Validerings- och editeringsverktyg för SMS-massutskick

Matias Westin

Examensarbete för ingenjörsexamen (YH)

Utbildningsprogrammet för informationsteknik

Vasa 2015

EXAMENSARBETE

Författare: Matias Westin

Utbildningsprogram: Informationsteknik, Vasa

Handledare: Mikael Jakas & Kaj Wikman

Titel: *Validerings- och editeringsverktyg för SMS-massutskick*

Datum: 02.06.2015 Sidantal: 39 Bilagor: 2

Abstrakt

Detta examensarbete utfördes åt Arena Interactive AB. Examensarbetet behandlar utvecklingen av ett listhanteringsverktyg för strukturering och validering av CSV-filer med listor som innehåller textmeddelanden och tillhörande information om sändaren för massutskick. Företaget använder sig av applikationer som kräver specifika liststrukturer och formatstandarder på dessa filer. Verktyget kommer att automatisera steg som förut gjordes för det mesta för hand vid strukturering av CSV-filerna.

Resultatet blev ett fungerande verktyg som utvecklades i Microsoft .NET-miljön och utvecklingen utfördes i Visual Studio 2010 med programmeringsspråket C#. Ramverket som valdes är WPF och verktyget utvecklades med MVVM-struktur.

Språk: svenska Nyckelord: C#, WPF, MVVM

BACHELOR'S THESIS

Author:	Matias Westin
Degree Programme:	Information Technology, Vaasa
Supervisors:	Mikael Jakas & Kaj Wikman

Title: *Validation and Editing Tool for SMS Mass Mailing*

Date: 02.06.2015 Number of pages: 39 Appendices: 2

Summary

This thesis was done at the request of Arena Interactive AB. The goal of the thesis is the development of a list-management tool for structuring and validating CSV files with lists containing text messages and related information about the sender, for easy mass mailing. The company uses applications that require specific list structures and format standards for these files. The tool's job is to automate steps that were for the most part done manually when structuring CSV files in the past.

The result was a viable tool developed in the Microsoft .NET environment. The development was done in Visual Studio 2010 with the programming language C#. The chosen framework is WPF, and the tool was developed with the MVVM structure.

Language: Swedish Key words: C#, WPF, MVVM

OPINNÄYTETYÖ

Tekijä: Matias Westin

Koulutusohjelma: Tietotekniikka, Vaasa

Ohjaajat: Mikael Jakas & Kaj Wikman

Nimike: *Validointi- ja editointityökalu SMS-massapostitukseen*

Päivämäärä: 02.06.2015 Sivumäärä: 39 Liitteet: 2

Tiivistelmä

Tämä opinnäytetyö tehtiin Arena Interactive AB:lle. Opinnäytetyö käsittelee luettelotyökalun kehitystä. Työkalua käytetään CSV-tiedostojen rakentamiseen ja validointiin. Näissä tiedostoissa on luetteloita, jotka sisältävät tekstiviestejä ja niihin liittyviä tietoja lähettäjästä massapostitusta varten. Yritys käyttää sovelluksia, jotka vaativat erityisiä luettelorakenteita ja formaattistandardeja näissä tiedostoissa. Työkalu automatisoi vaiheita jotka aiemmin tehtiin manuaalisesti kun CSV-tiedosto rakennettiin.

Tuloksena on toimiva työkalu joka on kehitetty Microsoft .NET -ympäristössä ja kehitys tapahtui Visual Studio 2010 -ohjelmassa C#-ohjelmointikielen avulla. Sovelluskehykseksi valittiin WPF ja työkalu kehitettiin MVVM-rakenteen avulla.

Kieli: ruotsi Avainsanat: C#, WPF, MVVM

Innehållsförteckning

ORDFÖRKLARINGAR

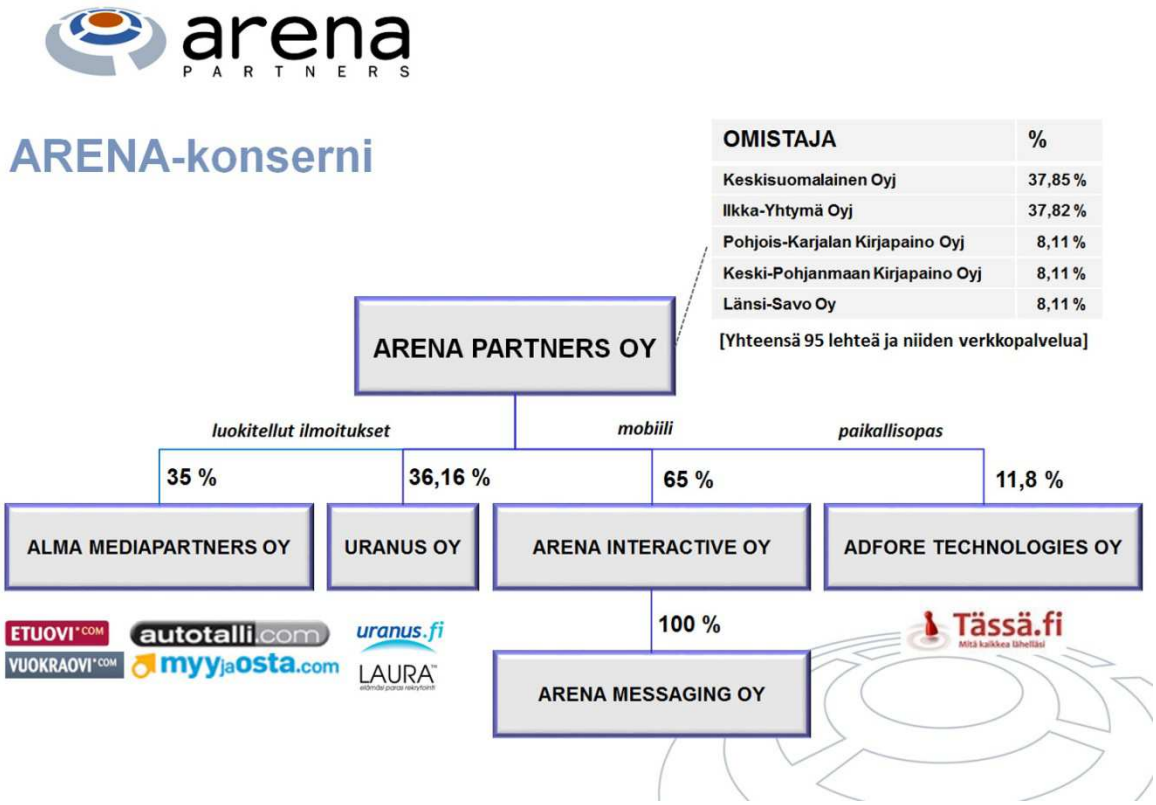
Blocklist	Är en lista med nummer som man inte vill inkludera i utskickslistan, med denna funktion tar bort dessa nummer från utskickslistan.
Datagrid	Ett rutnät med kolumner och rader, där varje ruta kan ha unik data.
Delimiter	Delimiter är det tecken som används för att separera kolumner i t.ex. CSV-filer. Ett exempel på hur texten kan se ut där semikolon är delimiter: data1;data2;data3.
Checksum	Checksum används för att söka fel vid inmatning av data, den använder i förhand specificerade algoritmer för att räkna ut den totala summan av bytes som ingår i filen.
CSV	Comma-separated value. CSV-filer är vanliga textfiler som innehåller data i ett format som är strukturerad för tabeller.
ITU	International Telecommunication Union är Förenta Nationernas fackorgan för informations- och kommunikationsteknik.
Regex	Regular Expression (Reguljära uttryck) är en sekvens av tecken som bildar ett sökmönster, som används för att lätt kunna hitta och ersätta olika tecken.
SMS peering	SMS peering gör det möjligt för mobiloperatörer att skicka data direkt sinsemellan, för minska behovet av dyra SMS-transport tjänster.
Käll-filen	I detta fall så betyder det den ursprungliga CSV-filen som användaren valde för att valideras.
UDH	User Data Header är en binär struktur som kan finnas i början av ett SMS-meddelande. UDH innehåller inte någon text, utan istället så specificerar den hur ett meddelande skall formateras och proccesseras.

1 Uppdragsgivare och uppgift

Uppdragsgivare för detta examensarbete var Arena Interactive, som gav i uppdrag att utveckla ett nytt listhanteringsverktyg. Detta verktyg skall innehålla funktioner som två av deras föregående listhanteringsverktyg innehåller och vidareutveckla dessa funktioner, samt att implementera nya.

1.1 Arena Interactive

Arena Interactive är ett företag som utvecklar och upprätthåller mobiltjänster. Företaget har över 250 kunder och de kommer mestadels från Finland, men en del också från resten av skandinavien. Till de främsta kunderna hör tidningskoncernerna. Arena Interactive hör till Arena Partners, andra delar som hör till Arena Partners är Alma Mediapartners, Uranus och Adfore Technologies (se figur 1).



Figur 1. Förteckning vem som äger mest av Arena Partners [1].

Arena Partners ägs av olika tidningar, Keskisuomalainen, Ilkka-Yhtämä, Pohjois-Karjalan Kirjapaino, Keski-Pohjanmaan Kirjapaino, Länsi-Savo m.fl. Till övriga icke-ägarkunder hör Iltalehti, TallinkSilja, Verkkokauppa.com, Vattenfall, Sanoma Games, WWF Suomi, Stadium osv. Flera exempel kan ses i figur 2.[2]

Figur 2. Några kunder till Arena Interactive [3].

Arena Interactive grundades i början på 2000-talet som ett utvecklingsbolag, som på den tiden hette Arena Partners. Tidningshusen i Finland konstaterade att de hade för lite resurser för att enskilt börja utveckla någon webb- och likadana tjänster. Inom Arena Partners så fanns det en avdelning som hette AP-mobile. Då utvecklade de SMS- och MMS-tjänster, sålde ringsignaler och operatörslogon och dylikt som var populärt på den tiden. De hade även WAP-tjänster, så man kunde surfa och betala per artikel, men idag är det inte mera aktuellt. År 2007 konstaterade de att det fanns potential att sälja tjänster åt andra också än bara ägarkunderna. Så då blev Arena Interactive i sin nuvarande form grundat 1 april 2007, som började marknadsföra sina tjänster inte bara i Finland utan även i Sverige och andra nordiska länder. (Intervju med CTO Patrik Norrgård på Arena Interactive Ab Oy 02.10.2014)

År 2010 köpte Arena Interactive upp en konkurrent från Helsingfors vid namn Steam Communications. Steam Communcations kunder bestod av bl.a. Verkkokauppa.com, YLE som i sin tur Arena Iteractive har värvat tack vare uppköpet. I och med detta köp så finns

de nu på tre kontor i Finland. Produktutveckling sker i Vasa, försäljning och marknadsföring i Jyväskylä, administration, försäljning och support i Helsingfors. Nuvarande VD:n heter Timo Wallius och jobbar i kontoret i Helsingfors.

Övriga sektorer som Arena Interactive försöker slå sig in på är tjänster till företag inom eldistribution, så fokus ligger inte enbart inom tidningssektorn.

