Системата ще работи върху надеждно защитен уеб сървър, достъпът до който се осъществява по защитен протокол (HTTPS).

Системата ще осигурява следните нива на защита на достъпа до ресурсите:

- В съответствие с чл. 21, ал.5 от НОИИСРЕАУ, интегритетът на предаваните електронни изявления през интернет чрез уеббазирани потребителски интерфейси се осигурява чрез използване на протокол HTTPS, като за установяване на криптирана връзка с потребителя на услугата се използва протокол TLS (Transport Layer Security –Сигурност на транспортния слой), версия 1.1 или по-висока, дефиниран в Препоръка RFC 4346, приета от IETF (The Internet Engineering Task Force–Целева група за Интернет инженеринг) през април 2006 г. В съответствие с чл.21, ал.6 интегритетът на предаваните изявления чрез програмни интерфейси се осигурява чрез използване на протокол HTTPS, като за установяване на криптирана връзка с потребителя на услугата се използва протокол TLS (Transport Layer Security–Сигурност на транспортния слой), версия 1.2 или по-висока,дефиниран в Препоръка RFC 5246, приета от IETF (The Internet Engineering Task Force –Целева група за Интернет инженеринг) през август 2008 г.
- Не се допуска съхранението на пароли на администратори, на вътрешни и външни потребители и на акаунти за достъп на системи (ако такива се използват) в явен вид. Всички пароли трябва да бъдат защитени с подходящи сигурни алгоритми (напр. BCrypt, PBKDF2, scrypt (RFC 7914) за съхранение на пароли и където е възможно, да се използва и прозрачно криптиране на данните в СУБД със сертификати (transparent data-at-rest encryption);
- Да бъде предвидена система за ежедневно създаване на резервни копия на данните, които да се съхраняват извън инфраструктурата на системата по ред, определен с наредбата по чл. 43, ал. 2 от НОИИСРЕАУ;
- Не се допуска използването на Self-Signed сертификати за публични услуги;
- Всички уебстраници (вътрешни и публично достъпни в Интернет) трябва да бъдат достъпни единствено и само през протокол HTTPS. Криптирането трябва да се базира на сигурен сертификат с валидирана идентичност (Verified Identity), позволяващ задължително прилагане на TLS 1.2, който е издаден от удостоверителен орган, разпознаван от най-често използваните браузъри (Microsoft Internet Explorer, Google Chorme, Mozilla Firefox). Идентификацията се осъществява двустранно по протокол TLS (Transport Layer Security –Сигурност на транспортния слой), версия 1.2 или по-висока, дефиниран в Препоръка RFC 5246, приета от IETF (The Internet Engineering Task Force –Целева група за Интернет инженеринг) през август 2008 г. Ежегодното преиздаване и подновяване на сертификата трябва да бъде включено като разходи и дейности в гаранционната поддръжка за целия срок на поддръжката;
- Ще бъдат извършени тестове за сигурност на всички уебстраници, като минимум чрез автоматизираните средства на SSL Labs за изпитване на сървърна сигурност (https://www.ssllabs.com/ssltest/). За нуждите на автентикация с КЕП трябва да се предвиди имплементирането на обратен прокси сървър (Reverse Proxy) с балансиране на натоварването, който да препраща клиентските сертификати към вътрешните приложни сървъри с нестандартно поле (дефинирано в процеса на

разработка на Системата) в HTTP Header-а. Схемата за проксиране на заявките трябва да бъде защитена от Spoofing;

- Като временна мярка за съвместимост настройките на уебсървърите и Reverse Proxy сървърите ще бъдат балансирани така, че Системата да позволява използване и на клиентски браузъри, поддържащи по-стария протокол TLS 1.1. Това изключение от общите изисквания за информационна сигурност не се прилага за достъпа на служебни потребители от държавната администрация и доставчици на обществени услуги, които имат служебен достъп до ресурси на Системата;
- При разгръщането на всички уебуслуги (Web Services) ще се използва единствено протокол HTTPS със задължително прилагане на минимум TLS 1.2;
- Програмният код ще включва методи за автоматична санитизация на въвежданите данни и потребителски действия за защита от злонамерени атаки, като минимум SQL инжекции, XSS атаки и други познати методи за атаки, и ще отговаря, където е необходимо, на Наредбата за оперативна съвместимост и информационна сигурност;
- При проектирането и разработката на компонентите на Системата и при подготовката и разгръщането на средите ще се спазват последните актуални препоръки на OWASP (Open Web Application Security Project);
- Ще бъде изграден модул за проследимост на действия и събития в Системата. За всяко действие (добавяне, изтриване, модификация, четене) трябва да регистрира и съдържа информация със следните атрибути:
    - Уникален номер;
    - Точно време на възникване на събитието;
    - Вид (номенклатура от идентификатори за вид събитие);
    - Данни за информационна система, където е възникнало събитието;
    - Име или идентификатор на компонент в информационната система, регистрирал събитието;
    - Приоритет;
    - Описание на събитието;
    - Данни за събитието.
- Астрономическото време за удостоверяване настъпването на факти с правно или техническо значение се отчита с точност до година, дата, час, минута, секунда и при технологична необходимост - милисекунда, изписани в съответствие със стандарта БДС ISO 8601:2006;
- Астрономическото време за удостоверяване настъпването на факти с правно значение и на такива, за които се изисква противопоставимост, ще бъде удостоверявано с електронен времеви печат по смисъла на Глава III, Раздел 6 от Регламент ЕС 910/2014. Трябва да бъде реализирана функционалност за получаване на точно астрономическо време, отговарящо на горните условия, и от доставчик на доверителни услуги или от държавен орган, осигуряващ такава услуга, отговаряща на изискванията на RFC 3161;

- Ще бъдат проведени тестове за проникване (penetration tests), с които да се идентифицират и коригират слаби места в сигурността на системата.

7.18. Използваемост

7.18.1. Общи изисквания за използваемост и достъпност

Технологичната платформа на база на която ще бъде разработена системата отговаря на утвърдените стандарти за достъпност (accessibility) и ползваемост (usability). При дизайна на потребителския интерфейс се използва модула за теми на ASP.Net MVC (theming engine), което ще позволи да се имплементира унифицирана рамка, или основна страница (Master page). Всички други страници от системата взимат общата рамка на основната страница, където е зададена основната HTML структура и референциите към CSS файловете и другите ресурсни файлове. По този начин се гарантира, че всяка една от страниците ще има сходен дизайн и ще бъде валидирана като документ с ниво „strict" според препоръките на W3C/WAI.

По отношение на потребителския интерфейс, софтуерното решение ще се придържа към следните принципи:

- При проектирането и разработката на софтуерните компоненти и потребителските интерфейси ще се спазват стандартите за достъпност на потребителския интерфейс за хора с увреждания WCAG 2.0, съответстващ на ISO/IEC 40500:2012;
- Всички ресурси ще са достъпни чрез GET заявка на уникален адрес (URL). Не се допуска използване на POST за достигане до формуляр за подаване не заявление, за генериране на справка и други;
- Функционалностите на потребителския интерфейс на Системата ще бъдат независими от използваните от потребителите интернет браузъри и устройства, при условие че последните са версии в период на поддръжка от съответните производители. Трябва да бъде осигурена възможност за ползване на публичните модули на приложимите услуги през мобилни устройства – таблети и смарт-телефони, чрез оптимизация на потребителските интерфейси за мобилни устройства (Responsive Design);
- Не се допуска използване на капча (Captcha) като механизъм за ограничаване на достъпа до документи и/или услуги. Алтернативно, Системата трябва да поддържа "Rate Limiting" и/или "Throttling" съгласно изискванията в т. 7.1.1. от настоящите изисквания. Допуска се използването на Captcha единствено при иденетифицирани много последователни опити от предполагаем „бот";
- ще бъде осигурен бърз и лесен достъп до електронните услуги и те ще бъдат промотирани с подходящи навигационни елементи на публичната интернет страница – банери, елементи от главното меню и др.;
- Публичните уеб страници на Системата ще бъдат проектирани и оптимизирани за ефективно и бързо индексиране от търсещи машини с цел популяризиране сред потребителите и по-добра откриваемост при търсене по ключови думи и фрази. При разработката на страниците и при изготвяне на автоматизираните процедури

за разгръщане на нова версия на Системата ще се използват инструменти за минимизиране и оптимизация на размера на изходния код (HTML, JavaScript и пр.) с оглед намаляване обема на файловете и по-бързо зареждане на страниците;

- Не се допуска използването на HTML Frames, за да не се пречи на оптимизациите за търсещи машини;
- При разработката на публични уеббазирани страници ще се използват и ще се реализира поддръжка на:
  - Стандартните семантични елементи на HTML5 (HTML Semantic Elements);
  - JSON-LD 1.0 (http://www.w3.org/TR/json-ld/);
  - Open Graph Protocol (http://ogp.me) за осигуряване на поддръжка за качествено споделяне на ресурси в социални мрежи и мобилни приложения;

- В екранните форми на Системата ще се използват потребителски бутони с унифициран размер и лесни за разбиране текстове в еднакъв стил.
- Всички текстови елементи от потребителския интерфейс ще бъдат визуализирани с шрифтове, които са подходящи за изобразяване на екран и които осигуряват максимална съвместимост и еднакво възпроизвеждане под различни клиентски операционни системи и браузъри. Не се допуска използването на серифни шрифтове (Serif).
- Полета, опции от менюта и командни бутони, които не са разрешени конкретно за ролята на влезлия в системата потребител, няма да бъдат достъпни за този потребител. Това не отменя необходимостта от ограничаване на достъпа до бизнес логиката на приложението чрез декларативен или програмен подход.
- Всяка екранна форма ще има наименование, което да се изписва в горната част на екранната форма. Наименованията ще подсказват на потребителя какво е предназначението на формата.
- Всички търсения ще са нечувствителни към малки и главни букви.
- Полетата за пароли ще различават малки и главни букви.
- Полетата за потребителски имена ще позволяват използване на имейл адреси като потребителско име, включително да допускат всички символи, регламентирани в RFC 1123, за наименуването на хостове;
- Главните и малките букви на въвежданите данни се запазват непроменени, не се допуска Системата да променя капитализацията на данните, въвеждани от потребителите.
- Системата ще позволява въвеждане на данни, съдържащи както български, така и символи на официалните езици на ЕС.
- Наименованията на полетата следва да са достатъчно описателни, като максимално се доближават до характера на съдържащите се в тях данни.
- Системата ще поддържа прекъсване на потребителски сесии при липса на активност. Времето трябва да може да се променя от администратора на системата без промяна в изходния код. Настройките за време за прекъсване на неактивни сесии ще включват и възможността администраторите да дефинират стилизирана страница с информативно съобщение, към която Системата да пренасочва автоматично браузърите на потребителите в случай на прекъсната сесия;
- Дългите списъци с резултати ще се разделят на номерирани страници с подходящи навигационни елементи за преминаване към предишна, следваща,

първа и последна страница, към конкретна страница. Навигационните елементи трябва да са логически обособени и свързани със съответния списък и да се визуализират в началото и в края на HTML контейнера, съдържащ списъка;

- За големите йерархически категоризации ще се предвиди възможност за навигация по нива или чрез отложено зареждане (lazy load).

7.18.2. Интернационализация

Изпълнителят, вземайки предвид изискванията посочени от Възложителя в техническата спецификация, предвижда:

- Системата ще може да съхранява и едновременно да визуализира данни и съдържание, което е въведено/генерирано на различни езици, а именно – двуезична на български и на английски език;
- Всички софтуерни компоненти на Системата, използваните софтуерни библиотеки и развойни комплекти, приложните сървъри и сървърите за управление на бази данни, елементите от потребителския интерфейс, програмно-приложните интерфейси, уебуслугите и др. ще се поддържат стандартно и ще са конфигурирани изрично за спазване на минимум Unicode 5.2 стандарт при съхранението и обработката на текстови данни, съответно трябва да се използва само UTF-8 кодиране на текстовите данни.
- Всички публично достъпни потребителски интерфейси ще поддържат многоезичност - български и английски език.
- Публичната част на Системата ще бъде разработена и ще включва набори с текстове на минимум два официални езика в ЕС, а именно български и английски език. Преводите на английски език ще бъдат осъществени професионално, като няма да се допуска използването на средства за машинен превод без ръчна проверка и корекции от професионални преводачи.
- Прикачените документи, ще бъдат приложени към поле съдържащо описание на тяхното наименование на български и на английски език, като и поле за въвеждане на дата на издаване, независимо дали тя се съдържа в номера на документа. Наименованията на документите на английски език ще бъдат зададени като типови опции от меню за избор, като се ползват унифицираните им наименования в ЕС или при липса на унифицирани наименования, да се използват най-често използваните еквиваленти в страните от ЕС.
- Версиите на съдържанието на съответните езици ще включват всички текстове, които се визуализират във всички елементи на потребителския интерфейс, справките, генерираните от системата електронни документи, съобщения, нотификации, имейл съобщения, номенклатурите и таксономиите и др. Данните, които се съхраняват в системата само на български език, се изписват/визуализират на български език;
- Системата ще позволява превод на всички многоезични текстове с подходящ потребителски интерфейс, достъпен за администратори на Системата, без промени в изходния код. Модулът за превод на текстове, използвани в Системата, ще поддържа и контекстни референции, които да позволяват на администраторите да тестват и да проверяват бързо и лесно направените преводи и тяхната съгласуваност в реалните екрани, страници и документи;

- Публичната част на Системата ще позволява превключване между работните езици на потребителския интерфейс в реално време от профила на потребителя и от подходящ, видим и лесно достъпен навигационен елемент в горната част на всяка страница, който включва не само текст, но и подходяща интернационална икона за съответния език;
- При визуализация на числа ще се използва разделител за хиляди (интервал).
- При визуализация на дати и точно време в елементи от потребителския интерфейс в генерирани справки или в електронни документи всички формати за дата и час щеса съобразени с избрания от потребителя език/локация в настройките на неговия профил:
    - За България стандартният формат е „DD.MM.YYYY HH:MM:SS", като наличието на време към датата е в зависимост от вида на визуализираната информация и бизнес-смисъла от показването на точно време;
    - Системата трябва да поддържа и всички формати съгласно ISO БДС 8601:2006.

Както е посочено в изискванията на Възложителя, системата ще да бъде разработена с такива формати за обмен на данни, които да позволяват ползването им за доклади в системата EDAMIS на Евростат, както и прехвърлянето им в перспективната система по Регламент (ЕО) №1013/2006 относно превози на отпадъци, която е предмет на друга разработка и е с предопределени параметри доколкото тя трябва да прехвърля данни в „Централното приложение на ЕС", както е описано в точка техническото задание. Поради това е необходима координация между екипа разработващ настоящата система и екипа, които изготвя задание за система по Регламент (ЕО) №1013/2006, независимо от това, кой екип първи ще започне работа. Възложител и на двете разработки се явява изпълнителният директор на ПУДООС.

