Examensarbete

HÖGSKOLAN I SKÖVDE

MIGRATION AV APPAR TILL WINDOWS PHONE 8

MIGRATION OF APPS FOR WINDOWS PHONE 8

Examensarbete inom huvudområdet informationssystemutveckling
Grundnivå 15 högskolepoäng
Vårtermin 2013

Peter Johansson

Handledare: Mikael Berndtsson
Examinator: Eva Söderström

Sammanfattning

Fler människor börjar använda olika mobila operativsystem i mobiltelefonen samt appar som enbart fungerar på plattformen. Det leder till att app-leverantörer måste utveckla varje app till varje specifik plattform vilket är dyrt. De vanligaste operativsystemen som används idag är Android samt IOS men hösten 2012 släpptes Windows Phone 8 vilket innebär ytterligare en plattform att ta hänsyn till. I sammarbete med företaget Sogeti Jönköping fokuserar studien på att hitta en metod som gör det möjligt att migrera appar mellan plattformar och till Windows Phone 8. Studien innefattar också att identifiera för och nackdelar med olika app-typer genom en litteraturstudie och intervjuer samt att testa om migrationsmetoderna fungerar praktisk genom en implementation.

Resultatet har påvisat att det finns två stycken olika metoder för migration vilket är PhoneGap och Xamarin. Genom implementationstester har Xamarin visat sig vara mest lämplig att använda i ett framtida större app-projekt.

Nyckelord: [Windows Phone 8, Xamarin, PhoneGap, Systemmigration, Appar]

Abstract

More people are starting to use different mobile operating systems in their mobile phones and apps that only work on the specific platform. This leads to app-developers must develop each app on each specific platform which is expensive. The most common operating systems today are Android and iOS, but during fall of 2012 Windows Phone 8 was released, which means yet another platform to consider. In cooperation with the company Sogeti Jönköping this study focuses on finding a method that makes it possible to migrate applications between platforms and Windows Phone 8. The study also includes identifying the pros and cons of different app-types through a literature review and interviews, as well as testing whether the migration methods work through a practical implementation.

The results have shown that there are two different solutions to migration which is PhoneGap and Xamarin. Through implementation tests Xamarin was the most suitable for future app-projects.

Keywords: [Windows Phone 8, Xamarin, PhoneGap, System migration, Apps]

Förord

Jag vill passa på att tacka de personer som hjälpt mig att utföra mitt examensarbete. Ett stort tack till min handledare Mikael Berndtsson på högskolan som har intresserat sig och guidat mig genom mitt examensarbete. Ett stort tack till handledaren Marcus Schelin och övrig personal på Sogeti Jönköping som gjort det möjligt för mig att utföra mitt examensarbete.

Innehållsförteckning

1 Introduktion

I kapitlet presenteras studien allmänt och varför ämnet är intressant att studera. Kapitlet börjar med att ge en introduktion om mobilt användande och fortsätter sedan med att ge en bild om problemområdet, vilket är migration mellan plattformar. Kapitlet avslutas med en kort introduktion om hur studien ska utföras och vad resultatet bidrar till.

1.1 Inledning

Idag använder många människor mobiltelefonen som ett redskap både privat och i arbetet. I samband med att fler använder mobiltelefonen utvecklas även tekniken inom det mobila området. På senare år har smarta telefoner och surfplattor blivit oerhört populära bland användarna. Smarta telefoner jämfört mot gamla mobiltelefoner har stöd för avancerad teknik som strömmande videouppspelning och avancerade spel. SCB (2013) skriver att 97 % av svenskarna någon gång under det första kvartalet 2012 använde en mobiltelefon eller en smarttelefon. De smarta telefonerna möjliggör också uppkoppling till internet för att kunna ta del av den teknik och program som finns representerade på telefonerna. SCB (2013) skriver också att 59 % av svenskarna kopplar upp sig på internet via smarta telefoner utanför hemmet. Smarta telefoner har många funktioner som kan hjälpa och underhålla användaren, både i arbete och privat. En stor del av smarta telefoners användande går ut på att använda telefonens applikationer eller appar som de också kallas. SCB (2013) skriver att en app som används flitigt bland användarna är olika GPS-appar där användaren kan se vart denne befinner sig eller vad denne är i närheten av, exempelvis en viss restaurang eller en butik. Nästan 40 % av svenskarna har under det första kvartalet 2012 använt en sådan app. I samband med att fler människor använder mobiltelefonen som en naturlig del i vardagen ställer det också krav på de som tillverkar mobiltelefoner och appar för telefonerna. De måste hänga med den snabba utveckling som sker för att användarna på ett enkelt sätt kan använda de tjänster som en smart telefon kan tillhandahålla, samtidigt som de hela tiden måste uppdatera utbudet samt utveckla nya appar.

Gartner (2012) skriver att nästan 170 miljoner smarta telefoner blev sålda i världen under deras senaste analys av tredje kvartalet 2012, se figur 1. Det populäraste mobila operativsystemet var Android med ca 72 % andelar följt av IOS med ca 14 %, lite längre ned på listan kom Windows Phone med 2,4 % av andelarna. Windows Phone har jämfört mot år 2011 klättrat från 1,5 % till nuvarande 2,4 %. En ytterligare uppgång kan komma att ske i samband med att nya telefoner släpps som har Windows Phone 8 som operativsystem skriver Gartner (2012).

Worldwide Mobile Device Sales to End Users by Operating System in 2Q12 (Thousands of Units)

Operating System	2Q122Q12 Market Share (%)		2Q112Q11 Market Share (%)	
	Units		Units	
Android	98,529.3	64.1	46,775.9	43.4
iOS	28,935.0	18.8	19,628.8	18.2
Symbian	9,071.5	5.9	23,853.2	22.1
Research In Motion	7,991.2	5.2	12,652.3	11.7
Bada	4,208.8	2.7	2,055.8	1.9
Microsoft	4,087.0	2.7	1,723.8	1.6
Others	863.3	0.6	1,050.6	1.0
Total	**153,686.1**	**100.0**	**107,740.4**	**100.0**

Worldwide Mobile Device Sales to End Users by Operating System in 3Q12 (Thousands of Units)

Operating System	3Q123Q12 Market Share (%)		3Q113Q11 Market Share (%)	
	Units		Units	
Android	122,480.0	72.4	60,490.4	52.5
iOS	23,550.3	13.9	17,295.3	15.0
Research In Motion	8,946.8	5.3	12,701.1	11.0
Bada	5,054.7	3.0	2,478.5	2.2
Symbian	4,404.9	2.6	19,500.1	16.9
Microsoft	4,058.2	2.4	1,701.9	1.5
Others	683.7	0.4	1,018.1	0.9
Total	**169,178.6**	**100.0**	**115,185.4**	**100.0**

Figur 1 - Försäljning av mobila operativsystem under kvartalen 2 & 3 år 2012 (Gartner)

Windows Phone 8 bygger på operativsystemet Windows 8 som Microsoft lanserade i slutet av oktober 2012. I samband med att Windows 8 lanserades och att fler och fler personer får upp ögonen för operativsystemet kan användningen öka både privat och för företag. För företag som använder Microsoft relaterade produkter och system kan det inom en snar framtid eller om det inte redan skett bli aktuellt för en uppgradering till Windows 8. När fler och fler börjar använda operativsystemet finns det en chans att personer också börjar intressera sig och använda Windows Phone 8 som också kom i samband med Windows 8 lanseringen. När ett nytt operativsystem släpps och användare börjar använda systemet är det viktigt för leverantörerna att hänga med i utvecklingen och kunna erbjuda applikationer och tjänster som app-utveckling till kunderna. Kunder som hade sitt favoritspel på Android eller kartappen i IOS vill kunna använda samma eller en liknande app i Windows Phone 8. Gartner (2012) skriver att när fler telefoner kommer att släppas med Windows Phone 8 kommer troligtvis efterfrågan av appar att öka bland kunder och blir där med en viktig del för leverantörerna att kunna möta efterfrågan med utbud.

När användningen av en ny plattform inom företag, organisationer eller privat växer kommer komplexiten och kostnader för leverantörerna att öka. Några av dessa är exempelvis:

- Appen måste anpassas och programmeras för varje enskild plattform
- Licens och utvecklingskostnader för varje enskild plattform
- Tids och resurskrävande
- Uppdateringar måste ske på flera olika plattformar vilket leder till högre kostnader

Dey & Sarma (2007) skriver att migration av data mellan källor är nödvändig för att kunna bevara information när ny teknik utvecklas och börjar användas, vilket blir aktuellt när nya telefoner från mobiltillverkarna kommer som använder operativsystemet Windows Phone 8. När ett system eller en app ska migreras till en ny plattform är det sällan en enkel procedur. Ståhl (2012) skriver att det är viktigt att göra en uppskattning på hur lång tid det kommer att ta för att utföra själva migrationen och få den fungera på en ny plattform. Anledningen till det är för att kunna se om ett projekt kan vara ekonomiskt hållbart eller om det är bättre att enbart använda det tidigare systemet. Ståhl (2012) menar också att det inte går att säga hur en migration eller en portning bör gå till eftersom varje enskilt fall är annorlunda. När en migration mellan plattformar sker kan antingen systemet fungera på båda plattformarna eller endast på den nya plattformen.

Tidigare har appar i Windows Phone enbart fungerat på den egna plattformen. Frågan är ifall det kommit en metod som gör det möjligt att utföra en migration som bidrar till att en liknande applikation på en annan plattform kan fungera på Windows Phone 8. För att kunna avgöra huruvida det fungerar att migrera till Windows Phone 8 måste migrationsmetoder identifieras som kan klara av att köras i en .NET struktur. För att identifiera dessa metoder behövs insamlad data om både migration, appar och om olika mobila operativsystem. Metoderna måste också testas praktiskt om de fungerar eftersom det är svårt att avgöra ett enskilt fall enligt Ståhl (2012). För studiens resultat kommer jag att analysera varje utvald migrationsmetod och jämföra dem mot varandra samt hur de skulle kunna bidra till besparingar för en leverantör.

2 Bakgrund

I kapitlet presenteras den litteratur som används i studien för att kunna hjälpa till att besvara frågeställningen. Kapitlet inleds med en introduktion till företaget som studien är gjord i samband med samt en teoretisk grund om mobila plattformar och systemmigration.

2.1 Om företaget

Studien är gjord i samarbete med företaget Sogeti i Jönköping. Sogeti är ett konsultbolag med säte i 21 svenska städer och har cirka 1150 anställda konsulter. De har i uppgift att leverera högkvalitativa IT-konsulttjänster till den lokala marknaden. Några av de tjänster som Sogeti levererar är IT-styrning som inkluderar ledning och styrfrågor, IT-lösningar som inkluderar utveckling och integration av applikationer samt IT-förvaltning som inkluderar system och driftförvaltning. Sogeti Jönköping har cirka 60 anställda och arbetar främst med företag och organisationer i Jönköping med omnejd. De områden Sogeti jobbar med i Jönköping är bl.a. mobilitet och appar, e-handel, projektledning och verksamhetsutveckling.

Samarbetet med Sogeti Jönköping har i studien innefattat att tillsammans hitta olika lösningar för att på ett kostnadseffektivt sätt kunna migrera befintliga appar till Windows Phone 8. I samarbetet har jag fått tagit del av personalens kunskaper angående app-utveckling samt utveckling av appar till Windows Phone 8. Det uppvisade resultatet med studien är tänkt att kunna stå för en grund för ett större migrationsprojekt som Sogeti Jönköping kan arbeta vidare med.

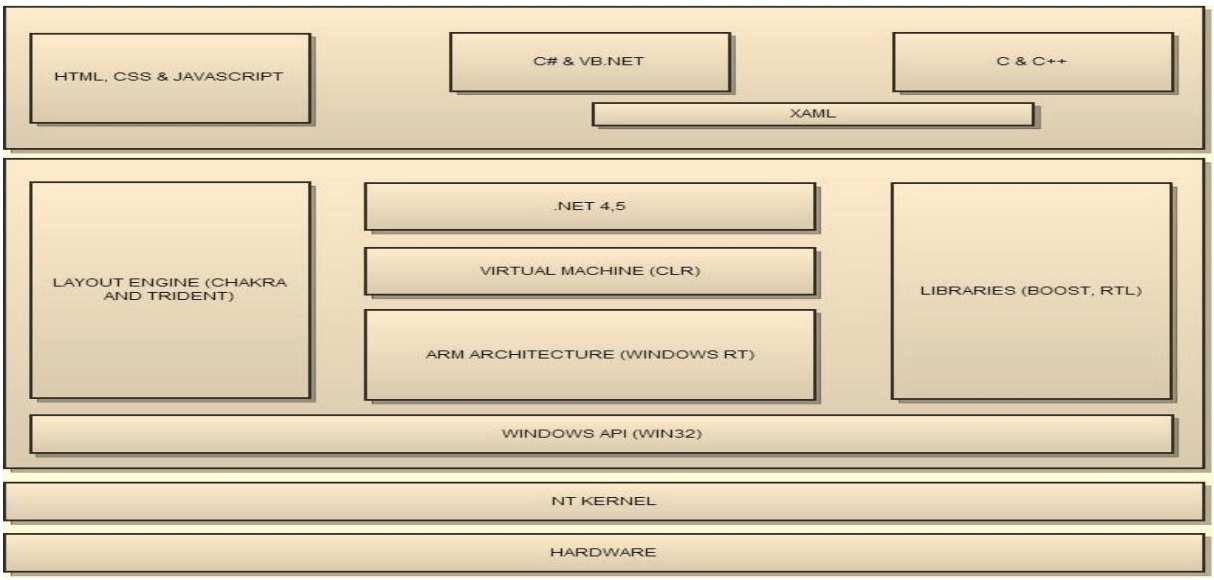
2.2 Teoretisk grund

Kapitelet inleds med mobil utveckling som ska introducera ämnet och varför det ökar. Sedan tas generell migration och mobil migration mellan plattformar upp, vilket studien fokuserar på i helhet. Mobila operativsystem ger en introduktion om vilka operativsystem finns och är populära att använda idag, dessa operativsystem kan köra olika typer av appar. App-typerna är viktiga att undersöka eftersom det finns olika krav hos olika kunder. Avslutningsvis tas olika ramverk och API:er upp som underlättar implementation vid utveckling generellt och mobil utveckling och migration.

2.2.1 Utveckling på mobila plattformar

När tekniken i samhället går framåt vill användare kunna ta del av det. En av dessa tekniska delar är mobiliteten och användningen av smarta telefoner med tillhörande appar. Rahimian & Habibi (2008) skriver att den mobila utvecklingen har ökat pga. fler har fått tillgång till en snabb anslutning till internet via mobilen, vilket gör att användarna har möjlighet att använda applikationer och internet nästan överallt. Ökningen leder till att fler systemutvecklare ser potential för ekonomiska vinster med att börja med utveckling för mobiler. Till skillnad från traditionell utveckling finns det flera olika aspekter att beakta. Rahimian & Habibi (2008) skriver att utvecklaren måste ta hänsyn till att det finns telefoner med olika hårdvaruegenskaper som bl.a. storlek på telefonen, CPU och minne på telefonen. Det finns också olika telefoner som använder hårdvaran på olika sätt exempelvis kamera och videoinspelning. Ytterligare en del som kan ställa till det är portningen eller migrationen mellan olika system skriver Rahimian & Habibi (2008). I samband med den ökade användningen av mobila lösningar bland företag och privatpersoner växer även användningen av olika telefoner och plattformar. Det är vanligt att samma app finns på olika plattformar och innan en app kan migreras till en annan plattform måste plattformens egenskaper beaktas i form av exempelvis arkitektur.

Tillborg, Persson & Bentlöv (2012) skriver att IOS och Android är de stora mobila operativsystemen som används i Sverige idag och att flest användare köper telefoner som är relaterade till dessa. Andra mobila operativsystem som finns är Windows Phone 8 som kom i samband med Microsoft nya operativsystem Windows 8 och är uppbyggt på samma struktur som Windows Phone 8, se Figur 2. De operativsystem som verkar på olika plattformar har också olika egenskaper i form av exempelvis exekvering av kod. När en app ska migreras mellan en plattform till en annan skriver Dey & Sarma (2007) att det måste vara lönsamt att göra en migration som inte leder till ökade kostnader för företag som utvecklar appar. Det är därför viktigt att hitta ett kostnadseffektivt sätt som också överensstämmer med plattformarnas utformningar.



