Låsapplikation för Android

LUNDS
UNIVERSITET
Lunds Tekniska Högskola

LTH Ingenjörshögskolan vid Campus Helsingborg
Datateknik

Examensarbete:
Andreas Alsensjö
Duong Tran

LTH Ingenjörshögskolan vid Campus Helsingborg
Lunds universitet
Box 882
251 08 Helsingborg

LTH School of Engineering
Lund University
Box 882
SE-251 08 Helsingborg
Sweden

Tryckt i Sverige
Media-Tryck
Biblioteksdirektionen
Lunds universitet
Lund 2012

Sammanfattning

I detta examensarbete utvecklas en applikation för Android som skall kunna låsa upp och låsa dörrlås. Detta görs via telefonen, som kommunicerar med en server på internet. Denna server kommunicerar med ett Arduinokort som i sin tur kommunicerar med låset.

Telefonen har en applikation, som är gjord för Android version 1.6 och senare. För att detta skall fungera krävs det att telefonen har tillgång till internet.

Servern använder Pachube som protokoll för kommunikation för tillfället, men skall med tiden ersättas med Pubsub.

Ett Arduinokort styr motorn till låset, det är uppkopplat mot servern via Pachube som säger till när låset skall ändra status. Det finns många olika Arduinokort. I detta arbete används ett kretskort som heter UNO, som kompletterades med en Ethernet Shield för att kunna komma ut på webben.

Låset måste vara av typen cylinderlås för att det skall vara möjligt att koppla på motorn som vrider om låset.

Detta examensarbete utfördes under en tolv veckors period och resultatet blev en fungerande prototyp.

Nyckelord: Android, Arduino, Lås, Pachube, Pubsub, Applikation

Abstract

In this final degree project, a system for an Android application that should be able to lock and unlock door locks has been developed. This will be done by using a mobile phone that is communicating with an online server, this server is communicating with the Arduino board which in turn is communicating with the door lock.

The phone has an application made for Android version 1.6 and after. To make this application work the phone must have access to the Internet.

The server is using Pachube as the protocol for the communication at this moment but will shortly be replaced by Pubsub.

The Arduino board is the part of the system that controls the physical lock. It is connected to the server by Pachube that tell when the lock needs to change status. There are many different Arduino boards. In this system an Arduino UNO board is used with an Ethernet Shield to get access to the Internet.

The lock needs to be a type of cylinder lock, this is needed to be able to hook the part that lock and unlock the door lock.

This final degree project was performed under a twelve week period and the result was a working prototype.

Keywords: Android, Arduino, Lock, Pachube, Pubsub, Application

Innehållsförteckning

1 Inledning

1.1 Bakgrund

Examensjobbet görs för Dunderbar AB i Malmö, ett nystartat företag som tidigare utvecklat en applikation för iPhone[1] som låser och låser upp dörrlås. Dunderbar grundades 2011 med visionen att en uppkopplad värld är en smartare värld. Företagets syfte är att utveckla och driva olika applikationsbaserade lösningar för uppkopplade produkter. Den första produkten är det internetuppkopplade låset för iPhone. I versionen till iPhone användes Pachube som protokoll. Problemet med den applikationen var att låset inte låste eller låste upp i realtid. Det saknades ett protokoll som kunde kommunicera bättre i realtid. Tanken var att Pubsub skulle lösa dessa problem och fungerade det bra skulle även iPhone versionen gå över till Pubsub.

Det unika i detta arbete är att Arduino, Pachube och Pubsub är open source.

Det finns många idéer på internet om att kunna öppna dörrar med en mobil applikation. En idé är från företaget Zaplox[2], där användningsområdet är tänkt att vara bredare t.ex. vid boende på hotell. Där får användaren något som Zaplox kallar “mobile key”, en nyckel till låset som är giltig under en viss tid. Detta skiljer sig från Dunderbars lösning. Dunderbar har en permanent nyckel till ett lås. Detta begränsar användningsområdet då en användare alltid kommer ha möjlighet att öppna eller låsa ett lås, även till exempel efter ett hotellbesök skulle det vara möjligt att öppna och låsa dörren.

En annan konkurrent är ASSA ABLOY[3] som har en lösning som går ut på att telefonen ersätter nyckeln. Det fungerar som så att användaren håller telefonen nära en dosa monterad på väggen bredvid låset. Själva kommunikationen sker med hjälp av NFC, Near Field Communication[4]. Detta kräver att användaren är på plats för att kunna öppna dörren vilket är en begränsning för låset om låset skall öppnas när användaren inte är på plats. Företagen är hemlighetsfulla med hur de har implementerat lösningen utan visar bara att det fungerar och hur användaren gör för att använda det.

Den största skillnaden är just att detta system länkas ihop med open source hårdvara och mjukvara som är tillgängligt för alla. Detta gör att denna lösning är betydligt billigare än andra konkurrenter. Priset för ett UNO-kort är ungefär 200 svenska kronor och för en Ethernet Shield ungefär 300 svenska kronor.

1.2 Syfte och Mål

Dunderbar ville utöka sin tjänst till ytterligare en plattform, Android, och eliminera fördröjningar i kommunikationen . Tanken var att med en färdig lösning på applikationen skulle Pubsub och Pachube jämföras för att se vilket av protokollen som var bäst. Applikationen skall vara användarvänlig och lättanvänd. Det skulle vara tydliga menyer som inte skulle kunna missuppfattas.

Slutmålet är att få en fungerande prototyp som är så pass stabil att den aldrig kraschar vid eventuella felinmatningar eller feltryck.

Vid leveransen av applikationen var en mycket hög prioritet att en icke teknisk kunnig person enkelt skall kunna installera och använda applikationen tillsammans med låset.

1.3 Problemformuleringar

Att göra en låsapplikation till Android och att få telefon och Arduino att kommunicera över internet. Denna kommunikation ska göras i realtid med Pubsub som protokoll. Applikationen skall vara så användarvänlig som möjligt.

Detta examensarbete har utgått från följande frågeställningar:

* Finns det något liknande?
* Vilket protokoll fungerar bäst?
* Kommer Pubsub lösa problemet med fördröjningar?
* Fungerar Arduino Ethernet med detta projekt?
* Hur blir applikationen så användarvänlig som möjligt?

