussa on ainoastaan sen helppo kierrettävyys lapsen toimesta. Koska ilmoituspaneelia ei pysty estämään, sieltä on mahdollista ottaa GPS- ja WLAN-toiminnot pois päältä lapsen toimesta. Asetusten oltaessa pois päältä, mitään sijaintitietoja ei pysty tallentamaan. Toimenpide on kuitenkin havaittavissa vanhempien toimesta helposti. Toteutuksen kannalta helppoa oli myös käytönvalvontasovelluksen kannalta oleellinen ominaisuus, että sovellus pysyy käynnissä aina; sekä laitteen uudelleen käynnistyksen yhteydessä, että silloin jos sovellus kaatuu odottamattomasti.

Tiedon tallentamiseksi prototyypissä valittiin avain-arvo-pari-muotoinen tallennusmuoto. Siinä esimerkiksi käyttäjät tallennetaan avaimen ”users” alle, jonka arvoksi asetetaan lista käyttäjistä JSON-muodossa. Tallennusmuoto valittiin siksi, että sen käyttö on hyvin yksinkertainen ja nopea. Sen lisäksi tietoja siirretään hyvin paljon taustajärjestelmään ja takaisin JSON-muodossa. Toisaalta

tiedon hakeminen JSON-muodossa on mahdotonta, jolloin tieto pitää ensin muuntaa Java-olioiksi tai tehdä haku taustajärjestelmän kautta. Yhden käyttäjän päivittämisen toteuttamiseksi pitää muun muassa ensin hakea koko lista käyttäjistä ja käydä jokainen yksitellen läpi, kunnes oikea löytyy. Tämän jälkeen on mahdollista päivittää halutut kentät ja muuntaa koko lista takaisin JSON-muotoon ja tallentaa koko käyttäjälista uudelleen. Tämän ja monen muun toimenpiteen suorittaminen esimerkiksi SQLite-tietokannalla olisi ollut huomattavasti kevyempi operaatio.

Lopuksi mainittakoon, että sovellus tuntui parantavan laitteen akun kestoa huomattavasti. Tätä ominaisuutta ei kuitenkaan ole tutkittu syvällisemmin. Mahdollinen syy tähän on prototyypin taustapalvelun suorituksen korkea priorisointi, jolloin muut taustalla olevat raskaammat prosessit eivät pääse kuluttamaan laitteen akkua.

6.2. Tietoturva ja yksityisyys

Prototyyppi vaatii asentuakseen huomattavan määrän oikeuksia järjestelmän eri toimintoihin toimiakseen kunnolla. Vaadittavia oikeuksia on muun muassa pääsy Internettiin, lukuoikeus käyttäjätileihin, lähtevien ja saapuvien puheluiden tarkkailu, aktiivisten sovellusten lukuoikeus, järjestelmän uudelleenkäynnistyksen tarkkailu, lukeminen ulkoisista tallennuslaitteista, pääsy paikannustietoihin ja verkon tilan tarkkailu. Kaikki sovelluksen vaatimat oikeudet listataan käyttäjälle asennusvaiheessa. Lista on hyvin pitkä ja se saattaa huolettaa käyttäjiä. Oikeuksia ei kuitenkaan voi pyytää jälkikäteen käyttäjältä siinä vaiheessa, kun ominaisuutta tarvitaan. Tällä tavoin se tulisi käyttäjälle paremmin selväksi, että mistä syystä sovellus kyseistä oikeutta vaatii.

Internettiin pääsyä vaaditaan hyvin yleisesti kaikissa sovelluksissa, eikä se todennäköisesti aiheuta ihmetystä käyttäjille. Lukuoikeutta järjestelmän käyttäjätileihin tarvitaan, jotta sitä kautta saadaan pääkäyttäjän oikea sähköpostiosoite. Käyttäjätilejä voi laitteessa olla useampiin eri sovelluksiin, kuten Google, Facebook, LinkedIn, Dropbox ja WhatsApp. Sovelluksen on mahdollista lukea kaikkiin käyttäjätileihin liitetyt sähköpostiosoitteet ja puhelinnumerot. Huolettavaa käyttäjän kannalta on se, ettei hän tiedä luovuttaako sovellus nämä tiedot ulkopuolisille tahoille.

