СУ”Св. Климент Охридски”
Факултет по Математика и Информатика
Катедра”Информационни технологии”

ДИПЛОМНА РАБОТА

Тема: Система за мобилни информационни услуги, базирана на кратки съобщения

Дипломант: Мирослав Георгиев Господинов
Научен ръководител: доц. Васил Георгиев

София
2007г.

СЪДЪРЖАНИЕ

1.Въведение

Мобилните услуги се ползват с изключителна популярност сред потребителите на мобилните оператори. Съществуват различни видове услуги: гласови, кратки съобщения (SMS message), мултимедийни съобщения(MMS message), поточно видео (Streaming video) и т.н. Всички те имат своите специфични приложения. С голям интерес сред потребителите се ползват услугите свързани с кратки съобщения, даващи им възможност за участие в различни игри (радио, телевизионни и други) или да получат полезна информация (за пътен трафик, прогноза за времето, програма за телевизия и кино, информация за валутните курсове или за движението на пари по банковите им сметки и т.н.). Голяма част от информационните услуги се предоставят от доставчици на услуги с добавена стойност (VASP - value-added service provider), които се грижат за обработката и изпращането на необходимата информация.

1.1 Описание на задачата

Целта на разработката е реализирането на платформа за предоставянето на определен тип информационна услуга с помощта на кратки съобщения. От гледна точка на потребителя услугата предоставя възможност за изпращане на кратко съобщение на даден номер и получаване на определена информация в зависимост от текста на съобщението. Обработката на съобщението и определянето каква информация да се изпрати на потребителя се реализира от платформата. Примерно потребителят е заявил желание да получи прогнозата за времето и изпраща кратко съобщение на определен номер, платформата анализира изпратеното съобщение и връща отговор в зависимост какво е пратил потребителя. Ако потребителят заяви желание да използва услугата „получаване прогнозата за времето” и иска да получи информация за времето в град София, изпраща кратко съобщение със съдържание „София” и получава информация за времето в град София. Ако на следващия ден потребителят се намира в град

Велико Търново и изпрати кратко съобщение „Велико Търново” системата ще се погрижи да изпрати коректен отговор, тоест прогнозата за времето в град Велико Търново. Платформата претендира да бъде универсална и с минимални корекции може да реализира различен вид услуги, свързани например с изискване на информация за банкови преводи, за пътния трафик, астрологическа прогноза според зодиакалния знак и т.н. За реализирането на услугата се използва база от данни, уеб модул, базов модул и комуникационен модул за връзка с центъра за кратки съобщения (SMSC – short message service center) посредством протокола SMPP (short message peer- to-peer). В базата от данни се съхранява необходимата информация за услугата. Едно приложение от този тип се нуждае от периодично обновяване на информацията, която ще се предоставя. За улеснение на тази задача, се грижи уеб модулът, който се явява удобен интерфейс за промяната на информацията. Базовият модул реализира основната логиката на приложението свързана с анализирането на изпратеното кратко съобщение и определянето на връщания отговор от системата. Комуникационният модул е съществена част от услугата, защото той е отговорен за обмяната на информация с центъра за кратки съобщения (SMSC). Имплементира протокола SMPP съгласно стандарта.

1.2. Практическа полза от разработката и потенциални потребители

Системата за мобилни информационни услуги посредством кратки съобщения дава възможност за предоставянето на голям кръг от услуги на клиентите на мобилните оператори. Чрез мобилните си апарати те имат постоянен достъп до необходимата им информация. Платформата се стреми да задоволи непрекъснато нарастващите им потребности от мобилни услуги. Разработената библиотека за комуникация с центъра за кратки съобщения(SMSC) може да се използва самостоятелно от софтуерни разработчици и да осигури бързо и лесно реализиране на различни приложения изискващи взаимодействие с него.

Потенциални потребители на системата за мобилни информационни услуги посредством кратки съобщения са мобилните оператори и VASP(value-added

service provider). Те ще могат да предоставят на клиентите си разнообразни услуги като увеличат същевременно и приходите си.

1.3 Структура на дипломната работа

Дипломната работа се състои от 7 глави- въведение, преглед на Short Message Service, SMPP протокол, избор на технология за реализация, описание на реализация, ръководство за работа, заключение и възможности за бъдещо развитие.

Глава 1 – кратко представя задачите и целите проекта

Глава 2 – описание на Short Message Service(SMS), неговото развитие, използваемост както и функциите и разположението на центъра за кратки съобщения в мобилната мрежа

Глава 3 – описание на SMPP протокола, кратка история, приложение, реализации и функционалност.

Глава 4 – описание на избраните технологии за реализация на проекта

Глава 5 – описание на проектирането и реализацията на приложението. Разглеждане на класовете и основните методи.

Глава 6 – описание за работа с приложението от гледна точка на администратора и потребителя. Ръководство за инсталиране и конфигуриране на системата за предоставяне на мобилни информационни услуги посредством кратки съобщения.

Глава 7 – заключение и описание за възможностите за бъдещо развитие на системата.

Приложение – съдържа информация за използваните съкращения и термини в изложението.

Използвана литература – списък на използвана литература.

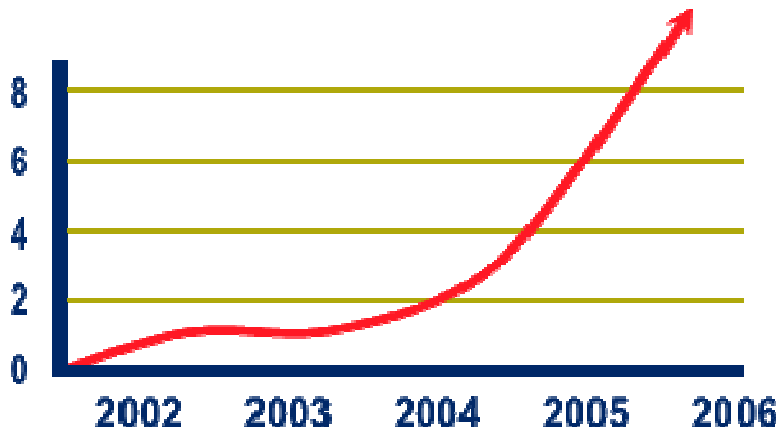
2. Преглед на Short message service

Най-популярната услуга в мобилните мрежи след гласовата е предаването на текстови съобщения. Те могат да бъдат от абонат до абонат , от абонат до приложение или от приложение до абонат. Всяко текстово съобщение е с големина 140 байта, като в зависимост от използваната кодова схема може да бъде съответно 160, 140 или 70 символа. SMS-ът е услуга от първата генерация на GSM стандарта. Тя е налична във всички мобилни телефони, което е и една от причините за нейната широка използваемост. За да се преодолеят някои от ограниченията на SMS е разработен EMS (Enhanced message service). Той дава възможност да се изпраща по-богат текст, монофонични мелодии и черно-бели картинки между EMS съвместими мобилни телефони. Следващата стъпка в развитието на SMS е MMS(Multimedia message service). MMS предоставя възможност за изпращане на комбинация от текст, картини, звук и видео. Ограничението на едно MMS съобщение е 300 килобайта. Тази услуга все още не е много популярна поради липсата й на поддръжка в някой по-стари апарати и относително високата цена на мултимедийните съобщения в мрежата на мобилните оператори. SMS съобщенията в мобилната мрежа се обслужват от центъра за кратки съобщения(SMSC). Принципът му на работа е съхрани и предай (store-and-forward). Когато абонатът е изключил телефона си, не е в обхват на мрежата или по някаква друга причина не може да получи своето съобщение, то се съхранява и му се доставя когато условията го позволяват.

2.1 История на Short Message Service

Идеята за наличие на услуга за предаване на текстови съобщения между потребителите на мобилни телефони се заражда още през 1985 година. Първото комерсиално кратко съобщение било изпратено от персонален компютър до мобилен телефон в британската мрежа на Vodafone на 3-ти декември 1992 година. Счита се, че първото съобщение, написано от мобилен телефон е изпратено от

Riki Pihonen инженер в Nokia през 1993 година [9]. Още в началото кратките съобщения намерили своето място чрез създаването на информационни системи за осведомяване на населението при критични ситуации и важни събития. Първоначалният растеж на текстовите съобщения бил бавен. През 1995 година се падали средно 0,4 съобщения на абонат за месец. Днес изпращането на кратки съобщения е една от най-разпространените мобилни услуги, като 72% от всички потребители на мобилни телефони или 1,9 милиарда от 2,7 милиарда абоната в края на 2006 година ползват активно SMS. В страни като Финландия, Швеция и Норвегия над 90% от населението използва кратки съобщения. Най-голяма средна използваемост на текстови съобщения от абонатите на мобилните оператори е във Филипините, където се падат по 15 съобщения на абонат на ден. Следната графика показва тенденцията в растежа на използваемост на SMS.



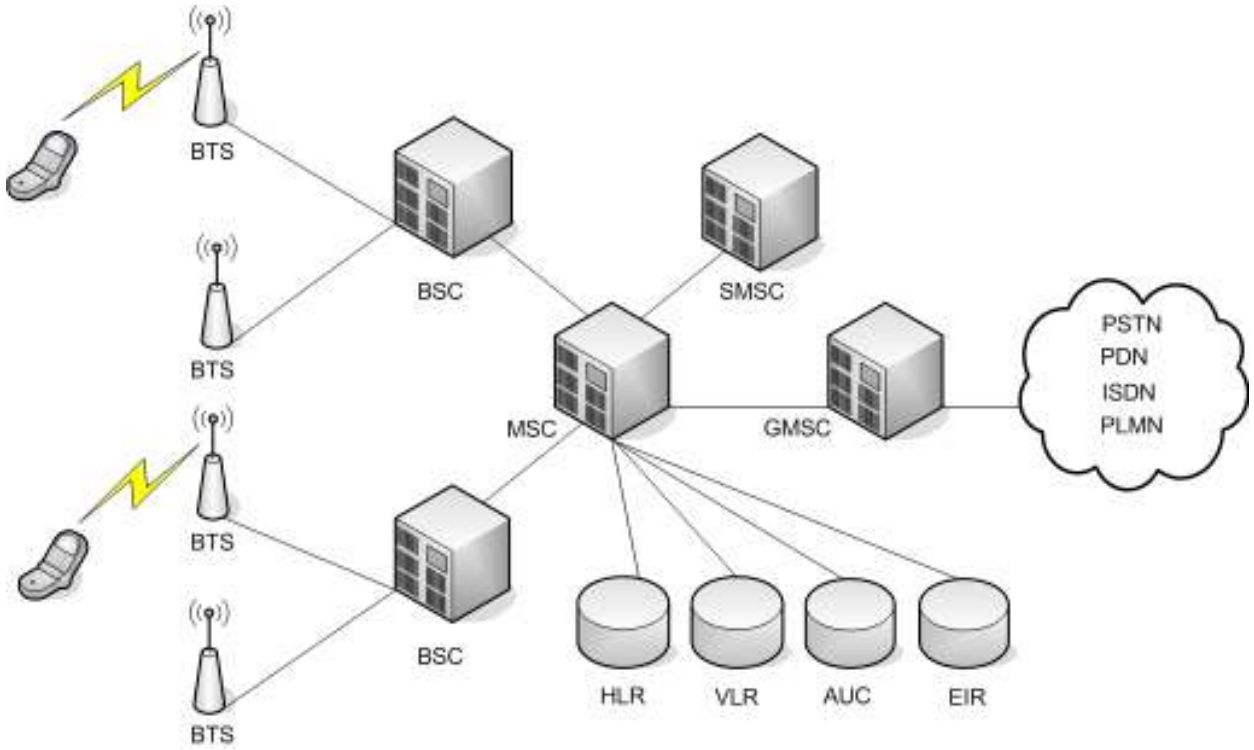
Фиг.1 Брой Текстови съобщения за месец в милиарди

Мобилната услуга SMS първоначално била проектирана за GSM мрежи. Сега тя е налична в широк кръг от мрежи включително и UMTS. Приходите от текстови съобщения в глобален мащаб за 2006 година надминават 80 милиарда долара. Единствено бъдещето ще покаже дали кратките съобщения ще продължат своя възход или ще бъдат изместени от мултимедийните съобщения .

2.2 Структура на GSM мрежа

Глобалната система за мобилни комуникации GSM (Global system for mobile communication) се характеризира със спецификации и препоръки (квазистандарт) за отделни функционални единици на цялостната мобилна система и дефиниции на най-важните особености [6]. При проектиране на GSM са били поставени следните главни цели:

- подпомагане мобилността на абоната и международния роуминг;
- интеграция в цифровата мрежа с интеграция на услугите ISDN;
- голям капацитет;
- качество на преноса на говорни сигнали, съизмеримо със съществуващите аналогови системи;
- защита от подслушване.