7.19. Изисквания за използваемост на потребителския интерфейс

Потребителският интерфейс ще бъде изграден при спазване на следните изисквания:

- Електронните форми за подаване на заявления и за обявяване на обстоятелства ще бъдат реализирани с AJAX или с аналогична технология, като по този начин се гарантират следните функционалности:
    - Контекстна валидация на въвежданите данни на ниво "поле" от форма и контекстни съобщения за грешка/невалидни данни в реално време;
    - Възможност за избор на стойности от номенклатури чрез търсене в списък по част от дума (autocomplete) и визуализиране на записи, отговарящи на въведеното до момента, без да е необходимо пълните номенклатури да са заредени в браузъра на клиента и потребителят да скролира дълги списъци с повече от 10 стойности;
- В електронните форми ще бъде реализирана валидация на въвежданите от потребителите данни на ниво "поле" (in-line validation). Валидацията ще се извършва в реално време на сървъра, като при успешна валидация данните от

съответното поле следва да бъдат запазени от сървъра;

- Системата ще гарантира, че въведените, валидираните и запазените от сървъра данни остават достъпни за потребителите дори за процеси, които не са приключили, така че при волно, неволно или автоматично прекъсване на потребителската сесия поради изтичане на периода за допустима липса на активност потребителят ще може да продължи съответния процес след повторно влизане в системата, без да загуби въведените до момента данни и прикачените до момента електронни документи;
- Ще бъде реализирана възможност за добавяне и редактиране от страна на администраторите на Системата, без да са необходими промени в изходния код, на контекстна помощна информация за:
    - всяка електронна форма или стъпка от процес, за която има отделен екран/форма;
    - всяка група полета за въвеждане на данни (в случаите, в които определени полета от формата са групирани тематично);
    - всяко отделно поле за въвеждане на данни;
- Ще бъде разработена контекстна помощна информация за всички процеси, екрани и електронни форми, включително ясни указания за попълване и разяснения за особеностите при попълване на различните групи полета или на отделни полета;
- Контекстната помощна информация, указанията към потребителите и информативните текстове за всяка електронна административна услуга няма да съдържа акроними, имена и референции към нормативни документи, които са въведени като обикновен текст (plain-text). Всички акроними, референции към нормативни документи, формуляри, изисквания и др. ще бъдат разработени като хипервръзки към съответните актуални версии на нормативни документи и/или към съответния речник/списък с акроними и термини;
- Достъпът на потребителя до контекстната помощна информация ще бъде реализиран по унифициран и консистентен начин чрез подходящи навигационни елементи, като например чрез подходящо разположени микро бутони с икони, разположени до/пред/след етикета на съответния елемент, за който се отнася контекстната помощ, или чрез обработка на "Mouse Hover/Mouse Over" събития;
- При проектирането и реализацията на потребителския интерфейс ще се отчете, че той трябва да бъде еднакво използваем и от мобилни устройства (напр. таблети), които не разполагат с мишка, но имат чувствителни на допир екрани.
- Потребителският интерфейс ще бъде достъпен за хора с увреждания съгласно изискванията на чл. 48, ал. 5 от ЗОП.

7.20. Изисквания за използваемост в случаи на прекъснати бизнес процеси

- Системата ще може да съхранява перманентно всеки започнал процес/процедура по подаване на заявление или обявяване на обстоятелства, текущия му статус и всички въведени данни и прикачени документи дори ако потребителят е прекъснал волно или неволно потребителската си сесия.
- При вход в системата потребителят ще получава прегледна и ясна нотификация, че има започнати, но недовършени, неизпратени или неподписани заявления, и ще бъде подканен да отвори модула за преглед на историята на транзакциите.

Модулът за преглед на историята на транзакциите ще поддържа следните функционалности:

- Визуализиране на списък с историята на подадените заявления, като минимум със следните колони:
  - Дата;
  - входящ номер;
  - код на тупа формуляр;
  - подател (име на потребител и имена на физическото лице - подател);
  - статус на заявлението;
- Да предлага видни и лесни за използване от потребителите контроли/инструменти:
  - за филтриране на списъка (от дата до дата, за предефинирани периоди, като "последния един месец", "последната една година";
  - сортиране на списъка по всяка от колоните, без това да премахва текущия филтър;
  - свободно търсене по ключови думи по всички колони в списъка и метаданните на прикачените/свързаните документи със заявленията, което да води до динамично филтриране на списъка.

7.21. Изисквания за проактивно информиране на отребителите

За всички публични интернет страници ще бъде реализирана функционалност за публикуване на всяко периодично обновявано съдържание в стандартен формат (RSS 2.x, Atom или еквивалент), например:

- Новини;
- Обявления;
- нормативни документи;
- отговори по ЗДОИ и др.

Изпълнителя предвижда Системата да поддържа възможност за автоматично генериране на електронни бюлетини, които да се разпращат периодично или при

настъпване на събития по електронна поща до регистрираните в Системата потребители, които са заявили или са се съгласили да получават такива бюлетини, като те ще имат възможност да настройват предпочитанията през потребителския си профил в Системата.

7.22. Системен журнал

Изграждането решение ще осигурява проследимост на действията на всеки потребител (одит), както и версия на предишното състояние на данните, които той е променил в резултат на своите действия - системен журнал.

Атрибутите, които ще се запазват при всеки запис, ще включват като минимум следните данни:

- дата/час на действието;
- модул на системата, в който се извършва действието;
- действие;
- обект, над който е извършено действието;
- допълнителна информация;
- IP адрес и браузър на потребителя.

Размерът на журнала на потребителските действия нараства по време на работа на всяка система, което налага по-различното му третиране от гледна точка на организация на базата данни:

- по време на работа на Системата потребителският журнал ще се записва в специализиран компонент, който поддържа много бързо добавяне на записи; този подход се налага, за да не се забавя излишно работата на Системата;
- специална фонова задача ще акумулира записаните данни и да ги организира в отделна специално предвидена за целта база данни, отделна от работната база данни на Системата;
- данните в специализираната база данни ще се архивират и изчистват, като в специализираната база данни ще бъде достъпна информация за не повече от 2 месеца назад; при необходимост от информация за предишен период администраторът на Системата първо ще възстанови архивните данни;
- ще бъде предоставен достъп до системния журнал на органите на реда чрез потребителски или програмен интерфейс; за достъпа ще се изисква електронна идентификация.

7.23. Дизайн на бази данни и взаимодействие с тях

При реализацията на решението, ще се използва стандартна релационна база данни Postgres. Предлаганото решение се базира на модел на данни използваш СУБД

Postgres, но поради използването на гъвкав слой за достъп до базата данни базиран на .NET Entity Framework, предлаганото решение при нужда може да бъде пренесено и върху друга стандартна релационна база данни, като например Microsoft SQL Server.

При използване на база данни (релационна или нерелационна(NoSQL) следва да бъдат следвани добрите практики за дизайн и взаимоедйствие с базата данни, в т.ч.:

- дизайнът на схемата на базата данни (ако има такава) трябва да бъде с максимално ниво на нормализация, освен ако това не би навредило сериозно на производителността;
- базата данни трябва да може да оперира в клъстър; в определени случаи следва да бъде използван т.нар. sharding;
- имената на таблиците и колоните трябва да следват унифицирана конвенция;
- трябва да бъдат създадени индекси по определени колони, така че да се оптимизират най-често използваните заявки; създаването на индекс трябва да е мотивирано и подкрепено със замервания;
- връзките между таблици трябва да са дефинирани чрез foreign key;
- периодично трябва да бъде правен анализ на заявките, включително чрез EXPLAIN (при SQL бази данни), и да бъдат предприети мерки за оптимизиране на бавните такива;
- задължително трябва да се използват транзакции, като нивото на изолация трябва да бъде мотивирано в предадената документация;
- при операции върху много записи (batch) следва да се избягват дългопродължаващи транзакции;
- заявките трябва да бъдат ограничени в броя записи, които връщат;
- при използване на ORM или на друг слой на абстракция между приложението и базата данни, трябва да се минимизира броят на излишните заявки (т.нар. n+1 selects проблем);
- при използване на нерелационна база данни трябва да се използват по-бързи и компактни протоколи за комуникация, ако такива са достъпни.

8. ПОДХОД ЗА ИЗПЪЛНЕНИЕ НА ИЗИСКВАНИЯТА КЪМ ПОРЪЧКАТА

8.1.Методики използвани при реализацията

8.1.1. Методология за анализ на процесите и моделиране

Реализацията на софтуерни системи с нарастваща сложност изисква прилагането на общи стандарти и ефективни методологии за проектиране и разработка. От появата си през 1997г. до сега Унифицираният език за моделиране – UML™ – се превърна фактически в стандартен „графичен език за визуализиране, специфициране, конструиране и документиране на елементите на една софтуерно-интензивна система“. Сред предимствата на един UML подход са:

- Плавен преход между отделните етапи при разработката на софтуерния продукт – от формулирането на изискванията до крайната реализация, чрез използване на междинни модели за анализ и проектиране на архитектурата

и поведението на системата, които позволяват всяко изискване да бъде адресирано на подходящото ниво на абстракция;

- Възможност за ефикасно разширяване на съществуващи модели с цел включване на допълнителни функционални изисквания в една бързо променяща се бизнес среда – проектът може да расте запазвайки ефективна и ясна архитектура;
- Стандартен графичен език за изразяване и дискусия на идеи, изисквания и проектантски решения, който улеснява комуникацията, особено при паралелна разработка от различни екипи (например при outsorcing на разработката);
- Един UML-базиран процес за разработка позволява да се настрои количеството и качеството на проектната документация според изискванията на конкретния проект (време, обем, бюджет, нужда от бъдещо разширяване).

UML моделирането ще бъде основно средство в процеса на анализа и проектирането на системата. С негова помощ ще се постигнат няколко основни цели:

- формално описание на единнатаа система и нейните компоненти (подсистеми);
- описание на начина на имплементация на работата на функционалността;
- провеждане на експерименти с модела (симулация) и база за взимане на решения за оптимизиране на системата.

UML осигурява независимост на създадените модели от техническата реализация в информационните и комуникационни технологии. Прилагането на стандарта в този проект ще бъде гъвкаво и съобразено със спецификата на анализираните процеси, по време на проектиране и създаване на моделите ще се запазва смисловото съдържание на моделираните обекти.

UML дефинира правила за изграждане на различни типове диаграми, които служат за графично представяне на различни аспекти на софтуерната система. Общият UML модел на който ще бъде разработен за софтуерната система се разглежда като изграден от два взаимно допълващи се модела:

Структурен модел (Structural model), който показва структурата на системата и подсистемите, използвайки обекти, атрибути, операции и връзки;

Функционален модел (Functional model), наричан още динамичен модел (Dynamic model), който показва функционалността на системата, обособена в конкретните и модули и измененията на системата във времето.

Всеки елемент от модела се описва с дадена UML диаграма - частично графично представяне на един от двата системни модела. В нотацията на UML съществуват 14 официални типа диаграми:

- **Клас диаграма** (Class diagram) – описва класове от обекти и връзките между тях;

- **Обектна диаграма** (Object diagram) – представя цялостен или частичен изглед на даден обект на системата;
- **Диаграма на пакети** (Package diagram) – представя йерархична структура на модул или компонент на системата;
- **Диаграми на компоненти** (Component diagram) – описва структурата и връзките между компонентите;
- **Диаграма на съставна структура** (Composite structure diagram) – представя декомпозиция на клас и комуникацията му с другите класове;
- **Диаграма за разгръщане/диаграма на внедряването** (Deployment diagram) – представя връзката между софтуерната и хардуерната реализация на системата;
- **Профилна диаграма** (Profile Diagram) – използва се за описване на класовете и пакетите;
- **Случай на употреба** (Use Case Diagram) – представя начина на взаимодействие между потребител и системата, употребява се за специфициране на функционалните изисквания към разработваната софтуерна система и изготвяне на тестове;
- **Диаграма на дейност** (Activity diagram) – описва последователни или паралелни контролни потоци;
- **Диаграма на на състоянията** (State diagram) – описва промяната на състоянието на определен обект или система по време на неговия жизнен цикъл в резултат на възникващи събития;
- **Диаграма на последователност** (Sequence diagram) – служи за представяне на взаимодействието между обектите, като се набляга върху последователността на дейностите;
- **Диаграма на комуникация** (Communication diagram) – описва взаимодействията между обектите, като особено внимание се отделя на връзките между тях;
- **Диаграма за преглед на взаимодействието** (Interaction overview diagram) – представлява комбинация от диаграми на последователност и комуникационни диаграми;
- **Времева диаграма** (Timing Diagram) – представя взаимодействие между обекти, като се набляга върху синхронизацията на обектите.

BPMN

Business Process Model and Notation (BPMN) е другият стандарт, който ще се използва за визуално моделиране на бизнес процеси под формата на диаграма – **Business Process Diagram (BPD)**. BPMN надгражда Unified Modeling Language (UML), като целта е да се разшири техническата насоченост на описанието в UML, създавайки универсален

език за описание на бизнес процеси, който да е еднакво четим и удобен за бизнес потребителите (мениджъри, бизнес анализатори и др.)и в същото време да дава възможност за описание на сложна семантика на бизнес процесите.

Поради спецификата на проекта и широкия набор от функционалности, която трябва да се реализира, използването на неформалния метод би довело до създаване на описания с твърде голям обем, чието систематично интерпретиране би било сериозно затруднение. Подобна сложна система е много по-подходяща за описание чрез диаграми и модели на данните, отколкото с текстово описание на изискванията. По тази причина най-подходящият подход към дейността е използването на полуформален подход, чрез комбинирането на документ със спецификация на изискванията, с модел от UML диаграми, представящи систематично изискванията към архитектурата на системата и отделните потребителски случаи (use cases).

8.1.2. Метод за спецификация на потребителските случаи

Потребителските случаи (use case) се смятат за стандартна нотация при изготвянето на изискванията в обектно-ориентирано моделиране. Потребителските случаи на употреба могат да подобрят комуникацията между заинтересованите страни, експерти по бизнес анализ, разработчици и други специалисти участващи в разработването на системата.

Разглеждат се три различни нива на абстракция на изискванията (бизнес, потребителски и системни) и това как потребителските случаи могат да бъдат определящ фактор при разработването на детайлните изисквания за разработване на системата.

✓ **Класификация на изискванията**

Изискванията към Система като тази, определят неговото функционално поведение, както и типовете информация, до която трябва да се осигури достъп, как тя се трансформира и организира.