Mahdollisuutta lukea saapuvien puheluiden puhelinnumeroita tarvitaan, jos halutaan estää puhelut numeroista, joita ei löydy laitteen puhelinluettelosta. Lähtevien puheluiden tarkkailua tarvitaan, kun halutaan pitää kirjaa puheajasta. Kaikki puhelinnumerot on jälleen lähetettävissä eteenpäin, eikä käyttäjä voi olla varma mihin tarkoituksiin niitä mahdollisesti käytetään.

Aktiivisten sovellusten lukuoikeutta tarvitaan, kun lasketaan sovellusten käyttöaikaa ja estetään kiellettyjen sovellusten käyttö. Tämä oikeus on otettu pois käytöstä Android versiosta 5.0 lähtien [Android 5.0, 2015]. Ominaisuutta hyödyntämällä sovelluksilla on kyky nähdä kaikki sovellukset, mitä laitteessa käytetään. Androidiin tehtyjä sovelluksia on niin monia, että käytettyjä sovelluksia analysoimalla käyttäjästä voi saada paljonkin henkilökohtaista tietoa ulos. Ilman tätä ominaisuutta monen käytönvalvontaohjelmiston yksi tärkeimmistä ominaisuuksista ei ole enää käytettävissä.

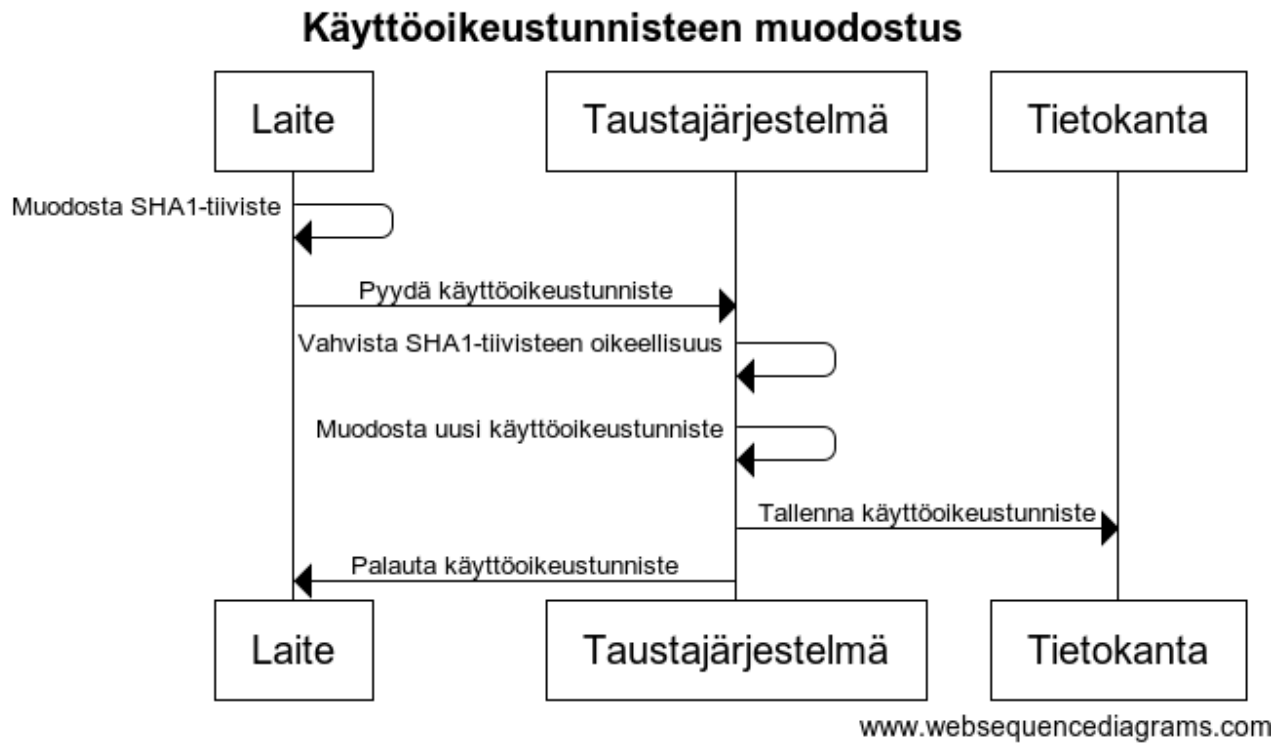
Järjestelmän uudelleenkäynnistyksen tarkkailu mahdollistaa sovelluksen käynnistyvän aina järjestelmän käynnistyksen yhteydessä. Tämä on oleellinen ominaisuus, jotta lapsi ei voi sammuttaa