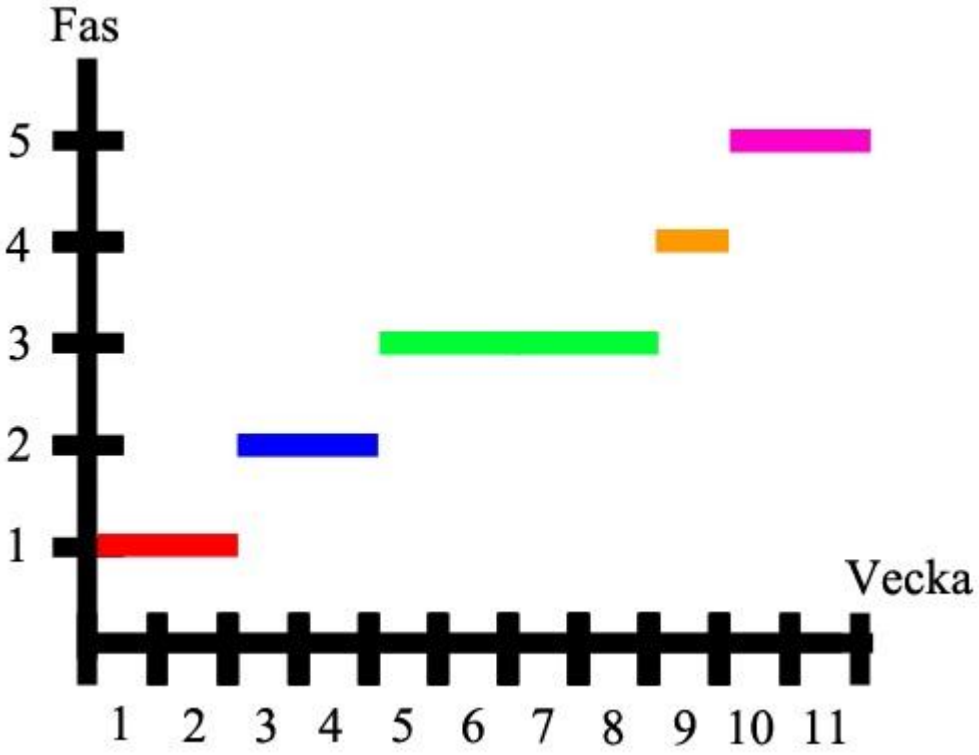
1.4 Avgränsningar

Dunderbar har en bild framför sig att när detta system är komplett skall låset kunna känna av om dörren är stängd eller låset vridits om manuellt med nyckel. Detta ska lösas med hjälp av sensorer.

I detta examensarbete ville företaget endast ha en fungerande prototyp som klarade att skicka kommando till låset från telefonen via servern. Det ingår endast i detta examensarbete att servern kommunicerar med låset, inte tvärtom.

2 Metod

Detta examensarbete har utvecklats via en iterativ metodik under tolv veckor i nära kontakt till handledaren med träffar minst en gång i veckan.
Examensarbetet utvecklades under fem faser enligt Figur 1.



Figur 1. Tidsdiagram på hur lång tid det tog för varje fas.

2.1 Fas 1 – Bli bekanta med Arduino

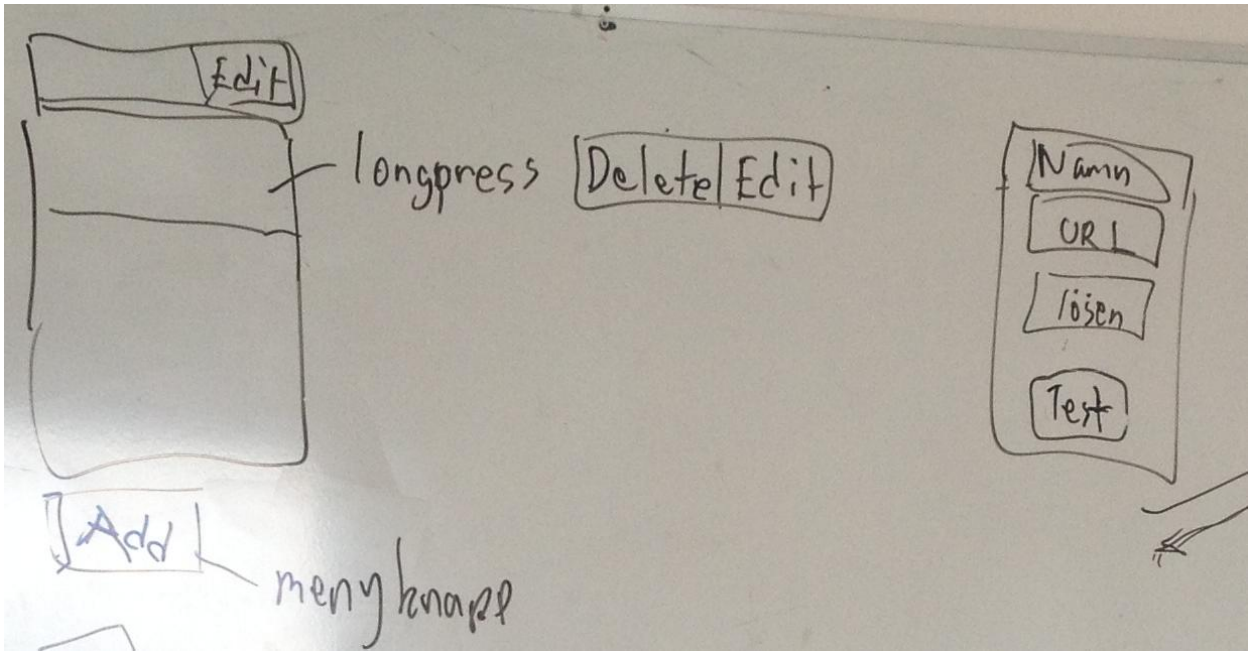
Första två veckorna gick ut på att bekanta sig med Arduino[5] och dess utvecklingsmiljö. Att få enkla exempel med Arduino och dioder att fungera.

2.2 Fas 2 – Hämta webbsida och skicka data till server

Nästa steg i utvecklingen var att förstå kommunikationen mellan en telefon och servern via internet bättre.
Handledaren satte upp en hemsida på servern som användes för att testa om det gick att hämta och skicka data till denna.

2.3 Fas 3 – Skapa Androidapplikation

När applikationen skulle börja utvecklas uttryckte handledarna en önskan om hur det skulle se ut vid leverans, som Figur 2 visar. Hur detta sedan uppnåddes fick examensarbetarna själv välja.



Figur 2. Handledarens önskemål på hur applikationen skall se ut.

2.4 Fas 4 – Testning mot ett riktigt lås

När applikationen bedömdes testbar anordnades ett hackaton där applikationen testades mot ett riktigt lås. Här var tanken också att eventuella fel skulle upptäckas.

2.5 Fas 5 – Stabilisera applikation och leverera färdig produkt

Efter testning mot ett riktigt lås återstod det att rensa applikationen från buggar och krascher för att sedan leverera en färdig produkt redo för Android Market.

2.6 Källkritik

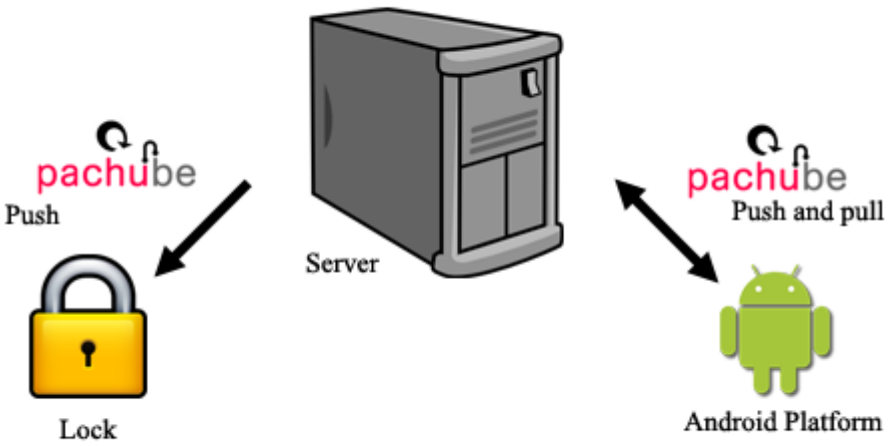
Den viktigaste källan i detta examensarbete är developer.android.com, Pachube.com, Pubsub.io och Arduino.cc som är hemsidor skapade av företagen själv. Därför anses dessa hemsidor som väldigt pålitliga.