Фиг. 2 Структура на GSM мрежа

На най-ниско ниво са мобилните станции MS (mobile station), които се явяват крайните устройства на мобилните абонати. Базовата станция се състои от два функционални блока – управляващо устройство BSC (Base Station Controller) и базова приемо-предавателна станция BTS(Base Transceiver Station) [5]. Когато клетките са малки е целесъобразно да се използва един контролер на базови станции за управление на няколко базови станции. По такъв начин възниква подсистема от базови станции BSS (Base Station Subsystem). На най-високо ниво е радиотелефонната централа MSC(Mobile Switching Center), която осигурява свързващи функции за определена група базови станции. В цялата мрежа може да има няколко такива централи. Съставна част от мобилната мрежа са базите данни за абонатите. Домашната база данни HLR (Home Location Register) е част от мобилната мрежа, която съдържа данни за абонамента и друга информация за всички абонати на мрежата [7]. Тя може да поддържа данни за стотици хиляди абонати. В HLR се поддържат специфични за абонатите параметри като:

- идентификационен номер на абонат;
- абонатния профил;
- информация за мобилната централа, която обслужва абоната.

Всеки абонат е регистриран в една HLR, която действа като фиксирана точка, съхраняваща информация за абоната. За да се намали натоварването на HLR, е въведена база данни VLR(Visitor Location Register), която подпомага HLR при управление на множеството запитвания, свързани с абонатите (например локализиране на абонат). Тя съхранява временна информация за абонатите предвижващи се през мрежата. Това следене е важно, защото по всяко време мрежата трябва да знае къде е всеки абонат, за да му предоставя обслужване. Друга база данни е центъра за проверка на автентичността AUC (AUthentication Center), който съдържа за всеки абонат по един секретен ключ. Същият ключ е записан и в SIM картата на потребителя. Регистърът за идентифициране на (абонатните) съоръжения EIR(Equipment Identification Register) съдържа информация за абонатните мобилни устройства, като поддържа например черен списък с номера на откраднати устройства. Центърът за кратки съобщения е

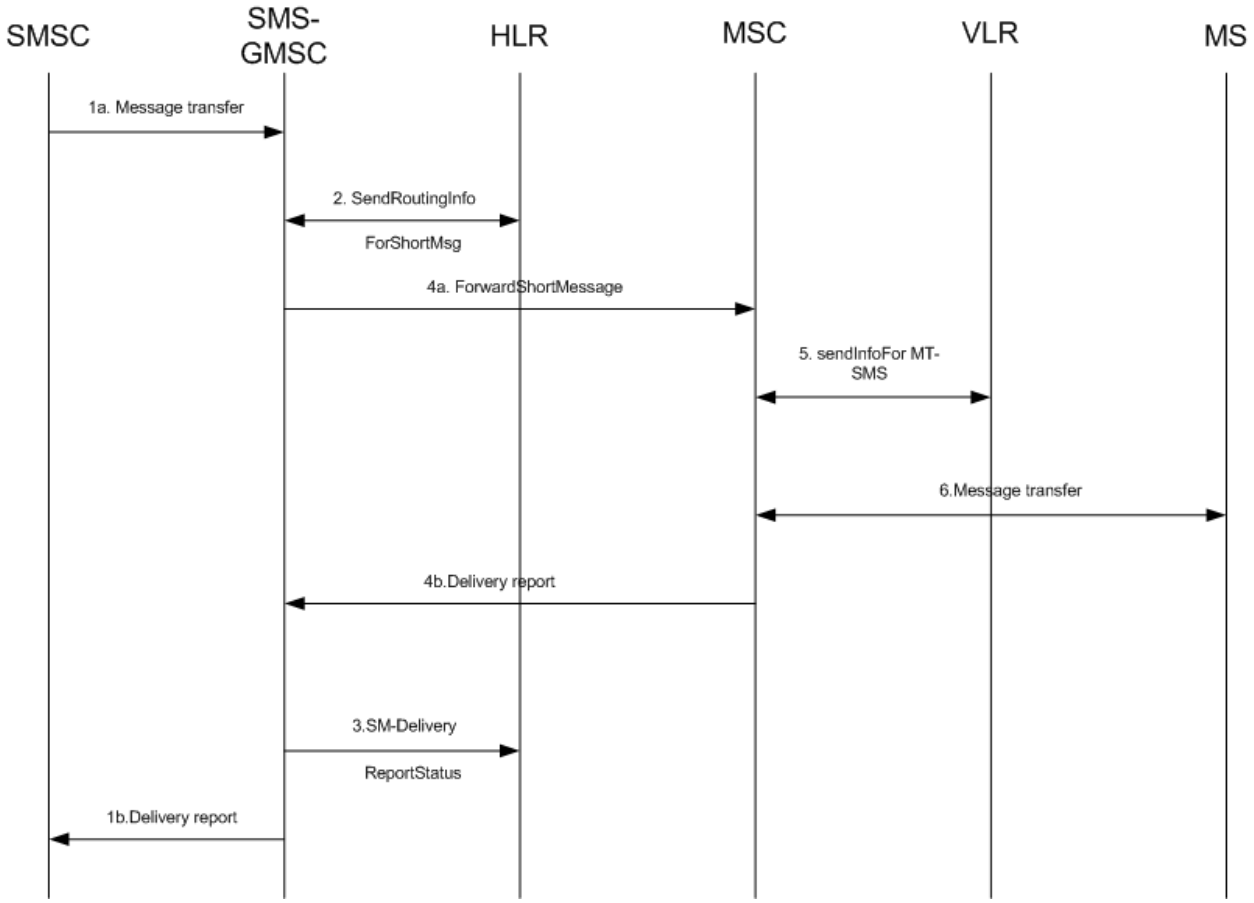
свързан с мобилната радиоцентрала (MSC). Той действа като централа за текстовите съобщения. Изпратеното от абоната кратко съобщение се прехвърля от мрежата до SMSC на неговия оператор. Там от съобщението се извлича телефонния номер на получателя и съобщението се препраща. SMSC не се причислява към GSM стандарта, но централите на мобилната мрежа трябва да участват в прехвърлянето на съобщения. За услугите SMS се използва специална функционалност на MSC. Функционалността на интерфейс за кратки съобщения от центъра за кратки съобщения към мобилен телефон се изпълнява от MSC-шлюз за SMS (SMS-GMSC). Маршрутизирането на кратки съобщения в обратна посока (от мобилния телефон към SMSC) става от MSC за взаимодействие с SMS (SMS-IWMSC). Фиг.3 илюстрира задачите на двата вида MSC.

Фиг.3 Задача на MSC-шлюз за SMS и MSC за взаимодействие с SMS

2.3 Разглеждане потока от кратки съобщения в мобилната мрежа

След като разгледахме основната структура на GSM мрежата и разположението на центъра за кратки съобщения в нея, ще се запознаем и с потока на кратки съобщения от него до мобилен телефон и обратно. Прието е съобщение изпратено от центъра за кратки съобщения до мобилно устройство да се нарича MT(Mobile terminated) съобщение, а съобщение изпратено от мобилно устройство до центъра за кратки съобщения да се нарича MO(Mobile originated)

съобщение [4]. На фиг.4 и фиг.5 са показани съответно MT и MO поток на кратко съобщение при отсъствие на грешки.



Фиг.4 MT поток на съобщението от SMSC до мобилен терминал

1) Трансфер на съобщението от SMSC до SMS-GMSC

2) SMSC прави заявка и получава необходимата информация от HLR за изпращане на съобщението

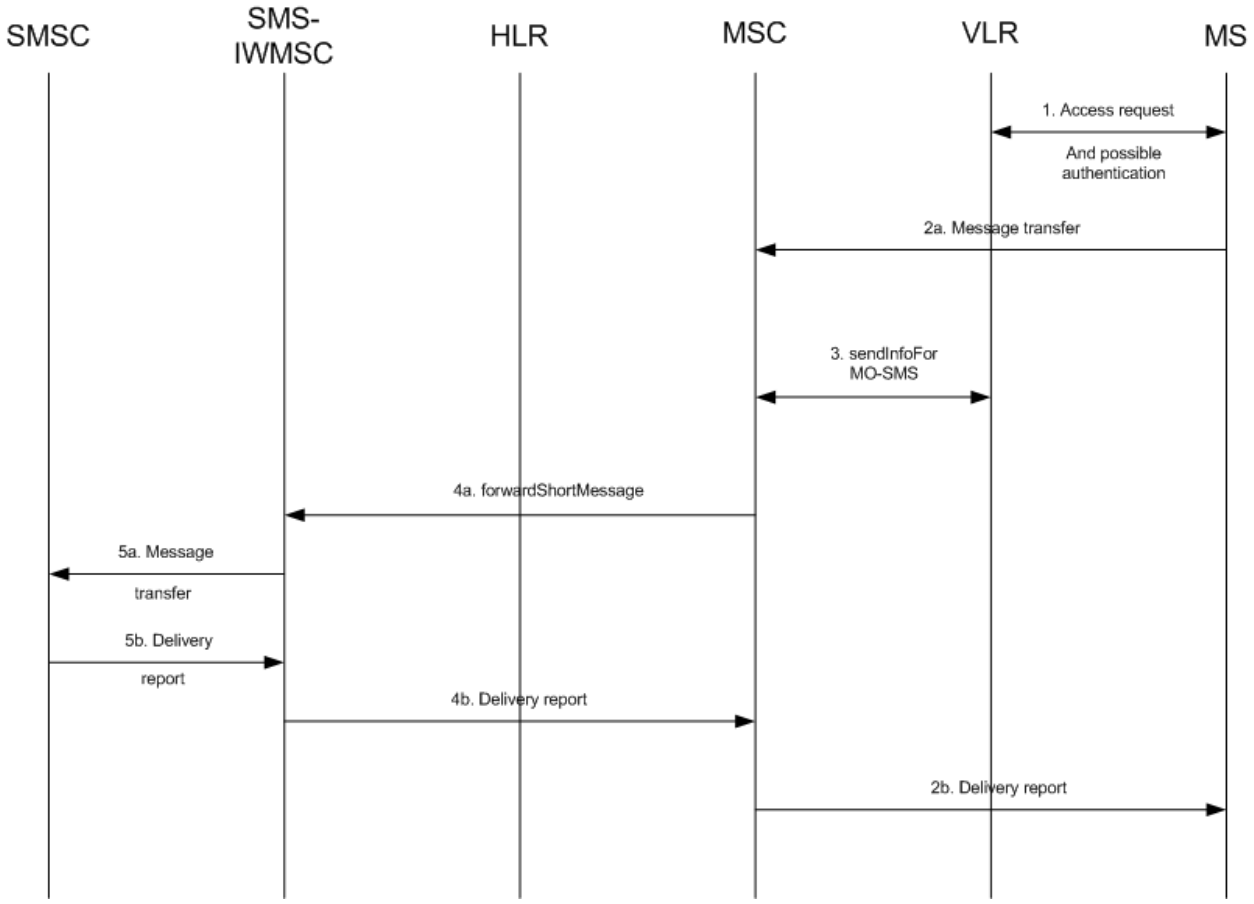
3) SMSC добавя на параметри в HLR

4) SMSC изпраща съобщението до MSC използвайки forward short message операцията

5) MSC изискване информация за абоната от VLR

6) MSC доставя съобщението до абоната

Всяка операция връща информация дали е изпълнена успешно и/ или параметри



Фиг.5 MO поток на съобщение от мобилен терминал до SMSC.

1) Изискване на достъп до мобилната мрежа от MS
2) MS предава съобщението до MSC
3) MSC изисква на необходимата информация от VLR
4) MSC изпраща съобщението до SMS-IWMSC използвайки forward short message операцията
5) Доставка на съобщението до SMSC

Всяка операция връща информация дали е изпълнена успешно и/ или параметри.

3. SMPP протокол

Short Message Peer to Peer (SMPP) е отворен, стандартизиран протокол, който осигурява гъвкав комуникационен интерфейс за обмяна на кратки съобщения между SMSC и SMS приложения [1]. Той работи на приложното ниво на OSI модела. Не разполага с никакви средства за контрол на грешките при комуникация и разчита за тази задача изцяло на протоколите от долните слоеве. Прието е външни приложения, които се свързват с центъра за кратки съобщения с цел получаване и/ или изпращане на съобщения да се наричат ESME(External Short Message Entity) [2]. ESME могат също да отправят запитване, да изтриват или заменят кратко съобщение посредством SMPP. Протоколът е базиран на двойка заявка-отговор(request-response) от PDU(Protocol data units) пакети. PDU са кодирани двоично за по-голяма ефективност.