puhelinta ja käynnistää sitä uudelleen, päästen näin pelaamaan yli vanhempiensa määrittämien rajojen. Oikeus ei altista käyttäjää yksityisyyden tai tietoturvan uhkille, mutta antaa käyttäjän tietää, että sovellus on aina taustalla käynnissä.

Ulkoisilta tallennuslaitteilta lukemista tarvitaan, kun sovellukseen lisätään käyttäjiä ja niille halutaan antaa profiilikuva. Valokuvat usein tallennetaan ulkoisille tallennuslaitteille, josta käyttäjälle annetaan mahdollisuus valita sopiva kuva käyttäjälle. Androidin tietoturva-arkkitehtuurin vuoksi kolmannen osapuolen sovelluksilla ei ole ikinä mahdollisuutta kuitenkaan lukea muiden sovellusten tallentamaa sisäistä tietoa. Ulkoisille tallennuslaitteille on tarkoitettu tallennettavan ainoastaan julkiseksi tarkoitettavaa tietoa. Tämä ei välttämättä ole kuitenkaan kaikille käyttäjille selvää. Käyttäjä saattaa luulla, että sovellus voi lukea mitä tahansa tietoa, mitä käyttäjän laitteeseen on tallennettu.

Mahdollisuus lukea käyttäjän paikannustietoja tarvitaan lapsen sijaintitietojen tallentamista varten. Sijaintitietoja ei lueta kuin laitteissa, joihin on määritelty lasten profiili ja paikannusominaisuus on kytketty päälle. Paikannuslaitteen käyttöoikeutta on silti vaadittu myös vanhempien puhelimiin asennettuihin sovelluksiin. Sovellus saattaa tallentaa kaikki käyttäjän sijainnit ja välittää esimeriksi ulkopuoleisille tahoille. Ominaisuus vaatii kuitenkin toimiakseen, että laitteessa on GPS- tai WLAN-toiminto kytkettynä päälle.

Siitä huolimatta, että mitään tietoja ei luovutettaisi tarkoituksella ulkopuolisille tahoille, on mahdollista, että joku murtautuu taustajärjestelmään ja varastaa kaikki tiedot. Jotta ulkopuoliset tahot eivät pääse käsiksi taustajärjestelmän laitteille avoimiin rajapintoihin hakeakseen muiden käyttäjien tietoja, on se täytynyt suojata. Prototyypin ensikäynnistyksen yhteydessä sovellus pyytää taustajärjestelmältä käyttöoikeustunnistetta, minkä avulla kaikki pyynnöt valtuutetaan. Käyttöoikeustunnisteen pyyntö tapahtuu siten, että sovellus lukee laitteelle yksilöllisen tunnisteen ja yhdistää sen taustajärjestelmän tuntemaan salaiseen avaimeen, minkä jälkeen näiden kombinaatio salataan SHA1-salauksella. Salattu avain-tunniste-pari lähetetään selväkielisen laitteen tunnisteen kanssa taustajärjestelmään. Taustajärjestelmä varmistaa, että lähetetty laitteen tunniste ja salainen avain muodostaa saman SHA1-tiivisteen, kuin laitteen lähettämä. Mikäli tiivisteet täsmäävät, taustajärjestelmä muodostaa käyttöoikeustunnisteen ja tallentaa sen tietokantaan. Käyttöoikeustunnisteelle asetetaan samalla voimassaoloaika, minkä jälkeen muodostettu tunniste ei enää toimi. Prototyyppi pyytää uuden tunnisteen automaattisesti, mikäli taustajärjestelmä palauttaa vastauksen, että käyttöoikeustunniste on vanhentunut (Kuva 15).