Определянето на изискванията дават възможност на участниците в проекта да уточнят ясно целта, насоката и да изяснят очакванията от информационните цели. Тъй като често участниците – бъдещи потребителите на системата дават доста обобщени бизнес изисквания, а екипът разработващ системата има нужда от точни, ясни и недвусмислено очертани изисквания се налага допълнително изясняване, с цел те да се трансформират в детайлни, пълни и подлежащи на тестване спецификации, на базата на които да бъде разработена система максимално отговаряща на нуждите на Възложителя.

✓ **Бизнес изисквания**

Бизнес изискванията представят цели на високо равнище, които следва да се реализират чрез внедряването на системата. Те са описани в Техническата спецификация, която описва визията и обхвата на проекта. Бизнес изискванията идентифицират първичните ползи, които системата ще донесе на Поръчителя и потребителите. Те представляват най-високото ниво абстракция във веригата на изискванията. Те описват целите, възможностите и основните потребителски типове, които дават обобщена представа, за това как би следвало да функционира системата.

✓ **Потребителски изисквания**

Потребителските изисквания описват задачите и действията, които даден потребител трябва да може да извършва с помощта на системата. Тези изисквания трябва

да се уточнят с хората, които на практика ще използват системата. Тези потребители са в състояние да опишат не само задачите, които е нужно да извършват в системата, но и нефункционалните й характеристики, които са важни за доброто приемане и лесната работа със системата.

Потребителските изисквания следва да са в синхрон с бизнес изискванията. Те могат да бъдат уточнени най-пълноценно посредством описанието на потребителските сценарии. Фокусират се върху конкретните нужди при използване на системата и съответно са доста по-значими от традиционният подход за извличане на изисквания, посредством директно питане на потребителите какво биха искали да прави системата.

✓ **Детайлни системни изисквания**

Те представляват системните изисквания на много детайлно ниво, което подпомага пълното, детайлно специфициране на изискванията, позволяваща използването на така специфицираните изисквания при изграждане на компонентите от разработчиците.

Тези изисквания трябва да съответстват на потребителските и бизнес изисквания и да съдържат точни и ясни критерии, които да позволяват верификация.

Функционалните изисквания определят функционалностите, които разработващият екип следва да реализира при изграждането на системата, така че системата да удовлетворява бизнес изискванията. Бизнес изискванията улавят очакваното поведение на системата, което може са бъде изразено чрез услуги, задачи и функции, от които системата се нуждае, за да отговори на целта, заложена в проекта.

Информационните изисквания дефинират информационните нужди на Поръчителя и заинтересованите страни. Те описват типовете информация и данни, които системата следва да предостави, или до които да даде достъп. Те специфицират предоставените данни, посредством качеството, което те следва да имат, източникът от който идват, как би следвало да бъдат обработвани, как следва да бъдат комбинирани за анализ и кои аналитични методи следва да бъдат използвани.

Другите изисквания, извън функционалните и информационните, могат да бъдат специфицирани, за да се опишат допълнителни аспекти на системата, като изисквания към интерфейса и средата (законова, културна, политическа).

✓ **Качествени характеристики на изискванията**

Добавянето на качествени характеристика към описанието на функционалните, информационните и други изисквания, позволява оценяването им в определени измерения, които са важни или за потребителите или за екипа, разработващ системата.

Характеристиките на изискванията са свойства или качества, които системата трябва да притежава. Те могат да съдържат стандарти, регулации и условия, с които системата трябва да бъде съобразена: описания на външния интерфейс, изисквания за производителността, дизайна и ограничения при изпълнението; качествени характеристики.

✓ **Идентифициране на изисквания посредством потребителски случаи (use cases)**

Анализаторите от дълги години използват потребителските случаи на употреба (use cases), за да опишат начините, по които потребителя взаимодейства със системата. Целта е да се извлекат релевантни и точни изисквания. Използването на потребителски

случаи възниква с напредъка на обектно-ориентирания подход към програмирането, подходът който е заложен в основата на подхода за проектиране и разработка на предлаганото решение.

- ✓ **Модел на потребителските случаи и процес на изготвяне на детайлната спецификация**

Описанието на потребителските случаи често не дава достатъчно пълна информация на екипа разработващ системата за функционалностите, които те следва да разработят, както и за това каква и коя информация следва да бъде достъпна. Събирането на всички изисквания от потребителските случаи може да доведе до формирането на твърде тромави и комплексни потребителски случаи. От друга страна, спирането на развитието на изискванията по време на стадия на събиране на потребителските изисквания ще доведе до много пропуски в информацията, от която екипът разработващ системата ще има нужда по време на стадия на конструирането ѝ. За да се понижи тази несигурност всеки потребителски случай трябва да бъде разработен заедно с пълните си системни изисквания.

Всеки потребителски случай на употреба води до определянето на няколко системни изисквания, които ще позволят на специфицирането на дадена функционалност в разработеното приложение и съответно няколко потребителски случая могат да формират спецификацията на една и съща функционалност или компонент.

Специфицирането на функционалностите, компоненти и изискванията към тяхното качество може да бъде описано чрез детайлни системни изисквания, които могат да бъдат асоциирани с потребителските случаи по няколко начина.

Подходът, който ще бъде избран зависи от това дали екипът, разработващ системата трябва да създаде дизайна, конструкцията и тестовете, базирайки се на потребителски случаи, на детайлни системни изисквания или на комбинация от двете. Избирайки подходящия подход е важно, да се избягва дублирането на информация от различни източници, което би направило спазването на изискванията много по-трудно.

- ✓ **Итеративно и инкрементално разработване на детайлната спецификация**

Екипът на Изпълнителя предлага итеративния и инкрементален подход, който е в основата на RUP да бъде приложен по същия начин към дейността по изготвяне на детайлната спецификация. Итеративният подход към изготвянето на детайлната спецификация включва пресяването на потребителските случаи посредством няколко итерации. Итеративните стъпки към събирането на изисквания от потребителски случаи е строго индивидуално според ситуацията. Предварително няма как да се определи точен брой итерации, които ще бъдат необходими, за да се покрият изискванията във всички ситуации. Но може да се каже, че спецификацията на итеративните изисквания винаги преминава през същите логически стъпки във всички ситуации.

8.1.2.1.Подход към реализация на Спецификация на изискванията и спецификация на потребителските случаи

Дейностите свързани с анализа на изискванията ще бъдат от ключово значение за успеха на разработката, тъй като Системата съдържа множество обвързани компоненти, които представляват сериозно предизвикателство пред използването на гъвкави подходи

от екстремното програмиране базирани на прототипиране и на изграждане на системата итеративно на малки „парчета". Потребителите на системата трудно могат да оценят изградения модел, ако той не е до голяма степен завършен, както и ако не е имплементирана съответна аналитична функционалност, която да покаже как потребителя консумира информацията. По този начин възможността за изграждане чрез итеративно подобряване на прототипи (throw away prototypes), не би била разумна от гледна точка на нужния ресурс и време за проектиране, разработка, зареждане с данни и тестване на тези прототипи. В тази връзка, избраният подход акцентира върху правилното специфициране на изискванията и максимално близка работа с екипа на Възложителя през целия процес на проектиране и разработка, така че да се постигне голяма степен на завършеност на реализацията преди пилотното използване, а след това да се премине към отстраняване на грешки и доразвиване на основната функционалност.

Предлаганата методология за разработка на процеса – RUP, поставя спецификацията на потребителските случаи (use case) в основата на процеса на проектиране, разработване и тестване на системата. Модела на потребителските случаи е гръбнака на формалното описание на системата, той се изготвя в процеса на спецификация на изискванията, като дори може да се каже че процеса на спецификация на изискванията бива иницииран чрез процеса на спецификация на потребителските случаи, тъй като дефинирането на use case сценарии е естествен начин за извличане на изискванията към системата.

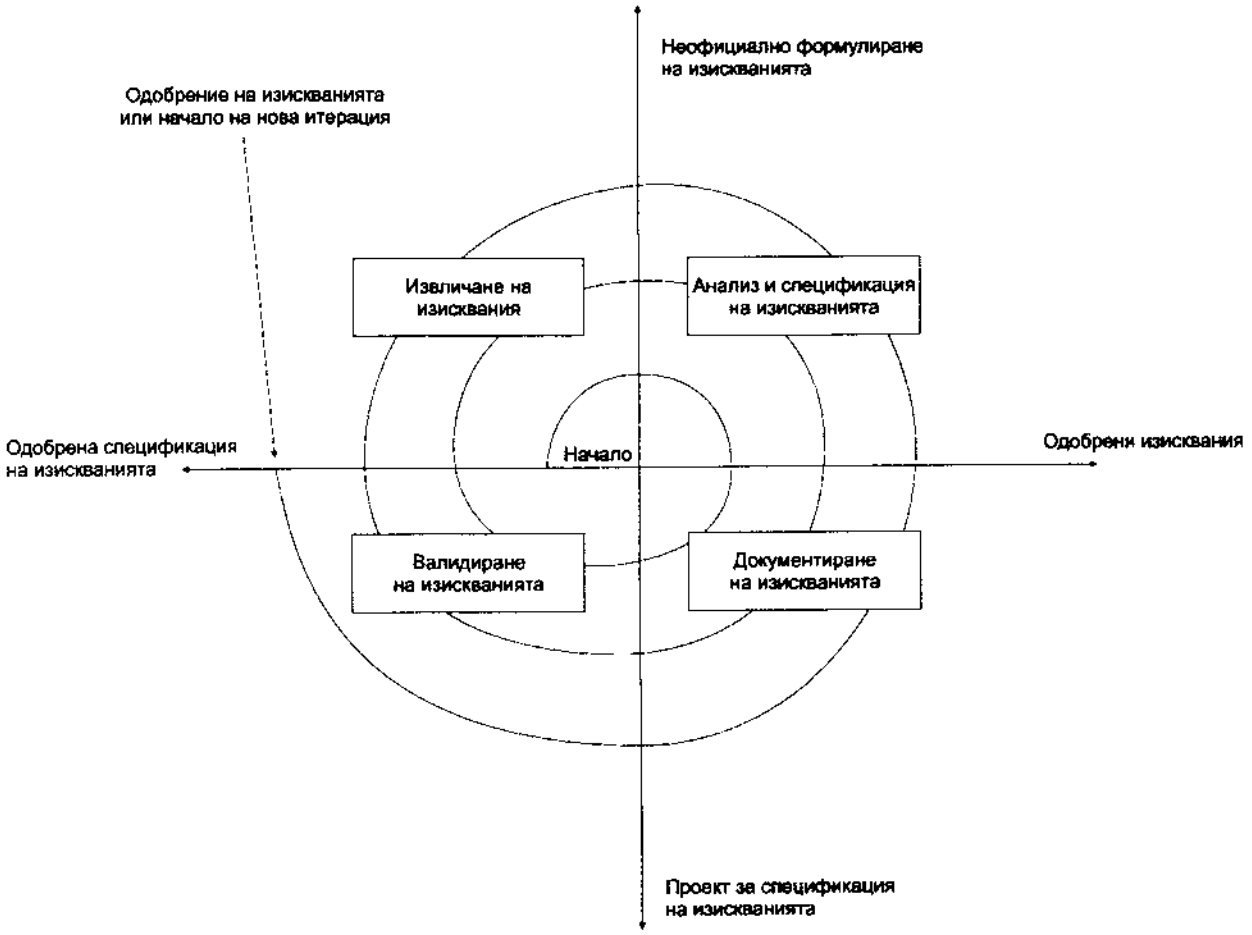
Първоначалния анализ на наличните източници на информация и срещи с екипа на Възложителят ще определят обхвата на модела на потребителските случаи, който следва да бъде правилно дефиниран в спецификацията за да се постигне пълното постигане на целите на проекта. Продуктът от тази дейност ще бъде изготвянето на спецификацията на изискванията както и на use cases, които да послужат като основа за проектирането и тестването на системата. Работата по спецификация на изискванията ще бъде надградена в процеса на изготвяне на модела на потребителските случаи (use case model), а по-късно и в етапа на проектиране, където ще се специфицира подхода за реализация на специфицираните изисквания.

При изпълнение на дейността по спецификация на изисквания ще бъде възприет итеративен подход към подготовката на детайлната спецификация. При този подход екипа на Изпълнителя ще извършва в итерации основните стъпки от процеса на формализиране на изискванията:

- Извличане на изискванията на база на изследване на идентифицираните източници на информация, както и на база на резултатите от проведените срещи и дискусии с екипа на Възложителя;
- Анализ и дефиниране на изискванията, в процес на обсъждане с Възложителя;
- Документиране на изискванията в проект на Спецификация на изискванията;
- Валидиране на изискванията, документирани в проекта на Спецификацията на изискванията, завършващо с обратна връзка от екипа на Възложителя.

На база на обратната връзка от Възложителя, в края на итерацията от стъпки, екипът от анализа или инициира наново процеса на извличане и анализ на изисквания, или се приема процеса на специфициране на изискванията за приключен.

Итеративния подход и съответстващите му дейности и резултати са описани графично на следната диаграма:



Фигура 15 - Диаграма на итеративния подход и съответстващите му дейности и резултати

8.1.2.2.Подход към осигуряване на качеството на продукта от дейностите по спецификация на изискванията

При осигуряването на качеството на извършените дейности ще бъде използван като референтен модел препоръките на IEEE - Reccomendation for SRS. Използвайки IEEE Standard 830, Recommended Practice for Software Requirements Specifications, ръководството на екипа ще съблюдава правилата и критериите за качество при изготвяне на спецификацията. Целта на въпросните действия по осигуряване на качеството е да се гарантира, че изготвения документ ще предложи стабилна и ясна рамка за проектирането и изграждането на технологичната реализация. Така специфицираните изисквания ще отговарят на изискванията за качество на спецификацията на изискванията:

- ✓ **Точност**

Този атрибут не е формално проверим в процеса на качествен контрол, но итеративния модел на валидиране на изисквания позволява на крайния потребител – екипът на Възложителя да провери документираните изисквания, по отношение на това как те представят описаната визия за изискванията към решението.

- ✓ **Недвусмисленост**

Недвусмислеността на изискванията ще бъде проверена чрез преглед на изискванията от трети лица, тъй като основните проблеми при изготвянето на изискванията идват в различното тълкуване на едно и също изискване от разработчиците и бъдещите потребители.

✓ **Пълнота**

Тук ще се приложи сравнение със стандартите за съдържание на спецификацията, така че да се сравни дали всички задължителни елементи присъстват и са опасни в нужната пълнота. При итеративния процес на подготовка, всички непълни или отсъстващите елементи ще бъдат ясно индикирани и коментирани, така че в крайния вариант да не се допусне одобрение без наличие на пълния набор от елементи.

✓ **Непротиворечивост**

Чрез използване на речник на термините и дефиниции за основните предположения ще се избегне конфликт между отделни части от документа за спецификация на изискванията, както и между спецификацията и други документи свързани с изпълнението на проекта.