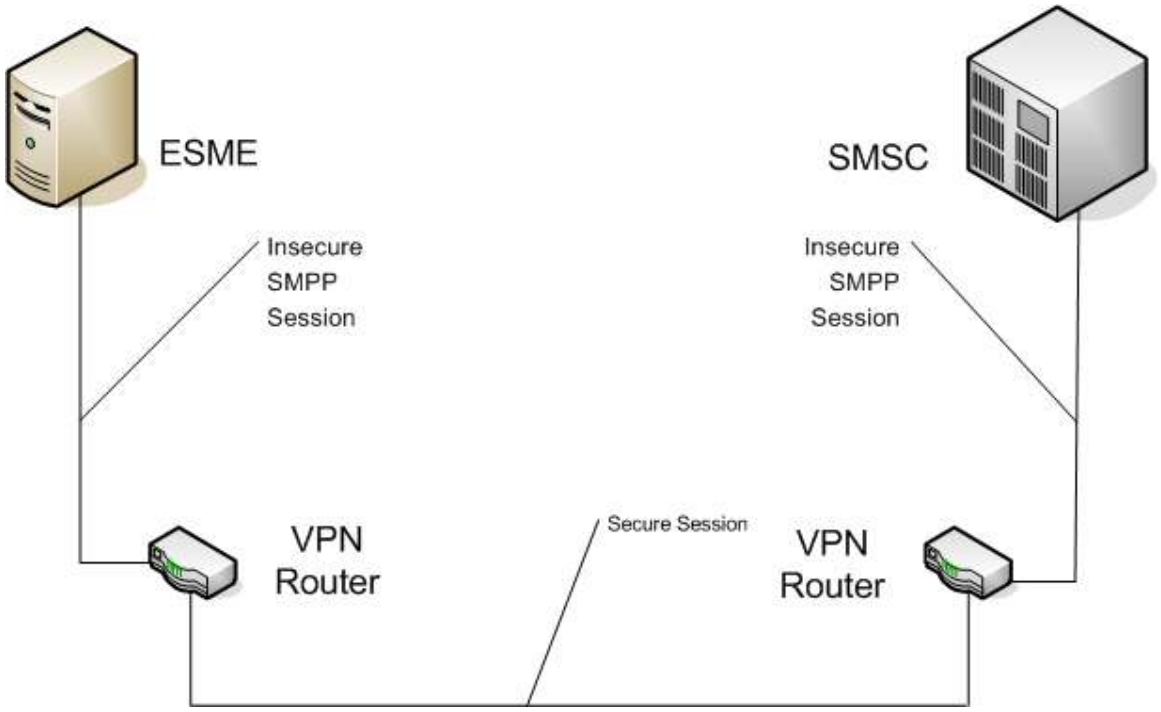
3.1 История на SMPP

SMPP е бил проектиран първоначално от Aldiscon, малка ирландска компания, която по-късно била купена от Logica. Logica придобила и CMG, основният конкурент на Aldiscon, и сега присъства на пазара под името LogicaCMG. През 1999 година тя официално сложила ръка над SMPP със създаването на SMPP Developers Forum, по-късно преименуван на SMS Forum, който се грижи за стандартизирането и развитието на протокола. Най-широко използваната версия на SMPP е 3.3, след нея се появява версия 3.4, а последната е 5.0.

3.2 Свързване на ESME към SMSC

За осъществяване на SMPP сесия между ESME и центъра за кратки съобщения първо трябва да има налична TCP/IP или X.25 мрежова свързаност. SMSC „слуша” за конекции на един или няколко TCP/IP порта или X.25

интерфейса. Обикновено обмяната на съобщения с ESME се осъществява през Internet или друг вид несигурни податливи на атаки мрежи. Това е причината на мрежовата сигурност да се обърне сериозно внимание. Евентуален пробив в комуникацията би нанесъл огромни поражения не само в материален, но и в морален аспект както на мобилния оператор така и на доставчика на информационни услуги. За да се избегне това в практиката се използват наети линии или се изграждат IPSec тунели между отделните страни в комуникацията. На фиг. 6 е илюстриран най-често използвания модел на мрежова свързаност между ESME и SMSC.



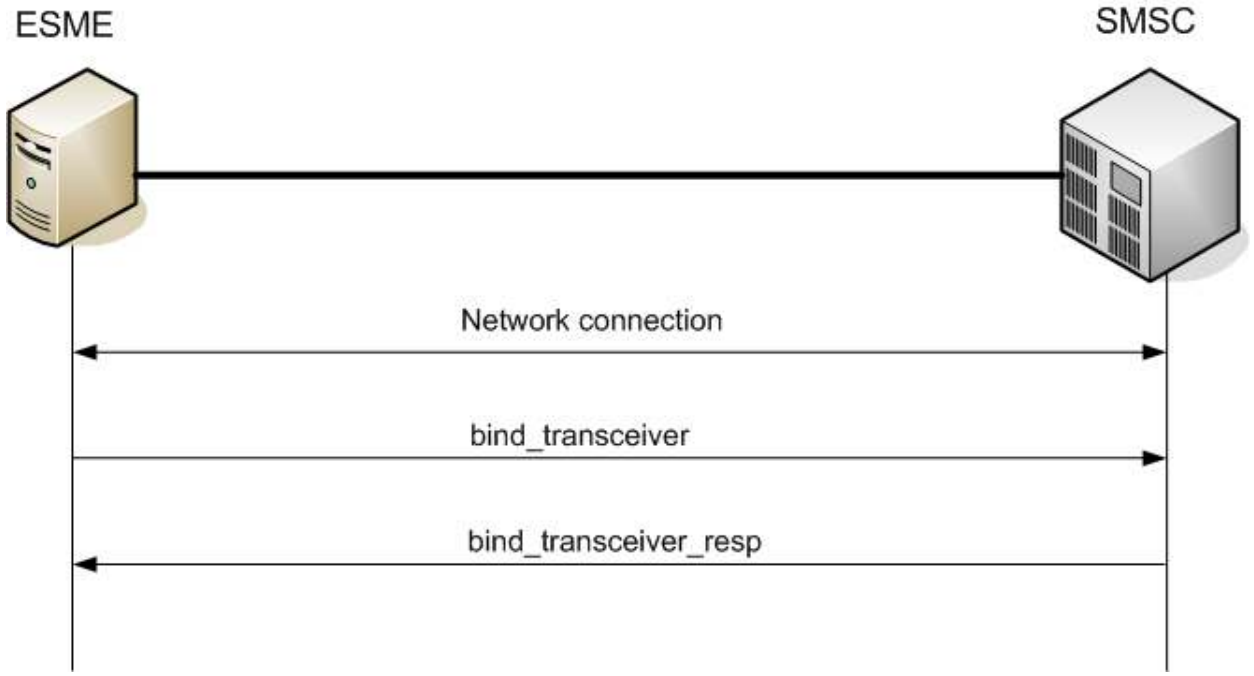
Фиг.6 Връзка между EMSE и SMSC

Обикновено SMPP сесията се инициира от ESME. Различаване следните три типа сесии:

- Transmitter(TX) – когато приложението се свърже като Transmitter , то може да предава кратки съобщения до SMSC, който има грижата да ги достави до абоната. Този вид сесия позволява на ESME да изтрива, да прави запитване или да заменя кратко съобщение;

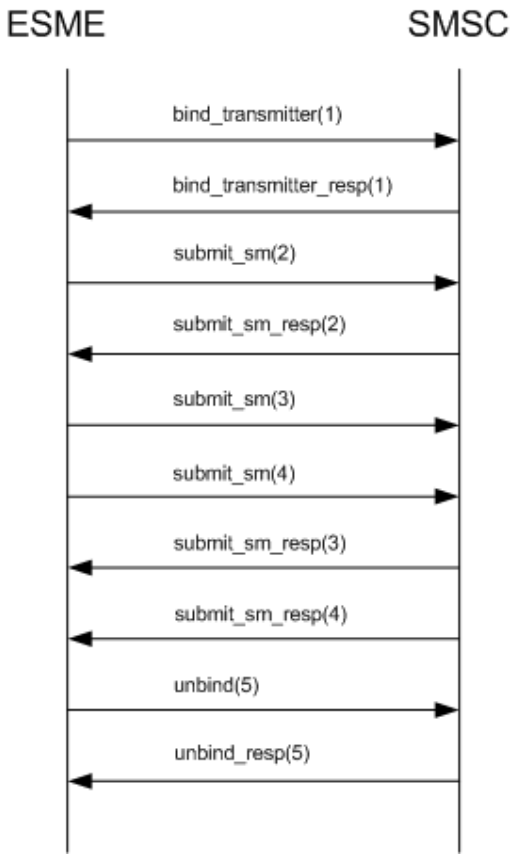
- Receiver(RX) – при тази сесия приложението може само да получава съобщения от центъра за кратки съобщения;
- Transceiver(TRX) – това е сесия, която се явява комбинация от Transmitter(TX) и Receiver(RX), като така едновременно в една сесия могат да се предават и получават съобщения.

В SMPP версия 3.3 отсъства възможност за създаване на Transceiver връзка. За да може да се получават и изпращат съобщения едновременно ESME трябва да осъществи две връзки една Transmitter и една Receiver. Прието, е когато SMSC инициира сесия с ESME това да се нарича Outbind сесия. Тъй като е възможно да възникнат мрежови проблеми по време на комуникацията е задължително приложението да следи за това и при разпадане на връзката да се опита отново да се свърже с центъра за кратки съобщения. Всички мерки за предоставяне на безотказна услуга трябва да бъдат взети тъй като обикновено ESME работят в режим 24/7 и при минимален отказ. Фиг.7 показва примерно осъществяване на Transceiver(TRX) връзка с SMSC.



Фиг.7 Осъществяване на Transceiver свързаност

Комуникацията по SMPP се основана на обмен на пакети заявки и пакети, съдържащи отговор на тези заявки. Тя може да бъде както синхронна така и асинхронна. Това означава, че ESME може да изпрати няколко заявки до SMSC без синхронно да изчаква отговора за всяка от тях. Фиг.8 показва една типична асинхронна SMPP Transmitter сесия. Transmitter, Receiver и Transceiver сесиите са напълно аналогични като разликата е само във вида на изпращаните PDU пакети.



Фиг.8 SMPP Transmitter сесия

3.3 SMPP операции и PDUs

SMPP е протокол основно базиран на множество от операции, всяка от които е под формата на пакет заявка (request PDU) и на пакет отговор (response PDU). Например ако ESME иска да изпрати кратко съобщение, той може да изпрати submit_sm пакет до центъра за кратки съобщения. Центърът ще отговори

с submit_sm_resp пакет, който дава информация дали операцията е била успешна или не. Ако SMSC желае да достави съобщение до ESME, той може да изпрати deliver_sm пакет(PDU) до ESME, който ще отговори с deliver_sm_resp като потвърждение за получаването. Някои от SMPP операциите са специфични за ESME, други за центъра за кратки съобщения, а трети зависят от типа на установената сесия. ESME може да изпраща deliver_sm само ако има Receiver(RX) или Transceiver(TRX) сесия. SMPP операциите могат да бъдат категоризирани в няколко групи:

- Session Management операции – тези операции са отговорни за осъществяването на SMPP сесия между ESME и SMSC;
- Message Submission операции – тези операции са отговорни за изпращането на кратки съобщения от ESME до SMSC;
- Message Delivery операции - тези операции са отговорни за получаване на съобщения от ESME;
- Message Broadcast операции - тези операции са отговорни за cell broadcast услуги на SMSC;
- Ancillary операции – те дават възможност за запитване, изтриване или замяна на кратко съобщение в SMSC.

Session Management операции

SMPP PDU	Описание
bind_transmitter	Използва се за осъществяване на transmitter SMPP сесия. PDU съдържа идентификафионен номер и парола
bind_transmitter_resp	PDU отговор на bind_transmitter. Съдържа информация за успешно или не свързване като transmitter.
bind_receiver	Използва се за осъществяване на receiver SMPP сесия. Съдържа идентификационна информация, парола и допълнително е възможно да съдържа и информация за обслужваните от ESME адреси.
bind_receiver_resp	PDU отговор на bind_receiver. Съдържа информация за успешно или не свързване като receiver.
bind_transceiver	Използва се за осъществяване на transceiver SMPP сесия. Съдържа

	идентификационна информация, парола и допълнително е възможно да съдържа и информация за обслужваните от ESME адреси.
bind_transceiver_resp	PDU отговор на bind_transceiver. Съдържа информация за успешно или не свързване като transceiver.
Outbind	Използва се от SMSC за да осъществи outbind сесия с ESME. В отговор на това ESME изпраща bind_reciver, bind_trasmitter.
Unbind	Използва се за прекратяване на SMPP сесия. Може да бъде изпратено от SMSC или ESME.
unbind_resp	PDU, който може да бъде изпратен от ESME или SMSC. Съдържа информация за потвърждаване на заявката за прекъсване на SMPP сесия.
enquire_link	Изпраща се от SMSC или ESME през определен интервал от време за тестване на мрежовата връзка. Получаващата страна трябва да изпрати потвърждение за да успешен теста.
enquire_link_resp	Използва се за потвърждаване на enquire_link заявка изпратена от SMSC или ESME.
alert_notification	Използва се от SMSC за да уведоми ESME ,че мобилен абонат е станал активен, а преди това е бил неактивен.
general_nack	Това PDU, може да бъде изпратено от ESME или SMSC за да уведоми за получаването на невалиден пакет. Получателя на general_ack бива информиран ,че отсрещната страна не разпознава PDU поради сгрешен размер или съдържание.