Figur 2. Uppbyggnad av Windows Phone 8 arkitektur nedifrån och upp

2.2.2 Systemmigration

Ståhl (2012) skriver att en migration eller en portering är en förflyttning av källkod från ett målsystem till ett eller flera system där det nya systemet är tänkt att efterlikna det gamla. När ett befintligt system eller data ska flyttas från en källa till en annan källa måste migrationstekniken beaktas för att kunna flytta över data från källa A till källa B. Dey & Sarma (2007) skriver att det finns viktiga faktorer för en lyckad datamigration mellan olika källor. Validering och överensstämmelse av data är en viktig faktor för att användarna ska kunna härleda den gamla källan till den nya. Om den nya källan inte kan verifiera innehållet i form av exempelvis text och bild kan problem uppstå för användaren eftersom den inte känner igen sig när den använder det nya systemet.

Andersson (2010) skriver att migration mellan system ofta innebär problem i arkitekturen som gör att transformeringen inte kan göras. När plattformar med olika egenskaper och som har skilda uppbyggnader gör det att flertalet olika aspekter måste beaktas. Några av dem är hur operativsystemet kör och behandlar applikationerna på plattformen, vilka programmeringsspråk som fungerar på plattformen eller hur information kan visas grafiskt på plattformen. Dey & Sarma (2007) skriver också att de verktyg och metoder som kan användas för att migrera mellan plattformar måste stämma överens på projektet och arkitekturen. Om det inte går att applicera en metod som fungerar kan det bli kostsamt för företaget i form av tid och ekonomiska resurser. Det kan leda till att systemet måste göras om på nytt eller göra stora uppdateringar i den befintliga implementationen.

Pehrson (2011) skriver om olika former av migration och tester mellan mobila plattformar. IOS som används på iPhone telefoner är ett mobilt operativsystem som använder sig av programmeringsspråket Objective-C som är plattformsberoende, vilket gör att andra mobila plattformar inte kan använda sig av språket vid körning exempelvis operativsystemet Android, vilket medför till att migrationen av appen inte kan gå helt smärtfritt. Pehrson (2011) skriver att han använt sig av en multiplattform vilket möjliggör körning på flera olika operativsystem, exempelvis på IOS och Android. Denna multiplattform har flertalet olika verktyg för att kunna göra migrationen möjlig. Pehrson (2011) skriver bl.a. om PhoneGap, ett ramverk som bygger på webbaserade tekniker som HTML(HyperText Markup Language) och CSS(Cascading Style Sheet), vilket möjliggör körning på andra plattformar eftersom formaten är standardiserade på alla enheter som kan ha en webbläsare.

När ett migrationsarbete mellan plattformar påbörjas går det använda sig av vissa tekniker. Två av dessa tekniker kallas top-down och bottom-up. Ståhl (2012) skriver att top-down utgår från den modul som startar programmet för att sedan kunna starta upp nästa modul vilket kan vara det visuella i ett system. När en visuell del fungerar fortsätter man exempelvis med nästa visuella modul för att sedan gå vidare ner i arkitekturen och påbörja arbetet med mer hårdvarunära och systemnära moduler. Att börja med det visuella kan vara en fördel rent psykologiskt då utvecklaren faktiskt ser att någonting händer med systemet och för projektet framåt. Den andra tekniken kallas bottom-up vilket innebär att utvecklaren börjar med att arbeta nära applikationens kärna, exempelvis nära systemmoduler och hårdvara. När en viss modul fungerar börjar utvecklaren med nästa närliggande modul och arbetar sedan uppåt mot applikationens mer visuella delar.

2.2.3 Mobila operativsystem

Ett mobilt operativsystem är det övergripande system som möjliggör användning av telefonen och dess funktioner. Android och IOS är enligt Gartner (2012) de två stora och Windows Phone som är på uppgång. Android och IOS är främst de som leverantörer arbetar med idag men i samband med Windows Phone 8 lanserats kommer arbetet öka för leverantörerna.

Android

Android är en mjukvara och ett operativsystem vars mobila plattform heter Java ME som främst används i smarta telefoner och pekplattor. Google (2013) skriver att Android bygger på en linuxkärna vilket möjliggör en öppnare miljö för utvecklare och användare och är använt i mer än 190 länder idag. Första versionen av Android kom år 2008 och den nuvarande versionen 4,2 heter Jellybean och kom i slutet av 2012.

Android använder sig av det plattformsoberoende språket Java som programmeringsspråk vilket gör att andra plattformar kan tolka språket Java vid körning. Pehrson (2011) skriver att Android dock använder sig av ett skräddarsytt programpaket JRE(Java Runtime Environment) vilket gör att enbart appar skapade i Android kan köras i smarta telefoner med Android som operativsystem. Pehrson (2011) skriver att det är ett hinder vid utveckling mellan andra mobila operativsystem.

Android använder flera olika verktyg baserade på en utvecklingsmiljö som heter Android SDK(Software Development Kit). För att kunna utveckla mjukavara i Java behövs ett utvecklingsverktyg ett s.k. IDE(Integrated Development Environment) som exempelvis kan vara Eclipse och för att kunna utveckla nativeappar behövs ytterligare bibliotek som heter

Dalvik som är ett JRE för Android. Vid utveckling av appar på Android behövs även en emulator för att kunna testa appen direkt i IDE, denna emulator ingår i Android SDK.

Android har en appbutik som de kallar ”Google Play” där mobila appar finns tillgängligt att ladda hem. Google (2013) skriver att det i oktober 2012 fanns ungefär 700.000 olika appar tillgängliga. För att kunna utveckla appar behövs ingen summa betalas men ska appen göras tillgänglig på ”Google Play” tas en engångsumma på första appen på 25$. Dessutom tar Google 30 % av hela försäljningen av appen. Innan en app kan läggas upp på Google Play ska den granskas för att inga skadliga appar ska finnas på Google Play. Det som granskas är den kod som skickas in, vilket görs i en speciell miljö kallad en sandlådebox.

IOS

IOS är ett mobilt operativsystem som är utvecklat och ägs av Apple Inc. Apple (2013) skriver att deras första smarta telefoner iPhone lanserades år 2007 och har sedan släppt en ny version under varje år till 2012. Den senaste modellen heter iPhone 5 och använder sig av operativsystemet IOS 6.1.3. IOS bygger på en hybridkärna som heter XNU(Not Unix) vilket används i macdatorns operativsystem OS X.

Alasdair (2012) skriver att IOS använder sig av programmeringsspråket Objective-C vilket är ett plattformsberoende språk som enbart går att kompilera på enheter som kör IOS eller OS X. För att kunna utveckla till IOS behöver utvecklaren registrera sig som utvecklare hos Apple vilket kostar 99 $ per år, vilket gör det möjligt för utvecklaren att ladda ner programvara och den IOS SDK som är nödvändig för att skapa appar på plattformen. IOS SDK innehåller bl.a. bibliotek för att kunna använda hårdvarunära funktioner som lokalisering av mobil och kamera samt för att kunna göra transaktioner av data över nätverk. Alasdair (2012) skriver att ett av de vanligaste IDE:rna för att utveckla appar på IOS är Xcode och att det behövs en macdator med operativsystemet OS X för att kunna använda Xcode och programmera appar till IOS.

Apple (2013) skriver att de tillhandlahåller en appbutik som kallas ”App Store” där användare kan ladda hem appar till IOS. För att kunna ladda hem appar behövs ett gratiskonto på mediabiblioteket iTunes som är en samlingstjänst för bl.a. musik, filmer och böcker för IOS och OS X. Apple (2013) skriver att de i januari 2013 har cirka 780.000 olika appar att tillgå för nedladdning. För att kunna distribuera appar till ”App Store” behöver utvecklaren sedan tidigare vara registrerad och godkänd som IOS utvecklare. Apple (2013) skriver att det är gratis att publicera appar men att Apple tar 30 % av intäkterna från appen. Efter det granskas appen för att se att innehållet är tillförlitligt i form av kod och inget olagligt innehåll finns. Det görs även en koll att utvecklaren har använt sig av IOS SDK eller tredje-part tillägg som Apple anser som godkända att använda.

Windows Phone

Windows Phone är ett mobilt operativsystem som utvecklas och drivs av Microsoft. Windows Phone bygger på en tidigare plattform, Windows Mobile som har Windows CE som kärna. Microsoft (2013) skriver att en ny version av Windows Phone släpptes i samband med Windows 8 lanseringen i oktober 2012 som kallas Windows Phone 8 som bygger på en Windows NT kärna. Skillnaden mellan dessa två kärnor gör det inte möjligt att migrera mellan den tidigare Windows Phone versionen Windows Phone 7 och den nyaste versionen.

Sedan Windows Phone 7 har operativsystemet använts på smarta telefoner som tillhandahåller appar för både företag och privatpersoner. I samband med Windows 8 lanseringen som Windows Phone 8 bygger på finns det möjligt att använda ett nytt gränssnitt som heter ”Modern UI”. Microsoft (2013) skriver att ”Modern UI” gör det möjligt för användaren att interagera mer med applikationen. Ett exempel är att kunna titta på startskärmen eller sin smarta telefon om en app har förändrats sedan sist användaren tittade. Användaren slipper att gå in i själva appen för att se om exempelvis ett nytt meddelande har inkommit eller om vädret har förändrats.

Lee & Chuvyrov (2012) skriver att Windows Phone använder sig av Microsoft egna programmeringsspråk c# eller Visual Basic inom plattformen .NET. Utöver c# eller Visual Basic används det deklarativa språket XAML(Extensible Avalon Markup Language) för att kunna utveckla och definiera gränssnittets element. Språket c# är helt plattformsberoende vilket gör att apparna i Windows Phone bara går att använda på .NET plattformen. Microsoft (2013) skriver att för att kunna utveckla på plattformen behövs ett mobilt SDK till utvecklingsmiljön vilket är Windows Phone 8 SDK, vilket innehåller bibliotek för att funktioner i Windows Phone 8 för att bl.a. kunna använda kartor och positionering eller för att kunna låsa telefonen. Det IDE som används är Visual Studio 2012 och för att kunna utveckla på plattformen behövs ett exemplar av Windows 8 som operativsystem. I SDK finns även en emulator baserat på en virtuell maskin som gör det möjligt att visuellt titta och testa appen under utveckling.

Windows Phone har en appbutik som kallas ”Windows Marketplace” där Microsoft (2013) skriver att i januari 2013 finns det cirka 120.000 unika appar tillgängliga för användarna. Lee & Chuvyrov (2012) skriver att för att få tillgång till publicitet på ”Windows Marketplace” måste utvecklaren betala en årsavgift på 99$ per år, vilket innebär att utvecklaren kan publicera obegränsat med appar som användaren får betala för samt fem stycken appar användaren får gratis. Innan det går att publicera sin app måste applikationen granskas samt certifieras av Microsoft. Anledningen till det betyder att de måste kolla igenom om appen är tillförlitlig när det gäller kod samt ett Windows API används och att ett språk som används i den region du publicerar existerar.

2.2.4 Apptyper

En app är en förkortning för applikation som är ett körbart program för en eller flera användare. I studien kommer app betecknas som en program som körs och används i mobilen. En app kan ha många syften för användaren, exempelvis textredigering av dokument, en GPS, ett spel eller åtkomst till sociala medier. Det finns tre stycken olika typer av appar som används vid utveckling på mobila plattformar. I studien kommer jag ha fokus på två stycken app-typer, vilket är native och hybrid.

Nativeapp

En nativeapp är en applikation på telefonen som liknar ett program som exempelvis kan användas på en dator. Nativeappen består av data och information som finns att tillgå för användaren i exempelvis kontaktboken eller ett i schackspel. Stark (2010) skriver att likt ett datorprogram kan nativeappen använda telefonens hårdvarunära funktioner exempelvis kamera eller ljudutgångarna, vilket gör att kraftfulla appar kan skapas där mjuk och hårdvara kan integreras med varandra samtidigt som telefonens resurser i minnet kan användas fullt ut. Stark (2010) skriver vidare att nativeappar kan köpas eller kan laddas hem via en appbutik som finns tillgänglig i telefonen och när appen är nedladdad kan den endast

användas på den egna telefonen vilket gör att varje nativeapp som utvecklas är plattformsberoende. Det är heller inget krav på internetåtkomst för att nativeappen ska fungera. Nativeappen använder olika programmeringsspråk beroende på vilket operativsystem som används, se Tabell 1. Tillborg, Persson & Bentlöv (2012) skriver att eftersom nativeappen är plattformsberoende måste varje specifik app använda olika programmeringsspråk för olika enheter, vilket gör att extra kostnader tillkommer vid utvecklingen om appen finns på flera olika plattformar då det kräver mer programmeringskunskaper hos utvecklaren. Beroende på plattform kan också licens och uppdateringskostander tillkomma för ägaren av plattformen.

Tabell 1 Programmeringsspråk som användas vid utveckling av en nativeapp samt var den är tillgänglig för nedladdning.

Operativsystem	Språk	Hämtas/Tillgänglig
Android	C/C++ & Java	Google Play
Windows Phone 8	C#/XAML & VB.NET	Windows MarketPlace
IOS	Objective-C	App Store via iTunes

Webbapp

En webbapp är en applikation på telefonen som liknar en hemsida vars syfte kan variera. Det kan vara allt från bokning av en kommuns sporthall till en miniräknare. Stark, J. (2010) skriver att en webbapp bygger på webbaserade teknologier som exempelvis HTML, CSS och Javascript. Webbappar är tillgängliga via en webbadess eller en URL(uniform resource locator) som dessa kallas och kräver därför en uppkoppling mot internet via en webbläsare för att appen ska kunna gå att använda, se Tabell 2. Mobilmedia (2012) skriver att webbappen går att använda oberoende operativsystem eller plattform, vilket möjliggör att många telefoner samtidigt kan nå applikationen. Stark (2010) skriver också att en webbapp inte kan vara installerad lokalt på telefonen eller laddas ner via en appbutik utan används direkt via en URL. Att appen inte används lokalt gör att telefonens hårdvarunära funktioner inte kan användas fullt ut. Mobilmedia (2012) skriver även att vissa kodspråk som exempelvis flash i inte kan användas på vissa plattformar.

Tabell 2 Programmeringsspråk som användas vid utveckling av en webbapp samt var den är tillgänglig för nedladdning eller användning.

Operativsystem	Språk	Hämtas/Tillgänglig
Android	HTML/CSS/Javascript	URL
Windows Phone 8	HTML/CSS/Javascript	URL
IOS	HTML/CSS/Javascript	URL

Hybridapp
En hybridapp är en applikation på telefonen som fungerar likt en nativeapp men har egenskaper som en webbapp, där av namnet hybridapp. Ghatol & Patel (2012) skriver att en hybridapp har egenskaper som en nativeapp i form att det går att kommunicera med vissa av telefonens hårdvarufunktioner. Hybridappen har också egenskaper likt en webbapp som använder funktioner med hjälp av HTML/CSS och JavaScript. Exempel på funktioner är bl.a. gränssnitt, kommunikation med server och applikationslogik. Tillborg, Persson & Bentlöv (2012) skriver att hybridappar bygger på framtagna ramverk som gör att appen blir kompatibel med andra plattformar än enbart målplattformen. En hybridapp går att ladda hem via telefonens appbutik och efter installation är appen tillgänglig via telefonens gränssnitt, se Tabell 3. Tillborg, Persson & Bentlöv (2012) skriver också att uppdateringar av appen möjliggör besparingar ifall den finns tillgänglig på flera plattformar med samma kodbas, eftersom det inte då finns olika programmeringsspråk att beakta vid utvecklingen. De kostnader som kan komma är i form av licens och uppdateringskostander tillkomma för ägaren av plattformen eftersom appen läggs upp i appbutiken. En av de företag som provat på hybridappen är Handelsbanken. Genom att ha en webbdel där användaren kopplar upp sig via internet och kan se sitt saldo och göra transaktioner samt att ha en nativedel där användaren kan använda kameran för att scanna in ett OCR-nummer(optisk teckenläsning). Genom att ha en webbdel som kan uppdateras externt behövs inte lika många uppdateringar av själva appen i appbutiken göras, vilket gör besparingar då bara en uppdatering behöver göras i webbdelen. Pehrson (2011) skriver att tester som gjorts har dock påvisat att multiplattformsutvecklingen har sämre prestanda än de plattformsoberoende SDKn. Främst har brister i minneshanteringen visats sig då multiplattformsutvecklingen kan använda upp till sju gånger mer minne än ett plattformsberoende SDK vid en databastransaktion enligt Pehrson (2011).