✓ **Приоритизирани по важност**

Процеса на приоритизиране на изискванията следва да бъде извършен от екипа на Възложителя, така че от една страна да се обсъди аргументацията на приоритетите, така че екипът по изпълнение да възприеме логиката и да научи съответните ключови предположения, а от друга страна да се насочи вниманието и на екипа по разработка.

✓ **Проверими**

Работата по проверка на този индикатор за качество ще бъде изключително полезна в рамките на подготовката на тестовия план. Проверката се изразява в дефиниране на ясен критерий за проверка на всяко изискване.

✓ **Променими**

С цел лесната итеративна работа по подобрение на изискванията те ще бъдат документирани в съответствие със структурата на спецификация при спазване на ясни правила за идентифициране и реферирание на отделните изисквания, осигуряващи че всяко изискване се включва еднократно в спецификацията, а се референцира от другите части на документа, ако има зависимост с друго изискване. По този начин промените в документа не нарушават неговата консистентност.

✓ **Проследими**

С цел проследимост на развитието на изискванията, всяко изискване ще бъде идентифицирано с уникален номер, като се поддържа и информация за неговата идентификация в по-ранни версии на документа.

8.1.3. Методика за проектиране

Проектирането на компонентите ще бъде извършено от екипа на Изпълнителя съобразно съвременни, доказани и препоръчвани практики в областта на софтуерния дизайн, които са доказали своята приложимост при разработката на комплексни софтуерни системи.

В съответствие с RUP, процесът на проектиране ще бъде извършен като итеративен процес на дефиниране и подобряване на архитектурата, на дизайна на компонентите, интерфейсите и останалите характеристики на компонентите. В основата на дейността ще се състои в това изискванията към системата да се анализират с цел да се изработи визия за нейната вътрешна структура, която след това да се развие в цялостна архитектура, която да послужи за основа на разработката. Така разработената архитектура ще описва как системата се декомпозира и организира на отделни компоненти и интерфейсите между тези компоненти. След това детайлно се проектира и функционалността и интерфейса на всеки от компонентите, така че да се постигне необходимото ниво на детайлност на това описание до степен, която да позволи на разработчиците да извършат самото изграждане на компонента. Описаният подход , избран от екипа на Изпълнителя, се характеризира от следните основни дейности:

- Дизайн на софтуерната архитектура (дизайн на високо ниво) – описва се структурата и организацията на системата на високо ниво. Идентифицират се отделните компоненти, от които тя се състои.
- Дизайн на компонентите/модулите – всеки компонент/модул се описва с ниво на детайлност, което е достатъчно за неговото конструиране.

Резултатът от дейността по проектиране ще бъде логическия дизайн и организация е съвкупност от модели и артефакти, които съдържат/описват взетите важни решения в процеса на тяхното изготвяне.

Прилагането на гъвкава и модулна архитектура, структурирана по начин, който едновременно осигурява консистентност на данните и максимална защита от грешки, а от друга страна позволява лесно надграждане и разширяване на системата, без този процес да бъде съпътстван от регресивни грешки поради скрита свързаност на модулите, е изключително сложна задача. За постигне на нейните цели следва да се спазват основните принципи на обектно ориентирания дизайн.

8.1.3.1.Принципи за дизайн

В основата на добрият софтуерен дизайн е спазването на определени принципи за дизайн, които са доказали своята ефективност и ефикасност при множество успешни софтуерни системи:

- ✓ **Абстракция** – абстракцията може да бъде определена като процес на "забравяне" на информация, така че дадено множество от различни по своята същност неща да могат да бъдат третирани като еднакви. Абстракцията по същество представлява опростяване на проблемната област, като цената която се плаща е загуба на информация;
- ✓ **Свързаност и кохезия** – свързаността/обвързаността (copupling) дефинира силата на връзката между отделните модули, докато кохезията (cohesion) дефинира доколко вътрешните елементи на даден модул са взаимно (логически) свързани и служат за изпълнението на неговите функции. При наличие на висока свързаност между модулите, ако трябва да се направи изменение в даден модул е твърде вероятно това да наложи промени и в свързаните модули. Обратно при ниска свързаност промените обикновено се ограничават до модула, който

реализира изменената функционалност. Високата кохезия предполага, че логически свързаните елементи са групирани заедно. Това спомага да се намали сложността, тъй като се елиминират сложните последователности от взаимодействия между различните модули. В идеалния случай модулите, от които се състои дадена софтуерна система трябва да имат ниска свързаност и висока кохезия.

- ✓ **Декомпозиция и модулност** – големите и сложни софтуерни системи е добре да се декомпозират на отделни под-системи и модули, като целта е да се разположат различните функционалности и отговорности в различни компоненти. По този начин се постига разделяне на отделни компоненти, чиято сложност е управляема.
- ✓ **Енкапсулация** – енкапсулацията (encapsulation/information hiding) представлява групиране и пакетиране на елементите и вътрешните детайли на дадена абстракция, правейки ги недостъпни отвън. Целта е да се постигне потенциал за промяна: вътрешните механизми на компонента могат да бъдат променяни и подобрявани без това да оказва влияние на останалите компоненти. Обикновено се енкапсулират онези аспекти на компонента, които има вероятност да бъдат променяни в бъдеще.
- ✓ **Разделяне на интерфейсите от имплементацията** – принципът е свързан с принципа на скриване на информацията. Интеракцията с дадена абстракция се осъществява само чрез строго дефинирани публично достъпни интерфейси, вътрешната структура остава скрита.
- ✓ **Достатъчност, пълнота и простота на дизайна** – постигането на достатъчност, пълнота и простота означава, че софтуерния компонент обхваща всички важни характеристики на абстракцията, която реализира и нищо повече.

8.1.3.2.Шаблони за дизайн (Design patterns)

При обектно ориентирания дизайн стремежът е системата да се планира като съвкупност от взаимодействащи обекти. Всеки обект представя даден елемент от системата, която се моделира и се характеризира със своето състояние и поведение. Обектът представлява абстракция групираща в себе си данни и процедури. Интеракцията с обекта става посредством добре дефиниран интерфейс на обекта. Ключова роля при обектно ориентирания дизайн играят понятията наследяване и полиморфизъм.

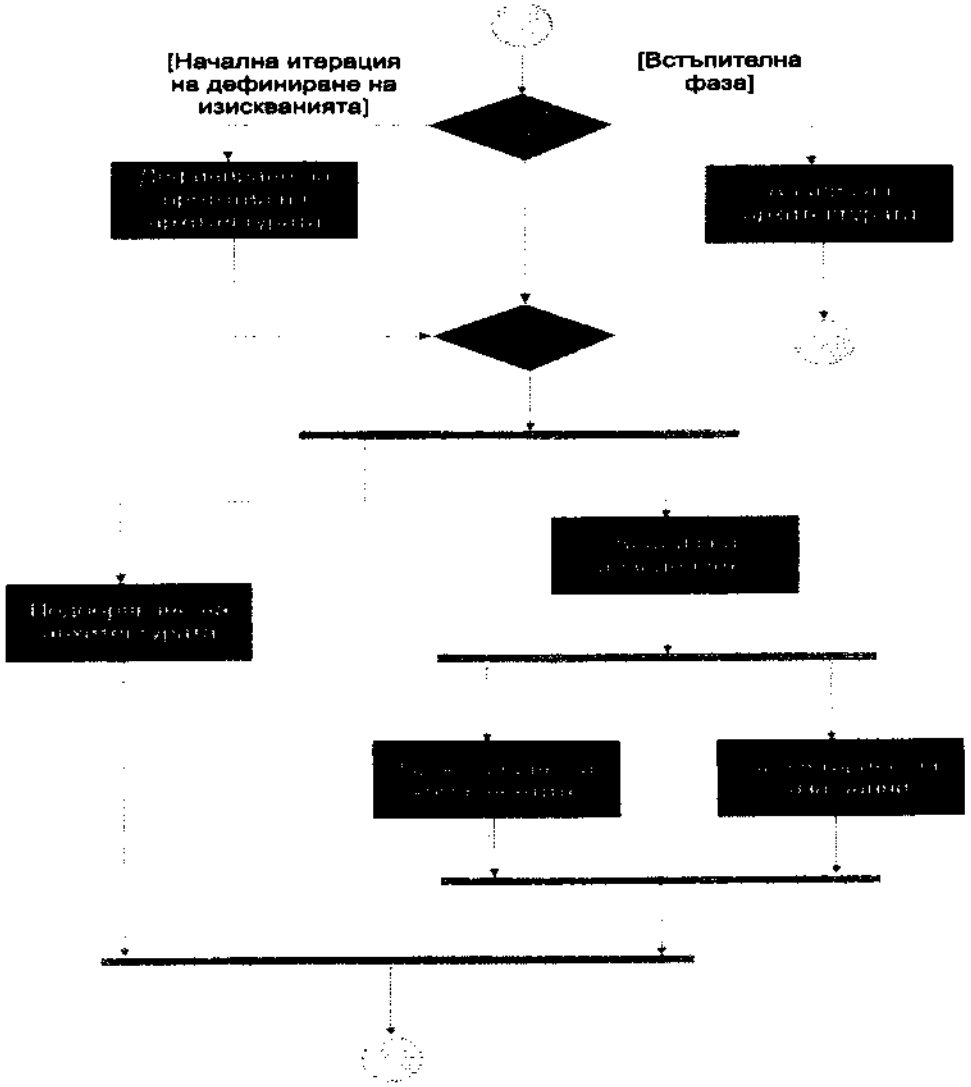
Подходът при обектно ориентирания дизайн е проблемния домейн да се декомпозира на отделни обекти следвайки принципите на дизайн от общото към частното и свързване на частите съобразно техните отговорности. Обектно-ориентирания анализ и дизайн по същество е процес на последователни действия на опростяване т.е. оперирайки с проблемния домейн в него се „инжектират" структури, които го декомпозират и опростяват. Структурите, които служат за декомпозиция и опростяване, представляват шаблони за дизайн (design patterns), които са доказали своята ефективност при много различни ситуации. Докато различните архитектурни стилове за софтуерен дизайн представляват шаблони от високо ниво (определят макроархитектурата), съществуват шаблони за дизайн описващи детайлите на по-ниско, локално ниво (определят микроархитуктурата). Такива видове са: Creational patterns

(builder, factory, prototype, singleton), Structural patterns (adapter, bridge, composite, decorator, façade, flyweight, proxy), Behavioral patterns (command, interpreter, iterator, mediator, memento, observer, state, strategy, template, visitor).

8.1.3.3.Приложение на итеративния подход в обектно – ориентирано проектиране

Основните методи в процеса на разработка на софтуер, в предлаганата от Изпълнителя методология – процес за разработка на софтуер RUP, се базират изключително на итеративния обектно-ориентиран подход на анализ и дизайн на системата. В началните итерации артефактите ще бъдат предимно документи, описващи изискванията и съдържащи аналитични и технически UML модели. Следващите итерации ще създадат софтуерни версии, които реализират специфицираните изисквания. Последните итерации ще се съсредоточат върху тестове, отстраняване на програмни грешки, внедряване на софтуера и евентуално очертаване на бъдещото й развитие.

На диаграмата е показана архитектурата на процесите за обектно ориентиран анализ и проектиране:



Фигура 16 - Архитектурата на процесите за обектно ориентиран анализ и проектиране

В началото фокусът се насочва върху изграждането на първоначална архитектура на компонентите (Define a Candidate Architecture), за да може да бъде дадена отправна точка за главната работа по анализ. В случай, че вече съществува архитектура, която е била изготвена на предишни итерации, в предишни проекти или зададена като изискване, фокусът на работата се измества върху нейното актуализиране (Refine the Architecture). Следващата стъпка е да се анализира очакваното поведение на системата и да се специфицират конкретните елементи, които изграждат това поведение (Analyze Behavior).

След като първоначалните елементи бъдат идентифицирани, следва да бъдат допълнително детайлизирани в етапите на дизайн на компонентите и базата данни. Дизайна на компоненти (Design Components) изготвя множество компоненти, които доставят изискваното поведение на системата. Паралелно с това, анализ на начините за съхранението им в базата данни бива извършен в дизайна на базата данни (Design the Database).

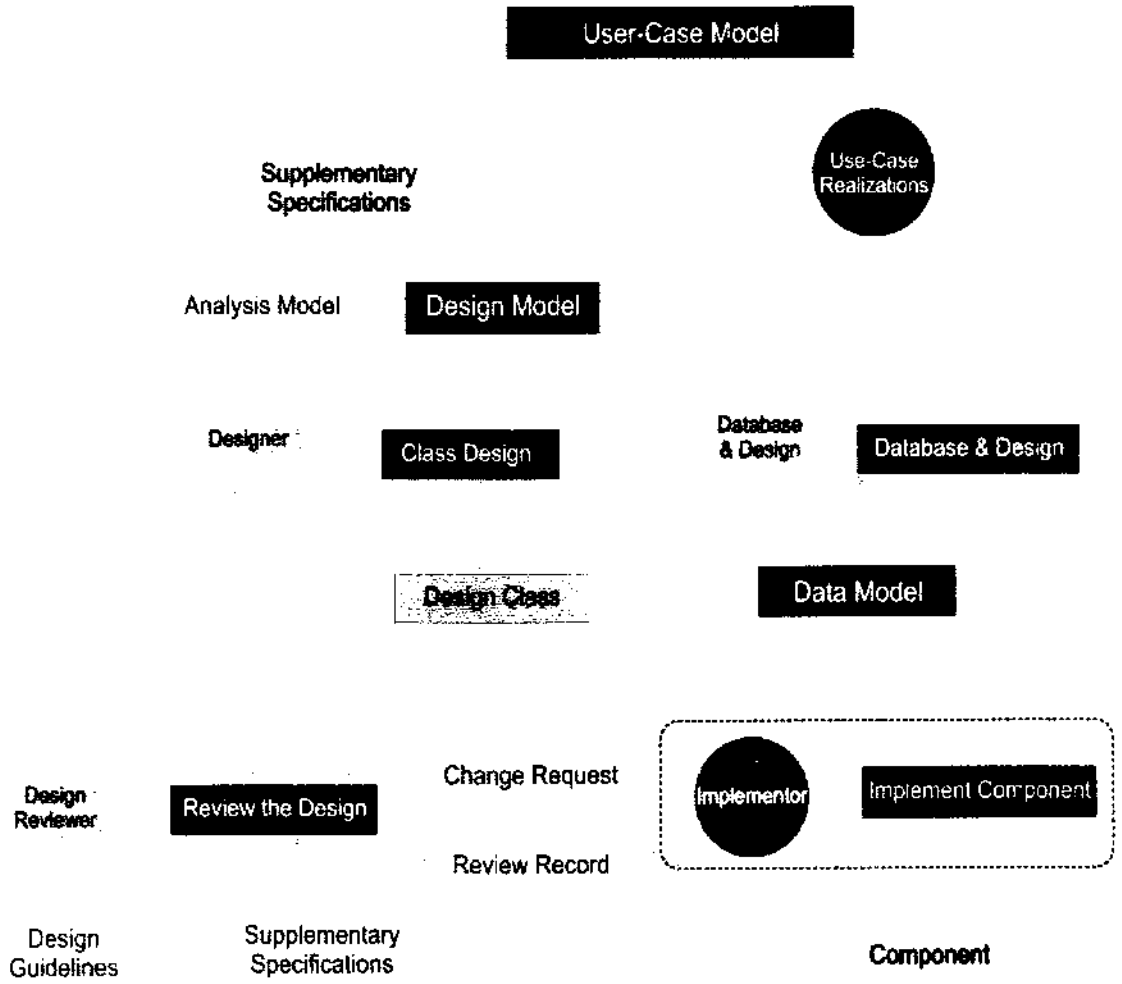
Реализацията на тази концепция и постигането на обектно-ориентиран анализ и дизайн на системата се извършва на базата на готовите дефинирани изисквания под

формата на Модел на потребителските случаи (Use Cases), от които се изгражда тяхната реализация под формата на Клас диаграми (Class diagrams) и Диаграми на последователността и взаимодействието (Sequence diagrams и Collaboration diagrams, съответно). Това е имплементационния модел на системата, който включва още Дизайн модел и Модел на данните.