Message Submission операции

SMPP PDU	Описание
submit_sm	Когато ESME е установил transmitter или transceiver сесия с центъра за кратки съобщения , той използва този PDU за да изпрати съобщение като в него се специфицира изпращач, получател и текст на съобщението. Други параметри са приоритет, кодова схема , период на валидност.
submit_sm_resp	PDU , което се връща от SMSC в отговор на submit_sm. Съдържа информация за успешна или не операция. Като параметър се връща message_id, който може да се използва за изтриване,

	запитване или замяна на кратко съобщение.
submit_muti	Вариант на submit_sm PDU, който поддържа изпращане на съобщения до 255 абоната
submit_multi_resp	PDU ,което се връща от SMSC в отговор на submit_multi. Той е подобен на submit_sm_resp. Основната разлика е , че някои от получателите са отхвърлени от SMSC, PDU специфицира списък с получателите като за всеки отхвърлен получател има грешка описваща причините за това. Включен е и message_id, който може да се използва за изтриване , запитване или замяна на кратко съобщение.
data_sm	Явява се опростена версия на submit_sm, проектиран е за приложения , които не изискват разширената функционалност на submit_sm. ESME имплементиращи WAP over SMS използват тази операция.
data_sm_resp	PDU , което се връща от SMSC. Съдържа информация дали е успешна или не операцията.

Message Delivery операции

SMPP PDU	Описание
deliver_sm	Deliver_sm е семантично противоположна операция на submit_sm. Използва се за доставяне на съобщение до ESME
deliver_sm_resp	Съдържа информация дали успешно е прието или отхвърлено съобщението от ESME
data_sm	Може да се използва за доставяне на съобщение от SMSC до ESME. ESME обикновено реализира WAP over SMS използвайки тази операция
data_sm_resp	PDU, което се връща от ESME. Съдържа информация дали е успешна или не операцията

Message Broadcast операции

SMPP PDU	Описание
broadcast_sm	Използва се за изпращане на broadcast съобщение от страна на ESME

broadcast_sm_resp	PDU, което се връща от SMSC. Съдържа информация дали е успешна или не операцията

Ancillary операции

SMPP PDU	Описание
cancel_sm	Използва се за изтриване на предишно изпратено съобщение до SMSC. Съдържа source address и message_id върнато от submit_sm_resp, submit_sm_multi_resp, data_sm_resp
cancel_sm_resp	PDU, което се връща от SMSC. Съдържа информация дали е успешна или не операцията.
query_sm	Използва се за запитване за статуса на предишно изпратено съобщение. Съдържа source address и message_id върнат от submit_sm_resp, submit_sm_multi_resp, data_sm_resp
query_sm_resp	PDU, което се връща от SMSC. Съдържа информация дали операцията е успешна или не.
replace_sm	Използва се от ESME с параметър message_id за заместване на предишно изпратено съобщение. С използването на други полета е възможно да се промени текст, валидност и други атрибути на съобщението.
replace_sm_resp	PDU, което се връща от SMSC. Съдържа информация дали е успешна или не операцията.
cancel_broadcast_sm	Използва се за изтириване на придищно изпратено broadcast съобщение.
cancel_broadcast_sm_ resp	PDU, което се връща като отговор от SMSC. Съдържа информация дали е успешна или не операцията.
query_broadcast_sm	Използва се за запитване на статуса на предишно изпратено broadcast съобщение. Съдържа source address и message_id върнат от broadcast_sm_resp.
query_broadcast_sm_ resp	PDU, което се връща от SMSC. Съдържа информация дали операцията е успешна или не. За успешните съдържа информация включваща текущия статус на съобщението.

3.4 Формат на PDU

SMPP PDU се състои от хедър (header) и тяло. Хедърът е задължителна част за всеки пакет и трябва да присъства. Тялото може и да не бъде включено. Възможно е да гледаме на пакета по два начина: като 16 байта хедър плюс байтове за тяло или като 4 байта command_lenght плюс байтове за останалата част на пакета [3]. Фиг.9 илюстрира формата на SMPP PDU пакет.

<table>
<tr><td colspan="5">SMPP PDU</td></tr>
<tr><td colspan="4">PDU Header(mandatory)</td><td>PDU Body(Optional)</td></tr>
<tr><td>Command lenght</td><td>Command id</td><td>Command Status</td><td>Sequence number</td><td>PDU Body</td></tr>
<tr><td>4 octets</td><td>4 octets</td><td>4 octets</td><td>4 octets</td><td>Length=(Command Length value -16) octets</td></tr>
<tr><td>4 octets</td><td colspan="4">Command Length - 4</td></tr>
</table>

Фиг.9 Формат на PDU

Command_lenght – представя целият размер на PDU включително хедър и тяло. Причината за наличието на полето command_lenght, е че SMPP е двоичен протокол поддържащ асинхронна комуникация, което означава ,че ESME или SMSC трябва да поддържа възможност за декодиране на PDU намиращ се в мрежовия буфер заедно с други пакети. Ключов елемент в определянето на PDU е неговата дължина. Приложението прочита първите 4 байта, след това може да извлече и обработи целия пакет.

Command_id – определя коя SMPP операция се използва. Ние наричаме тези операции с имена submit_sm, bind_transmitter , но в пакета те са представени чрез числа. Command_id-тата, които се използват за пакети заявки(request PDU) се намират в интервала 0x00000000 – 0x000001FF, а тези за пакети отговор(response PDU) се намират в интервала 0x80000000 – 0x800001FF. Например replace_sm

има command_id=0x00000007, а съответният му отговор replace_sm_resp има command_id=0x80000007.

Command_status – представя кодът на грешка, който се връща от SMSC или ESME. Това поле има значение само за пакетите отговор, в пакетите заявка има стойност 0x00000000.

Sequence number – уникално идентифицира всеки пакет в SMPP сесия.

Тялото може да не присъства в PDU. Пример за такъв пакет е enquire_link. В тялото се съдържат задължителни и опционални параметри , които зависят от command_id на пакета. Опционалните параметри дават възможност за бъдещо развитие на протокола с добавянето на нови параметри. Те се намират винаги в края на пакета, като наредбата им не от значение. За отделен PDU, SMSC и ESME могат да решат, кои параметри да се използват и кои не. Всички опционални параметри се дефинират в TLV(Tag, Length, Value) формат.

Parameter Name	Size	Type	Description
Tag	2	Integer	Използва се уникално да идентифицира опционален параметър. Винаги е с дължина два байта.
Length	2	Integer	Показва дължината на полето Value в байтове. Стойността на това поле не включва в себе си дължината на Tag и Length полетата. Винаги е с дължина два байта.
Value	Variable	variable	Съдържа данни за опционалният параметър

По нататък ще разгледаме списък от задължителните и опционални параметри. За дефинирането на SMPP задължителните и опционални параметри се използват следните типове данни:

Integer – Положителна стойност за дефинирането на броя на байтовете. В терминологията на SMPP е прието байтовете да се наричат octets. При предаването на octets старшия бит се предава пръв(Big Endian кодиране);

C-Octet String – Поредица от ASCII символи завършващи с NULL символ. Текстът “Hello” може да бъде кодиран в 6 octets(5 символа от „Hello” и един octet за NULL) както следва 0x48656C6C6F00;

C-Octet String (Decimal) – Поредица от ASCII символи, всеки символ представлява десетична цифра(0-9). Използва се когато десетични числа са представени като символен низ, кодира се като всяка ASCII последователност от символи. Поредицата от символи завършва с NULL символ. Следният C-Octet String(Decimal) „123456789” ще бъде кодиран както следва „0x31323334353637383900”;

C-Octet String (Hex) – Поредица от ASCII символи, всеки символ представлява шестнадесетична цифра(0-F). Използва се когато шестнадесетичните числа са представени като символен низ, кодира се като всяка ASCII последователност от символи. Поредицата от символи завършва с NULL символ. Следният C-Octet String(Hex) “A2F5ED278FC” ще бъде кодиран както следва „0x413246354544323738464300" ;

Octet String – Поредица от octets , не е необходимо да завършва с NULL символ.

Нека разгледаме примерен SMPP PDU и как може да бъде декодиран.

Пример: SMPP PDU (Стойностите са в шестнадесетичен формат):

```
00 00 00 2F 00 00 00 02 00 00 00 00 00 00 00 01
53 4D 50 50 33 54 45 53 54 00 73 65 63 72 65 74
30 38 00 53 55 42 4D 49 54 31 00 50 01 01 00
```

Шестнадесет байтовият хедър (header) се декодира както следва:

```
00 00 00 2F  Command Length 0x0000002F
00 00 00 02  Command ID 0x00000002 (bind_transmitter)
00 00 00 00  Command Status 0x00000000
00 00 00 01  Sequence Number 0x00000001
```

Останалата част съставлява тялото на пакета

```
53 4D 50 50 33 54 45 53 54 00  System_id (“SMPP3TEST”)
73 65 63 72 65 74 30 38 00     System_type (“secret08”)
53 55 42 4D 49 54 31 00        Password (“SUBMIT1”)
50                             Interface_version (0x50 “V5.0 compliant”)
01                             addr_ton (0x01)
01                             addr_npi (0x01)
```

00	addr_range (NULL)

3.5 SMPP параметри

Ще разгледаме някои от често използваните задължителни и опционални параметри намиращи се в тялото на PDU. Подробна информация за всички параметри използвани от протокола може да бъде намерена в спецификацията на SMPP.

- Service Type – този параметър се използва за да посочи асоциацията на кратко съобщение с SMS приложение. Може да бъде полезен на ESME като му позволява да заменя група от съобщения с един и същ service_type. В таблицата са посочени възможните стойности на параметъра.

Service Type	Описание
NULL	Default
CMT	Cellular Messaging
CPT	Cellular Paging
VMN	Voce Mail Notification
VMA	Voice Mail Alerting
WAP	WirelessAplication Protocol
USSD	Unstructured Supplementary Services Data
CBS	Cell Broadcast Service
GUTP	General UDP Transport Service

-TON(Type of Number): addr_ton, source_addr_ton, dest_addr_ton, esme_addr_ton. Тези параметри се използват към адреса(номер) на абонат или приложението. Възможните стойности са посочени в следната таблица.

TON	Стойност
Unknown	00000000
International	00000001

National	00000010
Network Specific	00000011
Subscriber number	00000100
Alphanumeric	00000101
Abbreviated	00000110
Reserved	

-NPI(Numeric Plan Indicator): addr_npi, source_addr_npi, dest_addr_npi, esme_addr_npi. Тези параметри определят NPI и се използват към адреса(номера) на абоната или приложението. Възможните стойности са посочени в следната таблица.

NPI	Стойност
Unknown	00000000
ISDN(E163/E164)	00000001
Data(X.121)	00000011
Telex(F.69)	00000100
Land Mobile(E.212)	00000110
National	00001000
Private	00001001
ERMES	00001010
Internet(IP)	00001110
WAP Client id	00010010
Reserved	

- esm_class – използва се задаването на някои специални атрибути на краткото съобщение. Присъства в submit_sm, submit_multi и data_sm. Размерът на този параметър е един байт. Може да се зададе типът на съобщението, режимът на предаване, както и информация за дълги съобщения.

- PID(protocol id) – този параметър се задава в зависимост от използваната мрежа(GSM,CDMA,TDMA). За GSM мрежа се определя съгласно стандарта.
- Priority – този параметър позволява на изпращащата страна да зададе приоритет на съобщението. За различните видове мрежи този параметър има различно значение. Следната таблица илюстрира какво значение имат различните стойности на параметъра в различните мрежи.

Priority	GSM(SMS)	GSM(CBS)	ANSI-163	ANSI-41(CBS)
0	Non-priority	Normal	Bulk	Normal
1	Priority	Immediate	Normal	Interactive
2	Priority	High priority	Urgent	Urgent
3	Priority	Reserved	Emergency	Emergency
4	N/A	Priority Background	N/A	N/A

За GSM мрежи всяка стойност по-голяма от нула оказва, че съобщението е с приоритет

- Scheduled_delivery_time – този параметър позволява да се изпрати отложено кратко съобщение. Може да се посочи точното време, когато да се направи първия опит за доставката на съобщението от центъра за кратки съобщения. Съществуват два начина за задаване на времето чрез абсолютен формат на времето и относително формат на времето. Абсолютният формат на времето се използва по подразбиране от SMPP и представлява 16 символен низ „YYMMDDhhmmsstnnp” където:

‘YY’	Последните две цифри от годината(00-99)
‘MM’	Месец(01-12)
‘DD’	Ден(01-31)
‘hh’	Час(00-23)
‘mm’	Минути(00-59)
‘ss’	Секунди(00-59)

‘t’	Десети от секундата(0-9)
‘nn’	Определяне разликата в четвърт час между местно време и UTC
‘p’	“+” , когато местното време е напред от UTC “-“ , когато местното време е назад от UTC

Относителното време е спрямо текущото време на SMSC. Форматът се представя чрез 16 символен низ „YYMMDDhhmmsstnnp”. Означенията са същите както при абсолютния формат на времето, но атрибута ‘t’ не трябва да се използва и му се задава стойност 0, ‘nn’ също трябва да има стойност 00 , а ‘p’ трябва да има стойност ‘R’.