Tabell 3 Programmeringsspråk som användas vid utveckling av en hybridapp samt var den är tillgänglig för nedladdning.

Operativsystem	Språk	Hämtas/Tillgänglig
Android	HTML/CSS/Javascript	Google Play
Windows Phone 8	HTML/CSS/Javascript	Windows MarketPlace
IOS	HTML/CSS/Javascript	App Store via iTunes

2.2.5 Ramverk & API
När en utvecklare sitter och skriver programkod används ofta ett ramverk för att underlätta programmeringen. Pressman (2010) skriver att ett ramverk är en generisk lösning för ett specifikt problem, vilket skulle kunna ses som en skelettkropp utan armar eller ben där den lösa armen och benet behöver integreras med kroppen för att kunna fungera. Från en utvecklares perspektiv skulle kroppen kunna vara ett specifikt domänproblem eller programmeringsproblem. Ett ramverk gör det möjligt för armen och benen att samarbeta med varandra fast de har olika förutsättningar. Riehle (2000) skriver att objekt-orienterade ramverk används i stor utsträckning för att öka produktiviteten hos utvecklarna samt möjliggöra att produkten kan utvecklas snabbare till marknaden genom att återanvända

design och programmeringskod. Ett ramverk gör det möjligt att lättare använda funktioner som anses som komplicerade för utvecklaren, vilket exempelvis kan vara kryptering, fil och minneshantering samt databashantering. I ett ramverk kan det finnas olika klassbibliotek och kompilatorer som kallas API:er.

Ett API gör det möjligt för kommunikation och integration mellan enheter. I ett API kan det finnas flera bibliotek som gör det möjligt för utvecklaren att använda en viss funktion eller resurs på ett förenklat sätt. Det kan vara systemnära resurser som operativsystemet använder eller funktioner som att skicka e-post. Jacobson, Brail & Woods (2011) skriver att ett API antingen kan vara privata API som företag använder för att kunna utveckla egna produkter eller publika API:s. Publika API:s är exempelvis Facebook och Twitter för mobila applikationer, de möjliggör att utvecklare kan integrera deras tjänster med egengjorda applikationer på plattformen. Jacobson, Brail & Woods (2011) skriver också att det finns tredje-parts API:er på marknaden. Ett av dem som använder tredje-parts API:er är det mobila ramverket PhoneGap.

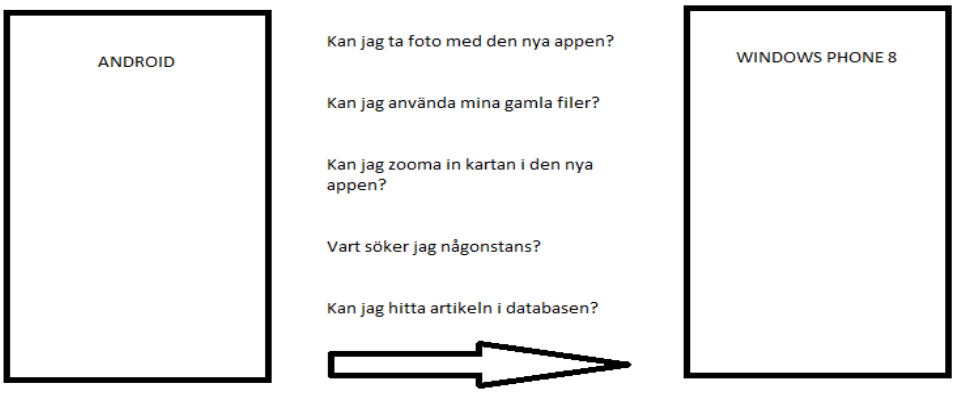
PhoneGap är ett öppet källkodsbaserat mobilt ramverk som använder standardiserade webb API:er för att kunna utveckla på flertalet olika mobila plattformar. De använder sig av webbaserade programmeringsspråk som HTML, CSS och JavaScript som gör det möjligt för plattformsoberoende lösningar på olika plattformar. PhoneGap (2013) skriver att de använder flertalet API:er för att göra det möjligt att använda systemnära resurser som kamera, filer och lagring på telefonen.

3 Problemformulering

I kapitlet beskrivs problemet för studien och varför det är viktigt att lösa problemet. I början av kapitlet introduceras problemet kort för att vidare i kapitlet gå in i detalj. Problemet beskrivs allmänt och vad problemet innebär för Sogeti som leverantör av appar. Kapitlet tar också upp syfte samt en huvudfråga och två delproblem kopplade till huvudfrågan. Kapitlet avslutas med ett förväntat resultat och en avgränsning för problemet.

3.1 Problembeskrivning

Det finns idag många företag som redan har sina appar tillgängliga på operativsystemen Android och IOS. I samband med att andra plattformar och operativsystem som Windows Phone 8 växer fram behöver appar konverteras till en ytterligare plattform. Anledningen till det leder till både ökade kostnader i form av utveckling och plattformskostnader vilket är tids och resurskrävande för både kund och leverantörer. Företagen som använder produkten har också ofta helt olika krav på sin app. En del företag kräver inte mycket resurser i form av exempelvis hårdvara av telefonen, medan en del företag vill använda avancerad teknik som video och ljuduppspelning med speciella mediaspelare. Beroende på vilken typ av app ett företag behöver kan helt olika metoder för migration mellan plattformar behöva användas.



Figur 3 – Samma app fungerar inte likadant från en plattform till en annan plattform

När en leverantör tar på sig uppdraget att migrera en app från en kunds tidigare plattform till en ny plattform måste leverantören undersöka helheten med den tidigare appen. Kunden som efterfrågar appen på den nya plattformen vill i största möjliga mån känna igen sig i användandet som finns i den befintliga appen. Problemet är att gränssnittet och utformningen av apparna ofta skiljer sig åt mellan plattformarna. Det som är representerat i IOS eller Android behöver inte vara sig likt i Windows Phone 8. Ett annat problem är att funktioner som finns och fungerar i IOS eller Android kanske inte alls fungerar i Windows Phone 8, vilket gör att information och viktig data utesluts i appen på den nya plattformen.

Frågan som leverantörer ofta ställer sig är om de på något sätt går att hålla ner kostnaderna när en befintlig app på en plattform ska flyttas till en ny plattform. Precis som Pehrson (2011) skriver finns det problem angående arkitekturen mellan olika plattformar. Eftersom plattformar ofta använder plattformsberoende funktioner och kod blir det ett problem när en funktion eller data ska flyttas direkt mellan den gamla och den nya plattformen. Frågan är då om det går att hitta ett standardiserat sätt för att migrera appar mellan plattformar och

speciellt till Windows Phone 8. Det har visat sig att det inte funnits några tidigare arbeten eller litteratur som behandlat just migration till Windows Phone 8, vilket gör problemområdet extra intressant eftersom det är nytt samt vad för resultat som kan hittas inom området.

Går det att flytta över befintlig data och information till Windows Phone 8 för att användaren ska få en likvärdig upplevelse från den gamla plattformen? Är det möjligt att få gränssnitt och funktioner att likna varandra och kan det göras på ett effektivt sätt med avseende på tid och resurser? Finns det befintliga ramverk och API:er som kan stödja en migration mellan plattformar eller behövs nyutveckling och hur kan resultatet bli?

Sogeti har flertalet kunder med olika behov och krav i användningen av sina appar. En del kunder behöver använda appar som kräver mycket av telefonens minne och lagring, en del kunder vill ha en enklare applikation som endast behöver läsa och skriva data från en databas, en del har krav på att appen ska köras direkt i telefonen och inte via en webbläsare. Kunden vill också kunna ta del av appen snarast möjligt efter det att projektet startat och dessutom vill de att kostnaderna hålls nere.

Det är också viktigt för Sogeti att migrationsmetoden går att standardisera, vilket betyder att metoden går att använda i riktiga projekt. Vilka typer av roller och kompetenser i form av personal behövs, kan fria verktyg och språk användas och finns det dokumentation och support att tillgå från utvecklarna av metoden.

3.2 Frågeställning och syfte

Syften med studien är att undersöka ifall det på en kostnad och tidseffektivt sätt går att hitta en lösning för att migrera appar till Windows Phone 8. Syftet är också att hitta en lösning som underlättar för utvecklare att underhålla appar som migrerats. Syftet med studien är också till för att det ska underlätta för företag eller andra intressenter att få ökad kunskap inom migration mellan mobila plattformar.

Huvudfrågan för examensarbetet som ska besvaras är följande:

- Vilken eller vilka metoder kan användas för att migrera befintliga appar till operativsystemet Windows Phone 8?

För att kunna besvara huvudfrågan har två underfrågor skapats.

- Vilka för och nackdelar finns det med att använda olika app-typer?
- Vilken eller vilka metoder för migration mellan plattformar är möjlig att använda för att kunna implementera en enklare prototyp av en app?

Den första frågan måste besvaras för att avgöra vilken app-typ som kan användas tillsammans med de migrationsmetoder som finns till Windows Phone 8. Genom att undersöka generella för och nackdelar med olika app-typer kan migrationsmetoder lättare väljas ut. Den andra frågan måste besvaras för att kunna avgöra om migrationsmetoden fungerar praktiskt genom en implementation av olika utvalda funktionaliteter till en prototyp skapad för Windows Phone 8. Tillsammans bidrar de till att kunna besvara huvudfrågan för studien.

3.3 Avgränsning

Studien är avgränsad till att välja ut migrationsmetoder som har .NET eller olika webbspråk som kodbas. Metoderna ska också vara avgränsade till app-typerna native eller hybrid eftersom det går att testa hårdvarunära funktionalitet med dessa två. Metoderna är också avgränsade till att implementeras i en prototyp med Windows Phone 8 SDK som bakomliggande uppsättning av verktyg.

3.4 Förväntat resultat

Det förväntade resultatet är att få fram en migrationsmetod som kan stå till grund och säkerhetsställa att funktioner från en app baserat på IOS eller Android kan migreras och köras i en .NET miljö som finns representerat i Windows Phone 8.

4 Metod

I kapitlet beskrivs de forskningsmetoder och den implementationsmetod som används i studien. Metoderna beskriver det arbetssätt som planerats vid insamling av data till resultatet och det praktiska arbetet med prototypen.

4.1 Metodval

I studien har olika metoder valts för att kunna besvara den frågeställning som finns i studien. För att kunna besvara huvudfrågeställningen har en kombination av olika metoder valts ut som ska hjälpa till att besvara de delfrågor som finns i studien.

För att kunna besvara den första delen av frågeställningen *”Vilka för och nackdelar finns det med att använda olika app-typer?”* ska en kvalitativ ansats med inriktning på intervjuer samt en litteraturstudie göras. Patel & Davidsson (2003) skriver att en kvalitativ inriktad forskning ofta fokuserar på datainsamling med mjuk data och att mänskliga aspekter med deras egna tolkningar kring ett ämne inkluderas i insamlingen. En litteraturstudie är när information samlas in genom litterära verk för att ge svar kring ett problemområde. Winter (1992) skriver att en litteraturstudie ska ge ett teoretiskt ramverk som ska beskriva centrala och nyckelbegrepp kring problemområdet. Patel & Davidsson (2003) skriver att en litteraturstudie kan göras var och när som helst och planeras fritt av forskaren.

En kvalitativ ansatts med inriktning på intervjuer har valts på denna fråga eftersom att det funnits möjlighet till kontakt med personer ute på företaget med kunskaper inom området, vilket har lett till en uppfattning och tolkning samt en djupare förståelse för problemområdet. För att få en start på studien har en litteraturstudie valts som är tänkt att ge en känsla för berörda ämnen som appar, mobila operativsystem samt systemmigration generellt. Litteraturstudien är tänkt att ge information till första delfrågeställningen samt att ge ”kött på benen” inför den andra delfrågeställningen. Litteraturen har sedan använts för att komplettera de svar som framkommit i samband med intervjuerna till respektive ämne.

För att kunna besvara den andra delen av frågeställningen *”Vilken eller vilka metoder för migration mellan plattformar är möjlig att använda för att kunna implementera en enklare prototyp av en app?”* ska en induktiv forskningsansats med deduktiva inslag användas. Patel & Davidsson (2003) skriver att vid en induktiv forskningsansats samlar forskaren in data och information utan att först förankra den. Det innebär att den insamlade informationen därför skapar en egen teori. En induktiv forskningsansats innebär att genom empirin kunna ge ett resultat och analys från ett problem. Med deduktiva inslag skriver Patel & Davidsson (2003) att genom befintliga teorier dras slutsatser om enskilda företeelser. I deduktiva inslag anses objektiviteten i tidigare forskning vara positivt då forskarens subjektivitet i ämnet kan minska.

Det induktiva angripssättet har valts eftersom problemet är nytt och att det funnits en liten mängd teoretisk grund om migration till Windows Phone 8. Genom att samla in information om allmän systemmigration och mobil utveckling har problemområden hittats. För att kunna dra en slutsats och nå bevis huruvida mobil systemmigration till Windows Phone 8 fungerar eller inte ska deduktiva inslag användas. Med deduktiva inslag har ett val gjorts genom att titta på enskilda mobila ramverk och API:er för att se om det fungerar att migrera mellan mobila plattformar till Windows Phone 8. För att kunna få hjälp angående valda ramverk har en dokumentguide valts som hjälpmedel pga. det finns liten vetenskaplig forskning kring dessa enskilda ramverk.

Till implementeringen har en bottom-up metod valts. Ståhl (2012) skriver att denna metod går ut på att börja längst ner i arkitekturen vid kärnan och få en eller flera moduler att fungera tillsammans. När de fungerar och applikationen kan kompilera på plattformen utan fel byggs ytterligare en modul in i applikationen, vilket gör att funktionaliteten kan testas på plattformen i olika steg, vilket gör att fel och problem påvisas på ett enkelt sätt. Implementeringsmetoden har valts pga. tiden som avsett för studien då det går snabbare att arbeta direkt med funktionerna istället för att börja arbeta med de grafiska elementen.

4.2 Intervjuer

I studien har kvalitativa intervjuer med fokus på låg standardisering valts. Trost (2010) skriver att låg standardisering innebär att frågekonstruktören anpassar intervjun efter den rådande andan. Det behövs inte någon speciell ordning på frågorna och följdfrågor kan ställas när de passar in i sammanhanget under intervjun. Vid låg standardisering är det variationsmöjligheterna som ger en styrka vid intervjutillfället. I studien har också ostrukturerade intervjufrågor valts. Trost (2010) skriver att ostrukturerade frågor ger en öppenhet till intervjukandidatens svar. Det finns möjlighet för denne att svara på ett helt annat sätt än vad frågekonstruktören tänkt sig från början som är i kontrast till att ha förbestämda svar på de frågor som ges till intervjukandidaten.

Låg standardisering har valts för intervjupersonen skulle kunna ge mycket och detaljerad information kring frågorna, vilket skulle öka kunskapen för ämnet. Om personer med insikt inom ämnet fick tala fritt ansågs mer information kunna utvinnas och som i slutändan skulle kunna ge en bredare teori. Ostrukturerade intervjuer valdes i samband med valet av låg standardisering, vilket gjordes pga. att intervjuerna också skulle ge ett öppnare svar än om svarsalternativ skulle väljas. Eftersom det finns en risk att intervjukandidaterna svarar helt skilt på vissa frågor har denna risk beaktats. I de fall där svaren skiljer sig avsevärt efter intervjuanalysen ska dessa följas upp ytterligare mot litteratur för att säkerställa svaret om det är betydande för studiens resultat.