Дизайн моделът е обектен модел, който описва реализацията на потребителски случаи и служи за извеждане на Модела на имплементацията и неговия програмен код.

Моделът на данните е подмножество на имплементационния модел, което описва логическия и физически вид на постоянните (персистентни) данни в системата. Той включва и видовете поведения в базата данни, например записани процедури, активатори, ограничения и др.

На диаграмата е показана процесната архитектура за дизайн на компонентите:



фигура 17. Диаграма на процесната архитектура за дизайн на компонентите

Използването на модели ще позволи лесна промяна и бъдещо разширение на съществуващите услуги и добавяне на нови електронни услуги.

8.1.3.4.Методика за проектиране на потребителските интерфейси

Потребителският интерфейс ще бъде изграден чрез използването на стандартите HTML 5 и CSS 3. Всички уеб страници ще бъдат изградени така, че структурата и съдържанието на страницата (HTML) да бъдат в максимална степен отделени от тяхното визуално представяне (CSS). Това ще позволи при необходимост лесната модификация на визуалните елементи.

При изграждането на потребителският интерфейс ще бъде използван подхода Responsive Web Design (RWD), който ще осигури гъвкавост и възможност за оптимално представяне на данните върху различните браузъри. За реализирането на богат, динамичен и удобен за работа потребителски интерфейс, който не изисква често презареждане на съдържанието от сървъра, презентационния слой ще бъде реализиран като Single Web Application. Ще бъдат използвани библиотеки за динамично уеб съдържание jQuery (http://jquery.com/), KnockoutJS (http://knockoutjs.com) или AngularJS (http://angularjs.org/). Конкретната библиотека, която ще бъде използвана за динамично уеб съдържание ще бъде избрана съобразно конкретните изисквания към потребителския интерфейс дефинирани в резултат на анализа.

При проектирането на потребителския интерфейс ще бъде използван подхода залегнал в стандарта Human centered design processes for interactive systems (ISO 13407) и препоръките на W3C в Web Content Accessibility Guidelines 2.0 (ISO/IEC 40500:2012).

Потребителският интерфейс ще спазва следните добри практики:

- ✓ Интерфейсът предоставян от всеки модул ще бъде ориентиран към реализираните от него функции. Когато потребителят има нужда от достъп до информация в друг модул ще бъде предвидена възможност за пряк достъп до нея без да е необходима навигация през менютата на системата
- ✓ Опциите в менютата и достъпната функционалност в интерфейса на системата ще зависят от ролята и правата за достъп на потребителя, екранните форми няма да бъдат утежнени с излишни функции
- ✓ Потребителят ще бъде информиран от системата за успеха или неуспеха на изпълняваните от него операции
- ✓ Логически свързаните полета ще бъдат групирани;
- ✓ Всяко поле ще бъде означено с етикет подсказващ неговата функция;
- ✓ Полетата ще бъдат разположени на екрана, така че да се ограничи дължината на вертикалния скрол. Хоризонтален скрол ще се допуска по изключение;
- ✓ Всяка форма извършваща промяна в данните ще притежава бутон "Отказ", при натискането на който ще се отменя действието
- ✓ Съобщенията за грешка във въвежданите данни ще съдържат ясна информация за вида на грешката и очаквания формат на данните.

8.1.4. Методология за разработка

Предлаганата методология за разработка се основава на софтуерната спецификация на изискванията и софтуерния дизайн на компонентите, както и на изготвените в рамките на изпълнението му план за тестване и тестовите сценарии за контролиране на качеството на софтуера, част от паралелния нему процес по контрол на качеството на разработката.

Цялостната производствена процедура обхваща няколко паралелни процеса и може да бъде обобщена в следните стъпки:

- ✓ Стъпка 1 - Разработване на софтуерните модули (units);
- ✓ Стъпка 2 - Разработване на план за тестване и тестовите сценарии;
- ✓ Стъпка 3 - Разработка на модулните тестове (unit tests)
- ✓ Стъпка 4 - Проверка (тестване) на софтуерните модули;
- ✓ Стъпка 5 - Интегриране (build) на софтуерните модули;
- ✓ Стъпка 6 - Разработване и провеждане на интеграционни тестове (integration tests);
- ✓ Стъпка 7 - Зареждане на системата с начални данни;
- ✓ Стъпка 8 - Разработване и провеждане на системни тестове;
- ✓ Стъпка 9 - Произвеждане на крайния продукт и документация.

8.1.1.1.Методика за разработване на модела на данни

Предложената методика за разработване на модела на данните в системата обхваща всички етапи от жизнения цикъл на проектирането на базата данни, като отчита спецификите на отделните приложения с които трябва да се интегрира системата. Една от основните специфики на приложенията, която ще бъде взета под внимание е тяхната архитектура – някои от приложенията с които се интегрира системата са разпределени, а други са централизирани, в допълнение всяко от тях е реализирано на базата на различни технологии и платформи.

Основните етапи от жизнения цикъл на разработването на модела на данни са следните:

- Проучване и анализ на изискванията
- Изграждане на логически модел
- Проектиране на базата данни

По-надолу ще развием в детайли всеки един от тези етапи.

- ✓ **Проучване и анализ на изискванията**

Основната цел на този етап от разработването на модела данни е да се определят потребителските изисквания към базите данни. За целта е необходимо да се идентифицират компонентите и приложенията, взаимодействащи си със системата. Това са всички разработени приложения, които ще бъдат интегрирани със системата, както и възможните нови такива.

На този етап трябва ясно да се определят нуждите на потребителите, изискванията към системата и ограниченията върху решението.

Изследването на системата е особено важен етап в процеса на цялата разработка. Допуснатите пропуски на този етап рефлектират върху пълнотата на проекта. Най-общо е необходимо да се проведе проучване, както бе описано в предходната точка в следните насоки:

- Функционирането на съществуващата система и несъвършенствата в нея;
- Икономическата и организационна среда на системата и възможните промени в бъдеще;
- Проблемите, слабите места и ограниченията на системата, както и породилите ги причини;
- Основните процеси и начина на тяхното реализиране;
- Типовете данни и техния обем;
- Справките и отчетите, необходими за функциониране на системата;
- Потребителските изисквания и възможностите за тяхната реализация.

По време на проучването и анализа е важно да се осмислят взаимовръзките между отделните компоненти на системата. Указаните области на проучване не се изследват последователно. Те са дадени в този ред за по-голяма прегледност. При изследването е важно да се постигнат крайните резултати, т.е. да се опишат основните потребителски изисквания и да не се пропуснат някои важни детайли.

✓ **Изграждане на логически модел**

Когато се проектира една база данни, е необходимо да се вземат решения по отношение на създаването на най-удобния модел на дадена система от реалния свят. При създаване на модела на данните се извършва организиране на изискванията на системата в едно логическо представяне на базата данни. Моделът на данните се състои от обекти и техните атрибути и ограничения, дефиниции на релации между обектите и ограничения върху тези релации, следователно разработването на логически модел на данните се осъществява чрез определяне на обектите, техните атрибути и ограничения и техните релации. Създаденият по този начин модел позволява да се идентифицират съответните таблици, които трябва да бъдат създадени, колоните, съдържащи се в тях, релациите между таблиците.

- Определяне на обектите и техните колони – когато се определят обектите, е необходимо, от изискванията на системата да се дефинират основните логически подразделения на информацията. Докато се разглеждат изискванията на системата, се дефинират основните обекти и събития. В резултат на това се добавят таблици към дизайна на базата данни, които им съответстват.

След като са дефинирани всички таблици, които могат да бъдат определени в този момент, трябва да се дефинират колоните на тези таблици. Тази информация се взема директно от изискванията на системата, в които е определено какви данни трябва да се поддържат за обектите и събитията.

- ✓ **Определяне на релациите между обектите** – по време на процеса на дефиниране на релациите между таблиците може да се открие, че е необходимо да се промени създадения до този момент дизайн. Започва се с избиране на една от основните таблици и определяне на обектите, които имат релации към тази таблица.

След като се установят релациите между таблиците, трябва да се дефинира типа на тези релации. Всяка линия се маркира в двата си края или с цифрата 1, или със символа ‚n'. Цифрата 1 се отнася до страната едно на релацията, а символът ‚n' се отнася до страната много на релацията. За да се определи типа на релацията, се разглеждат данните, които съдържат таблиците и видовете взаимоотношения между тях. В един нормализиран проект на базата данни обаче релациите много към много трябва да се модифицират, като се добави една свързваща таблица и между нея и във всяка първоначална таблица да се създадат релации едно към много. Аналогично се определят и типовете на останалите релации.

- ✓ **Определяне на ограниченията върху данните** - работните правила включват всички ограничения за една система, включително за целостта на данните и сигурността.

На този етап от процеса на изграждане на модела на данните трябва да се обърне внимание на спецификите и ограниченията върху данните. Например дадено поле може да не е задължително във формата, а да е позволено да се попълни на по-късен етап. В такъв случай полетата, които съдържат информация, която не е задължителна, да бъдат маркирани като опционални, т.е. да позволят съхраняване на стойност NULL в тези полета. От друга страна, ако разгледаме поле, което е задължително за попълване, колоната трябва задължително да има попълнена стойност и тя да съществува като стойност в колоната ID от съотвяетната таблица.

8.1.1.2.Проектиране на базата данни

Теорията на проектиране на релационни бази данни се състои от следните основни понятия:

- Таблици и уникалност - Първичен ключ (primary key) уникален идентификатор на ред, който представлява колона или група от колони, използвани за разграничаване всеки отделен ред от останалите редове в таблицата. Може да бъде прост или съставен. Простият ключ е създаден от една колона, съдържаща уникални стойности за всеки ред от таблицата.

Всяка таблица може да има само един първичен ключ, дори когато няколко колони или комбинации от колони съдържат уникални стойности (наречени алтернативни ключове).

Изборът на първичен ключ трябва да се основава на принципите за:

  - минималност (избират се толкова колони, колкото е необходимо);
  - стабилност (избират се колони, които рядко биват променяни);
  - простота (от колкото е възможно по-прост тип)
- Външни ключове и домейни - Външен ключ (foreign key) колона или група от колони в дадена таблица, представляваща връзка с друга таблица посредством нейния първичен (родителски) ключ.

Родителски ключ е ключът, към който сочи един външен ключ. Родителският ключ трябва да бъде уникален идентификатор, за да може да се определи към кой ред от таблицата на родителския ключ сочи външният ключ. Броят и типът на колоните, съставящи външния ключ, трябва да съответства на броя и типа на колоните на родителския ключ, но могат да се използват различни имена за тях. Не е задължително стойностите на външния ключ да бъдат уникални в своята-собствена таблица. Те трябва да бъдат в същата област от допустими стойности (домейн), на която принадлежат стойностите на родителския ключ.

Домейн съвкупност от стойности, които са допустими за дадена колона.

- Релационни връзки - Релация (relationship) се нарича връзка между таблици, която е базирана на първичен ключ от едната таблица и външен ключ от другата таблица.

Съществуват три типа релации между таблиците: релации едно към едно; релации едно към много; релации много към много.

Релации едно към едно - две таблици А и В са свързани с релация едно към едно, ако за всеки ред от таблицата А има най-много един съответстващ ред от таблицата В, но всеки ред от таблицата В може да има точно един съответстващ ред от таблица А.

Релации едно към много - две таблици А и В са свързани с релация едно към много, ако за всеки ред от таблицата А има нула или повече съответстващи редове от таблицата В, но всеки ред от таблицата В може да има точно един съответстващ ред от таблицата А.

Релации много към много - две таблици А и В са свързани с релация много към много, ако за всеки ред от таблица А има много съответстващи редове от таблицата В и обратно. Този тип релация се създава, като се дефинира трета таблица, наречена свързваща таблица (junction table), която съдържа първичните ключове от двете таблици като външни ключове; първичният ключ в свързващата таблица е съставен от двата външни ключа.

- Правила за запазване на целостта на данните - целостта на данните е важно понятие за проектиране на базите данни. Има четири вида цялост на данните:
    - цялост на обект – едно от изискванията на проектирането на релационна
    - база данни е възможността да се разграничат различните инстанции на даден обект. Това понятие е известно като цялост на обект и се реализира чрез създаване на първичен ключ. Според това правило за цялост, колоните съставящи първичния ключ, не могат да имат стойност NULL. Релационните бази данни поддържат специална стойност NULL, която указва неизвестните стойности (unknown);
    - цялост на област – свързана е с осигуряване на валидност на стойностите на колоните, т.е. да принадлежат на допустима област от стойности. Реализира се с определяне на типа на колоните, допускане на стойност NULL, ограничения, стойност по подразбиране, дефиниране на външен ключ;
    - цялост на връзка – запазва дефинираните отношения (релации) между таблиците, когато се въвеждат, променят или изтриват редове. Целостта на връзките гарантира, че съществува съгласуваност на стойностите на

ключовете между таблиците – първичните и външните в съответните таблици. Реализира се с дефиниране на ограничението външен ключ. Когато се реализира цялост на връзките, не се допуска да бъдат добавени редове в една таблица, която е страната "много" на релацията, ако в първичната таблица, която е страната "едно" на релацията, липсва съответен ред. Също така не се допуска да се променят стойностите на колоните на първичния ключ в една таблица, която е страната "едно" на релацията, ако в свързаната таблица има поне един съответен ред. Не се допуска да се изтриват редове от една първична таблица, ако има свързани редове в таблицата с външните ключове;

- дефинирана от потребителя цялост – дава възможност за определянето на специфични бизнес правила, които не могат да се отнесат към някоя от другите категории цялост. Реализира се чрез създаване на ограничения, съхранени процедури и тригери.