Arena Interactive erbjuder tjänster för kund- och marknadsmeddelanden över SMS- och MMS som t.ex. för gruppmeddelanden, tävlingar, Ärrä-koder. Andra tjänster som erbjuds är mobilsajter och mobilapplikationer för organisationer som vill marknadsföra via mobilen. Även mobilbetalning och medellandeförmedling, elektronsika biljetter och kuponger är tjänster som erbjuds. Arena Interactive har också marknadsinriktade SMS-integrationer. De har inriktade SMS-integrationer för mediabranschen, hotell och restaurang, logistik. [2]

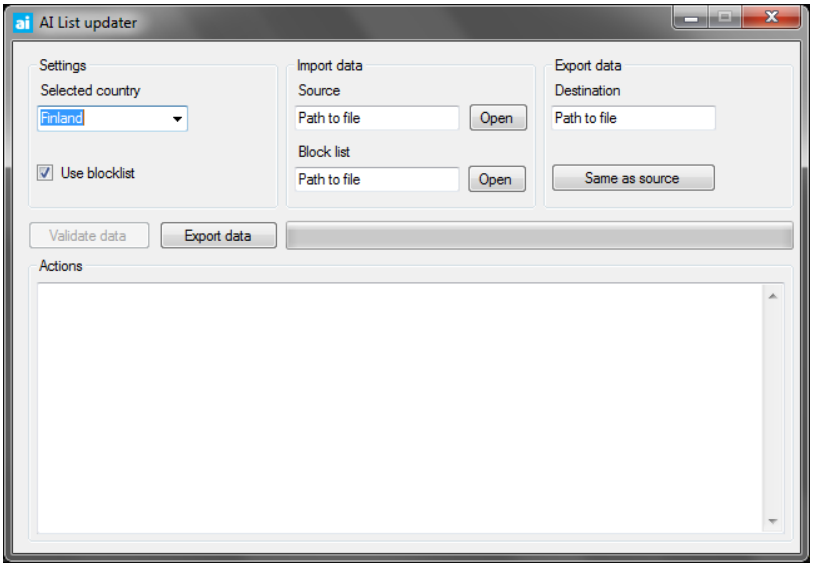
1.2 Beskrivning av uppgiften

Uppgiften var att utveckla ett listhanteringsverktyg för strukturering och validering av CSV-filer med listor som innehåller textmeddelanden och tillhörande information om sändaren för massutskick. Företagets SMS-orienterade tjänster Send och MobiSend kräver specifika liststrukturer och standard format för att datan skall kunna uppladdas och processeras. Källfilens mest relevanta data är mobiltelefonnumren. Telefonnumren är länken till mottagare av olika former av SMS-marknadsföring och marknadskommunikation.

Verktyget kommer att automatisera steg som förut gjordes för det mesta för hand vid strukturering av CSV-filerna.

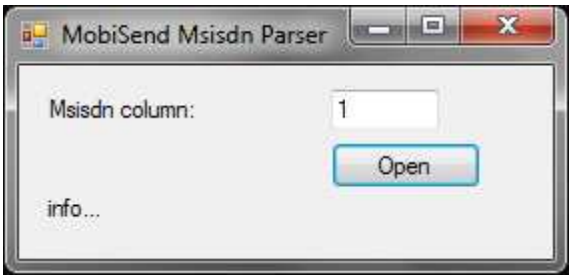
1.3 Existerande lösning

Före detta verktyg användes två separata listhanteringsverktyg för att förarbeta och strukturera CSV-filerna. AIListUpdater *(ListUpdater)* (se figur 3) och MobiSendParser (se figur 4) är båda verktyg för internt bruk.



Figur 3. Bild på AIListUpdater.

Med AIListUpdater är det möjligt att validera en text- eller CSV-fil med en kolumn. Med detta program är det även möjligt att välja bland flera länders telefonnumersregler och så finns även funktionen att ha en ”blocklist”, som är en lista med nummer som man inte vill inkludera i utskickslistan, med denna funktion så tas då dessa nummer bort från utskickslistan.



Figur 4. Bild på MobiSendParser.

MobiSendParser finns till för att hantera mera avancerade listor med mer specifik kundinformation. Med MobiSendParser är det möjligt att välja en kolumn i taget och

validera finska telefonnummer. Men om flera kolumner behöver valideras, så blir det mycket arbete i Excel för att resultatdatan måste manuellt bytas ut i varje icke validerad kolumn i orginalfilen mot de validerade kolumnerna som exporteras till en ny fil.

2 Mobila nät

I detta kapitel tas det upp om mobiltelefonisystemet GSM och lite om dess historia, samt information om hur ett mobiltelefonnummer är uppbyggt.

2.1 GSM

Den första generationens (1G) analoga mobiltelefonisystem NMT (Nordic Mobile Telephony), byttes ut år 1991 ut i norden till den andra generationens (2G) digitala mobiltelefonisystem som heter GSM (Global System for Mobile Communications). Nuförtiden så används GSM-tekniken i ett drygt hundratal länder. Till att börja med så användes GSM enbart för att talkommunikation. Som Jörg Eberspächer, Christian Bettstetter och Hans-Joerg Vögel säger i sin bok: *" In the beginning, GSM was used almost exclusively for speech communication; however, the Short Message Service (SMS) soom became extremely popular with GSM users: several billion text messages are being exchanged between mobile users each month."*. Tack vare GSM så är det nu möjligt att skicka SMS (Short Message Service) och möjligheten att ha flera telefonsamtal än med NMT. Med GSM så introducerades SIM-kort (Subscriber Identity Module). Detta kort innehåller information om telefonnummer, vilka tjänster som ingår i abonnemanget och innehåller teknik för att kryptera samtal. Senare så implementerades flera tjänster till GSM som MMS (Multimedia Messaging Service), WAP-sidor (Wireless Application Protocol), GPRS (General Packet Radio Service) och EDGE (Enhanced Data Rates for GSM Evolution) (2.5G). Idag utökas GSM-nätet med ett parallellt 3G- och 4G-nätverk, med vilka möjliggör snabbare internetuppkoppling i tätorter.

För överföring av konversationer och data så finns det fyra olika GSM frekvenser, GSM-850, GSM-900, GSM-1800 och GSM-1900, alla dessa värden är i megahertz. USA

använder sig av GSM-850 och GSM-1900 och Europa använder GSM-900 och GSM-1800.

Det finns även en annan mobilnätstandard som inte fungerar på samma sätt som GSM, men används för samma ändamål. Standarden CDMA (Code division multiple access), används av en stor del av mobiloperatörerna i USA (t.ex. Sprint, Verizon och U.S. Cellular). En skillnad mellan GSM och CDMA är vid de tillfällen när man ringer upp någon. Med GSM så kan man både ringa och använda sig utav datapaket samtidigt, men med CDMA så kan du endast använda en i taget.[4]

2.2 Mobilnummer

Ett telefonnummer är en sekvens siffror som är upp till 15 siffror långt utan att räkna med prefixet. Ett telefonnummer används för att kunna identifiera en abonnent både inom samma land som operatören och utomlands. Hur numret skall se ut bestäms av flera olika nummerplaner och E.164 standarden. E.164 är en ITU-T (The International public telecommunication numbering plan) rekommendation som definierar en numreringsplan för PSTN (public switched telephone network). Telefonnumret innehåller internationellt prefix, landsnummer, nationell destinationskod samt abonnentnummer. Regler för hur ett telefonnummer skall se ut varierar beroende på från vilket land det kommer (se bilaga1). [5]

2.2.1 Internationellt prefix

En siffersekvens som kommer före landsnumret för ett utlandssamtal. För alla länder inom EU så är utlandsprefixet 00. Det går även att använda + tecknet på mobiltelefoner som utlandsprefix. [6]

I Finland är utlandsprefixet 00, men det är möjligt att välja operatör genom att använda operatörens specifika nummer (se figur 5).

990	TeliaSonera
991	Elisa
992	Globetel
994	TDC
99511	Helistaminne
99532	Elisa
99533	TDC

99541	DNA
99555	DNA
99559	Elisa
99566	Datalahti
99577	TDC
99588	Globetel

99590	TeliaSonera
99599	Nettia
996	DNA
9977	Elisa
998	Orange Business
999	Elisa

Figur 5. Lista på finska mobilnäts riktnummer [7]

2.2.2 Landsnummer

Landsnummer är det nummer som kommer efter utlandsprefixet vid telefonering till utlandet. Några länder har bara en siffra i sitt landsnummer, t.ex. USA, Kanada, Ryssland och Kazakstan. Alla landsnummer är inte till för olika länder, utan t.ex. 800 är för internationella gratisnummer dvs. mottagaren betalar. [8]

Exempel på hur ett landsnummer kan se ut kan ses i listan under (se figur 6).

Finland	358	**Tyskland**	49
Sverige	46	**Litauen**	370
Norge	47	**Lettland**	371
Estland	372	**Ryssland**	7

Figur 6. Lista på landsnummer [9]

2.2.3 Nationell destinationskod

Nationell destinationskod, även kallad riktnummer, är en siffersekvens som används för att identifiera geografiska och icke-geografiska samtalsområden för telefoni. Riktnumret är oftast sammanslaget med det nationella prefixet, som vanligen är 0, och nationell destinationskod. Förut så användes den inledande nollan för att koppla samtal till den manuella växeln. Riktnumret behöver inte användas vid samtal inom samma geografiska riktnummerområde. [10]

En lista på Finlands geografiska areakoder (se figur 7).

Norra Karelen	013	**Åbo och Björneborg**	02
Mellersta Finland	014	**Tavastland**	03
S:t Michel	015	**Kymmene**	05
Lappland	016	**Vasa**	06
Kuopio	017	**Uleåborg**	08
Åland	018	**Helsingfors**	09
Nyland	019		

Figur 7. Lista på Finlands geografiska areakoder.[11]

2.2.4 Mobil-prefix

Mobil-prefix är de nummer som kommer före abonnentnumret. Förut i Finland så visade de tre första siffrorna i ett nummer från vilken operatör numret var införskaffat (t.ex. 040 TeliaSonera, 050 Elisa (förut Randiolinja), 041 och 044 DNA, 045 Saunalahti). Nuförtiden är det möjligt att byta operatör utan att byta telefonnummer. [12]

Några exempel på mobil-prefix (se figur 8) är alla nummer mellan 040 och 049 samt 050 för Finland.

Finland	04x, 0457, 050, 0500	**Tyskland**	015xx, 0160-0163, 017x
Sverige	070, 072, 073, 076	**Litauen**	06xx
Norge	4, 59, 9	**Lettland**	2xx
Estland	5x	**Ryssland**	89xx

Figur 8. Lista på mobil-prefix. [13]

2.2.5 Abonnentnummer

Abonnentnummer är det nummer som fås av operatören vid anskaffning av ett SIM-kort. Detta nummer var förut bundet till det SIM-kort som det blev anslutet till, men nuförtiden är det möjligt att använda samma nummer vid byte av SIM-kort eller operatör. Abonnonetnumrets längd och struktur beror på från vilket land det kommer.

2.3 SMS

Detta kapitel behandlar tjänsten SMS och SMS-protokoll samt lite kort om vad SMSC är.