Kuva 15. Sekvenssi-diagrammi käyttöoikeustunnisteen pyytämisestä taustajärjestelmältä.

6.3. Käytönvalvonnan tulevaisuus

Tällä hetkellä ohjelmistokehittäjien mahdollisuudet toteuttaa käytönvalvontasovelluksia Android-laitteisiin tai mihinkään muuhunkaan laitteeseen on hyvin heikko. Mahdollisia tulevaisuudennäkymiä on erilaisia. Yhdessä mallissa käytönvalvonta on lähes täysin käyttöjärjestelmätasolle toteutettuna. Toisessa mahdollisessa vaihtoehdossa kolmannet osapuolet voisivat saada käytönvalvontaoikeudet erillistä hakemusta vastaan. Kolmas vaihtoehto on hajauttaa käytönvalvonta täysin sovelluskohtaiseksi.

Käyttöjärjestelmätasolle toteutetun käytönvalvonnan etuna on, että sillä on tarpeelliset oikeudet hallita kaikki tilanteet haluamallaan tavalla, esimerkiksi sulkemaan sovelluksen, kun käyttöaika tulee umpeen tai estämään koko sovelluksen käynnistämisen. Käyttöjärjestelmätason käytönvalvonnalla olisi myös mahdollisuus tarkkailla täydellisesti koko verkkoliikennettä, salattuja yhteyksiä lukuun ottamatta. Tilanne, jossa vanhempi pystyisi tarkastelemaan lastensa viestintäohjelmistojen välityksellä tapahtuvaa yksityiskohtaista keskustelua, voi olla kuitenkin eettisesti arveluttavaa ja saattaa loukata lapsen yksityisyyttä. Eettinen raja vanhemman ja lapsen välisessä valvonnassa on häilyvä. Vanhemmat haluavat pitää huolen, ettei lapsi pääse altistumaan haitalliselle sisällölle tai ole tekemisissä epäilyttävien ihmisten kanssa. Toisaalta lapsenkin yksityisyyttä pitää varjella, koska hänellä on täysi oikeus käydä yksityiskeskusteluita aivan kuten

vanhemmillaankin. Sovellustasolla ongelman voisi ratkaista siten, että viestintäohjelman kautta käytävistä keskusteluista tallennettaisiin ainoastaan ajankohdat ja osallistujat vanhempien luettavaksi. Käyttöjärjestelmätasolle tehdyllä käytönvalvontasovelluksellakaan ei kuitenkaan ole tarpeeksi tietämystä kolmansien osapuolten sovellusten yksityiskohdista, mikä tekee kyseisten tietojen saamisesta mahdotonta ilman ennalta määrättyjä standardeja ja rajapintoja. Sama pätee erilaista mediasisältöä tuottavien sovellusten, kuten selain ja kuvapalvelut, sisällön suodatukseen. Jokaisen kuvapalvelun ja jokaisen Internet-sivuston pitäisi pystyä tarjoamaan käytönvalvontaohjelmistolle vaadittava tietämys tarjotusta sisällöstä, jotta voitaisiin tehdä oikea päätös suodatuksen tarpeesta. Näiden lisäksi käyttöjärjestelmätason toteutuksen heikkoutena on, että sen pitäisi olla tarpeeksi monipuolinen kaikkien käyttäjien tarpeisiin. Käytönvalvonnan vastuun asettaminen yhdelle sovellukselle olisi siis tällä hetkellä hyvin hankala toteuttaa täysin automatisoidusti. Valvonnan päävastuu pitää olla vanhemmilla, joiden tulisi tehdä päätökset sallituista sovelluksista ja Internet-sivustoista. Sisällön suodatusta lukuun ottamatta käyttöjärjestelmätason käytönvalvonta voisi toimia hyvin, mikäli sen monipuolisuuteen olisi myös panostettu tarpeeksi.