- validity period – този параметър показва времето след , което дадено съобщение ще бъде изтрито от SMSC ако не бъде доставено. Може да се зададе в абсолютен формат на времето или относителен формат на времето

- DCS(Data coding scheme) – задава кодовата схема на краткото съобщение. Възможните стойности са показани в следната таблица.

Стойност	Описание
00000000	The default alphabet
00000001	IA5(CCITT T.50)/ASCII
00000010	Octet unspecified
00000011	Latin 1(ISO-8859-1)
00000100	Octet unspecified
00000101	JIS(X 0208-1990)
00000110	Cyrillic(ISO-8859-5)
00000111	Latin/Hebrew(ISO-8859-8)
00001000	UCS2(ISO/IEC-10646)
00001001	Pictogram Encoding
00001010	ISO-2022-JP(Music Codes)

00001011	Reserved
00001101	Reserved
00001101	Extended Kaji JIS
00001111 …….. 10111111	Reserved
1100xxxx	GSM MWI control
1101xxxx	GSM MWI control
1110xxxx	Reserved
1111xxxx	GSM Class Control

Най-често използвани са кодовата схема по подразбиране(00000000) и UCS2 (00001000) даващ възможност за кодиране на всякакви символи.

- register_delivery – използва се за изискване на потвърждение от SMSC дали е доставено съобщението или не

- replace_if_present – използва се за замяна на предишно изпратено съобщение до центъра за кратки съобщения, което не е доставено. Може да приема две стойности

Стойност	Описание
0	Не заменя съобщение
1	Заменя съобщение
2-255	Reserved

3.6 Съществуващи SMPP реализации

Съществуват различни библиотеки имплементиращи протокола SMPP. Някои от тях са платени , а други могат да се използват свободно. Голяма част от свободно разпространяваните библиотеки са за операционната система Linux, което се явява пречка за програмиращите под Windows. Една част от

библиотеките не реализират пълната функционалност на протокола, което е проблем при създаването на гъвкави и надеждни приложения. Днес голям процент от доставчиците на услуги с добавена стойност използват предимно пакета Kannel за осъществяване на връзка с центъра за кратки съобщения и реализиране на сравнително елементарни мобилни услуги. Той е функционален и надежден , но за съжаление няма версия за Windows. При откриване на проблем в използвана от нас безплатна реализация на SMPP нямаме гаранция, че той ще бъде отстранен, което поражда опасност за безпроблемната работа на рискови мобилни приложения. Всички тези фактори са причина за създаването на библиотека за операционната система Windows имплементираща пълна спецификация на SMPP протокола, даваща възможност за бързо и лесно създаване на приложения. Библиотеката е написана на езика C# , а свободна библиотека на този език за програмиране отсъства към момента.

4. Избор на технология за реализация

Изборът на технология трябва да бъде съобразен с целите и задачите на проекта. Съществуват множество възможности, но със своята гъвкавост технологията .NET се явява изключително удачен избор при изграждането на модулите на приложението. Визията на Microsoft за .NET е да създаде платформа, която да може да обединява хетерогенна инфраструктура от сървъри, да интегрира бизнес процеси на различни компании по стандартен начин , и да предоставя на потребителя достъп до информация, която им е нужна , по всяко време от всяко място и всяко устройство. Правилният избор на език за програмиране може значително да ускори и улесни разработката на приложението. Тъй като приложението ще има уеб модул ще ни бъде необходим и уеб сървър . За да разширим възможностите на приложението и да съхраним нужната ни информация ще трябва да използваме и система за управление на база данни.

4.1 Програмен език

За език за програмиране е избран C#. Той е специално създаден за .NET Framework и съобразен с неговите особености. C# е съвременен, обектно ориентиран и типово обезопасен език за програмиране, който е наследник на C и C++ . Той комбинира леснотата на използване на Java с мощността на C++. C# въвежда нови концепции – разделяне на типовете на два вида – стойностни (value types) и референтни (reference type), автоматично управление на паметта, делегати, събития, атрибути, XML документация и други [8]. Стандартизиран е от ECMA и ISO. C# е насочен към компонентно-орентираното програмиране, при което софтуерът се изгражда чрез съединяването на различни готови компоненти и описание на логиката на взаимодействие с тях. При проектирането на .NET Framework и езика C# компонентният подход е залегнал на най-дълбоко архитектурно ниво. Те са изградени с идеята да осигурят висока сигурност и надеждност на изпълнявания софтуер. .NET Framework предоставя среда за контролирано изпълнение на управляван код, с което прави невъзможно възникването на някои от най-неприятните проблеми, свързани с управлението на паметта, неправилното преобразуване на типове и други. C# наследява всички тези характеристики от .NET Framework и добавя към тях допълнителни механизми за предпазване на програмистите от често срещани грешки.

4.2 База данни

Правилният избор на система за управление на релационнаа база от данни(Relation Database Management System – RDBMS) е изключително важно с оглед на приложението, неговата безпроблемна работа и бъдещето му развитие. Много често системите за управление на релационни бази от данни се наричат „сървъри за управление на бази от данни(СУБД)” или просто „Database сървъри”. Популярни RDBMS системи са Oracle, Informix, MS SQL Server, IBM DB2, Sybase, MySQL. Всички те имат своите предимства и недостатъци и изборът им зависи от

поставените цели. СУБД като Oracle и Informix са лидери, осигуряващи висока надеждност, способност за работа с голям брой потребители и огромно натоварване. Те са комерсиални продукти работещи в повечето случаи на UNIX операционни системи. MySQL би бил разумен избор, но при него отсъстват доста характеристики, а и за комерсиални цели се заплаща лиценз. Най-подходящ за целите на проекта, вземайки под внимание и избраната среда и програмен език е системата за управление на релационни бази данни MS SQL Server 2005. Тя се характеризира със лесна администрация подобно на останалите сървъри на Microsoft, добра производителност, висока скалируемост и висока надеждност. Сървърът поддържа всички характеристики на съвременните RDBMS системи. При малки проекти може да се използва неговата безплатна ограничена версия MS SQL Server 2005 Express Edition.

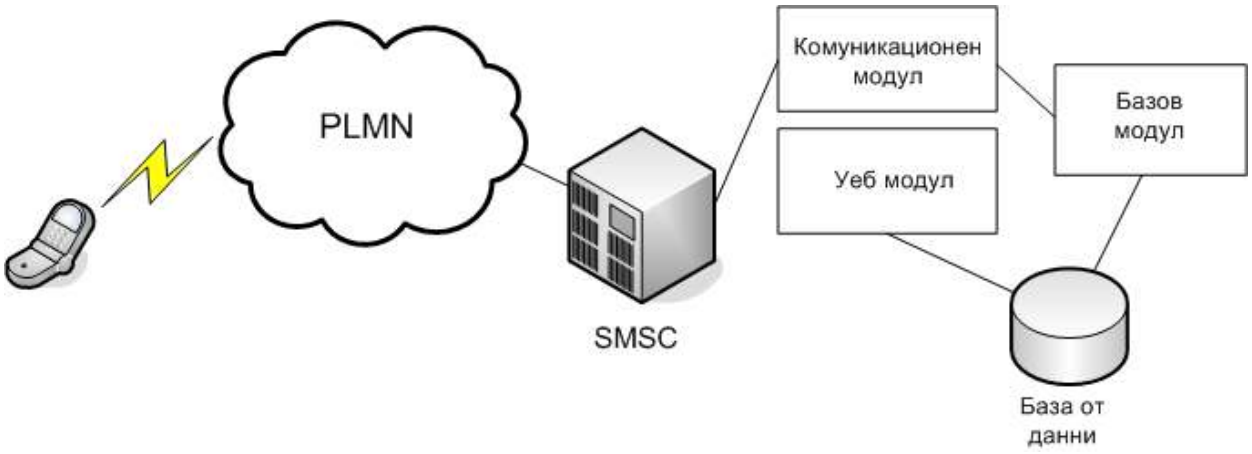
4.3 Уеб сървър

Наличието на уеб модул изисква и използването на уеб сървър. Съществуват множество варианти, но изборът зависи от нуждите на проекта и операционната система, която ще се използва. Най-често използваните уеб сървъри са Apache и Microsoft IIS. Apache има версии за Unix\Linux операционни системи както и за Windows. Той е надежден и сигурен . В случая на поставената ни задача Microsoft IIS се явява по-подходящ избор. Той работи чудесно под операционна система Windows, интегрира се с .NET Framework и е лесен за администриране и поддръжка. В миналото му се носеше слава, че е несигурен , но в последните си версии Microsoft са подобрили значително сигурността и той по нищо не отстъпва на основния си конкурент Apache.

5. Реализация

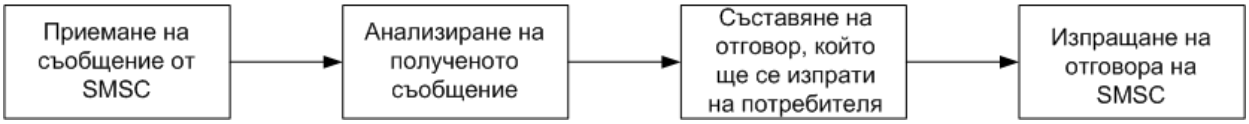
За реализирането на проекта се изисква задълбочено изследване и анализ на поставените цели и задачи. Разработваното приложение е разделено на три

функционални модула. Всеки от тях изпълнява определена задача. Комуникационният модул е отговорен за осъществяването на връзката и правилната комуникация с центъра за кратки съобщения. Той трябва да е гъвкаво, надеждно и сигурно проектиран и именно за това на него е отделено най-голямо внимание. Имплементира протокола SMPP спрямо стандарта и дава възможност с негова помощ да се разработват разнообразни приложения. Уеб модулът се грижи за обновяването на информацията в базата от данни. Той се използва от операторите на приложението, давайки им възможност бързо и лесно да актуализират информацията. Базовият модул приема, анализира и определя отговора, който ще се върне на потребителя. Той взаимодейства с комуникационният модул и базата от данни. Проектиран е да обработва множество съобщения едновременно. На фиг.10 е показана блокова диаграма на приложението.



Фиг.10 Блок диаграма на приложението

Работата на системата за мобилни информационни услуги посредством кратки съобщения може да бъде обобщена в четири фази. Всяка фаза изпълнява точно определена функция от работния процес, след завършването й предава резултата на следващата. На фиг.11 са показни етапите през, които преминава работата на системата.



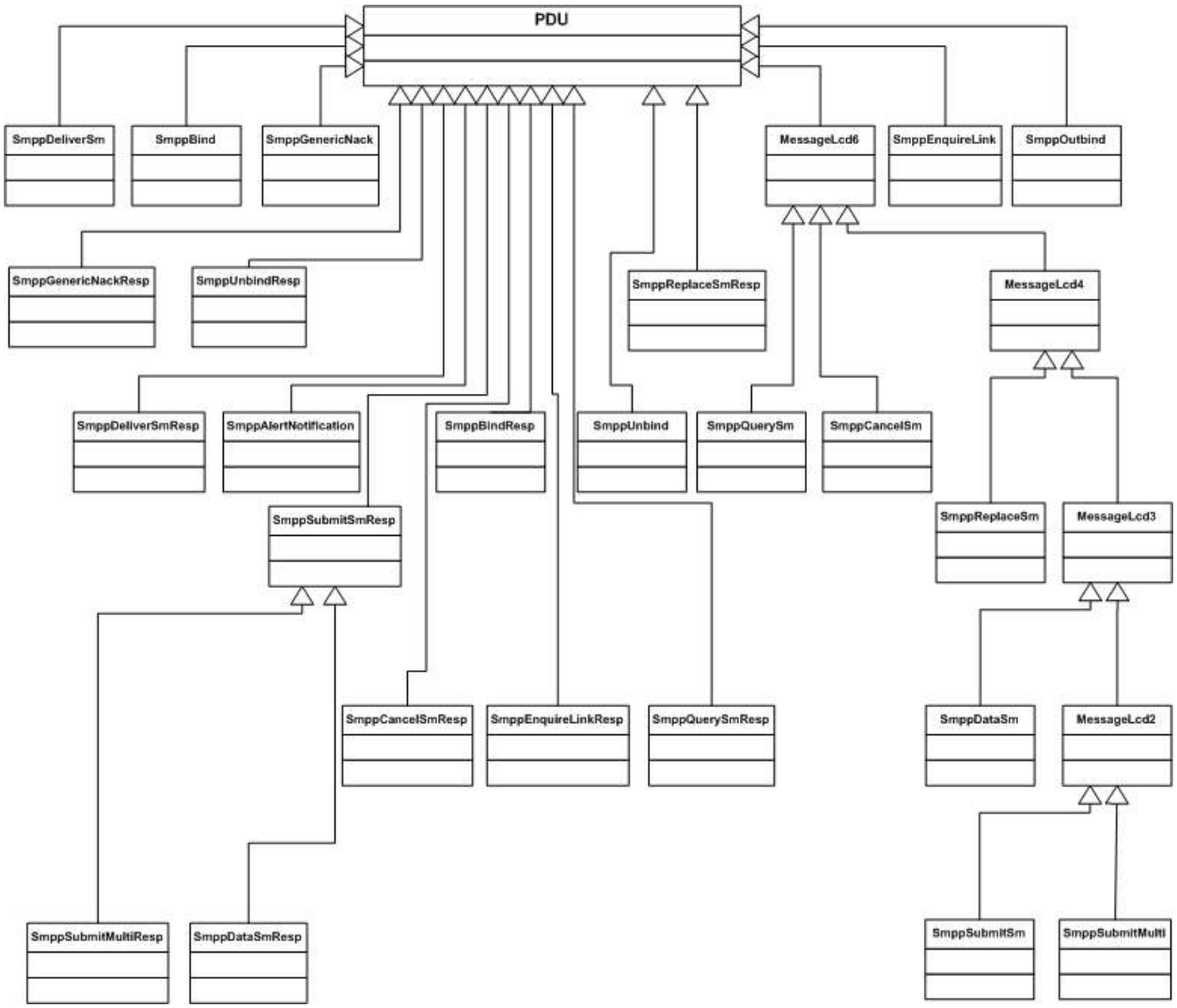
Фиг.11 Етапи от работата на приложението