2.3.1 Allmänt om SMS

SMS (Short Message Service) är en tjänst för korta textmeddelanden som sänds mellan mobiltelefoner eller från dator till mobiltelefon. SMS har många användningsområden, men personliga textmeddelanden är det vanligaste användningsområdet. Som Friedhelm Hillebrand säger i sin bok: *”SMS has many applications globally, but personal text messaging, measured in terms of traffic and users, is by far the most common SMS application”*. Ett meddelande kan vara upp till 160 tecken lång med sju-bitars teckenuppsättning. Detta har senare också utökats till kedjade meddelanden viket möjliggör 6 x 153 tecken långa meddelanden. För att detta skall vara möjligt att para ihop sammanhängande meddelanden, så används UDH (User Data Header) för att para ihop meddelanden. Orsaken till varför det är 160 tecken är för att år 1985 så hade en av GSM's ordförande Friedhelm Hillebrand en uppgift att bestämma hur långt ett meddelande skulle vara. Hillebrands teknik gick ut på att han skrev slumpmässigt utvalda meningar och frågor på sin skrivmaskin och såg att medeltalet tecken som använts var strax under 160. GSM använder sig av en teckenlista som heter GSM 03.38 (se figur 9) [14, 15]

Basic Character Set

	0x00	0x10	0x20	0x30	0x40	0x50	0x60	0x70
0x00	@	Δ	SP	0	¡	P	¿	p
0x01	£	_	!	1	A	Q	a	q
0x02	$	Φ	"	2	B	R	b	r
0x03	¥	Γ	#	3	C	S	c	s
0x04	è	Λ	¤	4	D	T	d	t
0x05	é	Ω	%	5	E	U	e	u
0x06	ù	Π	&	6	F	V	f	v
0x07	ì	Ψ	'	7	G	W	g	w
0x08	ò	Σ	(	8	H	X	h	x
0x09	Ç	Θ	)	9	I	Y	i	y
0x0A	LF	Ξ	*	:	J	Z	j	z
0x0B	Ø	ESC	+	;	K	Ä	k	ä
0x0C	ø	Æ	,	<	L	Ö	l	ö
0x0D	CR	æ	-	=	M	Ñ	m	ñ
0x0E	Å	ß	.	>	N	Ü	n	ü
0x0F	å	É	/	?	O	§	o	à

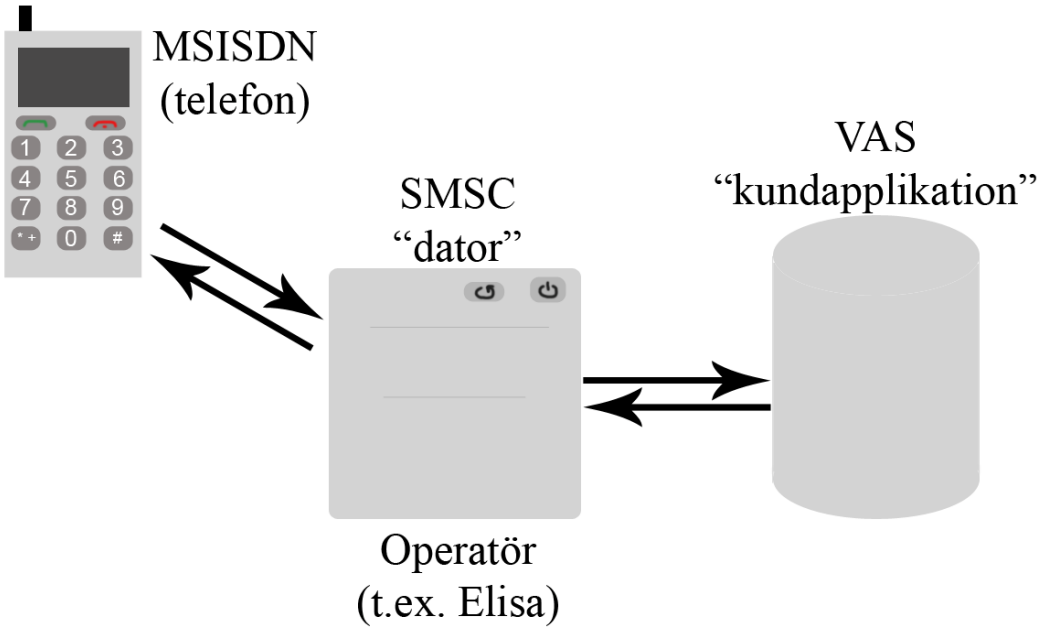
Basic Character Set Extension

	0x00	0x10	0x20	0x30	0x40	0x50	0x60	0x70
0x00					|			
0x01								
0x02								
0x03								
0x04		^						
0x05							€	
0x06								
0x07								
0x08			{					
0x09			}					
0x0A	FF							
0x0B		SS2						
0x0C				[				
0x0D	CR2			~				
0x0E				]				
0x0F			\					

Figur 9. GSM 03.38 teckenlista [16].

2.3.2 SMSC

SMSC (Short Message Service Centre) är ett nätverks element i mobiltelefonnätverket (se figur 10), vars uppgift är att spara, skicka vidare, konvertera och leverera SMS. SMSC används ofta för att göra möjligt för tredje parter (dvs. VAS (Value-Added Service providers) så som nyhets organisationer) att skicka meddelanden, ofta i massutskick, men kan även användas för "SMS peering".



Figur 10. En illustration på hur SMSC fungerar.

2.3.3 SMS-protokoll

EMI (External Machine Interface) är ett protokoll som används mest för att ansluta SMSC för mobiltelefoner. EMI är utvecklad av CMG (Computer Management Group) som nu är en del av Acision (Acision har bildats efter att IT-företaget Logica splittrades upp). Med detta protokoll är det möjligt för användaren att specificera adressen för den som sänder ett meddelande. Adressen kan antingen vara användarspecifik eller ett personifierat telefonnummer. Men detta beror på ifall operatören tillåter detta. [17]

CIMD2 (Computer Interface to Message Distribution) är ett protokoll som är utvecklad av Nokia för deras SMSC som heter Nokia Siemens Networks. CIMD2 PDUs (Protocol Data Unit) är binärt kodade för bättre prestanda. [18]

SMPP (Short Message Peer-to-Peer) är ett öppet protokoll. Det är designad för att ge ett flexibelt datakommunikationsgränssnitt för överföring av data mellan ett Message Centre och ett SMS applikationssystem. SMPP var ursprungligen utvecklad av Aldiscon, som är ett litet Irländskt företag som senare blev uppköpt av Logica. [19]

Nedan är ett exempel på ett meddelande som skickar "hello" som är formaterat i olika protokoll.

EMI	stx01/00045/O/30/66677789///1//////68656C6C6F/CEetx stx01/00041/R/30/A//66677789:180594141236/F3etx stx01/00052/O/30/66677789///1/558/0138////68656C6C6F/3Aetx stx01/00041/R/30/A//66677789:180594141430/EFetx

Detta exempel är uppbyggt på fyra paket. Där den första raden är ett paket som innehåller en förfrågan från sändaren att få skicka ett meddelande som innehåller texten "hello".

stx:	Början av paketet
01:	Transaktionsnummer
00045:	Meddelandelängd (för första paketet)
O / R:	Operation (Operation) och Result (Resultat)
30:	Sändningsoperation
66677789:	Mottagarnummer
180594141236:	Tidsstämpel i DDMMYYHHmmss
68656C6C6F:	Meddelandet "hello" uttryckt i hexadecimaler
CE:	Paketets checksum (för första paketet)
/:	Separator
etx:	Slutet av paketet

CIMD2	<STX>03:007<TAB>021:12345678<TAB>033:hello<TAB><ETX> <STX>53:007<TAB>021:12345678<TAB>060:971107131212<TAB><ETX>

Detta exempel är uppbyggt på två paket. Där den första raden är ett paket som innehåller en förfrågan från sändaren att få skicka ett meddelande som innehåller texten "hello". Det andra paketet innehåller svaret på förfrågan. I detta fall godkänns meddelandet med respons koden 53.

<STX>:	Början av paketet
<TAB>:	Användas för att separera värden (Delimiter)
007:	Paketnummer
03:	Skicka ett meddelande (Operationskod)
53:	Meddelandet godkänt (Responskod)
021:12345678:	Mottagarnummer
033:	Meddelandefält
060:	Tidsstämpel i UNIX-format
<ETX>:	Slutet av paketet

SMPP
'command_length', (60) ... 00 00 00 3C
'command_id', (4) ... 00 00 00 04
'command_status', (0) ... 00 00 00 00
'sequence_number', (5) ... 00 00 00 05

'service_type', () ... 00
'source_addr_ton', (2) ... 02
'source_addr_npi', (8) ... 08
'source_addr', (555) ... 35 35 35 00
'dest_addr_ton', (1) ... 01
'dest_addr_npi', (1) ... 01
'dest_addr', (555555555) ... 35 35 35 35 35 35 35 35 35 00
'esm_class', (0) ... 00
'protocol_id', (0) ... 00
'priority_flag', (0) ... 00
'schedule_delivery_time', (0) ... 00
'validity_period', (0) ... 00
'registered_delivery', (0) ... 00
'replace_if_present_flag', (0) ... 00
'data_coding', (0) ... 00
'sm_default_msg_id', (0) ... 00

Detta exempel skickas som ett enda paket. Det verkliga paketet innehåller inte parameterförklarningarna som syns på bilden. All meddelandeinformation skall uttryckas med hexadecimala värden.

3 Standarder

Här presenteras de tekniker som användes under arbetets gång.

3.1 CSV

CSV (Comma-separated value) ibland kallad för character-separated values, för att separator tecknet behöver inte vara ett kommatecken. CSV-formatet användes redan år 1967. CSV-filer är vanliga textfiler som innehåller data i ett format som är strukturerad för tabeller. CSV-filer kan användas för att exportera och importera data mellan två olika program som kan kräva helt olika former av in- och utgående data-format. CSV-filer kan användas av vilket kalkylprogram som helst, som t.ex. Microsoft Excel, Open Office eller Google Spreadsheets. Även med vanliga textredigerare som Microsoft Notepad eller Notepad++ går det att öppna CSV-filer. [20]

3.2 C#

C# (C-Sharp) är ett objektorienterat programmeringsspråk som utvecklades av Microsoft. C# är baserat på C++, men liknar Java. Som Andrew Troelsen säger i sin bok: *”C# is a programming language whose core syntax looks very similar to the syntax of Java.”*. C# är ett av de programmeringsspråk som är designat för CLI (Common Language Infastructure). Programmeringsspråket är designat för att vara ett plattformsoberoende språk, men det används oftast på Windows. Det finns två olika projekt som innehåller en C#-kompilator med öppen källkod dessa heter Mono och DotGNU. Den nyaste versionen av C# är 5.0 och den släpptes 15 augusti 2012. [21]

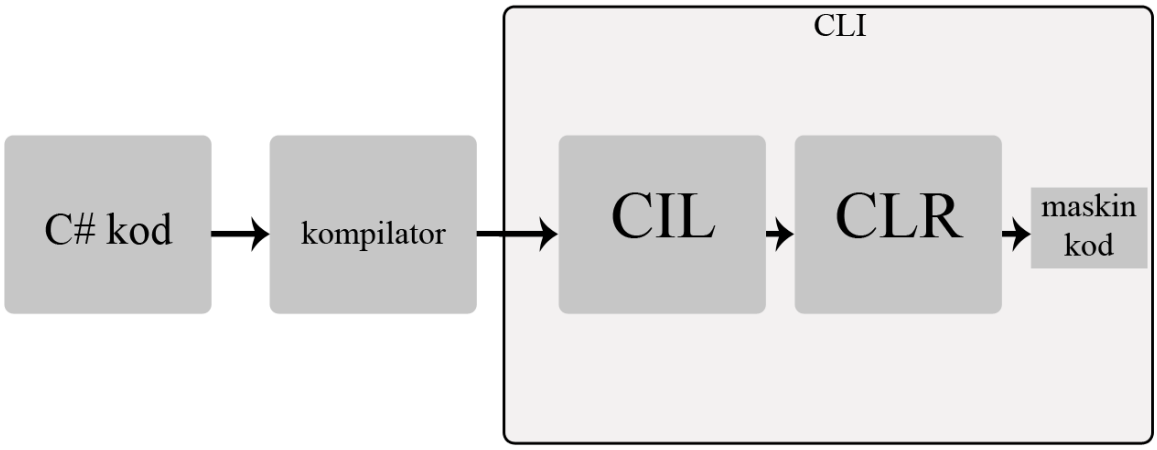
Nedan är ett exempel på en enkelt exempel på inmatning av data och if-sats.

```
class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Skriv ett tal som är mindre än 50");
            string inmatat = Console.ReadLine();
            int x = int.Parse(inmatat);

            if (x < 50)
            {
                Console.WriteLine("Bra gjort!");
            }
            else
            {
                Console.WriteLine("Nu gjorde du inte som jag skrev");
            }

            Console.ReadLine();
        }
    }
```

Figur 11. Exempel på C#-kod

3.2.1 CLI

CLI (Common Language Infrastructure) är en öppen specifikation som är utvecklad hos Microsoft. CLI beskriver exekverbar kod och en exekveringsmiljö som är den viktigaste biten i flera implementationer t.ex. .NET Framework, Mono och Portable .NET. Med denna specifikation så är det möjligt att använda flera hög-nivå språk på olika plattformar utan att behöva skriva om dem för den specifika arkitekturen. [22]



Figur 12. En illustration av CLI

3.2.2 CLR

CLR (Common Language Runtime) är en virtuell maskin och är en viktig del av Microsofts .NET ramverk. CLR har hand om exekveringen av .NET program. CLR är en implementering av standarden CLI. CLR exekverar en typ av bytekod som kallas CIL (Common Intermediate Language). CIL är ett mellanliggande objektorienterad högnivå-assemblyspråk.