8.1.1.3.Разработване на софтуерните модули (units)

Софтуерните модули се разработват от програмистите на базата на софтуерните спецификации и дизайн, при спазване на вътрешните конвенции за кодиране.

Планът на проекта е основата за наблюдение на дейностите и за предприемане на коригиращи действия. Развитието на проекта се контролира главно чрез сравняване на реално получения продукт и усилията и средствата, вложени в изпълнението на задачата с времевата диаграма (schedule) на плана по зададените вътрешни срокове или контролни точки от структурирането на задачите. Когато изпълнението се отклонява съществено от плана, трябва да се предприемат подходящи коригиращи действия. Едно отклонение е значително, когато, ако се остави нерешено, ще възпрепятства постигането на целите на проекта. Коригиращите действия включват:

- Промяна на начина на работа;
- Добавяне на ресурси;
- Смяна на процесите;
- Преразглеждане на рисковете;
- Промяна на изискванията;
- Преразглеждане на оценките и плановете;
- Предоговаряне на споразуменията.

8.1.1.4.Интегриране (build) на софтуерните модули

Завършените софтуерни модули се интегрират от програмистите в цялостен софтуерен продукт на базата на софтуерните спецификации и дизайн, при спазване на вътрешните конвенции за кодиране. Тук се реализират и интерфейсите за интеграция с външните за софтуерния продукт системи.

Както и при разработването на софтуерните модули и тук планът на проекта е основата за наблюдение на дейностите и за предприемане на коригиращи действия.

8.1.5. Методика за внедряване

За изпълнение на дейностите с необходимото качество, в срок и в съответствие с изискванията по проекта, при планиране ще бъдат предприети действия за определяне на:

- процедурата за внедряване, включително подготвителните действия за внедряване на системата;
- ресурсите, необходими за изпълнение, включително екипа от експерти;
- спецификация на тестовете за приемане на системата в окончателен вариант;
- процедура за изграждане на продукционната среда;
- график за изпълнение на дейностите.

Организацията на изпълнение като минимум ще включва:

- разработване на инструкцията за изпълнение на процедурата за внедряване;
- подготовка на експертите, включени в екипа за внедряване, за изпълнение на дейностите по процедурите за изграждане на продукционна среда и за внедряване;
- разработване на образците на документи за отчитане изпълнението на дейностите.

Всички дейности във фазата на планирането ще бъдат съгласувани с Възложителя.

Резултатите от дейностите по планиране внедряването на системата ще бъдат обобщени в План за внедряване, който ще бъде представен на Възложителя за одобрение – на хартия и в електронен формат. Планът за внедряване като минимум ще включва:

- Подготовка за внедряване – описание на подготвителните дейности, които трябва да бъдат извършени преди внедряване;
- Процедура за внедряване – детайлно описание на дейностите с подробни указания за изпълнение на всяка стъпка;
- Процедура за изграждане на продукционна среда с подробно описание на действията за инсталация и конфигуриране на компонентите на системата;
- Екип за изпълнение – списък на експертите, включени в екипа за изпълнение и данни за контакт (стационарен телефон, мобилен номер, е-мейл адрес, административен адрес за кореспонденция и т. н.);
- Спецификация на тестовете за приемане на системата;
- Образец на протокол за изграждане на продукционна среда;
- Образец на протокол от проведени тестове за приемане на система, в който ще се отразяват резултатите от изпълнение на тестовете;
- Образец на протокол за внедряване за отчитане изпълнението на дейностите;

- График за изпълнение.

✓ **Изграждане на продукционна среда**

Изграждането на продукционната среда ще бъде извършено по начина, определен във фазата на планиране на внедряването и в съответствие с разработената процедурата за изграждане на продукционна среда.

✓ **Изпълнение на дейностите по внедряване**

При внедряване на компонентите ще бъдат извършени действията, определени във фазата на планиране на внедряването, в съответствие с разработената процедура за внедряване.

Дейностите ще бъдат извършени по одобрените от Възложителя План за внедряване и график за изпълнение и по предварителна оценка ще включват:

- параметризация на базата данни, включително създаване на системните класификатори, конфигуриране на системата;
- зареждане в базата данни на мигрираните данни;
- провеждане на тестове за приемане на системата, в хода които ще се извърши и представяне пред екип на Възложителя на реализираните функционалности, което ще гарантира навременно запознаване със системата.

✓ **Отчитане изпълнението на дейностите по внедряване**

Отчитане изпълнението на дейностите по внедряване ще бъде включено в междинния доклад за изпълнение на Етап 4: Внедряване, в който ще бъдат изложени поставените цели и постигнатите резултатите. Ако в хода на изпълнение са възникнали проблеми, те ще бъдат документирани. Ще бъдат описани и предприетите действия за преодоляването им. Към доклада ще бъдат приложени оригинали на:

- протокол за изграждане на продукционна среда;
- протокол от изпълнението на дейностите по внедряване;
- протокол от проведените тестове за приемане на системата.

8.1.1.5.Начин на прилагане на предлагания подход за внедряване на компонентите

✓ **Планиране на внедряването и организация на изпълнението**

Във фазата на планиране на внедряването ще бъдат предприети следните конкретни стъпки за подготовка и изпълнение на дейностите:

- ще бъде разработена процедурата за внедряване с указания за изпълнение на всяка стъпка;
- ще бъдат определени ресурсите, необходими за изпълнение на дейностите в срок и с необходимото качество;
- ще бъде сформиран екип за изпълнение – експерти, които ще извършат дейностите по внедряване;
- ще бъде извършен преглед на изготвената при планиране на тестването

Спецификация на тестовете за приемане на системата, включително и на изготвената процедура за връщане на системата в последно работоспособно състояние (Rollback Procedure) и ако е необходимо, ще бъде актуализирани обхвата и вида на планираните тестовете за приемане;

- ще бъде разработен образец на протокол от проведени тестове за приемане на системата, в който ще се отразяват резултатите от тестовете;
- ще бъдат разработена процедурата за създаване на продукционната среда с подробно описание на последователността от действия;
- ще бъде подготвен образец на протокол за приемане на дейностите по изграждане на продукционната среда;
- ще бъде подготвен образец на протокол за внедряване за отчитане изпълнението на дейностите по внедряване;
- ще бъде изготвен детайлен график за изпълнение на дейностите по:
    - изграждане на продукционната среда;
    - внедряване и провеждане на тестове за приемане, съпроводени с представяне пред екип на Възложителя на реализираните функционалности с конкретна информация за период на изпълнение и експертите, които ще извършат дейностите.

Всички дейности във фазата на планирането ще бъдат съгласувани с Възложителя, като част от тях ще бъдат изпълнени със съдействието на екипа на Възложителя.

Резултатите от дейностите по планиране внедряването на системата ще бъдат обобщени в План за внедряване, който ще бъде представен на Възложителя за одобрение, в 2 екземпляра на хартия и в редактируема електронна форма ('.doc' или '.docx' файлов формат). Планът за внедряване като минимум ще включва:

- Подготовка за внедряване – описание на подготвителните дейности, които трябва да бъдат извършени преди внедряване;
- Процедура за внедряване – детайлно описание на процедурата и дейностите, които ще бъдат извършени, с подробни указания за изпълнение на всяка стъпка;
- Процедура за изграждане на продукционна среда с приложено ръководство за администратора за инсталация и конфигуриране на системата с подробни указания от типа „стъпка по стъпка" за последователността от действия за подготовка на продукционната среда
- Екип за изпълнение – списък на експерти с данни за контакт (телефон, мобилен номер, е-мейл адрес, адрес за кореспонденция и т. н.);
- Спецификация на тестове за приемане на внедряването, включително и на изготвената процедура за връщане на системата в последно работоспособно състояние (Rollback Procedure);

- Образец на протокол от проведени тестове за приемане на внедряването, в които ще се отразяват резултатите от тестовете;
- Образец на протокол за внедряване, който ще послужи за отчитане изпълнението на дейностите по внедряване;
- Процедура за създаване на продукционната среда с приложено ръководство;
- Образец на протокол за отчитане изпълнението на дейностите по изграждане на продукционната среда;
- Детайлен график за изпълнение на дейностите по внедряване.

✓ **Изграждане на продукционна среда**

Изграждането на продукционната среда, в която ще функционира Системата, ще се извърши в съответствие с одобрения План за внедряване и разработената Процедура за създаване на продукционна среда, която като минимум ще включва следните действия:

- инсталация и настройка на операционната система и друг системен софтуер;
- инсталация на компонентите (софтуерни компоненти и база данни);
- настройки на конфигурационните файлове;
- изпълнение на сервизни скриптове, ако е необходимо и т. н.

За изпълнение на дейностите по изграждане на продукционната среда ще бъде изготвен протокол.

✓ **Изпълнение на дейностите по внедряване**

Цялостното внедряване на компонентите ще бъде проведено по одобрените от Възложителя План за внедряване и детайлен график за изпълнение на дейностите.

Процедурата по внедряване на компонентите по предварителна оценка като минимум включва:

- параметризация на базата данни, включително създаване на системните класификатори, конфигуриране на системата;
- зареждане в базата данни на „плоски файлове“ с агрегирани данни;
- провеждане на тестове за приемане на внедряването и отразяване на резултатите в съответния документ.

Тестовете за приемане на внедряването ще бъдат извършени съвместно с екипа на Възложителя, като резултатите ще бъдат отразени в протокол.

✓ **Процедура за създаване на продукционната среда**

Описание ще включва подробна информация за процедурата и действията по създаване на продукционната среда. Към описанието ще бъде приложено ръководство за инсталация и конфигуриране за администратора с подробни указания стъпка по стъпка за действията, които трябва да бъдат извършени.

Действията по изграждане на продукционната среда, в която ще функционира системата като минимум ще включват:

- инсталация и настройка на операционната система;
- инсталация на система (софтуерни компоненти и база данни);
- параметризация на базата данни;
- настройки на конфигурационните файлове;
- изпълнение на скриптове, ако е необходимо и т. н.
- подпис на протокол за отчитане изпълнението на дейностите по изграждане на продукционната среда.

В хода на планиране на внедряването с Възложителя ще бъде обсъден подход за параметризация на системата.

Процедурата и дейностите за изграждане на продукционната среда ще бъдат специфицирани при планиране на внедряването, ще бъде разработено и ръководството с детайлна информация за действията, които следва да бъдат извършени с подробни указания от типа „стъпка по стъпка". Процедурата и приложенията ще бъдат представени на Възложителя за одобрение като част от Плана за внедряване.

✓ ***Образец* на *протокол за изграждане* на *продукционна среда***

Най-общо протоколът ще съдържа информация за:

- Възложител;
- Място на изпълнение на дейностите по изграждане на продукционна среда;
- Представител(и) на Възложителя, присъствал(и) и осъществил(и) контрол на изпълнение на дейностите по изграждане на продукционна среда;
- Експерт(и), извършил дейностите по изграждане на продукционна среда;
- Период на изпълнение;
- Извършени действия – по процедурата за изграждане на продукционна среда и допълнителни дейности, ако е възникнала необходимост от такива;

Образецът ще бъде разработен в окончателен вид при планиране на внедряването и ще бъде представен на Възложителя за одобрение като част от Плана за внедряване.

✓ ***Образец на* протокол *за провеждане на* тестове *за приемане на системата***

Най-общо протоколът ще съдържа информация за:

- Възложител;
- Място на изпълнение на тестовете за приемане на системата;
- Представител(и) на Възложителя, участвал(и) в провеждането на тестовете за приемане на системата;
- Експерт(и), провели тестовете за приемане на системата;
- Таблица с планираните тестове, като се всеки тест ще има описание на критерият, въз основа на който се оценява резултатът от изпълнение. Например:

№ по ред	Тест	Критерий за успешно изпълнение	Получен резултат (да/не)
1	2	3	4
1.			
2.			
3.			
4.			

Забележки:

1. Колони 1-3 се попълват от Изпълнителя при планиране на тестовете за приемане на системата.

2. Колона 4 се попълва от представители на Възложителя при провеждане на тестването.

Образецът ще бъде разработен в окончателен вид при планиране на внедряването и ще бъде представен на Възложителя за одобрение като част от Плана за внедряване.

✓ ***Образец на* протокол *за внедряване***

Най-общо протоколът ще съдържа информация за:

- Възложител;
- Място на изпълнение на дейностите по внедряване;
- Представител(и) на Възложителя, присъствал(и) и осъществил(и) контрол на изпълнение на дейностите по внедряване;
- Експерт(и), извършил дейностите по внедряване;
- Период на изпълнение;
- Извършени действия – по процедурата за внедряване и допълнителни, ако е установен необходимост от такива;

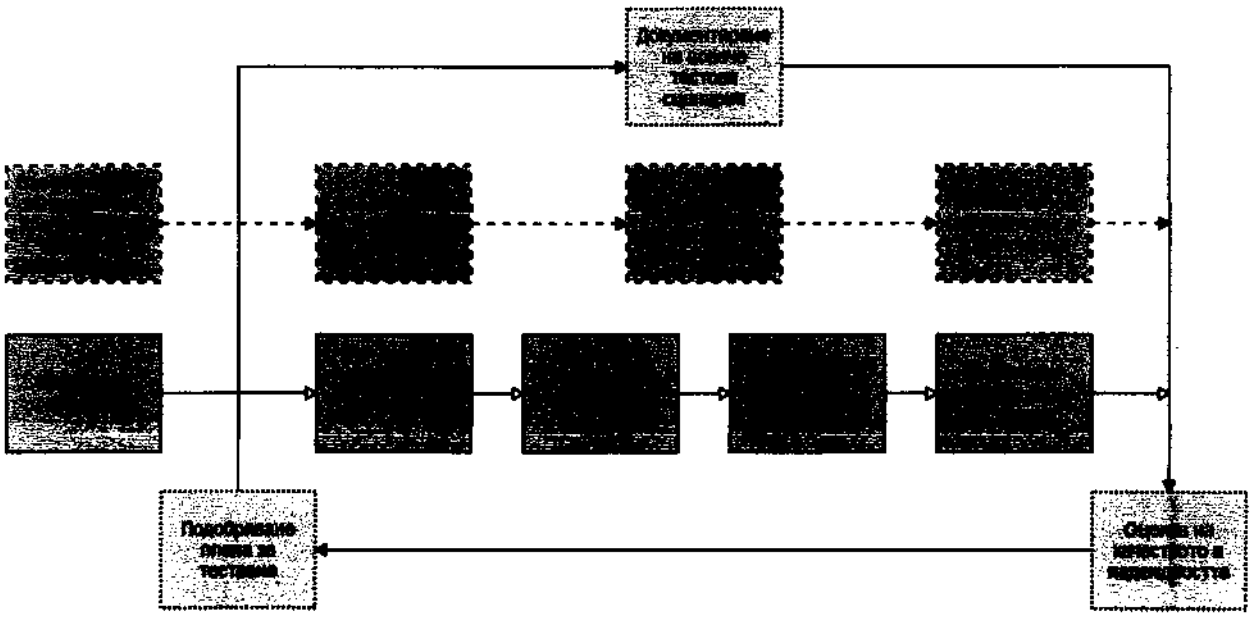
Образецът ще бъде разработен в окончателен вид при планиране на внедряването и ще бъде представен на Възложителя за одобрение като част от Плана за внедряване.