По време на първата фаза приложението приема съобщение от центъра за кратки съобщения. SMSC изпраща пакет deliver_sm до приложението, което прихваща настъпилото събитие. Всяко получено съобщение се обработва в отделна нишка, което предоставя възможност за едновременно приемане на много съобщения. Полученото кратко съобщение се анализира, като се извлича необходимата информация от него. Съгласно анализа се определя какъв отговор да бъде изпратен на потребителя. Ако той е написал верен код или наименование на град му се изпраща исканата информация, в противен случай системата ще изпрати служебно съобщение, че не разпознава заявката. След определяне на отговора, той се изпраща от приложението до центъра за кратки съобщения посредством SMPP операцията submit_sm.

5.1 Комуникационен модул

Комуникационният модул има за цел да осигури правилна и надеждна комуникация с центъра за кратки съобщения. Той е изключително важен елемент от системата тъй като неговата безпроблемна работа осигурява стабилност на цялото приложение. Може да комуникира с центрове за кратки съобщения на различни производители. Комуникационният модул представлява имплементация на протокола SMPP версия 3.4 съгласно стандарта. Възможно е да се използва извън рамките на приложението за създаването на системи, които се нуждаят от взаимодействие с център за кратки съобщения. Реализирането на модула изисква внимателен анализ, на работата и операциите на протокола както и подбирането на подходящи структури от данни за представянето на параметрите. Езикът C# е

достатъчно гъвкав и мощен за да осигури всички нужни средства за изпълнението на задачата. Комуникационният модул се състои от три групи от класове. Едната група имплементира PDU и операциите свързани с него. На фиг.12 е показана клас диаграмата на класовете реализиращи PDU и SMPP операциите.



Фиг.12 Йерархия на класовете реализиращи PDU и SMPP операциите

Клас PDU - абстрактен клас, който е базов в йерархията от класове. Той представя структурата на пакета и имплементира общата функционалност на пакетите заявка(request PDU) и пакетите отговор(response PDU). Реализира методи за инициализиране на пакет, извличане на информация от пакет,

пресмятане дължина на пакет и добавянето и в пакета, генерирането на поредните номера на пакетите и др.

Клас MessageLcd6 – абстрактен клас, наследява класа PDU. Добавя няколко параметъра(sourceAddressTon, sourceAddressNpi, sourceAddress), общи за операциите data_sm, query_sm, submit_sm, submit_multi, cancel_sm и replace_sm.

Клас MessageLcd4 – абстрактен клас, наследява клас MessageLcd6. Добавя параметър(RegisteredDelivery), общ за операциите submit_sm,submit_multi, data_sm и replace_sm.

Клас MessageLcd3 – абстрактен клас, наследява, класа MessageLcd4. Добавя параметри(ServiceType, EsmClass, DataCoding), общи за операциите submit_sm, submit_multi и data_sm.

Клас MessageLcd2 – абстрактен клас, наследява класа MessageLcd3. Добавя параметрите(protocolId, PriorityFlag, ScheduleDeliveryTime, ValidityPeriod, ReplaceIfPresentFlag, DefaultMessageId, ShortMessage) общи за операциите submit_sm и submit_multi.

Класове **SmppQuerySm** и **SmppCancelSm** – класове наследяващи класа **MessageLcd6** и реализират функционалността на SMPP операциите query_sm и cance_sm.

Клас SmppReplaceSm – клас наследяващ класа **MessageLcd4**. Реализира функционалността на операцията replace_sm.

Клас SmppDataSm – клас наследяващ класа MessageLcd3. Реализира функционалността на операцията data_sm.

Класове SmppSubmitSm и **SmppSubmitMulti** – класове наследяващи класа **MessageLcd2**. Реализира операциите submit_sm и submit_multi.

SmppSubmitMultiResp и **SmppDataSmResp** – класове наследяващи класа **SmppSubmitSmResp**. Реализират съответно пакетите submit_multi_resp и data_sm_resp. Класовете **SmppBind**, **SmppBindResp**, **SmppUnbind**, **SmppUnbindResp**, **SmppDeliverSm**, **SmppDeliverSmResp**, **SmppEnquireLink**, **SmppEnquireLinkResp**, **SmppAlertNotification**, **SmppOutBind**, **SmppCancelSmResp**, **SmppReplaceSmResp**, **SmppQuerySmResp,**

SmppGenerickNack и **SmppGenerickNackResp** наследяват абстрактният клас **PDU**. Реализират пакетите bind, bind_resp, unbind, unbind_resp, deliver_sm, deliver_sm_resp, enquire_link, enquirelink_resp, alert_notification, outbind, cancel_sm_resp, replace_sm_resp, query_sm_resp, generick_nack и generick_nack_resp с техните специфични особености. Друга група от класове в комуникационният модул се грижи за реализирането на събития свързани с SMPP операциите. **SmppEventArgs** се явява базов клас, а останалите класове (**AlertEventArgs**, **BindEventArgs**, **BindRespEventArgs**, **CancelSmRespArgs**, **CancelSmRespEventArgs**, **DataSmEventArgs**, **DataSmRespEventArgs**, **DeliverSmEventArgs**, **DeliverSmRespEventArgs**, **EnquireLinkEventArgs**, **EnquireLinkRespEventArgs**, **GenericNackEventArgs**, **QuerySmEventArgs**, **QuerySmRespEventArgs**, **ReplaceSmEventArgs**, **ReplaceSmRespEventArgs**, **SubmitMultiEventArgs**, **SubmitMultiRespEventArgs**, **SubmitSmEventArgs**, **SubmitSmRespEventArgs**, **UnbindEventArgs** и **UnbindRespEventArgs**) са му наследници. Класът **CommonErrorEventArgs** наследява **System.EventArgs** и се използва за вътрешни нужди. Третата група от класове в комуникационният модул, имплементира множество помощни операции свързани с PDU:

Клас DestinationAddress – клас, който реализира списък от адреси на абонати за нуждите на операцията submit_multi.

Клас UnsuccessAddress – определя unsuccessaddress, който се използва при отговора на операцията submit_multi.

Клас PDUFactory – клас, който имплементира механизъм за извличане на пакет от входящия поток от данни и генерирането на пакет базиран на полето command.

Клас PDUUtil – клас, който реализира общата функционалност на пакетите заявка(request PDU). Имплементира методи за добавяне на текст на кратко съобщение в пакет и за задаване на параметри в TLV таблицата.

Клас SmppStringUtil – клас, който се грижи за операциите свързани с обработката на стрингове(strings) в пакета. Реализира методи за извличане на данни от тип string и C-string.

Клас TLVTable – клас , който се реализира tag, length, value (TLV) таблицата , която се съдържа в SMPP PDU.

Клас UnsignedNumConverter – клас, който се грижи за преобразуването от big-endian наредба в little-endian наредба и обратно. Big-endian и little-endian са термини описващи наредбата на последователността от байтове в паметта на компютъра. За пример нека разгледаме числото 1025(2 на десета степен плюс 1) съхранено в четири байта:

00000000 00000000 00000100 00000001

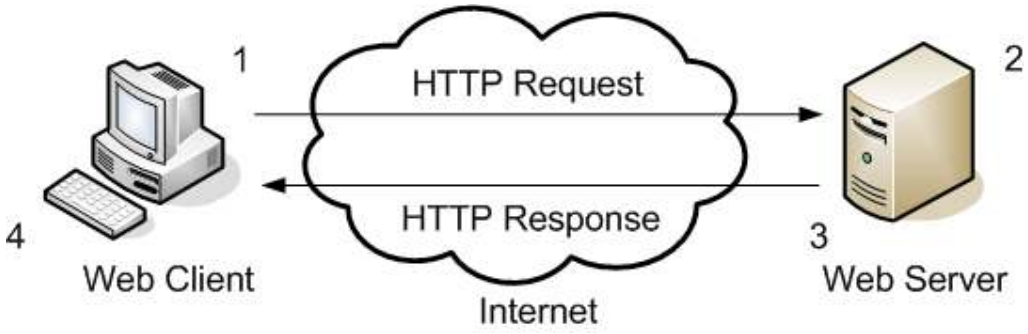
Адрес	**Big-Endian** **Предоставяне на числото 1025**	**Little-Endian** **Представяне на числото 1025**
00	**00000000**	**00000001**
01	**00000000**	**00000100**
02	**00000100**	**00000000**
03	**00000001**	**00000000**

Компютърните мрежи работят с наредбата big-endian. Когато компютри работещи с little-endian си предават информация по мрежата (ip адреси за пример), те трябва да конвертират информацията в big-endian. По същия начин , когато получат информация по мрежата, те трябва да я конвертирато обратно в little-endian. Класът **AsyncSocketClient** се грижи за асинхронната комуникация с центъра за кратки съобщения, а класа **SMPPCommunicator** се явява обвиващ клас на цялата библиотека.

5.2 Уеб модул

За реализирането на уеб модулът е използвана библиотеката ASP.NET, която е стандартна част от .Net Framework. Тя дава програмен модел и съвкупност от технологии, чрез които може да се изграждат сложни уеб приложения. Те

използват модела заявка-отговор(request-response). Този модел е илюстриран на фиг. 13



Фиг.13 Модел заявка-отговор

1.Потребителят въвежда в браузъра адрес на страницата(URL). Браузърът изпраща HTTP заявка(request) към уеб сървъра.
2.Сървърът получава заявката и я обработва. В случая с ASP.NET, IIS намира процес, който може да обработи дадената заявка.
3.Резултатът от вече обработената заявка се изпраща обратно към потребителя/ клиента под формата на HTTP отговор(response).
4.Браузърът показва получения отговор като уеб страница.

Основният компонент на ASP.NET е уеб формата – абстракция на HTML страницата, която интернет потребителите виждат в браузъра си. Замисълът на създателите на ASP.NET е работата с уеб формите да бъде интуитивна и максимално улеснена. Осигурява се високо ниво на абстракция, предоставя се на разработчиците разнообразие от контроли и се намалява нуждата от писането на чист HTML код. Уеб модулът осъществява връзка с базата от данни и осигурява удобен и лесен интерфейс за изпълнение на заявки към нея. Необходимата информация за свързване с базата от данни се намира във файлът web.config, от където се прочита при зареждане на страницата. Визуализирането на данните става с помощта на уеб контролите GridView и ListBox. За осигуряване на сигурността на системата и за да се избегнат атаки от вида SQL injection върху базата от данните заявките към нея са параметризирани. За обновяването на

информацията се използва съхранената процедура UpdateTableWeather, което предоставя възможност за по-сигурно и бързо изпълнение на заявката в сравнение с подаване и в „чист” вид.