Patel & Davidsson (2003) skriver att det finns två stycken olika sätt att registrera vad som sägs under intervjun. Det första alternativet är att spela in intervjupersonen för att sedan sammanställa hela intervjun vad intervjupersonen sagt, vilket ofta tar lång tid då hela intervjun skrivs ut på papper. Det andra alternativet är att anteckna vad intervjukandidaten säger under intervjun, vilket inte tar lika lång tid att sammanställa men intervjun måste sammanställas direkt efter intervjun för att inte glömma något viktigt.

I studien har jag valt att göra sammanfattningar och gå igenom vad som sägs direkt efter intervjun. Anledningen till det är för att inte glömma något viktigt. Det som sammanfattats under intervjun och det material som är tänkt för resultatet ska konfirmeras av intervjupersonen.

4.2.1 Intervjufrågor

I studien har fyra stycken olika kategorier av intervjufrågor valts. Trost (2010) skriver att en intervju blir mer seriös om det finns ett sammanhang eller ett fokus om vad intervjun går ut på. I enlighet med valet av ostrukturerade intervjuer är intervjufrågorna i listan de frågor som ställts under intervjutillfällena, dock är det inte alltid samma frågor som ställts för varje deltagare. Frågorna ställdes efter den rådande andan och efter vilket ämne en intervjukandidat intresserade sig mer för och pratade mer om.

Introduktion

- Berätta kort om vad studien går ut på.
- Syfte med intervjun och upplägg på frågorna.
- Hur resultatet kommer att användas.

Intervjuperson och arbete

- Vad har du för yrkesroll här på företaget och vad du gör en typisk dag på arbetet?
- Hur länge har du arbetat inom yrket/liknande yrken?
- Hur länge har du arbetat på företaget?

Kunder och app-utveckling

- Hur många konsulter har ni aktiva och hur många arbetar med app-utveckling idag?
- Vilka plattformar efterfrågar kunderna idag?
    - Tror du att kunder kommer efterfråga Windows Phone 8? Varför? / Varför inte?
- Hur anser du att utvecklingskostander kan hållas nere i samband med app utveckling?
- Har du tidigare arbetat med migration mellan olika plattformar på befintliga appar ex, mellan IOS och Android?
    - Hur har arbetet fungerat? Har det funnits svårigheter?
- Har du tidigare arbetat med migration mellan Android och Windows Phone (7, 7.5)?
    - Hur har arbetet fungerat? Har det funnits svårigheter?
- Vilka olika typer av appar anser du att det finns?
    - Vad har du för åsikter gällande dem (fördelar/nackdelar)?

Avslutning

- Är det något du vill lägga till eller kommentera?
- Finns det andra personer på företaget som skulle vara lämpliga att intervjua?
- Tack!

4.3 Implementering

I studien ska en implementering göras för att testa om de migrationsmetoder som valts ut fungerar i praktiken. För att komma fram hur prototypen ska konstrueras ska en planering av implementationen ske samt ett urval göras för vad som ska finnas med i prototypen. Denna implementering ska göras med hjälp av den bottom-up metod som valts ut för studien.

4.3.1 Planering

Innan arbetet med prototypen startar görs en planering av implementationen i fyra steg.

- Förberedelse
- Val av prototyp
- Implementation
- Utvärdering

Förberedelse handlar om att införskaffa kunskap inom området för mobila plattformar och ramverk som kan vara till nytta. *Val av prototyp* handlar om att utifrån ett visst antal befintliga appar välja ut tre stycken funktionaliteter i ett system inför en implementation av prototyp. *Implementation* handlar om att försöka implementera de tre olika funktionerna

som valdes i urvalet till prototypen. De tre funktionerna ska implementeras i varje typ av migrationsmetod. *Utvärdering* handlar om se om metoden stödjer de tre funktioner som valts ut och för att kunna avgöra om migrationsmetoden fungerar praktiskt eller inte för ett riktigt projekt.

4.3.2 Urval till prototyp

För att kunna hitta på en prototyp som ska användas i studien ska populära och rekommenderade gratisappar i IOS och Androids appbutiker och som inte är ett spel användas. Anledningen till det är för att kunna hitta funktioner som är vanligt förekommande i en app idag.

4.3.3 Implementeringsmetod

I studien ska en bottom-up metod användas för att implementera prototypen. Bottom-up innebär att börja arbetet nära applikationens kärna som möjligt dvs. nära systemmoduler och hårdvara. När en viss modul fungerar som tänkt börjar utvecklaren med nästa närliggande modul och arbetar uppåt mot applikationens mer visuella delar. I implementeringsmetoden har fyra stycken olika uppgifter planerats för varje funktionalitet från urvalet som ska implementeras i samband med migrationsmetoden:

- Inkludera bibliotek och nödvändiga resurser för vald funktionalitet
- Initiera och skapa metoder som kan ta emot data från bibliotek och API:er
- Skapa metoder som möjliggör körning av funktionaliteten på emulator i IDE
- Köra igång emulatorn för att se att funktionaliteten fungerar eller inte
    - Om inte, återgå till steg 2 för att söka efter fel.

5 Genomförande

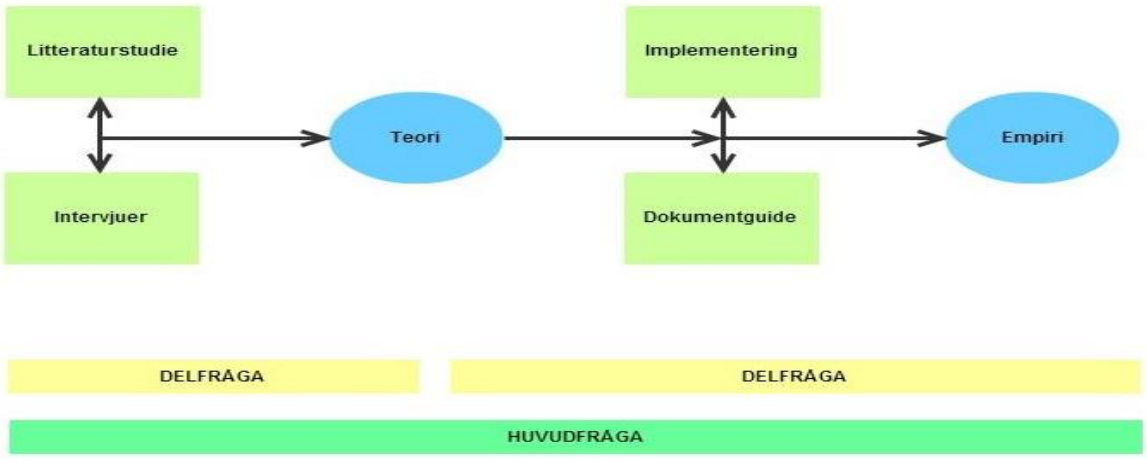
I kapitlet redovisas hur genomförandet av studien gått till. Kapitlet börjar med att beskriva tillvägagångsättet av studien med hjälp av de utvalda metoderna. Kapitlet innefattar också en beskrivning av insamlingen av data genom en litteraturstudie, dokumentguide samt intervjuer. Kapitlet avslutas med hur implementeringen av prototypen gått till metodologiskt och praktiskt.

5.1 Tillvägagångsätt

Studien startade med en kortare introduktion till arbetsplatsen och en träff med arbetarna på företaget tillsammans med handledaren. I samråd med handledaren diskuterade vi fram en uppgift som kunde passa båda parter för studien. Det företaget ville ha hjälp med är hur en eventuell migration mellan mobila plattformar skulle kunna ske och om det kunde göras till Windows Phone 8 på ett effektivt sätt.

Tillvägagångssättet för att kunna besvara huvudfrågan har innefattat en litteraturstudie, intervjuer med personer på företaget, en dokumentguide samt en implementering av en prototyp, se Figur 4. För att få information angående problemområdet har en litteraturstudie med insamling av centrala begrepp inom området gjorts, vilket kombinerats med intervjuer med olika personer på företaget med insikt i app-utveckling. Tillsammans har denna insamling bidragit till en teori där intervjuerna räknas in som empiri och använts för att besvara delfrågan *”Vilka för och nackdelar finns det med att använda olika app-typer?”*. Teorin har sedan använts som en del av en bakomliggande grund för ämnet systemmigration inför implementeringen av prototypen. En dokumentguide har gjorts i samband med implementeringen för att få hjälp och vägledning med specifika programmeringsproblem samt för dokumentation om ramverk och API:er. Genom att samla ihop data har en teori kunnat bildas. Teorin har sedan kompletterats med egna upptäcker inom mobil migration till Windows Phone 8 och en egen empiri har därför kunnat skapas. Genom att använda teorin plus egna upptäckter har den andra delfrågan kunnat besvaras *”Vilken eller vilka metoder för migration mellan plattformar är möjlig att använda för att kunna implementera en enklare prototyp av en app?”*

Det samlade resultatet av det som kommit ut ur tillvägagångsättet besvarar huvudfrågan*” Vilken eller vilka metoder kan användas för att migrera befintliga appar till operativsystemet Windows Phone 8?”*



Figur 4 – Tillvägagångsätt vid genomförandet av studien

5.2 Datainsamling

De data som samlats in i studien kommer från tre olika källor, intervjuer, dokument och litteratur.

I början av studien gjordes en litteraturstudie för att hitta viktiga centrala begrepp som kunde komma till hands i studien. Information eftersöktes i böcker på bibliotek genom att söka i deras databaser efter ord som *"Apple, Android, systemmigration, app och Windows Phone*". Utöver böcker användes också tidigare forskningar i form av fulltexter från digitala vetenskapliga arkivet. I det digitala vetenskapliga arkivet gjordes samma typer av sökningar som i bibliotektes databas. När en text hittades lästes sammanfattningen för att avgöra om det fanns intressanta fynd att hitta. En tanke med litteraturstudie var att få en grund och förberedelse inför kommande intervjuer med personer på företaget. Litteraturstudien gjordes också för att kunna samla in resultat till en av delfrågeställningen *"Vilka för och nackdelar finns det med att använda olika app-typer?"* samt att få grundläggande information gällande den andra delfrågeställningen *"Vilken eller vilka metoder för migration mellan plattformar är möjlig att använda för att kunna implementera en enklare prototyp av en app?"*.

Strax efter litteraturstudien var klar började intervjuer göras för kompletterande data som skulle leda till teori. Under studien gjordes totalt två stycken intervjuer, båda med personer med kunskap om app-utveckling generellt och inom företaget. När intervjukandidater skulle hittas och väljas ut började jag med att göra en intervju med min handledare på företaget. Efter intervjun fick jag information om vem eller vilka som skulle kunna vara tänkbara kandidater för ytterligare intervjuer. Sedan togs en första kontakt via mail som innehöll om vad min studie på företaget handlade om samt om de kunde tänka sig ställa upp på en intervju. Datum och tid valdes ut genom mail på företaget där båda parter fick godkänna om en intervju kunde ske på utsatt datum och tid.

När intervjun var bekräftad och ett datum och tid var utsatt gjordes en intervju i ett ostört konferensrum på kontoret. Vid inledningen av intervjuerna har också forskningsetiska principer nämnts för intervjukandidaterna. Intervjukandidaten fick en repetition om vad studien skulle handla om samt information att studien skulle komma att publiceras när den var klar. Intervjudeltagarna fick också information om att de skulle vara anonyma och att deras namn inte skulle publiceras i samband med studien eftersom de bad om det. Intervjuerna började alltid med lite småprat för att sedan starta själva intervjun.

Första kategorin från intervjufrågorna var en *introduktion* där intervjupersonen fick en kort förståelse för studie och hur intervjun kommer att gå till. Andra kategorin handlade om *intervjuperson och dess arbete* där intervjupersonen fick frågor om sitt arbete och sin yrkesroll. Den tredje kategorin handlade om *kunder och app-utveckling* där intervjupersonen fick frågor om deras arbete kring app-utveckling på företaget samt frågor gällande deras kunder. Den fjärde och sista kategorin var en *avslutning* där intervjupersonen fick frågor om hur den upplevde intervjun samt ett tack för den ställde upp på en intervju. De fyra kategorierna ställdes alltid i samma följd men frågorna inom kategorin hade inte alltid samma följd.

Under intervjuerna ställdes frågor utifrån hur samtalet och situationen artade sig. Anledningen till det är för att intervjupersonen skulle kunna ha chans att ge detaljerade och egna svar som möjligt. Ifall intervjupersonen kom in på ett område denne gillade skulle det kunna leda till bättre svar än att avbryta och gå vidare med nästa fråga. Under intervjuförloppet ställdes även följdfrågor titt som tätt. Det var just följdfrågorna som var

viktiga, eftersom intervjukandidaten gav ett djupare resonemang kring ämnet. En intervju pågick under cirka 20-30 minuter och antecknades alltid med penna och papper

Efter en intervju sammanställdes anteckningarna av intervjun. När intervjun var över togs anteckningsblocket fram för att se över vad som sades. Det var viktigt att göra sammanställningen direkt efter för att inte glömma av något av intresse. Intervjusammanställningen fördes över från papper till en textredigerare på datorn där en prioriteringslista gjordes. Denna prioriteringslista gjordes på en grund av vad intervjupersonen tyckte var det viktigaste med intervjun samt vad jag tyckte var det viktigaste som kom ut från intervjun. Intervjupersonen fick också konfirmera att det som material som samlats in från intervjun stämde för att det skulle kunna ligga som en grund för delar av resultatet. De svar som sammanfattades efter intervjuerna hade inga nämndvärda skillnader som kunde påverka resultatet för studien. När intervjusammanställningen var gjord och de viktigaste upptäckterna från intervjun var gjorda vägdes dem in i resultatet för delfrågeställningen *”Vilka för och nackdelar finns det med att använda olika app-typer?”*.

Eftersom att Windows Phone 8 är nytt och det inte finns mycket forskning kring migration till plattformen har en dokumentguide använts som ett stöd. Under hela implementationen användes en dokumentguide som stöd och innehöll bl.a. forum och hjälpande webbsidor. I början av dokumentguiden gjordes sökning efter olika migrationsmetoder och mobila ramverk och i syfte för att hjälpa till med implementationen av prototypen. För att hitta dokument har främst Google använts som sökmotor med fraser som ”*mobile frameworks, cross-based app development, app migration for windows phone 8*”. Dokumentguiden har sedan använts för att hjälpa till vid implementationen av prototypen där olika programmeringsproblem har eftersökts, vilket främst gjorts på de webbsidor som tillhandahåller information om mobila ramverk och API:er eller företagens egna webbsidor men också genom olika programmeringsforum.

5.3 Implementation

Vid implementationens inledning hade dokumentation angående vilka olika verktyg som behövdes lästs igenom för att kunna utveckla med de valda migrationsmetoderna. Verktyg som laddades ner och installerades var bl.a. Visual Studio som IDE, ramverk för metoden samt Windows 8 SDK. Det IDE som valdes ut hade använts förut. Anledningen till det är för att det inte skulle ta någon längre tid att sätta sig in i hur verktyget fungerade.