Mahdollisuus tarjota kolmansille osapuolille kyky toteuttaa käytönvalvontaa, toisi mukaan kilpailua, jonka myötä tulisi uusia ideoita ja erilaisia lähestymistapoja ratkaista käytönvalvonnan ongelmia. Sen lisäksi käyttäjille olisi tarjolla useita vaihtoehtoja valita juuri heille sopivin sovellus käytönvalvontaan. Kuten edellisessä kappaleessa kuitenkin todettiin, kolmansille osapuolille liiallisten valvontaoikeuksien tarjoaminen voi olla tietoturvan ja yksityisyyden kannalta huono asia. Tästä johtuen käyttöjärjestelmä voisi tarjota kehittäjille mahdollisuutta hakea sertifikaattia, joka mahdollistaisi pääsyn järjestelmän suojattuihin rajapintoihin. Kriteereitä sertifikaatin myöntämiselle ei käsitellä tässä tutkielmassa. Sisällön suodattamisen osalta tämä lähestymistapa kohtaa samat ongelmat kuin käyttöjärjestelmätasolle tehty käytönvalvontaohjelmisto.

Hajautetussa mallissa käytönvalvonta olisi jossain määrin toteutettuna lähes jokaisessa sovelluksessa. Oikeastaan tämä malli ainoastaan täydentää sisällön suodatuksen osalta kahta edellä mainittua lähestymistapaa. Android-käyttöjärjestelmän versiosta 4.3 lähtien taulutietokoneissa on toteutettu käytönvalvonta juuri tätä mallia käyttäen. Androidin käyttöjärjestelmätasolle on luotu kehys, mikä pystyy valvomaan kaikkia sovelluksia ja hallitsemaan käyttäjien tunnistautumiset. Sen lisäksi kaikille sovelluksille tarjotaan mahdollisuus tuoda käytönvalvonta-asetuksia omaan sovellukseensa. Tässä herää jälleen kysymys, miksi kehittäjät haluaisivat nähdä vaivaa tehdäkseen ylimääräisiä asetusnäkymiä ja käytönvalvontatarkistuksia. Internet-suodatusta helpottamaan luotu ICRA-järjestelmä ei toiminut, koska hyvin harvat kehittäjät halusivat lisätä luokituksia omille sivustoilleen. Haluttomuus johtui pääosin siitä, että käyttäjäliikenne olisi ohjautunut hyvin helposti kilpaileville sivustoille, jonne ei luokituksia ole lisättynä. Jos yksi viestinvälityssovellus tarjoaa vanhemmille jonkinlaiset keinot valvoa lastensa viestittelyä, lapset todennäköisesti alkaisivat käyttämään sovellusta, jossa valvonta ei ole mahdollista. Toisen viestintäsovelluksen asennus ei tietenkään ole mahdollista, jos vanhemmat ovat estäneet kaikkien sovelluskauppojen käytön. Toisaalta myös vanhemmat, jotka näkevät käytönvalvonnan tärkeänä, asentavat todennäköisesti

sovelluksen, joka sisältää käytönvalvonnan. Asiaa monimutkaistaa entisestään se, että viestintäsovelluksen ideana on keskustella ihmisten kanssa, jotka käyttävät samaa sovellusta. Lapset, jotka voivat käyttää ainoastaan käytönvalvonnalla tuettua viestintäsovellusta, saattavat jäädä kaveripiirin ulkopuolelle, mikäli muut lapset eivät käytä samaa sovellusta. Käyttöjärjestelmän tulisi pakottaa kaikille sovelluksille käytönvalvonnan ominaisuudet, jotta ominaisuus tulisi tehtyä jokaiseen sovellukseen. Sen lisäksi, ettei käytönvalvontaa kuitenkaan tarvita aivan jokaiseen sovellukseen, käytönvalvontaominaisuuksien toteutuksen toimivuuden ja kattavuuden valvominen olisi erittäin työlästä. Yksinkertaista ja teknisesti sekä eettisesti toimivaa mallia saamme todennäköisesti vielä odottaa.