Hemsidan developer.android.com är gjord av Android för utvecklare och var den viktigaste källan under arbetets gång. Det finns där specifikationer för hur alla klasser fungerar och vilka metoder som finns. I början ansågs denna källa vara väldigt pålitlig, detta bekräftades också genom att de klasser och metoder som valts till applikationen fungerar som väntat.

3 Teknisk bakgrund

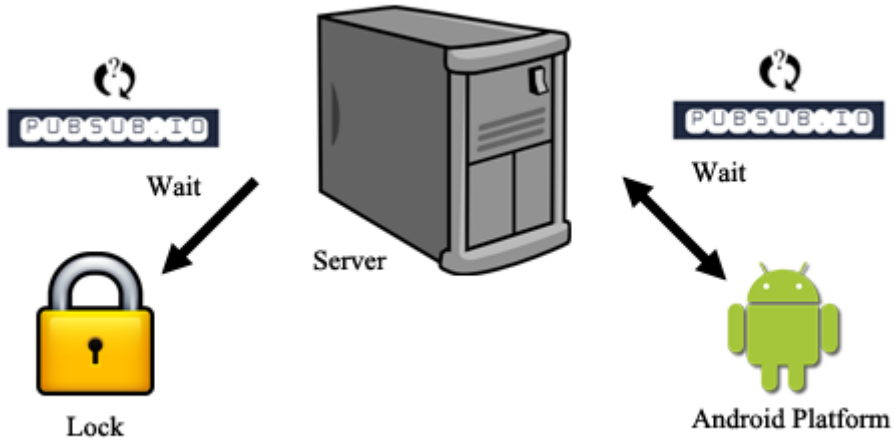
3.1 Pachube och Pubsub

Pachube skapades 2007 av arkitekten Usman Haque[6]. Pubsub var menat att bli redo för användning under våren men komplikationer uppstod och det beräknas bli färdigt under sommaren 2012[7].
Pachube är ett protokoll på transportlagret i OSI-modellen och Pubsub är ett protokoll på applikationslagret i OSI-modellen. Dessa protokoll får hårdvara att kommunicera med internet och denna kommunikation skall vara i realtid. Tanken är att en användare skall kunna styra enheter och ändra värden i script på en server med hjälp av dessa protokoll.



Figur 3. Systemet med pachube som protokoll för kommunikation. Dubbelriktad kommunikation mellan server och telefon, enkelriktad kommunikation mellan server och lås.

Pachube fungerar genom att servern skickar och tar emot data efter en viss tid som väljs av användaren, det är detta Figur 3 visar. Push används när applikation eller server vill göra något, till exempel att låsa låset. Pull används när telefonen behöver något från servern.
I detta fall med ett lås måste användaren kunna uppdatera tillräckligt ofta. Varje gång pull eller push görs skapas en ny TCP-koppling mellan inblandade parter. Detta gör att onödigt mycket datatrafik uppstår med detta protokoll.



Figur 4. Systemet med Pubsub som protokoll för kommunikation. Dubbelriktad kommunikation mellan server och telefon, enkelriktad kommunikation mellan server och lås.

Handledaren har haft kontakt med grundaren av Pubsub som förklarade skillnaden mellan Pachube och Pubsub enligt följande:

Pubsubs idé är att göra kopplingen en gång och sedan hålla denna uppe och vänta tills något har ändrats istället för att göra pull och push efter en viss tidskonstant.

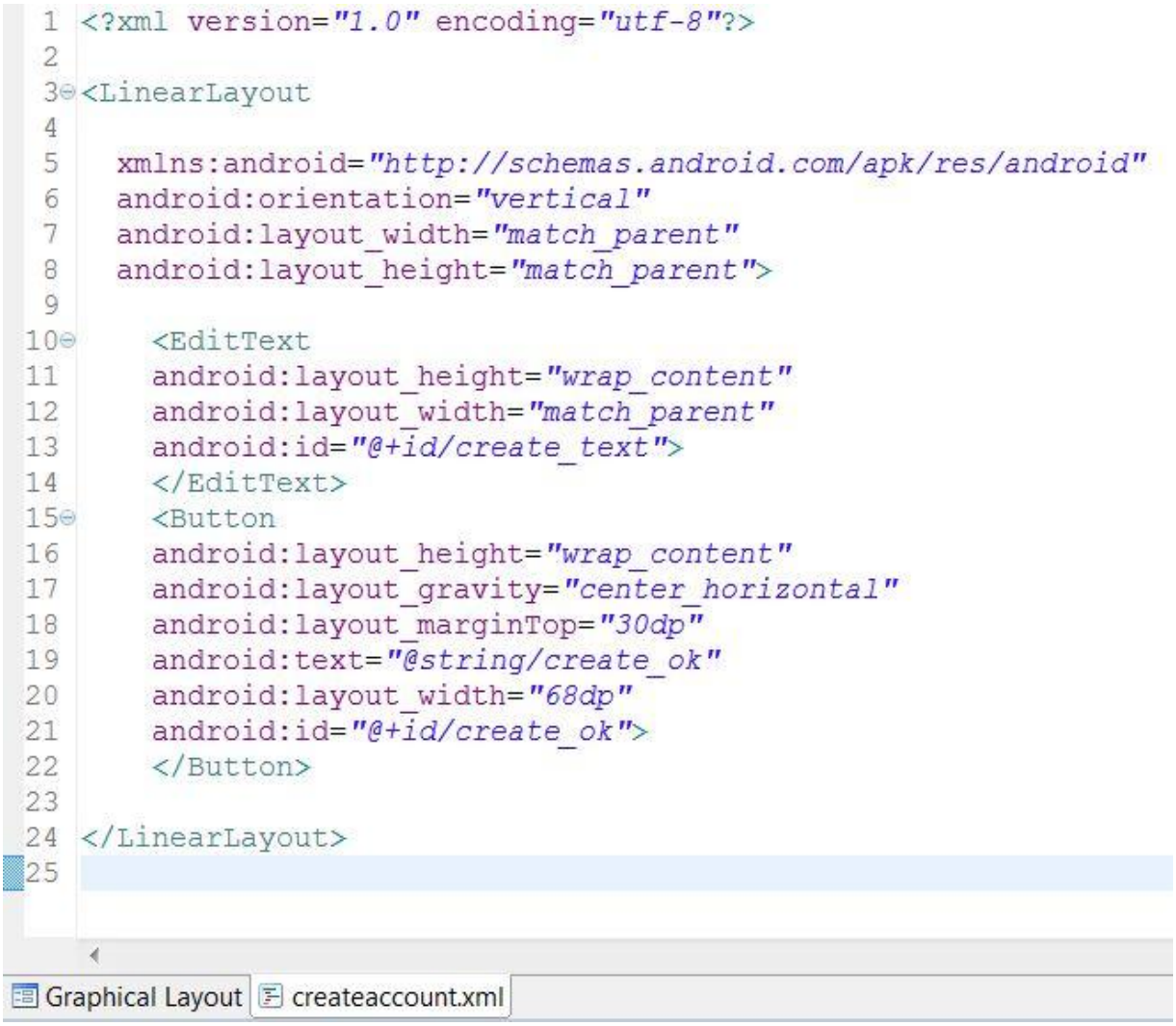
Detta visas i Figur 4 där Pubsub väntar på en ändring från telefonen eller servern. Detta protokoll gör att datatrafik uppstår endast när en ändring är gjord.