Val av prototyp

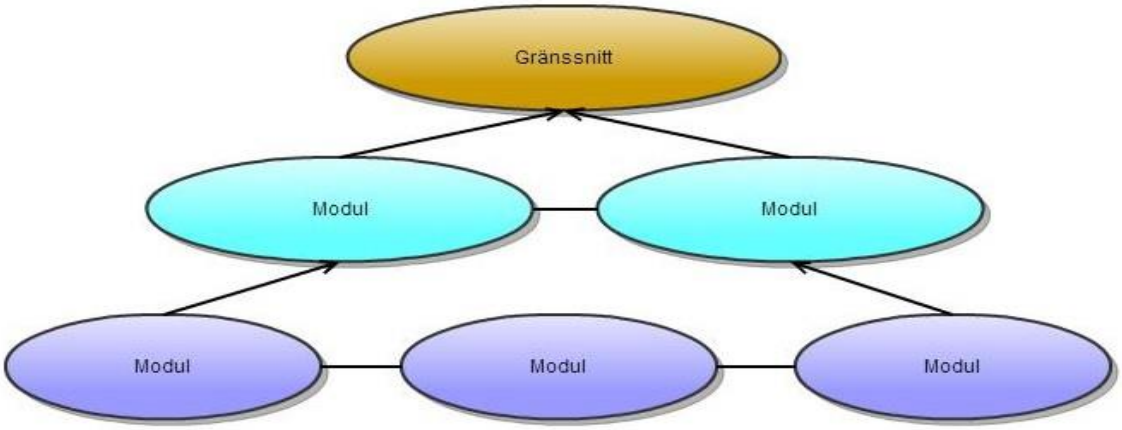
Efter att ha sökt i appbutikerna Google Play och App Store valdes tre stycken olika funktionaliteter ut. Funktionaliteterna implementerades och testades med de migrationsmetoder som använts i studien. De olika funktionaliteter som valts ut är:

- Kunna läsa in en .pdf (Adobe Portable Document) fil och sedan visa filen från telefonens minne i appens gränssnitt.
- Kunna använda telefonens kamera för att ta kort och sedan visa det senaste kortet som tagits i appens gränssnitt.
- Uppkoppling till en lokal databas för att komma åt data.

Valen som har gjorts kommer från befintliga populära appar. De appar som varit referenser utifrån de valen är bl.a. ”Facebook” där en användare kan skicka exempelvis en PDF fil till en annan person och ladda ner filen till telefonen för att sedan visa filen, ”Instagram” som använder kameran för fototagning, ”tv.nu” för en uppkoppling mot en databas.

Implementation av de olika valen

När den valda funktionaliteten skulle implementeras i prototypen följdes bottom-up metoden, se Figur 5. Eftersom varje metod har sin egen arkitektur, se kapitel 6.2.2, 6.2.3 togs det i beaktning då data i kedjan måste följa ett visst sätt för att till sist kunna visas för användaren.



Figur 5 – Uppbyggnad av Bottom-up metod nedifrån och upp

När implementationen av funktionerna skulle börja kodas startade arbetet genom att titta på hur arkitekturen för migrationsmetoden såg ut. I PhoneGap startades varje funktion med att ge hårdvaruaccess genom att inkludera ramverket med bibliotek och ge eventuella accesser av hårdvaran till appen genom att ändra i konfigurationen. I Xamarin behövdes inte speciell hårdvaruaccess än att inkludera originalbibliotek då arbetet sker direkt genom det SDK som är tänkt att användas för utveckling i Windows Phone 8. Efter det fungerade kunde nästa steg tas som var att inkludera plugin och bibliotek i de båda metoderna för att kunna skapa funktioner som stödjer användning av hårdvaran. Sedan skrevs funktioner som gjorde att den utvalda funktionen skulle fungera, vilket gjordes enligt den strategi som valts för att få funktionerna fungera, se kapitel 4.3.3. Slutligen gjordes en enkel design för gränssnittet som kunde visa upp de funktioner som valts ut.

6 Resultat

I kapitlet redovisas de resultat som framkommit under studien. Resultatet är indelat i tre delar. Första delen som redovisas är de för och nackdelar som hittats genom litteratur och intervjuer angående olika app-typer. Andra delen som redovisas är de olika metoder för migration mellan IOS och Android till Windows Phone 8 som hittats. Den tredje delen som redovisas är den prototyp som framtagits genom att använda implementeringsmetoden bottom-up. Resultatet från prototypen visar om metoden för migration fungerar praktiskt.

Migrationsmetoderna som valts ut uppfyller syftet med studien och genom att använda dem bidra till att app-projekt blir kostnad och tidseffektiva. Genom att använda metoderna kan kod återanvändas mellan olika plattformar och till Windows Phone 8, vilket även bidrar till att utvecklare lättare kan underhålla apparna.

6.1 Apptyper

I kapitlet redovisas de för och nackdelar med olika app-typer som framkommit under arbetet. App-typerna och deras egenskaper har visats sig vara varierande. Eftersom det finns många olika användare med olika behov samt många olika typer av genres av appar är det omöjligt att hitta exakta för och nackdelar för varje enskild användare och app. För att komma fram till ett resultat och besvara delfrågeställningen *”Vilka för och nackdelar finns det med att använda olika app-typer?”* har litteratur samlats in genom tidigare forskning, se kapitel 2.2.4. Det har även gjorts två stycken intervjuer i studien som hjälpt till att besvara frågeställningen, se kapitel 5.1.2. Det har även hittats fördelar vid implementeringen av prototypen, dessa fördelar nämns som empiri. Alla för och nackdelar som hittas har listats under respektive app-typ i punktform.

6.1.1 Fördelar och nackdelar med Nativeappar

En av de två mest använda app-typerna har visat sig vara nativeappar som används i de flesta smarta telefoner idag. En bidragande orsak till att den är vanligt förekommande är att det är lätt för användaren att hitta och ladda hem olika appar som är av typen native. De laddas ner på internet via telefonens appbutik. Vid en intervju med en person på företaget framkom det även att många av deras kunder vill ha nativeappar för användning i verksamheten. Anledningen till det är pga. den stabillitet och känsla i användningen som är en styrka med nativeappen. Även andra delar som push-notiser och andra inbygga funktioner i telefonen är ofta viktiga för kunder. Anledningen till det är för att de då inte missar något viktigt, exempelvis att en vara tagit slut i sortimentet. Om det finns stöd för push-notiser i appen underlättar det vetskapen om att en beställning måste göras. Den stora nackdelen har dock visats sig vara de höga utvecklingskostnaderna pga. plattformsberoendet. Plattformsberoendet gör att utvecklare måste kunna flertalet olika programmeringsspråk när en nativeapp ska migreras till en annan plattform. Under intervjuerna sa en av kandidaterna att det kan vara svårt att hitta utvecklare som kan flera olika programmeringsspråk och som dessutom inte sitter i ett annat projekt för tillfället.

Fördelar

- Enkelt att distribuera en nativeapp via en appbutik. [litteratur, se kapitel 2.2.3, 2.2.4]
- Koden körs lokalt i telefonens minne vilket gör att appar ger en god prestanda och bra känsla för användaren. [litteratur + intervju, se kapitel 2.2.4, 6.1.1]
- Behöver inte vara uppkopplad på internet för att använda apparna (om inte apparna behöver hämta information exempelvis via en databas). [litteratur, se kapitel 2.2.4]

- Går att ta del av telefonens inbyggda funktioner som exempelvis kamera. [empiri, se kapitel 6.3]
- Det går att använda push-notiser för exempelvis en påminnelse eller att information om ett nytt meddelande anlänt till mailen. [intervju, se kapitel 6.1.1]

Nackdelar
- Plattformsberoende. [litteratur, se kapitel 2.2.4]
- Kräver utvecklare som kan ett eller flera specifika programmeringsspråk. [intervju + litteratur, se kapitel 2.2.4, 6.1.1]
- Licenskostnader tillkommer för att kunna utveckla på plattformen. [litteratur, se kapitel 2.2.3]
- Distribueringskostnader och väntetid vid godkännande av appar vid distribution. [litteratur, se kapitel 2.2.3]

6.1.2 Fördelar och nackdelar med Webbappar

Den andra app-typen som är vanlig idag har visat sig vara webbappen, dock är den inte är lika vanlig som nativeappen men det finns ändå en uppsjö av olika webbappar tillgängliga. I en av intervjuerna berättar en deltagare att en av anledningarna är att webbapparna inte är lika vanliga. Anledningen till det är för att HTML5 inte fick genomslag förens under 2012 och blev en utvecklingsstandard då. Det finns olika egenskaper för en webbapp, en del appar körs lokalt via en app som laddas ner till enheten medan en del appar körs direkt via webbläsaren. En stor fördel är att innehållet kan uppdateras externt utan att behöva gå in och ändra eller eventuellt ändra lite av innehållet i appen. Vid en intervju framkom det att det är relativt enkelt och billigt att göra en webbapp jämfört mot en native app. Det är lättare att hitta personer med kunskaper inom webbprogrammering än plattformsspecifika språk och speciellt om det rör sig om flera olika plattformsspråk. En stor fördel är också att en responsiv design kan göras vilket underlättar arbete då webbsidan blir anpassningsbar till olika enheter säger intervjukandidaten.

Fördelar
- Plattformsoberoende. [litteratur, se kapitel 2.2.4]
- Billigare utvecklingskostnader pga. programmeringsspråk (lättare att hitta personer med kunskap om exempelvis HTML och CSS). [intervju, se kapitel 6.1.2]
- Kan göra anpassningsbara med flera enheter som exempelvis tablets med hjälp av responsiv design. [intervju, se kapitel 6.1.2]
- Uppdateringar sker direkt genom att besöka webbplatsen. [litteratur + intervju, se kapitel 2.2.4, 6.1.2]
- Inga licenskostnader för att distribuera appen [litteratur, se kapitel 2.2.4]

Nackdelar
- Måste ha tillgång till en internetuppkoppling. [litteratur, se kapitel 2.2.4]
- Kan inte distribueras via telefonernas appbutiker. [litteratur, se kapitel 2.2.4]
- Kan inte använda telefonens hårdvarunära funktioner som exempelvis kameran och finns ingen möjlighet att använda push-notiser i telefonen. [litteratur, se kapitel 2.2.4]
- Måste marknadsföras via egna källor. [litteratur, se kapitel 2.2.4]

6.1.3 Fördelar och nackdelar med Hybridappar

När det kommer till blandningen mellan en webbapp och en nativeapp är resultatet svårt att få grepp om då det både finns anhängare och motståndare med appen. En anledning till det är att hybridappen är ny typ och att tekniken hela tiden förändras i en rask takt och att ramverk och API:er ständigt uppdateras. Vid en intervju med en person på företaget tror personen att denna typ av app kommer att vara framtiden. Dock tror personen att utvecklingen i både ramverk och mobiler måste utvecklas mer för att kännas ännu mer ”native” i användningen än vad den gör idag. Intervjupersonen nämner också att hybridappar främst kan vara bra för interna appar inom företaget. För att de ska kännas mer ”native” måste bl.a. minneshanteringen stärkas precis som Pehrson (2011) skriver. Det har dock framkommit att det finns besparingar att göra i både tid och pengar, då samma app ofta utvecklas i samma programmeringsspråk för alla plattformar.

Fördelar

- Plattformsoberoende. [litteratur, se kapitel 2.2.4]
- Ofta lätt att komma igång med att utveckla och använda hårdvarunära funktioner som exemplvis kamera via färdiga bibliotek. [empiri, se kapitel 6.3]
- Finns tillgänglig att ladda hem via appbutiker. [litteratur, se kapitel 2.2.4]
- Den webb-baserade delen av appen går att uppdatera externt vilket gör att själva appen inte behöver uppdateras i appbutiken. [litteratur, se kapitel 2.2.4]
- Bra för användning vid interna appar inom företaget. [intervju, se kapitel 6.1.3]

Nackdelar

- Tar mer resurser i form av minne vid processer. [litteratur, se kapitel 2.2.4]
- Beroende på vilka ramverk eller API:er som används är det inte säkert att all funktionalitet fungerar på alla enheter. [empiri, se kapitel 6.3]
- Många uppdateringar av ramverk som kan vara tidskrävande vid uppdatering. [empiri, se kapitel 7.2]

6.2 Metod för migration till Windows Phone 8

I kapitlet redovisas de migrationsmetoder som hittats genom att studera olika typer av dokument och webbsidor, se kapitel 5.2. Metoderna har också valts ut med hjälp från den teori som samlats in genom intervjuer och litteratur. Två stycken olika metoder för att migrera mellan plattformar till Windows Phone 8 har valts ut, en metod bygger på ett ramverk för hybridappar som heter PhoneGap och en metod bygger på ett ramverk för nativeappar som heter Xamarin.

6.2.1 Urval av migrationsmetoder

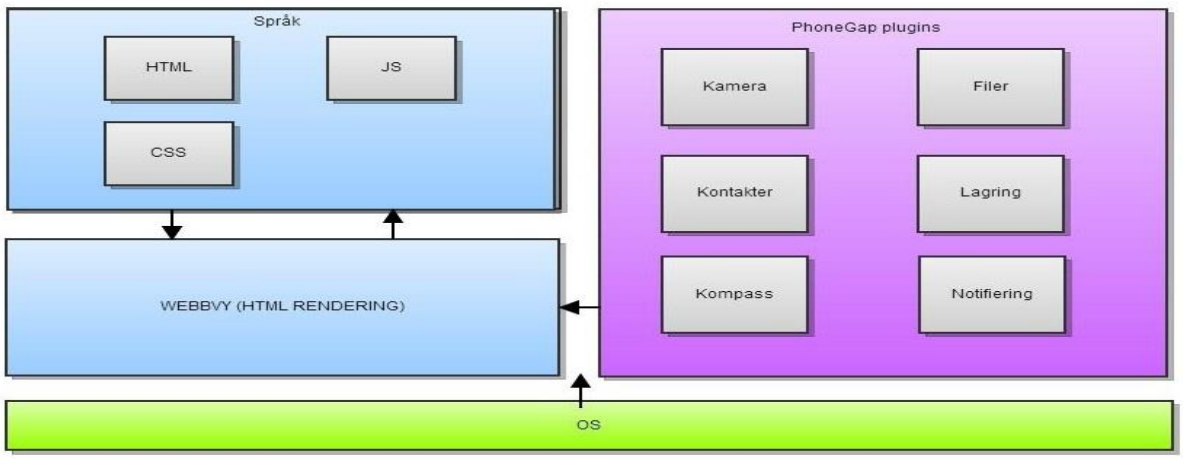
Vid urvalet av metoderna har jag främst valt dem av två stycken synpunkter. Det ena är att metoderna är två ledande metoder för migration idag och har en stor användarbas, vilket gör att det finns bra dokumentation om dem. De har också valts ut eftersom de ingår i varsin kategori av app-typ för att kunna ge en bredare spridning beroende på vilken funktionalitet användarna vill ha. I urvalet finns inte någon webbapp med tillhörande metod med, vilket beror på att dessa metoder ofta fokuserar på gränssnittsdesign istället för mer tekniska aspekter som exempelvis hårdvaruåtkomst vilket studien har mer fokus på. Vid urvalet har också egenskaper som efterfrågats av företaget gjorts vilket exempelvis är utvecklingstid och om kod på tidigare plattform behövs eller inte för att kunna migrera.

6.2.2 PhoneGap

Den första metoden som valts är PhoneGap vilket är ett öppet ramverk som använder sig av standardiserade API:er till olika plattformar för att kunna utnyttja plattformsberoende funktioner som exempelvis filhantering och kamera.

Ramverk: PhoneGap
Utgivare: Adobe System Inc
API: Apache Cordova
Version: 2.5.0
Plattformstöd för: IOS, Android, Windows Phone 7 & 8, Symbian, Bada och Blackberry
Programmeringsspråk: HTML 5, CSS och JavaScript
Hårdvarustöd till Windows Phone 8: Accelerometer (sensor), kamera, kompass, mobilkontakter, filhantering, geografisk lokalisering, media, nätverk, push-notifikation och databaskoppling
Licenskostnad: Gratis
Dokumentering: Bra dokumentering på hemsidan
Support: Finns möjlighet att köpa till support från $ 24.95 per månad

PhoneGap gör det möjligt att skapa hybridappar till flertalet olika mobila operativsystem som IOS, Android och Windows Phone 7 & 8 genom inbäddat innehåll med hjälp av befintliga webbteknologier. För att kunna göra en fullständig migration mellan en tidigare plattform till Windows Phone 8 måste först all kod från en plattform skrivas om till HTML 5, CSS och JavaScript. Om exempelvis kunden har en IOS app och vill flytta över den till flera operativsystem bl.a. Windows Phone 8 måste Objective-C kod manuellt skrivas om till HTML 5, CSS och JavaScript med stöd från PhoneGaps ramverk, vilket görs via IDE:t Visual Studio 2010 eller 2012 som PhoneGap har som standardverktyg för utveckling. PhoneGap har stöd för en emulator via Visual Studio som gör att ett gränssnitt av telefonen visas vilket möjliggör kompilering och testning av den app som utvecklas på plattformen. Då det finns mycket dokumentation i form av bla. installationsguide och API:s samt att det finns möjlighet till support, vilket gör att metoden upplevs som ett enkelt alternativ samt att det inte behöver kosta mycket resurser i form av personal, tid eller olika verktyg. Det som framförallt gör det som ett enkelt och billigt alternativ för migration är de webbaserade programmeringsspråken. Det behövs heller inte någon kod från tidigare plattform för att kunna migrera. Det som gör det möjligt för PhoneGap att fungera som ett multiplattformstöd är hur arkitekturen är uppbyggd, se Figur 6.