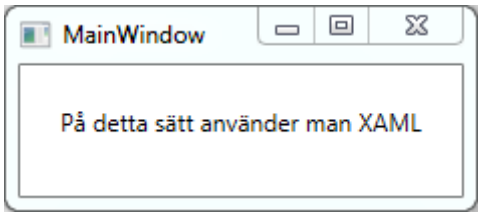
Fördelarna med CLR är att utvecklaren inte måste tänka på vilken specifik CPU han/hon utvecklar till. CLR har även hand om minneshantering, trådhantering, undantaghantering, skräpinsamling och datasäkerhet.

Vid kompilering av kod skriven i ett programmeringsspråk som stöds av .NET (t.ex. C#, VB.NET eller C++) så blir koden först konverterad till ett assembly med CIL-kod. Sedan vid exekveringen av detta assembly så kompileras CIL-koden till maskinkod av JIT-kompilatorn (Just In Time). [23]

3.3 XAML

XAML (Extensible Application Markup Language) är Microsofts variant av XML för att beskriva ett grafiskt användargränssnitt. XAML är en viktig komponent av .NET framework 3.0 och .NET framework 4.0, det används framför allt i WPF, Silverlight, Windows Workflow Foundation och Windows store applikationer. I WPF så används XAML för att definiera gränssnittselement, händelser och andra funktioner (se figur 13). [24]

```
<StackPanel>
    <TextBlock Margin="20">På detta sätt använder man XAML</TextBlock>
</StackPanel>
```



Figur 13. En illustration på hur man skriver ut text med XAML.

3.4 WPF

Windows Presentation Foundation är ett grafiskt system för rendering av användargränssnitt för .NET applikationer. WPF som förut gick under namnet "Avalon" släpptes i samband med .NET framework 3.0 år 2006 och kommit som standard med alla Windows-versioner sedan Windows Vista och Windows Server 2008. Istället för att använda sig av det äldre GDI(Graphics Device Interface) systemet, så använder sig WPF utav grafikbiblioteket DirectX. DirectX använder sig utav datorns grafikkort när det skall rendera något på skärmen. Vilket gör att processorns kapacitet frigörs och kan användas till annat.

WPF använder sig utav XAML för att definiera och länka olika element i grafiska gränssnittet.

WPF lämpar sig för skrivbordsapplikationer eller som en komponent av en hemsida. [25]

3.5 MVC

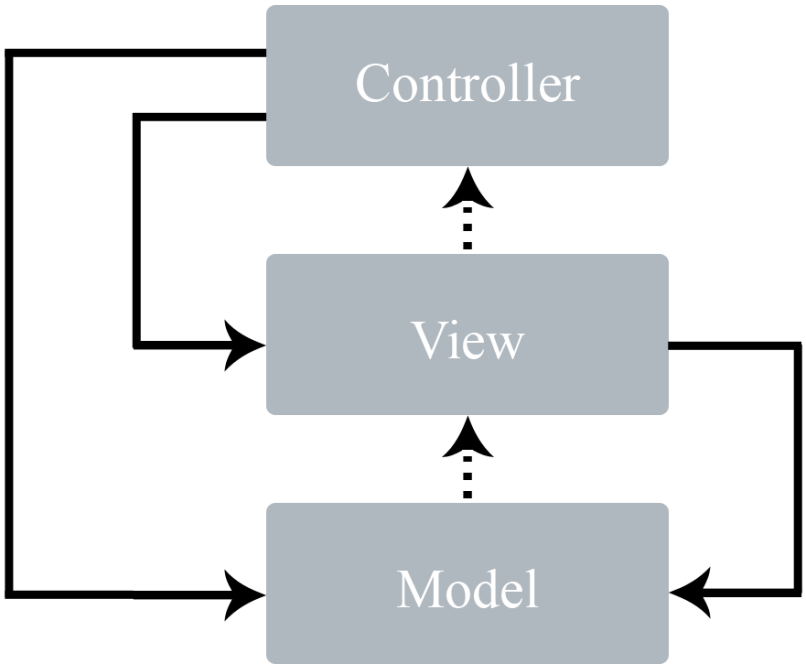
MVC (Model View Control) är ett sätt att strukturera koden när man skall implementera ett användargränssnitt. Vid utveckling av mera komplicerade applikationer kan det vara bra att separera data- (Model) och presentationsdelen (View) av koden. Detta gör så att det är möjligt att göra ändringar i presentationsdelen utan att det uppstår fel vid datahanteringen, samt vid ändringar till data delen så behövs inte några ändringar i presentationsdelen. Detta problem löses med hjälp av en mellanliggande komponent vid namn Controller.

Model: Model meddelar dess anslutna Views och Controllers när det har gjorts ändringar i dess *"state"*. Detta gör så att Viewen kan producera en uppdaterad *"output"*, och att Controllern kan byta vilka kommandon som är tillgängliga.

View: View begär information från Model och använder informationen för att rendera den till lämplig form för interaktion, som vanligtvis är ett användargränssnitt.

Controller: Controllern kan skicka kommandon till Modeln för att uppdatera Modelns tillstånd (t.ex. vid editering av ett dokument). Det kan även skicka kommandon till dess associerade View för att ändra på hur Viewen visar Modeln (t.ex. genom att scrolla genom ett dokument).

MVC är ett av de äldsta sätten att strukturera koden vid kodning av grafiska användargränssnitt, och ett av de första försöken att beskriva och implementera mjukvarukonstruktioner i relation till deras ansvarsområden. Fastän att MVC var ursprungligen utvecklad för att användas vid strukturering av skrivbordsapplikationer så är det i dagsläget mest använt vid utveckling av webapplikationer. Det har även utvecklats flera kommerisiella och icke-kommersiella webapplikationsramverk som grundar sig på samma designmönster. [26]



Figur 14. En illustration på hur MVC fungerar.

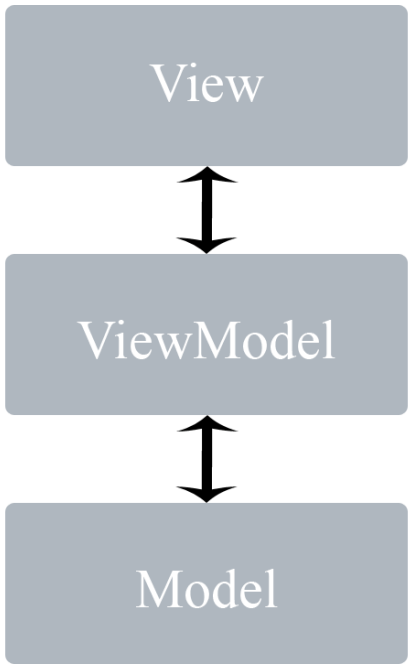
3.6 MVVM

MVVM (Model View ViewModel) är ett designmönster som används vid utveckling av applikationer med ett grafiskt användargränssnitt. MVVM är en specialiserad version av design mönstret för presentation modellen för Microsoft utvecklat av Martin Fowler. MVVM är baserad på MVC. MVVM är ämnat för utvecklingsplattformer för användargränssnitt som stöder händelsedriven programmering, specifikt WPF (Windows Presentation Foundation) och Silverlight på .NET-plattformar kodade med XAML- och .NET-programmeringsspråk. MVVM var designad för att använda datakopplingsfunktioner i WPF, för att bättre kunna separera View-lager utvecklingen från resten av mönstret genom att nästintill ta bort all grafisk användargränssnitskod från View lagret. Detta gör det lättare för flera personer att arbeta på ett projekt samtidigt.

Model: Model representerar en uppsättning av klasser som beskriver businesslogiken och data. Den definierar även regler för hur data kan bli ändrad och manipulerad.

View: Views uppgift är att visa alla element som visas i ett användargränssnitt, som knappar, etiketter och andra kontroller.

ViewModel: ViewModel's uppgift är att exponera metoder, kommandon och andra egenskaper som kontrollerar vyns ”view” tillstånd, manipulera modellens resultat på bas av vyns utförande, och aktivera händelsen i vyn. [27]



Figur 15. En illustration på hur MVVM fungerar.

4 Användargränssnitt

Ett användargränssnitt är det medium som maskin och människa använder för att kommunicera med varandra. Målet med användargränssnitt är att användaren skall effektivt kunna hantera maskinen, samt att maskinen skall kunna skicka tillbaks information som hjälper användaren, och att användaren skall känna att han/hon lyckades med något. Som Ben Shneiderman och Catherine Plaisant säger i sin bok: *”Effective interfaces generate positive feelings of success, competence, mastery, and clarity in the user community”*.

4.1 Saker som uppgör ett bra användargränssnitt

Klarhet: Gränssnittet skall vara lättuppfattad, ha ett lättläst språk och ha ett bra flöde.

Kortfattad: Inte för mycket på skärmen så att användaren blir förvirrad.

Familjärt: Även om någon använder ett gränssnitt för första gången så kan det fortfarande finnas vissa saker som är bekanta.

Responsivt: Ett bra gränssnitt skall inte kännas trögt. Med detta menas att gränssnittet skall visa tydligt för användaren vad som händer, det skall även visa ifall information som användaren har fyllt i har blivit processerad.

Konsistens: Ett gränssnitt skall se likadant ut genom hela programmet för att användaren inte måste lära sig nya saker hela tiden.

Estetik: Ett gränssnitt behöver inte se bra ut för att göra sitt jobb, gör man något som ser bra ut så blir tiden som användaren spenderar med ditt program trevligare.

Effektivitet: Ett bra användargränssnitt skall göra så att användaren är mer produktiv genom genvägar och bra design.

Förlåtelse: Ett bra användargränssnitt skall inte straffa användaren för deras misstag, istället så skall det ge korrekt information hur man kan gå till väga för att fixa dessa misstag. [28]

5 Verktyg

I detta kapitel behandlas de verktyg som användes under arbetets gång.

5.1 Visual Studio 2010

Visual Studio 2010 är en avancerad programutvecklingsmiljö utvecklad av Microsoft. Med Visual Studio så kan man utveckla program och webbapplikationer för .NET-plattformen. Visual Studio stöder de vanligaste programmeringsspråken som t.ex. C, C++, C#, VB.NET (Visual Basic .NET). Skall man använda sig av något ovanligare språk så går det att ladda ner det separat. Utvecklaren får stöd vid programmeringen med hjälp av funktionen IntelliSense som automatiskt kan fylla i programmets syntax. Det finns en gratis version av Visual Studio vid namn Visual Studio Express edition som har de viktigaste funktionerna. Det finns även Professional, Premium och Ultimate versioner för den mer seriösa användaren. [29]

5.2 Notepad++

Notepad++ är utvecklad av Don Ho, och är ett textediteringsprogram med öppen källkod. Notepad++ kan ses som en ersättare till Windows egna textediteringsprogram Notepad, fast med mycket flera funktioner. Verktyget har blivit populärt på grund av sitt breda stöd på många programmeringsspråk och scriptsspråk. Färgkodningsfunktionen i verktyget är en annan stark fördel för användaren, när koden blir mer lättläsligt. [30]

6 Utförande

I detta kapitel så tas det upp hur det gick till vid planeringen av detta projekt och lite om hur programmet är uppbyggt och fungerar.

6.1 Forskning och planering

Eftersom detta projekt var en vidareutveckling av två ursprungliga program AIListUpdater och MobiSendParser. Det första som gjordes var att studera dessa program, samt att prata med de personer som har gjort dem. För det är viktigt att veta hur de fungerar och för att se om någon del av koden kunde återanvändas. Dessa program är gjorda i Windows Forms och är strukturerade enligt MVC-designmönstret.