3.2 Android

Android[8] är ett operativsystem för mobiltelefoner och surfplattor. För att att utveckla i Android måste utvecklaren installera Androids SDK[9] som är ett paket som möjliggör utveckling för Android plattformar. Androids SDK är ett kit som innehåller klassbibliotek, debugger och emulator. Detta kit läggs till i Eclipse som är utvecklingsmiljön för Android.

Utvecklingen sker i Java och XML. XML står för Extensible Markup Language[10] och används för att dela strukturerad information. XML liknar HTML, men är mer strikt då programmet inte kommer att köras om filen innehåller fel. I Android används XML för att beskriva applikationers användargränssnitt och hur informationen ska se ut vilket visas i Figur 5.

Användaren kan antingen koda direkt i XML-filerna, eller använda Androids grafiska layoutverktyg genom att dra objekt (t.ex. knappar) dit där användaren vill ha dem.



Figur 5. XML-kod för en vy som innehåller en textruta och en knapp.

Android har olika versioner med olika funktionalitet och detta arbete är gjort för Android version 1.6. Detta valdes för att applikationen inte bör kräva någon extra funktionalitet och version 1.6 används av många användare idag. Som hjälp till androidutvecklare finns Androids egen hemsida www.developer.android.com där det finns möjlighet att ladda ner deras SDK, läsa det senaste om Android utveckling, kolla lösningsexempel och specifikationer för klassbiblioteket.

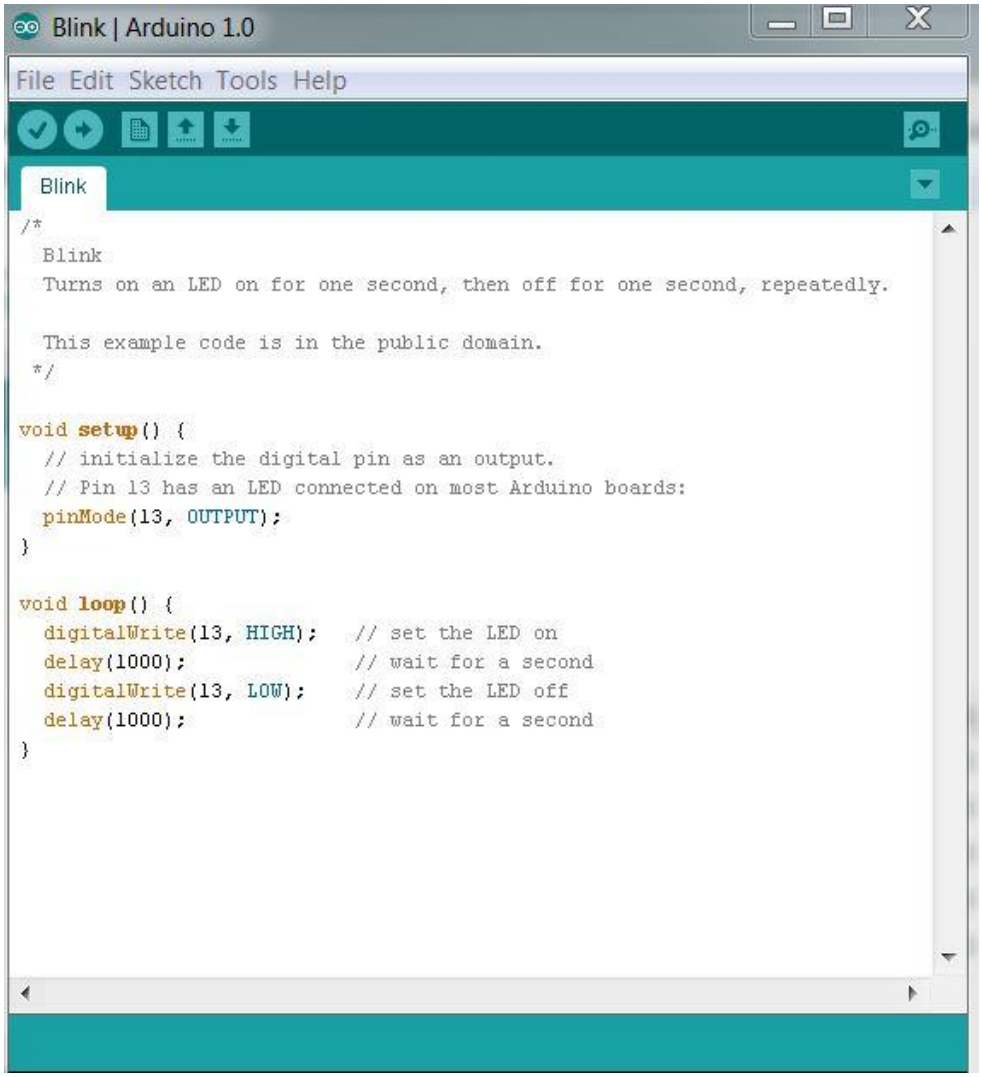
Applikationsutveckling i Android är uppbyggt av vyer som innehåller olika objekt. Varje vy har en egen klass som innehåller representanter för objekt i vyn. Denna klass i Android kallas för en activity. För att ett objekt ska veta när det ska användas använder Android en "lyssnare" för varje objekt. Till exempel används "ButtonListener" för att kolla om användaren har tryckt på en knapp. En applikation innehåller flera vyer och därmed också flera activities. För att hoppa mellan vyer skapar klassen en intent. En intent är den nuvarande vyn som är aktiv på telefonen och genom att skapa en ny intent med en annan vy kan telefonen byta vy.

3.3 Arduino

Arduino är en open source plattform som kan styra fysiska enheter genom Arduinos egen utvecklingsmiljö. Tillämpningar i Arduino skrivs i ett språk liknande C++.

Arduinos utvecklingsmiljö går att köra på Windows, Macintosh OSX och Linux vilket gör den unik eftersom andra processorers utvecklingsmiljö oftast är begränsade till Windows.

Arduino är lätt att lära sig för nya användare då det finns mycket exempel att följa och utvecklingsmiljön[11] är enkel att förstå. Det enklaste exemplet är en blinkande diod som Figur 6 visar.



Figur 6. Kod i Arduinos utvecklingsmiljö som visar ett exempel på en lampa som blinkar en gång i sekunden. Det är i pin 13 på kortet som lampan placeras.