8.1.6. Методология за тестване

Изпълнителя ще проведе тестване на софтуерното решение в създадената тестова среда, с цел да потвърди, че разработеното решение отговаря на функционалните и нефункционалните изисквания. Това се постига чрез осъществяване на следните подцели на тестването:

- Откриване на всички грешки в кода, които екипът трябва отстранява;
- Откриване на грешки при дизайна;
- Откриване на повреди от неочаквано потребителско поведение;
- Тестване на всички елементи на решението.

Разработения софтуер ще се тества обстойно, за да се провери дали покрива изискванията на Възложителя. На следващата диаграма е показан процеса на тестване, който екипът на Изпълнителя ще следва:



фигура 18. Процес на тестване

Процесът на тестване съпътства изпълнението на проекта и се състои от етапите: Планиране, Анализ и проектиране, Реализация и изпълнение, Анализ на резултатите и Заключителни дейности.

8.1.1.6. Примерен тестов план

Настоящият тестов план е предварителен и ще бъде надграден във фазата на анализ.

Етапи на тестване:

1) Планиране

В етапа на Планиране се определят целите, използваните техники и методология за тестване, извършва се планиране и разпределение на ресурсите, изготвяне на график за провеждане на тестването, подготовка и приемане на тест план. Тест планът включва кратко описание на типовете тестване, базирани на анализа на изискванията, описание на различните тестови среди, структура на тест екипа, времевата рамка на тестовите задачи. Изборът на подходяща методология за тестване се базира главно на определяне на основните модули, под модули и компоненти на програмната система и идентифициране на критичните точки за бизнеса и отделните групи потребители на системата.

2) Анализ и проектиране

В етапа на Анализ и проектиране се определя последователността на тестовете и изискванията към тестовата среда, проектиране на тестовете и подготовка на тестови данни (валидни и невалидни).

Проектирането на тестове включва:

- Определяне на групите свойства (features) на програмната система;

- Определяне на основните части и подчасти на програмната система с цел по-лесно проектиране на тестовете чрез разделяне на множества, ориентирани към съставните части;
- Определяне на критичните точки за бизнес процесите, реализирани в програмната система;
- Определяне на типичните ежедневни сценарии за работа на различните групи потребители на програмната система;
- Дефиниране на критичните свойства (critical features), които трябва да бъдат тествани многократно през процеса на разработка;
- Дефиниране на задължителните свойства (required features), които трябва да бъдат тествани на отделни фази през процеса на разработка;
- Дефиниране на допълнителните свойства, подпомагащи процесите в програмната система (additional features), които могат да бъдат тествани в зависимост от времето и ресурсите;
- Определянето на критериите за приемане на програмната система (acceptance criteria).

След изготвянето им те се обсъждат с Възложителя. На базата на договорените критерии за приемане на системата тест екипът подготвя План за провеждане на приемателни тестове на системата. Този план бива съгласуван и одобрен от Възложителя и след това става база за проектиране на приемателните тестове (acceptance testing).

Подходът при проектиране и подготовка на тестови случаи е тясно обвързан с изискванията към системата и се изпълнява в следната последователност:

- Определяне на основните части и подчасти на програмната система – Така се постига по-гъвкаво и ефикасно проследяване на тестовете за функционалното покритие на свойствата на програмната система (т.н. functional coverage).
- Идентифициране на критичните точки за бизнеса и отделните групи потребители на програмната система – По този начин се определят критичните точки за бизнес процесите, реализирани в програмната система (т.н. business critical points), както и типичните ежедневни сценарии за работа на различните групи потребители на програмната система (т.н. everyday business scenario).
- Определяне на групите свойства на програмната система (critical, required additional features) – Това значително подпомага процеса на разработка и тестване.

Избор на подходящи техники за проектиране на тестовете – Тази дейност включва преценка за спецификата на програмната система: дали е публична или критична откъм сигурността; доколко е сложна, комплексна или обикновена; има ли специфика във входните тестови данни; кои от избраните техники за проектиране на тестовете тест екипът владее добре.
Всички изготвени тестови сценарии подлежат на одобрение от оторизиран представител на Възложителя като след това стават база за проектиране на тестовете.

3) Реализация и изпълнение

Изпълнителя ще подготви тестова среда на сървъри предоставени от Възложителя.

Етапът на Реализация включва избор на тестове, генериране на тестови данни, изготвяне на тестовете и реалното им изпълнение. Реализацията на тестването включва:

- Избор на тест;
- Уточняване на основен и алтернативен начин на изпълнение, както и типичните изключения;
- Създаване на тест процедури (валидни комбинации на тестовете с подходящи тестови данни);
- Изпълнение на одобрените тест процедури.

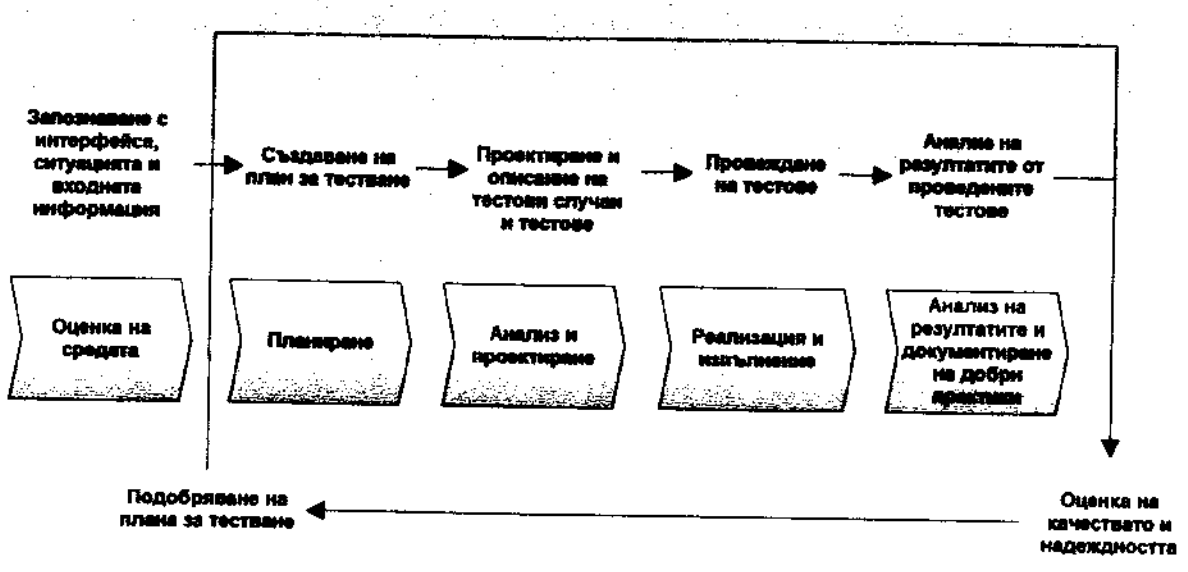
4) Анализ на резултатите

Етапът на Анализ на резултатите се състои от отчитане на получените резултати в избрания формат и проверка на условията за завършване на тестовете.

5) Заключителни дейности

Заключителните дейности обхващат изготвянето на обобщени справки, описание на добрите практики, оценка на проекта с цел подобряване на фирмените тестови процеси, архивиране на материалите.

Обобщените дейности по реализиране на тестовия процес са схематично представени на следната фигура:



фигура 19. Обобщени дейности по реализиране на тестовия процес

Съществуват четири основни понятия, свързани с подготовката на тестовите случай, реализацията на тестването и изграждането на тестовата среда:

Тестов случай (Test case)

Тестовият случай има за цел да тества поведението на определен модул (клас) в дадена ситуация. Обикновено се реализира чрез тестова функция. Всеки тест поставя тествания обект в подходящо за теста състояние, след което следи поведението и

резултатите му в тестваната ситуация. Тестовите случаи извършват същинското тестване и трябва да са независими един от друг и от реда им на изпълнение;

Тестово множество (Test suite)

Тестовото множество се състои от набор от тестови случаи, които са логически свързани. Има за цел да тества поведението на определен модул (клас) в различни ситуации. Показва кои тестове са логически свързани. Реализира се чрез тестов клас, който има достъп до всички данни на тествания. Всеки тест поставя тествания обект в подходящо за теста състояние и следи поведението и резултатите му.

Тестова база (Test fixture)

Тестова база представлява обвивка на тестови множества. Има за цел да създаде необходимите условия за провеждане на тестовете. Отговорен е за инициализиращите и завършващите действия, както и за същинското изпълнение на тестовете. Отново, тестовете трябва да са независими един от друг и от реда им на изпълнение. Реализира се чрез йерархия от тестови класове. В базовите класове се дефинират общите операции, свързани с тестването (инициализация и завършване на теста). Те се предефинират в наследниците (тестови множества), ако е необходимо. Всеки тест поставя тествания обект в подходящо за теста състояние, инициализира го по подходящ начин, провежда теста и разрушава тествания обект. Изключително важно е, тестваната система (компонент) да възстанови първоначалното си състояние след завършване на теста.

Тестова среда (Test harness)

Тестовата среда съдържа тестовата база и предоставя условия за създаване, добавяне и изтриване на тестови множества, както и за провеждане на съответните им тестове. Разполага със средства за контрол на тестването, запазване и анализ на резултатите.

Като добри практики при подготовката на тестови случаи може да посочим:

- Идентифициране на основните функционалности и идентифициране на ключовите функционалности за бизнес процеса;
- Идентифициране на основните части и подчасти на програмната система с цел по-лесно проектиране на тестовете чрез разделяне на множества, ориентирани към отделните модули;
- Идентифициране на основните групи тестове и тестови сценарии за итерация със системата;
- Идентифициране на критичните функционалности (critical features), които трябва да бъдат тествани многократно през процеса на разработка;
- Идентифициране на задължителните функционалности (required features), които трябва да бъдат тествани на отделни фази през процеса на разработка;
- Идентифициране на допълнителни функционалности, подпомагащи процесите в програмната система (additional features), които могат да бъдат тествани в зависимост от времето и ресурсите;
- Документиране на тестовите сценарии;
- Използване на различни графични инструменти за описание на тестовите сценарии

и инструменти за създаване на обвързаност между различните тестови сценарии и обвързаност между тестови сценарии и системна функционалност;

- Подготовка на тестови данни (валидни и невалидни);
- Определяне на условията за провеждане на тестовите сценарии;
- Определя последователността на тестовете и изискванията към тестовата среда, проектиране на тестовете и подготовка на тестови данни
- Определяне на критериите (изискванията към резултата от проведения тест) за преминаване/непреминаване на всеки тест.

Видове тестове залегнали в тестовия план

- Единица тестване – изпълняван се за най-малките тестови софтуерни единици – класове и методи;
- Компонентно тестване – изпълнява се за индивидуални компоненти и модули за да увери, че те коректно реализират бизнес функционалността;
- Системно тестване – след като компонентите и модулите са обединени системата се тества като цяло:
    - Функционално тестване;
    - Тестване на потребителския интерфейс;
    - Тестване на интерфейсите с останалите системи;
    - Тестване на производителността;
    - Тестване на сигурността и контрола на достъпа;
    - Тестване на възстановяемостта на системата след срив;
    - Тестване конфигурацията;
    - Регресивно тестване.

<u>Цели на тестването</u>

- Цели на Единица (Unit) тестването - проверка на единицата (клас или метод) според Дизайн модела и Модела на Имплементацията. Проверка на правилната обработка на въведените данни и получаване на очакваните резултати от всяка единица. Тестват се класовете с ключова функционалност;
- Цели на компонентното тестване - целите на компонентното тестване са изпълними прототипи, като резултата на всяка итерация. Функционалността на компонентите се тества и за тяхното съответствие с бизнес изискванията и стандартите;
- Цели на системното тестване.
    - Функционално тестване - основна цел на Функционалното тестване е Модела на потребителските случаи. Разработват се тестови случаи на основата на този модел като стремежът е да се покрият голямо количество

случаи с различни комбинации от входно- изходни данни. Тези тестови случаи могат да служат и като метрики за напредъка на проекта;

- Тестване на потребителския интерфейс - проверка за наличието на всички полета от формите и проверка за дължината и типовете на данните в тези полета; проверка, дали интерфейса съдържа всички списъци от предефинирани стойности, специфицирани в документацията; проверка на правилната навигация между екраните; проверка за поведението на интерфейса при въвеждане на невалидни и валидни данни; проверка на изгледа на интерфейса;
- Тестване на производителността - проверка времето за отговор; проверка на времето на отговор за оперативните справки; проверка на времето на разпространение на данните;
- Сигурност и тестване контрола на достъп - защита на данните при трансфера им до АМ; защита на данните при разпространението им; защита от неоторизиран достъп да системната функционалност; валидация на различните нива на достъпа на данни; валидиране на журналите;
- Тестване при възстановяване след срив - тестване на времето за възстановяване след срив; проверка на необработените съобщения по време на срив;
- Тестване на комуникационната инфраструктура между модулите и връзката с външните потребители;
- Регресионно тестване - тестване коректното функциониране на системата след промени, когато е създадена нова версия и нови характеристики са прибавени или са поправени съществени дефекти.

Описание

- Единица тестване - най-ниските нива на единиците се тестват първи и тогава се използват за тестване на по-високите нива. Операцията се повтаря, докато не се стигне до най-горните нива.

  За тестване на всяка единица:

Входен критерий	Веднага след имплементацията на ключова единица, тя трябва да бъде тествана от разработчика
Цел на теста	Да се осигури правилна навигация, вход на данните, обработката им и получаване на очакваните резултати
Техника	Изпълнение на всяка функция, използвайки валидни и невалидни данни. Необходимите съобщения за грешки се генерират, при вход на невалидни данни.

Необходими инструменти	Инструмент за автоматизирано тестване Инструмент за проверка на кода Инструмент за следене на грешките
Критерии за завършване	Всички планирани тестове са изпълнени. Откритите дефекти са коригирани.