Nästa steg var att hålla ett möte med de tekniska handledarna och övriga personer som var involverade i detta projekt för att fastställa kravspecifikationen. Några saker som togs upp var vilka funktioner som detta program skulle innehålla, vilka länders telefonnummer som skall gå att använda i detta program, språkinställningar så som svenska, finska och engelska. Det togs även upp vilka format som skall stödas vid importering och exportering av filer, inställningarna från föregående session skall spara till nästa, och lite hur användargränssnittet skall se ut.

Efter detta satsades flera dagar på att studerade telefonnummers regler till olika länder. Det var svårt att hitta pålitlig och lättförståelig information, för en del av sidorna som användes så var motsägande till varandra.

Efter att tillräckligt med information hade blivit insamlat om alla länder så var det dags att tänka på hur designen på programmet skulle se ut. Enligt kravspecifikationen så skulle designen vara lättförståelig och gärna vara gjord som en "Wizard", dvs. att programmet skall gå steg för steg med användaren och ge tillräckligt med information vid varje steg. Ett par olika designalternativ av programmet tillverkades och handledaren bestämde sig för en design. Tack vare detta programmeringsskede så höjdes kunskaperna om WPF.

6.2 Val av kodningsmetod

Eftersom kurser i WPF och MVVM hade nyligen hållits i skolan, så var det färskt i minnet. Det var inte så konstigt att det blev just den metodiken som valdes att arbeta med. Företaget hade använt sig mest utav MVC förut, men det var ett ypperligt tillfälle för dem att få lite mera information om hur MVVM fungerar. Som det nämndes i rubriken 3.6 om MVVM så baserar sig MVVM på MVC. Hoppet mellan dessa borde inte vara alltför svårt. WPF valdes för att det var smidigt att göra grafiska gränssnitt i.

6.3 Model

Detta kapitel behandlar de klasser som används i detta projekt.

6.3.1 CFile

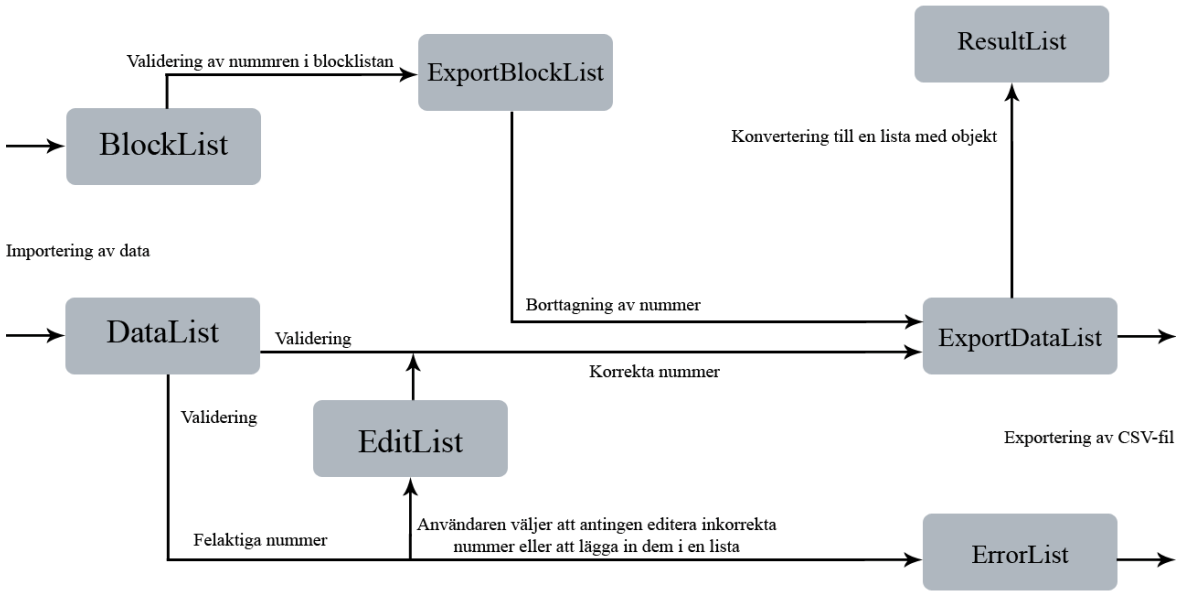
Klassen CFile används för att kunna göra varje rad i en CSV-lista till ett objekt som behövs när listorna skall visas grafiskt i ett DataGrid, det används i både ”Visa data” och ”Visa resultat” samt vid editeringstillfället.

6.3.2 FileHandler

FileHandler-klassen innehåller följande metoder:

- Öppna CSV- och txt-filer.
- Skriva till nya CSV-filer.
- Välja vilken mapp resultat filen skall exporteras till.
- Validera listorna.
- Se till att användaren har satt in korrekt formaterat värde i textrutan vid val av kolumner.

FileHandler har även alla de olika listor som används för att behandla och modifiera dessa CSV-filers innehåll. Det finns olika listor för olika stadier i valideringen. De listor används är en lista med strängar som tas in vid inläsningen av CSV-filen, vid namn DataList. Denna lista är den huvudsakliga listan som kommer användas vid valideringen. Det finns även en lista vid namn BlockList, denna lista fungerar på samma sätt som DataList, men den används för att ta bort de nummer som användaren inte vill ha med. Även de nummer som är med i BlockList så kommer att valideras, så det inte finns några inkorrekta nummer i den listan. Efter att numren från BlockList har blivit validerade så flyttas de till en ny lista som heter ExportBlockList. ExportBlockList används senare för att ta bort de nummer som inte skall vara med i ExportDataList. Efter att DataList har blivit validerat skickas de felaktiga numren antingen till en ErrorList eller till en EditList, beroende på vad man har valt vid inställningarna. ErrorList sparar undan alla felaktiga nummer och som sedan exporteras till en separat CSV-fil. EditList används när användaren vill kunna editera de felaktiga numren. Efter att användaren har skrivit in ett nytt nummer så valideras det numret för att sedan läggas till i ExportDataList. Som har nämnts tidigare måste man använda sig utav objekt vid användning utav ett DataGrid, därför är EditList en lista med objekt. Efter att alla nummer validerats och är korrekta skickas de till ExportDataList, som är den slutliga listan som används när den skall exporteras till en CSV-fil. När man vill se hur resultatet kommer att se ut så skall det även denna gång visas i ett DataGrid, så därför är ResultList en lista med objekt som är konverterad från ExportDataList.



Figur 16. Bild över hur listorna fungerar.

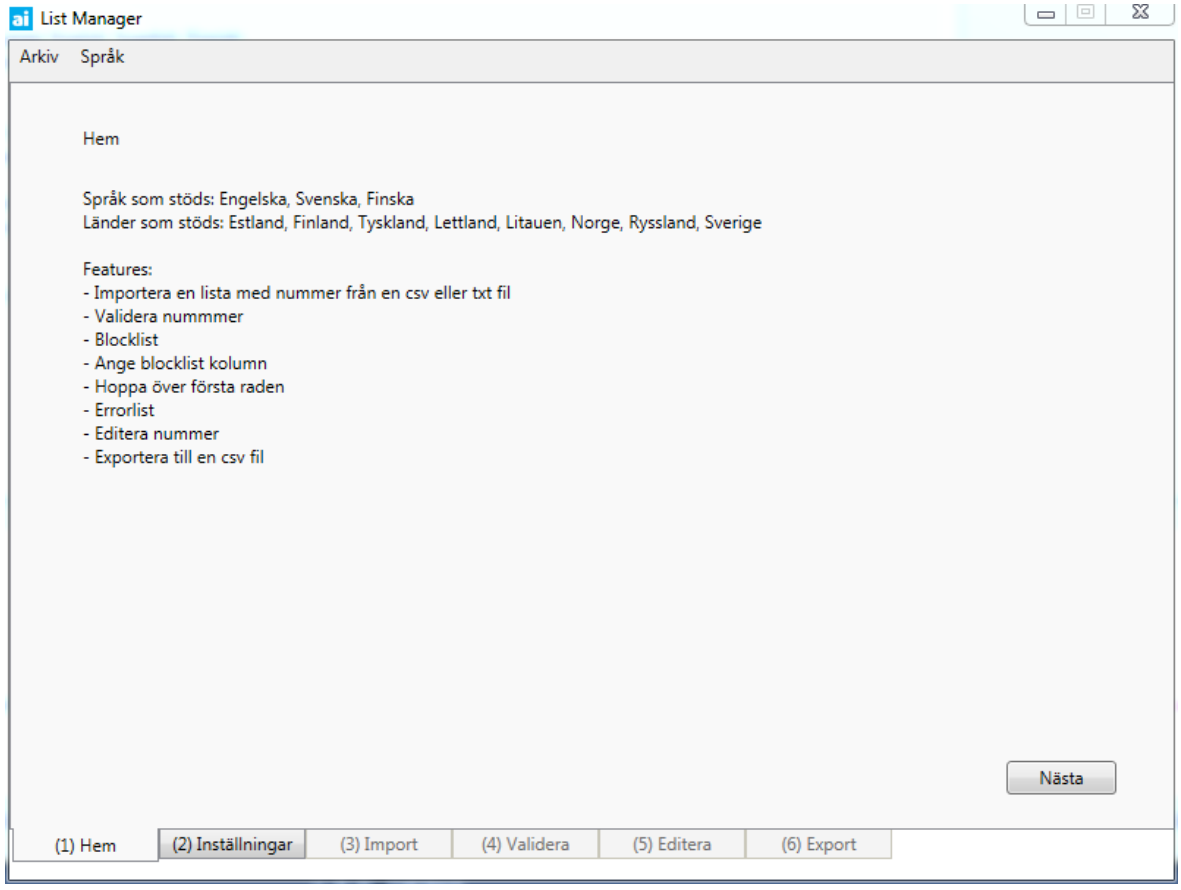
6.3.3 Validator

Klassen Validator innehåller en metod som heter FormatGsmNumber. Denna metod tar in ett nummer i taget och information om vilket land det numret kommer från. Validatorns uppgift är att se så att numret är korrekt formaterat enligt de regler från det land numret kommer från och så ser validatorn till att det inte finns mellanslag, extra bokstäver eller tecken i numret. För att radera dessa fel använder sig validatorn av en Regex som bara tillåter siffror. Där det kändes enklare att använda sig utav if-satser istället för Regex skrevs reglerna för varje land, vilka prefix numret skall ha, hur långt telefonnumret är osv.

6.4 View

I menyn kan användaren välja om man vill göra en ny session eller stänga programmet. Användaren kan även välja vilket språk som skall användas i programmet, i nuläget finns det att välja mellan svenska, engelska och finska.

Hemrutan (se figur 17) visar information om vilka språk som stöds, vilka länders telefonnummer som stöds, samt vilka funktioner verktyget innehåller. Med denna sida var det tänkt att användaren skulle klart och tydligt kunna se nya egenskaper som blir lagda till i framtida versioner.