Arduino är baserad på microprocessorerna Atmel´s ATMEGA8 och ATMEGA168.

Det finns 18 olika Arduinokort att välja mellan och detta projekt använde ett UNO-kort och en Ethernet Shield.

Figur 7a. Arduino UNO.

Figur 7b. Ethernet Shield.

Figur 7a är ett Arduino UNO[12] kort och Figur 7b är en Arduino Ethernet Shield[13]. Skölden läggs ovanpå UNO kortet för att kunna användas.

Programmering av kortet sker via USB-kabel mellan kortet och datorns USB-port. Kortet sparar senast inlagda programeringskod vilket betyder att när användaren använder själva kortet behöver inte kortet vara anslutet till datorn för att kunna köra kod.

Arduino har ett mycket brett användningsområde för den fantasifulle då det är väldigt svårt att sätta en begränsning till vad som kan göras med Arduino.

4 Genomförande och Utveckling

Arbetet delades in olika faser som tog olika lång tid. Detta gjordes för att dela in arbetet i mindre delar och sedan koppla ihop delarna med varandra.

4.1 Fas 1 – Bli bekanta med Arduino

Efter att laddat ner Arduinos utvecklingsmiljö var den första uppgiften att få dioder att blinka och att kunna variera lysstyrkan på dioder med en potentiometer. Även en fysisk knapp lades till som stängde av och slog på strömmen till dioderna. Vilket kan liknas med att skicka ström till motorn som vrider låset för öppning eller låsning.

Från början var det menat att Arduinokortet skulle vara av typen Arduino Ethernet[14] . Detta kort var så pass nytt att det saknades drivrutiner, vilket ledde till att handledarna beslutade att övergå till Arduino UNO kompletterad med en Arduino Ethernet Shield. Skölden kopplas ovanpå UNO-kortet genom anpassade PIN:ar. Skölden gör det möjligt att kommunicera med internet genom en ethernet sladd.

4.2 Fas 2 – Hämta webbsida och skicka data till server

Vid hämtning av en webbsida var det HTML-koden för webbsidan som skulle hämtas. Detta gjordes för att kunna läsa information från servern.

På servern finns en textfil som innehåller statusen för ett lås. Denna textfil nås med hjälp av Android-klassen URL[15]. URL klassen används för att skapa koppling till URL-adressen för servern och för att öppna textfilen med statusen. För att kunna ändra i textfilen användes Android-klasserna HttpClient[16] och HttpPost[17]. HttpClient skapar kontakt med webbservern och exekverar det som skall skickas om det finns en kontakt.
HttpPost innehåller det som skall skickas och vart det skall skickas.

I denna fas var informationen som skickas till servern hårdkodad, denna information krypterades senare i applikationen.

4.3 Fas 3 – Skapa Androidapplikation

Användaren väljer ett namn till varje lås som läggs till och dessa namn visas i en lista på telefonen när applikationen körs. Varje gång användaren stänger av applikationen måste telefonen veta hur denna lista skall se ut nästa gång applikationen startas och därför lagras låsen i en lokal databas på mobiltelefonen. Det som sparas till databasen är namnet på låset, låsets serveradress och låsets pinkod.

För databasen användes Android-klasserna SQLiteDatabase[18] och SQLiteOpenHelper[19]. SQLiteDatabase är själva databasen som innehåller all information, medan SQLiteOpenHelper hanterar ändringar och skapar själva databasen.

Då det handlar om att öppna och låsa dörrar till ett hem krävs det att applikationen är säker. Därför har applikationen en fyrsiffrig kod som hindrar obehöriga från att komma åt låsen.
Denna kod får användaren välja vid uppstart av applikationen första gången, och den lagras lokalt på mobiltelefonen. För att spara undan den fyrsiffriga koden användes klasserna FileInputStream[20] och FileOutputStream[21]. FileInputStream läser koden från den lokala filen på telefonen. FileOutputStream skriver till den lokala filen med önskad kod.
Den lokala filen valdes för att koden skall vara skyddad. Denna metod gör att koden aldrig skickas från telefonen och att den inte går att komma åt om obehöriga får tag på databasen.

Själva listan med lås var viktig att kunna uppdatera dynamiskt för att slippa starta om applikationen varje gång ett lås lagts till, tagits bort eller ändrats. Listan skall även vara möjlig att scrolla i om det skulle behövas och det skall vara möjligt att koppla en ikon till varje objekt i listan. Ikon önskas för att visa om ett lås har statusen öppen eller låst. Android-klassen ListView[22] hade det som efterfrågades av listan med lås och blev därför lösningen.

Det riktiga låset är kopplat via Arduino till servern med hjälp av Pachube[23], applikationen är kopplad till servern med hjälp av Pachube.

Applikationen skickar ett meddelande till servern om att statusen skall ändras. Servern skickar då ett meddelande till Arduinokortet om att statusen skall ändras. Arduinokortet startar en motor som vrider om låset.

Kod från Fas 2 återanvändes med mindre justeringar för att skicka data till servern. Att hämta status från servern uteslöts från denna applikation då det medförde krockar i servern. Användaren kan endast säga vad denne vill göra, inte kolla statusen för ett lås.

Tidigare har företaget gjort en iPhoneversion som använder Pachube som server. Denna version hade fördröjningar som var oönskade och servern belastades onödigt mycket, därför ville Dunderbar ersätta Pachube med Pubsub[24].

I detta projekt är säkerheten viktig då inga obehöriga ska kunna öppna lås. Detta löses med hjälp av android-klasserna Mac[25] och SecretKeySpec[26] . Mac klassen används till att kryptera nycklarna som skall skickas med en algoritm som uppdragsgivaren valt.
Varje lås har ett lösenord som används för att skapa en privat nyckel som genereras med SecretKeySpec. Denna nyckel krypteras sedan tillsammans med en publik nyckel från servern och önskad status för att sedan skickas tillbaka till servern.

4.4 Fas 4 – Testning mot ett riktigt lås

När applikationen bedömdes klar var nästa steg att testa applikationen mot ett riktigt lås.

Det största problemet som uppstod var när applikationen kollade status på låset samtidigt som den skickade en ändring. Det var här det upptäcktes krockar i servern då applikationen försökte göra pull och push till servern samtidigt. Det uteslöts för tillfälligt att applikationen skulle göra pull för status från servern.

Detta innebär att det endast är möjligt att säga vad användaren vill göra, inte ta reda på vilken status ett lås har.

Dessutom var det möjligt att få applikationen att krascha. Detta var accepterat just då på grund av att det viktiga var att om användaren gjorde rätt skulle det fungera.

Krascherna uppstod när användaren skrev fel indata, till exempel bokstäver istället för siffror. Även om telefonen saknade nätverk kraschade applikationen.

