# Utvecklingsproblem - vår historia

**Nätverk**
En fråga vi ställde oss tidigt var hur två mobiltelefoner ska kunna utbyta data med varandra. Under projektets gång har vi kommit på ett flertal lösningar, alla med sina egna unika problem. Dessa problem har upptaget mycket av vår tid. Nedan följer en lista över de versioner vi har prövat.

*1. Server v.1*
Vår första idé var att låta att data mellan telefonerna skickas via vår server. Varje telefon skulle regelbundet ta kontakt med servern och tala om sin IP-adress. QR-koden hos varje telefon var tänkt att vara en unik kod som identifierar användaren. När en telefon (A) hade skannat användarkoden på en annan (B), skulle A skicka den skannade koden till servern. Servern kan då se vilken IP den användaren var ansluten till senast han var aktiv och kan då sända ett broadcast-meddelande till den adressen. När B tar emot broadcasten kan han se användarkoden för att se om han avsågs vara mottagaren. Vi hade tänkt att servern har hand om profilmatchningen och skickar gemensamma intressen till A och B. Problemet med denna idé var att vi tyckte servern fick för mycket ansvar samt att den var för komplex, eftersom det skickades så pass mycket data fram och tillbaka, med omständiga protokoll.

*2. WiFi-direct*
Den andra idén var att använda oss av WiFi-direct (https://en.wikipedia.org/wiki/Wi-Fi_Direct) för att ansluta direkt mellan enheterna, utan att blanda in servern. Servern skulle inte behöva lagra varje användares IP-adress eller profil, profilmatchningen kunde ske i telefonerna istället. Denna idé verkade lovande, men eftersom Androids P2P API var ganska svårt att sätta sig in i tog det lång tid innan vi kom igång med det. När vi läste mer om hur WiFi-direct identifierar användare insåg vi att vi inte kunde använda oss av det, eftersom vi inte på automatisk väg kunde identifiera vilken enhet att anluta till. Varje enhet identifieras av ett namn som användaren kan ändra. Problemet är då dels att namnet inte är unikt och även att A inte vet vilket namn han ska ansluta till.

*3. Direktkoppling via wlan*
Vi insåg att om man känner till den andra enhetens IP-adress blir det mycket enklare att ansluta mellan A och B. Vi lät då QR-koden genereras från IP-adressen istället för användarkoden. Då kunde A direkt öppna en socket och ansluta till B när den skannat och läst av Bs adress. Den här idén är den bästa vi lyckades komma på, men även den har ett stort problem. Det är inte alla nätverk som tillåter sådana anslutningar som vi försöker uprätta mellan två enheter på samma nätverk. När vi testade på skolan över nätverket eduroam gick det inte att upprätta någon anslutning alls. När vi testade på Electricitys nät uppstod helt andra problem. Ibland kunde vi koppla ihop två telefoner och ibland gick det inte. Vi tror att detta beror på hur ElectriCitys WiFi är uppbyggt. Eftersom det är uppbyggt av flera routrar och accesspunkter som

bildar ett virtuellt nätverk, så misstänker vi att routingtabellen sparas efter din första anslutning, och att routingen därför misslyckas när du är ansluten till en annan router eller accesspunkt.

*4. Server v.2*
När vi insåg att det fanns problem med att koppla ihop två enheter som båda var på ElectriCitys nät, tänkte vi att det kanske var bäst att låta trafiken gå via servern trots allt. Vår fjärde idé gick ut på att A skulle skicka den skannade IP-adressen samt sin egen data till servern som sedan skickar datan vidare till den just mottagna adressen. När B tagit emot datan skickar han motsvarande data till A via servern. När vi testade den här lösningen fungerade det väldigt bra på våra egna nätverk, men det gick inte att upprätta anslutningar till en enhet på ElectriCitys nät utifrån. Vi gick då tillbaka till att använda vår tredje lösning.

*5. Server v.3*
Vi har spekulerat i en femte lösning (vi har inte hunnit realisera den än) då vi låter varje enhet med jämna mellanrum (någon sekund) kontaktar servern och kolla om någon har skannat. Då ska alltså inte servern öppna anslutningar till telefonerna, bara tvärtom. A sänder alltså ett meddelande som lagras på servern tills B frågar servern efter inkommna meddelanden.

**Användarkod**
Under projektets gång har vi spekulerat i hur vi genererar användarkoden. Koden ska vara unik för varje användare och ska vara densamma om användaren byter mobil.

1. Vår första idé var att ta en unik kod från telefonen (IMEI eller MAC) och kryptera den för att få användarkoden.
    - Problemet med detta var att användaren får en ny kod då han byter mobil. Om vi låter all data säkerhetskopieras till servern kan man alltså aldrig återfå den informationen.
2. Vi fick då idén att generera koden från användarens telefonnummer. Detta är unikt, och numret behålls när man byter mobil.
    - Probemet med detta var att alla operatörer inte tillät att man kan ta reda på telefonnummret från SIM-kortet.
3. Den lösningen som vi till slut använde oss av var att kryptera användarens Google-konto. För att kunna ladda ner appen från Google play är man tvungen att ha ett sådant konto, så vi vet garanterat att varje användare har ett. Den är unik, behålls vid telefonbyte och är enkel att ta reda på.

**Utmärkelser**
Utmärkelsesystemet har varit med som idé från första början. Vår första version var väldigt uppdelad i olika klasser och subklasser. När systemet började fungera för första gången ställdes frågan om vi verkligen behövde alla subklasser. Svaret blev nej, då vi inte ännu hade kommit på olika typer av utmärkelser. Systemet skrevs om så att det bara finns en klass “Achievement” samt en klass för krav, “Demand”, och det är så det ser ut idag. Tyvärr har det visat sig att vår första idé var mycket bättre, då vi nu behöver skriva om befintlig kod när nya

typer av utmärkelser ska implementeras. Om vi ska ta appen vidare behöver stora delar av koden skrivas om i Achievement-paketet.

**Profilen**

Vi har haft flera idéer om hur utseendet på fönstret där man skriver in sin profilinformation skulle se ut. Först var det tänkt att profilfönstret skulle visa de olika kategorierna (musik, film m.m.) och varje kategori var tänkt att ha sitt eget fönster där man kan skriva. När det visade sig vara för svårt att realisera, skapade vi ett gränssnitt med bara ett fönster, vilket var betydligt enklare att programmera och designa, men som inte var lika användarvänligt. Inlägg måste separeras med kommatecken, något som vi från början vill undvika då det inte är intuitivt för alla.