7. Yhteenveto

Tässä tutkielmassa käsiteltiin käytönvalvonnan toteutuksen haasteita ja ratkaisumenetelmiä Android-käyttöjärjestelmissä. Tutkimusmenetelmänä toteutettiin käytönvalvontaohjelmiston prototyyppi, johon toteutettiin käytönvalvonnassa usein esiintyviä ominaisuuksia. Toteutuksissa kiinnitettiin erityisesti huomiota ratkaisun toimivuuteen, mutta myös käytettävyyteen ja joustavuuteen. Näiden lisäksi pohdittiin muiden ratkaisumallien toimivuutta ja käytönvalvonnan tulevaisuutta.

Toimivan käytönvalvontaohjelmiston toteuttaminen todettiin mahdolliseksi, vaikkakin tietyin rajoituksin. Osa käytönvalvonnan ominaisuuksista on jollain tapaa kierrettävissä ja osaan toteutuksista jouduttiin käyttämään menetelmiä, mitkä saattavat lakata toimimasta tulevaisuuden Android-versioissa. Monen toteutuksen käyttöönotto on monimutkainen toimenpide loppukäyttäjälle, minkä johdosta koko sovelluksen käytettävyys on huono. Toimivaa kolmannen osapuolen sisällönsuodatusta on erittäin haasteellista toteuttaa vielä nykypäivänäkään. Android-kehittäjien valitsema linja tuntuisi kuitenkin oikealta suunnalta mobiililaitteiden käytönvalvonnan tulevaisuutta ajatellen.

Viiteluettelo

[Ableson et al., 2011] W. Frank Ableson, Robi Sen, Chris King and C. Enrique Ortiz, *Android in Action 3rd edition*, Manning Publications Co., 2011.

[Amadeo, 2013] Ron Amadeo, *Google's iron grip on Android: Controlling open source by any means necessary*, 2013. Available at http://arstechnica.com/gadgets/2013/10/googles-iron-grip-on-android-controlling-open-source-by-any-means-necessary. Checked 9.12.2014.

[Android, 2015] Android, *About Android*, 2015. Available at http://developer.android.com/about/index.html. Checked 3.3.2015.

[Android 4.3, 2013] Android, *Jelly Bean 4.3*, 2013. Available at http://developer.android.com/about/versions/jelly-bean.html. Checked 10.6.2015.

[Android 5.0, 2015] Android, *Android 5.0 Behavior Changes*, 2015. Available at https://developer.android.com/about/versions/android-5.0-changes.html. Checked 15.3.2015.

[Android Back Stack, 2015] Android, *Tasks and Back Stack*, 2015. Available at http://developer.android.com/guide/components/tasks-and-back-stack.html. Checked 10.6.2015.

[Android Terms and Conditions, 2012] Android, *Terms and conditions*, 2012. Available at https://developer.android.com/sdk/terms.html. Checked 3.6.2015.

[Android Studio, 2015] Android, *Android Studio*, 2015. Available at https://developer.android.com/tools/studio/index.html. Checked 19.5.2015.

[Android WebView, 2015] Android, *Building Web Apps in WebView*, 2015. Available at http://developer.android.com/guide/webapps/webview.html. Checked 7.6.2015.

[Archer, 2009] Phil Archer, *ICRAfail*, 2009. Available at http://philarcher.org/icra/ICRAfail.pdf. Checked 6.4.2015.

[Brady, 2008] Patrick Brady, *Anatomy & Physiology of an Android*, 2008. Available at https://sites.google.com/site/io/anatomy--physiology-of-an-android. Checked 25.3.2015.

[Common Sense Media, 2013] Common Sense Media, *Zero to eight*, 2013. Available at https://www.commonsensemedia.org/sites/default/files/research/zero-to-eight-2013.pdf. Checked 7.9.2014.