- Компонентно тестване - тестването на компонентите е подобно на системното функционално тестване, свързано е с реализацията на потребителските случаи;
- Системно тестване

✓ Функционално тестване - базирано е на метода на „черната кутия", въвеждат се данни и се следи изхода:

Входен критерий	След приключване на Модела на потребителските случаи, дефиниране на потребителския графичен интерфейс. Изпълним прототип съществува.
Цел на теста	Да се осигури правилна навигация, вход на данните, обработката им и получаване на очакваните резултати. Да се увери, че цялата функционалност на потребителските случаи е представена и стандартите са спазени.
Техника	Изпълнение на сценарий на потребителски случай с валидни и невалидни данни за да се увери, че: – Очакваните резултати се получават, при вход от валидни данни; – Необходимата грешка или съобщение се визуализират при невалиден вход; – Всяко бизнес правило е правилно приложено; – Всички стандарти са спазени.
Необходими инструменти	Автоматизиран инструмент за тестови скриптове; Инструменти за генериране на данни; Средство за следене на дефектите.
Критерии за завършване	Всички ключови потребителски случаи са тествани; Всички изисквания на приложените стандарти са спазени; Всички критични дефекти са отстранени.

✓ Тестване на потребителския интерфейс:

Входен критерий	Изискванията за потребителския интерфейс са дефинирани и съществува прототип

Цел на теста	Проверка чрез навигация по приложението дали правилно отразява бизнес функциите и изискванията, включително от прозорец към прозорец, от поле към поле. Проверка, че обектите и характеристиките на прозорците, такива като менюта, размери, статус, фокус отговарят на стандартите.
Техника	Създаване на тестове за всеки прозорец за проверка на неговите обекти. Създаване на тестове за проверка поведението на екраните при вход на валидни и невалидни данни.
Необходими инструменти	Автоматизиран инструмент за тестови скриптове; Инструменти за генериране на данни; Средство за следене на дефектите.
Критерии за завършване	Всеки екран отговоря на специфицираните софтуерни стандарти.

- ✓ Тестване на производителността - целта му е да провери изискванията за производителността до каква степен са удовлетворени. Изпълнява се многократно като се променя нивото на натовареност на системата. Изпълняват се след системните тестове във времето, на преди тестовете за конфигурацията и тестовете при срив на системата:

Цел на теста	Валидиране времето за отговор на системата при нормални условия.
Техника	Използват се наборите от тестови случаи разработени при функционалното тестване. Тестовете вървят на една машина и се повтарят за много клиенти.
Необходими инструменти	Автоматизиран инструмент за тестови скриптове; Инструменти за генериране на данни; Средство за следене на дефектите.
Критерии за завършване	Тестовете са изпълняват без грешки и времето за отговор е в рамките на границите зададени в допълнителните изисквания.

- ✓ Стрес тестове:

Цел на теста	Валидиране времето за отговор на системата при условие, че максималния брой потребители извършват действия със системата. Валидиране времето за отговор на системата при условие, че много потребители модифицират едни и същи данни.

Техника	Използват се наборите от тестови случаи разработени при функционалното тестване. Тестовете вървят на една машина и се повтарят за максимален брой клиент.
Необходими инструменти	Автоматизиран инструмент за тестови скриптове; Инструменти за генериране на данни; Средство за следене на дефектите.
Критерии за завършване	Тестовете са изпълняват без грешки и времето за отговор е в рамките на границите зададени в допълнителните изисквания.

- ✓ Сигурност и тестване на контрола за достъп - фокусира се към две основни области на сигурността; сигурност на ниво приложение, включваща рестриктивен достъп до бизнес функциите; сигурност на ниво система, включваща проверка на потребителите:

Входен критерий	Стартира при наличието на прототип с вграден контрол на достъпа
Цел на теста	Сигурност на приложението: Проверка на операциите – Създаване, Четене, Редактиране, Изтриване в зависимост от правата на потребителите и техните роли. Системна сигурност: Достъп само за автентикирани потребители.
Техника	Идентифициране на всеки тип потребител и определяне да видовете разрешени функции. Създаване на тестове за проверка. Промяна на типа на потребителя и повторно изпълнение на тестовете.
Необходими инструменти	Автоматизиран инструмент за тестови скриптове. Инструменти за генериране на данни. Средство за следене на дефектите.
Критерии за завършване	Функционалните тестове преминават без грешка, а при неавтентикиран или неоторизиран опит за достъп се извежда съответното съобщение.

- ✓ Тестване на системата след срив и за възможности за възстановяване:

Входен критерий	След успешно преминаване на системните тестове.

Цел на теста	Проверка процесите на възстановяване – автоматизирани или ръчни и състояние на системата след срив и последващо възстановяване.
Техника	Използва се набора то тестови случаи за функционално тестване, като се пускат след: Спиране на тока на клиентската станция; Спиране на тока на сървъра на приложението; Спиране на тока на сървъра на базата данни; Прекратяване на мрежовата връзка.
Необходими инструменти	Същите инструменти като при Функционално тестване.
Критерии за завършване	След задействане на процедурата по възстановяване системата трябва да е в правилното състояние.

✓ Тестване на конфигурацията:

Входен критерий	След успешно преминаване на системните тестове.
Цел на теста	Валидиране и проверка , че функциите на приложението се изпълняват правилно в средата на конфигурацията.
Техника	Използване на системни скриптове. Отваряне, затваряне на различни приложения по време на тестовете или преди започването им.
Необходими инструменти	Същите инструменти като при Функционално тестване.
Критерии за завършване	За всяка конфигурация тестовете преминават успешно с необходимите резултати.

✓ Регресионно тестване - регресионното тестване не е отделен тип тестване. То е повторение на тестовете, с цел проверка коректната работа на системата след като в нея са въведени промени:

Входен критерий	Нова версия на системата след като в нея са направени промени.
Цел на теста	Проверка поведението на системата след внедряване на промените.
Техника	Използват се вече разработените тестови случаи, набори от случаи, данни и скриптове за повторно тестове на системата. Възможна промяна и на тестовите случаи или данни в зависимост от вида на направените промени.

Необходими инструменти	Същите инструменти като при Функционално тестване.
Критерии за завършване	Същите критерии като изпълняваните тестове.

Контекст за извършване на тестовете

Източниците на информация за провеждане на тестовете:

- План за разработка на софтуерния продукт – включва очакваните дати за провеждане на тестовете, както и резултатите, които следва да се представят от тестването.
- Спецификация на изискванията – включва нефункционалните изисквания към системата.
- Визия – Във Визията са заложени основни нужди и характеристики и продукта
- Модел на дейностите - Моделът на дейностите включва Бизнес модел и Модел на потребителските случаи. Моделът на потребителските случаи представя предвижданите функции и среда на системата и отразява нейните функционални спецификации, а Бизнес моделът описва начина, по който се изпълняват бизнес потребителски случаи.
- Документ Софтуерна архитектура - представя комплексен архитектурен изглед на системата, използвайки за целта редица различни архитектурни разрези, показващи отделни нейни аспекти.
- Дизайн модел – обектен модел, който описва реализацията на потребителските случаи и служи за извеждане на Модела на имплементацията и неговия програмен код.
- Модел на данните – Моделът на данните е подмножество на имплементационния модел, което описва логическия и физически вид на постоянните (персистентни) данни в системата.
- Модел на имплементацията - Моделът на имплементацията събира на едно място компонентите и съдържащите ги имплементационни подсистеми. Компонентите включват както тези, които подлежат на предаване като отчетни резултати (например изпълнимите компоненти), така и тези, от които се извеждат предаваните компоненти (например файлове с програмен код).

8.1.7. Подход за създаване на тестова среда

Подходът за създаване на тестова среда ще взема предвид факта, че настоящият проект изисква по-висока сложност от стандартните проекти за разбаботка по отношение на създаването на средата за тестване. Средата за тестване в рамките на настоящият проект трябва да послужи и като среда в която ще се валидират данните от тестовата миграция на сайтове, регистри и услуги. По тази причина след изготвяне на детайлната

спецификация и проектирането на средствата за миграция ще бъде изготвена детайлна процедура за подготовка на тестова среда.
Подготовката на тестовата среда ще включва като минимум следните стъпки:

1. Инсталиране на СУБД Postgres на определен от Възложителя хардуер – сървър за базата данни
2. Създаване на базата данни на Системата
3. Инсталиране на приложния сървър на определен от Възложителя сървър за приложения – може да се използва Internet information server,в случай че приложния сървър е под операционна система Windows или Kestrel, за Linux/Unix.
4. Инсталиране на приложението на Системата (ASP.NET MVC приложение)
5. Инсталиране на приложението на подсистема – система за управление на съдържанието (ASP.NET MVC приложение)
6. Инсталиране на разработеното средство за миграция (самостоятелно .Net приложение)
7. Настройка на връзката между компонентите и подсистемите
8. Настройка на връзката между средството за миграция и външните системи
9. Извършване на миграцията
10. Зареждане на информация за потребители
11. Инициализиране на мета данните за отделните микросайтове на Системата
12. Провеждаме на автоматизирани интеграционни тестове на цялостната функционалност

8.1.8. Методика за изпълнение на дейностите по обучение

Внедряването на системата изисква да бъдат инструктирани ползвателите участващи в нейното практическо приложение, чрез проевждане на обучение. Освен обучението в задача 2.1, което се отняса за персонала на 5- те пилотни центъра, изпълнителят ще проведе по настоящата задача, последващи полудневни обучения на място в ПУДООС и в 5-те общински администрации – Шумен, Разград, Левски, Съединение и Созопол, на които да могат да присъстват и преките ползватели от останалите 17 общини.

Като минимум обучението включва представяне на целите, задачите, структурата и функционалностите на системата, начин за работа с нея, въвеждане на информация, видове доклади и справки, които тя може да генерира и т.н. Целта е да се създаде необходимия капацитет за работа със системата на всички въвлечени страни и за правилното функциониране на системата.

Изпълнителят ще:

- изготви концепция на обучението, план за действие, учебна програма и съдържание на обучението, обучителни материали в PPT презентация/и и др.

подходящи формати, да размножи комплекти с обучителни материали за всеки участник;
- осигури обучители;
- осигури логистиката за цялостното организиране и провеждане на обученията.

За провеждането на обученията „Смарт Системс 2010" ЕООД ще осигури за своя сметка:

- Необходимия хардуер;
- Необходимия софтуер;
- Зала за провеждане на обученията;
- Учебни материали;
- Лектори;
- Разходите за предоставяне на полудневно обучение на място в 5-те общински администрации, в общините където се намират 5-те пилотни центъра за събиране и съхранение на опасни битови отпадъци, както и в ПУДООС (общо 6 полудневни обучения на място, както и тези за текущо дистанционно предоставяне на консултации и отстраняване на място на евентуални проблеми с функционирето на системата в гаранционния период;

„Смарт Системс 2010" ЕООД ще организира и да проведе обучения за следните групи и ползватели на софтуерното решение, в рамките на Задача 2.1 и Задача 3.4 от Техническото задание:

- **Потребителска група 1** – потребители от 22 общини - Шумен, Разград, Левски, Съединение и Созопол и 17 по-малки общини – Велики Преслав, Смядово, Каспичан, Хитрино, Лозница, Самуил, Исперих, Завет, Цар Калоян, Пордим, Никопол, Белене, Марица, Калояново, Хисаря, Приморско и Царево – Задача 3.4;
- **Потребителска група 2** - потребители в 5 (пет) пилотни общински центрове/площадки за събиране на опасни битови отпадъци, на територията на Шумен, Разград, Левски, Съединение и Созопол - Задача 2.1.
- **Потребителска група 3** - длъжностни лица в ПУДООС – Задача 3.4

Тъй като системата ще има специализирани модули за групи потребители и доставчици на информация за потребителите ще бъдат организирани специални обучения за ползване на определени функционалности, съгласно определените роли.

„Смарт Системс 2010" ЕООД ще разработи график на обученията на място в 5-те общински администрации и в ПУДООС, ще съгласува датите и ще подготви обученията, като отчита възможностите и ангажиментите на Възложителя за участие на определените служители. За целта, „Смарт Системс 2010" ЕООД ще отправи предложение до Възложителя, към което ще представи и план за обучение. Планът следва да бъдат одобрен от Възложителя.

Изпълнителят ще разработи обучителна програма и необходимите обучителни материали, които ще размножи и предостави на обучаемите. Изпълнителят поема за своя сметка и отговаря за цялостната логистика по организиране и провеждане на обученията, Изпълнителят поема и отговаря изцяло за съдържателната страна на обученията (напр. обезпечаване на обучители, програма, материали, обратна връзка и анкетна карта от участниците, др.).

Изпълнителят ще организира и проведе обученията по Задача 2.1 от Техническото задание, за персонала на 5-те общински пилотни центъра за събиране и съхранение на опасни битови отпадъци, където ще се тества създадената софтуерна система.

Учебни материали

Учебните материали ще са на български език и ще съдържат материали по предварително разработена програма за обучение.

Предоставените на хартиен носител и/или в електронна форма учебни материали ще отговарят на изискванията за публичност и информация.

Присъствен списък

По време на учебните занимания на обучаваните ще се предоставят предварително подготвените регистрационни форми, в които те ще могат да попълнят своите данни и ежедневно да отбелязват присъствието си. При провежданото обучение ще се използва семинарна форма на обучение в комбинация с практически занятия.

Присъствен списък
на участниците в .....
Обучител:.....

№	**Име, Фамилия**	**Дирекция**	**Длъжност**	**e-mail**	**Период на обучението .....**	**Получени материали /подпис/**
1.						
2.						
3.						
4.						
5.						
6.						
7.						
8.						
9.						

10.						

Анкетни карти

В края на обучението участниците ще попълват анкетни карти, които „Смарт Системс 2010“ ЕООД ще обобщи в доклад. Оценка на проведеното обучение може да се извърши, чрез анкетиране на обучаемите. На оценка ще подлежат: съдържанието и формата на обучението, придобитите знания и умения, системата на обучение, стила на преподаване, подхода на лектора, процеса и структурата на обучението, приложимостта на наученото в практиката. За целта се определят показатели, измерващи ефективността от проведените обучения (например степен на удовлетвореност на участниците в обучението, повишаване на качеството на работата на участниците след посещение на обучението, намаляване на допусканите грешки, и др.) „Смарт Системс 2010“ ЕООД ще да приложи съвкупност от различни методи за оценка.

Имена:.....
Институция:.....
Позиция:.....

1. По мое мнение обучението беше:

a) много успешно ☐

b) добро ☐

c) незадоволително ☐

2. **По мое мнение продължителността на обучението беше:**

a) много дълга ☐

b) много кратка ☐

c) нормална ☐

3. **С оглед бъдеща дейност, свързана със системата, обучението беше:**

a) ненужно ☐

b) необходимо ☐

c) много добро ☐

4. **Според мен допълнително следва да се включи в програмата на обучението /моля да напишете Вашите предложения/:**