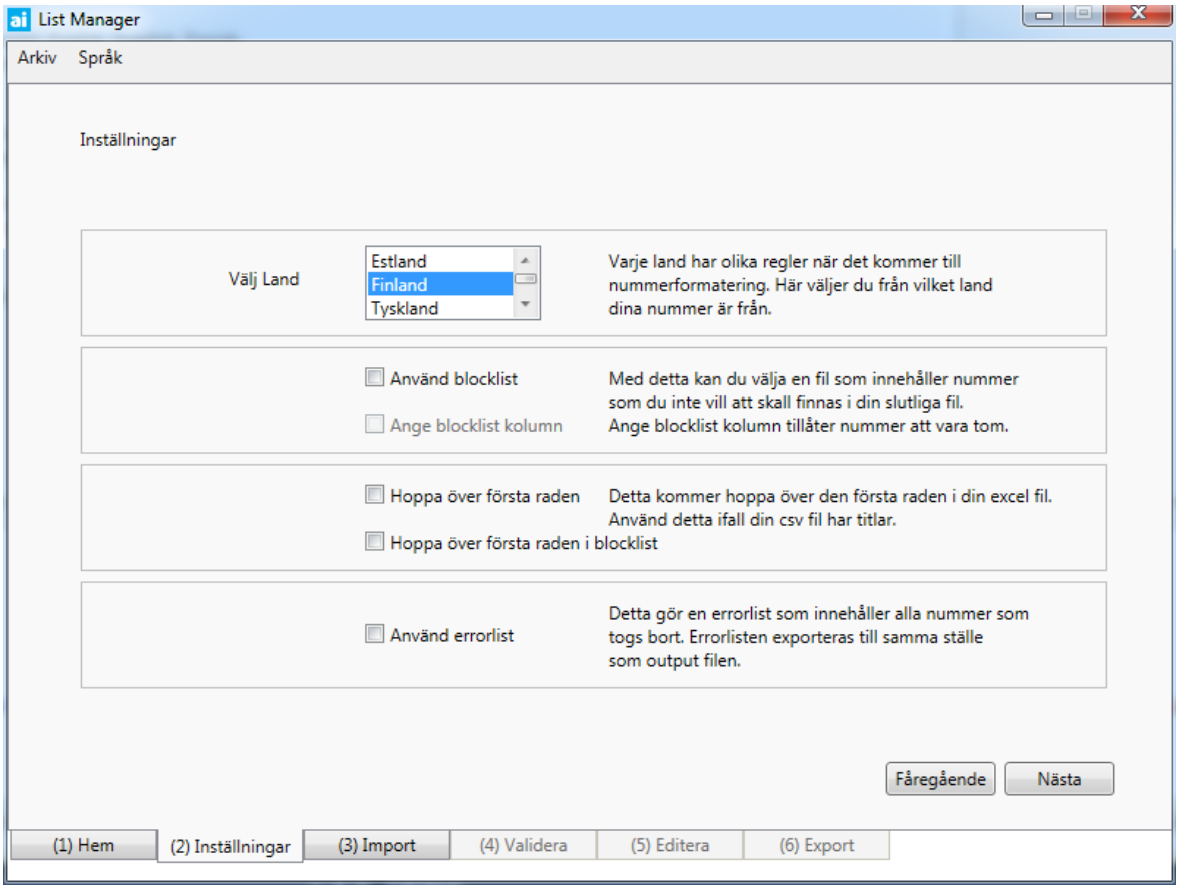


Figur 17. Hem i List Manager.

Vid rutan Inställningar (se figur 18) skall man välja från vilket eller vilka land/länder telefonnumren i listan kommer från. Det finns även möjligheten att använda en ”blocklist”, som är en lista med nummer som man inte vill inkludera i utskickslistan. Med denna funktion tas då dessa nummer bort från utskickslistan. Denna lista ges av användaren. Det finns också en funktion att ange vilken kolumnrad telefonnumren ligger i blocklistan, denna funktion skall användas utifall att användarens blocklist inte innehåller bara

telefonnummer, utan även annan information. Funktionen som kallas för ”hoppa över första raden” skall användas ifall den lista man har innehåller en rad med titlar, samma sak gäller i blocklist. Denna funktion skulle ha automatiserats, för det borde vara möjligt. Använder man sig utav errorlist-funktionen hoppar programmet över editeringsfasen och istället dumpar verktyget alla felaktiga nummer till en Errorlist som sedan exporteras till en Errorlist-fil.

För att underlätta användningen utav detta verktyg så kommer verktyget ihåg vilka inställningar användaren använde senaste gång. Vid nästa programstart är de inställningarna färdigt ikryssade.

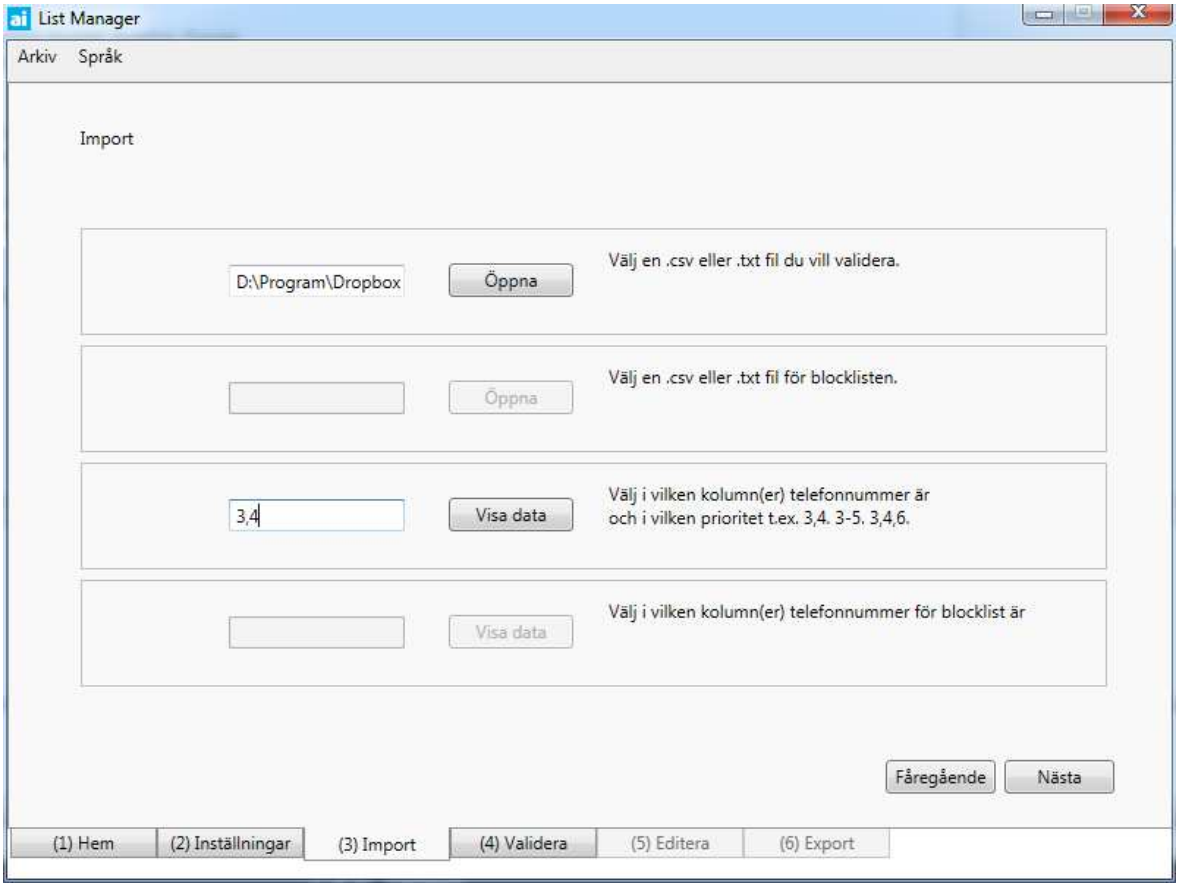


Figur 18. Inställningar i List Manager.

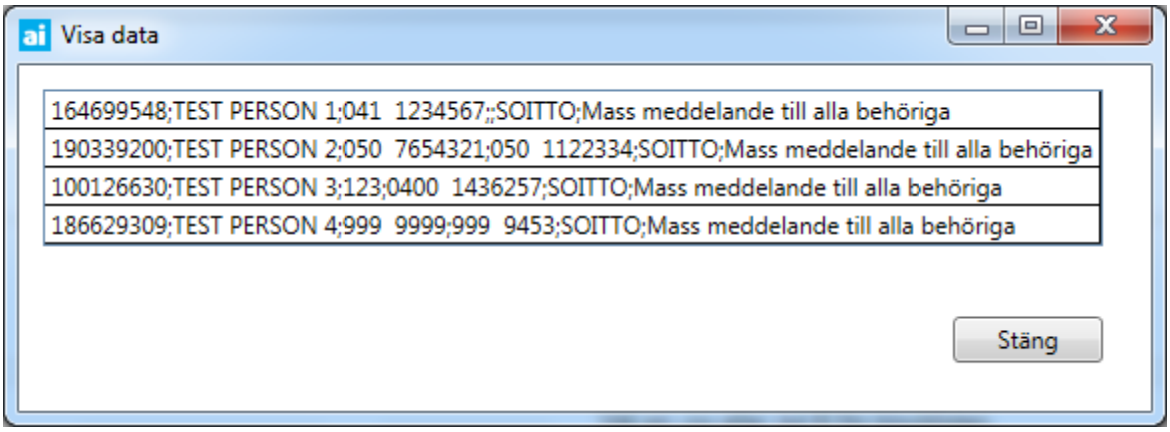
Vid importrutan (se figur 19) anges vilken CSV- eller txt-fil man vill använda, och om en blocklist används så anges dess filsteg. När dessa filer har valts så kan man trycka på ”Visa data-knappen”. Denna knapp öppnar ett nytt fönster där man ser vad filen innehåller (se figur 20). Med hjälp av denna information så skall man se i vilken eller vilka kolumn(er) telefonnumren ligger. Det finns en sådan funktion att man kan skriva in flera kolumnnummer och separera dem med ett kommatecken, samt om alla kolumner kommer efter varandra går det att lägga startnummer följt av ett streck och sedan ett slutnummer.

Verktyget kommer i senare skede att se till att det är bara ett telefonnummer per rad. För att kunna bestämma vilket kolumn som skall prioriteras så finns det logik så att det första numret man skriver in är högsta prioritet. Grundtanken med ”Visa data-rutan” var att man skulle kunna välja grafiskt i Datagridet vilken kolumn telefonnumren ligger i, men p.g.a. en bugg så visas inte korrekt data.

För att användaren inte i misstag skall trycka på fel sak så finns det några flikar och knappar som det inte går att trycka på förrän man har t.ex. öppnat en fil och valt vilken kolumn telefonnumren är. Vid inmatning av kolumnnummer så finns en valideringsmetod som använder sig utav en Regex, för att kolla upp ifall den data man har matat in är korrekt formaterat enligt de regler som har skrivits upp. Om det inte är korrekt formaterat visas det med en röd ram runt textrutan och ”Nästa-knappen” blir otryckbar.