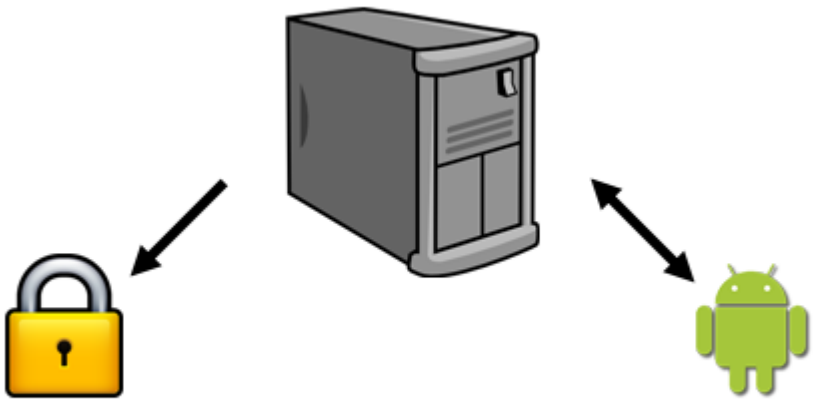
4.5 Fas 5 – Stabilisera applikation och leverera en färdig produkt

Det är väldigt viktigt att applikationen aldrig kraschar, därför blev det hög prioritet att hantera varje situation där en krasch kunde uppstå.
Det löstes med Android-klassen Dialog[27] som visade ett felmeddelande när något gick fel.

Vid tiden för denna rapportskrivning, 2012-05-02, finns applikationen inte på Android Market men den är redo för publicering.

5 Resultat

5.1 Översikt



Figur 8. Översikt över systemet, där pilarna är kommunikationen i Pachube

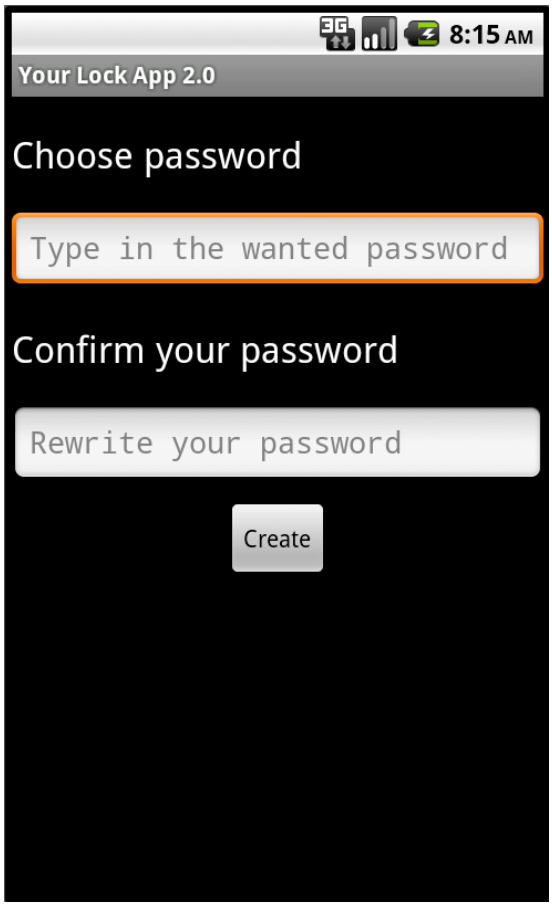
Systemet som har utvecklats fungerar som Figur 8 visar.
Androidapplikationen säger till servern om den vill låsa eller att låsa upp.

Att pilen är dubbelriktad mellan applikation och server beror på att när användaren vill ändra status behövs en publik nyckel. Denna nyckel behövs för att kryptera informationen som skickas tillbaka servern. Krypteringen är en säkerhetsåtgärd för att servern ska veta att den får information från rätt avsändare. Servern förväntar sig en kryptering med rätt algoritm och nycklar tillbaka. När servern fått ett giltigt kommando skickas detta kommando till Arduinon som vrider om låset.

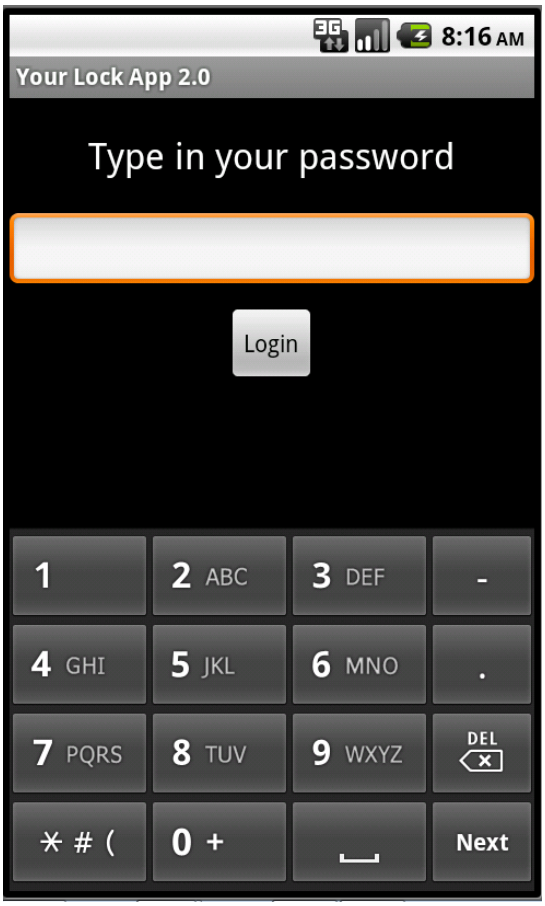
Anledningen till att pilen endast är enkelriktad är att servern inte kan läsa av vilken status låset har. Servern kan endast skicka till låset vilken status som önskas. Det finns implementerat att Arduinokortet kan skicka information till servern, det vill säga att denna kommunikation också kan vara dubbelriktad. Detta system kan inte känna av om låset vridits om manuellt och därför skickar Arduinokortet aldrig något till servern.

Säkerheten mellan Arduinokortet och servern har samma säkerhetsnivå som säkerheten mellan telefonen och servern. De använder samma hemliga nyckel vid kryptering och signering. Säkerheten bedöms därför vara lika säker i hela systemet.

5.2 Androidapplikation



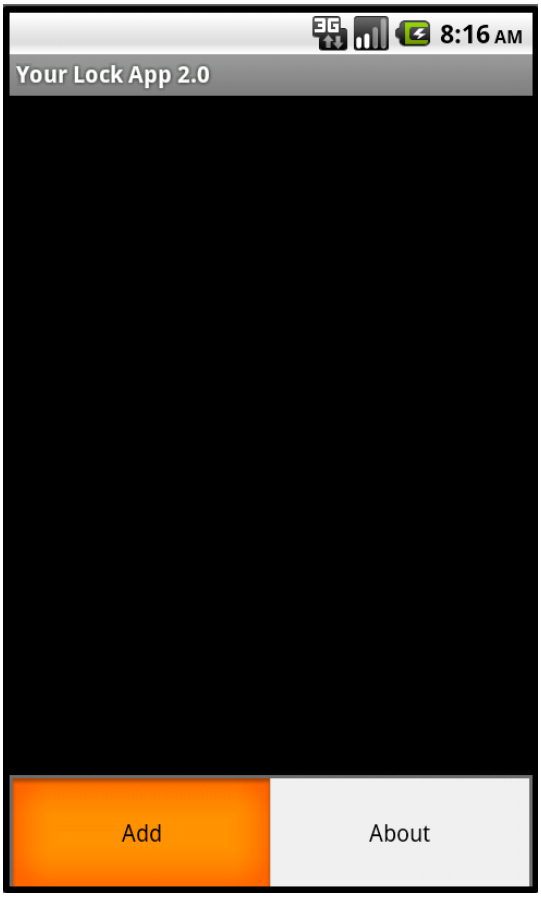
Figur 9a. Vy vid skapning av lösenord.