5.3 Базов модул

Базовият модул реализира основната логика на платформата за кратки съобщения. Той е свързан както с комуникационния модул така и с базата от данни. Задачата му е да обработи входящо кратко съобщение, да анализира неговите параметри и да върне на потребителя подходящ отговор. За своята работа той се нуждае от набор от входни параметри(ip адрес на центъра за кратки съобщения, порт на центъра за кратки съобщения, потребителско име, парола, номер за изпращане на кратките съобщения, параметри необходими за връзка с базата от данни както и път където да се записва log файлът от работата на системата). Тези параметри са записани в конфигурационен файл, който е в XML формат. Реализиран е клас **Configuration**, който се грижи за обработката на параметрите като разполага с методи за прочитане и записване в конфигурационния файл. Системата може да обработи няколко входящи съобщения едновременно. Това се постига с помощта на нишки. Обработката на всяко входящо съобщение се стартира в отделна нишка. Базовият модул прочита от базата от данни информация за градовете и текущата прогноза към момента за всеки от тях. Тази информация се съхранява в паметта, за да се осигури бърз анализ в последствие на изпратения от потребителя текст. Самият анализ на текста и намирането на съвпадение с предварително зададени имена на градове се осъществява с помощта на регулярни изрази. За реализирането на този механизъм се грижи методът Match, който търси текст в зададен предварително шаблон. Нека разгледаме следния пример, потребител изпраща текст „Veliko Tarnovo”, системата извлича този текст от краткото съобщение и проверява дали има съвпадение с предварително дефинирания шаблон от имена (този шаблон се задава за всеки град и се пази в базата от данни), който е Veliko Tarnovo|Veliko

Turnovo|Veliko Tyrnovo|V.Turnovo|V.Tarnovo|V.Tyrnovo за град Велико Търново. По-долу е показана имплементацията на метода Match с помощта на регулярните изрази в езика за програмиране C#:

```
public static bool Match(string text, string pattern)
{
            Regex regex = new Regex(pattern);
            Match match = regex.Match(text.Trim().ToLower());

            if (match.Success) return true;
            else return false;
}
```

При съвпадение на името на град с този в шаблона се връща на потребителя кратко съобщение с информация за прогнозата за времето. За изпращането на съобщение се използва метода `public SmppSubmitSm SendSMS(Pdu.TonType sourceTon,Pdu.NpiType sourceNpи, stringsourceAddress,Pdu.TonType destinationTon,Pdu.NpiType destinationNpi,string destinationAddress,string shortMessage,Pdu.DataCodingType dcs`, който връща обект от тип SmppSubmitSm. След анализа на съобщението, приетото и изпратеното кратко съобщение се запазват в базата от данни. Цели се при нужда да се проверят евентуални действия на потребителя или да се извади информация за активността през определен период от време. Тази операция се осъществява след обработката на съобщението, а не по време на анализа, за да се увеличи скоростта на приложението, т.е. целият анализ се извършва в паметта на компютъра и после се въвежда информацията в базата от данни. Методът, който се използва за записване в базата от данни е InsertTableSMS със следната сигнатура:

```
public void InsertTableSMS(SqlConnection connection,Pdu.TonType sourceTon,
Pdu.NpiType sourceNpi,long sourceAddress, Pdu.TonType destinationTon,
Pdu.NpiType destinationNpi, long destinationAddress, Pdu.DataCodingType dcs,
DateTime time, string shortMessage)
```

Възможни проблеми при изпълнението на програмата се записват в log файл с точната дата и час. Реализиран е графичен потребителски интерфейс(GUI) с помощта на библиотеката Windows.Forms налична стандартно в .Net Framework. Тя предоставя възможност за изграждането на прозорци и диалози с графични контроли в тях, чрез които се извършва интерактивно взаимодействие с

потребителя. При настолните графични приложения потребителският интерфейс позволява на потребителя директно да взаимодейства с програмата чрез мишка и клавиатура, а програмата прихваща неговите действия и ги обработва по подходящ начин. Изградения графичен интерфейс е опростен и дава възможност за лесно взаимодействие със системата. Класът **ConfigurationDialog** реализира диалогов прозорец, който позволява да се улесни промяната на някои от конфигурационните параметри бързо и лесно и да се избегне прякото редактиране на конфигурационният файл на приложението.

5.4 База от данни

Проектирането на базата от данни е етап при, който след организацията на данните, те се представят като таблици и връзки между тях. Първо системата се изгражда на ниво данни и след това се изграждат останалите слоеве на приложението. За правилното функциониране на системата са ни необходими две таблици: „TableWeather” и „TableSMS”. Таблицата “TableWeather” съхранява информация за градовете и прогнозата за времето във всеки от тях. Състои се от следните полета:

- City *varchar(50) , NOT NULL* – съдържа името на град;
- Text *varchar(100), NOT NULL* – съдържа информация за възможните начини за изписване на наименование на град. Това поле задава шаблон от възможните думи, които биха отговаряли за даден град. Разделител между думите е символа „|” , който е избран поради спцифичното му значение в обработката на регулярни изрази в езика C#. Изпратеният от потребителя текст се проверява дали е наличен в този шаблон;
- ShortMessage *varchar(160), NOT NULL* – съдържа прогнозата на времето за град. Дължината е 160 символа , съгласно максималната дължина на едно кратко съобщение.

Таблицата „TableSMS” съхранява информация за получените и изпратените кратки съобщения. Състои се от следните полета:

- Id *int, NOT NULL, auto_increment, primary key* – уникално идентифицира записите в таблицата;
- SourceTon *tinyint, NOT NULL* – съдържа информация относно типът на TON(Type of Number) дефиниран от изпраща на съобщението;
- SourceNPI *tinyint, NOT NULL* – съдържа информация относно типът на NPI(Number Plan Indicator) дефиниран от изпращача на съобщението;
- SourceAddress *bigint, NOT NULL* – съдържа информация за номера на изпращача на съобщението;
- DestTON *tinyint, NOT NULL* – съдържа информация относно типът на TON(Type of Number), който ще се изпрати на получателя;
- DestNPI *tinyint, NOT NULL* – съдържа информация относно типът на NPI, който ще се изпрати на получателя;
- DestAddress – *bigint, NOT NULL* - съдържа информация относно номера до когото ще бъде изпратено съобщението;
- DCS – *tinyint, NOT NULL* – съдържа информация относно кодовата схема на съобщението;
- Date – *datetime , NOT NULL* – съдържа информация относно датата на получаване/изпращане на съобщението;
- ShortMessage – *varchar(160), NOT NULL* – съдържа текста на краткото съобщение. Дължината е 160, което е и максималната възможна дължина на едно съобшение.

Таблицата “TableSMS” е изключително полезна, тъй като предоставя подробна точна информация за приетите и изпратени съобщения и може да бъде използвана за създаването на допълнителен статистически модул за в бъдеще. За нуждите на платформата за мобилни услуги посредством кратки съобщения наличните таблици са напълно достачни за да реализират необходимата ни функционалност.

Възможно е в следващите версии на системата да се добавят и нови таблици като примерно таблица на абонатите на услугата и др, които да разширят системата и да предоставят нови възможности на потребителите.

6. Ръководство за работа

Ръководството за работа със ситемата за мобилни информационни услуги посредством кратки съобщения описва необходимите знания за инсталиране и конфигуриране на платформата нужни на администратора, както и запознава потребителя с работата на приложението.

6.1 Ръководство на администратора

Систематата за мобилни информационни услуги посредством кратки съобщения е проектирана да работи с Windows 2000/XP/Vista операционна система с инсталиран на нея Microsotf .Net Framework 2.0. Минималната конфигурация на системата трябва да включва процесор 1 Ghz, 512 RAM памет, и 1Gb дисково пространство. За съхранението на данните е необходим Microsoft SQL Server 2005 Standard Edition или Microsoft SQL Server 2005 Express Edition. Необходимо е да се създаде база от данни с име WeatherSMS както и таблиците TableWeather и TableSMS със необходимите атрибути. За обслужването на уеб модулът на приложението е нужно да има инсталиран Microsoft IIS Server. Настройката на системата става чрез редактиране на конфигурационният файл SMSPlatform.config. Той е в XML формат и е възможно да се променя с обикновен текстов редактор. Ето примерен конфигурационен файл:

```
<?xml version="1.0" encoding="utf-8"?>
<Configuration xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <ConnectionString>Server=.\SQLExpress; Database=WeatherSMS;Integrated Security=false; Persist Security
Info=True;Password=miroslav123;User ID=sa;Initial Catalog=WeatherSMS; </ConnectionString>
```

```
  <Number>90799</Number>
  <SMPP_Host_Address>10.96.211.50</SMPP_Host_Address>
  <SMPP_Port>3701</SMPP_Port>
  <SMPP_System_ID>SMPP1</SMPP_System_ID>
  <SMPP_System_Password>smsc11</SMPP_System_Password>
  <EnquireLinkInterval>10</EnquireLinkInterval>
  <FailSMS>Sistemata ne razbira tvoiata zaivka.</FailSMS>
  <LogPath>c:\SMSPlatform.log</LogPath>
</Configuration>
```

ConnectionString - задава необходимата информация за връзката с базата от данни.

Number – задава краткият номер, който ще е необходим за услугата. Този номер се предоставя от мобилния оператор.

SMPP_Host_Address - задава IP адрес на центъра за кратки съобщения. Предоставя се от мобилния оператор.

SMPP_Port – задава порт на центъра за крактки съобщения. Предоставя се от мобилния оператор.

SMPP_System_ID – задава потребителско име необходимо за свръзването с центъра за кратки съобщения. Предоставя се от мобилния оператор.

SMPP_System_Password – задава парола необходима за свързването с центъра за кратки съобщения. Предоставя се от мобилния оператор.

EnquireLinkInterval – задава се интервал в секунди, който определя след колко време системата да израти операцията enquire_link на центъра за кратки съобщения. Когато няма обмяна на данни между приложението и SMSC те си обменят enquire_link пакети, с които подържат връзката между тях.При отсъствие на такъв пакет връзката се разпада. На всеки enquire_link пакет центъра за кратки съобщения връща отговор enquire_link_resp.

FailSMS – задава се текст на съобщението, което ще се изпраща на потребителя при негова грешна заявка.

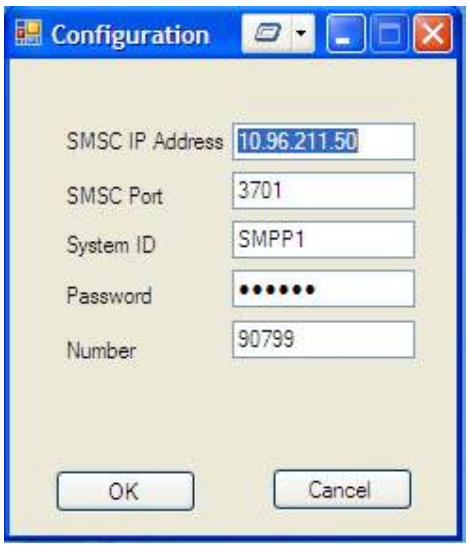
LogPath – указва пътя, където ще се пази log файлът на приложението.

След конфигурирането на входните параметрите на приложението системата е готова за работа. Тя се старира от файла SMSPlatform.exe. Ако няма проблеми с

параметрите и връзката на системата с базата от данни, се зарежда потребителският графичен интерфейс. Той е показан на фиг.14

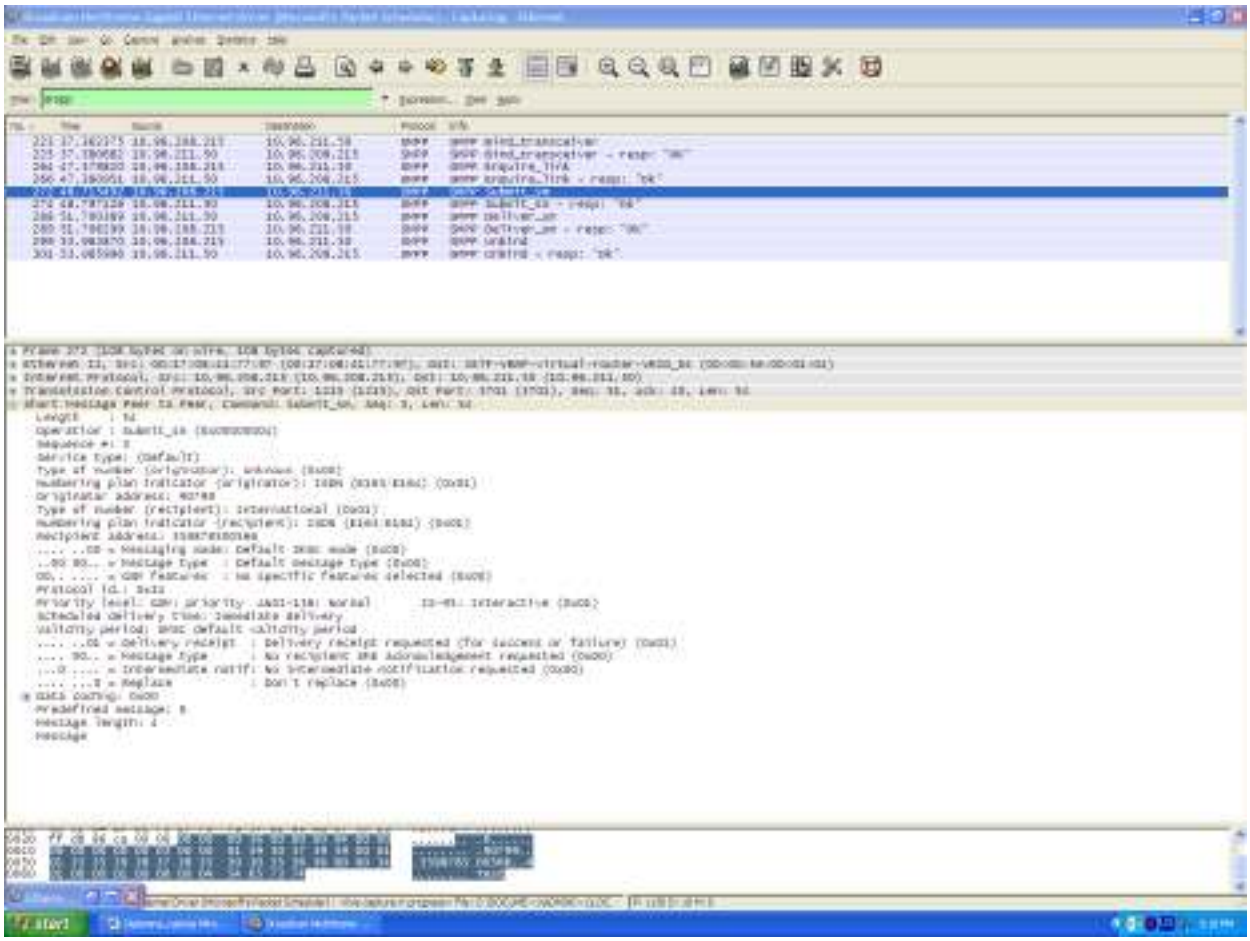
Фиг.14 Потребителски интерфейс

Налични са два бутона Start и Stop, които стартират и спират работата на приложението. Тези събития се записват с точната дата и час в log файла на системата и могат да бъдат проверени при нужда. В StatusBar лентата на приложението се визиулизира статуса на връзката на системата с центъра за кратки съобщения. От менюто Edit може да се избере опцията Configuration. Появява се прозорец, който позволява да се променят някои от основните параметри на приложението бързо и лесно, без да се налага да се редактира конфигурационният файл директно. На фиг.15 е показан конфигурационният прозорец.