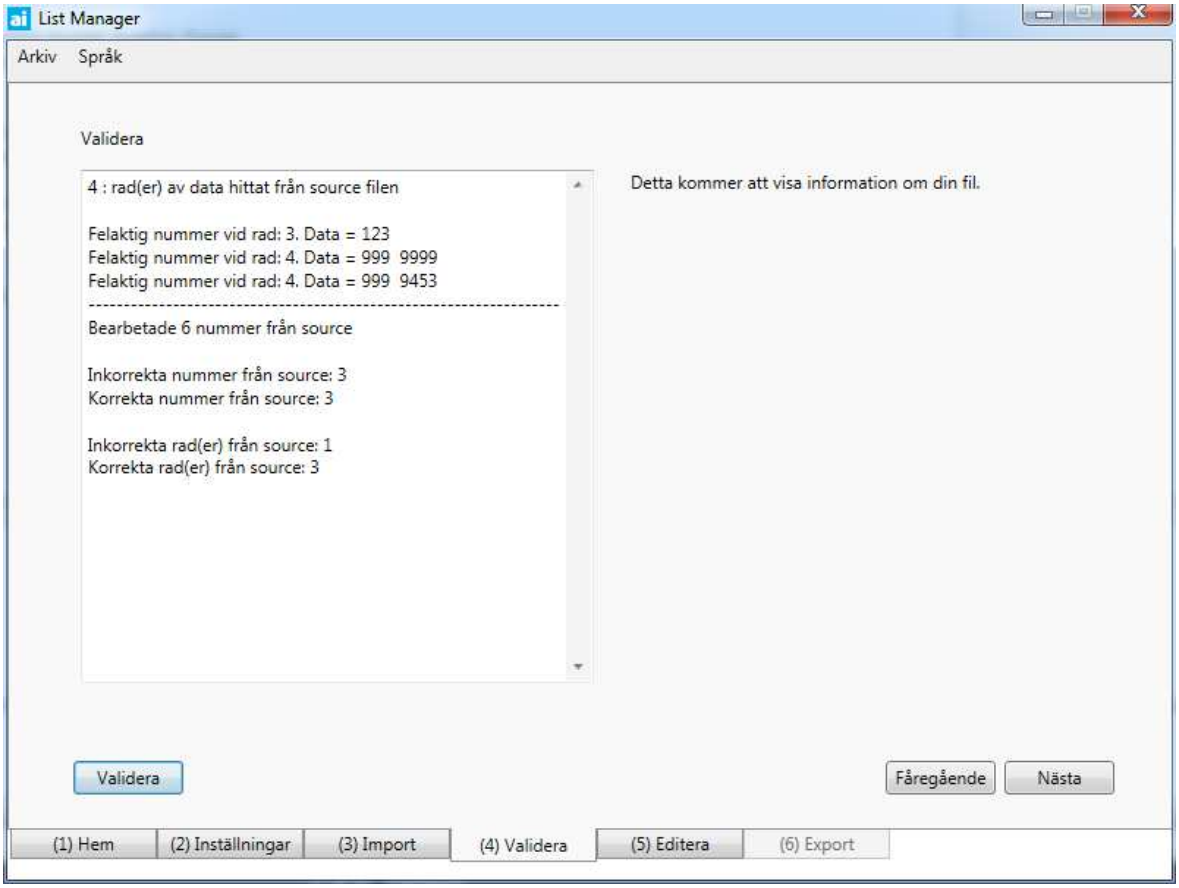


Figur 19. Import i List Manager.



Figur 20. Visa data i List Manager.

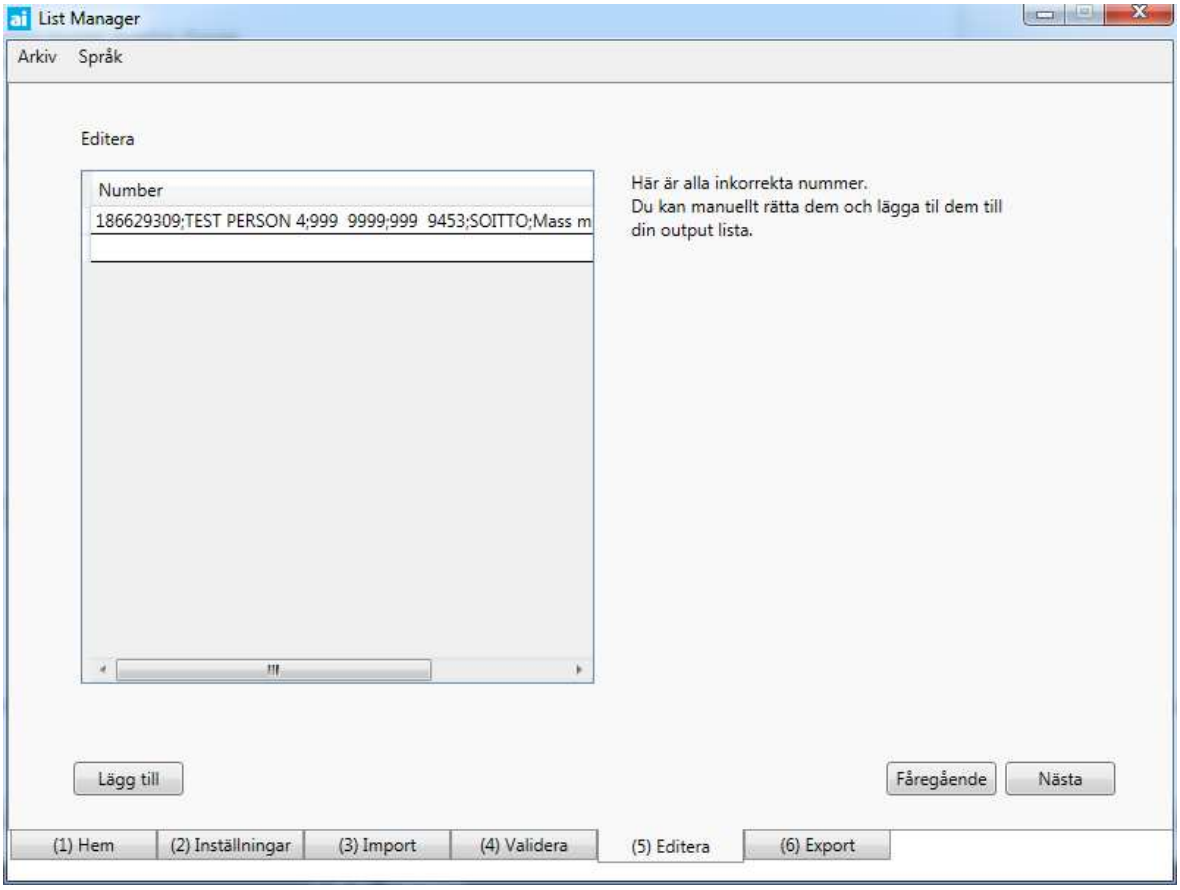
Validera-rutan (se figur 21) visar först hur många rader med data det har hittat i käll-filen och blocklist-filen om en sådan är vald. Efter man tryckt på Validera-knappen visar det data på vilka telefonnummer som är inkorrekta och på vilken rad de är, samt hur många de är.



Figur 21. Validera i List Manager.

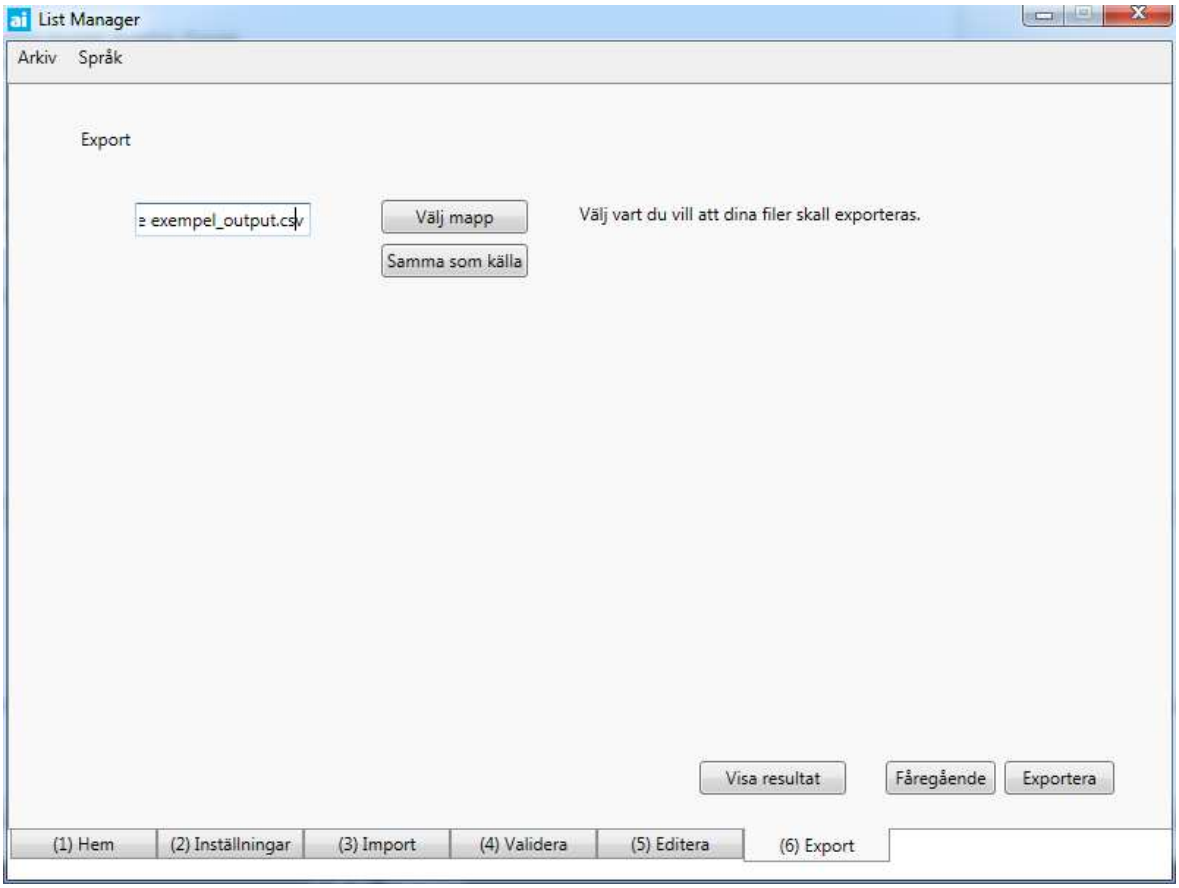
Vid Editera-fliken (se figur 22) så kan man editera inkorrekta rader, för att sedan kunna trycka på Lägg till knappen som validerar dessa nummer ännu en gång, och ifall de är korrekta så läggs de till i den slutliga listan. Grundtanken med editeringen var att det skulle

gå att editera de kolumner där telefonnumren finns, samt att det skulle vara lättare att navigera sig genom.

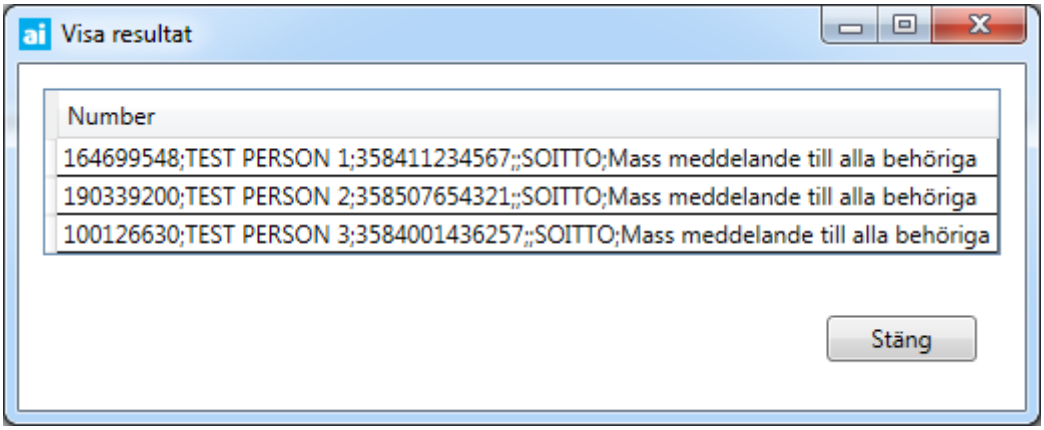


Figur 22. Editera i List Manager.

Till sist så skall man välja en destinationsmapp på Exportrutan (se figur 23). Där kan man antingen välja en specifik mapp eller så trycker man på knappen ”Samma som källa”, som lägger det i samma mapp som källfilen befinner sig i. Innan man exporterar kan man se hur resultatet kommer att se ut (se figur 24) genom att trycka på knappen ”Visa resultat”.



Figur 23. Export i List Manager.



Figur 24. Visa resultat i List Manager.

6.5 ViewModel

I detta kapitel så beskrivs vad projektets ViewModel innehåller.

6.5.1 MainViewModel

Programmet använder sig utav en ViewModel som kallas för MainViewModel. ViewModeln har hand om data som kommer från View och skickar det vidare till Model, där finns även logiken till alla knappar och komponenter. Den skickar även den information som skall visas i textrutan i Valideringsrutan.

6.5.2 RelayCommand

RelayCommand är en implementering av Icommand-gränssnittet. Detta gränssnitt innehåller två metoder, Execute och CanExecute. I Execute-metoden kodar man vad som skall hända när man trycker på någon komponent. Med CanExecute så kan man sätta villkor som skall uppnås för att Commandot skall köras eller inte. Detta användes för att kunna återanvända programkod för samma Command till t.ex. flera olika knappar.