Figur 9b. Vy vid inloggning

När applikationen startas första gången möts användaren av en vy som Figur 9a visar. Användaren väljer en fyrsiffrig kod som behövs för att kunna logga in. Denna funktion behövs för att hindra obehöriga att komma in i applikationen vilket var ett säkerhetskrav.

Varje gång applikationen startas måste användaren logga in med sin fyrsiffriga kod som Figur 9b visar. Då koden för inloggning är fyra siffror kommer automatiskt det numeriska tangentbordet upp för att undvika missförstånd.

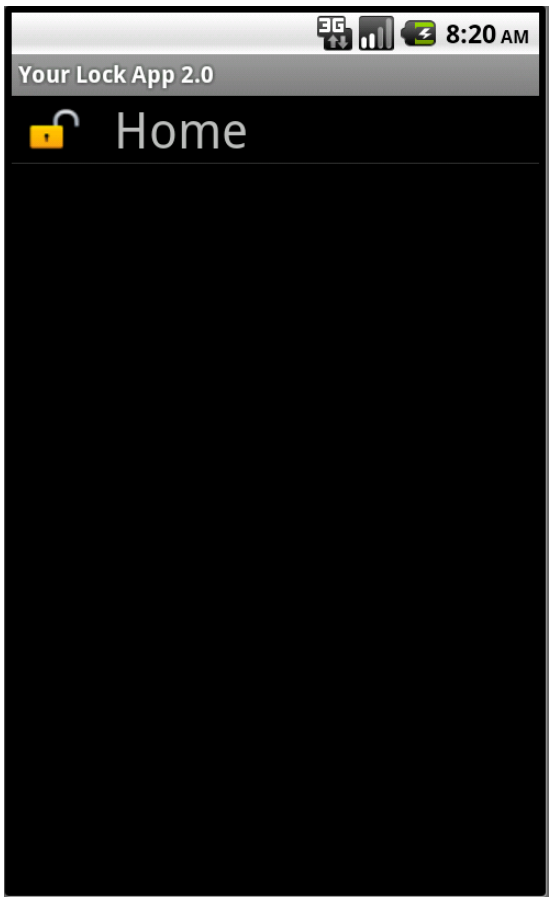


Figur 10a. Tom lista med lås.

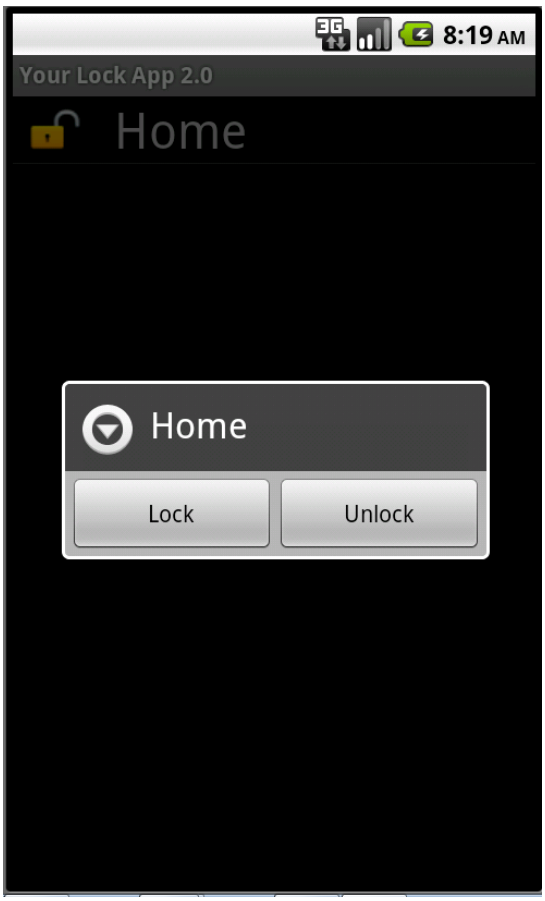
Figur 10b. Tilläggning av ett lås.

Figur 10a visar själva listan på låsen. Klickar användaren på telefonens menyknapp kommer knapparna “Add” och “About” upp. När användaren vill lägga till ett lås, görs detta genom att klicka på Add. About visar information om hur applikationen fungerar.

Figur 10b visar menyn för att lägga till ett lås. Överst skriver användaren vad låset skall heta, i mitten skrivs URL-adressen till servern för låset in och längst ner skrivs lösenordet för låset in.



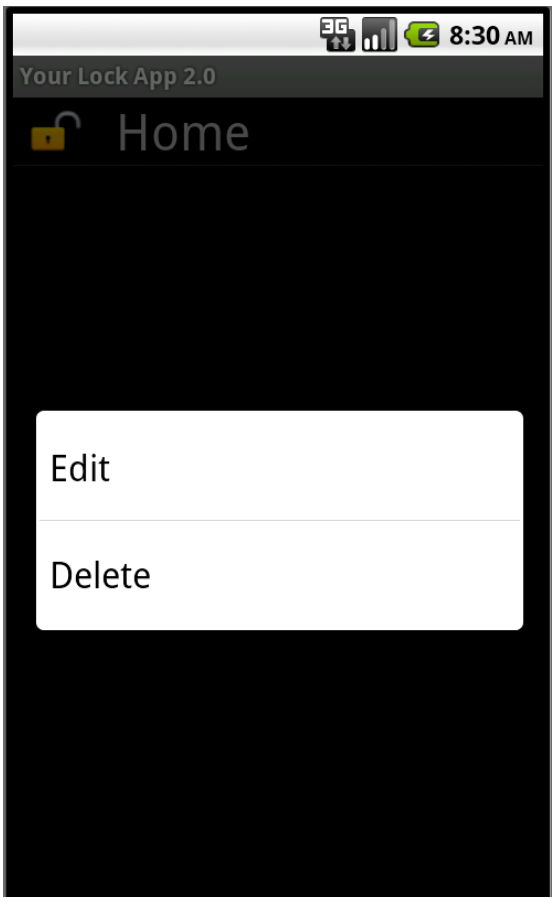
Figur 11a. En lista med ett lås.



Figur 11b. Låsa eller låsa upp ett lås.

Figur 11a visar listan med användarens lås. Ikonen bredvid ändrar sig till en låst ikon när användaren låser låset och en upplåst ikon när användaren låser upp låset. Denna lista uppdateras dynamiskt och senast tillagt lås placeras nederst i listan.

För att ändra status på ett lås klickar användaren på raden för det lås som skall ändras och menyn ovan kommer fram. Vid låsning väljer användaren "Lock" och vid öppning väljer användaren "Unlock" som syns i Figur 11b. Ett viktigt krav var att applikationen skulle vara användarvänlig och att menyerna skulle vara väldigt tydliga.