Figur 6 – Uppbyggnad av PhoneGaps arkitektur

Längst ner i arkitekturen finns plattformspecifika funktioner exempelvis hårdvarunära funktioner och data från IOS. Funktionerna finns samlade i den SDK som finns på plattformen. Ett steg upp i hierarkin finns plugins som är skapade med hjälp av PhoneGap som möjliggör användning av funktionerna från Windows Phone. Plattformsspecifika SDK och PhoneGaps plugin kommunicerar via ett OS API mellan de understa lagren i arkitekturen. När sedan ett plugin inkluderas i en app möjliggör det användning av exempelvis telefonens kamera och för att göra appens gränssnitt visuellt används en renderingsmotor. Denna renderingsmotor kommunicerar med både pluginen och det högre applikationslagret. För att göra kommunikationen möjlig används API:er som skickar och tar emot data mellan lagren. Att arkitekturen är gjord för API:er möjliggör inte bara körning på Windows Phone 8 utan på många andra plattformar och enheter som har stöd för inbäddat innehåll.

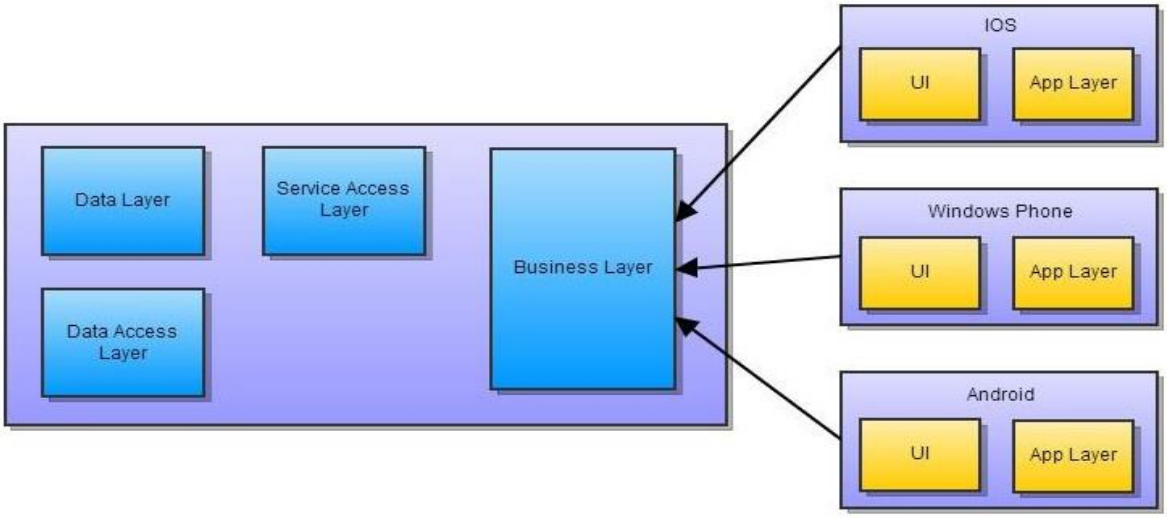
6.2.3 Xamarin

Det andra metoden som valts är Xamarin som bygger på öppen källkod som heter MONO vilket är baserat på .NET. Xamarin gör det möjligt att utveckla plattformsoberoende på olika typer av systemkärnor med hjälp av ett plattformsberoende ramverk och språk.

Ramverk: Xamarin
Utgivare: Xamarin Inc
API: Xamarin.Android / Xamarin.IO
Version: 2.0
Plattformstöd för: IOS, Android, Windows Phone 7 & 8
Programmeringsspråk: c#
Hårdvarustöd till Windows Phone 8: Fullt stöd för all hårdvara
Licenskostnad: Gratis (kostar för extra funktionalitet bl.a kunna använda Visual Studio)
Dokumentering: Bra dokumentering på hemsidan
Support: Finns möjlighet att köpa till support från $ 999 per år

Xamarin gör det möjligt att utveckla och köra appar på samma plattform. Xamarin bygger på .NET vilket gör att i ”runtime” körs c# kod i samband med plattformsspecifik struktur och med plattformens SDK. Oavsett vilket mobilt operativsystem appen kommer från kan appen byggas och köras. Det som produceras är i slutändan en native app som går att köras på IOS, Android och på Windows Phone 8. Om en kund har en app på en tidigare plattform exempelvis IOS måste kodbasen skrivas om till c# för att kunna ta del av Xamarin och där med kunna använda samma app på flera plattformar. Appar som kommer från IOS bygger på CocoaTouch SDK och kan utökas och använda .NET referenser genom Xamarin.IO. Appar som kommer från Android bygger på Google’s Android SDK och kan utökas och använda .NET referenser genom Xamarin.Android. När det gäller appar skrivna för Windows Phone används deras standard Windows Phone SDK. Alla plattformar kan använda .NET bibliotek och referenser vilket gör det möjligt att inkludera tillägg och olika insticksprogram som finns inkluderade i ramverket. För att kunna utveckla med hjälp av Xamarin behövs antingen Xamarin Studio eller Visual Studio som IDE. Xamarin Studio är främst för appar som är gjorde i Android, IOS eller om gratisalternativet av Xamarin väljs då det kostar extra för att få använda Visual Studio som utvecklingsverktyg. För utveckling i Windows Phone rekommenderas Visual Studio att användas.

Den arkitektur som används är en vanlig struktur som används flitigt i objektorienterade programmeringsprojekt, se Figur 7.



Figur 7 - Uppbyggnad av Xamarins arkitektur

I ett projekt som skapas för att flytta över befintlig struktur från en app är det rekommenderat att skapa ett delat kodbibliotek för alla olika plattformar. I det delade kodbiblioteket finns funktionalitet som fungerar på oberoende plattform. I biblioteket finns DL (Data Layer) som innehåller data som fysiskt laddas in till telefonen från exempelvis en databas eller en fil. DAL (Data Access Layer) vilket är ett API som innehåller metoder för att hämta, ändra eller ta bort information som kommer från DL. SAL (Service Access Layer) innehåller nätverkstjänster för exempelvis molnet i form av en webbservice. BL (Business Layer) vilket inkluderar all affärslogik i form av modeller som appen innehåller exempelvis kundhantering eller orders. BL är det överhängande lagret som sedan kommunicerar med den plattformsspecifika instansen. I den plattformsspecifika instansen för IOS, Android och Windows Phone finns två lager. Det ena är UI lagret där gränssnittet specificeras och applikationslagret som kommunicerar med BL.

6.3 Implementation för prototyp

I kapitlet beskrivs hur prototypen med de enskilda funktionerna som valts ut har implementerats med hjälp av varje migrationsmetod.

6.3.1 PhoneGap

I en hybridapp utvecklad med hjälp av PhoneGap är det lättast att följa arkitekturen för implementation. I Visual Studio startas ett nytt projekt med PhoneGap som bakomliggande ramverk. För att det ska finnas stöd för att kommunicera mot Windows Phone 8 SDK ska version 2.3.0 eller senare av API:et Apache Cordova användas. I implementationen har version 2.4.0 används för att testa om metoden fungerar praktiskt utifrån de tre val som gjorts till prototypen. Vad gäller implementationsmetoden har bottom-up metoden använts för att följa arkitekturen i varje funktion, vilket har gjorts genom att följa den från hårdvarunivå upp till gränssnittsnivå.

I ett PhoneGap projekt är det främst tre stycken mappar som används vid utveckling, vilket ligger under mappen ”/www” med undermapparna ”/js” och ”/css”. Det är här allt innehåll hamnar när en hybridapp byggs. Det finns även möjlighet att bygga med plattformspecifik

kod men det är inget som använts vid implementationen då det ligger utanför PhoneGaps ramverk.

För att kunna använda API:t måste det inkluderas i projektet, vilket görs genom att inkludera en JavaScript fil i exempelvis ”/js/index.js” och det är API:t som gör det möjligt till åtkomst till Windows Phone funktioner.

```
// ladda in Apache Cordova till appen
<script type="text/javascript" charset="utf-8" src="cordova-2.4.0.js"></script>
```

Kamera

För att kunna använda kamerans funktioner måste Apache Cordova få åtkomst till kameran. Anledningen till det är pga. originalkamerans mjukvara inte kan användas utan det är modifierad version som PhoneGap använder sig av. För att få tillstånd att använda den modifierade kameran behöver konfigurationen ändras i appens manifest. I manifestet finns bl.a. vilka tillåtelser och restriktioner över vilken hårdvara som appen får använda.
För att börja använda kameran laddas API:et in och ger åtkomst till kameran.

```
var picSource;   // input från kamera
var picDest;    // output från kamera

// ladda in Apache Cordova till enheten
document.addEventListener("deviceready", onDeviceReady, false);

function onDeviceReady() {
        picSource = navigator.camera.PictureSourceType;
        picDest = navigator.camera.DestinationType;
            }
```

Den input som kommer från kameran är bl.a. inställningar om hur stor fokusposition en bild ska ha eller hur bra kvalitén på bilden ska vara. Den output som ges är själva bilden och bl.a. kan bildformat och storlek på bilden väljas. För att kunna ta ett kort skapades en funktion för ändamålet.

```
// Ta ett nytt kort och ställa in kvalitet samt ha liten fokusposition på kameran
function takeNewPhoto() {
        navigator.camera.getPicture(onPhotoDataSuccess, onFail, {
          quality: 100,
          picDest: picDest.DATA_URL
        });
    }
```

När sedan emulatorn startas upp visade det sig att det går att ta exempelfoton och påvisar att kameran för PhoneGap fungerar.

Filhantering

Det behövs ingen åtgärd för att få åtkomst till filsystemet i telefonen utan det är inkluderat i Apache Cordovas struktur. I dokumentationen går det utläsa att det inte finns något stöd till Windows Phone 8 att varken ladda hem en PDF fil eller att öppna den. Efter testning visar det sig att det inte finns något stöd för att kunna ladda ner en fil ner till telefonens minne utan det går endast att visa filen om den finns tillgänglig på internet. Det samma gäller filer i PDF format, om filen placeras i telefonens minne finns det ingen möjlighet att öppna den

inbäddat inne i appen. Det går att öppna filen utanför appen om en ytterligare app laddas hem till telefonen men då faller själva poängen med appen som skapas för ändamålet.

Databasanslutning
Det har visat sig att det inte finns något stöd för att använda en SQL server som lokal databas i PhoneGap. SQL är den databas som i vanliga fall används i Windows Phone 8 för lagring av data. Dock har det visat sig att det går att använda en annan typ av lagring av data som är ”Local Storage”. Det är olikt en databas på det sättet att det inte finns ett frågespråk eller ett schema att följa vilket blir en nackdel om det data som finns blir en större mängd som exempelvis ska sorteras. För att använda ”Local Storage” behövs ingen åtgärd för åtkomst utan de finns inkluderat i Apache Cordovas struktur.

```
// Lägg till data i Local Storage
function AddToStorage() {
   window.localStorage.setItem('sports', 'soccer');
}

// Hämta och visa data i Local Storage
alert(localStorage.getItem('sports'));
```

6.3.2 Xamarin
När Xamarin används som metod för utveckling i Windows Phone 8 används lättast Visual Studio. I Visual Studio finns det redan inbyggt stöd för att kommunicera med Windows Phone 8 SDK, vilket görs genom att inkludera basbiblioteket ”`Microsoft.Phone;`” till app projekt som skapats. Det är fördelaktigt att följa arkitekturen med en delad kodbas när appar ska konstrueras med hjälp av Xamarin. Anledningen till det är pga. det blir lättare att länka till ”Xamarin.IO” som är resurser för IOS och ”Xamarin.Android” som är resurser för Android när utvecklingen i Visual Studio är klar och gjord för Windows Phone 8 appen. Vad gäller implementationsmetoden har bottom-up metoden använts för att följa arkitekturen i varje funktion, vilket har gjorts genom att följa den från hårdvarunivå upp till gränssnittsnivå.

Kamera
När en kamerafunktion ska användas i Windows Phone 8 används ett bibliotek som gör det möjligt att öppna kameran med ett bibliotek som heter ”`Microsoft.Phone.Tasks;`”
Sedan skrivs funktioner för att kunna initiera kamera åtkomsten och för att kunna prenumerera på en händelse som avgör om ett kort tagits eller inte.

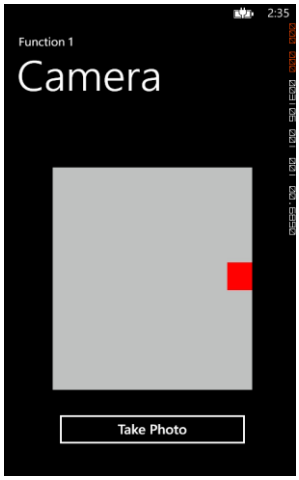
```
// Ett nytt fototagningsobjekt skapas och prenumenerar på ett event som tar fotot
_captureTask = new CameraCaptureTask();
_captureTask.Completed += CameraTaskCompleted;
```

Funktionen som läser in strömmen från kameran om ett kort har tagits eller inte.

```
// Event som avgör om vi tagit ett kort eller inte med kameran
private void CameraTaskCompleted(object sender, PhotoResult pr)
{
   if (pr.ChosenPhoto != null) // om vi har tagit ett kort
      {
        var imgLocal = new byte[(int)pr.ChosenPhoto.Length]; // kortet som tagits
        pr.ChosenPhoto.Read(imgLocal, 0, imgLocal.Length);
        pr.ChosenPhoto.Seek(0, System.IO.SeekOrigin.Begin);
```

```
        var bitmapImage = PictureDecoder.DecodeJpeg(pr.ChosenPhoto); // gör om till
bildformatet .jpg
        imgCaptured.Source = bitmapImage; // lagrad bild
       }
}
```

När ett kort tagits måste det bli renderat av appen. Då behövs ett bibliotek läggas till i applagret som heter ”Microsoft.Phone.Controls;” som sedan kan kommunicera med appens gränssnitt. När kommunikationen fungerar mellan lagren designas gränssnittet, se Figur 8.



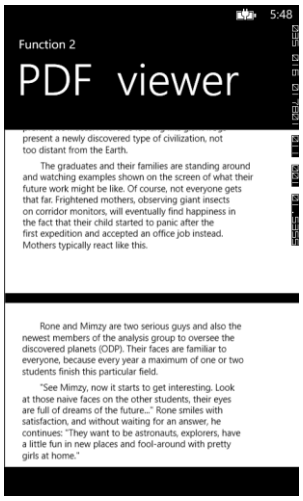
Figur 8 – Kamerafunktion utvecklad i Visual Studio

Filhantering

Det har visat sig att det inte finns något inbyggt stöd för att kunna öppna PDF inne i appen. För att kunna möjliggöra det har ett tillägg laddats hem som heter ”ComponentOne PDFviewer” vilket gör det möjligt att visa och läsa PDF filer internt i appen. För att kunna använda tillägget måste en referens till biblioteket inkluderas ”c1.phone.pdfviewer” och en grafisk komponent måste också inkluderas ”C1PdfViewer”.