6.6 Testning

Under utvecklingen av detta program testades det med jämna mellanrum av handledarna för att få feedback på arbetet och för att få reda på om det finns några buggar som borde fixas. Av allt som gjordes under detta projekt så var just buggfixandet det som tog mest tid. För att när en bugg fixades så hittades två nya. Det blev många sena kvällar där det prövades med olika telefonnummers listor med olika problem för att hitta buggar i validatorn, försöka fixa så att inställningarna sparas korrekt osv.

När alla korrigerbara buggar fixades skickades den slutliga versionen in för intern testning. Där handledarna gjorde många tester på listor som var riktiga utskick från tidigare kunduppdrag.

7 Resultat och diskussion

Här presenteras resultatet, möjligheter till vidareutveckling, de problem som uppstod under utvecklingen av verktyget, samt diskussion.

7.1 Resultat

Resultat blev ett fungerande verktyg med utrymme för en del vidareutveckling. En del mindre funktioner som var inplanerade blev aldrig klara och därmed aldrig implementerade. Men det blev sist och slutligen ett program som gör vad det i grunden skulle göra. Programmet testades i flera dagar internt inom företaget innan kunderna började använda sig utav det.

7.2 Problem

Under utvecklingen har jag stött på många problem, en del har tagit veckor innan jag har löst dem. En del problem fick jag helt enkelt ge upp med för de var högt över vad mina kunskaper klarar av.

En stor del av dessa problem har något MVVM-relaterat. Hela konceptet att strukturera koden var ganska svårt att hålla reda på för de projekt jag har gjort tidigare har varit väldigt små i jämförelse med detta projekt. Jag kunde väldigt lite om MVVM när jag började mitt projekt och lärde mig lite därefter. Att veta om man skall placera en sak till Model, View eller ViewModel och vad den skall få veta, är något som jag aldrig fick riktigt klart för mig.

Ett problem som jag försökte lösa i flera veckor men som jag till sist gav upp med var att kunna grafiskt välja vilken kolumn eller kolumner telefonnummer var. Problemet var att jag använde mig av ett DataGrid och inte fick in de värden jag ville ha. Ett DataGrid behöver få in värden som objekt, och jag lyckades inte omvandla och länka mina listor så att det fungerade. Om jag skulle klarat av att fixa detta problem så skulle jag stött på mera problem, när jag behöver veta vilka kolumner användaren väljer i DataGridet och vad det

värdet betyder i jämförelse med mina arrays, och sedan kunna lägga in detta värde i min kolumnvalstextruta.

Ett annat problem var att när jag började skriva programmet så tänkte jag inte alls på threading. När jag kom på det på slutet så för att få threadingen att fungera så skulle jag vara tvungen att skriva om största delen av koden, för koden var inte tillräckligt uppdelad i vykod och processeringskod.

7.3 Vidareutveckling

En av de planerade funktionerna som aldrig blev implementerade är en automatisk uppdaterare till programmet. Denna funktion kunde vara bra att ha ifall man i framtiden vill lägga till ett land, eller om man gör några andra ändringar. Med en automatisk uppdateringsfunktion så ser man även till att alla kunder använder sig utav samma version av programmet, vilket leder till att det blir lättare vid underhåll av programmet.

7.4 Diskussion

Det har varit väldigt ovanligt att arbeta med ett så mycket större projekt jämfört med tidigare erfarenheter, men det har varit väldigt lärorikt. Jag har fått lära mig att planering innan man börjar koda programmet är mycket viktigt. I mitt fall fick jag aldrig någon bra grund, jag började med att göra en enkel version av programmet och sedan lägga till funktioner efter hand. Men den största orsaken till detta problem var min bristande kunskap inom ämnet, med lite mera erfarenhet så kommer detta problem att minska med tiden. Det blev mycket extra arbete som i slutändan gav en produkt som inte är lika bra som om jag hade haft allting på klart och hade planerat ordentligt.

Fastän jag inte ännu i denna dag har allt på klart så lärde jag mig en del om hur MVVM fungerar och överlag hur man borde strukturera koden i sitt projekt.

Jag blev inte helt nöjd med slutliga designen, för jag hade bestämt mig för en version till att börja med, men vartefter tiden gick så lades flera funktioner till och det blev trångt och klumpigt.

Jag lärde mig även att jag borde arbeta på ett projekt oftare än vad jag gjorde eller att dokumentera bättre, för efter sommaren hade jag ganska långa pauser p.g.a. skolgången. Största delen av tiden gick till att komma ihåg vad jag höll på med förra gången.

Jag fick en bra inblick i hur det kan vara att ha ett arbete inom programmering. För min del går det att jämföra programmering med t.ex. att läsa en bok. Läser man boken för att man måste göra en recension så kan det bli tråkigt, men om man läser boken på fritiden utan någon press så kan det vara det hur roligt som helst. Med det menar jag att programmering kommer antagligen att bli mera en hobby än mitt yrke i framtiden.

8 Källförteckning

[1] Arena Partners [Online]
http://www.arenapartners.fi/sivut/images/stories/sisalto/konsernirakenne_iso.jpg [hämtat: 11.11.2014]

[2] Arena Interactive [Online]
http://www.arenainteractive.fi/ [hämtat: 09.10.2014]

[3] Arena Interactive kunder [Online]
http://www.arenainteractive.fi/wp-content/uploads/2014/03/Asiakkaita.png[hämtat: 11.11.2014]

[4] Eberspächer, Jörg. Bettstetter, Christian. &Vögel, Hans-Joerg. (2009). *GSM - Architecture, Protocols and Services.* Wiley.

[5] Telefonnummer [Online]
http://sv.wikipedia.org/wiki/Telefonnummer [hämtat: 15.10.2014]

[6] Internationellt prefix [Online]
http://www.itu.int/dms_pub/itu-t/opb/sp/T-SP-E.164C-2011-PDF-E.pdf [hämtat: 16.10.2014]

[7] Telefonnummer i Finland [Online]
https://www.viestintavirasto.fi/sv/internettelefoni/numreringitelefonnatet/mobilnat/mobilnatensriktnummer.html [hämtat: 05.11.2014]

[8] Landsnummer [Online]
http://www.itu.int/dms_pub/itu-t/opb/sp/T-SP-E.164C-2011-PDF-E.pdf [hämtat: 16.10.2014]

[9] Telephone Area Codes [Online]
http://www.lincmad.com/world.html [hämtat: 16.10.2014]

[10] Nationell destinationskod [Online]
http://www.itu.int/dms_pub/itu-t/opb/sp/T-SP-E.164C-2011-PDF-E.pdf [hämtat: 16.10.2014]

[11] Finlands teleområden [Online]
https://www.viestintavirasto.fi/sv/internettelefoni/numreringitelefonnatet/lokalsamtalochteleomraden/kartaoverteleomradena.html [hämtat: 16.10.2014]

[12] Telephone numbers in Finland [Online]
http://en.wikipedia.org/wiki/Telephone_numbers_in_Finland [hämtat: 16.10.2014]

[13] Mobil-prefix [Online]
http://en.wikipedia.org/wiki/List_of_mobile_phone_number_series_by_country [hämtat: 16.10.2014]

[14] Hillebrand, Friedhelm F. (2010). *Short Message Service (SMS): The Creation of Personal Global Text Messaging.* Wiley.

[15] SMS 160 character limit [Online]
http://www.tatango.com/blog/why-is-a-text-message-only-160-characters/ [hämtat: 27.09.2014]

[16] GSM character set[Online]
http://en.wikipedia.org/wiki/GSM_03.38 [hämtat: 11.11.2014]

[17] EMI UCP [Online]
http://www.nowsms.com/discus/messages/1/EMI_UCP_Specification_40-8156.pdf [hämtat: 01.10.2014]

[18] CIMD [Online]
https://www.elisa.ee/UserFiles/Dokumendid/CMID2_tehniline_spetsifikatsioon.pdf [hämtat: 01.10.2014]

[19] SMPP [Online]
http://docs.nimta.com/smppv34_gsmumts_ig_v10.pdf [hämtat: 01.10.2014]

[20] CSV [Online]
http://www.coolutils.com/Formats/CSV [hämtat: 21.09.2014]

[21] Troelsen, Andrew. (2012). *Pro C# 5.0 and the .NET 4.5 Framework sixth edition.* Apress

[22] CLI [Online]
http://www.pearsonhighered.com/samplechapter/0321154932.pdf [hämtat: 24.09.2014]

[23] CLR [Online]
http://vb.net-informations.com/framework/common_language_runtime.htm [hämtat: 25.09.2014]

[24] Lecrenski, Nick. Lecrenski Stephen. & Asley, Kevin. (2012). *Professional Windows 8 Programming : Application Development with C# and XAML.* Wiley.

[25] Vice, Ryan och Siddiqi, Muhammad Shujaat. (2012). *MVVM Survival Guide for Enterprise Architectures in Silverlight and WPF.* Packt Publishing.

[26] MVC [Online]
http://msdn.microsoft.com/en-us/library/ff649643.aspx [hämtat: 13.10.2014]

[27] Vice, Ryan och Siddiqi, Muhammad Shujaat. (2012). *MVVM Survival Guide for Enterprise Architectures in Silverlight and WPF*. Packt Publishing.

[28] Shneiderman, Ben & Plaisant, Catherine. (2004). *Designing the User Interface: Strategies for Effective Human-Computer Interaction (4th Edition).* Addison Wesley.

[29] Johnson, Bruce. (2014). *Professional Visual Studio 2013.* Wrox.

[30] Notepad++ [Online]
http://notepad-plus-plus.org/ [hämtat: 03.09.2014]

Regler för mobiltelefonnummer

Finland	
Nummer att ta bort från prefix	0
Landsnummer	358
Längd på abonnentnummer	4 till 9
Mobil-prefix	04x, 0457, 050, 0500

Sverige	
Nummer att ta bort från prefix	0
Landsnummer	46
Längd på abonnentnummer	5 till 9
Mobil-prefix	070, 072, 073, 076

Norge	
Nummer att ta bort från prefix	
Landsnummer	47
Längd på abonnentnummer	8
Mobil-prefix	4, 59, 9

Estland	
Nummer att ta bort från prefix	
Landsnummer	372
Längd på abonnentnummer	7 till 8
Mobil-prefix	5x

Tyskland	
Nummer att ta bort från prefix	0
Landsnummer	49
Längd på abonnentnummer	10 till 11
Mobil-prefix	015x, 0152x, 0155x, 0157x, 0159x, 0160-0163, 017x

Litauen	
Nummer att ta bort från prefix	0 (förut 8)
Landsnummer	370
Längd på abonnentnummer	8
Mobil-prefix	06xx

Lettland	
Nummer att ta bort från prefix	
Landsnummer	371
Längd på abonnentnummer	8
Mobil-prefix	2xx

Ryssland	
Nummer att ta bort från prefix	8
Landsnummer	7
Längd på abonnentnummer	8 till 10
Mobil-prefix	890x, 891x, 892x, 893x, 8950, 8951, 8952,8953

Källor:

http://www.wtng.info/wtng-cod.html
http://www.onesimcard.com/how-to-dial/
http://en.wikipedia.org/wiki/Telephone_numbers_in_Finland
http://en.wikipedia.org/wiki/Telephone_numbers_in_Sweden
http://en.wikipedia.org/wiki/Telephone_numbers_in_Norway
http://en.wikipedia.org/wiki/Telephone_numbers_in_Estonia
http://en.wikipedia.org/wiki/Telephone_numbers_in_Germany
http://en.wikipedia.org/wiki/Telephone_numbers_in_Lithuania
http://en.wikipedia.org/wiki/Telephone_numbers_in_Latvia
http://en.wikipedia.org/wiki/Telephone_numbers_in_Russia
http://en.wikipedia.org/wiki/List_of_mobile_phone_number_series_by_country