Figur 12a. Vy vid ändring eller borttagning av ett lås

Figur 12b. Ändring av ett lås.

Vid ändring eller borttagning av lås måste användaren trycka länge på den rad som önskas. Då kommer menyn fram som visas i Figur 12a. Vid borttagning trycker användaren “Delete” och vid ändring trycker användaren ”Edit”.

Trycker användaren på ”Edit” visas vyn som Figur 12b visar. Användaren kan endast ändra namn och URL-adress för ett lås.

6 Slutsats

Vid projektets start var det viktigaste att applikationen var användarvänlig och att applikationen inte upplevdes långsam när användaren låser eller låser upp. Det rådde delade meningar om vad som var användarvänligt men som applikationen ser ut nu gör företaget mest nöjd. Handledaren tycker att applikationen är användarvänlig och menyn är tydlig. Det finns även en "About"-knapp som kommer fram när användaren klickar på menyknappen som beskriver hur applikationen fungerar.

Motsvarande applikation för iPhone upplevdes som långsam. Målet för Androidversionen var att göra den snabbare. Handledaren tror att detta problem skall lösas genom att ersätta Pachube med Pubsub. Det uppstod problem i Pubsub som gjorde att det inte fanns tid att vänta på att dessa problem skulle bli åtgärdade, därför valde handledaren att Pachube skall användas tills vidare.

Det finns ett par företag som håller på med liknande men där används inte open source. Andra företags lösningar är också mer avancerade och mer funktionsrika. Det är länken mellan telefon, internet protokoll och Arduino som gör detta system unikt. Lösningen i detta examensarbete är riktad till privatpersoner för att användas med privata lås.

Handledaren har nära kontakt med grundaren av Pubsub och båda två är övertygade om att Pubsub kommer vara den bästa lösningen för detta projekt.

Företagets version till iPhone använder Arduino UNO-kort kompletterat med en Ethernet Shield för att kunna nå internet. Till Androidversionen ville företaget ha Arduino Ethernet istället, skillnaden är helt enkelt att med Arduino Ethernet behövs inte någon sköld då sköldens funktioner finns inbyggda i Arduino Ethernet. Problemet var att Ethernet kortet var så pass nytt att det inte fanns kompletta drivrutiner till den och därför fick även Android versionen ha Arduino UNO och Ethernet Shield i lösningen. Implementerad lösning kostar ungefär 500 kronor och när Arduino Ethernet ersätter Arduino UNO och Ethernet Shield beräknas priset sjunka ungefär 100 kronor.

7 Framtida Utvecklingsmiljöer

En önskan är att i framtiden kunna komplettera projektet med en sensor som känner av om dörren är stängd och om låset har vridits om manuellt och meddela servern detta. Det hade gjort servern och telefonen mer pålitlig och steget att strunta i nycklar helt hade varit betydligt närmare.

Pachube skall så fort som möjligt ersättas med Pubsub, förmodligen sker detta i juni 2012.

För att göra applikationen mer användarvänlig skulle inmatningen för URL-adressen kunna förenklas genom att ha ett unikt id för varje lås. Detta id skrivs in istället för hela URL-adressen när användaren lägger till låset.

8 Terminologi

buggar - Mindre fel som inte stänger ner applikationen.
hackaton - Tillfälle där alla i projektet träffas, för att integrera alla delar.
intent – Nuvarande vy på telefonen.
krasch - Allvarligt fel som stänger ner applikationen.
krockar - Flera parter försöker komma åt samma sak.
källkod - Koden för en viss hemsida.
NFC - Kommunikationsstandard för smartphones med kort räckvidd.
open source - Öppen programvara, där källkod är tillgänglig att använda.
publik nyckel – Nyckel som inte är hemlig.
pull - Hämta data.
push - Skicka data.
realtid - En händelse som ska hända direkt.
scrolla - Bläddra igenom.
SDK - Software developer kit
sköld - Tillbehör till ett kretskort.
vy - Det användaren ser på telefonen

9 Referenser

Demonstration av iPhoneversionen:
[1] http://www.youtube.com/watch?v=y-MaDnMlr6g - 21 maj 2012

Konkurrenter:
[2] http://www.zaplox.com/ - 21 maj 2012
[3]http://www.assaabloy.com/en/com/About-ASSA-ABLOY/Strategy/Product-leadership/Cell-phone-replaces-key/ - 22 maj 2012

NFC - Teknik som konkurrent använder:
[4] http://www.nfc-forum.org/aboutnfc/ - 22 maj 2012

Om Arduino:
[5] http://www.arduino.cc/ - 31 jan 2012

Presentation av Pachubes grundare:
[6] http://www.youtube.com/watch?v=Xd5nffPgqkA - 21 maj 2012

Om Pubsub:
[7] http://pubsub.io/about - 21 maj 2012

Androids hemsida:
[8] http://www.android.com/ - 21 maj 2012

Androids utvecklingsmiljö:
[9] http://developer.android.com/sdk/index.html - 21 maj 2012

Om XML:
[10] http://www.w3.org/standards/xml/core - 21 maj 2012

Arduinos utvecklingsmiljö:
[11] http://arduino.cc/en/Main/Software - 31 jan 2012

Arduino hårdvara:
[12] http://arduino.cc/en/Main/ArduinoBoardUno - 6 feb 2012
[13] http://arduino.cc/en/Main/ArduinoEthernetShield - 6 feb 2012
[14] http://arduino.cc/en/Main/ArduinoBoardEthernet - 31 jan 2012

Om Android-klassen URL:
[15] http://developer.android.com/reference/java/net/URL.html - 13 feb 2012

Om Android-klassen HttpClient:
[16]http://developer.android.com/reference/org/apache/http/client/HttpClient.html - 14 feb 2012

Om Android-klassen HttpPost:
[17]http://developer.android.com/reference/org/apache/http/client/methods/HttpPost.html - 14 feb 2012

Om Android-klassen SQLiteDatabase:
[18]http://developer.android.com/reference/android/database/sqlite/SQLiteDatabase.html- 5 mars 2012

Om Android-klassen SQLiteOpenHelper:
[19]http://developer.android.com/reference/android/database/sqlite/SQLiteOpenHelper.h - 5 mars 2012

Om Android-klassen FileInputStream:
[20] http://developer.android.com/reference/java/io/FileInputStream.html - 26 mars 2012

Om Android-klassen FileOutputStream:
[21] http://developer.android.com/reference/java/io/FileOutputStream.html - 26 mars 2012

Om Android-klassen ListView:
[22] http://developer.android.com/reference/android/widget/ListView.html - 27 feb 2012

Pachubes hemsida:
[23] https://pachube.com/ - 3 april 2012

Pubsubs hemsida:
[24] http://pubsub.io/ - 4 april 2012

Om Android-klassen Mac:
[25] http://developer.android.com/reference/javax/crypto/Mac.html - 3 april 2012

Om Android-klassen SecretKeySpec:
[26]http://developer.android.com/reference/javax/crypto/spec/SecretKeySpec.html - 3 april 2012

Om Android-klassen Dialog:
[27] http://developer.android.com/reference/android/app/Dialog.html - 6 mars 2012