När appen sedan startas har en funktion skapats för att läsa in en specifik PDF fil i form av en bok. PDF filen anpassas även här för visningen i gränssnittet genom att ställa in max längd och höjd. Efter filen lästs in kan den visas genom att använda den grafiska komponenten som inkluderades i projektet, se Figur 9.

```
private void PhoneApplicationPage_Loaded(object sender, RoutedEventArgs e)
{
   var pdf = Application.GetResourceStream(new Uri("/FilesAppWP8;component/ebook.pdf",
UriKind.Relative));
   pdf1.LoadDocument(pdf.Stream);
   pdf1.FontSize = 20;
   pdf1.Height = 550;
}
```



Figur 9 – PDFfunktion utvecklad i Visual Studio

Databasanslutning

Vad gäller databasanslutning har det visat sig att det finns stöd till Windows Phone 8. I resultatet har jag använt mig av en lokal databas tillsammans med ett verktyg som heter ”Entity Framework” som är ett mappningsverktyg i .NET för att kunna använda databaser på ett enkelt och effektivt sätt. Verktyget gör det enkelt att utföra transaktioner till och från databasen. Genom att använda mig av Xamarins arkitektur har jag skapat en databas (DL), ett transaktionslager (DAL) och ett affärslager (BL). En SQL server skapades i DL och för att kommunicera med databasen skapades DAL med hjälp av ”Entity Framework”

```
public class Repository : DataContext
    {
        public Repository(string connectionString): base(connectionString)
        {}

        // Transaktion till tabellen Phone
        public Table<BL.Customer> Phone
        {
            get
            {
                return GetTable<BL.Customer>();
            }
        }
    }
```

Genom att DAL kommunicerar med BL kan transaktioner göras mellan kontexterna.

```
public void AddPhone (Customer pnr)
        {
            using (var db = new Repository(ConnectionString))
            {
                db.Phone.InsertOnSubmit(pnr);
                db.SubmitChanges();
            }
        }
```

Slutligen startas applikationen upp och skapar databasen och läser in data som finns i DAL, se Figur 10.

```
            using (var db = new Repository(ConnectionString))
            {
                if (!db.DatabaseExists())
                    db.CreateDatabase();
```

```
            LoadData();
        }
```

Figur 10 – Databasfunktion utvecklad i Visual Studio

7 Analys

I kapitlet analyseras och återkopplas den teori som använts och den empiri som framkommit i studien och som lett till ett resultat. I analysen binds resultatet från de olika delarna i resultatet samman för att få en heltäckande analys jämfört om det skulle göras mot enskilda delar. Kapitlet börjar med att analysera fem viktiga kategorier vid val av app-typ och migrationsmetodernas implementation analyseras. Kapitlet avslutas sedan med att jämföra de två valda metoder som valts i studien.

7.1 Migrationsmetod baserat på app-typ

I resultatet har valet av migrationsmetod grundat sig i vilken app-typ som är lämplig att välja vid en eventuell migration mellan plattformar. Valet av vilken app-typ som är mest lämplig har genom teori visat sig bestå av flera kategorier, se tabell 4. I analysen har fem stycken kategorier valts ut som anses viktiga vid valet av app. Kategorierna har valts ut genom att analysera den teori och empiri som framkommit med studien.

Tabell 4 Kategorier med egenskaper för app-typer.

Kategori	Plus	Minus	Neutral
Prestanda	Native	Hybrid	Webb
Plattformsoberoende	Webb, Hybrid	Native	
Hårdvarufunktioner	Native	Webb	Hybrid
Utvecklingskostnad	Hybrid, Webb	Native	
Distribution	Native, Hybrid	Webb	

Prestanda

Prestandan har visat sig vara en viktig del vid valet av app-typ. Stark (2010) skriver att nativeappen har en god prestanda och att det finns möjlighet att skapa appar som är kraftfulla vid användning av dem. Det har också påvisats att en nativeapp ger mer känsla vid körning än exempelvis en hybridapp, vilket beror på att den har en sämre prestanda jämfört mot native. Anledningen till det är pga. den högre minneshanteringen vid körning av appen som använder mer resurser. En native app är alltså snabbare vid körning än någon annan app-typ.

Plattformsoberoende

Den viktigaste aspekten till varför en migration mellan plattformar är nödvändig är beroendet till plattformen. Nativeappen som är en av de vanligaste app-typerna idag är också källan till att appar måste migreras överhuvudtaget då det finns ett behov av att appen ska synas och användas på flera plattformar. Vad gäller hybrid och webbappar har det påvisats att de kan köras på en oberoende plattform på bekostnad av att all funktionalitet inte kan utföras som exempelvis filhantering.

Hårdvarunära funktioner

När det kommer till användning av hårdvaran av en enhet har det påvisats att native har det bästa stödet för åtkomst och användning av hårdvara. Stark (2010) skriver att appar som är

utvecklade som native finns möjligheten att använda fullt hårdvarustöd, vilket också är något som påvisats vara viktigt för användare som kräver lite mer funktionalitet av en app. Det har påvisats att det inte finns något hårdvarustöd i webbappar men att hybridappar har ett visst stöd vilket beror på om ramverket kan integreras med arkitekturen och få åtkomst till hårdvaran via ett SDK.

Utvecklingskostnad
Kostnader har påvisats vara en essentiell punkt när det kommer till vilken app-typ som väljs och om det går att hålla ner kostnader vid utvecklingen. Tillborg, Persson & Bentlöv (2012) skriver att en hybridapp håller ner kostnaden avsevärt då samma app kan utvecklas till flera plattformar med samma programmeringsspråk. Det har dock påvisats att hybridappar ofta är bättre för enkla ändamål som att visa enkla listor av data för interna företagsappar, vilket beror på den minskade prestandan när fler funktioner läggs till. Det har påvisats att nativeappen har höga kostnader vad gäller utveckling, vilket inte bara avser utvecklingskostnad i form av personal som kan olika programmeringsspråk utan även att olika verktyg och IDE behövs samt kunskap om dem.

Distribution
När en app ska nå slutanvändaren har det visat sig finnas vissa skillnader. Både native och hybridappen kan distribueras genom plattformens appbutik, vilket har påvisats att det ger en fördel då personer på ett enkelt sätt kan hitta nya och rekommenderade appar. Det har också visat sig att det går att betala för att synas genom vissa distribueringskanaler vilket gör appen ännu mer synlighet för användarna. Webbappar har inte den fördelen utan det har påvisats att webbappen måste användas via en URL menar Stark (2010). Webbappar måste också ha en egen marknadsföring vilket också påverkar kostanden i slutändan.

7.2 PhoneGap och Xamarin
De två migrationsmetoder som valts ut för studien har analyserats i hur bra dokumentation som funnits för vald metod och hur denna metod har fungerat att följa vid en implementation, se tabell 5.

Tabell 5 Dokumentation och implementation i de två valda metoderna.

Kategori	Plus	Minus
Dokumentation	PhoneGap, Xamarin	
Implementation	Xamarin	PhoneGap

Dokumentation
Det har visat sig finnas mycket bra dokumentation om båda metoderna. Det har visat sig att det finns en gedigen genomgång hur metoderna fungerar i både arkitektur och hur metoderna fungerar i användning. I PhoneGap har det visat sig att det funnits bra dokumentation om hur installationen sker för att komma igång med metoden. Det har även visat sig finnas en gedigen dokumentation om hur deras API kan användas för att implementera hårdvarunära funktioner. Xamarin har också påvisat god dokumentation hur enkelt det går att komma igång med installation och utveckling. Vid utveckling för Windows Phone 8 har det även visat sig att det går att använda deras egna SDKs dokumentation som

uppslagsverk. I båda metoderna finns det bra dokumentation om vilka verktyg som går att använda samt hur de fungerar ihop med metoderna. Vad gäller *distributionen* av appar som skapas med hjälp av metoderna har det visat sig enligt dokumentationen fungera att distribuera dem båda via app butiker som plattformen tillhandahåller.

Implementation
Det har visat sig att implementationen av funktionerna fungerat av varierande karaktär. Vad gäller Xamarin har dokumentationen fungerat bra att följa, både Xamarins och Windows Phone 8 SDKs dokumentation. Då Visual Studio används som IDE och Xamarin använder .NET som kodbas har det visat sig finnas mycket referensmaterial att följa. Det finns utöver den officiella dokumentationen en uppsjö av andra dokument och forum som hjälper utvecklaren vid implementering av kod. Genom att implementera funktionalitet som läser in PDF-filer har en extern modul använts vilket också påvisar att vissa tredje-parts tillägg fungerar väl, vilket underlättar olika typer av programmeringsproblem eftersom ”hjulet inte behöver uppfinnas igen”. Den funktionalitet som implementerats i prototypen med Xamarin har visat sig fungera väl vad gäller *hårdvarunära funktioner* då prototypen klarar att köras vid testning. Även *utvecklingskostnaden* kan hållas nere i form av personal då endast språk inom .NET behövs, dock tillkommer kostnader för både IDE, support och vissa tredje-parts tillägg. Xamarin har påvisat att metoden gör nativeappar *plattformsoberoende* genom att kunna köra samma kodbas till Windows Phone, Android och IOS.

Det har visat sig att implementationen av prototypen med hjälp av PhoneGap till Windows Phone 8 fungerat mindre bra. Det största problemet som identifierats är att det inte finns stöd för alla enheter och funktioner till Windows Phone 8 i dagens läge vilket gör att metoden inte fungerat fullt ut. Vad gäller den dokumentation och referensmaterial som finns på PhoneGaps hemsida har det visat sig att det fungerat väl att följa vid implementationen. PhoneGap använder också Visual Studio som primärt IDE vilket har visats sig fungera bra. Det har dock visat sig vara en krånglig process att uppdatera versionen av PhoneGap då en ny konfiguration alltid behövs göras, vilket är en tidskrävande och ineffektivt då PhoneGap ofta släpper nya uppdateringar. Vad gäller externa moduler exempelvis att läsa in PDF-filer i appen har inte funnits att tillgå i PhoneGap. Det finns externa plugins som kan användas till PhoneGap för denna typ av problem men idag finns inget stöd att använda dem till Windows Phone 8. Det har också visat sig att det inte går att använda de tredje-parts tillägg som exempelvis kunde användas i Xamarin. Den funktionalitet som gick att implementera i prototypen med PhoneGap har fungerat bra vad gäller *prestanda* då prototypen klarar att köras på ett satisfierat sätt vid testning. PhoneGap har också visat sig vara helt *plattformsoberoende* pga. av de webbspråk som används. Något som inte visat sig fungera bra är *hårdvarunära funktioner* då endast en av tre funktioner gick att implementera. Vad gäller *utvecklingskostnaden* går det att hålla ner kostnaderna pga. HTML, CSS och Javascript används, dock blir de låga utvecklingskostnaderna till bekostnad på vad för utbud som kan skapas i appen. Eftersom det idag inte finns fullt stöd till Windows Phone 8 blir metoden begränsad till vad som går att göra, exempelvis går det inte använda en SQL databas som är populär idag.

7.3 Val av metod för migration till Windows Phone 8

I denna avslutande del jämförs metoderna mot varandra och åsyftar till vilken metod som fungerat bäst för denna studie. Utifrån de implementationstester som gjorts har metoderna också jämförts baserat på utvecklingskostnader, utvecklingstid, personalkostnad, om koden ägs eller inte, support, dokumentation och prestanda.

Xamarin är ett gratisalternativ vilket gör att utvecklingskostnaderna kan hållas nere, dock tillkommer det kostnader när användningsskalan växer och när extra funktionalitet som exempelvis en PDF läsare behövs. Vad gäller utvecklingstiden behövs endast en kodbas skriven i .NET som kan användas för alla typer av plattformer. Personalkostnaden kan hållas nere genom att .NET är stort och det finns många utvecklare som har kompetens inom området och lämpliga för uppgiften. Det har påvisats att koden som ska migreras inte behöver ägas, bara projektet är väl definierat och utvecklarna vet vad som ska göras. Xamarin har visat sig ha bra support för en summa pengar och en väldigt god dokumentation. Xamarins prestanda visat sig vara den bättre eftersom att apparna körs native jämfört mot hybridlösningar. Något som också visat sig väl med Xamarin är möjligheten till användning av inbyggd .NET funktionalitet som exempelvis ”Entity Framework” för databashantering.

Genom att summera ihop och analysera helheten i vad som framkommit med studien har det visat sig att Xamarin är det bästa valet för migrationsmetod. Det som främst gör att PhoneGap inte visat sig vara det bättre alternativet är att det ännu inte finns fullt stöd för att använda Windows Phone 8 hårdvarunära funktioner samt andra olika tillägg. Xamarin är ett bra val då standardfunktionalitet ska och kan användas och det finns tillhörande bibliotek som är kompatibla med Xamarin. Det som är negativt med Xamarin är de extra kostnader som tillkommer vid större projekt där extra funktionalitet behövs.

8 Slutsats

I kapitlet sammanfattas hela studien i korthet och slutsatser görs. Kapitlet börjar med att först gå igenom studiens huvudfråga samt delfrågor. Frågorna besvaras åter igen samt hur de kunnat bli besvarade. Sedan sammanfattas vad företag inklusive Sogeti kan tänka på när de ska migrera mellan plattformar samt vilken app-typ som är lämplig att använda för olika ändamål. I slutet tas även vilken metod som fungerat bäst i denna studie. I kapitlet finns också ett diskussionskapitel där viktiga aspekter med studien diskuteras.

8.1 Redogörelse av slutsatser

I studien har denna huvudfråga varit i fokus:

- Vilken eller vilka metoder kan användas för att migrera befintliga appar till operativsystemet Windows Phone 8?

För att kunna besvara denna fråga har två stycken delfrågor behövt besvaras:

- Vilka för och nackdelar finns det med att använda olika app-typer?
- Vilken eller vilka metoder för migration mellan plattformar är möjlig att använda för att kunna implementera en enklare prototyp av en app?

Första delfrågeställningen har besvarats genom att genomföra två intervjuer som varit ostrukturerade med låg standardisering, vilket har kompletterats med en litteraturstudie om allmänna begrepp samt information om appar och systemmigration. Frågan har lett till insikt och svar i vilka för och nackdelar det finns med olika app-typer och i vilken kontext de kan användas i eller inte. Det har visat sig att följande app-typer är lämpliga att använda i förslagen kontext:

- *Native* kan användas när användaren kräver mycket av appen dvs. när hög prestanda behövs och när avancerade funktioner behöver användas.
- *Webb* kan användas när användaren inte kräver mycket av hårdvaran. Appen kan användas som exempelvis en plustjänst till en ordinarie hemsida eller en webbapplikation.
- *Hybrid* kan användas när användaren inte kräver mycket av appen dvs. när användaren behöver ha enkel listning och inte särskilt avancerade funktioner. Ett bra ändamål för denna typ är företag som vill synas på många plattformar eller använda som en intern företagsapp.

Andra delfrågeställningen har besvarats genom att använda en dokumentguide för att hitta olika migrationsmetoder och hur dem fungerar praktiskt. Dokumentguiden har fungerat som en mall vid implementationen av prototypen som fokuserat på tre olika funktioner som testar om metoden fungerat eller inte till Windows Phone 8. Frågan har gett svar till vilka migrationsmetoder som finns och lett till ett urval av två metoder som sedan testats genom implementation av en prototyp på tre olika funktioner. De två metoder som hittats och implementerats i prototypen är följande, se Tabell 6:

- *PhoneGap* En hybridapp som använder webbaserade språk och underlättar körning på flera plattformar som Windows Phone, IOS och Android. PhoneGap har visat sig fungerat mindre bra i avseende på stöd till Windows Phone 8.
- *Xamarin* En nativeapp som använder .NET och c# som språk och underlättar körning på flera plattformar som Windows Phone, IOS och Android. Xamarin har visat sig fungera bra på plattformen Windows Phone 8.

Tabell 6 Teori och implementation kopplat till utvalda metoder

Metod	Teori	Implementation
PhoneGap	Fungerar	Fungerar delvis
Xamarin	Fungerar	Fungerar

Slutligen har delfrågorna tillsammans bidragit till att huvudfrågan med studien kunnat besvaras vilket ger svaret att migrationsmetoderna PhoneGap och Xamarin finns för förfogande när det kommer till att migrera till Windows Phone 8.

De företag som vill använda studiens utvalda migrationsmetoder till Windows Phone 8 har ett par olika saker att tänka på vid en migration, de aspekter som hittats är:

- Kan metoden göra att vi kan spara pengar?
- Kan metoden göra att vi sparar tid?
- Kan metoden underlätta underhåll i form av uppdateringar?
- Är metoden lätt att följa, finns det bra dokumentation?