[F-Secure, 2014] F-Secure, *Mobile Threat Report Q1 2014*, 2014. Available at https://www.f-secure.com/documents/996508/1030743/Mobile_Threat_Report_Q1_2014.pdf. Checked 9.12.2014.

[Gartner, 2014] Gartner, *Gartner Says Sales of Tablets Will Represent Less Than 10 Percent of All Devices in 2014*. 2014. Available at http://www.gartner.com/newsroom/id/2875017. Checked 8.12.2014.

[Gaudin, 2015] Kevin Gaudin, *Application crash reports for Android*, 2015. Available at http://www.acra.ch/. Checked 15.3.2015.

[Gson, 2015] Open source project, *Gson*, 2015. Available at https://code.google.com/p/google-gson/. Checked 15.3.2015.

[Google Maps, 2015] Google Maps, *Shapes*, 2015. Available at https://developers.google.com/maps/documentation/android/shapes. Checked 3.6.2015.

[ICRA, 2015] Family Online Safety Institute, *ICRA*, 2015. Available at https://www.fosi.org/icra/. Checked 15.4.2015.

[Heart Research Associates, 2011] Heart Research Associatest, *Who Needs Parental Controls?*, 2011. Available at https://www.fosi.org/documents/77/Who_Needs_Parental_Controls_Survey_Findings.pdf. Checked 19.6.2015.

[LittleProxy, 2011] LittleShoot, *LittleProxy*, 2011. Available at http://www.littleshoot.org/littleproxy/. Checked 3.6.2015.

[Lockheimer, 2012] Hiroshi Lockheimer, *Android And Security*, 2012. Available at http://googlemobile.blogspot.fi/2012/02/android-and-security.html. Checked 3.3.2015.

[MSDN, 2015] Microsoft developer network, *GetForegroundWindow function*, 2015. Available at https://msdn.microsoft.com/en-us/library/ms633505%28VS.85%29.aspx?ppud=4. Checked 14.4.2015.

[OpenSignal, 2014] OpenSignal, *Android Fragmentation Visualized*, 2014. Available at http://opensignal.com/reports/2014/android-fragmentation/. Checked 3.3.2015.

[Poiesz, 2013] Benjamin Poiesz, *Android Official, Find your lost phone with Android Device Manager*, 2013. Available at http://officialandroid.blogspot.fi/2013/08/find-your-lost-phone-with-android.html. Checked 16.3.2015.

[Raphael, 2012] JR Raphael, *Inside Android 4.2's powerful new security system*, 2012. Available at http://www.computerworld.com/article/2473570/android/exclusive--inside-android-4-2-s-powerful-new-security-system.html. Checked 3.3.2015.

[Statista, 2014] Statista, *Number of available applications in the Google Play Store from December 2009 to July 2014*, 2014. Available at http://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/. Checked 9.12.2014.

[Spiegel, 2013] Spiegel Online International, *NSA Can Spy on Smart Phone Data*, 2013. Available at http://www.spiegel.de/international/world/privacy-scandal-nsa-can-spy-on-smart-phone-data-a-920971.html. Checked 16.3.2015.

[The Guardian, 2014] The Guardian, *Angry Birds and 'leaky' phone apps targeted by NSA and GCHQ for user data*, 2014. Available at http://www.theguardian.com/world/2014/jan/27/nsa-gchq-smartphone-app-angry-birds-personal-data. Checked 25.3.2015.

[Visionmobile, 2013] Vision Mobile, *Developer segmentation*, 2013.

[W3Schools, 2014] W3Schools, *Mobile Devices Statistics*, 2014. Available at http://www.w3schools.com/browsers/browsers_mobile.asp. Checked 7.12.2014.

[Windows Vista, 2007] Microsoft, *Windows Vista Fact Sheet*, 2007. Available at http://www.microsoft.com/presspass/newsroom/windows/factsheets/WindowsVistaFS.mspx. Checked 26.3.2015.