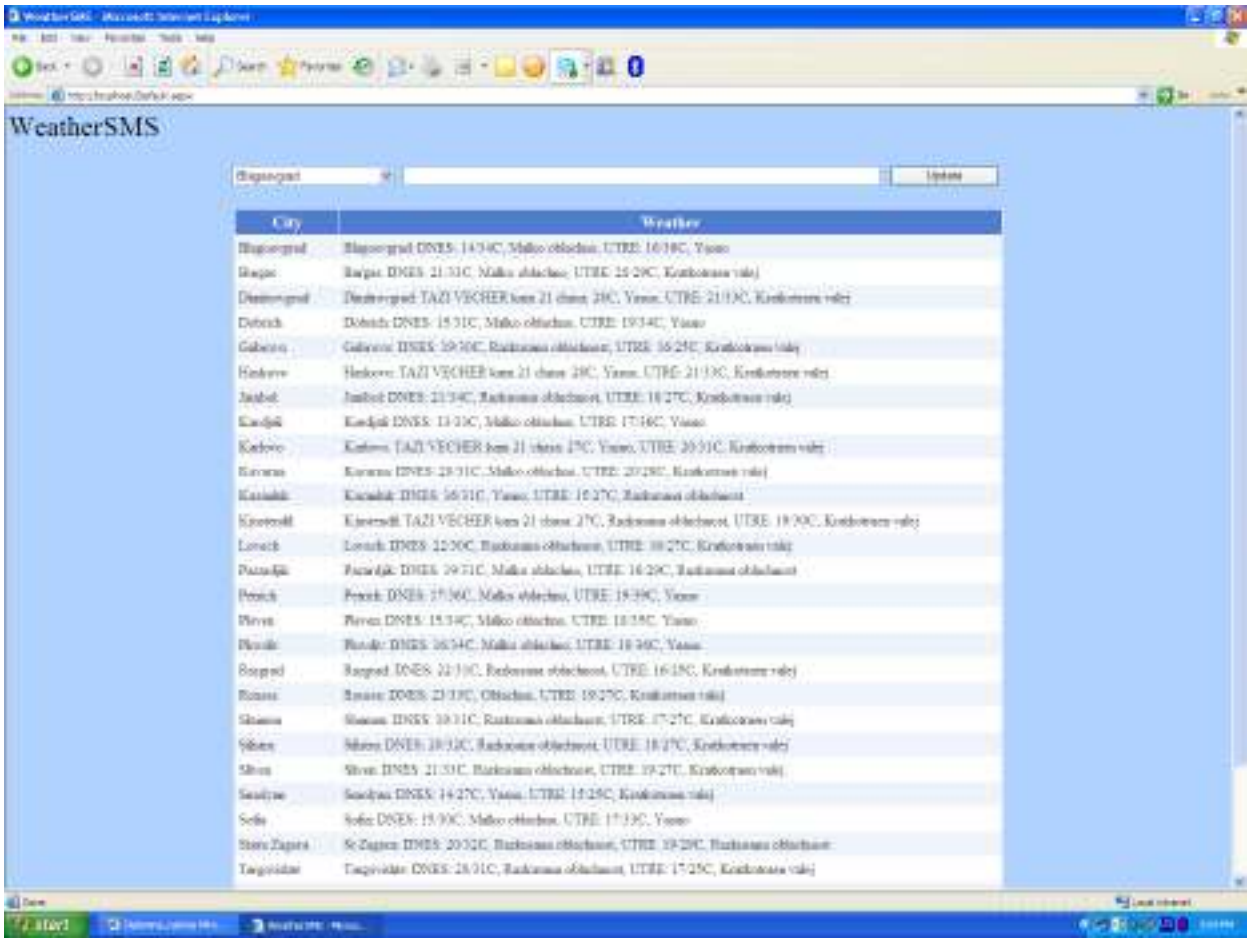
Фиг.15 Конфигурационен прозорец на приложението

Възможно е да възникнат непредвидени ситуации при работата с приложението(изгубване на връзка с базата от данни и др.), информация за тях се записва в log файла. Администраторът на приложението, преглеждайки го, може да установи причината за възникването на евентуалните проблеми. Удобен и много лесен начин за проверка на мрежовата комуникация между приложенението и центъра за кратки съобщения е инструмента Ethereal. Той е протоколен анализатор, използва се за отстраняване на мрежови проблеми, анализ, разработка на мрежови протоколи и за образователни цели. Разполага с филтър за протокола SMPP. Администраторът на приложението може да го използва за да наблюдава и изследва на ниско ниво мрежовите операции. На фиг.16 са показани пакети анализирани с Ethereal



Фиг.16 Анализиране на пакети с Ethereal

Уеб модулът дава лесен начин за промяна на информацията за времето на даден град без директно да се достъпва базата от данни. За конфигурирането на връзката с нея е необходимо да се редактира файлът web.config. На фиг.17 е показан уеб интерфейсът на приложението.



Фиг.17 Уеб интерфейс на приложението

Работата с него е изключително лесна. От DropDownListBox контролата на приложението се избира града, за който искаме да променим прогнозата за времето. В текстовото поле въвеждаме новата прогноза за времето и натискаме бутона Update. Записът се обновява в базата от данни и се показва в интерфейса с актулните прогнози.Полето за въвеждане на прогнозата за времето е ограничено до 160 символа, така че няма реална опасност да се въведе по-дълго съобщение от едно sms съобщение.

6.2 Ръководство на потребителя

Осъществяването на взаимодействие на потребителя със системата за мобилни информационни услуги посредсвом кратки съобщения се извършва посредством мобилен апарат с възможност за изпращане на кратки съобщения. Мобилният оператор, към който е свързана платформата предоставя кратък номер, на който абонатите могат да изпращат своите съобщения. Съобщението трябва да съдържа кодът или наименованието на населеното място, за което потребителят желае да получи моментната прогноза за времето. Възможно е текстът да бъде написан с малки, главни или с малки и главни букви, където и да е в краткото съобщение. Задължително е съобщенията да се изписват на латиница. Ако потребителят е написал правилния код или наименование на населено място и системата има информация за това населено място, той получава кратко съобщение съдържащо моментната прогноза за времето за него, в противен случай бива уведомен, че е системата не разпознава неговата заявката. Получаването на отговор на заявката на потребителя се осъществява почти мигновено. Възможно е в определени ситуации да се забави отговора на потребителската заявка. Най – често причини това да се случва, е моментна загуба на покритие (примерно, ако сме в движение и попаднем в радиосянка) или грешка при изпращането от страна на центъра за кратки съобщения. При тези ситуации съобщението попада в retry механизма на SMSC и в зависимост от грешката (загуба на покритие или друга) се прави опит през различен период от време то да бъде доставено на абоната. Често срещана грешка от страна на потребителите е да изпишат наименованието на даден град с букви на латиница и кирилица. Примерно при изписването на град София потребителят е възможно да изпише „Sofi” на латиница, след което по невнимание да промени езика на въвеждане на кирилица и да напише буквата „а”. На пръв поглед наименованието “Sofia” изглежда напълно коректно, но в действителност последната буква на града е написана на кирилица, а не на латиница. При тази ситуация системата няма да приеме за коректна потребителската заявка.

7. Заключение и възможности за бъдещо развитие

Реализираната система за мобилни информационни услуги посредством кратки съобщения задоволява нарастващата нужда на потребителите на мобилни устройства от бърз и лесен достъп до информация. Възможно е само с промени по информацията в таблиците на базата от данни да се създадат разнообразни по съдържание информационни услуги. Реализираната имплементация на протокола SMPP предоставя възможност на различни разработчици да създават своите приложения на нейната база. Възможностите за развитие на проект от това естество са многобройни поради разнообразието от потребности на потребителите, които може да адресира. В случай, че се наблегне на разширяване на функциолността на приложението е възможно да се предостави избор потребителите да се абонират за услугата и всеки ден, например да получават прогнозата за времето в определен час. Информация за всички абонати ще бъде запазена в отделна таблица. Възможно е да се реализира механизъм за обработка на submit_sm_resp операциите. Това ще позоволи да се вземе message_id на изпратеното съобщението и на по-късен етап да се направи запитване към центъра за кратки съобщения за неговия статус или да се изтрие съобщение. По отношение на графичният интерфейс е възможно да се добави графична контрола, която да визуализира в реално време натоварването на системата. Уеб модулът може да бъде разширен като се добави администраторска част, позовляваща по-голяма възможност за промяна на базата от данни. Може да бъде реализиран и статистически инструмент, който ще дава информация за активността на потребителите на системата. Разработеното приложение реализира поставените цели и задачи. Използва се лесно и работи ефективно и сигурно. Обяснени са системните изисквания на реализацията, както и начина на работа с нея. Предоставя възможност да се разшири сравнително безпроблемно според нуждите на потребителите и да покрие един по-голям кръг от бъдещи услуги.

Приложение

AUC – Authentication Center (център за проверка на автентичността)

DCS – Data Coding Scheme (кодова схема)

EIR – Equipment Identity Register (регистър/ база данни за идентификация на оборудване)

EMS – Enhanced Message Service (технология за изпращане на дълги съобщения)

GSM – Global System for Mobile Communication (Глобална система за мобилни комуникации

HLR – Home Location Register (регистър/ база за домашни абонати)

ISDN – Integrated Services Digital Network (цифрова мрежа с интеграция на услугите)

MSC – Mobile Switched Center (централа за връзка с подвижни абонати или мобилна централа)

MMS – Multimedia Message Service (технология за изпращане на мултимедийни съобщения)

NPI – Number Planning Indicator (тип номерационна схема използвана в телекомуникациите)

PLMN - Public Land Mobile Network(обществена наземна мобилна мрежа)

PSTN – Public Switched Telephone Network (обществена комутируема телефонна мрежа)

PDN – Public Data Network (обществена мрежа за данни)

SIM – Subscriber Identity Module (модул за идентификация на абоната)

SMPP – Short Message Peer to Peer (протокол за връзка с център за кратки съобщения)

SMS – Short Message Service (услуга за изпращане на кратки съобщения)

SMSC – Short Message Service Center (център за изпращане на кратки съобщения)

TON – Type of Number (тип на номер използван в телекомуникациите)

UMTS – Universal Mobile Telecommunication System (Универсални мобилни телекомуникационни услуги)

VLR – Visitor Location Register (регистър/ база данни за временно пребиваващите абонати)

Използвана литература

[1] SMSForum, “Short Message Peer to Peer Protocol Specification 3.4”, document version 12 octomber 1999 Issue 1.2

[2] SMSForum, “Short Message Peer to Peer Protocol Specification 5.0”, document version 5.0

[3] SMSForum, “ SMPP ver3.4 protocol implementation guide for GSM/UMTS”, document version 1.0

[4] 3GPP, “Technical Realization of Short Message Service(SMS)” , TS 03.40 version 7.5.0 Release 1998

[5] доц. инж. Любомир Добош, доц.инж. Ян Духа, проф.инж. Станислав Марцехевски, доц.инж. Владимир Визер, „Мобилни радиомрежи”, издателство „Джиев Трейд” ООД, София 2005г.

[6] проф. Христо Кабакчиев, „Лекции по мобилни комуникации”

[7] Siegmund M. Redl, “GSM and Personal Communications Handbook”, издателство „Artech House”, 2005г.

[8] Светлин Наков и колектив, ”Програмиране за .NET Framework”, издателство „Фабер”, 2005г.

[9] Кратък преглед на SMPP

http://en.wikipedia.org/wiki/Short_message_peer-to-peer_protocol