Den metod som resultatet påvisar som bäst att använda är Xamarin. Metoden har påvisat att det finns möjlighet att spara pengar genom att använda samma kodbas för samma app. Det gör också att tid går att spara då det finns många utvecklare som behärskar språket c# samt det är relativt enkelt att sätta sig in i. Genom att enbart uppdatera på ett ställe dvs. i det delade kodbiblioteket underlättar det underhållningen av appen samt sparar tid. Resultatet påvisar även att Xamarin har bra dokumentation som har visat sig vara enkelt att följa vid en implementation av en prototyp då det finns bra referensmaterial.

För intressenter som vill arbeta med resultatet om migration till Windows Phone 8 kan komma igång genom att införskaffa de rekommenderade verktyg som finns listade. Det är rekommenderat att använda sig av Xamarin som migrationsmetod eftersom resultaten påvisar att metoden fungerar bra till Windows Phone 8. Det är också rekommenderat att läsa igenom hela dokumentationen för Xamarin, se referenser. Implementationsresultatet kan ses som en introduktion för att komma igång med metoden samt de olika verktyg som behövs.

8.2 Diskussion

I kapitlet diskuteras hur studien har gått. Kapitlet inleds med en diskussion om litteratur och metodval samt implementation av prototypen. Kapitlet avslutas sedan med att ta upp etiska aspekter och vad som kan göras i framtiden både för företaget och för andra studenter som skulle vilja fortsätta där denna studie slutar.

Det viktigaste jag kommit fram till med studien är att det finns metoder som gör det möjligt att migrera mellan plattformar. Dock vill jag poängtera att det inte går att göra migrationer helt smärtfritt till Windows Phone 8 då kodbasen behövs skrivas om. Möjligheterna att använda webbaserade tekniker dvs. PhoneGap anser jag idag inte är helt möjligt då det inte finns tillräckligt stöd genom de plugins och andra tillägg som finns för att få appen att fungera på ett grundläggande och tillfredställande sätt till Windows Phone 8, vilket jag tror kommer att utvecklas de närmaste åren då HTML5 standarden kommer att utvecklas och växa mer och mer.

8.2.1 Litteratur
Den litteratur som valts ut för studien är förhållandevis ny. Vid valet av litteratur tyckte jag att det var viktigt att välja ut källor som inte var mer än några år gamla, vilket kunde stärka trovärdigheten för studien eftersom ämnet är nytt. Det finns enstaka källor som sträcker sig tillbaka innan 2000 talet men de flesta källorna är inte äldre än 2007. Då ämnet som studien handlar om är nytt är det viktigt med aktuell information för att kunna hitta svar inom ämnet och dess relaterade frågor. De böcker som valts till studien kommer främst från utgivare som har lång erfarenhet av teknologiska utgåvor som exempelvis O'Reilly och Apress. Andra källor som valts ut är främst företagens egna hemsidor och andra publikationer som granskats och godkänts av respektive högskola eller universitet. Jag tycker att litteraturen hjälpt till att besvara oklarheter som funnits när studiens frågor besvarats i resultatet.

8.2.2 Metodval
De metodval som valts ut för studien har fungerat bra att använda samt varit enkla att följa under arbetets gång. Då frågeställningen delats upp i underfrågor och att dessa två frågor besvarats med hjälp av olika metoder har visat sig vara en fördel, vilket också gjort det möjligt att besvara huvudfrågan med studien.

Genom att välja en kvalitativ metod med fokus på intervjuer och en litteraturstudie har en grund till problemområdet kunnat hittas. Kvalitativa intervjuer var ett naturligt val för mig då jag i början av studien behövde djupgående information kring problemområdet. Genom att ge intervjupersonerna ett stort utrymme för att berätta om domän, ämnesområde och personliga erfarenheter samt andra tankar gällande utveckling och närliggande områden som personerna delade med sig av har empiri införskaffats. Genom att använda låg standardisering och ostrukturerade intervjuer tror jag att mer detaljerad information framkommit än att exempelvis använda intervjuer på ett kvantitativt sätt genom exempelvis enkäter. Om jag skulle valt exempelvis enkäter tror jag att det skulle blivit problem med vilka frågor som jag skulle ställa, samtidigt som jag troligtvis skulle missat viktig information som jag skulle fått med på köpet i de ostrukturerade intervjuerna. Jag tycker även det har varit bra att intervjua personer ensamma och inte i grupp. Anledningen till det är för att få höra varje enskild individs tyckande. I gruppintervjuer kan det finnas risk att en eller flera personer ”kör över” de andra i gruppen och de då inte får säga lika mycket om sin egen del. Jag funderade på om gruppintervjuer skulle göras men sedan beslöt jag mig för att göra enskilda intervjuer.

Då det funnits lite teori om just migration till Windows Phone 8 fungerade även det induktiva angrepsättet bra då det gav ett komplement till intervjuer genom litteratur. Genom att jag hittade allmän information om vad migration mellan plattformar var och hur det fungerar i det stora hela kunde informationen användas som en grund när problemområdet skulle identifieras och preciseras. Jag hade även nytta av att det fanns uppsatser som tog upp andra typer av porteringar av kod även om det inte gällde Windows Phone 8 men ändå kunde användas vidare som teori.

Samtidigt som jag sökte efter källor som kunde bilda en allmän teori hamnade jag på hemsidor och forum som handlade Windows Phone 8. Genom att läsa dessa hemsidor hamnade jag tillslut på sidor som tog upp migrationsmetoder samt ramverk och hur det kunde gå till att migrera till och från Windows Phone 8. Sidorna bidrog till de deduktiva inslagen som gjorde det möjligt att testa dessa ramverk vid implementationen av prototypen.

Det som har varit svårast med arbetet är just att det funnits lite information överlag, många källor pratar om migration men ofta mellan Android till IOS eller tvärtom, vilket kan ses som naturligt då Windows Phone och i synnerhet version 8 inte är använt i lika stor skala som Android eller IOS. Jag har haft tur som haft personer på företaget med god generell kunskap om appar samt träffat på Windows Phone i utvecklingssammanhang.

8.2.3 Implementation

Den implementationsmetod som valts har fungerat bra tillsammans med de två migrationsmetoder som valts ut. I implementationen har IDE:et Visual Studio 2012 använts och .NET version 4 vilket har underlättat en del då jag har erfarenhet av det sedan tidigare. Det har också underlättat att både PhoneGap och Xamarin har haft bra dokumentation om hur de kan inkluderas och installeras i Visual Studio. Då det också funnits bra referensmaterial om hur arkitekturen fungerar kring de båda metoderna har också implementationsmetoden bottom-up fungerat väldigt bra. Jag tror att om det inte funnits material kring hur arkitekturen fungerade hade det tagit längre tid med implementationen då jag hade varit tvungen att testa mig fram till en lösning vad gäller hur prototypen skulle struktureras. Vad gäller hur funktionerna som implementerats med hjälp av ramverken har det inte tagit längre tid då det är kod som kommer från välkända programmeringsspråk som c# och märkspråk som XAML och HTML.

8.2.4 Etiska aspekter

I studien har etiska aspekter tagits hänsyn till. I studien finns det inga speciella etiska ställningstaganden som kan göra att individer eller företag kan komma till skada. Jag har också tagit hänsyn till anonymitet i studien vilket innebär att intervjukandidaternas namn inte är publicerade. Anledningen till det är för att intervjukandidaterna inte ville intervjuas annars. Intervjukandidaterna har också fått reda på syfte med studien samt erhålla hela studiens material inklusive intervjusammanfattningar för granskning. Det har även gjorts en kortare presentation av studien för inblandade personer i studien. Till sist har jag fått ett godkänt till att studien får redovisas och publiceras med företagets namn.

8.2.5 Framtida arbete

Det framtida arbetet skulle kunna delas upp i två stycken olika delar, dels för hur Sogeti skulle kunna arbeta vidare med materialet i studien eller vad andra studenter skulle kunna fortsätta med i avseende på denna studie.

Det som Sogeti skulle kunna fortsätta med är förslagsvis att arbeta vidare med Xamarin och testa det i större projekt där både en IOS, Android och Windows Phone app använder Xamarins kodbas. Eftersom studien fokuserade på hur lösningen blev på Windows Phone 8 skulle det också vara intressant och se hur exempelvis prestandan blir på en Android mobil. Det skulle vara intressant att se hur lättare material som appar med inhämtning av data via en URL och filer skulle bete sig men också tyngre material som strömmande video eller kamerahantering.

Det som andra studenter skulle kunna arbeta vidare med i framtiden gällande denna studie är ett par saker. Det skulle vara intressant att följa HTML5 utveckling och i synnerhet om PhoneGap eller någon annan lösning som fokuserar på språket får ett bra stöd till Windows Phone 8 eller om det i framtiden finns en senare modell. Det skulle också vara intressant att göra jämförelser med andra metoder som ”Codenameone” som är en hybridlösning som också fokuserar på nativeappar med java som kodbas eller Rhodes som är en annan hybridlösning.

Denna studie har fokuserat på Windows Phone 8 men det finns andra plattformar som skulle kunna undersökas exempelvis det mobila operativsystem Blackberry som används relativt flitigt i USA. Det skulle vara intressant om det fanns metoder som klarade att hantera alla dagens operativsystem i telefonen. Ett annat stort område är tablets, vilket skulle kunna undersökas om det finns metoder att migrera mellan exempelvis iPad till Windows 8 tablets.

Referenser

Andersson, P. (2010). *Implementation of website for cognitive behavioural therapy using the development framework Symfony*. (Studentuppsats, kandidatexamen). Linköpings universitet.

Android (2013). *Code and documentation from Android VM team - Google Project Hosting*. Tillgänglig på Internet: https://code.google.com/p/dalvik [Hämtad 2013-05-31].

Alasdair, A. (2012). *Learning iOS Programming, 2nd Edition*. O'Reilly Media, Inc., Sebastopol, CA.

Apple Inc (2013). *Apple Developer*. Tillgänglig på Internet: https://developer.apple.com [Hämtad 2013-02-22].

Dey, J., & Sarma, N (2007). Data Migration Validation. *Drug Discovery & Development magazine:* Vol. 10, No.2, February, 2007, pp. 28-31.

The Eclipse Foundation (2013). *Eclipse - The Eclipse Foundation open source community website*. Tillgänglig på Internet: http://www.eclipse.org [Hämtad 2013-05-31].

Gartner (2012). *Gartner Says Worldwide Sales of Mobile Phones Declined 2.3 Percent in Second Quarter of 2012*. Tillgänglig på Internet: http://www.gartner.com/newsroom/id/2120015 [Hämtad 2013-02-20].

Gartner (2012). *Gartner Says Worldwide Sales of Mobile Phones Declined 3 Percent in Third Quarter of 2012; Smartphone Sales Increased 47 Percent*. Tillgänglig på Internet: http://www.gartner.com/newsroom/id/2237315 [Hämtad 2013-02-20].

Google (2013). *Android Developers*. Tillgänglig på Internet: http://developer.android.com [Hämtad 2013-02-22].

Jacobson, D., Brail, G. & Woods, D. (2011). *APIs: A Strategy Guide (first edition)*. O'Reilly Media, Inc., Sebastopol, CA.

Henry., L & Eugene, C (2012). *Beginning Windows Phone App Development*. *Apress*, US.

Microsoft (2013). *Windows Phone development*. Tillgänglig på Internet: http://msdn.microsoft.com/en-us/library/windowsphone/develop/ff402535(v=vs.105).aspx [Hämtad 2013-02-22].

Mobilmedia (2013). *Vad är skillnaden mellan en native-app och en webb-app?* Tillgänglig på Internet: http://www.mobilmedia.se/docs/mobilmedia_appochwebbapp.pdf [Hämtad 2013-02-15].

Pehrson, T. (2011). *Utvärdering av multiplattformsutvecklingsverktyg för smarta mobiler*. (Studentuppsats, kandidatexamen). Högskolan i Gävle.

Pressman, R. (2010). *Software Engineering – A Practitioners´s Approach*. New York, USA: The McGraw-Hill Companies, Inc.

Rahimian, V., & Habibi, J. (2008). On Performance of Mobile Systems: Challenges of Software Design and Evaluation. Presenterat vid *CCE 2008. 5th International Conference*, Mexico City 12-14 November, 2008.

Riehle (2000). Framework Design A Role Modeling Approach. Ph.D. Thesis, No. 13509. Zürich, Switzerland, ETH Zürich, 2000.

Sogeti (2013). *Sogeti - Lokala IT-lösningar på global grund*. Tillgänglig på Internet: http://www.sogeti.se [Hämtad 2013-02-20].

Stark, J. (2010). *Building Android Apps with HTML, CSS and JavaScript (first editon)*. O'Reilly Media, Inc., Sebastopol, CA.

Statistiska Centralbyrån. (2013). *Privatpersoners användning av datorer och internet 2012*. Tillgänglig på Internet: http://www.scb.se/statistik/_publikationer/LE0108_2012A01_BR_IT01BR1301.pdf [Hämtad 2013-02-12].

Ståhl, N. (2012). *Portering av programvara – metodik och fallstudie*. (Studentuppsats, kandidatexamen). KTH.

Tillborg, C., & Persson, L. & Bentlöv, H-E. (2012). *Utveckling av hybrid mobilapplikation för flera plattformar*. (Studentuppsats, kandidatexamen). Linnéuniversitetet.

Westfall, J., Augusto, R. & Allen, G. (2012). *Beginning Android Web Apps Development*. Apress s. 54-57.

Winter, J. (1992). *Problemformulering, undersökning och rapport*. Malmö: Almqvist & Wiksell.

Xamarin Inc (2013). *Part 1 - Understanding the Xamarin Mobile Platform*. Tillgänglig på Internet:http://docs.xamarin.com/guides/crossplatform/application_fundamentals/building_cross_platform_applications/part_1_-_understanding_the_xamarin_mobile_platform [Hämtad 2013-02-21].

Appendix A - Begreppslista

- API (Application Programming Interface). Regeluppsättning för kommunikation och integration mellan applikationer. Sker via funktionsanrop.
- Appbutik. En butik på internet via telefonen där användare kan köpa eller gratis ladda hem appar för telefonen.
- CSS (Cascading Style Sheet). Ett webbbaserat språk som används för presentation av text i form av exempelvis typstil eller färg.
- Emulator. Virtuellt mobilgränssnitt som visar hur appen skulle se ut på en verklig mobil.
- Enhet. Kan vara exempelvis en smart telefon eller en surfplatta, har inget med plattformen att göra.
- HTML (HyperText Markup Language). Ett webbbaserat märkspråk som används för att strukturera text.
- IDE (Integrated Development Environment) Ett del av en utvecklingsmiljö exempelvis till Android eller Windows Phone.
- Javascript. Ett skriptspråk som dynamiskt kan förändra en webbsida med olika funktioner.
- JRE (Java Runtime Environment). Ett programpaket för utveckling till Android enheter.
- Metro. En livebaserat teknik som visar och uppdaterar appar live på skrivbordet. Exempelvis en mejl-app som uppdaterar ikonen om ett nytt meddelande inkommer.
- Operativsystem. En länk mellan maskinvaran och programmen som gör det möjligt att använda programvaran på datorn. En plattform har ett eller flera OS.
- Plattform. En struktur för hur data och funktioner behandlas med hjälp av en speciell maskinvara. Är en teknisk förutsättning om ex, ett program fungerar på maskinen.
- Plugin. Ett slags program som körs i programmet för att underlätta en specifik uppgift exempelvis kunna spela upp ljud i ett visst program.
- SDK (Software Development Kit). Utvecklingsverktyg som gör det möjligt för utvecklaren att bygga applikationer mot en exempelvis en specifik del, program, operativsystem eller ramverk.
- Tablets (Surfplatta). En enhet likt en blandning på en smarttelefon och en dator vars syfte är att utföra enklare uppgifter som exempelvis surfa på internet eller kolla email.
- XAML (Extensible Avalon Markup Language). Ett programmeringspråk inom .net som används för att definiera gränssnitt och dess händelser.